Michael Affenzeller · Stephan M. Winkler ·
Anna V. Kononova · Heike Trautmann ·
Tea Tušar · Penousal Machado ·
Thomas Bäck (Eds.)

LNCS 15151

# Parallel Problem Solving from Nature – PPSN XVIII

18th International Conference, PPSN 2024
Hagenberg, Austria, September 14–18, 2024
Proceedings, Part IV


Springer

MOREMEDIA

**Lecture Notes in Computer Science** **15151**

The series Lecture Notes in Computer Science (LNCS), including its subseries Lecture Notes in Artificial Intelligence (LNAI) and Lecture Notes in Bioinformatics (LNBI), has established itself as a medium for the publication of new developments in computer science and information technology research, teaching, and education.

LNCS enjoys close cooperation with the computer science R & D community, the series counts many renowned academics among its volume editors and paper authors, and collaborates with prestigious societies. Its mission is to serve this international community by providing an invaluable service, mainly focused on the publication of conference and workshop proceedings and postproceedings. LNCS commenced publication in 1973.

Michael Affenzeller · Stephan M. Winkler ·
Anna V. Kononova · Heike Trautmann ·
Tea Tušar · Penousal Machado · Thomas Bäck
Editors

# Parallel Problem Solving from Nature – PPSN XVIII

18th International Conference, PPSN 2024
Hagenberg, Austria, September 14–18, 2024
Proceedings, Part IV

Springer

*Editors*
Michael Affenzeller
University of Applied Sciences Upper Austria
Wels, Austria

Anna V. Kononova
Leiden University
Leiden, The Netherlands

Tea Tušar
Jožef Stefan Institute
Ljubljana, Slovenia

Thomas Bäck
Leiden University
Leiden, The Netherlands

Stephan M. Winkler
University of Applied Sciences Upper Austria
Hagenberg, Austria

Heike Trautmann
University of Paderborn
Paderborn, Germany

Penousal Machado
University of Coimbra
Coimbra, Portugal

ISSN 0302-9743 ISSN 1611-3349 (electronic)
Lecture Notes in Computer Science
ISBN 978-3-031-70084-2 ISBN 978-3-031-70085-9 (eBook)
https://doi.org/10.1007/978-3-031-70085-9



This Springer imprint is published by the registered company Springer Nature Switzerland AG
The registered company address is: Gewerbestrasse 11, 6330 Cham, Switzerland

# Preface

Two years ago, in 2022, the international conference on Parallel Problem Solving from Nature (PPSN) returned to where it all started in 1990, namely to Dortmund, Germany. It was great to see that the community had overcome the pandemic and gathered with more than 100 participants attending in person.

On the last day of the conference, during the closing ceremony, we got the chance to propose the University of Applied Sciences Upper Austria (FH OÖ) as organizers and the Softwarepark Hagenberg as the location for PPSN 2024. We were convinced that FH OÖ as the (with respect to research and development) strongest university of applied sciences in Austria could be the ideal choice as host for PPSN 2024, especially as we presented the research group Heuristic and Evolutionary Algorithms Laboratory (HEAL), one of the most active groups in evolutionary algorithms in Austria, as the core group of the organization team. After some weeks, we were delighted to hear from the steering committee that we were chosen as organizers and Hagenberg as the location for this year's edition of PPSN.

We are pleased that a record number of researchers followed our call by submitting their papers for review. We received 294 submissions from which the program chairs selected the top 101 after an extensive peer-review process, which corresponds to an acceptance rate of 34.35%. Not all decisions were easy to make, but we benefited greatly from the careful reviews provided by the international program committee. With an average of 2.86 reviews per paper, most of the submissions received three reviews, while some received two. This led to a total of 840 reviews. Thanks to these reviews, we were able to decide about acceptance on a solid basis.

The papers included in these proceedings were assigned to 12 clusters, entitled *Combinatorial Optimization, Genetic Programming, Fitness Landscape Modeling and Analysis, Benchmarking and Performance Measures, Automated Algorithm Selection and Configuration, Numerical Optimization, Bayesian- and Surrogate-Assisted Optimization, Theoretical Aspects of Nature-Inspired Optimization, (Evolutionary) Machine Learning and Neuroevolution, Evolvable Hardware and Evolutionary Robotics, Multi-objective Optimization and Real-World Applications* which can hardly reflect, the true variety of research topics presented in the proceedings at hand. Following the tradition and spirit of PPSN, all papers were presented as posters. The eight poster sessions consisting of 12 or 13 papers each were compiled orthogonally to the clusters mentioned above to cover the range of topics as widely as possible. As a consequence, participants with different interests would find some relevant papers in every session and poster presenters were able to discuss related work in sessions different from their own.

As usual, the conference started with two days of workshops and tutorials (Saturday and Sunday), followed by three days of poster sessions and invited plenary talks (Monday to Wednesday). We are delighted that three highly renowned researchers from up-and-coming, related research fields accepted our invitation to give a keynote speech, which was be the first item on the program over the three days of the conference.

Two of our keynote speakers are young professors at excellent academic institutions, namely Oliver Schütze (Cinvestav-IPN, Mexico City) and Richard Küng (JKU Linz, Austria); the third keynoter is a researcher at Google Deepmind, namely Bernardino Romera-Paredes, with an equally impressive scientific record.

Needless to say, the success of such a conference depends on authors, reviewers, and organizers. We are grateful to all authors for submitting their best and latest work, to all the reviewers for the generous way they spent their time and provided their valuable expertise in preparing these reviews, to the workshop organizers and tutorial presenters for their contributions to enhancing the value of the conference, and to the local organizers who helped to make PPSN XVIII happen.

Last but not least, we would like to thank *Softwarepark Hagenberg* and the *University of Applied Sciences Upper Austria* for the donations. We are grateful for the long-standing support of *Springer* to this conference series. Finally, we thank the *RISC Software and Software Competence Center Hagenberg* for providing financial backing.

July 2024

Michael Affenzeller
Stephan M. Winkler
Anna V. Kononova
Heike Trautmann
Tea Tušar
Penousal Machado
Thomas Bäck

# Organization

## General Chairs

| | |
|---|---|
| Michael Affenzeller | University of Applied Sciences Upper Austria, Austria |
| Stephan Winkler | University of Applied Sciences Upper Austria, Austria |

## Honorary Chair

| | |
|---|---|
| Hans-Paul Schwefel | TU Dortmund, Germany |

## Program Committee Chairs

| | |
|---|---|
| Heike Trautmann | University of Paderborn, Germany |
| Tea Tušar | Jožef Stefan Institute, Slovenia |
| Penousal Machado | University of Coimbra, Portugal |
| Thomas Bäck | Leiden University, Netherlands |

## Proceedings Chair

| | |
|---|---|
| Anna V. Kononova | Leiden University, Netherlands |

## Tutorials Chair

| | |
|---|---|
| Fabricio Olivetti de França | Federal University of ABC, Brazil |

## Workshops Chair

| | |
|---|---|
| Roman Kalkreuth | RWTH Aachen University, Germany |

## Publicity Chairs

| | |
|---|---|
| Jan Zenisek | University of Applied Sciences Upper Austria, Austria |
| Christian Haider | University of Applied Sciences Upper Austria, Austria |
| Louise Buur | University of Applied Sciences Upper Austria, Austria |

## Technical Support Chairs

| | |
|---|---|
| Oliver Krauss | University of Applied Sciences Upper Austria, Austria |
| Du Nguyen Duy | Software Competence Center Hagenberg, Austria |

## Steering Committee

| | |
|---|---|
| Thomas Bäck | Leiden University, Netherlands |
| David W. Corne | Heriot-Watt University, UK |
| Carlos Cotta | University of Malaga, Spain |
| Kenneth De Jong | George Mason University, USA |
| Gusz E. Eiben | Vrije Universiteit Amsterdam, Netherlands |
| Bogdan Filipič | Jožef Stefan Institute, Slovenia |
| Emma Hart Edinburgh | Napier University, UK |
| Juan Julián Merelo Guervós | University of Granada, Spain |
| Günter Rudolph | TU Dortmund, Germany |
| Thomas P. Runarsson | University of Iceland, Iceland |
| Robert Schaefer | University of Krakow, Poland |
| Marc Schoenauer | Inria, France |
| Xin Yao | University of Birmingham, UK and SUSTech, China |

## Keynote Speakers

| | |
|---|---|
| Oliver Schütze | CINVESTAV-IPN, Mexico |
| Bernardino Romera-Paredes | Google DeepMind London, UK |
| Richard Küng | Johannes Kepler University Linz, Austria |

## Program Committee

| | |
|---|---|
| Michael Affenzeller | University of Applied Sciences Upper Austria, Austria |
| Hernán Aguirre | Shinshu University, Japan |
| Imène Ait Abderrahim | University of Djilali Bounaama Khemis Miliana, Algeria |
| Youhei Akimoto | University of Tsukuba, Japan |
| Richard Allmendinger | University of Manchester, UK |
| Marie Anastacio | Leiden University, Netherlands |
| Claus Aranha | University of Tsukuba, Japan |
| Dirk Arnold | Dalhousie University, Canada |
| Anne Auger | Inria, France |
| Dogan Aydin | Dumlupinar University, Turkey |
| Jaume Bacardit | Newcastle University, UK |
| Heder Bernardino | Federal University of Juiz de Fora, Brazil |
| Hans-Georg Beyer | Vorarlberg University of Applied Sciences, Austria |
| Martin Binder | Ludwig Maximilian University of Munich, Germany |
| Mauro Birattari | Université libre de Bruxelles, Belgium |
| Bernd Bischl | Ludwig Maximilian University of Munich, Germany |
| Julian Blank | Michigan State University, USA |
| Aymeric Blot | University College London, UK |
| Peter Bosman | Centrum Wiskunde & Informatica, Netherlands |
| Jakob Bossek | University of Paderborn, Germany |
| Anton Bouter | Centrum Wiskunde & Informatica, Netherlands |
| Jürgen Branke | University of Warwick, UK |
| Dimo Brockhoff | Inria, France |
| Alexander Brownlee | University of Stirling, UK |
| Larry Bull | University of the West of England, UK |
| Maxim Buzdalov | Aberystwyth University, UK |
| Stefano Cagnoni | University of Parma, Italy |
| Salvatore Calderaro | Palermo University, Italy |
| Pedro Carvalho | University of Aveiro, Portugal |
| Josu Ceberio | University of the Basque Country, Spain |
| Ying-Ping Chen | National Chiao Tung University, Taiwan |
| Francisco Chicano | University of Malaga, Spain |
| Miroslav Chlebik | University of Sussex, UK |
| Sung-Bae Cho | Yonsei University, South Korea |
| Carlos Coello Coello | CINVESTAV-IPN, Mexico |

| | |
|---|---|
| Jordan Cork | Jožef Stefan Institute, Slovenia |
| João Correia | University of Coimbra, Portugal |
| Gabriel Cortês | University of Coimbra, Portugal |
| Doğan Çörüş | Kadir Has University, Turkey |
| Ernesto Costa | University of Coimbra, Portugal |
| Carlos Cotta | University of Malaga, Spain |
| António Cunha | University of Minho, Portugal |
| Nguyen Dang | St Andrews University, UK |
| Kenneth De Jong | George Mason University, USA |
| Roy de Winter | Leiden University, Netherlands |
| Kalyanmoy Deb | Michigan State University, USA |
| Antonio Della Cioppa | University of Salerno, Italy |
| Antipov Denis | University of Adelaide, Australia |
| Bilel Derbel | Université de Lille, France |
| André Deutz | Leiden University, Netherlands |
| Konstantin Dietrich | TU Dresden, Germany |
| Benjamin Doerr | Ecole Polytechnique, France |
| Carola Doerr | Sorbonne University, France |
| John Drake | University of Leicester, UK |
| Rafał Dreżewski | AGH University of Science and Technology, Poland |
| Johann Dreo | Pasteur Institute, France |
| Paul Dufossé | ID Solutions Oncology, France |
| Tome Eftimov | Jožef Stefan Institute, Slovenia |
| Theresa Eimer | Leibniz University Hannover, Germany |
| Michael Emmerich | Leiden University, Netherlands |
| Andries Engelbrecht | University of Stellenbosch, South Africa |
| Anton Eremeev | Dostoevsky Omsk State University, Russia |
| Richard Everson | University of Exeter, UK |
| Pedro Ferreira | University of Lisbon, Portugal |
| Antonino Fiannaca | Italian National Research Council, Italy |
| Jonathan Fieldsend | University of Exeter, UK |
| Bogdan Filipič | Jožef Stefan Institute, Slovenia |
| Steffen Finck | Vorarlberg University of Applied Sciences, Austria |
| Marcus Gallagher | University of Queensland, Australia |
| José García-Nieto | University of Málaga, Spain |
| Mario Giacobini | University of Torino, Italy |
| Kyriakos Giannakoglou | National Technical University of Athens, Greece |
| Tobias Glasmachers | Ruhr-Universität Bochum, Germany |
| Christian Grimme | University of Münster, Germany |

| | |
|---|---|
| Alexander Hagg | Bonn-Rhein-Sieg University of Applied Sciences, Germany |
| Julia Handl | University of Manchester, UK |
| Nikolaus Hansen | Inria, France |
| Jin-Kao Hao | University of Angers, France |
| Hans Harder | Paderborn University, Germany |
| Emma Hart | Edinburgh Napier University, UK |
| Verena Heidrich-Meisner | CAU Kiel, Germany |
| Jonathan Heins | TU Dresden, Germany |
| Carlos Henggeler Antunes | University of Coimbra, Portugal |
| Carlos Ignacio Hernández Castellanos | National Autonomous University of Mexico, Mexico |
| Ishara Hewa Pathiranage | University of Adelaide, Australia |
| Martin Holeňa | Czech Academy of Sciences, Czechia |
| Andoni Irazusta Garrnendia | University of the Basque Country, Spain |
| Hisao Ishibuchi | Southern University of Science and Technology, China |
| Christian Jacob | University of Calgary, Canada |
| Domagoj Jakobović | University of Zagreb, Croatia |
| Anja Jankovic | RWTH Aachen University, Germany |
| Thomas Jansen | Aberystwyth University, UK |
| Laetitia Jourdan | Université de Lille, CRIStAL, CNRS, France |
| Bryant Julstrom | St. Cloud State University, USA |
| Timo Kötzing | Hasso Plattner Institute, Germany |
| Roman Kalkreuth | RWTH Aachen University, Germany |
| George Karakostas | McMaster University, Canada |
| Florian Karl | Ludwig Maximilian University of Munich, Germany |
| Ed Keedwell | University of Exeter, UK |
| Pascal Kerschke | TU Dresden, Germany |
| Marie-Eléonore Kessaci | University of Lille, France |
| Ahmed Kheiri | Lancaster University, UK |
| Wolfgang Konen | TH Cologne, Germany |
| Lars Kotthoff | University of Wyoming, USA |
| Oswin Krause | University of Copenhagen, Denmark |
| Krzysztof Krawiec | Poznan University of Technology, Poland |
| Martin S. Krejca | Ecole Polytechnique, France |
| William B. Langdon | University College London, UK |
| Manuel López-Ibáñez | University of Manchester, UK |
| William La Cava | Boston Children's Hospital, USA |
| Algirdas Lancinskas | Vilnius University, Lithuania |
| Yuri Lavinas | University of Toulouse, France |

| | |
|---|---|
| Per Kristian Lehre | University of Birmingham, UK |
| Johannes Lengler | ETH Zurich, Switzerland |
| Markus Leyser | TU Dresden, Germany |
| Ke Li | University of Exeter, UK |
| Arnaud Liefooghe | University of Lille, France |
| Giosuè Lo Bosco | University of Palermo, Italy |
| Fernando Lobo | University of Algarve, Portugal |
| Nuno Lourenço | University of Coimbra, Portugal |
| Jose A. Lozano | University of the Basque Country, Spain |
| Rodica Lung | Babes-Bolyai University, Romania |
| Chuan Luo | Peking University, China |
| Evelyne Lutton | INRAE, France |
| Jessica Mégane | University of Coimbra, Portugal |
| João Macedo | University of Coimbra, Portugal |
| Mikel Malagón | University of the Basque Country, Spain |
| Katherine Malan | University of South Africa, South Africa |
| Vittorio Maniezzo | University of Bologna, Italy |
| Valentin Margraf | Ludwig Maximilian University of Munich, Germany |
| Luis Martí | Center Inria Chile, Chile |
| Jörn Mehnen | University of Strathclyde, UK |
| Marjan Mernik | University of Maribor, Slovenia |
| Olaf Mersmann | Federal University of Applied Administrative Sciences, Germany |
| Silja Meyer-Nieberg | Bundeswehr University Munich, Germany |
| Efrén Mezura-Montes | University of Veracruz, Mexico |
| Krzysztof Michalak | Wroclaw University of Economics, Poland |
| Kaisa Miettinen | University of Jyväskylä, Finland |
| Edmondo Minisci | University of Strathclyde, UK |
| Gara Miranda Valladares | University of La Laguna, Spain |
| Mustafa Misir | Duke Kunshan University, China |
| Marco Montes de Oca | EnFi Inc. and Northeastern University, USA |
| Hugo Monzón | RIKEN, Japan |
| Mario Andrés Muñoz | University of Melbourne, Australia |
| Boris Naujoks | TH Cologne, Germany |
| Antonio J. Nebro | University of Málaga, Spain |
| Ferrante Neri | University of Surrey, UK |
| Aneta Neumann | University of Adelaide, Australia |
| Frank Neumann | University of Adelaide, Australia |
| Michael O'Neill | University College Dublin, Ireland |
| Gabriela Ochoa | University of Stirling, UK |
| Pietro S. Oliveto | University of Sheffield, UK |

| | |
|---|---|
| Una-May O'Reilly | Massachusetts Institute of Technology, USA |
| José Carlos Ortiz-Bayliss | Monterrey Institute of Technology and Higher Education, Mexico |
| Patryk Orzechowski | University of Pennsylvania, USA |
| Ender Özcan | University of Nottingham, UK |
| Ben Paechter | Edinburgh Napier University, UK |
| Gregor Papa | Jožef Stefan Institute, Slovenia |
| Luís Paquete | University of Coimbra, Portugal |
| Andrew J. Parkes | University of Nottingham, UK |
| Sebastian Peitz | Paderborn University, Germany |
| Kokila Kasuni | Perera University of Adelaide, Australia |
| Stjepan Picek | Radboud University, Netherlands |
| Martin Pilát | Charles University, Czechia |
| Nelishia Pillay | University of Pretoria, South Africa |
| Petr Pošík | Czech Technical University in Prague, Czechia |
| Raphael Patrick Prager | University of Münster, Germany |
| Oliver Preuß | Paderborn University, Germany |
| Mike Preuss | Leiden University, Netherlands |
| Michal Przewozniczek | Wroclaw University of Science and Technology, Poland |
| Chao Qian | Nanjing University, China |
| Günther Raidl | Vienna University of Technology, Austria |
| Elena Raponi | Leiden University, Netherlands |
| Khaled Rasheed | University of Georgia, USA |
| Alma Rahat | Swansea University, UK |
| Piotr Ratuszniak Koszalin | University of Technology, Poland |
| Tapabrata Ray | University of New South Wales, Australia |
| Quentin Renau | Edinburgh Napier University, UK |
| Riccardo Rizzo | Harvard University, USA |
| Angel Rodriguez-Fernandez | CINVESTAV-IPN, Mexico |
| Eduardo Rodriguez-Tello | CINVESTAV-IPN, Mexico |
| Andrea Roli | University of Bologna, Italy |
| Jeroen Rook | University of Twente, Netherlands |
| Jonathan Rowe | University of Birmingham, UK |
| Günter Rudolph | TU Dortmund, Germany |
| Conor Ryan | University of Limerick, Ireland |
| Saba Sadeghi Ahouei | University of Adelaide, Australia |
| Daniela Santos | Lutheran University of Brazil, Brazil |
| Frédéric Saubion | University of Angers, France |
| Lennart Schäpermeier | TU Dresden, Germany |
| Robert Schaefer | AGH University of Science and Technology, Poland |

| | |
|---|---|
| Andrea Schaerf | University of Udine, Italy |
| Larissa Schmid | Karlsruhe Institute of Technology, Germany |
| Lennart Schneider | Ludwig Maximilian University of Munich, Germany |
| Marc Schoenauer | Inria, France |
| Renzo Scholman | Centrum Wiskunde & Informatica, Netherlands |
| Oliver Schuetze | CINVESTAV-IPN, Mexico |
| Moritz Seiler | Paderborn University, Germany |
| Bernhard Sendhoff | Honda Research Institute Europe, Germany |
| Roman Senkerik | Tomas Bata University, Czechia |
| Marc Sevaux | University of South Brittany, France |
| Hadar Shavit | RWTH Aachen University, Germany |
| Ofer Shir | Tel-Hai College, Israel |
| Shinichi Shirakawa | Yokohama National University, Japan |
| Moshe Sipper | Ben-Gurion University of the Negev, Israel |
| Jim Smith | University of the West of England, UK |
| Konstantin Sonntag | Paderborn University, Germany |
| Giovanni Squillero | Politecnico di Torino, Italy |
| Sebastian Stich | CISPA Helmholtz Center for Information Security, Germany |
| Catalin Stoean | University of Craiova, Romania |
| Thomas Stützle | Université libre de Bruxelles, Belgium |
| Mihai Suciu | Babes-Bolyai University, Romania |
| Dirk Sudholt | University of Sheffield, UK |
| Andrew Sutton | University of Minnesota, USA |
| Urban Škvorc | Paderborn University, Germany |
| Ricardo Takahashi | Federal University of Minas Gerais, Brazil |
| Sara Tari | University of the Littoral Opal Coast, France |
| Daniel Tauritz | Auburn University, USA |
| Dirk Thierens | Utrecht University, Netherlands |
| Kevin Tierney | Bielefeld University, Germany |
| Renato Tinós | University of São Paulo, Brazil |
| Marco Tomassini | University of Lausanne, Switzerland |
| Alberto Tonda | INRAE, France |
| Jamal Toutouh | Massachusetts Institute of Technology, USA |
| Kento Uchida | Yokohama National University, Japan |
| Ryan J. Urbanowicz | University of Pennsylvania, USA |
| Niki van Stein | Leiden University, Netherlands |
| Nadarajen Veerapen | University of Lille, France |
| Filippo Vella | National Research Council, Italy |
| Sébastien Verel | University of the Littoral Opal Coast, France |
| Diederick Vermetten | Leiden University, Netherlands |

| | |
|---|---|
| Anh Viet Do | University of Adelaide, Australia |
| Adriano Vinhas | University of Coimbra, Portugal |
| Markus Wagner | University of Adelaide, Australia |
| Hanyang Wang | Huawei Technologies, UK |
| Hao Wang | Leiden University, Netherlands |
| Elizabeth Wanner | CEFET, Brazil |
| Tobias Weber | Otto von Guericke University Magdeburg, Germany |
| Thomas Weise | Hefei University, China |
| Marcel Wever | Ludwig Maximilian University of Munich, Germany |
| Darrell Whitley | Colorado State University, USA |
| Dennis Wilson | University of Toulouse, France |
| Carsten Witt | Technical University of Denmark, Denmark |
| Man Leung Wong | Lingnan University, Hong Kong, China |
| Kaifeng Yang | University of Applied Sciences Upper Austria, Austria |
| Shengxiang Yang | De Montfort University, UK |
| Furong Ye | Leiden University, Netherlands |
| Martin Zaefferer | DHBW Ravensburg, Germany |
| Aleš Zamuda | University of Maribor, Slovenia |
| Saúl Zapotecas-Martínez | INAOE, Mexico |
| Christine Zarges | Aberystwyth University, UK |
| Mengjie Zhang | Victoria University of Wellington, New Zealand |

# Contents – Part IV

## Multi-objective Optimization

# Multi-objective Optimization

# Selection Strategy Based on Proper Pareto Optimality in Evolutionary Multi-objective Optimization

Kai Li, Kangnian Lin, Ruihao Zheng, and Zhenkun Wang(✉)

School of System Design and Intelligent Manufacturing,
Southern University of Science and Technology, Shenzhen 518055, China
{12132672,12132686}@mail.sustech.edu.cn, wangzhenkun90@gmail.com

**Abstract.** On the multi-objective optimization problems (MOP), the dominance-resistant solution (DRS) refers to the solution that has inferior objective values but is difficult to dominate by other solutions. Prior studies have affirmed that DRSs are prevalent across MOPs and difficult to eliminate, leading to substantial performance deterioration in many multi-objective evolutionary algorithms (MOEAs). In this paper, we propose a metric inspired by proper Pareto optimality and then develop a selection strategy based on this metric (SPP) to mitigate the negative impact of DRSs. Furthermore, we implement SPP on multi-objective evolutionary algorithm based on decomposition (MOEA/D) and call the new algorithm MOEA/D-SPP. Specifically, the algorithm employs the penalty-based boundary intersection method to scalarize the MOP. Subsequently, SPP is integrated into the environmental selection. The strategy measures and sorts a set of solutions such that DRSs can be identified and removed. Finally, weight vectors are adjusted, thereby enhancing the population diversity. In experimental studies, MOEA/D-SPP outperforms five state-of-the-art MOEAs on DRS-MOPs, demonstrating the promising application of SPP.

**Keywords:** Multi-objective optimization · Dominance-resistant solution · Evolutionary algorithm · Proper Pareto optimality

## 1 Introduction

Numerous real-world challenges, such as logistics dispatch and printed-circuit board assembly, necessitate the simultaneous consideration of multiple conflicting objectives [11,27,43]. These optimization problems are called multi-objective optimization problems (MOPs), which have the general form as follows:

$$\begin{aligned} \min. \quad & \mathbf{f}(\mathbf{x}) = (f_1(\mathbf{x}), \ldots, f_m(\mathbf{x}))^\intercal, \\ \text{s.t.} \quad & \mathbf{x} \in \Omega, \end{aligned} \tag{1}$$

where $\mathbf{x} = (x_1, \ldots, x_n)^\intercal$ represents the decision vector, also known as the solution, and $\Omega \subset \mathbb{R}^n$ signifies the feasible region. The mapping $\mathbf{f} : \mathbb{R}^n \to \mathbb{R}^m$


M. Affenzeller et al. (Eds.): PPSN 2024, LNCS 15151, pp. 3–19, 2024.
https://doi.org/10.1007/978-3-031-70085-9_1

encompasses $m$ objective functions, and $\mathbf{f}(\mathbf{x})$ is the objective vector corresponding to $\mathbf{x}$. Some fundamental concepts used in this paper are established below.

**Definition 1.** *Given two vectors* $\mathbf{u}, \mathbf{v} \in \mathbb{R}^m$, $\mathbf{u}$ *is said to* **dominate** $\mathbf{v}$, *iff* $u_i \leq v_i$ *for every* $i \in \{1, \ldots, m\}$ *and* $u_j < v_j$ *for at least one* $j \in \{1, \ldots, m\}$.

**Definition 2.** *A decision vector* $\mathbf{x}^*$ *and the corresponding objective vector* $\mathbf{f}(\mathbf{x}^*)$ *are* **Pareto-optimal**, *if there is no* $\mathbf{x} \in \Omega$ *such that* $\mathbf{f}(\mathbf{x})$ *dominates* $\mathbf{f}(\mathbf{x}^*)$.

**Definition 3.** *The set of all Pareto-optimal solutions is called the* **Pareto set** *(denoted as PS), and its image in the objective space is called the* **Pareto front** *(denoted as PF).*

**Definition 4 (From [13]).** *A decision vector* $\mathbf{x} \in \Omega$ *is* **properly Pareto-optimal** *if it is Pareto-optimal and if there is some real number* $M > 0$ *such that for each* $f_i$ *and each* $\mathbf{x}'$ *satisfying* $f_i(\mathbf{x}') < f_i(\mathbf{x})$, *there exists at least one* $f_j$ *such that* $f_j(\mathbf{x}) < f_j(\mathbf{x}')$ *and*

$$\frac{f_i(\mathbf{x}) - f_i(\mathbf{x}')}{f_j(\mathbf{x}') - f_j(\mathbf{x})} \leq M. \tag{2}$$

Many investigations recognize the prevalence of dominance-resistant solutions (DRSs) within the feasible region of MOPs, exerting a considerable impact on the performance of various multi-objective evolutionary algorithms (MOEAs) [2,42,44]. The corresponding test problems denoted as DRS-MOPs are designed, such as mDTLZ [35] and MOP-CH [34]. A DRS has satisfactory values in at least one objective but exhibits significant deficiencies in others [18]. In other words, DRSs are close to the boundaries of the feasible objective region and remain distant from the PF. They are apart from Pareto-optimal, while few solutions can dominate them. Therefore, DRSs often persist as non-dominated solutions within the evolutionary population. For example, in Fig. 1, point B is located on the boundary parallel to the $f_3$ axis. Notably, B is markedly inferior to the objective vector on the PF, yet it is dominated solely by the objective vectors on line AB (i.e., red line in Fig. 1). B is recognized as dominance-resistant, due to the low possibility of discovering objective vector dominating B via evolutionary approaches. In addition, the boundary where B is situated is specifically named the hardly dominated boundary (HDB) [35]. This is because the objective vector on the HDB can improve only one objective value to obtain a better one.

Coping strategies have been developed in recent years according to the categories of MOEAs (i.e., dominance-based MOEAs [8], decomposition-based MOEAs [45], and indicator-based MOEAs [12]). Dominance-based MOEAs commonly incorporate relaxed forms of Pareto dominance [7,10,18,29,44]. Decomposition-based MOEAs utilize effective decomposition methods [14,15,30]. Indicator-based ones employ the environmental selection based on the hypervolume contribution [1,12,16]. However, the existing MOEAs encounter challenges in effectively addressing DRS-MOPs concerning convergence and diversity. In this paper, we propose a metric based on proper Pareto optimality. To

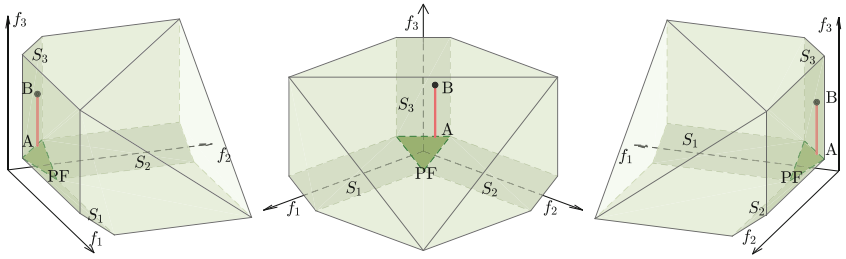



**Fig. 1.** Visual representations of a DRS-MOP from three distinct perspectives.

test the potential of this metric, we adopt multi-objective evolutionary algorithm based on decomposition (MOEA/D) [45] as the backbone. The proposed MOEA/D variant called MOEA/D-SPP achieves competitive performance in the experiments. The contributions of this paper are encapsulated in the following:

1. The PPO metric, inspired by proper Pareto optimality, is introduced. The PPO metric of DRS is generally large. As a result, it emerges as a valuable tool for discerning DRSs effectively.
2. SPP, a selection strategy based on the PPO metric for decomposition-based MOEAs, is designed. It assesses and ranks a solution subset of the population, enabling the identification and exclusion of DRSs.
3. MOEA/D-SPP, an MOEA/D variant incorporating the newly devised selection strategy, is proposed. Additionally, a mechanism for weight vector adjustment is integrated to augment the population diversity.

The remainder of this paper is organized as follows. Section 2 introduces the motivation of this paper. Section 3 elaborates on the details of MOEA/D-SPP. Section 4 presents the numerical experiments to demonstrate the performance of MOEA/D-SPP. Finally, Sect. 5 concludes this paper and discusses future works.

## 2 Motivation

### 2.1 Metric Based on Proper Pareto Optimality

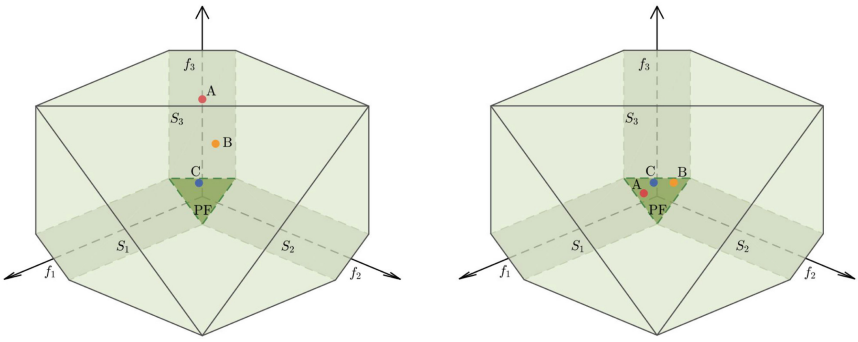
A solution $\mathbf{x}$ is properly Pareto-optimal when the maximum decrement among some objectives is satisfactory only at the cost of a finite increment in the others. The Definition 4 is equivalent to a properly Pareto-optimal decision vector $\mathbf{x} \in \Omega$ satisfying that:

$$h(\mathbf{x}, \mathbf{x}') = \frac{\max\limits_{1 \le i \le m} f_i(\mathbf{x}) - f_i(\mathbf{x}')}{\max\limits_{1 \le j \le m} f_j(\mathbf{x}') - f_j(\mathbf{x})} \le M < \infty, \tag{3}$$

where $\mathbf{x}' \in \Omega$ and $\max_{1 \le j \le m} f_j(\mathbf{x}') - f_j(\mathbf{x}) > 0$. We define a metric built upon $h(\mathbf{x}, \mathbf{x}')$. We refer to this metric as the PPO metric, which is formulated by:

$$\mathcal{M}_{\mathcal{S}}(\mathbf{x}) = \max_{\mathbf{x}' \in \mathcal{S}} h(\mathbf{x}, \mathbf{x}') = \max_{\mathbf{x}' \in \mathcal{S}} \left( \frac{\max\limits_{1 \le i \le m} f_i(\mathbf{x}) - f_i(\mathbf{x}')}{\max\limits_{1 \le j \le m} f_j(\mathbf{x}') - f_j(\mathbf{x})} \right), \tag{4}$$

where $\mathcal{S}$ is a solution set and $\mathbf{f}(\mathbf{x}) \neq \mathbf{f}(\mathbf{x}')$.




(a) Solution set with DRSs (b) Solution set without DRSs

**Fig. 2.** Two scenarios of the solution set $\mathcal{S}$ on a DRS-MOP.

Firstly, we examine how the PPO metric works on the solution set with DRSs. Recall that some objective values of the DRS are very poor while others are good. DRSs have a quite large value of the PPO metric. For example, Fig. 2(a) shows an example to illustrate how the PPO metric works on an MOP having DRSs. The solution set $\mathcal{S}$ is composed of three individuals A(0.25, 0.25, 1.75), B(0.25, 0.35, 1.05), and C(0.3, 0.25, 0.45), where A and B are DRSs. The PPO metric values are $\mathcal{M}_{\mathcal{S}}(\text{A}) = h(\text{A}, \text{C}) = 26$, $\mathcal{M}_{\mathcal{S}}(\text{B}) = h(\text{B}, \text{C}) = 12$, and $\mathcal{M}_{\mathcal{S}}(\text{C}) = h(\text{C}, \text{B}) \approx 0.083$, respectively. We can find that the PPO metric of DRS is larger because DRSs are extremely inferior to other solutions in some objectives.

Secondly, we investigate the solution set without DRSs. Figure 2(b) shows an example of the solution set without DRSs. The solutions are all on the PF. The solution set $\mathcal{S}$ is composed of three individuals A(0.45, 0.25, 0.35), B(0.15, 0.4, 0.45), and C(0.3, 0.25, 0.45). The PPO metric values are $\mathcal{M}_{\mathcal{S}}(\text{A}) = h(\text{A}, \text{B}) = 2$, $\mathcal{M}_{\mathcal{S}}(\text{B}) = h(\text{B}, \text{C}) = 1$, and $\mathcal{M}_{\mathcal{S}}(\text{C}) = h(\text{C}, \text{B}) = 1$, respectively. We can find that these values are generally small. This is because the largest difference in any objective is bound by the range of the PF. We also observe that these objective vectors have different PPO metric values. The calculation of the metric is based on the difference in the objectives, and thus different distributions of solutions can lead to distinct metric values. In other words, the distances between solutions and the density of solutions can affect the PPO metric values.

In short, The PPO metric is a potential criterion for identifying DRSs. When $\mathcal{M}_{\mathcal{S}}(\mathbf{x})$ is quite large, $\mathbf{x}$ strikes a poor trade-off between some objectives. We

can remove DRSs by removing the solutions that have large PPO metric values. Moreover, an additional scheme for the PPO metric is required to avoid accidentally eliminating a good solution.

### 2.2 MOEA/D

MOEA/D uses weight vectors and scalarization methods to convert an MOP into many single-objective optimization subproblems and solves these subproblems collaboratively. The effectiveness of MOEA/D is demonstrated in addressing various MOPs [36–38]. The performance of MOEA/D highly depends on the scalarization methods, and different scenarios may require different scalarization methods [19,46]. However, only some existing scalarization methods can cope with DRSs. For example, the penalty boundary intersection (PBI) method [45] fails as found in [34]. We suggest employing the PPO metric, which enables the application of any scalarization method in scenarios involving DRSs. $\mathcal{S}$ in the PPO metric is specified as $\mathcal{S}_i$ containing several closest solutions to the $i$-th subproblem and the offspring $o_i$. Thereafter, the DRSs with respect to the $i$-th subproblem can be distinguished.

In the following section, we propose MOEA/D-SPP, which selects solutions based on the PPO metric. The PBI method is used in MOEA/D-SPP. Let $\mathbf{w}$ be a weight vector and $\mathbf{z}^*$ be the ideal point, and the mathematical expression defining the PBI-based subproblem is as follows:

$$\min. \quad g^{\mathrm{pbi}}(\mathbf{x}|\mathbf{w}, \mathbf{z}^*) = d_1 + \theta d_2, \tag{5}$$

where

$$\begin{aligned} d_1 &= \frac{|(\mathbf{f}(\mathbf{x}) - \mathbf{z}^*)^{\mathsf{T}}\, \mathbf{w}|}{\|\mathbf{w}\|_2}\ , \\ d_2 &= \left\| \mathbf{f}(\mathbf{x}) - \mathbf{z}^* - d_1 \frac{\mathbf{w}}{\|\mathbf{w}\|_2} \right\|_2 . \end{aligned} \tag{6}$$

$\theta$ is a penalty parameter and we adopt $\theta = 10$ in this paper. Additionally, DRS-MOPs are often characterized by irregular PFs, while the performance of MOEA/D is also influenced by the shape of the PF [20,35,40,41]. Therefore, when applying MOEA/D to address DRS-MOPs, adjusting the weight vectors is also important to enhance its performance.

## 3 Algorithm

### 3.1 Framework

The framework of MOEA/D-SPP is outlined in Algorithm 1. The two key algorithmic components, namely **IdentifyDRS** and **AdjustWeights**, are detailed in Algorithm 2 and Algorithm 3, respectively. **IdentifyDRS** is the selection

strategy based on the PPO metric. **AdjustWeights** can adjust weight vectors properly. We adopt a multistage procedure [9]: the first stage is the same as MOEA/D-PBI; **IdentifyDRS** is introduced in the second stage; **IdentifyDRS** still executes and **AdjustWeights** improves the diversity finally.

**Algorithm 1.** MOEA/D-SPP

**Input:** $N$ (population size); $G_{max}$ (maximum number of generations); $T$ (neighborhood size); $\vartheta_{stage}$, and $a$ (control parameters).
**Output:** $\mathcal{P}$ (the final population).

1: Generate a random population $\mathcal{P} = \{\mathbf{x}_1, \ldots, \mathbf{x}_N\}$;
2: Generate the set of weight vectors $\mathcal{W} = \{\mathbf{w}_1, \ldots, \mathbf{w}_N\}$, and assign $T$ neighboring subproblems for each subproblem (denote the neighborhood of the $i$-th subproblem as $\mathcal{B}_i$);
3: Set $isUsed \leftarrow false$;
4: **for** $gen = 1$ to $G_{max}$ **do**
5:  **for** each subproblem $(i = 1, \ldots, N)$ **do**
6:  Select solutions from $\mathcal{B}_i$ to generate the offspring $\mathbf{o}_i$;
7:  **if** $gen > \vartheta_{stage}$ **then**
8:  $(\mathcal{P}, flag) \leftarrow$ **IdentifyDRS** $(\mathcal{P}, \mathbf{o}_i, T, a)$;
9:  **if** $flag = false$ **then**
10:  **Continue**;
11:  **end if**
12:  **end if**
13:  Update the $\mathcal{B}_i$ with $\mathbf{o}_i$;
14:  **end for**
15:  **if** $gen > 2 \cdot \vartheta_{stage}$ and $isUsed = false$ **then**
16:  $\mathcal{W} \leftarrow$ **AdjustWeights** $(\mathcal{P})$;
17:  $isUsed \leftarrow true$;
18:  **end if**
19: **end for**

The algorithm first initiates the population $\mathcal{P}$ and the set of weight vectors $\mathcal{W}$ and determines neighbors for each weight vector (lines 1–2). The neighborhood of the $i$-th subproblem is denoted as $\mathcal{B}_i$. Subsequently, $isUsed$ is introduced, which represents the evolutionary stage. In MOEA/D-SPP, different goals are achieved at different stages. It is set to 0 initially (line 3). After the initialization, the algorithm iterates over each subproblem. For the subproblem with index $i$ in each generation (line 5), the algorithm first selects solutions from $\mathcal{B}_i$ to generate the offspring $\mathbf{o}_i$ (line 6). Subsequently, if the generation number $gen$ surpasses the control parameter $\vartheta_{stage}$, the **IdentifyDRS** component is executed. This component employs the PPO metric to identify DRSs (lines 7–8). If the offspring $\mathbf{o}_i$ is considered to be a DRS (i.e., $flag = false$), the algorithm skips updating $\mathcal{B}_i$ with $\mathbf{o}_i$ and proceeds to the next subproblem (lines 9–11). Otherwise, $\mathbf{o}_i$ is used to update $\mathcal{B}_i$ according to the PBI method (line 13). If $gen \leq \vartheta_{stage}$, the **IdentifyDRS** component remains inactive and the subproblem is updated regularly. Moreover, if $gen$ surpasses $2 \cdot \vartheta_{stage}$ and $isUsed = false$ (line 15),

the **AdjustWeights** component is invoked only once. The weight vectors are adjusted to the appropriate distribution according to the current population, thereby enhancing the population diversity in subsequent iterations (line 16). It indicates **AdjustWeights** is used and let $isUsed$ be $true$ (line 17). Finally, the population $\mathcal{P}$ is returned as the output of the algorithm.

### 3.2 Selection Strategy Based on PPO Metric (SPP)

The **IdentifyDRS** component, as illustrated in Algorithm 2, plays a crucial role within the MOEA/D-SPP framework. It employs the PPO metric to assess the acceptance of new solutions and the removal of old ones, which can effectively distinguish the DRSs.

**Algorithm 2. IdentifyDRS** $(\mathcal{P}, \mathbf{o}_i, T, a)$

**Input:** $\mathcal{P}$ (the population); $\mathbf{o}_i$ (the offspring); $T$ (the neighborhood size) and $a$ (control parameter).

**Output:** $\mathcal{P}$ (the updated population); $flag$ (the symbol indicating whether to accept the offspring).

1: $\mathcal{S}_i \leftarrow \{T$ solutions closest to $\mathbf{w}_i$, $\mathbf{o}_i\}$;
2: Compute the PPO metric for each solution in $P_i$ according to Eq. (4);
3: $\mathbf{x}_{worst} \leftarrow \arg\max_{\mathbf{x}\in\mathcal{S}_i}\{\mathcal{M}_{\mathcal{S}_i}(\mathbf{x})\}$;
4: **if** $\mathbf{o}_i$ is $\mathbf{x}_{worst}$ or $\mathcal{M}_{\mathcal{S}_i}(o_i) \geq a$ **then**
5: $\quad flag \leftarrow false$;
6: $\quad$ **return**;
7: **else**
8: $\quad flag \leftarrow true$;
9: **end if**
10: **if** $\mathcal{M}_{\mathcal{S}_i}(\mathbf{x}_{worst}) \geq N^{1/(m-1)}$ **then**
11: $\quad$ **repeat**
12: $\quad\quad \mathbf{x}' \leftarrow$ Randomly select from $\mathcal{S}_i$;
13: $\quad$ **until** $\mathcal{M}_{\mathcal{S}_i}(\mathbf{x}') \leq a$ or $\mathbf{x}' = \arg\min_{\mathbf{x}\in\mathcal{S}_i}\{\mathcal{M}_{\mathcal{S}_i}(\mathbf{x})\}$
14: $\quad$ Replace the solution in $P$ corresponding to $\mathbf{x}_{worst}$ with $\mathbf{x}'$;
15: **end if**

The algorithmic component initiates by constructing a set $\mathcal{S}_i$ comprising the $T$ solutions closest to the weight vector $\mathbf{w}_i$ and the offspring $\mathbf{o}_i$ (line 1). For each solution in $\mathcal{S}_i$, the PPO metric is computed according to Eq. (4) (line 2). The solution with the worst PPO metric value, denoted as $\mathbf{x}_{worst}$, is identified (line 3). Thereafter, the algorithm checks whether the current offspring $\mathbf{o}_i$ is $\mathbf{x}_{worst}$ or if its PPO metric value exceeds the predefined value $a$ (line 4). If either condition is met, the offspring is deemed to have poor quality. Then the offspring should not be accepted and the algorithm returns $flag$ of $false$ and the unchanged $\mathcal{P}$ (lines 5–6). Otherwise, the offspring is considered suitable to be accepted and let $flag$ be $true$ (line 8).

As mentioned in Sect. 2.1, the PPO metric values of the solutions on the PF are related to the population density. The density of uniform sampling points suffers from the curse of dimensionality and the density is proportional to $q^{1/p}$, where $p$ is the space dimension and $q$ is the sample size [3,17]. We adopt the threshold of $N^{1/(m-1)}$ considering that the PF is an $(m-1)$ dimensional manifold. Specifically, when the PPO metric of $\mathbf{x}_{worst}$ exceeds $N^{1/(m-1)}$ (line 10), the algorithm substitutes the solution in $P$ that corresponds to $\mathbf{x}_{worst}$ by randomly selecting a suitable solution from $\mathcal{S}_i$ (lines 11–14).

### 3.3 Adjustment of Weight Vectors

The **AdjustWeights** component, shown in Algorithm 3, aims to improve the diversity of the population. It generates a set of new weight vectors based on the distribution of solutions in the current population.

**Algorithm 3. AdjustWeights** ($\mathcal{P}$)

**Input:** $\mathcal{P}$ (the population).
**Output:** $\mathcal{W}$ (the set of weight vectors).
1: $p \leftarrow$ Estimate a parameter via $\mathcal{P}$ for the PF shape [24];
2: $\mathcal{R} \leftarrow$ Generate a set of uniformly distributed weight vectors;
3: $w_i \leftarrow 1 - w_i^{1/p}$ for $i = 1, \ldots, m$ and every $\mathbf{w} \in \mathcal{R}$;
4: $\mathcal{W} \leftarrow \emptyset$;
5: **for** $i = 1$ to $m$ **do**
6: $\quad \mathbf{w}_{tmp} \leftarrow \arg\min_{\mathbf{w} \in \mathcal{R}} w_i$;
7: $\quad \mathcal{W} \leftarrow \mathcal{W} \cup \mathbf{w}_{tmp}$;
8: $\quad \mathcal{R} \leftarrow \mathcal{R} \setminus \mathbf{w}_{tmp}$;
9: **end for**
10: **while** $|\mathcal{W}| < N$ **do**
11: $\quad \mathbf{w}_{tmp} \leftarrow \arg\max_{\mathbf{w} \in \mathcal{R}} \left\{ \min_{\mathbf{v} \in \mathcal{W}} ||\mathbf{w} - \mathbf{v}||_2 \right\}$;
12: $\quad \mathcal{W} \leftarrow \mathcal{W} \cup \mathbf{w}_{tmp}$;
13: $\quad \mathcal{R} \leftarrow \mathcal{R} \setminus \mathbf{w}_{tmp}$;
14: **end while**

To estimate the shape of the PF, the component is inspired by [24] to estimate the PF shape by finding the most consistent contour curve of the modified $L_p$-norm distance (line 1). This estimation works as follows:

1. Firstly, compute the modified $L_p$-norm distances for objective vectors of $\mathcal{P}$ using candidate values of $p$ ($p > 0$). The set of distances is denoted as $G_p = \{g(\mathbf{x}|p) \mid \mathbf{x} \in \mathcal{P}\}$ where $g(\mathbf{x}|p) = \left(\sum_{i=1}^{m}(1 - f_i(\mathbf{x}))^p\right)^{1/p}$.
2. Secondly, select the $p$ value with smallest standard deviation of $G_p$.

In other words, the estimation determines a value of $p$.

After that, a sufficiently large set of uniformly distributed weight vectors $\mathcal{R}$ is generated by using the two-layer weight vectors generation method [6,23] (line 2).

The endpoints of weight vectors in $\mathcal{R}$ are located on the unit simplex. With $p$ and $\mathcal{R}$, the desired weight vectors can be obtained. Each $\mathbf{w} \in \mathcal{R}$ is transformed based on the value of $p$ (line 3). The transformation method is consistent with the aforementioned estimation method. Then, a subset $\mathcal{W}$ is selected from $\mathcal{R}$. The component finds $m$ extreme weight vectors as initialized elements in $\mathcal{W}$ (lines 5–8). In line 6, we can find that the extreme weight vectors represent those having the lowest value on at least one objective. The remaining $(N - m)$ weight vectors for $\mathcal{W}$ are chosen from $\mathcal{R}$ one by one (lines 10–14). Sparse weight vectors are prioritized for selection. The sparsity of a weight vector is defined in line 11. In other words, the sparsity is the minimum Euclidean distance from the weight vector to the one within $\mathcal{W}$.

## 4 Numerical Experiments

### 4.1 Benchmark Problems

In this section, we assess the proposed MOEA/D-SPP on mDTLZ1-mDTLZ4 [35]. These test problems exhibit two common features: 1) irregularity in the PF shape (e.g., the PF of mDTLZ1 is an inverted triangle); and 2) complexity in the feasible objective region (e.g., all problems exhibit HDBs). Note that an MOP featuring more than 3 objectives is viewed as a many-objective optimization problem. We consider 3 and 5 as two settings of the objective dimension. The number of decision variables is set to 10 for 3-objective problems and 12 for 5-objective problems in each instance.

Each algorithm undergoes 30 independent runs on each test instance. The population size is set to 105 for 3-objective instances and 210 for the 5-objective instances. The maximum number of function evaluations is set to 105,000 (i.e., 105×1000) for the 3-objective case and 252,000 (i.e., 210×1200) for the 5-objective case. All experiments are conducted on the PlatEMO platform [32].

### 4.2 Evaluation Metrics

The inverted generational distance (IGD) metric [4] and the hypervolume (HV) metric [47] are employed for assessing the performance of the obtained solution sets. IGD calculates the average distances from each objective vector to its nearest reference point of the PF. A smaller IGD value indicates a better performance of the algorithm. HV describes the size of the region covered by the obtained objective vectors and the reference point. A larger HV value represents the better performance of the algorithm.

The mean metric values are employed to show the statistical performance of each algorithm on each test problem [22,26,39]. Statistical analysis of the differences between mean metric values is conducted using Wilcoxon's rank-sum test at a significance level of 0.05 [21]. The symbols "+", "−", and "=" denote that the compared algorithm performs statistically better than, worse than, and equal to MOEA/D-SPP, respectively.

### 4.3 Compared Algorithms

We conduct a comparative analysis of our proposed algorithm against five representative MOEAs. The parameters of algorithms are configed according to the corresponding references. NSGA-II [8] is a classical dominance-based MOEA utilizing the Pareto dominance criterion and employing crowding distance for diversity evaluation. MOEA/D-Gen [15] utilizes generalized decomposition, with parameters $\delta$ and $\rho$ set to 0.01. mNSGA-II [31] addresses DRSs by modifying objective values, with control parameter $\alpha$ set to 0.01. PMEA [28] incorporates preprocessing and penalty mechanisms, with parameter $r$ set to 1.5. MOEA/D-OMDEA [33] integrates the OM-dominance criterion, with parameters $\alpha_1 = 0.25$, $\alpha_2 = 0.1$, $\tau = 0.7$, and $K = 30$ for 3-objective cases and $K = 50$ for 5-objective cases.

In our compared algorithms, the control parameters in MOEA/D-SPP are established as $a = 2$ and $\vartheta_{stage} = 300$. In general, the neighborhood size $T$ for MOEA/D variants is set to $\lceil N/10 \rceil$. All the algorithms except for MOEA/D-OMDEA employ the simulated binary crossover and polynomial mutation [5] to generate new solutions. The distribution indexes are set to 20. Besides, the crossover and mutation probabilities are configured at 1 and 1/ n respectively. MOEA/D-OMDEA follows the operators outlined in [25].

### 4.4 Comparison Results

The mean IGD and HV metric values achieved by algorithms on all instances are presented in Tables 1 and 2 respectively. The best result for each problem is highlighted in bold. the standard deviations of means are in parentheses.

In Table 1, MOEA/D-SPP exhibits the best mean IGD metric values on 3-objective mDTLZ1, mDTLZ2, and mDTLZ3. Furthermore, according to Wilcoxon's rank-sum test, MOEA/D-SPP outperforms NSGA-II, MOEA/D-Gen, mNSGA-II, PMEA, and MOEA/D-OMDEA on 7, 5, 6, 6, and 6 out of 8 instances, respectively. In contrast, it is significantly outperformed by NSGA-II, MOEA/D-Gen, and MOEA/D-OMDEA on 1, 2, and 1 out of 8 problems. In summary, MOEA/D-SPP demonstrates a substantial improvement in terms of the IGD metric compared to these competitors.

In Table 2, the best HV result of each instance is achieved by the following algorithms: MOEA/D-SPP on mDTLZ1-mDTLZ3 and PMEA on mDTLZ4. According to Wilcoxon's rank-sum test, MOEA/D-SPP outperforms NSGA-II, MOEA/D-Gen, mNSGA-II, PMEA, and MOEA/D-OMDEA on 8, 7, 7, 6, and 7 out of 8 instances, respectively. It is only outperformed by mNSGA-II, PMEA, and MOEA/D-OMDEA on 1, 2, and 1 out of 8 instances. Overall, MOEA/D-SPP also demonstrates advantages over these peer algorithms in terms of the HV metric.

The overall result of the IGD metric conforms to that of the HV metric. The proposed MOEA/D-SPP exhibits competitive performance with respect to both IGD and HV metrics on 3-objective and 5-objective mDTLZ1-mDTLZ4. Furthermore, Friedman tests are conducted to compare MOEA/D-SPP with five

**Table 1.** Means and standard deviations of IGD metric values obtained by algorithms.

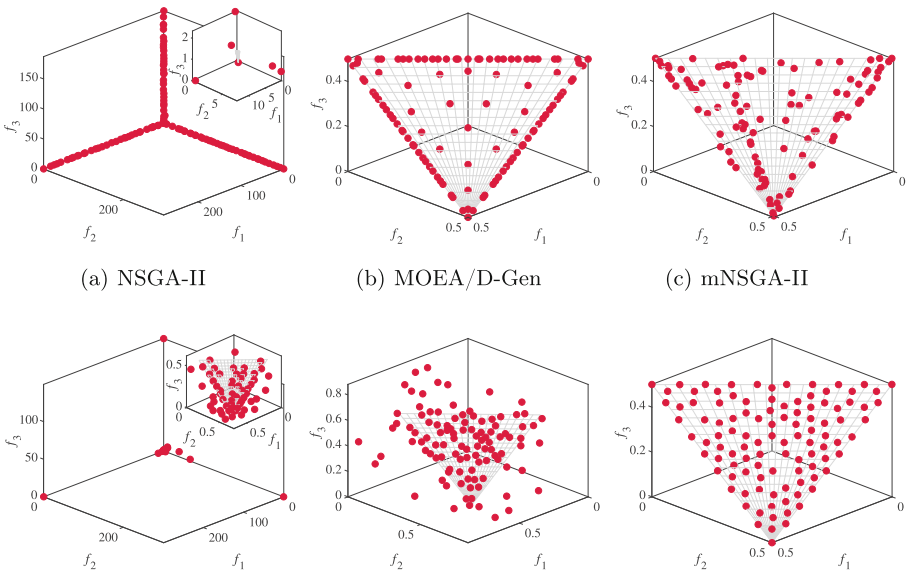
| IGD | $m$ | NSGA-II | MOEA/D-Gen | mNSGA-II | PMEA | MOEA/D-OMDEA | MOEA/D-SPP |
|---|---|---|---|---|---|---|---|
| mDTLZ1 | 3 | 1.0787e+0 - | 2.9330e-2 - | 2.6286e-2 - | 3.1430e-2 - | 1.7657e-1 - | **2.1800e-2** |
| | | (1.35e+0) | (4.38e-5) | (1.66e-3) | (1.37e-3) | (1.04e-1) | (1.06e-2) |
| mDTLZ2 | 3 | 1.1600e-1 - | 6.9634e-2 - | 6.3331e-2 - | 1.0411e-1 - | 7.6944e-2 - | **5.4112e-2** |
| | | (8.91e-3) | (2.69e-4) | (2.01e-3) | (2.18e-2) | (9.40e-3) | (9.91e-4) |
| mDTLZ3 | 3 | 2.0386e+0 - | 6.9580e-2 - | 6.4027e-2 - | 9.8289e-2 - | 1.2934e-1 - | **5.8790e-2** |
| | | (2.67e+0) | (3.35e-4) | (4.02e-3) | (1.47e-2) | (5.10e-2) | (4.06e-3) |
| mDTLZ4 | 3 | 1.3340e-1 - | 1.1960e-1 - | 8.0513e-2 = | 7.9877e-2 = | **7.7648e-2** = | 1.1494e-1 |
| | | (1.26e-2) | (1.17e-1) | (1.56e-2) | (1.05e-2) | (4.84e-3) | (1.71e-1) |
| mDTLZ1 | 5 | 2.8688e+1 - | 5.7983e-2 - | 1.5148e-1 - | 9.6846e-2 - | 2.7975e-1 - | **5.6686e-2** |
| | | (1.63e+1) | (2.61e-5) | (1.91e-2) | (6.68e-3) | (8.98e-2) | (3.45e-3) |
| mDTLZ2 | 5 | 2.4618e-1 - | **1.4887e-1** + | 2.1538e-1 - | 2.8372e-1 - | 1.8211e-1 + | 1.9844e-1 |
| | | (1.52e-2) | (5.02e-4) | (9.02e-3) | (2.84e-2) | (1.12e-2) | (2.47e-2) |
| mDTLZ3 | 5 | 4.7116e+1 - | **1.4780e-1** + | 3.9344e-1 - | 3.1063e-1 - | 4.4687e-1 - | 2.1978e-1 |
| | | (3.55e+1) | (9.80e-4) | (2.75e-2) | (3.63e-2) | (1.21e-1) | (3.01e-2) |
| mDTLZ4 | 5 | 2.8455e-1 + | 2.1408e-1 = | 2.2922e-1 = | **2.1218e-1** = | 3.3439e-1 - | 3.0939e-1 |
| | | (1.69e-2) | (4.45e-2) | (1.61e-2) | (1.78e-2) | (3.08e-2) | (2.05e-1) |
| ±/= | | 1/7/0 | 2/5/1 | 0/6/2 | 0/6/2 | 1/6/1 | – |

**Table 2.** Means and standard deviations of HV metric values obtained by algorithms.

| HV | $m$ | NSGA-II | MOEA/D-Gen | mNSGA-II | PMEA | MOEA/D-OMDEA | MOEA/D-SPP |
|---|---|---|---|---|---|---|---|
| mDTLZ1 | 3 | 2.8699e-2 - | 5.1641e-1 - | 5.0487e-1 - | 4.7351e-1 - | 1.8425e-1 - | **5.2067e-1** |
| | | (2.79e-2) | (4.89e-4) | (5.16e-3) | (7.56e-3) | (1.46e-1) | (3.01e-2) |
| mDTLZ2 | 3 | 8.6691e-1 - | 1.0237e+0 - | 1.0207e+0 - | 1.0152e+0 - | 1.0011e+0 - | **1.0340e+0** |
| | | (2.51e-2) | (1.35e-4) | (4.13e-3) | (9.03e-3) | (7.03e-3) | (1.16e-3) |
| mDTLZ3 | 3 | 3.5195e-2 - | 1.0234e+0 - | 1.0227e+0 - | 9.9438e-1 - | 8.4447e-1 - | **1.0314e+0** |
| | | (4.30e-2) | (5.67e-4) | (6.07e-3) | (1.06e-2) | (1.26e-1) | (4.75e-3) |
| mDTLZ4 | 3 | 7.9821e-1 - | 8.7509e-1 - | 9.5801e-1 + | **9.9996e-1** + | 9.7326e-1 + | 8.8277e-1 |
| | | (3.41e-2) | (1.38e-1) | (5.06e-2) | (4.33e-2) | (7.34e-3) | (1.83e-1) |
| mDTLZ1 | 5 | 2.6754e-5 - | 6.6176e-2 - | 1.2345e-2 - | 2.4288e-2 - | 1.7688e-3 - | **7.7432e-2** |
| | | (1.47e-4) | (3.26e-4) | (3.16e-3) | (3.63e-3) | (2.09e-3) | (6.42e-3) |
| mDTLZ2 | 5 | 1.8242e-1 - | 3.9667e-1 - | 2.6287e-1 - | 4.4651e-1 - | 4.0032e-1 - | **4.6319e-1** |
| | | (2.20e-2) | (1.50e-3) | (1.41e-2) | (4.40e-3) | (6.41e-3) | (2.30e-3) |
| mDTLZ3 | 5 | 0.0000e+0 - | 4.0003e-1 - | 3.3793e-2 - | 2.8197e-1 - | 8.9890e-2 - | **4.6964e-1** |
| | | (0.00e+0) | (1.87e-3) | (1.70e-2) | (1.41e-2) | (7.18e-2) | (3.11e-3) |
| mDTLZ4 | 5 | 1.3682e-1 - | 3.0907e-1 = | 2.0975e-1 - | **4.3240e-1** + | 1.1367e-1 - | 3.0900e-1 |
| | | (1.55e-2) | (6.46e-2) | (2.81e-2) | (4.44e-3) | (3.24e-2) | (1.30e-1) |
| ±/= | | 0/8/0 | 0/7/1 | 1/7/0 | 2/6/0 | 1/7/0 | – |

**Table 3.** Average ranks of representative algorithms and MOEA/D-SPP obtained by Friedman test.

| | IGD | HV |
|---|---|---|
| NSGA-II | 5.625 | 5.875 |
| MOEA/D-Gen | 2.5 | 2.625 |
| mNSGA-II | 3 | 3.75 |
| PMEA | 3.5 | 2.75 |
| MOEA/D-OMDEA | 4.125 | 4.375 |
| MOEA/D-SPP | **2.25** | **1.625** |



(a) NSGA-II (b) MOEA/D-Gen (c) mNSGA-II

(d) PMEA (e) MOEA/D-OMDEA (f) MOEA/D-SPP

**Fig. 3.** The approximate set with the median IGD metric value on 3-objective mDTLZ1.

other algorithms. Table 3 reveals that MOEA/D-SPP achieves the best average rankings across both metrics, signifying its superior performance over the other algorithms. We visualize the approximate set of each algorithm on 3-objective mDTLZ1 in Fig. 3. The approximate set with the median IGD metric value among 30 runs is selected. These figures reveal that only MOEA/D-SPP discovers a well-distributed solution set across the entire PF, effectively balancing the convergence and diversity.

### 4.5 Effectiveness Analysis

The proposed MOEA/D-SPP utilizes the PBI method and introduces two key algorithmic components: **IdentifyDRS** and **AdjustWeights**. To investigate the individual effectiveness of each component within the proposed algorithm, several ablation experiments are conducted. MOEA/D-SPP is compared with the following three variants:

- V-a: MOEA/D-SPP without the execution of any of the two components. In such a case, its behavior is identical to that of MOEA/D-PBI.
- V-b: MOEA/D-SPP without **AdjustWeights**.
- V-c: MOEA/D-SPP without **IdentifyDRS**.

Experimental results are presented in Table 4 and Table 5. The results show that MOEA/D-SPP outperforms other variants of MOEA/D-SPP in terms of both IGD and HV metrics across most instances. According to Table 6, V-a demonstrates competitive overall performance compared to V-b and V-c. On the one hand, **IdentifyDRS** eliminates DRSs and fosters population convergence. Thus it may make the population lack diversity in V-b. On the other hand, **AdjustWeights** estimates the PF shape using the population with DRSs in V-c, making the estimation highly inaccurate. Therefore, MOEA/D-SPP simultaneously employs the two strategies, striking a better balance between convergence and diversity and having the best performance.

**Table 4.** Means and standard deviations of IGD metric values obtained by MOEA/D-SPP and its variants.

| IGD | $m$ | V-a | V-b | V-c | MOEA/D-SPP |
|---|---|---|---|---|---|
| mDTLZ1 | 3 | 4.5380e-2 - | 3.0038e-2 - | 9.3394e-2 - | **2.1800e-2** |
| | | (4.19e-3) | (3.72e-4) | (4.27e-2) | (1.06e-2) |
| mDTLZ2 | 3 | 7.3003e-2 - | 7.8833e-2 - | 7.5943e-2 - | **5.4112e-2** |
| | | (7.08e-4) | (4.13e-3) | (1.41e-3) | (9.91e-4) |
| mDTLZ3 | 3 | 7.3931e-2 - | 1.0710e-1 - | 9.7051e-2 - | **5.8790e-2** |
| | | (1.35e-3) | (9.50e-3) | (4.08e-2) | (4.06e-3) |
| mDTLZ4 | 3 | 1.8893e-1 - | 1.5781e-1 - | 2.1567e-1 - | **1.1494e-1** |
| | | (1.72e-1) | (1.56e-1) | (2.21e-1) | (1.71e-1) |
| mDTLZ1 | 5 | 1.7306e-1 - | 8.4131e-2 - | 1.0720e-1 - | **5.6686e-2** |
| | | (1.12e-3) | (5.13e-3) | (5.07e-3) | (3.45e-3) |
| mDTLZ2 | 5 | 3.3092e-1 - | 3.2329e-1 - | **1.5836e-1** + | 1.9844e-1 |
| | | (3.07e-3) | (1.80e-2) | (6.90e-3) | (2.47e-2) |
| mDTLZ3 | 5 | 3.0808e-1 - | 3.2506e-1 - | **2.1385e-1** = | 2.1978e-1 |
| | | (1.91e-3) | (2.00e-2) | (1.70e-2) | (3.01e-2) |
| mDTLZ4 | 5 | 3.5856e-1 - | 3.3311e-1 - | 3.8760e-1 - | **3.0939e-1** |
| | | (1.85e-2) | (1.38e-1) | (1.96e-1) | (2.05e-1) |
| ±/= | | 0/8/0 | 0/8/0 | 1/6/1 | – |

**Table 5.** Means and standard deviations of HV metric values obtained by MOEA/D-SPP and its variants.

| HV | $m$ | V-a | V-b | V-c | MOEA/D-SPP |
|---|---|---|---|---|---|
| mDTLZ1 | 3 | 4.1774e-1 - | 5.0819e-1 - | 2.8524e-1 - | **5.2067e-1** |
| | | (1.54e-2) | (1.65e-3) | (7.52e-2) | (3.01e-2) |
| mDTLZ2 | 3 | 1.0157e+0 - | 1.0089e+0 - | 9.6192e-1 - | **1.0340e+0** |
| | | (1.51e-3) | (3.93e-3) | (6.11e-3) | (1.16e-3) |
| mDTLZ3 | 3 | 1.0143e+0 - | 1.0001e+0 - | 9.0952e-1 - | **1.0314e+0** |
| | | (3.27e-3) | (4.86e-3) | (1.02e-1) | (4.75e-3) |
| mDTLZ4 | 3 | 7.2188e-1 - | 8.5774e-1 = | 6.6940e-1 - | **8.8277e-1** |
| | | (2.49e-1) | (2.18e-1) | (2.02e-1) | (1.83e-1) |
| mDTLZ1 | 5 | 4.6000e-2 - | 7.0221e-2 - | 3.1131e-2 - | **7.7432e-2** |
| | | (7.03e-5) | (2.16e-3) | (3.29e-3) | (6.42e-3) |
| mDTLZ2 | 5 | 2.9611e-1 - | 4.0922e-1 - | 4.3055e-1 - | **4.6319e-1** |
| | | (2.94e-3) | (4.33e-3) | (3.80e-3) | (2.30e-3) |
| mDTLZ3 | 5 | 3.3234e-1 - | 4.0958e-1 - | 3.0990e-1 - | **4.6964e-1** |
| | | (4.40e-3) | (5.13e-3) | (1.18e-2) | (3.11e-3) |
| mDTLZ4 | 5 | 2.3319e-1 - | 3.0144e-1 = | 1.3606e-1 - | **3.0900e-1** |
| | | (5.87e-2) | (1.03e-1) | (1.10e-1) | (1.30e-1) |
| ±/= | | 0/8/0 | 0/6/2 | 0/8/0 | – |

**Table 6.** Average ranks of MOEA/D-SPP and its variants obtained by Friedman test

| | IGD | HV |
|---|---|---|
| V-a | 3 | 2.875 |
| V-b | 2.875 | 2.375 |
| V-c | 2.875 | 3.75 |
| MOEA/D-SPP | **1.25** | **1** |

## 5 Conclusion

In this study, we have introduced the PPO metric and proposed MOEA/D-SPP. The PPO metric aims to identify DRSs, while MOEA/D-SPP implements it to alleviate the adverse effects of DRSs. **AdjustWeights**, namely the one-shot weight vector adjustment in MOEA/D-SPP, further enhances the population diversity. To comprehensively assess the efficacy of the proposed algorithm, we have conducted experimental evaluations on benchmarks of DRS-MOPs, specifically 3-objective and 5-objective mDTLZ1-mDTLZ4. The performance of MOEA/D-SPP is compared against five representative MOEAs. The obtained results reveal the promising performance of MOEA/D-SPP in

addressing DRS-MOPs. Additionally, ablation experiments have validated the effectiveness of each algorithmic component in MOEA/D-SPP.

In the future, we plan to extend the applicability of the PPO metric to other categories of MOEAs (i.e., dominance-based MOEAs and indicator-based MOEAs), aiming to enhance their performance in addressing DRS-MOPs. We also intend to delve into real-world scenarios involving DRSs and apply our algorithm.

**Acknowledgments.** This work was supported by the National Natural Science Foundation of China (Grant No. 62106096), the Natural Science Foundation of Guangdong Province (Grant No. 2024A1515011759), the National Natural Science Foundation of Shenzhen (Grant No. JCYJ20220530113013031), and the 2023 Graduate Innovation Practice Fund Project of Southern University of Science and Technology.

## References

1. Bader, J., Zitzler, E.: HypE: an algorithm for fast hypervolume-based many-objective optimization. Evol. Comput. **19**(1), 45–76 (2011)
2. Batista, L.S., Campelo, F., Guimarães, F.G., Ramírez, J.A.: A comparison of dominance criteria in many-objective optimization problems. In: 2011 IEEE Congress of Evolutionary Computation (CEC), pp. 2359–2366. IEEE (2011)
3. Bellman, R.: Dynamic Programming, vol. 1, pp. 3–25. Princeton University Press, Princeton, NJ, USA (1958)
4. Coello, C.A.C., Cortés, N.C.: Solving multiobjective optimization problems using an artificial immune system. Genet. Program Evolvable Mach. **6**, 163–190 (2005)
5. Deb, K., Goyal, M., et al.: A combined genetic adaptive search (GeneAS) for engineering design. Comput. Sci. Inf. **26**, 30–45 (1996)
6. Deb, K., Jain, H.: An evolutionary many-objective optimization algorithm using reference-point-based nondominated sorting approach, part I: solving problems with box constraints. IEEE Trans. Evol. Comput. **18**(4), 577–601 (2013)
7. Deb, K., Mohan, M., Mishra, S.: Towards a quick computation of well-spread Pareto-optimal solutions. In: Fonseca, C.M., Fleming, P.J., Zitzler, E., Thiele, L., Deb, K. (eds.) EMO 2003. LNCS, vol. 2632, pp. 222–236. Springer, Heidelberg (2003). https://doi.org/10.1007/3-540-36970-8_16
8. Deb, K., Pratap, A., Agarwal, S., Meyarivan, T.: A fast and elitist multiobjective genetic algorithm: NSGA-II. IEEE Trans. Evol. Comput. **6**(2), 182–197 (2002)
9. Deb, K., do Val Lopes, C.L., Martins, F.V.C., Wanner, E.F.: Identifying Pareto fronts reliably using a multi-stage reference-vector-based framework. IEEE Trans. Evol. Comput. **28**(1), 252–266 (2024)
10. Di Pierro, F., Khu, S.T., Savic, D.A.: An investigation on preference order ranking scheme for multiobjective evolutionary optimization. IEEE Trans. Evol. Comput. **11**(1), 17–45 (2007)
11. Duan, J., He, Z., Yen, G.G.: Robust multiobjective optimization for vehicle routing problem with time windows. IEEE Trans. Cybern. **52**(8), 8300–8314 (2021)
12. Emmerich, M., Beume, N., Naujoks, B.: An EMO algorithm using the hypervolume measure as selection criterion. In: Coello Coello, C.A., Hernández Aguirre, A., Zitzler, E. (eds.) Evolutionary Multi-Criterion Optimization, pp. 62–76. Springer, Berlin, Heidelberg (2005). https://doi.org/10.1007/978-3-540-31880-4_5

13. Geoffrion, A.M.: Proper efficiency and the theory of vector maximization. J. Math. Anal. Appl. **22**(3), 618–630 (1968)
14. Giagkiozis, I., Fleming, P.J.: Methods for multi-objective optimization: an analysis. Inf. Sci. **293**, 338–350 (2015)
15. Giagkiozis, I., Purshouse, R.C., Fleming, P.J.: Generalized decomposition. In: Purshouse, R.C., Fleming, P.J., Fonseca, C.M., Greco, S., Shaw, J. (eds.) Evolutionary Multi-Criterion Optimization, pp. 428–442. Springer, Berlin, Heidelberg (2013). https://doi.org/10.1007/978-3-642-37140-0_33
16. Guerreiro, A.P., Fonseca, C.M.: Hypervolume sharpe-ratio indicator: formalization and first theoretical results. In: Handl, J., Hart, E., Lewis, P.R., López-Ibáñez, M., Ochoa, G., Paechter, B. (eds.) PPSN 2016. LNCS, vol. 9921, pp. 814–823. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-45823-6_76
17. Hastie, T., Tibshirani, R., Friedman, J.: The Elements of Statistical Learning. Springer, New York, NY (2009). https://doi.org/10.1007/978-0-387-84858-7
18. Ikeda, K., Kita, H., Kobayashi, S.: Failure of Pareto-based MOEAs: does non-dominated really mean near to optimal? In: Proceedings of the 2001 Congress on Evolutionary Computation (IEEE Cat. No. 01TH8546), vol. 2, pp. 957–962. IEEE (2001)
19. Ishibuchi, H., Akedo, N., Nojima, Y.: A study on the specification of a scalarizing function in MOEA/D for many-objective knapsack problems. In: Nicosia, G., Pardalos, P. (eds.) Learning and Intelligent Optimization, pp. 231–246. Springer, Berlin, Heidelberg (2013). https://doi.org/10.1007/978-3-642-44973-4_24
20. Ishibuchi, H., Setoguchi, Y., Masuda, H., Nojima, Y.: Performance of decomposition-based many-objective algorithms strongly depends on Pareto front shapes. IEEE Trans. Evol. Comput. **21**(2), 169–190 (2016)
21. Li, K., Wang, H., Wang, W., Wang, F., Cui, Z.: Improving artificial bee colony algorithm using modified nearest neighbor sequence. J. King Saud Univ. Comput. Inf. Sci. **34**(10), 8807–8824 (2022)
22. Li, K., et al.: A new artificial bee colony algorithm based on modified search strategy. Int. J. Comput. Sci. Math. **15**(4), 387–395 (2022)
23. Li, K., Deb, K., Zhang, Q., Kwong, S.: An evolutionary many-objective optimization algorithm based on dominance and decomposition. IEEE Trans. Evol. Comput. **19**(5), 694–716 (2014)
24. Liang, Z., Hu, K., Ma, X., Zhu, Z.: A many-objective evolutionary algorithm based on a two-round selection strategy. IEEE Trans. Cybern. **51**(3), 1417–1429 (2019)
25. Liu, H.l., Li, X.: The multiobjective evolutionary algorithm based on determined weight and sub-regional search. In: 2009 IEEE Congress on Evolutionary Computation (CEC), pp. 1928–1934. IEEE (2009)
26. Liu, Y., Liu, J., Teng, X.: Single-particle optimization for network embedding preserving both local and global information. Swarm Evol. Comput. **71**, 101069 (2022)
27. Liu, Y., Liu, J., Wu, K.: Cost-effective competition on social networks: a multi-objective optimization perspective. Inf. Sci. **620**, 31–46 (2023)
28. Liu, Y., Zhu, N., Li, M.: Solving many-objective optimization problems by a Pareto-based evolutionary algorithm with preprocessing and a penalty mechanism. IEEE Trans. Cybern. **51**(11), 5585–5594 (2020)
29. López Jaimes, A., Coello Coello, C.A., Aguirre, H., Tanaka, K.: Adaptive objective space partitioning using conflict information for many-objective optimization. In: Takahashi, R.H.C., Deb, K., Wanner, E.F., Greco, S. (eds.) EMO 2011. LNCS, vol. 6576, pp. 151–165. Springer, Heidelberg (2011). https://doi.org/10.1007/978-3-642-19893-9_11

30. Miettinen, K.: Nonlinear Multiobjective Optimization. Springer, Cham (2012). https://doi.org/10.1007/978-1-4615-5563-6
31. Pang, L.M., Ishibuchi, H., Shang, K.: NSGA-II with simple modification works well on a wide variety of many-objective problems. IEEE Access **8**, 190240–190250 (2020)
32. Tian, Y., Cheng, R., Zhang, X., Jin, Y.: PlatEMO: a MATLAB platform for evolutionary multi-objective optimization [educational forum]. IEEE Comput. Intell. Mag. **12**(4), 73–87 (2017)
33. Wang, Z., Li, Q., Li, G., Zhang, Q.: Multi-objective decomposition evolutionary algorithm with objective modification-based dominance and external archive. Appl. Soft Comput. **149**, 111006 (2023)
34. Wang, Z., Li, Q., Yang, Q., Ishibuchi, H.: The dilemma between eliminating dominance-resistant solutions and preserving boundary solutions of extremely convex Pareto fronts. Complex Intell. Syst. **9**(2), 1117–1126 (2023)
35. Wang, Z., Ong, Y.S., Ishibuchi, H.: On scalable multiobjective test problems with hardly dominated boundaries. IEEE Trans. Evol. Comput. **23**(2), 217–231 (2018)
36. Wang, Z., Yao, S., Li, G., Zhang, Q.: Multiobjective combinatorial optimization using a single deep reinforcement learning model. IEEE Trans. Cybern. **54**(3), 1984–1996 (2024)
37. Wang, Z., Zhang, Q., Zhou, A., Gong, M., Jiao, L.: Adaptive replacement strategies for MOEA/D. IEEE Trans. Cybern. **46**(2), 474–486 (2015)
38. Wang, Z., et al.: Multiobjective optimization-aided decision-making system for large-scale manufacturing planning. IEEE Trans. Cybern. **52**(8), 8326–8339 (2021)
39. Wei, Z., Wang, H., Wang, S., Zhang, S., Xiao, D.: Complementary environmental selection for evolutionary many-objective optimization. In: Zhang, H., et al. (eds.) International Conference on Neural Computing for Advanced Applications, pp. 346–359. Springer, Singapore (2023). https://doi.org/10.1007/978-981-99-5844-3_25
40. Wei, Z., et al.: Many-objective evolutionary algorithm based on parallel distance for handling irregular pareto fronts. Swarm Evol. Comput. **86**, 101539 (2024)
41. Ye, R., Chen, L., Liao, W., Zhang, J., Ishibuchi, H.: Data-driven preference sampling for pareto front learning. arXiv preprint arXiv:2404.08397 (2024)
42. Ye, R., Chen, L., Zhang, J., Ishibuchi, H.: Evolutionary preference sampling for pareto set learning. arXiv preprint arXiv:2404.08414 (2024)
43. Ye, R., Tang, M.: PraFFL: a preference-aware scheme in fair federated learning. arXiv preprint arXiv:2404.08973 (2024)
44. Yuan, Y., Xu, H., Wang, B., Yao, X.: A new dominance relation-based evolutionary algorithm for many-objective optimization. IEEE Trans. Evol. Comput. **20**(1), 16–37 (2015)
45. Zhang, Q., Li, H.: MOEA/D: a multiobjective evolutionary algorithm based on decomposition. IEEE Trans. Evol. Comput. **11**(6), 712–731 (2007)
46. Zheng, R., Wang, Z.: A generalized scalarization method for evolutionary multi-objective optimization. In: Proceedings of the AAAI Conference on Artificial Intelligence (AAAI), vol. 37, pp. 12518–12525 (2023)
47. Zitzler, E., Thiele, L.: Multiobjective evolutionary algorithms: a comparative case study and the strength Pareto approach. IEEE Trans. Evol. Comput. **3**(4), 257–271 (1999)

# Hypervolume Gradient Subspace Approximation

Kenneth Zhang[1], Angel E. Rodriguez-Fernandez[2], Ke Shang[1,3(✉)], Hisao Ishibuchi[1], and Oliver Schütze[2(✉)]

[1] Guangdong Provincial Key Laboratory of Brain-Inspired Intelligent Computation, Department of Computer Science and Engineering, Southern University of Science and Technology, Shenzhen 518055, China
11913022@mail.sustech.edu.cn, hisao@sustech.edu.cn

[2] Computer Science Department, Cinvestav, Mexico City, Mexico
angel.rodriguez@cinvestav.mx, schuetze@cs.cinvestav.mx

[3] National Engineering Laboratory for Big Data System Computing Technology, Shenzhen University, Shenzhen 518060, China
kshang@foxmail.com

**Abstract.** Multi-objective evolutionary algorithms (MOEAs) are powerful optimizers that are capable of solving black-box multi-objective optimization problems. Due to their stochastic nature, local search methods, including directed search algorithms, have been proposed to guide search directions in the decision variable space. In particular, recent studies have shown that the inclusion of local hypervolume-based gradient methods can lead to better convergence rates. In this paper, a set-based method of estimating hypervolume gradients without additional function evaluations or Jacobian information is proposed and integrated with SMS-EMOA to form a steady-state MOEA. The proposed algorithm is compared to some widely-used MOEAs on two- and three-objective benchmark suites, outperforming all other algorithms on all 6/6 two-objective problems and 12/17 three-objective problems.

**Keywords:** Multi-objective optimization · Multi-objective evolutionary algorithm · Hypervolume indicator · Hypervolume gradients

## 1 Introduction

We focus on the domain of continuous box-constrained multi-objective optimization problems (MOP):

$$\min_{x \in Q} \; F(x), \tag{1}$$

K. Zhang and A. E. Rodriguez-Fernandez—These authors contributed equally to this work.


M. Affenzeller et al. (Eds.): PPSN 2024, LNCS 15151, pp. 20–35, 2024.
https://doi.org/10.1007/978-3-031-70085-9_2

where $Q \subset \mathbb{R}^n$ denotes the domain bounded by $l, u \in \mathbb{R}^n$, i.e., $Q = \{x \in \mathbb{R}^n \mid l_i \leq x_i \leq u_i,\ i = 1, \ldots, n\}$, and $F : Q \to \mathbb{R}^M$ is the map defined by the individual objectives $f_i : Q \to \mathbb{R}$, $i = 1, \ldots, M$, i.e., $F(x) = (f_1(x), \ldots, f_M(x))^T$.

Effective multi-objective evolutionary algorithms (MOEA) exist under conditions where $F$ might be non-differentiable or discontinuous. Three key categories of MOEAs underpin many current approaches [22]:

1. NSGA-II and SPEA2 represent a class of *dominance-based* MOEAs that favors solutions based on the Pareto criterion [5,25].
2. MOEA/D and similar *decomposition-based* MOEAs decompose multi-objective problems into single-objective problems [23].
3. *Indicator-based* MOEAs, like SMS-EMOA, focus on maximizing or minimizing indicators such as the hypervolume indicator [1].

One critical characteristic of MOEAs is their stochastic search process [7]. Local search methods that incorporate spatial information (from the vicinity of existing individuals) and temporal information (from previously archived populations) have been developed to guide the search direction of the decision variables [10,11,13,14]. One such implementation, called HIGA-MO, exploits hypervolume gradients by using them with a gradient ascent algorithm [20]. Hypervolume, defined as the volume in the objective space covered by the members of a solution set [26], is a metric of exceptional importance as it underscores both convergence and diversity of the solutions (i.e., it is Pareto-compliant) [6]. Steering solutions toward regions with higher hypervolume using the hypervolume gradients, then, offers an elegant approach to increase the solution quality [15]. However, precise computation of the hypervolume gradients may required finite difference methods, and therefore necessitates additional function evaluations.

As a remedy, we introduce a set-based method, HVGSA, which extracts information from historical populations to approximate the hypervolume gradients (Sect. 2). This eliminates the need for additional function evaluations and computing Jacobians. Then, HVGSA is integrated into a steady-state SMS-EMOA, blending the latter's stochastic search power with the former's deterministic convergence (Sect. 3). We show that the resulting HVGSA-EMOA generally outperforms widely-used algorithms for selected benchmarks (Sect. 4).

## 2 HVGSA

### 2.1 Naive Formulation

Gradient subspace approximation (GSA) is a set-based method that estimates the gradients of a multivariate scalar function [16,17,19]. In hypervolume GSA (HVGSA), the hypervolume indicator replaces said multivariate scalar function. The problem to select the $\mu$ best elements with respect to hypervolume can be expressed as the optimization problem

$$\max_{\substack{X \subset Q \\ |X| = \mu}} HV_{\text{set}}(X), \tag{2}$$

where $HV_{\text{set}}$ is a set function that maps a set to its hypervolume with respect to some reference point. We stress that considering a *set* $X = \{x^{(1)}, \ldots, x^{(\mu)}\} \subset \mathbb{R}^n$ of cardinality $\mu$ is the same as considering a *point* $\mathbf{X}$ in $\mathbb{R}^{\mu n}$. To see this, consider $\mathbf{X} = (x_1^{(1)}, \ldots, x_n^{(1)}, x_1^{(2)}, \ldots, x_n^{(2)}, \ldots, x_1^{(\mu)}, \ldots, x_n^{(\mu)})^T \in \mathbb{R}^{\mu n}$. Hereby, $x_j^{(i)}$ denotes the $j$-th coordinate of $x^{(i)}$. For example, the set $A = \{(1,2)^T, (3,4)^T\} \subset \mathbb{R}^2$ can be represented by the point $\mathbf{A} = (1,2,3,4)^T \in \mathbb{R}^4$.

Consequently, problem (2) can be viewed as a single-objective optimization problem of dimension $\mu n$ in the decision space. It may be expressed as

$$\max_{\mathbf{X} \in \mathbb{R}^{\mu n}} HV(\mathbf{X}), \tag{3}$$

where $HV : \mathbb{R}^{\mu n} \to \mathbb{R}$. Hereafter we stick with the point representation.

Assume we are given a point $\mathbf{X}_0 \in \mathbb{R}^{\mu n}$ designated for local search. In the context of MOEAs, $\mathbf{X}_0$ represents a population with $\mu$ individuals in an $n$-dimensional decision space. The ideal search direction would certainly be the direction of the steepest hypervolume ascent,

$$v^* := \frac{\nabla HV(\mathbf{X}_0)}{\|\nabla HV(\mathbf{X}_0)\|_2}, \tag{4}$$

which is the solution of

$$\max_{v \in \mathbb{R}^{\mu n}} \langle \nabla HV(\mathbf{X}_0), v \rangle \quad \text{s.t. } \|v\|_2^2 = 1. \tag{5}$$

Before establishing a gradient-free formulation, we must first elucidate how hypervolume gradients aid in computing $v^*$ from limited neighborhood information. In addition to hypervolume gradients, assume we are given $r$ points $\mathbf{X}_1, \ldots, \mathbf{X}_r \in \mathbb{R}^{\mu n}$ in the neighborhood of $\mathbf{X}_0$ (from historical populations), we can compute a best-fit approximation $v_S^*$ of $v^*$ within the $r$-dimensional subspace $S := span(v_1, \ldots, v_r)$, where

$$v_i := \frac{\mathbf{X}_i - \mathbf{X}_0}{\|\mathbf{X}_i - \mathbf{X}_0\|_2}, \quad i = 1, \ldots, r. \tag{6}$$

Then, for every $v \in S \subset \mathbb{R}^{\mu n}$ there exists a $\lambda \in \mathbb{R}^r$ such that $v = V\lambda$, where

$$V = (v_1, \ldots, v_r) \in \mathbb{R}^{\mu n \times r}. \tag{7}$$

Restricting problem (5) to $S$ (since we will only have access to the neighborhood information in Sect. 2.2), it can be expressed as

$$\max_{\lambda \in \mathbb{R}^r} \langle \nabla HV(\mathbf{X}_0), \sum_{i=1}^{r} \lambda_i v_i \rangle \quad \text{s.t. } \| \sum_{i=1}^{r} \lambda_i v_i \|_2^2 = 1. \tag{8}$$

Problem (8) can be solved in two steps: (i) computing the weight vector $\lambda$ via solving the following normal equation system

$$V^T V \lambda = V^T \nabla HV(\mathbf{X}_0), \tag{9}$$

and (ii) arriving at the approximate search direction $v_S^*$. A more detailed proof is supplied in [16], in which Schütze et al. considered the Karush-Kuhn-Tucker system of (8) and derived the normalized search direction as

$$v_S^* = \frac{V\lambda^*}{\|V\lambda^*\|_2}, \tag{10}$$

where $\lambda^*$ is the solution of (9).

### 2.2 Gradient-Free Realization

We now develop a gradient-free approach. Suppose in addition to $\mathbf{X}_1, \ldots, \mathbf{X}_r \in \mathbb{R}^{\mu n}$ in the neighborhood of $\mathbf{X}_0$, we are further supplied with their hypervolume values $HV(\mathbf{X}_0)$ and $HV(\mathbf{X}_i)$, $i = 1, \ldots, r$, we turn to the right hand side of (9) and note its $i$-th component

$$(V^T \nabla HV(\mathbf{X}_0))_i = \nabla HV(\mathbf{X}_0)^T v_i = \frac{HV(\mathbf{X}_i) - HV(\mathbf{X}_0)}{\|\mathbf{X}_i - \mathbf{X}_0\|_2} + O(\|\mathbf{X}_i - \mathbf{X}_0\|_2), \tag{11}$$

which follows from Taylor approximation. Hence, using

$$d_i := \frac{HV(\mathbf{X}_i) - HV(\mathbf{X}_0)}{\|\mathbf{X}_i - \mathbf{X}_0\|_2}, \quad i = 1, \ldots, r, \tag{12}$$

and $d = (d_1, \ldots, d_r)^T \in \mathbb{R}^r$, we can, instead of (9), solve the system

$$V^T V\lambda = d, \tag{13}$$

which is entirely accessible from the given neighborhood information. In particular, by solving (13) and (10), we may obtain $\lambda^{GSA} \in \mathbb{R}^r$ and $v^{GSA} \in \mathbb{R}^{\mu n}$, respectively, without any gradient information.

### 2.3 Gradient Ascent

Using the approximated search direction above, an iteration of HVGSA behaves as follows: given the current iterate $\tilde{\mathbf{X}}_j$ where the subscript $j$ in this case indicates the iteration count, the search direction $v_j^{GSA}$ obtained via (13) and (10) in this iteration is used to perform the gradient ascent algorithm:

$$\tilde{\mathbf{X}}_{j+1} = \tilde{\mathbf{X}}_j + \eta v_j^{GSA}, \tag{14}$$

where $\eta > 0$ is the step size.

Recall that $\tilde{\mathbf{X}}_j \in \mathbb{R}^{\mu n}$ is a vector representation of the actual population $X_j \subset \mathbb{R}^n$. We could similarly partition $v_j^{GSA}$ into $\mu$ elements of $n$-dimensional vectors, i.e. $v_j^{GSA} = (\nu_j^{(1)}, \ldots, \nu_j^{(\mu)})$, and update $X_j$ individual-wise. This yields:

$$x_{j+1}^{(i)} = x_j^{(i)} + \eta \nu_j^{(i)}, \quad i = 1, \ldots, \mu. \tag{15}$$

### 2.4 Examples

We illustrate the effect of HVGSA (14) on two examples. First, we consider HVGSA as a standalone algorithm. The problem CONV2 given by $F : Q \to \mathbb{R}^2$

$$F(x) = \begin{pmatrix} x_1^2 + x_2^2 \\ (x_1 - 10)^2 + x_2^2 \end{pmatrix}, \tag{16}$$

where $Q = [0, 10] \times [-5, 5]$ has as Pareto set the line segment connecting $(0, 0)^T$ and $(10, 0)^T$. We set $\mu = 5$ and select the initial population (point) $\tilde{\mathbf{X}}_0 = \mathbf{X}_0 = \{a_0^{(1)}, \ldots, a_0^{(5)}\}$ randomly within $Q$. The dimension of (3) is hence $\mu n = 2 \cdot 5 = 10$.

As a standalone algorithm, we compute artificial neighborhood information $\mathbf{X}_1, \ldots, \mathbf{X}_r$ by sampling $x_i^{(j)}$ in the vicinity of $x_0^{(j)}$, $i = 1, \ldots, r$, $j = 1, \ldots, \mu$. Then, from (14) we compute the next population $\tilde{\mathbf{X}}_1$ with a small step size $\eta$. We repeat this process until the difference of the norm of two consecutive populations goes below a predefined threshold.

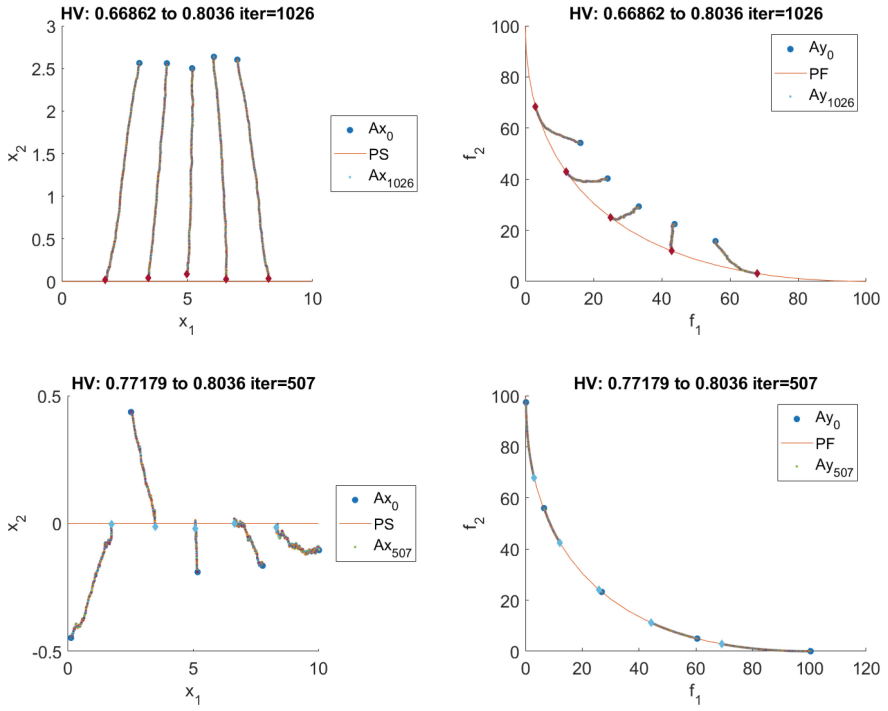



**Fig. 1.** Application of HVGSA on CONV2 using two initial populations. Above: $\mathbf{X}_0$ is "far" from the Pareto set. Below: $\mathbf{X}_0$ is already "close" to the Pareto set.

Figure 1 shows a numerical result of HVGSA on two such trajectories using $r = 5$. In both cases, the populations move toward the optimal 5-element HV

approximation set for this problem (using the reference point $R = (101, 101)^T$). Through sampling, each iteration costs $r$ HV evaluations and $r\mu n$ function evaluations. This could be significantly improved by utilizing information previously computed within an MOEA, as shown in the next example.

Consider the test problem ZDT2 using $n = 30$ decision variables. As a starting point we use the 50-th population of SMS-EMOA from an arbitrary run. With a population size of $\mu = 100$, the dimension of (3) is hence $\mu n = 3000$. First, we apply (14) using the 50-th population ($\tilde{\mathbf{X}}_{50} = \mathbf{X}_0$, recall that the subscript for $\tilde{\mathbf{X}}$ represents the iteration count) together with the previous $r = 10$ populations ($\tilde{\mathbf{X}}_{49} = \mathbf{X}_1, \ldots, \tilde{\mathbf{X}}_{40} = \mathbf{X}_{10}$). Using (14) and the fixed step size $\eta = 0.05$, we may compute $\tilde{\mathbf{X}}_{51}$. Furthermore, the generated set is again stored as neighborhood information, and therefore we can use $\tilde{\mathbf{X}}_{51} = \mathbf{X}_0, \tilde{\mathbf{X}}_{50} = \mathbf{X}_1, \ldots, \tilde{\mathbf{X}}_{41} = \mathbf{X}_{10}$ and again compute a new population using this setting.

Figure 2 shows the results for this approach after 10 and 30 iterations. Due to internal storage of the function values and the hypervolume values, these results come with a cost of 10 and 30 HV evaluations, and no additional function evaluations apart from the newly generated set are required. Had the HV gradients been approximated using the forward difference quotient, each step would have required 3,000 HV evaluations, leading to 30,000 and 90,000 HV evaluations with the same number of steps. While the use of actual gradients leads to (slightly) better results, it comes with an impractical amount of computational cost.

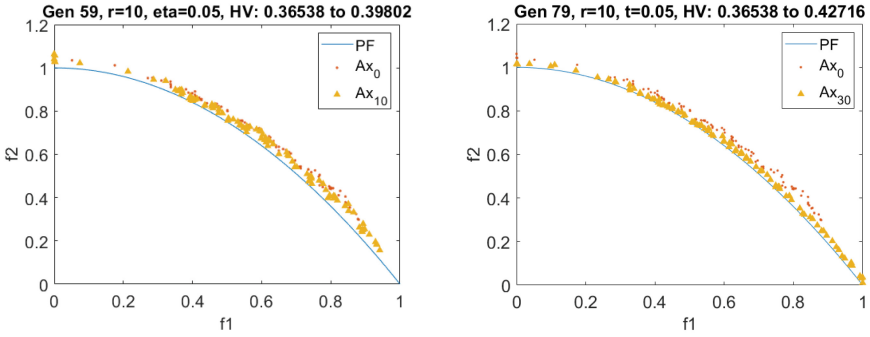



**Fig. 2.** Application of the HVGSA approach on a population obtained by SMS-EMOA on the problem ZDT2. Shown are the results after 10 and 30 iterations using the fixed step size $\eta = 0.05$.

## 3 HVGSA-EMOA

Efficient as it may be, HVGSA on its own lacks the ability to escape local extrema and to precisely locate ideal solutions. As a result, its performance on benchmark problems is incomparable to the widely-used MOEAs. To address these issues, we propose HVGSA-EMOA. We outline its three components:

1. Team selection: a subset of elite individuals is selected from the population based on their hypervolume contributions. This ensures the search only focuses on the most promising solutions.
2. HVGSA employment: the selected team undergoes HVGSA, where we gather neighborhood information from an external archive, and through a combinatorial trick we take advantage of limited neighborhood information.
3. Environmental selection: A reduction algorithm is applied to the generated solutions from both the HVGSA and the evolutionary steps.

**Algorithm 1:** HVGSA-EMOA
A more detailed pseudocode as well as the source code are provided in https://github.com/HisaoLabSUSTC/HVGSA-PPSN2024

```
Function HVGSA-EMOA(Algorithm, Problem)
  /* Part 0: parameter specification                              */
(r, k, η, teamCap, rest) ← (50, 10, 50, 5, 10)
Population ← Problem.Initialization()
Archive.Store(Population)
ref ← Max(Population) × 1.1
while Algorithm.notTerminated() do
    /* Part 1: team selection                                    */
    team ← TeamSelection(Population)
    /* Part 2: HVGSA                                            */
    team ← HVGSA(team, rest, Archive)
    /* Part 3: environmental selection                          */
    for i = 1 to Size(team) do
        Population ← Reduce(Population ∪ {team_i})
    end
    for i = 1 to Problem.N − Size(team) do
        Offspring ← GenerateOne(Population)
        Archive.Store(Offspring)
        Population ← Reduce(Population ∪ {Offspring})
    end
    ref ← max(Population) × 1.1
end
```

### 3.1 Team Selection

In Sect. 2.4, when we considered the test problem ZDT2 with $n = 30$, $\mu = 100$ and $r = 10$, the search dimension of (3) reaches a staggering 3000. Now consider the linear system (13), we know from (7) that the dimension of $V$ in this case is $3000 \times 10$. Two concerns are immediate: i) approximating the best direction in a 3000-dimensional space with a 10-dimensional subspace can be a drastic

undershot, and ii) the matrix multiplication $V^T V$ can be very inefficient due to the large size of the search dimension.

We resolve these two concerns by selecting a team that is smaller than the entire population. In this paper, this team size is capped at $\texttt{teamCap} = 5$. The resulting search dimension of ZDT2 is now 150 or below.

Following the idea proposed in [11] where elite solutions are selected for the local search step, we derive an elitist criterion for the nondominated front. We assign a probability $p_{\text{team}}$ to each individual within the population $P$:

$$p_{\text{team}}(x_i) = \begin{cases} 0 & \text{if } x_i \in P_{\text{dominated}} \\ \frac{\exp(-\lambda \cdot \text{rank}(x_i))}{|P_{\text{nondom}}|} & \text{if } x_i \in P_{\text{nondom}} \end{cases}, \quad i = 1, \ldots, \mu, \tag{17}$$

where $p_{\text{team}}$ is the probability of $x_i$ to be chosen as a search team member. If $x_i \in P_{\text{dominated}}$, then it should not be considered an elite individual fit for local search. If, on the other hand, $x_i \in P_{\text{nondom}}$, then it should be assigned a nonzero probability following a discretized exponential distribution based on its hypervolume contribution ranking $\texttt{rank}(x_i)$. In our algorithm, $\lambda = 1$, and the denominator $|P_{\text{nondom}}|$ is used as a normalization constant. With each solution selection, the probability distribution is renormalized. The most likely configuration is naturally the search team composed of the $\texttt{teamCap} = 5$ individuals with the highest hypervolume contribution.

Notice that the size of the search team can be less than `teamCap` if the nondominated front doesn't have enough solutions to fill the search team.

Let us summarize the team selection step:

1. Team size limitation: computational efficiency is improved with smaller teams, facilitating rapid HVGSA computations.
2. Elite team composition: top hypervolume contributors within the nondominated front are selected for local search, enhancing search effectiveness.

### 3.2 HVGSA Employment

With an elite team, our search space shrinks significantly: $\mu_{\text{team}} n \ll \mu_P n$. We now revise how neighborhood information is obtained. Previously, we used $r$ closest populations. A better method is to use an archive that stores all the solutions that cost function evaluations (this includes the initial population, the solutions generated from HVGSA, and the offsprings in the evolutionary step).

For pragmatic reasons, we bound our archive with at most $Mn$ solutions, where $M$ is the dimension of the objective space. When this limit is reached, new solutions replaces old solutions as such:

1. If the new solution dominates any old solution, the old solution is discarded and the new solution is stored.
2. Otherwise, the new solution(s) is temporarily stored. At the end of one `Store()` call, we compute the pairwise distance of all points in the archive. We truncate points with the smallest distances to their nearest neighbors.

Recall that we reduced the search dimension of ZDT2 from 3000 to 150 in Sect. 3.1. Even so, a 10-dimensional subspace remains insufficient for estimating hypervolume gradients. Referring to the search team as $\mathbf{X}_0$ and its $i$-th member as $\text{team}_i, i = 1, \ldots, \texttt{teamCap}$, we introduce two parameters:

1. $k = 10$ ($k$-nearest neighbors): we find the $k$ closest points $\text{team}_i^{(j)}, j = 1 \ldots, k$, in the archive for each member of the search team. If we have a team size of 5, this gives us a maximum of 50 individuals selected within the archive. These neighbors appear as non-red filled circles in Fig. 3. $k$ should not be too large, for then nearness and estimation precision is not guaranteed.
2. $r = 50$ (permutation selection): instead of simply taking all the closest neighbors to each team member ($\mathbf{X}_1 := (\text{team}_1^{(1)}, \ldots, \text{team}_5^{(1)})^T$) as a neighbor of $\mathbf{X}_0$, all the second-closest neighbors to each team member ($\mathbf{X}_2 := (\text{team}_1^{(2)}, \ldots, \text{team}_5^{(2)})^T$) as the next neighbor, etc. We can choose any permutation of those $k$ neighbors. There are $\mu_{\text{team}}^k$ such choices, and $r = 50$ of them are selected for HVGSA. In Fig. 3, the left image shows a permutation (blue filled circles) where the neighbor is $(\text{team}_1^{(1)}, \ldots, \text{team}_5^{(1)})^T$. On the right, only one individual is changed ($\text{team}_2^{(2)}$), but the resulting vector points in a different direction in the search space of (3).

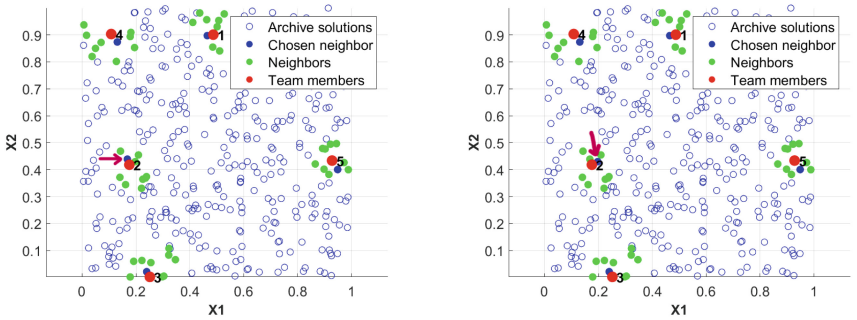



**Fig. 3.** Illustration of 10-NN on 5 individuals within a 2-dimensional decision space. Left: selecting the indices [1, 1, 1, 1, 1] in terms of closeness as the neighbor. Right: selecting the indices [1, 2, 1, 1, 1].

Going back to the ZDT2 example, we now have a 50-dimensional subspace within an 150-dimensional search space. This gives a higher confidence of the gradient approximation. With such a local search algorithm, we next consider its share of resource with respect to the evolutionary steps [8,11]. Two design problems are to be addressed: i) the frequency of the local search step, and ii) the choice between a small or large step size.

An ablation study is performed and an optimal configuration is found to invoke 1 HVGSA step every $\texttt{rest} = 10$ evolutionary steps, together with the usage of a large learning step $\eta = 50$. The reasons are twofold:

1. Every HVGSA step updates at most `teamCap` = 5 individuals at once. That is, only 5 function evaluations are required every 10 iterations. When $\mu = 100$ and $FE = 10000$, 100 evolutionary steps will be performed, which invokes 10 HVGSA steps and a total of 50 extra function evaluations.
2. We justify the large step size ($\eta = 50$) with the following considerations: i) in the early stages of MOEAs, it can rapidly drive the search team toward the Pareto front, significantly improving convergence (as demonstrated in Sect. 4), and ii) in the worst-case scenario, 50 wasted function evaluations are unlikely to significantly harm the performance of the underlying EMOA.

### 3.3 Environmental Selection

The underlying EMOA, then, is of crucial importance since it directly determines the performance of HVGSA-EMOA. We choose SMS-EMOA because of its strong performance, alignment with the usage of hypervolume, and its steady-state property.

MOEAs possess a steady-state property when they generate and evaluate one offspring at a time. This guarantees non-decreasing quality of solution. In SMS-EMOA, this offspring is compared to the existing population based on its hypervolume contribution. The individual with the least hypervolume contribution is then discarded by calling `Reduce()` (see Algorithm 2). It sorts the augmented population into nondominated fronts, and eliminates the individual with the least exclusive hypervolume contribution in the last front.

This ensures improvement in the overall hypervolume. Therefore, we apply the `Reduce()` function to each member of the search team after the HVGSA step as well. This corresponds to line 9–11 in Algorithm 1. Since the team is smaller than the needed population, we populate the next generation with SMS-EMOA's steady-state offspring generation (line 12-16 of Algorithm 1).

**Algorithm 2:** Reduce(Population)

```
1 Function Reduce(P)
2 {Front_1, ..., Front_v} ← fast-nondominated-sort(P)
3 worst ← argmin_{s∈Front_v} [Δ_𝒮 (s, Front_v)]
4 return (P\{worst})
```

Here, $\Delta_{\mathscr{S}}(s, \text{Front}_v) := HV(\text{Front}_v) - HV(\text{Front}_v \backslash \{s\})$.

*We observed an unexpected increase in performance on certain problems due to an implementation issue where solutions in the search team undergo `Reduce()` despite the HVGSA step not being invoked. This results an initial duplication of nondominated solutions with high hypervolume contributions (otherwise it couldn't be selected for HVGSA). In PlatEMO, repeated solutions both have zero hypervolume contribution and will be subsequently eliminated during the next environmental selection if no dominated solutions exist. We have decided to keep this design as the added selection pressure makes convergence faster.

## 4 Experimental Results

Complete results are displayed as supplementary materials at our GitHub page. We evaluated HVGSA-EMOA on established test problems and compared it against widely-used MOEAs using the average hypervolume as well as average hypervolume progression across 50 runs. Specifically, our methodology involves a population size of 100, 10000 function evaluations per run, and data aggregation across 50 independent runs with the hypervolume progression metrics captured at 20 evenly-spaced intervals across the 10000 function evaluations.

We implemented HVGSA-EMOA on the PlatEMO platform [18] and benchmarked it against established algorithms (NSGA-II, NSGA-III, SMS-EMOA, MOEA/D, SPEA2) available within PlatEMO. In all experiments, MOEA/D uses the Tchebycheff decomposition method. We replaced the original hypervolume calculation with the WFG algorithm [21] from the STK toolbox for enhanced contribution calculations and addressed a minor boundary solution bias in the reference point calculation. We used the hypervolume metric for performance evaluation due to its effectiveness in quantifying solution quality.

### 4.1 2D Problems

For 2D problems, we tested our algorithm on the ZDT test suite, which was first introduced in [24]. HVGSA-EMOA demonstrated superior hypervolume convergence on all ZDT problems (excluding ZDT4) and achieved the highest average hypervolume on ZDT4 despite the initial slow convergence (Table 1, Fig. 4). A large step size caused the HVGSA step to drastically improve our algorithm's performance in the early stages of the problem.

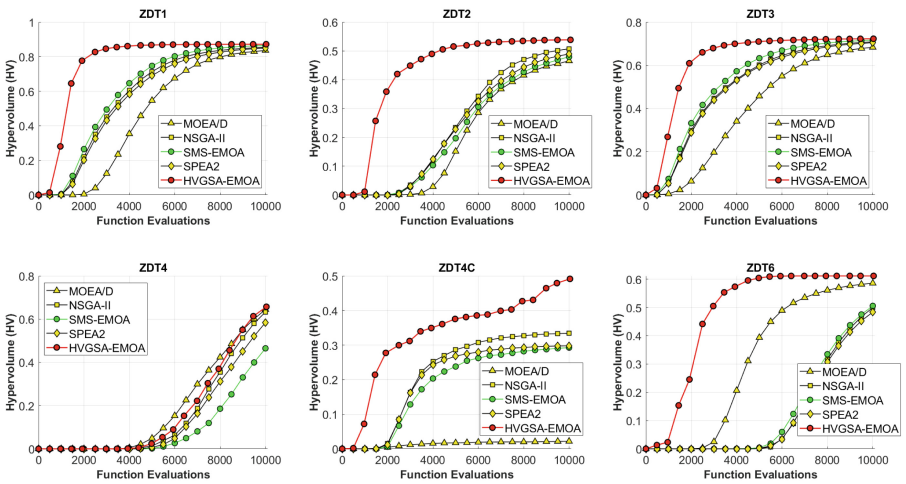



**Fig. 4.** Hypervolume convergence plot (averaged over 50 runs) against function evaluations for the ZDT test suite.

**Table 1.** Average hypervolume aggregated over 50 runs on 2D test suites (parenthesis: standard deviation; highlight: best).

| Problem | M | D | NSGA-II | MOEA/D | SPEA2 | SMS-EMOA | HVGSA-EMOA |
|---|---|---|---|---|---|---|---|
| ZDT1 | 2 | 30 | 8.5413e−1 (3.54e−3) - | 8.3604e−1 (1.38e−2) - | 8.4991e−1 (4.61e−3) - | 8.6440e−1 (2.30e−3) - | **8.7176e−1 (8.94e−4)** |
| ZDT2 | 2 | 30 | 5.0740e−1 (3.03e−2) - | 4.6550e−1 (7.40e−2) - | 4.8923e−1 (6.02e−2) - | 4.7942e−1 (7.21e−2) - | **5.3836e−1 (1.16e−3)** |
| ZDT3 | 2 | 30 | 7.0851e−1 (6.63e−3) - | 6.8321e−1 (2.07e−2) - | 7.0463e−1 (6.96e−3) - | 7.1628e−1 (8.09e−3) - | **7.2276e−1 (7.90e−3)** |
| ZDT4 | 2 | 10 | 6.3339e−1 (1.59e−1) = | 6.4461e−1 (9.88e−2) = | 5.8441e−1 (1.86e−1) - | 4.6501e−1 (2.15e−1) - | **6.5725e−1 (1.57e−1)** |
| ZDT4C | 2 | 10 | 3.3407e−1 (2.23e−1) - | 2.1720e−2 (7.93e−2) - | 2.9770e−1 (2.12e−1) - | 2.9267e−1 (2.03e−1) - | **4.9145e−1 (3.40e−1)** |
| ZDT6 | 2 | 10 | 4.9964e−1 (5.78e−2) - | 5.8577e−1 (7.04e−3) - | 4.8347e−1 (5.33e−2) - | 5.0550e−1 (4.29e−2) - | **6.1144e−1 (2.99e−5)** |
| +/-/ = | | | 0/5/1 | 0/5/1 | 0/6/0 | 0/6/0 | |

We analyzed ZDT4's unique challenges and proposed a simplified problem, ZDT4C. In the original ZDT4 problem, the variable bounds are $x_1 \in [0, 1]$, $x_i \in [-5, 5], i = 2, \ldots, n = 30$. ZDT4C solves a smaller problem where the variables are bounded in the unit hypercube, i.e. $x_i \in [0, 1], i = 1, \ldots, n = 30$. It is surprising to see that conventional MOEAs struggle to solve the simplified ZDT4C, yet performs quite well on the ZDT4 problem. This is due to the way polynomial mutation is implemented. Described in the appendix of [4], the polynomial mutation dynamically scales its search power based on the size of the search space. Simulated binary crossover (SBX), introduced in [3], exhibits a similar scaling behavior. Both operators, adapted from their binary counterparts, try to impose a probabilistic search behavior based on the available search space. Inspired by this observation, we implemented the `rest` parameter (explained at the end of Sect. 3.2) to strategically allocate more resource toward evolving around local regions following a gradient-based directional boost. On multimodal problems, gradient steps often become ineffective, leading to search stagnation in local extrema. This necessitates a greater reliance on evolutionary mechanisms.

### 4.2 3D Problems

We evaluated our algorithm on 3D problems using the DTLZ, IDTLZ, and WFG test suites [2,9,12]. In higher-dimensional spaces, the elitist property of our algorithm becomes less significant due to the increasing prevalence of nondominated solutions. Additionally, our fixed step size ($\eta = 50$) likely reduced optimal convergence in this context. Consequently, the striking performance gains observed with ZDT problems were less frequent. Still, our algorithm reached the best result at the end of 10000 function evaluations in 12 (out of 17) problems. These findings suggest the potential benefits of an adaptive step-size implementation and a more robust dominance strategy.

In Fig. 5, we have selected some distinct problems (see Table 2) where HVGSA-EMOA performs better (top row) and performs on par (bottom row) with the other algorithms. A discernible pattern of problems that HVGSA-EMOA performs well on is not immediately clear, but in future work we will continue to monitor and improve the performance of our algorithm on higher-dimensional problems.

**Table 2.** Average hypervolume aggregated over 50 runs on 3D test problems (parenthesis: standard deviation; highlight: best).

| Problem | M | D | NSGA-III | MOEA/D | SPEA2 | SMS-EMOA | HVGSA-EMOA |
|---|---|---|---|---|---|---|---|
| DTLZ1 | 3 | 7 | 6.8257e−1 (4.19e−1) = | **7.1124e−1 (4.02e−1)** = | 6.5528e−1 (4.47e−1) = | 6.4943e−1 (4.49e−1) = | 6.5869e−1 (4.56e−1) |
| DTLZ2 | 3 | 12 | 7.3996e−1 (9.29e−4) - | 7.0500e−1 (2.28e−3) - | 7.3292e−1 (2.20e−3) - | 7.5473e−1 (1.14e−4) = | **7.5474e−1 (1.41e−4)** |
| DTLZ4 | 3 | 12 | 6.4601e−1 (1.64e−1) - | 4.3733e−1 (2.27e−1) - | 5.8635e−1 (1.77e−1) - | 5.9196e−1 (1.77e−1) - | **7.5468e−1 (1.59e−4)** |
| DTLZ5 | 3 | 12 | 2.5744e−1 (1.42e−3) - | 2.5928e−1 (1.44e−4) - | 2.6387e−1 (4.44e−4) - | 2.6651e−1 (2.64e−5) = | **2.6652e−1 (2.69e−5)** |
| DTLZ6 | 3 | 12 | 2.4957e−1 (3.61e−2) - | 2.5857e−1 (8.85e−3) - | 2.6055e−1 (3.76e−2) - | 2.6658e−1 (8.05e−5) - | **2.6660e−1 (9.89e−6)** |
| DTLZ7 | 3 | 22 | 4.8990e−1 (1.92e−2) - | 4.5235e−1 (1.10e−2) - | 5.2100e−1 (7.99e−3) - | 5.5082e−1 (4.42e−2) - | **5.7125e−1 (1.96e−2)** |
| IDTLZ1 | 3 | 7 | 1.0054e−1 (9.70e−2) = | **1.5104e−1 (8.97e−2)** = | 1.4528e−1 (1.12e−1) = | 1.2318e−1 (1.13e−1) = | 1.3235e−1 (1.16e−1) |
| IDTLZ2 | 3 | 12 | 6.8599e−1 (4.10e−3) - | 6.8611e−1 (1.03e−3) - | 6.9895e−1 (2.08e−3) - | **7.1662e−1 (1.72e−3)** + | 7.1493e−1 (3.00e−3) |
| WFG1 | 3 | 12 | 8.6217e−1 (4.90e−2) - | 1.0242e +0 (6.82e−2) = | 9.2936e−1 (6.99e−2) - | 1.0299e +0 (3.40e−2) = | **1.0354e +0 (4.76e−2)** |
| WFG2 | 3 | 12 | 1.2140e +0 (6.22e−3) - | 1.1428e +0 (3.24e−2) - | 1.2107e +0 (2.89e−2) - | 1.2417e +0 (1.99e−2) = | **1.2437e +0 (4.02e−3)** |
| WFG3 | 3 | 12 | 4.7365e−1 (1.09e−2) - | 4.7338e−1 (2.10e−2) - | 4.8239e−1 (9.67e−3) - | **5.3072e−1 (6.45e−3)** = | 5.3010e−1 (5.91e−3) |
| WFG4 | 3 | 12 | 7.0645e−1 (4.75e−3) - | 6.7092e−1 (6.17e−3) - | 6.9563e−1 (5.19e−3) - | **7.4539e−1 (1.92e−3)** = | 7.4516e−1 (1.69e−3) |
| WFG5 | 3 | 12 | 6.7248e−1 (4.89e−3) - | 6.1147e−1 (5.37e−3) - | 6.6364e−1 (6.10e−3) - | 6.9648e−1 (4.28e−3) - | **6.9780e−1 (3.49e−3)** |
| WFG6 | 3 | 12 | 6.3361e−1 (1.78e−2) - | 5.9302e−1 (3.12e−2) - | 6.3351e−1 (1.97e−2) - | 6.7112e−1 (1.74e−2) = | **6.7504e−1 (2.03e−2)** |
| WFG7 | 3 | 12 | 7.1051e−1 (3.67e−3) - | 6.7461e−1 (7.43e−3) - | 7.0756e−1 (4.04e−3) - | 7.5404e−1 (7.91e−4) = | **7.5413e−1 (6.10e−4)** |
| WFG8 | 3 | 12 | 5.8913e−1 (5.21e−3) - | 5.6240e−1 (1.25e−2) - | 5.8674e−1 (5.14e−3) - | 6.3020e−1 (3.58e−3) = | **6.3160e−1 (3.16e−3)** |
| WFG9 | 3 | 12 | 6.7323e−1 (1.00e−2) - | 6.1065e−1 (4.23e−2) - | 6.6516e−1 (9.06e−3) - | 7.2037e−1 (4.17e−3) - | **7.2101e−1 (1.96e−2)** |
| +// = | | | 0/15/2 | 0/14/3 | 0/15/2 | 1/5/11 | |

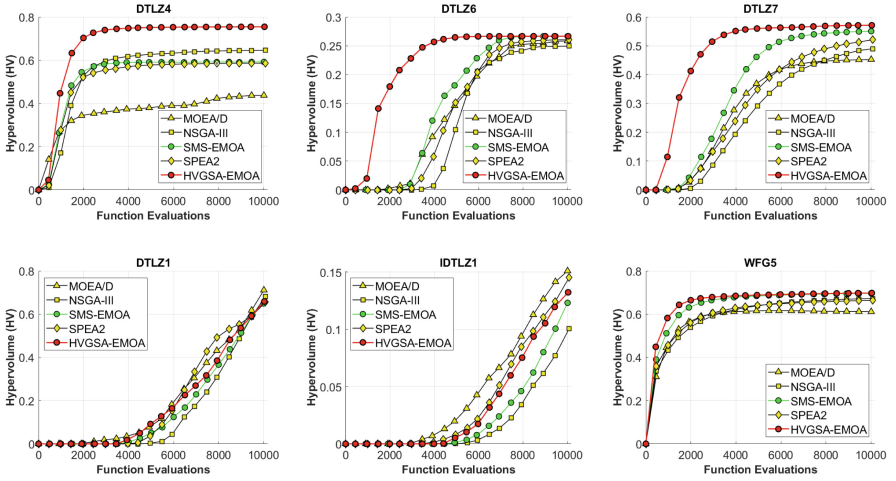



**Fig. 5.** Selected hypervolume convergence plot (averaged over 50 runs) against function evaluations for 3D problems. Above: HVGSA performs well on those problems. Below: HVGSA performs poorly on those problems, but still manages to increase the performance of SMS-EMOA.

## 5 Conclusions and Future Work

In this work, we have proposed HVSGA, a set-based method to estimate hypervolume gradients without using additional function evaluations or Jacobian information. We have further combined HVGSA with SMS-EMOA to form HVGSA-EMOA, and observed a significant improvement in the convergence rate on benchmarked two- and three-objective problems. In particular, when comput-

ing hypervolume gradients, HVGSA uses almost no extra function evaluations, whereas a forward difference method would require an impractical amount. Based on these observations, we are thrilled about HVGSA's potential and are actively working on constraint handling and on generalizing HVGSA-EMOA to many-objective problems.

Our source code of HVGSA-EMOA as well as supplementary materials are available at: https://github.com/HisaoLabSUSTC/HVGSA-PPSN2024.

**Acknowledgments.** This work was supported by National Natural Science Foundation of China (Grant No. 62250710163, 62376115), Guangdong Provincial Key Laboratory (Grant No. 2020B121201001), and Conahcyt project CBF2023-2024-1463.
**Disclosure of Interests.** The authors have no competing interests to declare that are relevant to the content of this article.

## References

1. Beume, N., Naujoks, B., Emmerich, M.: SMS-EMOA: multiobjective selection based on dominated hypervolume. Eur. J. Oper. Res. **181**(3), 1653–1669 (2007). https://doi.org/10.1016/j.ejor.2006.08.008. https://linkinghub.elsevier.com/retrieve/pii/S0377221706005443
2. Deb, K., Thiele, L., Laumanns, M., Zitzler, E.: Scalable multi-objective optimization test problems. In: Proceedings of the 2002 Congress on Evolutionary Computation, CEC 2002 (Cat. No.02TH8600), vol. 1, pp. 825–830 (2002). https://doi.org/10.1109/CEC.2002.1007032
3. Deb, K., Agrawal, R.B.: Simulated binary crossover for continuous search space. Complex Syst. **9** (1995). https://api.semanticscholar.org/CorpusID:18860538
4. Deb, K., Agrawal, S.: A niched-penalty approach for constraint handling in genetic algorithms. In: Artificial Neural Nets and Genetic Algorithms, pp. 235–243. Springer, Vienna (1999). https://doi.org/10.1007/978-3-7091-6384-9_40
5. Deb, K., Agrawal, S., Pratap, A., Meyarivan, T.: A fast elitist non-dominated sorting genetic algorithm for multi-objective optimization: NSGA-II. In: Schoenauer, M., et al. (eds.) PPSN 2000. LNCS, vol. 1917, pp. 849–858. Springer, Heidelberg (2000). https://doi.org/10.1007/3-540-45356-3_83
6. Fleischer, M.: The measure of pareto optima applications to multi-objective metaheuristics. In: Fonseca, C.M., Fleming, P.J., Zitzler, E., Thiele, L., Deb, K. (eds.) EMO 2003. LNCS, vol. 2632, pp. 519–533. Springer, Heidelberg (2003). https://doi.org/10.1007/3-540-36970-8_37
7. Goldberg, D.E.: Genetic Algorithms in Search, Optimization, and Machine Learning. Addison-Wesley Publishing Co., Reading, Mass (1989)
8. Ha, D.M.F., Deist, T.M., Bosman, P.A.N.: Hybridizing hypervolume-based evolutionary algorithms and gradient descent by dynamic resource allocation. In: Rudolph, G., Kononova, A.V., Aguirre, H., Kerschke, P., Ochoa, G., Tušar, T. (eds.) Parallel Problem Solving from Nature - PPSN XVII, pp. 179–192. Springer, Cham (2022). https://doi.org/10.1007/978-3-031-14721-0_13
9. Huband, S., Barone, L., While, L., Hingston, P.: A scalable multi-objective test problem toolkit. In: Coello Coello, C.A., Hernández Aguirre, A., Zitzler, E. (eds.) EMO 2005. LNCS, vol. 3410, pp. 280–295. Springer, Heidelberg (2005). https://doi.org/10.1007/978-3-540-31880-4_20

10. Ishibuchi, H., Murata, T.: A multi-objective genetic local search algorithm and its application to flowshop scheduling. IEEE Trans. Syst. Man Cybern. Part C (Appl. Rev.) **28**(3), 392–403 (1998). https://doi.org/10.1109/5326.704576
11. Ishibuchi, H., Yoshida, T., Murata, T.: Balance between genetic search and local search in memetic algorithms for multiobjective permutation flowshop scheduling. IEEE Trans. Evol. Comput. **7**(2), 204–223 (2003). https://doi.org/10.1109/TEVC.2003.810752
12. Ishibuchi, H., Setoguchi, Y., Masuda, H., Nojima, Y.: Performance of decomposition-based many-objective algorithms strongly depends on pareto front shapes. IEEE Trans. Evol. Comput. **21**(2), 169–190 (2017). https://doi.org/10.1109/TEVC.2016.2587749
13. Krasnogor, N., Smith, J.: A memetic algorithm with self-adaptive local search: TSP as a case study. In: Proceedings of the Genetic and Evolutionary Computation Conference (GECCO-2000), May 2000
14. Merz, P., Freisleben, B.: Genetic local search for the TSP: new results. In: Proceedings of 1997 IEEE International Conference on Evolutionary Computation (ICEC 1997), pp. 159–164. IEEE, Indianapolis, IN, USA (1997). https://doi.org/10.1109/ICEC.1997.592288. http://ieeexplore.ieee.org/document/592288/
15. Schütze, O., Sosa Hernández, V.A., Trautmann, H., Rudolph, G.: The hypervolume based directed search method for multi-objective optimization problems. J. Heuristics **22**, 273–300 (2016)
16. Schütze, O., Alvarado, S., Segura, C., Landa, R.: Gradient subspace approximation: a direct search method for memetic computing. Soft. Comput. **21**(21), 6331–6350 (2017). https://doi.org/10.1007/s00500-016-2187-x
17. Schütze, O., Uribe, L., Lara, A.: The gradient subspace approximation and its application to bi-objective optimization problems. In: Junge, O., Schütze, O., Froyland, G., Ober-Blöbaum, S., Padberg-Gehle, K. (eds.) SON 2020. SSDC, vol. 304, pp. 355–390. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-51264-4_15
18. Tian, Y., Cheng, R., Zhang, X., Jin, Y.: PlatEMO: a MATLAB platform for evolutionary multi-objective optimization [educational forum]. IEEE Comput. Intell. Mag. **12**(4), 73–87 (2017). https://doi.org/10.1109/MCI.2017.2742868
19. Uribe, L., Lara, A., Deb, K., Schütze, O.: A new gradient free local search mechanism for constrained multi-objective optimization problems. Swarm Evol. Comput. **67**, 100938 (2021). https://doi.org/10.1016/j.swevo.2021.100938. https://www.sciencedirect.com/science/article/pii/S2210650221000997
20. Wang, H., Deutz, A., Bäck, T., Emmerich, M.: Hypervolume indicator gradient ascent multi-objective optimization. In: Trautmann, H., et al. (eds.) EMO 2017. LNCS, vol. 10173, pp. 654–669. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-54157-0_44
21. While, L., Bradstreet, L., Barone, L.: A fast way of calculating exact hypervolumes. IEEE Trans. Evol. Comput. **16**(1), 86–95 (2012). https://doi.org/10.1109/TEVC.2010.2077298
22. Zhang, J., Xing, L.: A survey of multiobjective evolutionary algorithms. In: 22017 IEEE International Conference on Computational Science and Engineering (CSE) and IEEE International Conference on Embedded and Ubiquitous Computing (EUC), pp. 93–100. IEEE, Guangzhou, China, July 2017. https://doi.org/10.1109/CSE-EUC.2017.27. http://ieeexplore.ieee.org/document/8005779/
23. Zhang, Q., Li, H.: MOEA/D: a multiobjective evolutionary algorithm based on decomposition. IEEE Trans. Evol. Comput. **11**(6), 712–731 (2007). https://doi.org/10.1109/TEVC.2007.892759. http://ieeexplore.ieee.org/document/4358754/

24. Zitzler, E., Deb, K., Thiele, L.: Comparison of multiobjective evolutionary algorithms: empirical results. Evol. Comput. **8**(2), 173–195 (2000). https://doi.org/10.1162/106365600568202
25. Zitzler, E., Laumanns, M., Thiele, L.: SPEA2: improving the strength pareto evolutionary algorithm. Technical report, [object Object], May 2001. https://doi.org/10.3929/ETHZ-A-004284029. http://hdl.handle.net/20.500.11850/145755. Artwork Size: 21 p. Medium: application/pdf
26. Zitzler, E., Thiele, L.: Multiobjective optimization using evolutionary algorithms—a comparative case study. In: Eiben, A.E., Bäck, T., Schoenauer, M., Schwefel, H.-P. (eds.) PPSN 1998. LNCS, vol. 1498, pp. 292–301. Springer, Heidelberg (1998). https://doi.org/10.1007/BFb0056872

# LTR-HSS: A Learning-to-Rank Based Framework for Hypervolume Subset Selection

Cheng Gong[1,2,3], Ping Guo[1,3], Tianye Shu[2], Qingfu Zhang[1,3(✉)], and Hisao Ishibuchi[2(✉)]

[1] City University of Hong Kong, Hong Kong, China
{chenggong6-c,pingguo5-c}@my.cityu.edu.hk
[2] Southern University of Science and Technology, Shenzhen 518055, China
12132356@mail.sustech.edu.cn, hisao@sustech.edu.cn
[3] The City University of Hong Kong Shenzhen Research Institute, Shenzhen, China
qingfu.zhang@cityu.edu.hk

**Abstract.** Hypervolume subset selection (HSS) plays an important role in various aspects of the field of evolutionary multi-objective optimization, such as environmental selection and post-processing for decision-making. The goal of these problems is to find the optimal subset that maximizes the hypervolume from a given candidate solution set. Many methods have been developed to solve or approximately solve different types of HSS problems. However, existing approaches cannot effectively solve HSS problems with a large number of objectives within a short computation time. This drawback directly limits their applicability as a component for developing new EMO algorithms. In this paper, we propose a novel learning-to-rank based framework, named LTR-HSS, for solving the challenging HSS problems with a large number of objectives. The experimental results show that, compared to other state-of-the-art HSS methods, our proposed LTR-HSS requires a shorter computation time to solve HSS problems with large numbers of objectives while achieving superior or competitive hypervolume performance. This demonstrates the potential of our method to be integrated into algorithms for many-objective optimization.

**Keywords:** Hypervolume subset selection · Multi-objective optimization · Machine learning

## 1 Introduction

Recently, subset selection has received considerable attention from the evolutionary multi-objective optimization (EMO) [8] community. From the perspective of the optimization mechanism, subset selection is intrinsically involved in different stages of the entire EMO process. For example, the initial population can be considered as selecting a subset from the decision space with a specific population size. During the evolutionary stages, the environmental selection is always performed, which aims to select a good subset from the parent and offspring


M. Affenzeller et al. (Eds.): PPSN 2024, LNCS 15151, pp. 36–51, 2024.
https://doi.org/10.1007/978-3-031-70085-9_3

solutions as the next generation [3,27]. In an EMO algorithm with an external archive, subset selection is an essential post-processing procedure to select a pre-specified number of representative solutions for the decision-makers [14,30].

Different criteria can be used for subset selection, including IGD [9], IGD+ [15], hypervolume (HV) [34] and R2 [13]. Among these criteria, subset selection based on hypervolume has been extensively investigated due to the widespread use of HV as a performance indicator in multi-objective optimization. Hypervolume subset selection (HSS) [1] aims to select the optimal solution subset from the candidate solution set that has the best hypervolume value within a given cardinality limit. Mathematically, for a given candidate solution set $S$, an HSS problem with the $k$ cardinality limit aims to find the optimal subset $S^*_{sub}$ that satisfies:

$$S^*_{sub} = \underset{S_{sub} \in S, |S_{sub}|=k}{\arg\max} HV(S_{sub}) \tag{1}$$

It is usually very challenging to find the exact optimal solution to HSS problems, especially when the number of objectives of the candidate solutions is large. Actually, an HSS problem is NP-hard when the number of objectives exceeds two [1]. Some heuristic and handcrafted methods [7,18,22] have been proposed to search for the near-optimal solutions to the HSS problems with large numbers of objectives. However, they usually require an excessively long computation time due to the expensive hypervolume contribution calculation (HVC) in high-dimensional spaces. For example, solving a 10-objective HSS problem using the currently most efficient greedy HSS algorithm [7] can take hundreds or thousands of seconds in computation time. This limitation directly makes their application in many-objective optimization impractical, such as integrated as a component to develop new EMO algorithms [18,27]. Therefore, there is currently a growing need to develop efficient methods for solving HSS in high-dimensional spaces while maintaining high performance.

However, the design of new powerful algorithms for HSS requires extensive domain knowledge by experts, such as a deep understanding of the properties of hypervolume [28]. In recent years, machine learning methods have shown remarkable success in solving problems in the field of multi-objective optimization [23,32,33]. Using machine learning methods provides the advantage of avoiding the need for intricate heuristic algorithm design, while also facilitating efficient end-to-end output through a trained model.

In this paper, we propose to develop a machine learning-based method for efficiently solving high-dimensional HSS problems, which is referred to as *Learning-to-Rank HSS* (LTR-HSS) framework. More specifically, the original HSS problem is formulated as a sequential decision-making problem. We address the solution selection by ranking the remaining candidate solutions where a utility function is learned for ranking. A novel loss function is intricately designed by fully considering the interrelationship between the already selected solutions and each remaining candidate during the ranking process. To evaluate the effectiveness of our proposed LTR-HSS framework, we conduct extensive experiments on high-dimensional HSS problems with different shapes of candidate solution sets. The experimental results show that the proposed LTR-HSS significantly outperforms

the state-of-the-art HSS algorithms in terms of efficiency (i.e., shorter running time) while achieving competitive hypervolume performance. In addition, we demonstrate the good generalization ability of our proposed framework in solving different scales of HSS problems.

The rest of the paper is organized as follows: In Sect. 2, we review existing hypervolume subset selection work based on different types of methods. Section 3 elaborates on the detailed introduction of the proposed LTR-HSS framework. The experimental studies are shown in Sect. 4. Finally, in Sect. 5, the conclusion is given.

## 2 Related Work

Currently, there exist the following methods for solving HSS problems:

**Exact Algorithms:** One type of method for HSS involves the design of exact algorithms to find the optimal subset. In early studies [4,5,20], exact algorithms were designed for two-objective problems. Then, those algorithms were extended to multi-objective problems with three or more objectives [10,11]. Since the search for the optimal subset is time-consuming, exact algorithms are applicable only for two-objective problems or small candidate sets of multi-objective problems with three or more objectives. These exact algorithms cannot solve HSS problems with a large number of objectives, which is the focus of our research in this paper. Actually, there is no existing method that can exactly search for optimal solutions to HSS problems in high-dimensional objective spaces.

**Greedy Algorithms.** Another type of HSS method is to greedily search for a good (i.e., near-optimal) subset of the HSS problem. In [12], an efficient greedy algorithm for two-objective problems was proposed. In [17], a simple greedy inclusion algorithm was used for multi- and many-objective problems. Currently, the efficiency of those greedy algorithms has been improved by using two ideas. One idea [7] is to avoid unnecessary calculations of hypervolume contributions. The other idea [28] is to decrease the computation time for hypervolume contribution calculations using an approximate calculation method [25]. Greedy algorithms clearly decrease the computation time for HSS. However, the quality of obtained subsets is not always very good.

**Local search Methods.** Local search-based methods have also been used for solving HSS problems. The basic idea is to iteratively update the selected subset by replacing the inferior solution within the subset with the solution with a higher HVC from a candidate set [2]. Recently, some advanced methods [21,22] have been proposed to improve the efficiency of basic local search methods for HSS. The local search-based methods ensure that the HV of the selected subset does not decrease as the iterations progress. However, their efficiency can heavily depend on the distribution of solutions in the candidate solution set. It is not unusual that a long computation time is still required to achieve a good subset when solving HSS problems with a large number of objectives.

**Evolutionary Methods.** Recently, some researchers have tried to use evolutionary algorithms to solve HSS problems [25,26]. Their basic idea is to formulate the original subset selection problems as a two-objective optimization problem, where the first objective is the original optimization objective (i.e., Eq. (1)) and the second objective is the cardinality (i.e., the number of solutions in the selection subset) [16]. Although these algorithms can achieve a theoretically guaranteed approximation performance (e.g., $1 - 1/e$ approximation), they still have difficulties in solving HSS problems with a large number of objectives within a short computation time to be seamlessly integrated into EMO algorithms.

## 3 Proposed Framework

In this section, we introduce our proposed framework LTR-HSS. Our idea is to formulate the original HSS problem in (1) as a sequential decision-making problem. Then, at each decision iteration, we select the solution among the remaining candidate set that has the largest utility relative to the previously selected solution set. To accomplish our LTR-HSS framework, the most challenging problem is to learn a model for solution utility calculation, where the definition of the utility function and loss function are required.

### 3.1 Definition of Utility Function

As we have explained, during the sequential solution selection process, the solutions are ranked according to their utility and the one with the largest utility value is selected to add the subset. Considering that the hypervolume measures both the convergence and diversity of a solution set simultaneously, the utility function should depend on the convergence features and diversity features of a solution. Thus, in this paper, we propose to define the utility function $U$ of a solution $x_i$ as follows:

$$U(x_i, S) = \boldsymbol{w_c}^T \boldsymbol{F_c}(x_i) + \boldsymbol{w_d}^T \boldsymbol{F_d}(x_i, S), \forall x_i \in Z/S \tag{2}$$

where $Z$ is the candidate set and $S$ is the previously selected solution set; $\boldsymbol{F_c}$ and $\boldsymbol{F_d}$ denote the convergence feature and diversity feature of solution $x_i$; $\boldsymbol{w_c}$ and $\boldsymbol{w_d}$ are the learnable parameters. Actually, any definition of the utility function and the corresponding learnable model can be applied to our proposed framework. For example, we can use neural networks to model the utility function as $U(x_i, S) = \boldsymbol{w}_{\Theta_c}^T \boldsymbol{F_c}(x_i) \odot \boldsymbol{w}_{\Theta_d}^T \boldsymbol{F_d}(x_i, S), \forall x_i \in Z/S$, where $\odot$ can be any specific mathematical operator.

The next step is to define the convergence feature vector $\boldsymbol{F_c}$ and the diversity feature vector $\boldsymbol{F_d}$. Intuitively, the convergence of a solution depends on each of its objective values (i.e., its location in the objective space), while its diversity is influenced by its diversity relationship (treated as distance relation) with the previously selected subset. Thus, we propose to define the convergence feature vector and the diversity feature vector in the following ways.

The convergence feature vector of a solution $x_i$ is calculated as a $R^{M+1}$ vector:

$$\boldsymbol{F_c}(x_i) = [x_i^1 - r^1, x_i^2 - r^2, ..., x_i^M - r^M, v_i]^\top \tag{3}$$

where $v_i$ is the individual hypervolume of solution $x_i$, $r = [r^1, r^2, ..., r^M]$ is the reference point, and $x_i^j$ denotes the $j$-$th$ objective value of solution $x_i$.

The diversity feature vector of a solution $x_i$ is calculated as a 6-dimensional vector using representative distance features:

$$\boldsymbol{F_d}(x_i, S) = [f_d^1(x_i, S), f_d^2(x_i, S), ..., f_d^6(x_i, S)]^\top \tag{4}$$

where each element is a distance-related calculation of solution $x_i$ with the previously selected solution set $S$ as follows:

$$f_d^1(x_i, S) = \min_{x_j \in S} Cosine\text{-}Similarity(x_i, x_j). \tag{5}$$

$$f_d^2(x_i, S) = \max_{x_j \in S} Cosine\text{-}Similarity(x_i, x_j). \tag{6}$$

$$f_d^3(x_i, S) = \frac{1}{|S|} \sum_{x_j \in S} Cosine\text{-}Similarity(x_i, x_j). \tag{7}$$

$$f_d^4(x_i, S) = \min_{x_j \in S} Distance(x_i, x_j). \tag{8}$$

$$f_d^5(x_i, S) = \max_{x_j \in S} Distance(x_i, x_j). \tag{9}$$

$$f_d^6(x_i, S) = \frac{1}{|S|} \sum_{x_j \in S} Distance(x_i, x_j). \tag{10}$$

In Eq. (5)–(10), *Cosine-Similarity* denotes the operation that measures the similarity (i.e., angle distance) between two vectors of an inner product space, and *Distance* denotes the operation that measures the Euclidean distance between two vectors.

### 3.2 Definition of Loss Function

Motivated by the sequential decision-making process, we propose to learn the model parameter through maximum likelihood estimation. That is, the parameter values of the proposed utility model are optimized to maximize the likelihood of the solution selection process described by the model, based on the ranking list that was actually observed. Suppose we have the training dataset $D = \{X_i, Y_i\}_{i=1}^N$, where the $X_n$ denotes the candidate solution set, $Y_n$ denotes the corresponding ranking list, and $N$ is the number of training data pairs. Then, the model parameter can be optimized by maximizing the empirical likelihood of the observed training dataset $D$:

$$Maximize \quad E_{(X_i, Y_j) \in D}[logP(Y_i|X_i)] \tag{11}$$

where $P(Y_i|X_i)$ represents the probability of generating the ranking list $Y_i$ for the given solution set $X_i$. Therefore, we can define the loss for training the model as the likelihood loss of the generation probability:

$$Loss = -\frac{1}{N}\sum_{i=1}^{N} logP(Y_i|X_i) \tag{12}$$

Intuitively, we can view the solution selection process as iteratively selecting the top-ranked solution from the remaining candidates. For each pair of training data $\{X_i, Y_i\}$, the generation probability $P(Y_i|X_i)$ is calculated as follows:

$$\begin{aligned} P(Y_i|X_i) &= P(x_{y(1)}, x_{y(2)}, ..., x_{y(l)}|X_i) \\ &= P(x_{y(1)}|X_i)P(x_{y(2)}|X_i \backslash S_1)...P(x_{y(l)}|X_i \backslash S_{l-1}) \end{aligned} \tag{13}$$

where $y(i)$ denotes the index of solution ranked at position $i$ in the ranking list $Y_i$, $x_{y(i)}$ denotes the solution ranked at position $y(i)$, and $S_i = \{x_{y(1)}, ..., x_{y(i)}\}$ denotes the previously selected solutions until the *i-th* iteration; $l$ denotes the length of the ranking list (i.e., the number of solutions in the candidate set).

We assume the solution selection probability is only determined by its utility value. Then, the generation probability can be defined as follows:

$$P(x_{y(j)}|X_i \backslash S_{j-1}) = \frac{exp\left\{U(x_{y(j)}, S_{j-1})\right\}}{\sum_{k=j}^{l} exp\left\{U(x_{y(k)}, S_{j-1})\right\}} \tag{14}$$

Then, we can calculate $P(Y_i|X_i)$ as follows:

$$\begin{aligned} P(Y_i|X_i) &= \prod_{j=1}^{l} P(x_{y(j)}|X_i \backslash S_{j-1}) \\ &= \prod_{j=1}^{l} \frac{exp\left\{U(x_{y(j)}, S_{j-1})\right\}}{\sum_{k=j}^{l} exp\left\{U(x_{y(k)}, S_{j-1})\right\}} \end{aligned} \tag{15}$$

Finally, the loss function is calculated as follows:

$$\begin{aligned} Loss &= -\frac{1}{N}\sum_{i=1}^{N} logP(Y_i|X_i) \\ &= -\sum_{i=1}^{N}\sum_{j=1}^{l} log\frac{exp\left\{U(x_{y(j)}, S_{j-1})\right\}}{\sum_{k=j}^{l} exp\left\{U(x_{y(k)}, S_{j-1})\right\}} \end{aligned} \tag{16}$$

where each utility function can be calculated by combining Eq. (2) with the convergence feature vector Eq.(3) and diversity feature vector Eq. (4).

### 3.3 Optimization and Prediction

With the defined loss function, we introduce the optimization process and prediction process in this subsection.

**Algorithm 1:** Optimization Algorithm

**Input** : training data $D = \{X_i, Y_i\}_{i=1}^{N}$, learning rate $\eta$, iteration $N_{iter}$, batch size $N_{batch}$;
**Output:** learnable parameter $\boldsymbol{w_c}$ and $\boldsymbol{w_d}$
Initialize model parameter $\boldsymbol{w_c}$ and $\boldsymbol{w_d}$;
**for** $i \leftarrow 1$ *to* $N_{iter}$ **do**
  **for** $j \leftarrow 1$ *to* $N/N_{batch}$ **do**
    Sample a mini-batch;
    Calculate the batch loss according to Eq. (16);
    Compute gradient: $\nabla \boldsymbol{w_c}^{(i)}$ and $\nabla \boldsymbol{w_d}^{(i)}$;
    Update model: $\boldsymbol{w_c} = \boldsymbol{w_c} - \eta \times \nabla \boldsymbol{w_c}^{(i)}$;
    $\boldsymbol{w_d} = \boldsymbol{w_d} - \eta \times \nabla \boldsymbol{w_d}^{(i)}$;

**Algorithm 2:** Prediction

**Input** : candidate solution set $X$, model parameter $\boldsymbol{w_c}$ and $\boldsymbol{w_d}$, number of selected solutions $k$;
**Output:** selected subset $S$
Initialize $S \leftarrow \emptyset$;
**for** $i \leftarrow 1$ *to* $k$ **do**
  Calculate the utility value for each solution in $X \backslash S$ according to Eq. (2);
  $y \leftarrow argmax_{x_j \in X \backslash S} U(x_j, S)$;
  $S \leftarrow S \cup y$;

First, to generate the training dataset, we adopt a similar strategy as proposed in [29]. To enable our proposed method to handle HSS problems with solution sets of different shapes, we sample solutions from unit $l_p$ spheres with different curvatures $p$ to construct the candidate solution set. More specifically, we first randomly sample a point $\boldsymbol{x} \in R^m$ using an exponential power distribution with density $f(\boldsymbol{x}) = \frac{1}{2\Gamma(1+1/p)} e^{-|\boldsymbol{x}|^p}$, where $\Gamma(t) = \int_0^\infty x^{t-1} e^{-x} dx$ is the Gamma function. Then, we obtained the sampled point $\boldsymbol{s} = |\boldsymbol{x}|/\|\boldsymbol{x}\|_p$. We iterate this process until enough points are sampled to construct the candidate solution set. Note that a random curvature value $p \in [0.5, 2]$ is chosen in each iteration. Since the optimal subsets for HSS problems with large numbers of objectives are not available, we use the solution ranking list obtained by LGI-HSS algorithms as the labels of the training dataset. For each number of objectives, we generate 10,000 solution sets with the same size of 100 using different curvature values $p$ as the training dataset.

With the training dataset, we use stochastic gradient descent to optimize the model as shown in Algorithm 1. We first initialize the model parameters using values drawn from the normal distribution. During each epoch of the training phase, we randomly sample a mini-batch to calculate the gradient for the model parameters, which are then used to update the model. In our experiments, we

use Adam [19], an effective gradient-based optimization method with an adaptive learning rate. The learning rate is set to $\eta = 10^{-4}$. We use the default settings in PyTorch [24] for all other parameters of Adam. The number of epochs for training the model is set to 100, and the batch size is set to 64 during each training epoch.

After obtaining the trained model, we can predict the sequentially selected solutions until we obtain the final solution subset. We generate the test candidate solution sets using the same manner as generating the training dataset. For a given candidate solution set $X$, as shown in Algorithm 2, we first initialize an empty subset $S$. Then, in each iteration, we calculate the utility value for each of the unselected solutions in $X \backslash S$ using the trained model parameter according to Eq. (2). The solution with the largest utility value is selected to add into the subset $S$. This selection process is repeated until $k$ solutions are selected as the final subset.

## 4 Experimental Studies

### 4.1 Experimental Settings

*1) Problem Setting:* We extensively evaluate the performance of the proposed LTR-HSS using different HSS problems with a large number of objectives. For the number of objectives, we consider 8-, 10-, and 12-objective HSS problems cases. Each candidate solution set is configured to contain 100 points. For each number of objectives, we generate 100 candidate solution sets using 100 different random sphere parameters $p$ (i.e., each of the 100 candidate solution sets has different shapes from each other).

*2) Algorithms Setting:* Our proposed LTR-HSS is compared to state-of-the-art methods for solving HSS problems, mainly including LGI-HSS [7] and GAHSS [28]. LGI-HSS is currently the most efficient greedy HSS algorithm that employs the exact hypervolume contribution calculation. GAHSS is also designed to solve the HSS problems with a large number of objectives, which uses an R2-based method to approximately calculate the hypervolume contribution. We select these two methods for the main comparison because all other methods, such as local search methods (e.g., APL-HSS) and evolutionary methods (PROSS), cannot search for a subset with satisfactory hypervolume within a limited computation time. That is, these methods cannot efficiently solve HSS problems with a large number of objectives in our experimental settings. However, we still compare these methods with our proposed LTR-HSS to illustrate this point, including APL-HSS [22], PROSS [25], DSS [31], and CSS-MEA [6].

*3) Parameter Setting:* For GAHSS, the number of direction vectors is set to 300 for the main comparison. We also examine other different specifications of the number of direction vectors. All other compared algorithms use their default parameter settings. For a fair comparison, the reference point is set to $(1.1, ..., 1.1)^m$ for the hypervolume calculation used during all the algorithm implementations and performance evaluations.

The proposed LTR-HSS is coded in Python and all the experiments are measured on an Intel Core i7-8700K CPU with 16 GB of RAM, running in Windows 10. All codes and datasets in this work can be found at: https://github.com/HisaoLabSUSTC/LTRHSS-PPSN2024.

### 4.2 Performance Comparison

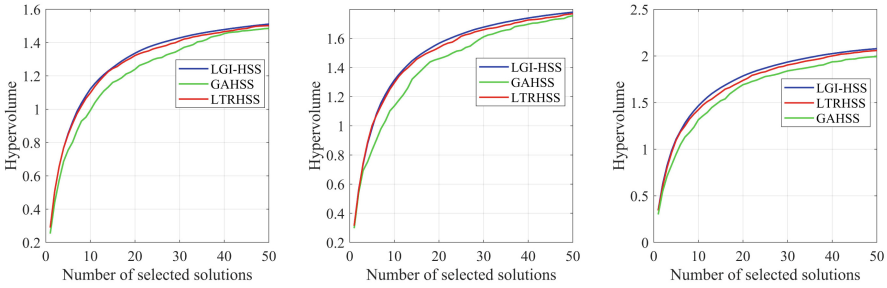
We show the computation time results of the proposed LTR-HSS and compared algorithms in Table 1 (with the number of selected solutions set to $k = 50$). Table 1 shows the total computation time of 100 test problems achieved by each algorithm and their average rank performance. We also calculate the speedup in latency (i.e., $\frac{\text{time of compared algorithm}}{\text{time of LTR-HSS}}$) achieved by LTR-HSS as evaluated in [28]. We can see that our proposed LTR-HSS has the shortest computation time among the compared algorithms. The average rank of 1 for LTR-HSS indicates that the proposed method is always faster than other algorithms across HSS problems with different numbers of objectives and shapes. We also observe that LTR-HSS achieves a significant speedup compared to LGI-HSS. Furthermore, the speedup attained by LTR-HSS in comparison to LGI-HSS and GAHSS becomes more pronounced as the number of objectives increases. These results clearly

**Table 1.** Comparison of the proposed LTR-HSS and other HSS methods in terms of computation time (in seconds). We report the total time of solving 100 test problems ($k = 50$) and the average rank performance of each method.

| | | LGI-HSS | GAHSS | LTR-HSS(Ours) | $\frac{\text{LGI-HSS}}{\text{LTR-HSS}}$ | $\frac{\text{GAHSS}}{\text{LTR-HSS}}$ |
|---|---|---|---|---|---|---|
| 8-objective | Total time | 25.2741 | 4.0091 | 3.0608 | 8.2574 | 1.3098 |
| | Avg. rank | 3 | 2 | 1 | / | / |
| 10-objective | Total time | 159.5903 | 4.3844 | 3.1597 | 50.5081 | 1.3876 |
| | Avg. rank | 3 | 2 | 1 | / | / |
| 12-objective | Total time | 962.1576 | 4.9726 | 3.2578 | 295.3432 | 1.5264 |
| | Avg. rank | 3 | 2 | 1 | / | / |

**Table 2.** Comparison of the proposed LTR-HSS and other HSS methods in terms of hypervolume. We report the average value of solving 100 test problems ($k = 50$) and the average rank performance of each method.

| | | LGI-HSS | GAHSS | LTR-HSS | $\frac{\text{LTRHSS}-\text{LGI-HSS}}{\text{LGI-HSS}}$ | $\frac{\text{LTRHSS}-\text{GAHSS}}{\text{GAHSS}}$ |
|---|---|---|---|---|---|---|
| 8-objective | Avg. HV | 1.8233 | 1.8094 | 1.8147 | −0.0088 | 0.0043 |
| | Avg. rank | 1 | 2.78 | 2.22 | / | / |
| 10-objective | Avg. HV | 2.1664 | 2.1352 | 2.1533 | −0.0054 | 0.0092 |
| | Avg. rank | 1 | 2.95 | 2.05 | / | / |
| 12-objective | Avg. HV | 2.6165 | 2.5696 | 2.6006 | −0.0065 | 0.0140 |
| | Avg. rank | 1 | 2.96 | 2.04 | / | / |



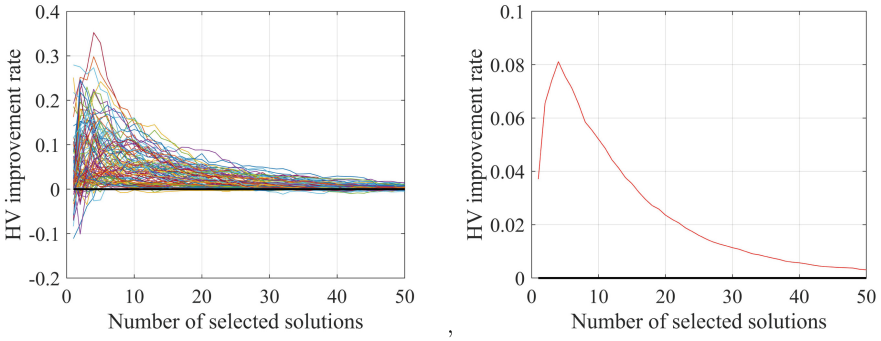

(a) 8-objective HSS (b) 10-objective HSS (c) 12-objective HSS

**Fig. 1.** The achieved hypervolume performances when selecting different numbers of solutions of compared algorithms in a single run.

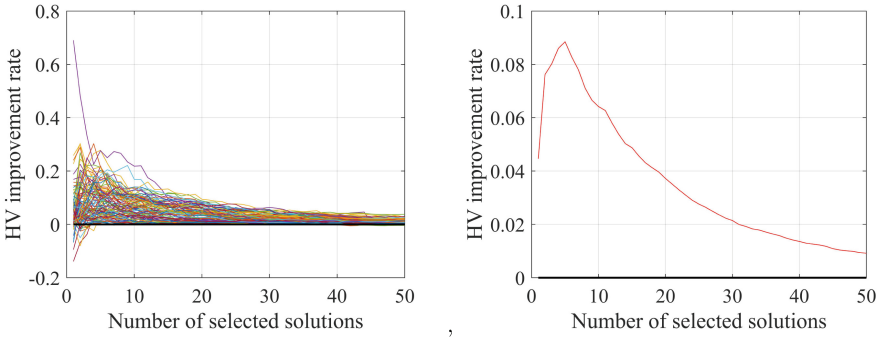
show the better applicability of the proposed LTR-HSS for many-objective optimization.

Table 2 shows the average hypervolume of the final subset achieved by each compared algorithm across 100 test problems. We also calculate the hypervolume improvement rate of LTR-HSS with respect to LGI-HSS and GAHSS (i.e., $\frac{\text{HV of LTR-HSS - HV of compared algorithm}}{\text{HV of compared algorithm}}$), as evaluated in [28]. We can observe that LGI-HSS achieves the best performance in terms of hypervolume, which is consistent with our intuition. This is because LGI-HSS uses the exact hypervolume contribution calculation, while our method uses a utility model trained from the selection process of LGI-HSS. However, as we have shown in Table 1, LGI-HSS becomes very time-consuming as the number of objectives increases, which limits its applicability for solving HSS problems with more than 10 objectives. Considering the small hypervolume improvement rate of LTR-HSS with respect to LGI-HSS, our proposed method is superior when applied to many-objective optimization. When compared to GAHSS, LTR-HSS achieves higher hypervolume performance. Also, the advantage of LTR-HSS over GAHSS becomes more pronounced as the number of objectives increases, since the average rank of LTR-HSS becomes smaller and the hypervolume improvement rate becomes larger as shown in Table 2. Considering the shorter computation time of LTR-HSS, our proposed method completely outperforms GAHSS.

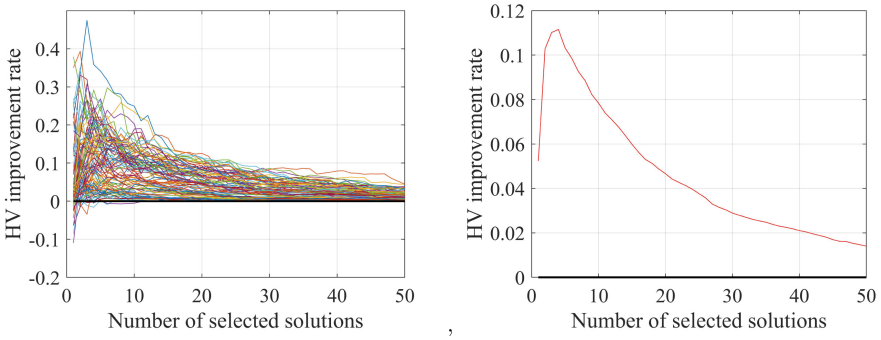
We also evaluate the performance of the proposed LTR-HSS when selecting different numbers of solutions (i.e., different $k$ in Eq. (1)). Figure 1 shows the achieved hypervolume performances when selecting different numbers of solutions of compared HSS algorithms in a single run. We can clearly observe that LTR-HSS can always achieve very similar hypervolume performance to that of LGI-HSS when selecting different numbers of solutions. Considering the significantly longer computation time required by LGI-HSS, our proposed LTR-HSS can effectively serve as a replacement for LGI-HSS in many-objective optimization.




(a) 8-objective HSS

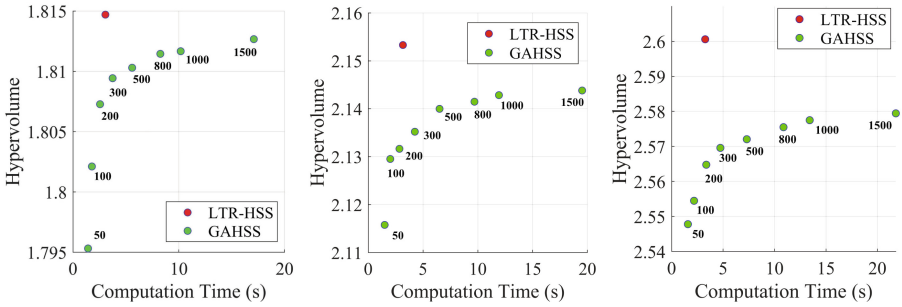



(b) 10-objective HSS




(c) 12-objective HSS

**Fig. 2.** Hypervolume improvement rate of LTRHSS with respect to GAHSS on each candidate solution set and the average hypervolume improvement rate for HSS with 8-objective, 10-objective, and 12-objective candidate solution sets.




(a) 8-objective HSS (b) 10-objective HSS (c) 12-objective HSS

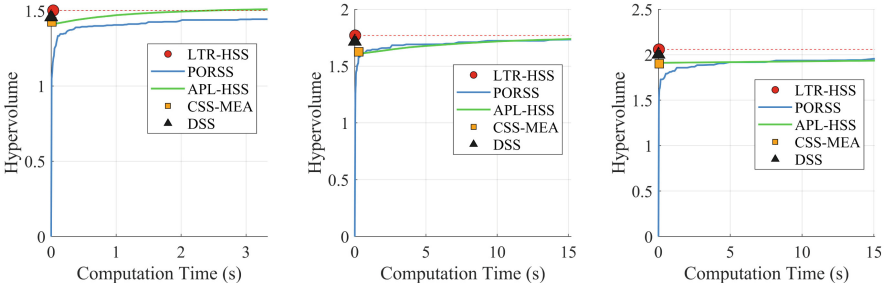
**Fig. 3.** The comparison between LTR-HSS and GAHSS in terms of the average hypervolume performances and computation time. We examine GAHSS with different numbers of direction vectors, as illustrated as the numbers in the figures.

When compared to GAHSS, as shown in Fig. 1, LGI-HSS consistently shows better hypervolume performance when selecting different numbers of solutions. In Fig. 2, we further plot the hypervolume improvement rate of LTR-HSS with respect to GAHSS for each of the 100 test problems and their average result. We can see that the hypervolume improvement rate is larger than zero when selecting different numbers of solutions for almost all the 100 test problems. The average hypervolume improvement rate is also larger than zero. These results show that the proposed LTR-HSS always has a stable better performance than GAHSS.

In Fig. 3, we further compare the performances of the LTR-HSS with GAHSS by setting GAHSS with different numbers of direction vectors, which serves as a parameter controlling the trade-off between accuracy and complexity. We evaluate GAHSS with the following numbers of direction vectors: $|\Lambda| = 50, 100, 200, 300, 500, 800, 1000$, and 1500. From Fig. 3, we can observe that the proposed LTR-HSS always outperforms GAHSS in terms of both hypervolume and computation time, even when tuning the parameter of GAHSS.

### 4.3 Comparison with Other State-of-the-Art HSS Methods

Some other advanced methods also have been developed for solving HSS problems. However, these methods have difficulties in effectively solving HSS problems with a large number of objectives within a limited computation time. In other words, they cannot be integrated into EMO algorithms due to their inefficiency. In this subsection, we compare the proposed LTR-HSS with other representative HSS methods to emphasize this point, including PROSS, APL-HSS, CSS-MEA, and DSS. PROSS is a state-of-the-art evolutionary algorithm for solving subset section problems, which is an improved version of the previous algorithm POSS. APL-HSS is a newly proposed local search HSS method that outperforms other existing HSS methods based on local search. We also compare




(a) 8-objective HSS (b) 10-objective HSS (c) 12-objective HSS

**Fig. 4.** The comparison between LTR-HSS and the other four HSS methods in terms of the hypervolume performances and computation time in a single run.

two general subset selection methods that usually achieve satisfactory performance: the distance-based subset selection (DSS) [31] and the clustering subset selection based on k-means (CSS-MEA) [6].

Figure 4 shows curves of their achieved hypervolume performance over computation time in a single run. Note that PROSS and ALP-HSS are iterative algorithms, and their performance improves as the computation time increases. We can observe that LTR-HSS always outperforms CSS-MEA and DSS in terms of hypervolume of the obtained final subset, while their computation times are all very short with no significant differences. CSS-MEA and DSS can solve HSS problems with a large number of objectives, since they are general subset selection methods without any specific mechanism for handling HVC calculation or approximate HVC calculation. When compared to PROSS and APL-HSS, the proposed LTR-HSS achieves better performance in terms of both hypervolume and computation time, even after a long iteration of PROSS and APL-HSS. Only on 8-objective HSS after a long computation time, APL-HSS can achieve a similar hypervolume performance as LTR-HSS. However, when handling HSS problems with a larger number of objectives, APL-HSS exhibits clear inferiority compared to LTR-HSS. In other words, both APL-HSS and PROSS cannot effectively solve HSS problems with a large number of objectives within a short computation time, which directly limits their applicability to existing EMO algorithms. These experimental results demonstrate the superiority of our proposed LTR-HSS over other state-of-the-art HSS methods.

## 5 Conclusion

In this paper, we proposed a learning-to-rank based framework for hypervolume subset selection, which is called LTR-HSS. We compared our proposed method with other state-of-the-art HSS methods. The experimental results showed our proposed LTR-HSS can efficiently solve HSS problems with large numbers of objectives within a very short computation time while achieving very good

hypervolume performance. More specifically, our LTR-HSS outperforms other HSS methods in terms of both computation time and hypervolume performance, besides LGI-HSS. When compared to LGI-HSS, our LTR-HSS is significantly faster while achieving competitive hypervolume performance. Considering the extremely long computation time of LGI-HSS, our proposed LTR-HSS is superior to all existing HSS methods when solving HSS problems with large numbers of objectives.

There are two primary research directions in the future. First, since our proposed LTR-HSS serves as a general framework, we will explore the use of more sophisticated models (e.g., deep neural networks) to define the utility function for improving the current performance. Secondly, we will investigate integrating the proposed HSS method as a component for developing new EMO algorithms. This is primarily aimed at addressing the limitation of the indicator-based SMS-EMOA, which struggles to efficiently handle problems with a large number of objectives in a reasonable computation time.

**Acknowledgments.** This work was supported by the National Key R&D Program of China (Grant No. 2023YFE0106300), National Natural Science Foundation of China (Grant No. 62250710163, 62376115, 62276223), Guangdong Provincial Key Laboratory (Grant No. 2020B121201001), and the Research Grants Council of the Hong Kong Special Administrative Region, China [GRF Project No. CityU 11215622].
**Disclosure of Interests.** The authors have no competing interests to declare that are relevant to the content of this article.

## References

1. Bader, J., Zitzler, E.: Hype: an algorithm for fast hypervolume-based many-objective optimization. Evol. Comput. **19**(1), 45–76 (2011)
2. Basseur, M., Derbel, B., Goëffon, A., Liefooghe, A.: Experiments on greedy and local search heuristics for d dimensional hypervolume subset selection. In: Proceedings of the Genetic and Evolutionary Computation Conference 2016, pp. 541–548 (2016)
3. Beume, N., Naujoks, B., Emmerich, M.: SMS-EMOA: multiobjective selection based on dominated hypervolume. Eur. J. Oper. Res. **181**(3), 1653–1669 (2007)
4. Bringmann, K., Friedrich, T., Klitzke, P.: Generic postprocessing via subset selection for hypervolume and epsilon-indicator. In: Bartz-Beielstein, T., Branke, J., Filipič, B., Smith, J. (eds.) PPSN 2014. LNCS, vol. 8672, pp. 518–527. Springer, Cham (2014). https://doi.org/10.1007/978-3-319-10762-2_51
5. Bringmann, K., Friedrich, T., Klitzke, P.: Two-dimensional subset selection for hypervolume and epsilon-indicator. In: Proceedings of the 2014 Annual Conference on Genetic and Evolutionary Computation, pp. 589–596 (2014)
6. Chen, W., Ishibuchi, H., Shang, K.: Clustering-based subset selection in evolutionary multiobjective optimization. In: 2021 IEEE International Conference on Systems, Man, and Cybernetics (SMC), pp. 468–475. IEEE (2021)
7. Chen, W., Ishibuchi, H., Shang, K.: Fast greedy subset selection from large candidate solution sets in evolutionary multiobjective optimization. IEEE Trans. Evol. Comput. **26**(4), 750–764 (2021)

8. Coello, C.C.: Evolutionary multi-objective optimization: a historical view of the field. IEEE Comput. Intell. Mag. **1**(1), 28–36 (2006)
9. Bringmann, K., Friedrich, T., Klitzke, P.: Generic postprocessing via subset selection for hypervolume and epsilon-indicator. In: Bartz-Beielstein, T., Branke, J., Filipič, B., Smith, J. (eds.) PPSN 2014. LNCS, vol. 8672, pp. 518–527. Springer, Cham (2014). https://doi.org/10.1007/978-3-319-10762-2_51
10. Gomes, R.J., Guerreiro, A.P., Kuhn, T., Paquete, L.: Implicit enumeration strategies for the hypervolume subset selection problem. Comput. Oper. Res. **100**, 244–253 (2018)
11. Groz, B., Maniu, S.: Hypervolume subset selection with small subsets. Evol. Comput. **27**(4), 611–637 (2019)
12. Guerreiro, A.P., Fonseca, C.M., Paquete, L.: Greedy hypervolume subset selection in low dimensions. Evol. Comput. **24**(3), 521–544 (2016)
13. Hansen, M.P., Jaszkiewicz, A.: Evaluating the quality of approximations to the non-dominated set. IMM, Department of Mathematical Modelling, Technical Universityof Denmark (1994)
14. Ishibuchi, H., Pang, L.M., Shang, K.: A new framework of evolutionary multi-objective algorithms with an unbounded external archive. In: Proceedings European Conference on Artificial Intelligence, pp. 283–290 (2020)
15. Ishibuchi, H., Masuda, H., Tanigaki, Y., Nojima, Y.: Modified distance calculation in generational distance and inverted generational distance. In: Gaspar-Cunha, A., Henggeler Antunes, C., Coello, C.C. (eds.) EMO 2015. LNCS, vol. 9019, pp. 110–125. Springer, Cham (2015). https://doi.org/10.1007/978-3-319-15892-1_8
16. Ishibuchi, H., Sakane, Y., Tsukamoto, N., Nojima, Y.: Selecting a small number of representative non-dominated solutions by a hypervolume-based solution selection approach. In: 2009 IEEE International Conference on Fuzzy Systems, pp. 1609–1614. IEEE (2009)
17. Ishibuchi, H., Setoguchi, Y., Masuda, H., Nojima, Y.: How to compare many-objective algorithms under different settings of population and archive sizes. In: 2016 IEEE Congress on Evolutionary Computation (CEC), pp. 1149–1156. IEEE (2016)
18. Jiang, S., Zhang, J., Ong, Y.S., Zhang, A.N., Tan, P.S.: A simple and fast hypervolume indicator-based multiobjective evolutionary algorithm. IEEE Trans. Cybern. **45**(10), 2202–2213 (2014)
19. Kingma, D.P., Ba, J.: Adam: a method for stochastic optimization. arXiv preprint arXiv:1412.6980 (2014)
20. Kuhn, T., Fonseca, C.M., Paquete, L., Ruzika, S., Duarte, M.M., Figueira, J.R.: Hypervolume subset selection in two dimensions: formulations and algorithms. Evol. Comput. **24**(3), 411–425 (2016)
21. Nan, Y., Shang, K., Ishibuchi, H., He, L.: Improving local search hypervolume subset selection in evolutionary multi-objective optimization. In: 2021 IEEE International Conference on Systems, Man, and Cybernetics (SMC), pp. 751–757. IEEE (2021)
22. Nan, Y., Shang, K., Ishibuchi, H., He, L.: An improved local search method for large-scale hypervolume subset selection. IEEE Trans. Evol. Comput. (2022)
23. Navon, A., Shamsian, A., Chechik, G., Fetaya, E.: Learning the pareto front with hypernetworks. arXiv preprint arXiv:2010.04104 (2020)
24. Paszke, A., et al.: PyTorch: an imperative style, high-performance deep learning library. In: Advances in Neural Information Processing Systems, vol. 32 (2019)

25. Qian, C., Bian, C., Feng, C.: Subset selection by pareto optimization with recombination. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 34, pp. 2408–2415 (2020)
26. Qian, C., Yu, Y., Zhou, Z.H.: Subset selection by pareto optimization. In: Advances in Neural Information Processing Systems, vol. 28 (2015)
27. Shang, K., Ishibuchi, H.: A new hypervolume-based evolutionary algorithm for many-objective optimization. IEEE Trans. Evol. Comput. **24**(5), 839–852 (2020)
28. Shang, K., Ishibuchi, H., Chen, W.: Greedy approximated hypervolume subset selection for many-objective optimization. In: Proceedings of the Genetic and Evolutionary Computation Conference, pp. 448–456 (2021)
29. Shang, K., Shu, T., Ishibuchi, H., Nan, Y., Pang, L.M.: Benchmarking large-scale subset selection in evolutionary multi-objective optimization. Inf. Sci. **622**, 755–770 (2023)
30. Shu, T., Shang, K., Ishibuchi, H., Nan, Y.: Effects of archive size on computation time and solution quality for multi-objective optimization. IEEE Trans. Evol. Comput. (2022)
31. Singh, H.K., Bhattacharjee, K.S., Ray, T.: Distance-based subset selection for benchmarking in evolutionary multi/many-objective optimization. IEEE Trans. Evol. Comput. **23**(5), 904–912 (2018)
32. Suresh, A., Deb, K.: Machine learning based prediction of new pareto-optimal solutions from pseudo-weights. IEEE Trans. Evol. Comput. (2023)
33. Zhang, X., Lin, X., Xue, B., Chen, Y., Zhang, Q.: Hypervolume maximization: a geometric view of pareto set learning. In: Advances in Neural Information Processing Systems, vol. 36 (2024)
34. Zitzler, E., Thiele, L.: Multiobjective optimization using evolutionary algorithms — a comparative case study. In: Eiben, A.E., Bäck, T., Schoenauer, M., Schwefel, H.-P. (eds.) PPSN 1998. LNCS, vol. 1498, pp. 292–301. Springer, Heidelberg (1998). https://doi.org/10.1007/BFb0056872

# Three Objectives Degrade the Convergence Ability of Dominance-Based Multi-objective Evolutionary Algorithms

Cheng Gong[1,2,3], Lie Meng Pang[2], Qingfu Zhang[1,3(✉)], and Hisao Ishibuchi[2(✉)]

[1] City University of Hong Kong, Hong Kong, China
chenggong6-c@my.cityu.edu.hk
[2] Southern University of Science and Technology, Shenzhen 518055, China
{panglm,hisao}@sustech.edu.cn
[3] The City University of Hong Kong Shenzhen Research Institute, Shenzhen, China
qingfu.zhang@cityu.edu.hk

**Abstract.** In the evolutionary multi-objective optimization (EMO) community, it is well known that the convergence ability of dominance-based multi-objective evolutionary algorithms (MOEAs) is severely deteriorated on many-objective problems with more than three objectives. In this paper, we clearly demonstrate that the convergence ability of NSGA-II deteriorates even in the case of three objectives. Our experimental results on multi-objective knapsack and traveling salesman problems with 2–6 objectives show that NSGA-II starts to deteriorate the quality of the current population after a number of generations even when it is applied to three-objective problems. Surprisingly, NSGA-III also shows a similar performance deterioration. We analyze the search behavior of NSGA-II, NSGA-III, three versions of MOEA/D, and SMS-EMOA. Then, we explain the reason for the performance deterioration of NSGA-II and NSGA-III, which exists in the environmental selection mechanism of each algorithm. Another interesting observation is that NSGA-II has the best or second best performance (next to MOEA/D with the weighted sum) among the examined algorithms on many-objective problems in early generations before it starts to show performance deterioration.

**Keywords:** Multi-objective optimization · Pareto dominance-based algorithms · Evolutionary multi-objective optimization

## 1 Introduction

In the evolutionary multi-objective optimization (EMO) community, it has been repeatedly pointed out in the literature [14,17] that the convergence ability of Pareto dominance-based multi-objective evolutionary algorithms (MOEAs) such as NSGA-II [8] and SPEA2 [19] is severely deteriorated on many-objective problems with more than three objectives. This is because almost all solutions in


M. Affenzeller et al. (Eds.): PPSN 2024, LNCS 15151, pp. 52–67, 2024.
https://doi.org/10.1007/978-3-031-70085-9_4

the current population quickly become non-dominated in many-objective optimization. Some studies also pointed out that the difficulty of many-objective optimization does not simply depend on the number of objectives [12,15]. However, it can be viewed as a general consensus in the EMO community that dominance-based MOEAs do not work well on many-objective problems with more than three objectives. This general consensus also implies that dominance-based MOEAs work well on multi-objective problems with two or three objectives.

In this paper, we present counter-examples to this consensus using NSGA-II. We examine the anytime performance of NSGA-II on multiobjective traveling salesman problems (MOTSP) and multi-objective knapsack problems (MOKP) with 2–6 objectives. Our experimental results clearly show that the quality of the current population starts to deteriorate in the middle of the execution of NSGA-II even in the case of only three objectives. Before this deterioration (i.e., in early generations), NSGA-II shows better performance than well-known algorithms such as MOEA/D [18], NSGA-III [7] and SMS-EMOA [3] even for six-objective MOTSP and MOKP. That is, our experimental results show that NSGA-II works well on many-objective problems (in early generations) and NSGA-II does not work well on three-objective problems (in later generations). These observations are clearly different from the above-mentioned consensus in the EMO community. We also demonstrate that NSGA-III shows a similar performance deterioration in the middle of its evolution.

None of the above-mentioned interesting observations is obtained when we start with a randomly generated initial population. This may be the reason why they have not been reported in the literature. In combinatorial optimization, it is not realistic in many cases to use a random initial population since (i) randomly generated solutions are far away from Pareto fronts and (ii) domain-specific heuristics are often available for single-objective problems. That is, the performance of MOEAs can be drastically improved by adding heuristic solutions to a random initial population [5,9,10]. All of the above-mentioned interesting observations are obtained from our experiments only when a few heuristic solutions are added to random initial solutions. We examine NSGA-II, two implementations of NSGA-III (in PlatEMO [16] and pymoo [4], three versions of MOEA/D [18] with different scalarizing functions, and SMS-EMOA [3]. In early generations (e.g., until the 100th generation), NSGA-II shows higher performance than the other algorithms (except for MOEA/D with the weighted sum) even for six-objective problems. Then, NSGA-II starts to deteriorate the quality of the current population even for three-objective problems. Surprisingly, NSGA-III also shows a similar performance deterioration. After reporting these interesting observations, we examine the reason why NSGA-II (and NSGA-III) shows such a performance deterioration even for three-objective problems.

This paper is organized as follows: In Sect. 2, we show the anytime performance of NSGA-II and other MOEAs for MOTSP and MOKP with 2–6 objectives using a purely random initial population. In Sect. 3, we perform the same experiments as in Sect. 2 using a random initial population with a few heuristic

solutions. We also examine the search behavior of NSGA-II in detail for three-objective problems in Sect. 3. Finally, Sect. 4 concludes this paper.

## 2 Performance Evaluation of NSGA-II Using Random Initial Solutions

In this section, we first examine the performance of NSGA-II [8] on MOTSP and MOKP problems using randomly generated initial populations. For comparison, we also examine the performance of MOEA/D [18] (representative of decomposition-based MOEAs), SMS-EMOA [3] (representative of indicator-based MOEAs), and NSGA-III [7] (an improved version of NSGA-II for handling many-objective optimization problems).

### 2.1 Test Problems

In our study, the multi-objective travelling salesman problem (MOTSP) and multi-objective knapsack problems (MOKP) are used to evaluate the performance of different MOEAs. We generate different MOTSP and MOKP with two to six objectives, respectively.

The MOTSP is an extended version of the standard TSP problem, where there exist multiple costs of travel between each pair of cities. Formally, for an $m$-objective MOTSP problem, given a set of nodes $V = \{1, ..., n\}$ and $m$ cost matrices $C = \{C_1, ..., C_m\}$, the goal is to minimize each of the $m$ cost objectives. The $k$-$th$ cost objective is defined as:

$$f_k(\boldsymbol{x}) = \sum_{i \in V} \sum_{j \in V, j \neq i} c_{ij}^k x_{ij}, \ k \in \{1, ..., m\} \tag{1}$$

where $c_{ij}^k$ is the elements of $C_k$ that denotes the $k$-$th$ cost of travel between node $i$ and $j$, and $x_{ij}$ is 1 if there exists an edge between node $i$ and $j$, otherwise 0. In our experiments, for each $m$-objective TSP problem, we generate $m$ independent matrices by assigning each pair of cities with $m$ numbers as the costs, which are randomly drawn from the interval [0,1).

For MOKP, we convert the multi-objective 0–1 knapsack problems proposed in [20] to the minimization problems by multiplying each objective by $-1$ and enforcing each objective to be positive. An $m$-objective 0–1 knapsack problem with $n$ items is formulated as follows:

$$\text{Minimize} \ \ \boldsymbol{F}(\boldsymbol{x}) = (f_1(\boldsymbol{x}), f_2(\boldsymbol{x}), ..., f_m(\boldsymbol{x})), \tag{2}$$

$$\text{subject to} \ \sum_{j=1}^{n} b_{ij} x_j \leq c_i, \ i = 1, 2, ..., m, \tag{3}$$

$$x_j \in \{0, 1\}, \ j = 1, 2, ..., n. \tag{4}$$

where $f_i(\boldsymbol{x}) = \sum_{j=1}^{n} a_{ij}(1 - x_j)$ and $c_i = p \sum_{j=1}^{n} b_{ij}$ for $i = 1, 2, ..., m$. In this formulation, $a_{ij}$ is the profit of item $j$ according to knapsack $i$, $b_{ij}$ is the weight

of item $j$ according to knapsack $i$, and $c_i$ is the capacity of knapsack $i$ with $p$ control the capacity size. In this paper, we generate different MOKPs with 500 items, where $a_{ij}$ and $b_{lj}$ were randomly specified as integers in the interval [10, 100] and $p$ is set to 0.5 (i.e., capacity is specified as 50% of the total weights).

### 2.2 Parameter Settings

For the parameter settings, the population size $N$ is set to 91 for two- and three-objective problems, 120 for four-objective problems, 210 for five-objective problems, and 258 for six-objective problems. The termination condition for each algorithm is set to 5,000 generations. All experiments are performed on the PlatEMO platform [16].

For SMS-EMOA, we use the exact hypervolume calculation for two- to four-objective problems and the approximated hypervolume calculation [2] for five- and six-objective problems. As for NSGA-III, it has been recently reported that a different implementation of the normalization mechanism can strongly affect its performance for multi-objective combinatorial problems [11]. To ensure the reliability of the experimental results, it's worth noting that we also implement a pymoo [4] version of NSGA-III that uses a different normalization mechanism from the PlatEMO [16] implementation of NSGA-III for a comparison purpose. For MOEA/D, we use the weighted sum, modified Tchebycheff, and penalty boundary intersection (PBI) functions as the scalarizing functions, denoted as MOEA/D-WS, MOEA/D-mTche, and MOEA/D-PBI, respectively. The original MOEA/D randomly assigns initial solutions to each weight vector. In our study, each initial solution is carefully assigned to its corresponding weight vector based on the best scalarizing function value.

We conduct 21 independent runs of each algorithm on each test problem. The hypervolume (HV) [21] is used as the indicator for the performance evaluation. For MOTSP and MOKP, which are the minimization problems, the reference point for calculating HV is defined as [9]:

$$ref = F^{max} + 0.1 \times (F^{max} - F^{min}) \tag{5}$$

where $F^{max} = (f_1^{max}, f_2^{max}, ..., f_m^{max})$ and $F^{min} = (f_1^{min}, f_2^{min}, ..., f_m^{min})$ are the maximum and minimum objective values ever found by all compared algorithms in our computational experiments, respectively. The final hypervolume value of the population at each generation is normalized by dividing it by the hypervolume of $F^{min}$ (which can be viewed as an estimated ideal point).

### 2.3 Experimental Results

Figure 1 and Fig. 2 show the average HV values over 5000 generations for MOTSP and MOKP problems, respectively, with randomly generated solutions are used as the initial population for the seven compared MOEAs.

We can observe that the performance of NSGA-II is similar to MOEA/D, SMS-EMOA, and NSGA-III on two-objective and three-objective problems.

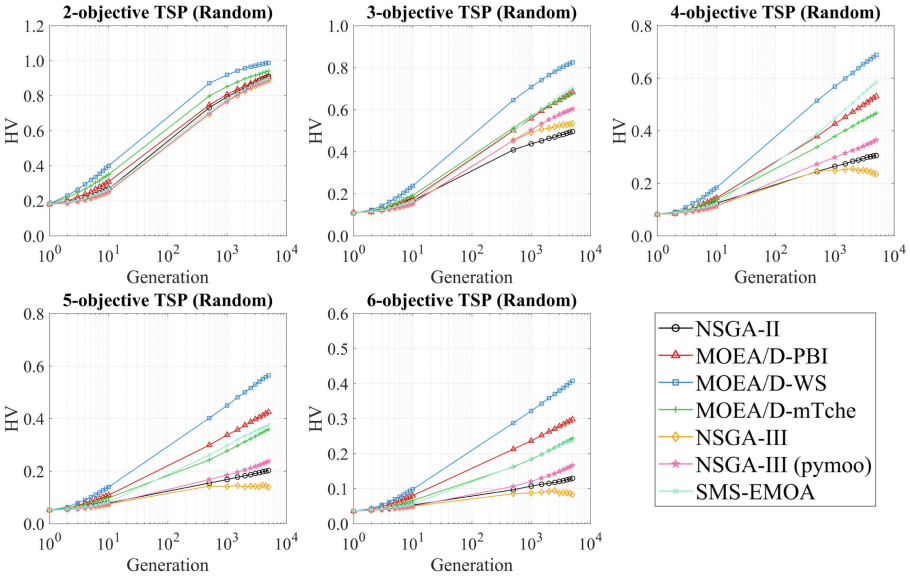



**Fig. 1.** Average HV for two-objective, three-objective, four-objective, five-objective, and six-objective TSP problems over 5000 generations, obtained from 21 runs. Randomly generated solutions are used as the initial population.

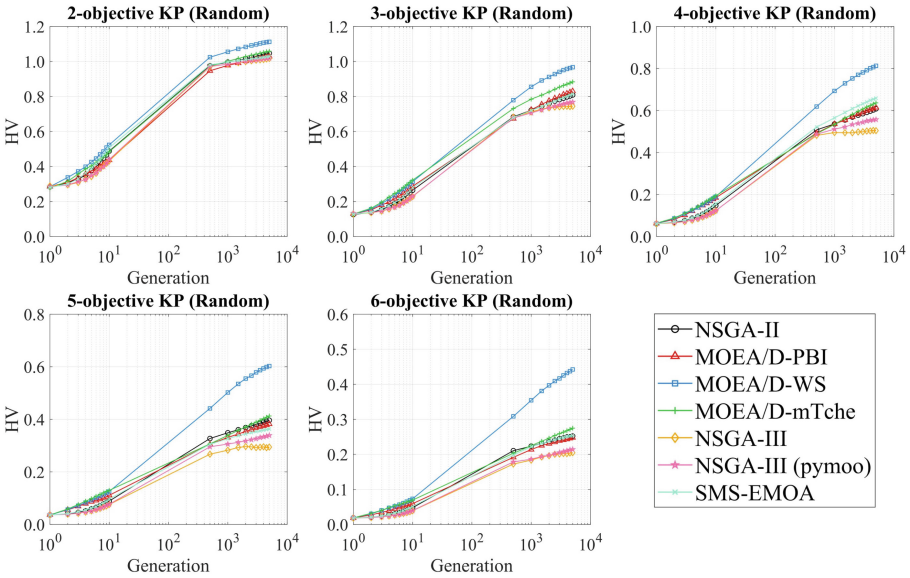



**Fig. 2.** Average HV for two-objective, three-objective, four-objective, five-objective, and six-objective KP problems over 5000 generations, obtained from 21 runs. Randomly generated solutions are used as the initial population.

However, the performance of NSGA-II is clearly inferior to MOEA/D and SMS-EMOA on four-, five- and six-objective problems. Moreover, the difference between NSGA-II and the others increases with the number of objectives. These observations are consistent with reported results in the literature [12], which indicate that NSGA-II usually cannot achieve satisfying performance when solving many-objective problems (i.e., problems with more than three objectives).

In general, none of these algorithms show significant degradation in performance throughout the evolutionary process. Based on these results, it appears that these algorithms are capable of maintaining and utilizing good solutions in the current generation. It is important to note that the performance stagnation observed in NSGA-III (i.e., its performance cannot be improved after around 1000 generations as indicated by the yellow line in Fig. 1) is due to the normalization mechanism implemented in PlatEMO, which has been explained in [11]. With pymoo's normalization mechanism implemented, NSGA-III exhibits improved performance, as shown by the pink line in Fig. 1.

## 3 Performance Evaluation of NSGA-II Using Heuristic Initial Solutions

In this section, we examine the performance of the seven MOEAs with random initial populations including a few heuristic solutions. The details of how to obtain the heuristic solutions are explained in the following subsections.

### 3.1 Heuristic Solutions for MOTSP and MOKP

In this study, we generate $(m+1)$ heuristic solutions for MOTSP and MOKP by using the following methods.

First, for a problem with $m$ objective $[f_1(\boldsymbol{x}), f_2(\boldsymbol{x}), ..., f_m(\boldsymbol{x})]$, we decompose the original multi-objective problem into $(m+1)$ single-objective problems by using the weighted sum scalarization:

$$g(\boldsymbol{x}) = w_1 f_1(\boldsymbol{x}) + w_2 f_2(\boldsymbol{x}) + ... + w_m f_m(\boldsymbol{x}) \tag{6}$$

where $w_1, w_2, ..., w_m$ are the elements of a given weight vector $\boldsymbol{w}$. In this paper, for a $m$-objective problem, we consider to use the $(m+1)$ weight vectors including: $m$ extreme weight vectors: $[\boldsymbol{w}_1 = (1, 0, ..., 0), \boldsymbol{w}_2 = (0, 1, ..., 0), ..., \boldsymbol{w}_m = (0, 0, ..., 1)]$, and one center weight vector: $\boldsymbol{w}_{m+1} = (1/m, 1/m, ..., 1/m)$. Then, we can obtain $(m+1)$ decomposed single-objective optimization problems. By solving each of these problems, we obtain $(m+1)$ heuristic solutions.

For MOTSP, we use a greedy algorithm to solve each decomposed single-objective problem as follows. The cost between two cities of a decomposed single-objective TSP is the weighted sum of the cost between them corresponding to each objective. For example, the cost between two cities $p$ and $q$ with a given weight vector $\boldsymbol{w} = (w_1, w_2, ..., w_m)$ is: $d^{\boldsymbol{w}}(p, q) = w_1 d_1(p, q) + w_2 d_2(p, q) + \ldots + w_m d_m(p, q)$, where $d_i(p, q), i = 1, 2, ..., m$ is the cost between two cities $p$ and

$q$ corresponding to the $i$-$th$ objective. From an arbitrary node as the starting point, we successively pick the node that results in the smallest weighted sum cost when moving from the previously selected node. We conduct the greedy algorithm by examining all nodes as the starting point. The resulting tour with the smallest cost is chosen as the heuristic solution corresponding to the given weight vector.

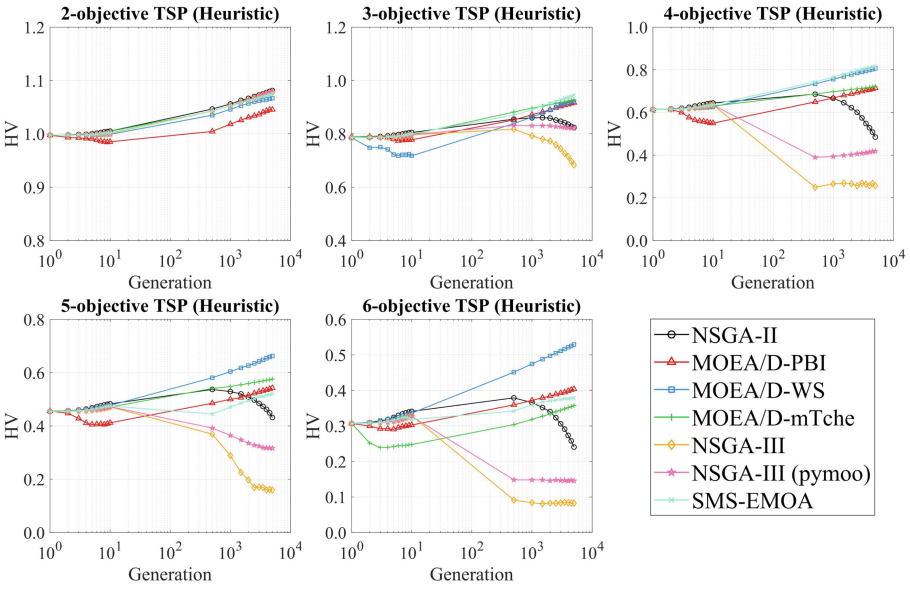



**Fig. 3.** Average HV for two- to six-objective TSP problems over 5000 generations, obtained from 21 runs. $(m+1)$ heuristic solutions together with other randomly generated solutions are used as the initial population.

For MOKP, given a weight vector $\boldsymbol{w}$, the original MOKP is decomposed into a single-objective KP with the optimization objective outlined in Eq. (2), subject to Eqs. (3)–(4), which can be viewed as an integer programming (IP) problem. In our experiments, we use MOSEK [1] as the solver based on the YALMIP [13] optimization framework to solve the formulated IP problem. The obtained solution is used as the heuristic solution to the single-objective KP.

### 3.2 Experimental Results

Figure 3 and Fig. 4 show the average HV values over 5000 generations for MOTSP and MOKP problems, respectively, with the $(m+1)$ heuristic solutions included in the initial population together with other randomly generated solutions for the seven compared MOEAs. We can see from the comparison with the previous

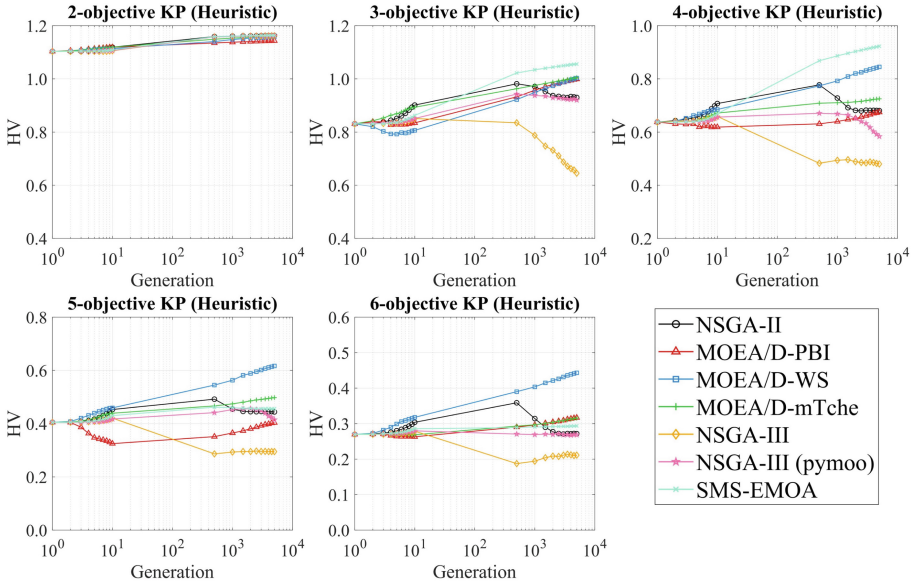



**Fig. 4.** Average HV for two- to six-objective KP problems over 5000 generations, obtained from 21 runs. $(m + 1)$ heuristic solutions together with other randomly generated solutions are used as the initial population.

results in Fig. 1 and Fig. 2 that initial populations with a few heuristic solutions have much higher HV values than with pure random initial solutions.

On the two-objective MOTSP and MOKP, no performance deterioration is observed for all the seven compared MOEAs. That is, the search mechanisms of these MOEAs can keep heuristic solutions in the current population when handling two-objective problems. However, when the number of objectives is more than two (i.e., three to six objectives), the achieved hypervolume performance of NSGA-II keeps decreasing after around 500 generations. That is, whereas NSGA-II can efficiently utilize heuristic solutions in early generations (since the HV performance is even better than other MOEAs in the early generations), it can delete good solutions in the middle of execution and deteriorate the quality of the current population by the generation updates even on three-objective problems. Similar performance deterioration is also observed for NSGA-III and NSGA-III (pymoo). However, for MOEA/D and SMS-EMOA, no such performance deterioration is observed. These observations indicate that certain mechanisms of NSGA-II (and NSGA-III) struggle to handle such multi-objective problems even in scenarios with only three objectives, which contradicts our intuition.

### 3.3 Search Behavior Analysis of NSGA-II

We first analyze the reason for the performance deterioration of NSGA-II as follows. At the environmental selection step of NSGA-II, we need to select $N$

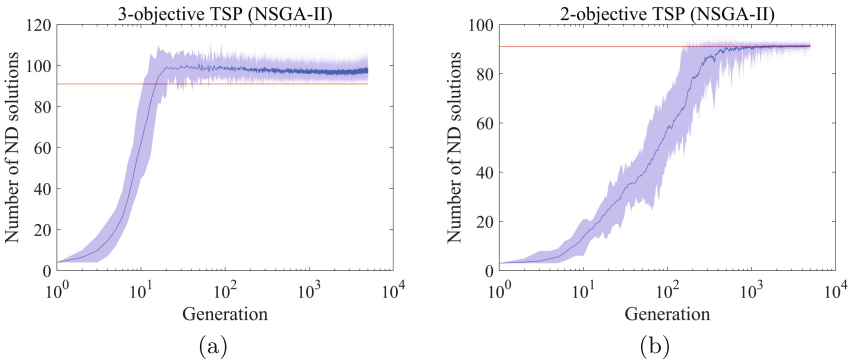



**Fig. 5.** Number of non-dominated solutions in the merged population across generations when using NSGA-II with heuristic solutions to solve three- and two-objective TSP. The red line indicates the population size, and the blue curve is the average result over 21 runs. (Color figure online)

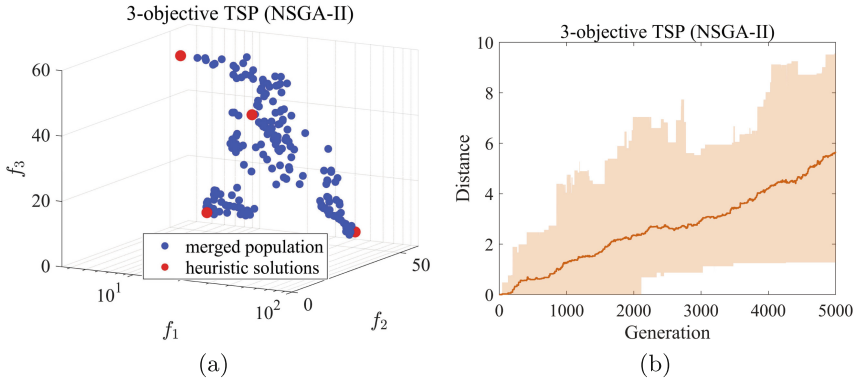
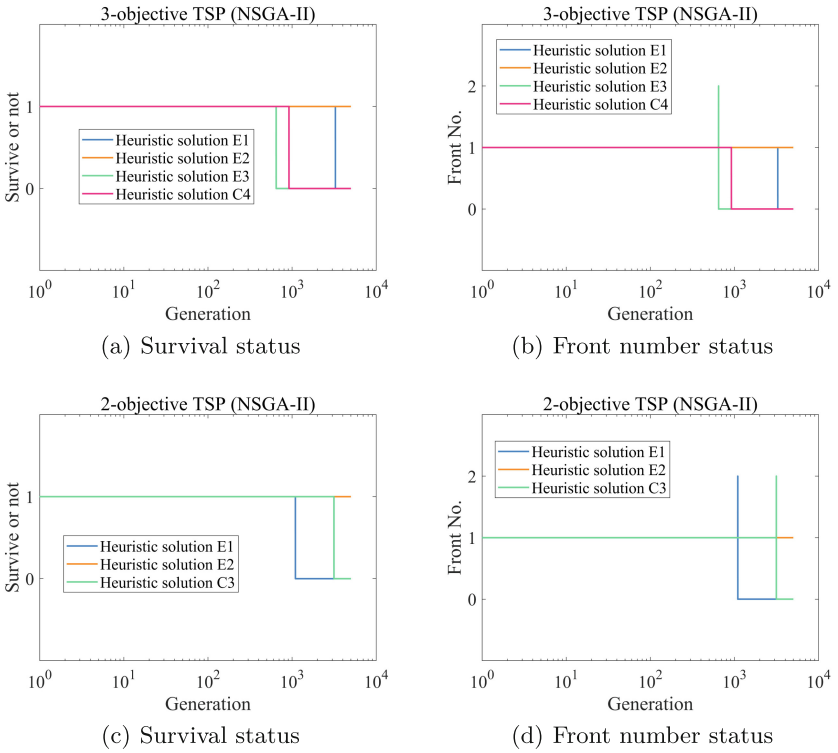



**Fig. 6.** (a) The solution distribution at the 1000th generation in a single run using NSGA-II with heuristic solutions to solve 3-objective TSP; (b) Average distance between the center heuristic solution and its closest solution in the merged population across generations.

solutions as the next population from the merged population with the size of $2N$ (i.e., the combination of the current population and offspring). Figure 5 (a) shows the number of non-dominated (ND) solutions in the merged population at different generations when using NGSA-II to solve the three-objective TSP in a single run. We can see that the number of non-dominated solutions quickly increases over 100% of the population size 91 (i.e., the red line in Fig. 5 (a)). Therefore, all these ND solutions reside in the first front as well as the last front, where no more solutions can be accommodated. Following the mechanism of NSGA-II, in this scenario, $N$ (i.e., 91) solutions are chosen from the ND solutions set completely depending on their crowding distance. If there exist

solutions that are close to the heuristic solutions, as shown in Fig. 6 (a), these heuristic solutions will have a small value of crowding distance which results in their elimination from the current population.

To further validate this explanation, we illustrate the survival status of the $(m+1)$ heuristic solutions and their corresponding front number (i.e., the front to which they belong) in Fig. 7 (a) and Fig. 7 (b), respectively. For example, in Fig. 7 (a), "1" means that the solution survives after the environmental selection and "0" otherwise. In Fig. 7 (a), a front number of "1" or "2" denotes that the solution is in the first or second front, respectively, while "0" indicates that the solution has been eliminated from the current population. In Fig. 7 (a), we can see that the extreme heuristic solutions (i.e., E1 and E3) and the center heuristic solution (i.e., C4) cannot survive in the current population after certain generations. In Fig. 7 (b), for heuristic solutions C4 and E1, their front numbers jump from "1"




(a) Survival status (b) Front number status

(c) Survival status (d) Front number status

**Fig. 7.** The survival status of heuristic solutions in the environmental selection of NSGA-II and their corresponding front number when solving three- and two-objective TSP in a single run. For the survival of heuristic solutions, "1" denotes that they survive in the environmental selection and "0" otherwise. For the front number, "1" and "2" denotes that the solution is in the first front and second front, respectively, and "0" denotes it is removed from the current population.

to "0". It means that these two solutions are eliminated since they have a worse crowding distance value. As for heuristic solution E3, its front number changes from "1" to "2", indicating a better solution that dominates it is produced which results in its elimination.

Since the good heuristic solutions E1 and C4 are removed from the current population with no better solutions produced as a replacement, the hypervolume of NSGA-II degrades as shown in Fig. 3. Actually, it is challenging to regenerate the center heuristic solution (or a similar good solution) once it has been eliminated. To demonstrate it, in Fig. 6 (b), we plot the distance between the center heuristic solution and its closet solution in the current population. We can see that the distance keeps increasing across the generations, which suggests that the solutions in the population have worse convergence towards the central Pareto front. It means that after the removal of the center heuristic solution, NSGA-II focuses solely on enhancing diversity without being able to improve convergence. All the performance deterioration of NSGA-II for four- to six-objective problems can also be attributed to these explanations.

As a comparison, in Fig. 5 (b), we plot the number of ND solutions in the merged population for using NSGA-II to solve two-objective TSP. We can see that the number of ND solutions is almost always smaller than the population size. In this scenario, heuristic solutions as ND solutions will survive until better solutions (i.e., solutions that dominate heuristic solutions) are generated. We also plot the survival curve and front number curve for two-objective TSP in a single run in Fig. 7 (c) and (d). We observe that the front numbers of heuristic solutions E1 and C3 jump from "1" to "2", which indicates that better solutions are generated to replace them. Although these two heuristic solutions cannot survive in the current population, the generated better solutions can serve as their replacements which explains why the hypervolume will not degrade.

### 3.4 Search Behavior Analysis of Other MOEAs

Similar to NSGA-II, NSGA-III (and NSGA-III (pymoo)) also experiences a performance deterioration as shown in Fig. 3 and Fig. 4. Here, we explain this phenomenon for NSGA-III as follows. Figure 8 (a) shows the number of non-dominated solutions across generations for NSGA-III on 3-objective TSP. We can observe that the number of ND solutions quickly increases over 100% of the population size. In this scenario, all these ND solutions including heuristic solutions compete for survival completely based on the niche-preservation operation proposed in NSGA-III. More specifically, first, a set of reference vectors is generated by using Das and Dennis's approach [6]. Since currently each reference vector has no associated solutions, a random reference vector is selected and the solution that has the shortest perpendicular distance to it will be added to the current population. This process is repeated until a total of $N$ solutions are selected. Therefore, there is a high possibility that the heuristic solutions cannot survive when they have a large perpendicular distance to the reference vectors. Figure 9 (a) and (b) show the survival curve and front number curve of using NSGA-III to solve three-objective TSP in a single run, respectively. We

can observe that all the heuristic solutions E1-3 and C4 cannot survive after certain generations. Also, their front numbers jump from "1" to "0", which indicates these heuristic solutions are removed from the current population without any better solutions being generated to take their place. The removal of these heuristic solutions without replacement makes it difficult to reproduce such good solutions when losing their important domain knowledge. This directly leads to the performance deterioration of NSGA-III.

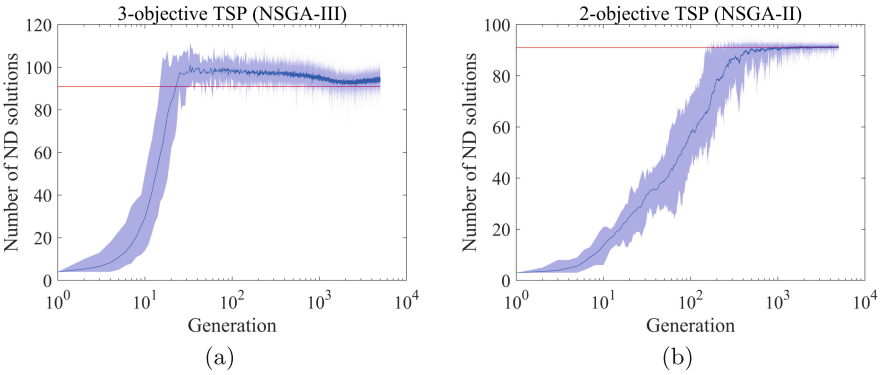



**Fig. 8.** Number of non-dominated solutions in the merged population across generations when using NSGA-III with heuristic solutions to solve three- and two-objective TSP.

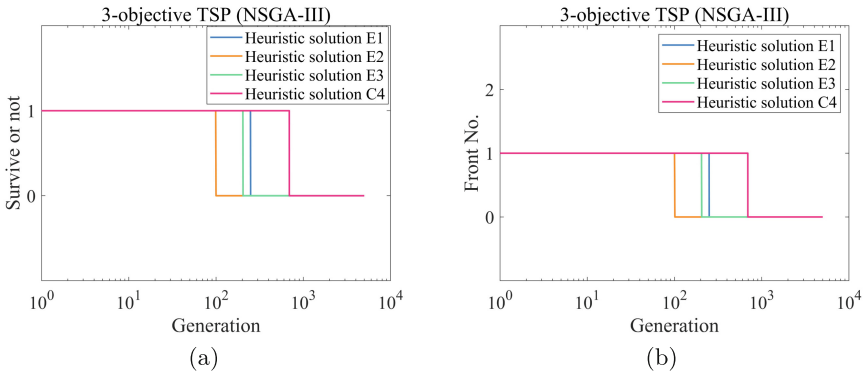



**Fig. 9.** The survival status of heuristic solutions in the environmental selection of NSGA-III and their corresponding front number when solving three-objective TSP in a single run.

Another interesting phenomenon is that NSGA-III shows an earlier performance deterioration than NSGA-II. This is because the extreme heuristic

solutions are more likely to be removed from the current population in earlier generations when using NSGA-III compared to NSGA-II, as shown in the comparison between Fig. 9 (a) and Fig. 7 (a). The high possibility of the removal of the extreme heuristic solutions for NSGA-III in earlier generations is also rational. As we have explained, the survivals of these heuristic solutions in NSGA-III completely depend on their perpendicular distances to reference vectors. Since the extreme heuristic solutions are generated by solving the weighted sum scalarization in Eq. (6) with $m$ extreme weight vectors: $[\boldsymbol{w}_1 = (1, 0, ..., 0), \boldsymbol{w}_2 = (0, 1, ..., 0), ..., \boldsymbol{w}_m = (0, 0, ..., 1)]$, they locate at the $m$ corners of an inverted triangular shape in the objective space. The reference vectors, however, are generated using Das and Dennis's approach, which samples from a triangular shape. As a result, there is a significant distance between the extreme heuristic solutions and the reference vectors, leading to the removal of heuristic solutions from the current population in the earlier generations.

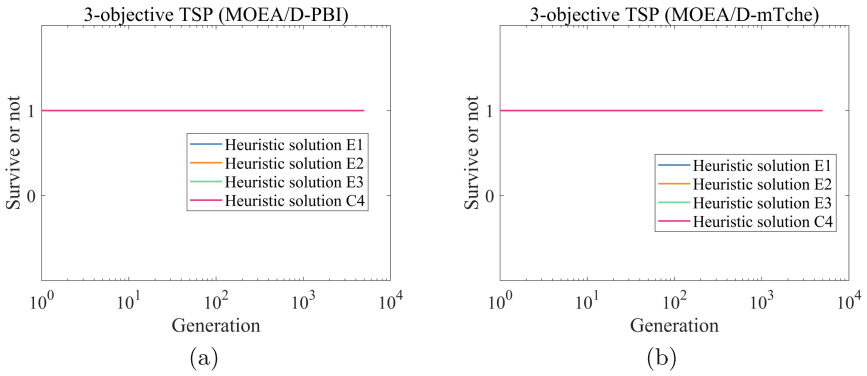



**Fig. 10.** The survival status of heuristic solutions in the environmental selection of MOEA/D-PBI and MOEA/D-mTche and their corresponding front number when solving three-objective TSP in a single run.

For MOEA/D with different scalarization functions, we observe no such performance deterioration. It indicates that MOEA/D can keep heuristic solutions in the current population and utilize them. For example, Fig. 10 shows the survival status when using MOEA/D-PBI and MOEA/D-mTche to solve three-objective TSP in a single run. We can see that all the heuristic solutions always survive in the current population. As we have explained for NSGA-II and NSGA-III, their non-dominated sorting mechanism will not work when the number of ND solutions exceeds the population size. However, the survival of solutions in MOEA/D depends on their scalarization values. Since the scalarization calculation of a solution considers both the convergence of the solution to the Pareto front and its distance to the reference vectors, the heuristic solutions will always have better scalarization values since they have very good convergence towards the Pareto front, which leads to their survival in the current population. In

other words, the environmental selection mechanism of MOEA/D will always consider the dominance relationships between solutions, even when the number of non-dominated solutions exceeds the population size.

For SMS-EMOA, its environmental selection mechanism ensures that solutions with larger hypervolume contributions survive. Consequently, its hypervolume performance consistently increases through the generation updates. In other words, the heuristic solutions can always survive until many solutions that have larger hypervolume contributions than them are generated.

## 4 Conclusion

In this paper, we demonstrated that the number of generations has much larger effects on the performance of NSGA-II on multi- and many-objective combinatorial optimization problems than the number of objectives. When we used a random initial population including a few heuristic solutions, NSGA-II showed better performance than other MOEAs in early generations (e.g., at the 100th generation) even for six-objective problems. In later generations, NSGA-II continued to deteriorate the quality of the current population. Finally, it showed poor performance at the final generation (10,000th generation) even for three-objective problems. NSGA-III also showed a similar performance deterioration. Our detailed search behavior analysis of NSGA-II found the following reasons for the above-mentioned observations. In early generations, heuristic initial solutions are not removed from the population since they are non-dominated and the number of non-dominated solutions in the merged current and offspring population is smaller than the population size. As a result, good solutions are generated from the heuristic initial solutions. Then, at some generations, the number of non-dominated solutions becomes larger than the population size. As a result, the selection of solutions for the next generation is mainly based on the crowding distance, which can remove good solutions (and well-converged solutions) from the population. This leads to continuous performance deterioration in later generations even in the case of three objectives. A similar explanation can be given to the performance deterioration of NSGA-III.

Our experimental results clearly demonstrated that the consensus in the EMO community about the performance of dominance-based MOEAs is not always applicable to multi-objective combinatorial optimization. Our observations can be used for the design of practically useful high-performance MOEAs for multi-objective combinatorial optimizations. This is because combinatorial optimization problems often have domain-specific heuristics to quickly generate good initial solutions. This is also because real-world problems are often expensive (i.e., a small number of generations is often used as a termination condition). One future research topic is to improve the performance of NSGA-II and NSGA-III by preventing their performance deterioration in later generations. We may need some additional mechanism to assign higher fitness to well-converged non-dominated solutions than other non-dominated solutions. Another future research topic is to utilize the high performance of NSGA-II in early generations in order to improve the performance of other MOEAs.

**Acknowledgments.** This work was supported by the National Key R&D Program of China (Grant No. 2023YFE0106300), National Natural Science Foundation of China (Grant No. 62250710163, 62376115, 62276223), Guangdong Provincial Key Laboratory (Grant No. 2020B121201001), and the Research Grants Council of the Hong Kong Special Administrative Region, China [GRF Project No. CityU 11215622].

**Disclosure of Interests.** The authors have no competing interests to declare that are relevant to the content of this article.

## References

1. ApS, M.: The MOSEK optimization toolbox for MATLAB manual. Version 9.0. (2019). http://docs.mosek.com/9.0/toolbox/index.html
2. Bader, J., Zitzler, E.: Hype: an algorithm for fast hypervolume-based many-objective optimization. Evol. Comput. **19**(1), 45–76 (2011)
3. Beume, N., Naujoks, B., Emmerich, M.: SMS-EMOA: multiobjective selection based on dominated hypervolume. Eur. J. Oper. Res. **181**(3), 1653–1669 (2007)
4. Blank, J., Deb, K.: Pymoo: multi-objective optimization in Python. IEEE Access **8**, 89497–89509 (2020)
5. Chen, T., Li, M., Yao, X.: Standing on the shoulders of giants: seeding search-based multi-objective optimization with prior knowledge for software service composition. Inf. Softw. Technol. **114**, 155–175 (2019)
6. Das, I., Dennis, J.E.: Normal-boundary intersection: a new method for generating the pareto surface in nonlinear multicriteria optimization problems. SIAM J. Optim. **8**(3), 631–657 (1998)
7. Deb, K., Jain, H.: An evolutionary many-objective optimization algorithm using reference-point-based nondominated sorting approach, part I: solving problems with box constraints. IEEE Trans. Evol. Comput. **18**(4), 577–601 (2013)
8. Deb, K., Pratap, A., Agarwal, S., Meyarivan, T.: A fast and elitist multiobjective genetic algorithm: NSGA-II. IEEE Trans. Evol. Comput. **6**(2), 182–197 (2002)
9. Gong, C., Nan, Y., Pang, L.M., Ishibuchi, H., Zhang, Q.: Initial populations with a few heuristic solutions significantly improve evolutionary multi-objective combinatorial optimization. In: 2023 IEEE Symposium Series on Computational Intelligence (SSCI), pp. 1398–1405. IEEE (2023)
10. Gong, C., Nan, Y., Pang, L.M., Zhang, Q., Ishibuchi, H.: Effects of including optimal solutions into initial population on evolutionary multiobjective optimization. In: Proceedings of the Genetic and Evolutionary Computation Conference, pp. 661–669 (2023)
11. Gong, C., Nan, Y., Pang, L.M., Zhang, Q., Ishibuchi, H.: Performance of NSGA-III on multi-objective combinatorial optimization problems heavily depends on its implementations. In: Proceedings of the Genetic and Evolutionary Computation Conference 2024 (2024, Accepted)
12. Ishibuchi, H., Akedo, N., Nojima, Y.: Behavior of multiobjective evolutionary algorithms on many-objective knapsack problems. IEEE Trans. Evol. Comput. **19**(2), 264–283 (2015)
13. Lofberg, J.: YALMIP: a toolbox for modeling and optimization in MATLAB. In: 2004 IEEE International Conference on Robotics and Automation (IEEE Cat. No. 04CH37508), pp. 284–289. IEEE (2004)

14. Mostaghim, S., Schmeck, H.: Distance based ranking in many-objective particle swarm optimization. In: Rudolph, G., Jansen, T., Beume, N., Lucas, S., Poloni, C. (eds.) PPSN 2008. LNCS, vol. 5199, pp. 753–762. Springer, Heidelberg (2008). https://doi.org/10.1007/978-3-540-87700-4_75
15. Schutze, O., Lara, A., Coello, C.A.C.: On the influence of the number of objectives on the hardness of a multiobjective optimization problem. IEEE Trans. Evol. Comput. **15**(4), 444–455 (2011)
16. Tian, Y., Cheng, R., Zhang, X., Jin, Y.: PlatEMO: a MATLAB platform for evolutionary multi-objective optimization [educational forum]. IEEE Comput. Intell. Mag. **12**(4), 73–87 (2017)
17. Wagner, T., Beume, N., Naujoks, B.: Pareto-, aggregation-, and indicator-based methods in many-objective optimization. In: Obayashi, S., Deb, K., Poloni, C., Hiroyasu, T., Murata, T. (eds.) EMO 2007. LNCS, vol. 4403, pp. 742–756. Springer, Heidelberg (2007). https://doi.org/10.1007/978-3-540-70928-2_56
18. Zhang, Q., Li, H.: MOEA/D: a multiobjective evolutionary algorithm based on decomposition. IEEE Trans. Evol. Comput. **11**(6), 712–731 (2007)
19. Zitzler, E., Laumanns, M., Thiele, L.: SPEA2: improving the strength pareto evolutionary algorithm. TIK Report **103** (2001)
20. Zitzler, E., Thiele, L.: Multiobjective evolutionary algorithms: a comparative case study and the strength pareto approach. IEEE Trans. Evol. Comput. **3**(4), 257–271 (1999)
21. Zitzler, E., Thiele, L., Laumanns, M., Fonseca, C.M., Da Fonseca, V.G.: Performance assessment of multiobjective optimizers: an analysis and review. IEEE Trans. Evol. Comput. **7**(2), 117–132 (2003)

# Many-Objective Cover Problem: Discovering Few Solutions to Cover Many Objectives

Yilu Liu[1,2(✉)], Chengyu Lu[1,2], Xi Lin[1,2], and Qingfu Zhang[1,2(✉)]

[1] Department of Computer Science, City University of Hong Kong, Kowloon, Hong Kong
{yilu.liu,chengyulu3-c,xi.lin}@my.cityu.edu.hk, qingfu.zhang@cityu.edu.hk
[2] City University of Hong Kong Shenzhen Research Institute, Shenzhen, China

**Abstract.** Many-objective optimization (MaO) is a basic issue in various research areas. Although Pareto optimality is a common criterion for MaO, it may bring many troubles when facing a huge number (e.g., up to 100) of objectives. This paper provides a new perspective on MaO by introducing a many-objective cover problem (MaCP). Given $m$ objectives, MaCP aims to find a solution set with size $k$ ($1 < k \ll m$) to cover all objectives (i.e., each objective can be approximately optimized by at least one solution in this set). We prove the NP-hard property of MaCP and develop a clustering-based swarm optimizer (CluSO) with a convergence guarantee to tackle MaCP. Then, we propose a decoupling many-objective test suite (DC-MaTS) with practical significance and use it to evaluate CluSO. Extensive experimental results on various test problems with up to 100 objectives demonstrate both the efficiency and effectiveness of CluSO, while also illustrating that MaCP is a feasible perspective on MaO.

**Keywords:** Multi-objective optimization · Many-objective optimization · Particle swarm optimization · Clustering

## 1 Introduction

In many real-world problems such as drug design [26], route planning [34], and data mining [33], it is very often that multiple objectives should be optimized simultaneously. Generally, these problems with multiple objectives to be optimized are termed multi-objective optimization problems (MOPs) [48]. If an MOP has more than three objectives, it can be further referred to as a many-objective optimization problem (MaOP) [28]. Due to the complicated relationships between different objectives, how to effectively solve MaOPs has become an emerging issue, which is often referred to as many-objective optimization (MaO) [28].

Currently, an MaOP is often tackled by finding a solution set using Pareto optimality [4]. The solutions in this set should be non-dominated to each other (i.e., for any two solutions in the set, one is superior to the other in at least one objective), and the objective values of all solutions in this set should approximate


M. Affenzeller et al. (Eds.): PPSN 2024, LNCS 15151, pp. 68–82, 2024.
https://doi.org/10.1007/978-3-031-70085-9_5

or reach the whole Pareto front (PF). Based on this idea, numerous algorithms have been designed in recent years, and many-objective evolutionary algorithms (MaOEAs) [28] have become the mainstream methodologies.

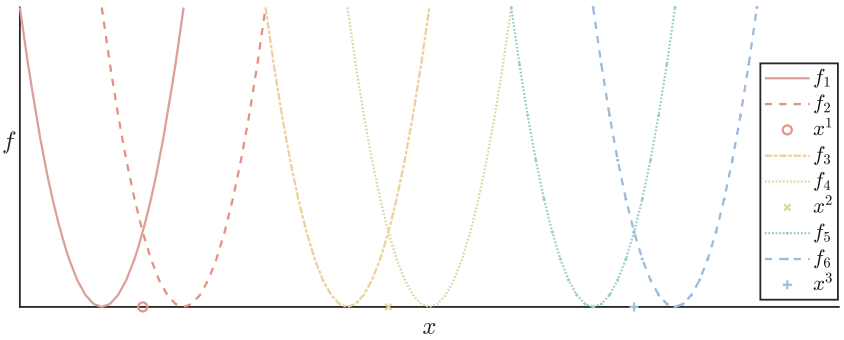



**Fig. 1.** An illustration of MaCP with $m = 6$ and $k = 3$. $\{f_1, f_2, f_3, f_4, f_5, f_6\}$ are six objectives desired to be minimized simultaneously, and a solution set $X = \{x^1, x^2, x^3\}$ can be discovered to cover all objectives.

However, when the number of objectives is extremely large (e.g., up to 100), the dimensionality of PF can be as large as $(m-1)$ [46], where $m$ is the number of objectives. Thus, using Pareto optimality may generate a tremendous number of non-dominated solutions, which easily brings the following three troubles [28]:

- The whole PF can be difficult to reach or approximate.
- The computational overhead can be very high.
- The selection of suitable solutions can be troublesome for decision-makers.

In view of these potential troubles, how to effectively solve MaOPs with a huge number of objectives remains a pending issue [28]. Therefore, there is a need to develop a novel methodology that can effectively tackle numerous objectives while directly providing decision-makers with a desirable number of solutions.

To this end, we introduce the idea of cover to MaO and propose a many-objective cover problem (MaCP). Similar to many classical covering problems such as set cover [21] and vertex cover [15], MaCP aims to discover a solution set with size $k$ $(1 < k \ll m)$ to cover all objectives (i.e., each objective can be approximately optimized by at least one solution in this set). Figure 1 gives an illustration of MaCP with $m = 6$ and $k = 3$, where the six objectives $\{f_1, f_2, f_3, f_4, f_5, f_6\}$ are desired to be minimized simultaneously, and a solution set $X = \{x^1, x^2, x^3\}$ can be found to cover all objectives (i.e., $\{f_1, f_2\}$, $\{f_3, f_4\}$, and $\{f_5, f_6\}$ can be approximately optimized by $x^1$, $x^2$, and $x^3$, respectively). Therefore, as long as MaCP is well addressed, we can directly provide decision-makers with a desirable number of solutions when facing many objectives.

We provide an explicit mathematical definition for MaCP and prove its NP-hard property. Then, We develop a clustering-based swarm optimizer (CluSO)

to solve MaCP and demonstrate its convergence property. Inspired by the idea of particle swarm optimization (PSO) [25], CluSO classifies all objectives into different clusters and iteratively optimizes a particle swarm under the guidance of the current best solution set and the global best solution set. When reaching the maximal number of iterations, the global best solution set is returned as the final solution set for MaCP.

While CluSO is developed to optimize many objectives, we argue that most existing many-objective test suites (MaTSs) are perhaps not suitable for validating its performance. A major reason is that, despite their arbitrarily large number of objectives, these test problems can be well addressed by a small number of solutions. Some compelling examples can be observed in DTLZ [14]. For DTLZ1-6, the first $(m-1)$ objectives can be optimized by a solution $\boldsymbol{x} = [x_1, ..., x_d]^T$ with $x_1 = 0$ or 1, where $d$ is the problem dimension. For DTLZ7, the first $(m-1)$ objectives can be optimized by a solution $\boldsymbol{x} = [0, ..., 0, x_d]^T$. Thus, if we aim to cover all the objectives in these MaTSs, only two solutions are enough (i.e., one for the first $(m-1)$ objectives, and another for the last one). Similar phenomena can also be found in other common MaTSs such as WFG [22] and MaF [9], which shows that they are unsuitable for evaluating CluSO. In fact, in many real-world scenarios, the optimal solutions of different objectives are often different [12,17,24], which motivates us to propose an new MaTS, termed decoupling MaTS (DC-MaTS). For each test problem in DC-MaTS, the optimal solution of each objective is different to each other, which makes it practically significant and very suitable for evaluating CluSO.

The contributions of this paper are summarized as follows.

- We provide a novel perspective on many-objective optimization (MaO) by introducing a many-objective cover problem (MaCP), which aims to directly provide decision-makers with a desirable number of solutions when facing many objectives.
- We prove the NP-hard property of MaCP and develop a clustering-based swarm optimizer (CluSO) with a convergence guarantee to solve MaCP.
- We propose a decoupling many-objective test suite (DC-MaTS) with practical significance and use it to evaluate CluSO. Extensive experimental results on various test problems with up to 100 objectives demonstrate both the efficiency and effectiveness of CluSO, while also illustrating that MaCP is a feasible perspective on MaO.

## 2 Related Work

### 2.1 Many-Objective Optimization

Currently, Pareto optimality is the main criterion for MaOPs, and MaOEAs are the mainstream methodologies. Based on Pareto optimality, numerous methods have been developed to enhance the capabilities of MaOEAs in solving MaOPs, which can be classified into three types [28]: dominance-based, decomposition-based, and indicator-based methods. Dominance-based methods mainly enhance

the selection pressure on non-dominated solutions by modified dominance relationships such as $\epsilon$-dominance [27], $\theta$-dominance [44], and grid-dominance [43]. Decomposition-based methods focus on decomposing MaOPs into multiple subproblems for optimization using techniques such as aggregation functions [45], reference points [13], and reference vectors [8]. Indicator-based methods, the third category, guide the selection of solutions using performance indicators such as hypervolume [20], R2 [3], and inverted generational distance (IGD) [23].

While these methods can effectively improve the diversity and convergence of solutions, they still face significant challenges such as computational cost and decision-making for selecting solutions. To enhance computational efficiency, some methods attempt to introduce surrogate models [10,37,47]. Additionally, to assist decision-makers in selecting solutions, some methods explore further subset selection from the obtained solution set [5,19,40]. However, it is still difficult for these methods to deal with a huge number (e.g., up to 100) of objectives. Therefore, we propose introducing MaCP as a new perspective on MaO, aiming to effectively tackle numerous objectives while directly providing decision-makers with a desirable number of solutions.

### 2.2 Covering Problems

In many real-world scenarios, it is often necessary to select a representative part of data from a large collection to fulfill specific objectives, which has led to the development of various covering problems. Classical covering problems mainly focus on two types of data structures: sets and graphs [38]. The main idea of these problems is to find a subset or subgraph to satisfy some covering objectives, such as minimum set cover [21], maximum set packing [41], minimum hitting set [39], minimum vertex cover [15], maximum clique [42], and minimum dominating set [18]. Building upon these fundamental problems, numerous practical applications have been proposed and studied, including facility location [36], sensor placement [35], and influence maximization [31]. Owing to the NP-hard nature of these problems and the impracticality of exhaustive search, many heuristic methods have been developed to address them [31,35,36].

In this paper, we serve many objectives using a small number of solutions, which aligns well with the aims of these covering problems. Thus, we introduce the idea of cover to MaO by proposing MaCP.

## 3 Problem Formulation

Similar to the definitions of many covering problems, we explicitly define MaCP as the following form.

**Definition 1 (Many-Objective Cover Problem).** *Given a set of $m$ ($m \geq 4$) objectives $F = \{f_1(\boldsymbol{x}), f_2(\boldsymbol{x}), \ldots, f_m(\boldsymbol{x})\}$ with $\boldsymbol{x} \in \mathcal{D} \subset \mathbb{R}^d$ ($\mathcal{D}$ is the decision space) and an integer $k$ with $1 < k \ll m$, MaCP aims to find a set of $k$ solutions*

$X = \{\boldsymbol{x}^1, \boldsymbol{x}^2, \ldots, \boldsymbol{x}^k\} \subset \mathcal{D}$ *to minimize the following objective function:*

$$\mathcal{G}(X) = \sum_{i=1}^{m} min_{1 \leq j \leq k} f_i(\boldsymbol{x}^j). \tag{1}$$

Similar to many MaO studies [28], we consider bounded continuous $\mathcal{D}$ in this paper. When $k = 1$, MaCP reduces to the simple aggregation with equal weights on all objectives. When $k = m$, it is to find the best solution for each objective independently. Thus, we consider the non-trivial case of $1 < k \ll m$. Moreover, a solution set for MaCP is called a many-objective cover set (MaCS). Next, we demonstrate the complexity of MaCP.

**Theorem 1 (NP-hard Property).** *MaCP is NP-hard.*

*Proof.* We prove this theorem by reducing MaCP to another NP-hard problem. Specifically, We consider an instance of MaCP by setting $f_i(\boldsymbol{x})$ as

$$f_i(\boldsymbol{x}) = \|\boldsymbol{v}_i - \boldsymbol{x}\|^2, \tag{2}$$

where $\boldsymbol{v}_i \in \mathcal{D}$ is a constant vector and $\|\boldsymbol{v}_i - \boldsymbol{x}\|^2$ denotes the squared Euclidean distance between $\boldsymbol{v}_i$ and $\boldsymbol{x}$. Thus, the objective function of this instance is

$$\mathcal{G}'(X) = \sum_{i=1}^{m} min_{1 \leq j \leq k} \|\boldsymbol{v}_i - \boldsymbol{x}^j\|^2. \tag{3}$$

From the form of $\mathcal{G}'(X)$, we can find that the aim of this instance is to find $k$ centers $\{\boldsymbol{x}^1, \boldsymbol{x}^2, \ldots, \boldsymbol{x}^k\}$ for $m$ points $\{\boldsymbol{v}_1, \boldsymbol{v}_2, \ldots, \boldsymbol{v}_m\}$ such that each point can be assigned to the optimal center, which is consistent with the aim of discrete clustering problem (DCP) [16]. Since DCP is NP-hard when $k > 1$ [16], we can prove the NP-hard property of MaCP and conclude the proof of Theorem 1.

## 4 Algorithm Design

In MaCP, each solution $\boldsymbol{x}^j$ is responsible for optimizing at least one objective, and the objectives optimized by $\boldsymbol{x}^j$ can be regarded as an objective cluster. Thus, we can use $C = \{c^1, c^2, \ldots, c^k\}$ to denote the set of objective clusters, where $c^j$ includes the objectives optimized by $\boldsymbol{x}^j$. Then, MaCP can be regarded as detecting a suitable $C$ for all objectives and finding the optimal solution from each cluster. Based on this idea, CluSO is designed to address MaCP, of which the pseudo-code is provided in Algorithm 1. At a high level, it mainly contains two phases: initialization (lines 1–4) and iterative optimization (lines 5–10).

In the first phase, a particle swarm $\Omega = \{\omega_1, \omega_2, \ldots, \omega_s\}$ is randomly initialized. Each particle $\omega_l \in \Omega$ $(l = 1, 2, \ldots, s)$ holds a position vector $\boldsymbol{p}^l \in \mathbb{R}^d$ and a velocity vector $\boldsymbol{v}^l \in \mathbb{R}^d$. The position vector denotes a candidate solution, while the velocity vector indicates a potential optimization direction. For convenience,

we use $P = \{\boldsymbol{p}^1, \boldsymbol{p}^2, \ldots, \boldsymbol{p}^s\}$ and $V = \{\boldsymbol{v}^1, \boldsymbol{v}^2, \ldots, \boldsymbol{v}^s\}$ to record the sets of position and velocity vectors, respectively. Then, $\boldsymbol{e}_i = \left[f_i(\boldsymbol{p}^1), f_i(\boldsymbol{p}^2), \ldots, f_i(\boldsymbol{p}^s)\right]^T$ is used to denote the objective embedding regarding $f_i$, as the $f_i$ values of these $s$ solutions can directly reflect some properties of $f_i$. Thus, we can obtain a set of objective embeddings $E = \{\boldsymbol{e}_1, \boldsymbol{e}_2, \ldots, \boldsymbol{e}_m\}$. After that, regarding the objective embeddings as the features for all objectives, the k-means clustering algorithm [1] is leveraged to classify all objectives into $k$ clusters and obtain the set of objective clusters $C = \{c^1, c^2, ..., c^k\}$. Next, two important solution sets, $X^{cur}$ and $X^{glo}$, are initialized as $X^{cur} = X^{glo} = \{\boldsymbol{q}^1, \boldsymbol{q}^2, \ldots, \boldsymbol{q}^k\}$, where $\boldsymbol{q}^j = argmin_{\boldsymbol{p}^l \in P} \sum_{f_i \in c^j} f_i(\boldsymbol{p}^l)$. Among them, $X^{cur}$ denotes the best MaCS in current $P$ (termed current best MaCS), while $X^{glo}$ represents the best MaCS found so far (termed global best MaCS).

In the second phase, $P$, $V$, $X^{cur}$, $X^{glo}$, and $C$ will experience an iterative optimization process. When the number of iterations reaches $T$, CluSO stops and returns $X^{glo}$ as the final obtained MaCS. In the following sections, we will demonstrate the iterative optimization process in detail and theoretically analyze the convergence property of CluSO.

**Algorithm 1:** CluSO

**Input:** objective number: $m$; problem dimension: $d$, solution number: $k$; swarm size: $s$; iteration number: $T$; learning parameters: $\alpha$, $\beta$;
**Output:** global best MaCS: $X^{glo}$.

Randomly initialize the sets of position and velocity vectors: $P, V \leftarrow InitPV(s, d)$;
Get the set of objective embeddings based on the objective values of all position vectors in $P$: $E \leftarrow ObjEmb(P, m)$;
Initialize the set of objective clusters using the k-means clustering algorithm: $C \leftarrow ObjClu(E, k)$;
Initialize the current best and global best MaCSs: $X^{cur}, X^{glo} \leftarrow InitX(P, C)$;
**for** $t = 1$ **to** $T$ **do**
    Update $P$ and $V$: $P, V \leftarrow UpdatePV(P, V, X^{cur}, X^{glo}, C, \alpha, \beta)$;
    Update $X^{cur}$: $X^{cur} \leftarrow UpdateXC(P, C)$;
    Update $X^{glo}$: $X^{glo} \leftarrow UpdateXG(X^{cur}, X^{glo})$;
    Update $C$: $C \leftarrow UpdateC(X^{glo})$;
**end**

### 4.1 Iterative Optimization

We first introduce the optimization process for $P$ and $V$. In the original single-objective PSO framework [25], each particle uses the personal best position and the global best position to guide the optimization for its position and velocity vectors. However, in CluSO, the solutions in $X^{cur}$ and $X^{glo}$ are responsible

for optimizing different objective clusters, and the performance of particles on different objective clusters also varies. Thus, for each particle, we record the objective cluster that achieves the best performance and guide the optimization of its position and velocity vectors using the solutions that are responsible for optimizing this objective cluster in $X^{cur}$ and $X^{glo}$. Specifically, for particle $\omega_l$, the most suitable solutions in $X^{cur}$ and $X^{glo}$ for guiding the optimization of $\boldsymbol{p}^l$ and $\boldsymbol{v}^l$ are $\boldsymbol{x}^{cur,\varphi}$ and $\boldsymbol{x}^{glo,\varphi}$, respectively, where $\varphi = argmin_{1\leq j\leq k} \sum_{f_i\in c^j} f_i(\boldsymbol{p}^l)$. Thereby, similar to the original single-objective PSO framework [25], $\boldsymbol{p}^l$ and $\boldsymbol{v}^l$ will be updated according to the following rules:

$$\begin{aligned} \boldsymbol{v}^l &\leftarrow \boldsymbol{r}_1\boldsymbol{v}^l + \alpha\boldsymbol{r}_2(\boldsymbol{x}^{cur,\varphi} - \boldsymbol{p}^l) + \beta\boldsymbol{r}_3(\boldsymbol{x}^{glo,\varphi} - \boldsymbol{p}^l), \\ \boldsymbol{p}^l &\leftarrow \boldsymbol{p}^l + \boldsymbol{v}^l, \end{aligned} \tag{4}$$

where $\boldsymbol{r}_1, \boldsymbol{r}_2, \boldsymbol{r}_3 \in [0,1]^d$ are three random vectors; $\alpha, \beta \in (0, +\infty)$ are two human-determined learning parameters. Then, after the update of $P$, $X^{cur}$ will be updated as

$$X^{cur} \leftarrow \left\{\boldsymbol{x}^{cur,1}, \boldsymbol{x}^{cur,2}, \ldots, \boldsymbol{x}^{cur,k}\right\}, \tag{5}$$

where $\boldsymbol{x}^{cur,j} = argmin_{\boldsymbol{p}^l\in P} \sum_{f_i\in c^j} f_i(\boldsymbol{p}^l)$. Next, $X^{glo}$ will be updated as the one in $X^{cur}$ or $X^{glo}$ that has better objective function value, which means that

$$X^{glo} \leftarrow \text{argmin}_{X\in\{X^{cur}, X^{glo}\}} \mathcal{G}(X). \tag{6}$$

Finally, $X^{glo} = \left\{\boldsymbol{x}^{glo,1}, \boldsymbol{x}^{glo,2}, \ldots, \boldsymbol{x}^{glo,k}\right\}$ will be further used to update the objective clusters in $C$. Specifically, for $c^j$, it will be updated by

$$c^j \leftarrow \left\{f_i | f_i(\boldsymbol{x}^{glo,j}) \leq f_i(\boldsymbol{x}^{glo,j'}), 1 \leq j' \leq k\right\}. \tag{7}$$

### 4.2 Convergence Analysis

**Theorem 2 (Convergence Property).** *In CluSO, $P$, $V$, $X^{cur}$, $X^{glo}$, and $C$ will converge to equilibria.*

*Proof.* We first demonstrate the convergence of $P$ and $V$. Given that the update of position and velocity vectors in $P$ and $V$ constantly obeys the rules in (4), we use the expected value of $\boldsymbol{r}_1, \boldsymbol{r}_2, \boldsymbol{r}_3$ to rewrite (4) as the following form:

$$\begin{aligned} \boldsymbol{v}^l &\leftarrow 0.5\boldsymbol{v}^l + 0.5\alpha(\boldsymbol{x}^{cur,\varphi} - \boldsymbol{p}^l) + 0.5\beta(\boldsymbol{x}^{glo,\varphi} - \boldsymbol{p}^l), \\ \boldsymbol{p}^l &\leftarrow \boldsymbol{p}^l + \boldsymbol{v}^l. \end{aligned} \tag{8}$$

Let $\boldsymbol{p}^l(t)$ and $\boldsymbol{v}^l(t)$ denote the state of $\boldsymbol{p}^l$ and $\boldsymbol{v}^l$ in iteration $t$, respectively. Then, (8) can be further rewritten as

$$\begin{aligned} \boldsymbol{v}^l(t+1) &= 0.5\boldsymbol{v}^l(t) + \eta\left(\boldsymbol{\rho} - \boldsymbol{p}^l(t)\right), \\ \boldsymbol{p}^l(t+1) &= \boldsymbol{p}^l(t) + \boldsymbol{v}^l(t+1), \end{aligned} \tag{9}$$

where

$$\begin{aligned} \eta &= 0.5(\alpha + \beta), \\ \boldsymbol{\rho} &= \frac{\alpha}{\alpha + \beta} \boldsymbol{x}^{cur,\varphi}(t) + \frac{\beta}{\alpha + \beta} \boldsymbol{x}^{glo,\varphi}(t). \end{aligned} \tag{10}$$

From the view of dynamical system, (9) can be simplified to the following dynamical equation:

$$\mathbf{Y}(t+1) = \mathbf{A}\mathbf{Y}(t) + \mathbf{B}, \tag{11}$$

where

$$\mathbf{Y}(t) = \begin{bmatrix} \boldsymbol{v}^l(t) \\ \boldsymbol{p}^l(t) \end{bmatrix}, \mathbf{A} = \begin{bmatrix} 0.5 & -\eta \\ 0.5 & 1-\eta \end{bmatrix}, \mathbf{B} = \begin{bmatrix} \eta\boldsymbol{\rho} \\ \eta\boldsymbol{\rho} \end{bmatrix}. \tag{12}$$

Then, the eigenvalues of $\mathbf{A}$, termed $\sigma$, satisfy

$$\sigma^2 - (1.5 - \eta)\sigma + 0.5 = 0, \tag{13}$$

where

$$\begin{aligned} \sigma_1 &= 0.75 - 0.5\eta + 0.5\sqrt{(1.5-\eta)^2 - 2}, \\ \sigma_2 &= 0.75 - 0.5\eta - 0.5\sqrt{(1.5-\eta)^2 - 2}. \end{aligned} \tag{14}$$

Since $\eta = 0.5(\alpha + \beta) > 0$, we can find that

$$|\sigma_1| < 1, |\sigma_2| < 1. \tag{15}$$

Based on the theories of dynamical system [6,7,11], (15) indicates that the dynamical system in (11) can converge to an equilibrium $\bar{\mathbf{Y}}$, i.e., $\bar{\mathbf{Y}}(t+1) = \bar{\mathbf{Y}}(t)$ holds. Thus, according to (11) and (12), we have

$$\bar{\mathbf{Y}} = \begin{bmatrix} \mathbf{0} \\ \boldsymbol{\rho} \end{bmatrix}. \tag{16}$$

which proves the convergence of $P$ and $V$. Then, according to the lines 5–10 in Algorithm 1, we can find that $X^{cur}$, $X^{glo}$, and $C$ can also converge when $P$ and $V$ reach equilibria. Thus, Theorem 2 holds.

## 5 Experimental Analysis

To demonstrate that MaCP is a feasible perspective on MaO, this section conducts a series of experiments to verify the efficiency and effectiveness of CluSO. First, we illustrate our proposed test suite DC-MaTS in detail, followed by a brief introduction to the baselines for comparison. After that, we present and analyze the corresponding experimental results. All experiments are conducted on a personal computer equipped with a 3.0 GHz Intel Core i7 processor.

### 5.1 Test Problems

Similar to many common MaTSs [9,14,22], our proposed DC-MaTS has the following standard form:

$$f_i(\boldsymbol{x}) = -g(x_i) + \frac{1}{d-1}\left(-g(x_i) + \sum_{a=1}^{d} g(x_a)\right), i = 1, 2, \ldots, m, \tag{17}$$

where $\boldsymbol{x} = [x_1, x_2, \ldots, x_d]^T$ denotes a solution with $0 \leq x_a \leq 1$ and $d \geq m$, and $g(x)$ is a monotonically increasing function with $x \in [0, 1]$, $g(0) = 0$, and $g(1) = 1$. Thus, we can find that $f_i(\boldsymbol{x}) \in [-1, 1]$ and the optimal solution to $f_i(\boldsymbol{x})$ is $\boldsymbol{x}^* = [x_1^*, x_2^*, \ldots, x_d^*]^T$ where $x_i^*$ equals 1 and all other elements equal 0. This property shows that the optimal solution of each objective is different to each other, which aligns with many real-world scenarios [12,17,24]. Then, we consider some common functions and give four concrete problems of DC-MaTS:

- $g(x) = x$ (DC-MaTS1).
- $g(x) = x^2$ (DC-MaTS2).
- $g(x) = \sqrt{x}$ (DC-MaTS3).
- $g(x) = \sin 0.5\pi x$ (DC-MaTS4).

In this paper, we set $d = m$ for simplicity and use these four types of problems with different settings of $m$ and $k$ (termed DC-MaTS($m$,$k$)) to evaluate CluSO.

### 5.2 Baselines

To the best of our knowledge, there has not been any algorithm other than CluSO that is specifically designed for solving MaCP. Thus, we compare CluSO to some commonly-used MaOEAs including MOEA/D [45], NSGA-III [13], and RVEA [8]. It should be noted that the purpose of the comparison is not to demonstrate the superiority of CluSO, which is perhaps predictable as the algorithm is specifically designed for the proposed MaCP, while the others are not. Instead, we would like to illustrate that MaCP is a feasible perspective on MaO, and the practically significant problems in NC-MaTS are challenging for existing MaOEAs. From this perspective, the comparison serves as a moderate spur to inspire more upcoming algorithms designed from the perspective of MaCP.

For a fair comparison, the code of CluSO is implemented by Python, and these three baselines are all implemented by the Pymoo package [2] in Python. With a trial-and-error process, the parameters with similar meanings in CluSO and baselines are uniformly set as follows: swarm (population) size $s = 600$ and iteration number $T = 100$. Besides, the learning parameters in CluSO are set as $\alpha = \beta = 1$, and other parameters in baselines are all set to default values.

A very important point is that all these baselines address MaOPs from the perspective of Pareto optimality. In all test environments, they return a large-size set (termed $U$, $|U| \gg k$) of non-dominated solutions rather than directly providing $k$ solutions like CluSO. For a fair comparison, the greedy algorithm, a widely-used solution selection method for decision-makers [5,19,40], is adopted to select the best $k$ solutions from $U$ as the final output of these baselines.

### 5.3 Experimental Results

We first evaluate the effectiveness ($\mathcal{G}(X)$) and efficiency (computational time) of CluSO on different problems of DC-MaTS. Similar to some related studies [29,30,32], we conduct 10 independent runs for each algorithm and report the corresponding results in Table 1 and Table 2, where the bold numbers indicate the best results. In all 16 instances of DC-MaTS, CluSO achieves the best $\mathcal{G}(X)$ value, and its computational time is only 9% $\sim$ 19% of the best baseline. The main reason is that all these baselines address these MaOPs from the perspective of Pareto optimality. When facing numerous objectives, they conduct many comparisons between solutions and generate a large number of non-dominated solutions, thus consuming lots of time and resulting in unremarkable $\mathcal{G}(X)$ values. By contrast, CluSO solves these MaOPs from the perspective of MaCP and avoids many unnecessary calculations, thus obtaining better $\mathcal{G}(X)$ values in an efficient manner.

However, the experimental results in Table 1 and Table 2 can also bring the following two questions:

- Do the objective values obtained by CluSO distribute evenly? In other words, is CluSO's optimal performance achieved by sacrificing certain objectives?
- For baselines, can the optimal solution set selected by the greedy algorithm represent their performance well?

**Table 1.** The means and standard deviations of $\mathcal{G}(X)$ and computational time (CT/s) obtained by CluSO and all baselines on DC-MaTS1 and DC-MaTS2.

| | MOEA/D | | NSGA-III | | RVEA | | CluSO | |
|---|---|---|---|---|---|---|---|---|
| | $\mathcal{G}(X)$ | CT | $\mathcal{G}(X)$ | CT | $\mathcal{G}(X)$ | CT | $\mathcal{G}(X)$ | CT |
| DC-MaTS1 (25,3) | –3.29 (±0.19) | 99.48 (±2.91) | –5.74 (±0.26) | 124.91 (±5.95) | –4.40 (±0.16) | 81.98 (±3.67) | **–8.26** (±0.50) | **7.60** (±0.13) |
| DC-MaTS1 (50,5) | –1.55 (±0.17) | 152.51 (±7.88) | –7.91 (±0.23) | 208.57 (±8.61) | –7.68 (±0.16) | 146.70 (±6.18) | **–16.06** (±0.89) | **13.75** (±0.13) |
| DC-MaTS1 (75,7) | –2.86 (±0.36) | 206.83 (±15.72) | –13.41 (±0.54) | 296.81 (±20.17) | –14.01 (±0.40) | 204.40 (±15.83) | **–23.31** (±0.85) | **19.63** (±0.08) |
| DC-MaTS1 (100,9) | –4.29 (±0.66) | 227.87 (±28.79) | –20.91 (±1.05) | 325.55 (±27.20) | –22.57 (±0.79) | 210.39 (±26.30) | **–30.29** (±1.25) | **25.55** (±0.15) |
| DC-MaTS2 (25,3) | –2.02 (±0.26) | 101.62 (±3.41) | –6.13 (±0.18) | 113.81 (±3.73) | –4.47 (±0.20) | 82.33 (±3.62) | **–8.64** (±0.49) | **8.51** (±0.14) |
| DC-MaTS2 (50,5) | –1.36 (±0.20) | 154.15 (±8.33) | –9.01 (±0.79) | 200.42 (±15.13) | –5.83 (±0.24) | 139.81 (±3.23) | **–16.34** (±0.59) | **15.79** (±0.26) |
| DC-MaTS2 (75,7) | –3.40 (±0.53) | 219.41 (±22.24) | –15.88 (±1.10) | 301.29 (±24.02) | –11.87 (±1.08) | 203.71 (±17.12) | **–24.04** (±1.18) | **23.04** (±0.41) |
| DC-MaTS2 (100,9) | –4.75 (±0.77) | 231.37 (±24.33) | –25.81 (±1.75) | 329.95 (±21.18) | –15.61 (±1.85) | 209.27 (±21.84) | **–31.80** (±1.19) | **30.14** (±0.70) |

**Table 2.** The means and standard deviations of $\mathcal{G}(X)$ and computational time (CT/s) obtained by CluSO and all baselines on DC-MaTS3 and DC-MaTS4.

| | MOEA/D | | NSGA-III | | RVEA | | CluSO | |
|---|---|---|---|---|---|---|---|---|
| | $\mathcal{G}(X)$ | CT | $\mathcal{G}(X)$ | CT | $\mathcal{G}(X)$ | CT | $\mathcal{G}(X)$ | CT |
| DC-MaTS3 (25,3) | –3.17 (±0.15) | 98.78 (±3.15) | –4.94 (±0.15) | 115.92 (±5.69) | –4.26 (±0.10) | 82.89 (±4.13) | **–6.98** (±0.32) | **8.50** (±0.24) |
| DC-MaTS3 (50,5) | –1.07 (±0.10) | 152.88 (±4.79) | –6.73 (±0.14) | 202.08 (±7.64) | –7.00 (±0.17) | 142.41 (±10.07) | **–13.20** (±0.40) | **15.38** (±0.10) |
| DC-MaTS3 (75,7) | –1.98 (±0.28) | 211.80 (±22.45) | –10.93 (±0.31) | 298.11 (±23.10) | –12.17 (±0.56) | 207.36 (±20.16) | **–18.59** (±0.81) | **21.52** (±0.11) |
| DC-MaTS3 (100,9) | –2.86 (±0.59) | 228.14 (±25.63) | –15.72 (±0.61) | 319.37 (±31.67) | –17.61 (±0.27) | 213.35 (±23.51) | **–23.94** (±1.05) | **29.62** (±0.54) |
| DC-MaTS4 (25,3) | –4.04 (±0.20) | 104.99 (±4.99) | –5.34 (±0.17) | 114.97 (±4.51) | –4.65 (±0.28) | 87.37 (±4.75) | **–9.29** (±0.66) | **11.45** (±0.18) |
| DC-MaTS4 (50,5) | –1.37 (±0.22) | 157.86 (±9.55) | –6.71 (±0.25) | 209.17 (±12.38) | –6.78 (±0.27) | 152.30 (±6.00) | **–16.77** (±0.97) | **21.68** (±0.17) |
| DC-MaTS4 (75,7) | –2.08 (±0.20) | 224.66 (±25.83) | –11.36 (±0.31) | 306.13 (±27.29) | –11.93 (±0.51) | 218.11 (±21.93) | **–24.82** (±0.96) | **32.77** (±0.33) |
| DC-MaTS4 (100,9) | –2.87 (±0.39) | 229.07 (±29.03) | –17.00 (±0.87) | 346.41 (±34.78) | –18.26 (±1.15) | 226.23 (±28.44) | **–31.56** (±1.52) | **43.33** (±0.37) |

To answer these two questions, we further present the objective values of complete non-dominated solution sets (obtained by all baselines) and $k$ solutions (generated by CluSO) on all objectives of different DC-MaTS1 problems, as shown in Fig. 2. It can be observed that the distribution of objective values obtained by CluSO is highly uniform. Moreover, when the total number of objectives is 25, 50, 75, and 100, CluSO achieves optimal results on 64% (16/25), 82% (41/50), 76% (57/75), and 84% (84/100) of the objectives, respectively, which shows the powerful capability of CluSO in optimizing many objectives and also demonstrates that MaCP is a feasible perspective on MaO.

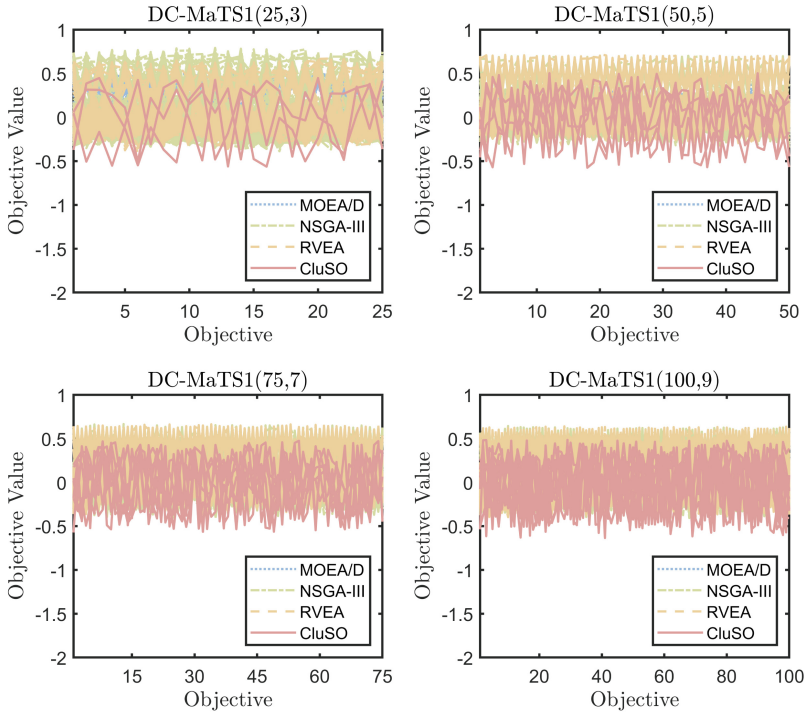



**Fig. 2.** The objective values of complete non-dominated solution sets (obtained by all baselines) and $k$ solutions (generated by CluSO) on all objectives of different DC-MaTS1 problems.

## 6 Conclusion

In this paper, we provide a new perspective on many-objective optimization by introducing a many-objective cover problem (MaCP), which aims to discover a few solutions to cover many objectives. We prove the NP-hard property of MaCP and develop a clustering-based swarm optimizer (CluSO) with a convergence guarantee to solve it. Then, we propose a decoupling many-objective test suite (DC-MaTS) with practical significance to evaluate CluSO. Extensive experimental results demonstrate both the efficiency and effectiveness of CluSO, which shows the potential to tackle problems with one hundred or even more objectives by optimizing MaCP. In the future, we plan to further analyze the nature of MaCP and design more strategies to enhance the performance of CluSO.

**Acknowledgments.** This work was supported by the Research Grants Council of the Hong Kong Special Administrative Region, China (CityU11215622), the Key Basic Research Foundation of Shenzhen, China (JCYJ20220818100005011), and the Natural Science Foundation of China (62276223).

**Disclosure of Interest.** The authors declare that they have no known competing interest that could appear to influence the work reported in this paper.

## References


1. Ahmed, M., Seraj, R., Islam, S.M.S.: The k-means algorithm: a comprehensive survey and performance evaluation. Electronics **9**(8), 1295 (2020)
2. Blank, J., Deb, K.: Pymoo: multi-objective optimization in python. IEEE Access **8**, 89497–89509 (2020)
3. Brockhoff, D., Wagner, T., Trautmann, H.: On the properties of the R2 indicator. In: Proceedings of the Annual Conference on Genetic and Evolutionary Computation, pp. 465–472 (2012)
4. Censor, Y.: Pareto optimality in multiobjective problems. Appl. Math. Optim. **4**(1), 41–59 (1977)
5. Chen, W., Ishibuchi, H., Shang, K.: Fast greedy subset selection from large candidate solution sets in evolutionary multiobjective optimization. IEEE Trans. Evol. Comput. **26**(4), 750–764 (2021)
6. Cheng, R., Jin, Y.: A competitive swarm optimizer for large scale optimization. IEEE Trans. Cybern. **45**(2), 191–204 (2014)
7. Cheng, R., Jin, Y.: A social learning particle swarm optimization algorithm for scalable optimization. Inf. Sci. **291**, 43–60 (2015)
8. Cheng, R., Jin, Y., Olhofer, M., Sendhoff, B.: A reference vector guided evolutionary algorithm for many-objective optimization. IEEE Trans. Evol. Comput. **20**(5), 773–791 (2016)
9. Cheng, R., Li, M., Tian, Y., Zhang, X., Yang, S., Jin, Y., Yao, X.: A benchmark test suite for evolutionary many-objective optimization. Complex Intell. Syst. **3**, 67–81 (2017)
10. Chugh, T., Jin, Y., Miettinen, K., Hakanen, J., Sindhya, K.: A surrogate-assisted reference vector guided evolutionary algorithm for computationally expensive many-objective optimization. IEEE Trans. Evol. Comput. **22**(1), 129–142 (2016)
11. Clerc, M., Kennedy, J.: The particle swarm - explosion, stability, and convergence in a multidimensional complex space. IEEE Trans. Evol. Comput. **6**(1), 58–73 (2002)
12. Cuate, O., Schütze, O.: Pareto explorer for solving real world applications. Res. Comput. Sci. **149**(3), 29–36 (2020)
13. Deb, K., Jain, H.: An evolutionary many-objective optimization algorithm using reference-point-based nondominated sorting approach, part I: Solving problems with box constraints. IEEE Trans. Evol. Comput. **18**(4), 577–601 (2014)
14. Deb, K., Thiele, L., Laumanns, M., Zitzler, E.: Scalable test problems for evolutionary multiobjective optimization. In: Evolutionary Multiobjective Optimization: Theoretical Advances and Applications, pp. 105–145. Springer, London (2005). https://doi.org/10.1007/1-84628-137-7_6
15. Dinur, I., Safra, S.: On the hardness of approximating minimum vertex cover. Annals of Mathematics, pp. 439–485 (2005)
16. Drineas, P., Frieze, A., Kannan, R., Vempala, S., Vinay, V.: Clustering large graphs via the singular value decomposition. Mach. Learn. **56**, 9–33 (2004)
17. Duro, J.A., Saxena, D.K.: Timing the decision support for real-world many-objective optimization problems. In: Proceedings of the Evolutionary Multi-Criterion Optimization, pp. 191–205 (2017)

18. Grandoni, F.: A note on the complexity of minimum dominating set. J. Discrete Algorithms **4**(2), 209–214 (2006)
19. Gu, Y.R., Bian, C., Li, M., Qian, C.: Subset selection for evolutionary multi-objective optimization. IEEE Trans. Evol. Comput. **28**(2), 403–417 (2023)
20. Guerreiro, A.P., Fonseca, C.M., Paquete, L.: The hypervolume indicator: computational problems and algorithms. ACM Comput. Surv. **54**(6), 1–42 (2021)
21. Hassin, R., Levin, A.: A better-than-greedy approximation algorithm for the minimum set cover problem. SIAM J. Comput. **35**(1), 189–200 (2005)
22. Huband, S., Hingston, P., Barone, L., While, L.: A review of multiobjective test problems and a scalable test problem toolkit. IEEE Trans. Evol. Comput. **10**(5), 477–506 (2006)
23. Ishibuchi, H., Imada, R., Setoguchi, Y., Nojima, Y.: Reference point specification in inverted generational distance for triangular linear Pareto front. IEEE Trans. Evol. Comput. **22**(6), 961–975 (2018)
24. Jaimes, A.L., Oyama, A., Fujii, K.: Space trajectory design: Analysis of a real-world many-objective optimization problem. In: Proceedings of the IEEE Congress on Evolutionary Computation, pp. 2809–2816 (2013)
25. Kennedy, J., Eberhart, R.: Particle swarm optimization. In: Proceedings of International Conference on Neural Networks, vol. 4, pp. 1942–1948 (1995)
26. Lambrinidis, G., Tsantili-Kakoulidou, A.: Multi-objective optimization methods in novel drug design. Expert Opin. Drug Discov. **16**(6), 647–658 (2021)
27. Laumanns, M., Thiele, L., Deb, K., Zitzler, E.: Combining convergence and diversity in evolutionary multiobjective optimization. Evol. Comput. **10**(3), 263–282 (2002)
28. Li, B., Li, J., Tang, K., Yao, X.: Many-objective evolutionary algorithms: a survey. ACM Comput. Surv. **48**(1), 1–35 (2015)
29. Li, K., Wang, H., Wang, W., Wang, F., Cui, Z.: Improving artificial bee colony algorithm using modified nearest neighbor sequence. J. King Saud Univ. Comput. Inf. Sci. **34**(10), 8807–8824 (2022)
30. Li, K., Xu, M., Zeng, T., Ye, T., Zhang, L., Wang, W., Wang, H.: A new artificial bee colony algorithm based on modified search strategy. Int. J. Comput. Sci. Math. **15**(4), 387–395 (2022)
31. Li, Y., Fan, J., Wang, Y., Tan, K.L.: Influence maximization on social graphs: a survey. IEEE Trans. Knowl. Data Eng. **30**(10), 1852–1872 (2018)
32. Liu, Y., Liu, J., Teng, X.: Single-particle optimization for network embedding preserving both local and global information. Swarm Evol. Comput. **71**, 101069 (2022)
33. Liu, Y., Liu, J., Wu, K.: Cost-effective competition on social networks: a multi-objective optimization perspective. Inf. Sci. **620**, 31–46 (2023)
34. Ntakolia, C., Iakovidis, D.K.: A swarm intelligence graph-based pathfinding algorithm (SIGPA) for multi-objective route planning. Comput. Oper. Res. **133**, 105358 (2021)
35. Ostachowicz, W., Soman, R., Malinowski, P.: Optimization of sensor placement for structural health monitoring: a review. Struct. Health Monit. **18**(3), 963–988 (2019)
36. Owen, S.H., Daskin, M.S.: Strategic facility location: a review. Eur. J. Oper. Res. **111**(3), 423–447 (1998)
37. Pan, L., He, C., Tian, Y., Wang, H., Zhang, X., Jin, Y.: A classification-based surrogate-assisted evolutionary algorithm for expensive many-objective optimization. IEEE Trans. Evol. Comput. **23**(1), 74–88 (2018)
38. Paschos, V.T.: A survey of approximately optimal solutions to some covering and packing problems. ACM Comput. Surv. **29**(2), 171–209 (1997)

39. Shi, L., Cai, X.: An exact fast algorithm for minimum hitting set. In: Proceedings of the International Joint Conference on Computational Science and Optimization, vol. 1, pp. 64–67 (2010)
40. Singh, H.K., Bhattacharjee, K.S., Ray, T.: Distance-based subset selection for benchmarking in evolutionary multi/many-objective optimization. IEEE Trans. Evol. Comput. **23**(5), 904–912 (2018)
41. Sviridenko, M., Ward, J.: Large neighborhood local search for the maximum set packing problem. In: Proceedings of the International Colloquium on Automata, Languages, and Programming, pp. 792–803 (2013)
42. Wu, Q., Hao, J.K.: A review on algorithms for maximum clique problems. Eur. J. Oper. Res. **242**(3), 693–709 (2015)
43. Yang, S., Li, M., Liu, X., Zheng, J.: A grid-based evolutionary algorithm for many-objective optimization. IEEE Trans. Evol. Comput. **17**(5), 721–736 (2013)
44. Yuan, Y., Xu, H., Wang, B., Yao, X.: A new dominance relation-based evolutionary algorithm for many-objective optimization. IEEE Trans. Evol. Comput. **20**(1), 16–37 (2016)
45. Zhang, Q., Li, H.: MOEA/D: a multiobjective evolutionary algorithm based on decomposition. IEEE Trans. Evol. Comput. **11**(6), 712–731 (2007)
46. Zhang, Q., Zhou, A., Jin, Y.: RM-MEDA: a regularity model-based multiobjective estimation of distribution algorithm. IEEE Trans. Evol. Comput. **12**(1), 41–63 (2008)
47. Zhang, T., Wang, H., Yuan, B., Jin, Y., Yao, X.: Surrogate-assisted evolutionary Q-learning for black-box dynamic time-linkage optimization problems. IEEE Trans. Evol. Comput. **27**(5), 1162–1176 (2022)
48. Zhou, A., Qu, B.Y., Li, H., Zhao, S.Z., Suganthan, P.N., Zhang, Q.: Multiobjective evolutionary algorithms: a survey of the state of the art. Swarm Evol. Comput. **1**(1), 32–49 (2011)

# Solution-Based Knowledge Discovery for Multi-objective Optimization

Clément Legrand[1(✉)], Diego Cattaruzza[2], Laetitia Jourdan[1], and Marie-Eléonore Kessaci[1]

[1] Univ. Lille, CNRS, Centrale Lille, UMR 9189 CRIStAL, 59000 Lille, France
{clement.legrand4.etu,laetitia.jourdan,
marie-eleonore.kessaci}@univ-lille.fr
[2] Univ. Lille, CNRS, Inria, Centrale Lille, UMR 9189 CRIStAL, 59000 Lille, France
diego.cattaruzza@centralelille.fr

**Abstract.** In the combinatorial optimization field, Knowledge Discovery (KD) mechanisms (e.g., data mining, neural networks) have received increasing interest over the years. KD mechanisms are based upon two main procedures, being the extraction of knowledge from solutions, and the injection of such knowledge into solutions. However, in a multi-objective (MO) context, the simultaneous optimization of many conflicting objectives can lead to the learning of contradictory knowledge. We propose to develop a Solution-based KD (SKD) mechanism suited to MO optimization. It is integrated within two existing metaheuristics: the Iterated MO Local Search (IMOLS) and the MO Evolutionary Algorithm based on Decomposition (MOEA/D). As a case study, we consider a bi-objective Vehicle Routing Problem with Time Windows (bVRPTW), to define accordingly the problem-dependent knowledge of the SKD mechanism. Our experiments show that using the KD mechanism we propose increases the performance of both IMOLS and MOEA/D algorithms.

**Keywords:** Knowledge Discovery · Multi-objective Optimization · Combinatorial Optimization · Routing Problems

## 1 Introduction

Efficient exploration of the search space is a key element of solving discrete optimization problems. Indeed, the search space is a set of regions containing solutions of different quality, from which it may be more or less difficult to escape. In this paper, we assume that solutions in the same region share common characteristics and, by wisely combining them, it is possible to reach more interesting regions with better-performing solutions. This assumption was verified on Solomon's benchmark, where 40% (resp. 25%) of arcs are shared between close (high-quality) solutions in instances with tight (resp. wide) time windows. In addition, this assumption is used by PILS [1], in which the structure of local optima guides the exploration towards regions that are difficult to reach by simple local search (LS) algorithms.


M. Affenzeller et al. (Eds.): PPSN 2024, LNCS 15151, pp. 83–99, 2024.
https://doi.org/10.1007/978-3-031-70085-9_6

Knowledge Discovery (KD) processes have already received various interests, in single-objective [1,11,23] and in multi-objective (MO) [18,24,32] optimization. In particular, the concept of *innovization* introduced by Deb et al. [10], focuses on the dependency between the decision variables of a solution to help an optimization algorithm to reach specific parts of the objective space. We investigate a different approach by using the representation of the solution (here, as a permutation) instead of directly using the decision variables. Our approach finds echoes in Estimation of Distribution Algorithms [22], and more recently in linkage learning for permutation problems [13], although our approach exploits only frequent common structures found instead of using bayesian networks to learn more precise dependencies between variables.

In this article, we further develop the notion of *Solution-based Knowledge Discovery* (SKD) metaheuristics, by extending the construction of knowledge groups developed in [18]. This work leads to a new model coherent with MO optimization algorithms. We instantiate the model with the Iterated MOLS (IMOLS) [5] and the MO Evolutionary Algorithm based on Decomposition (MOEA/D) [34].

Since the extraction and injection procedures are themselves dependent on the problem studied, we decided to base our study on a bi-objective Vehicle Routing Problem with Time Windows (bVRPTW) already presented in [19]. In this problem, we minimize the total traveling cost and waiting time of drivers, which are conflicting objectives [7]. Indeed, when a driver arrives too early a waiting time is incurred, increasing the duration of the route for the driver. Considering real-life situations (e.g. food delivery, medical transportation), this additional time may incur satisfaction issues. Moreover, in the classical version of the VRPTW, the first objective to optimize is the number of vehicles, which is a discrete objective function, and then the total traveled distance is minimized as a second objective. However, the use of two continuous objectives (the total cost and the total waiting time) together allows the generation of fronts that contain, in general, many more non-dominated solutions, and it is better suited to a MO context, especially when knowledge is extracted from solutions.

The article is structured as follows: Section 2 presents MO optimization concepts and the IMOLS and MOEA/D metaheuristics. Our contribution, the SKD metaheuristic, is presented in Sect. 3. The model is integrated into IMOLS and MOEA/D in Sect. 4. Section 5 presents the problem and defines the knowledge to extract and inject in this context. In Sect. 6 our experimental protocol is presented and the results obtained are discussed. We conclude in Sect. 7.

## 2 Context

### 2.1 Multi-objective Optimization

A *Multi-objective Combinatorial Optimization Problem* (MoCOP) is commonly formalized as follows [8]:

$$(MoCOP) = \begin{cases} Optimize\ F(x) = (f_1(x), f_2(x), \ldots, f_n(x)) \\ s.t.\ x \in \mathcal{D}, \end{cases} \quad (1)$$

where $n \geq 2$ objective functions $f_i$ have to be optimized, $x$ is a vector of decision variables, and $\mathcal{D}$ is the set of solutions. The *objective space* is the image of $F$.

We say that a solution $x$ dominates a solution $y$, noted $x \prec y$ in a minimization context, if and only if for all $i \in \{1 \dots n\}$, $f_i(x) \leq f_i(y)$ and there exists $j \in \{1 \dots n\}$ such that $f_j(x) < f_j(y)$. The dominance relation induces a partial order in the solution space. Indeed, there exist pairs of solutions that cannot be compared to each other. Such solutions are said to be *non-dominant*.

A *Pareto front* is defined as a set of non-dominated solutions. A feasible solution $x^* \in \mathcal{D}$ is called *Pareto optimal* if and only if there is no solution $x \in \mathcal{D}$ such that $x \prec x^*$. We solve a MoCOP by finding all the Pareto optimal solutions, which form together the *Pareto optimal set*. The image of the Pareto optimal set by the objective function $F$ provides the *true Pareto front*.

To compare Pareto fronts, and thus the algorithms providing them, many indicators have been developed [27]. In this paper, we consider the unary hypervolume (uHV) [35] metric. It is defined relatively to a reference point $Z_{ref}$, generally $(1.001, \dots, 1.001)$, and requires that the objectives of the solutions are normalized between 0 and 1. This indicator is to be maximized and allows a good evaluation of the front's accuracy, diversity, and cardinality. Geometrically, the uHV represents the volume of the objective space (bounded by $Z_{ref}$) covered by the members of a non-dominated set of solutions.

Many metaheuristics based on LS techniques, called MOLS [5], or using evolutionary algorithms, like Non-Dominated Sorting Genetic Algorithm (NSGA-II) [9], and MOEA/D [34], have been designed to solve MO problems. The following sections focus on iterated MOLS (Sect. 2.2) and MOEA/D (Sect. 2.3).

### 2.2 Iterated MOLS

A MOLS is an algorithm that iteratively explores solutions selected from a *current* population, by using LS procedures, accepts *candidates* during the search, and then updates an *external archive* of non-dominated solutions. We refer to the survey of Blot et al. [5], for a comprehensive overview of all possible mechanisms related to the conception of MOLS. A large part of MOLS algorithms is Pareto-based, meaning that they rely on the dominance criterion to accept neighbors during the search, contrarily to aggregation-based ones, which aggregate the different objectives to turn the problem into single-objective optimization. Among the Pareto-based algorithms, we find the Dominance-based MOLS (DMLS) algorithms [20] and the Pareto LS (PLS) [25].

In addition, it is possible to consider iterated MOLS (IMOLS), which mimic iterated LS, by using a *perturbation* procedure as a restart when a particular condition is reached (e.g., convergence of the MOLS).

### 2.3 MOEA/D

MOEA/D [34], is a genetic algorithm widely studied in the literature [33], approximating the Pareto front by decomposing the MO problem into several scalar objective subproblems. There exist many ways to generate $M$ scalar

problems, but in every case, it requires a set of weight vectors $w^1, \ldots, w^M$. A weight vector $w = (w_1, \ldots, w_n)$ is such that, $\forall i \in \{1, \ldots, n\}$ $w_i \geq 0$ and $\sum_{i=1}^{n} w_i = 1$, where $n$ is the number of objectives considered. During the execution of MOEA/D, a population of solutions is maintained, where the $i$-th solution of the population is the best solution found for the $i$-th subproblem. Usually, a random permutation of the subproblems is defined in the beginning so that subproblems are always solved in the same order. Subproblems are iteratively solved, by applying a genetic step composed of crossover and mutation operators. When the subproblem $i$ is optimized, two solutions from the population are selected for the crossover step. To perform that selection, two neighbor subproblems of subproblem $i$ (included) are chosen, knowing that the neighborhood of a subproblem contains the $m$ subproblems associated with the closest (for the Euclidean distance) weight vectors to weight vector $w^i$. The mutation is commonly replaced by a LS [6,15,17], to intensify the search in the regions identified with the crossover. If the final solution obtained is better than the initial solution considered for the subproblem, then it is replaced. The final solution is also tentatively added to an *external archive* storing the best non-dominated solutions found during the search and returned once the termination criterion of the algorithm is reached.

### 2.4 Unified View of IMOLS and MOEA/D

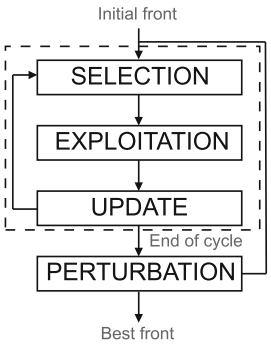



**Fig. 1.** The proposed unified view for IMOLS and MOEA/D metaheuristics.

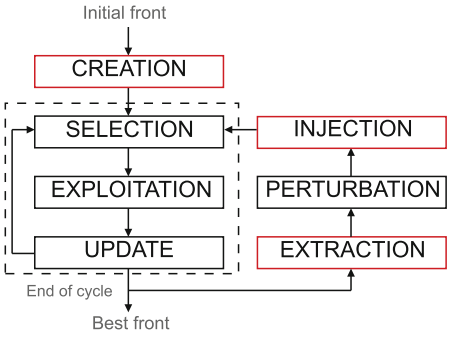



**Fig. 2.** The unified view integrating the three steps of the SKD.

This section shows the structural similarities between IMOLS and MOEA/D through a unified view. Our motivation behind this unification is to show how our knowledge discovery mechanism (SKD), presented in Sect. 3, can be integrated into algorithms sharing the same structural properties.

The IMOLS and MOEA/D frameworks can be abstracted with the following four main steps: `Selection`, `Exploitation`, `Update`, and `Perturbation`. The Fig. 1 shows how these steps interact together. The `Exploitation` step is used

for intensification while the `Perturbation` one for diversification. A *cycle* is defined as a succession of a fixed number of iterations of the three first steps (i.e., `Selection`, `Exploitation`, and `Update`). When a cycle ends, a `Perturbation` occurs, if a specific criterion is met, to update the current population before the next `Selection`, and so forth, until a termination criterion is reached (generally based on time or number of iterations). An external archive is maintained to track the best non-dominated solutions found and is finally returned. The steps are discussed below with details about their instantiation in IMOLS and MOEA/D.

The `Selection` step chooses one or several solutions to explore in the current population, initialized with the initial front provided. This choice can be done randomly, or following a criterion to focus on a specific region of the objective space. In IMOLS, the selection is directly performed from the current population. In MOEA/D, each subproblem is sequentially selected, and consequently, the associated solution is explored.

`Exploitation` is the intensification step of the algorithm where the search focuses on specific regions of the search space. During this step, the neighborhood of the selected solutions is exploited, until a criterion is reached, to generate new (better) candidate solutions. In IMOLS, the exploitation consists of accepting either non-dominated or dominating neighbors of the selected solutions, considering a reference set. Consequently, many iterations are needed to reach out to a Pareto local optima. In MOEA/D the exploitation consists of applying a single-objective LS [14], for the selected subproblem, until a local optimum is reached.

When new solutions are found after the exploitation, the `Update` step tentatively integrates them into the external archive and the current population. While the external archive generally relies on bounded mechanisms, it is possible to adopt different strategies to update the current population (e.g., replacement of the solution explored, keeping non-dominated solutions in priority).

In neighborhood-based algorithms and evolutionary ones, it is necessary to perturb solutions to escape regions with local optima. The `Perturbation` generates new solutions to be explored by applying random moves, destroy and repair mechanisms, or genetic operators. It acts like a diversification step where new regions of the search space can be identified and then explored. After the perturbation, the solutions are used to create a new current population, and a new cycle is started. In IMOLS, the perturbation relies on LS mechanisms. In MOEA/D, it corresponds to a crossover.

In the next section, we present the SKD mechanism. Its integration in IMOLS and MOEA/D is presented in Sect. 4.

## 3 Solution-Based Knowledge Discovery

### 3.1 Global Overview and Main Issues

In [18], the concept of *knowledge groups* is introduced. The idea is to divide the objective space into regions each representing a *knowledge group*. A knowledge group gathers structural elements of the solutions of the same region. If the

approach was promising, many obstacles remain and have to be tackled to ensure a model, that can be easily integrated into various MO algorithms. The first issue concerns the creation of knowledge groups. The creation proposed in [18] was dependent on the aggregations used in MOEA/D, which highly restricted its range of applications. Our proposition, detailed in Sect. 3.2 overcomes this limit, allowing many more integration possibilities. Then, the interaction between the extraction (resp. injection) procedure and the groups newly created, is presented in Sect. 3.3 (resp. Sect. 3.4). Although the interactions remain similar to those described in [18], we present them in a more flexible way to allow a better integration in metaheuristics.

### 3.2 Creation of Knowledge Groups

The problem is associating each knowledge group with a region of the objective space. We consider that each group is related to a representative, inducing the region of the group. In the following, we consider, for simplicity purposes, a bi-objective space. We propose two strategies to create the $k_G$ representatives of the groups. The first one, represented in Fig. 3, with $k_G = 5$ representatives named $g^i$, generates $k_G$ uniformly spread weight vectors. Then, to evaluate the proximity of a solution to a group we aggregate the objectives of the solution by using the weight vector associated with the group. This strategy is a simple variant of [18] allowing to use it in other algorithms than MOEA/D. The second strategy, represented in Fig. 4, links the extreme points of the current front with a straight line. Then, $k_G$ points (including the extreme points) are regularly created on the line. Each created point corresponds to a representative of a group. The proximity of a solution to a group is then evaluated by the Euclidean distance between the objective vector of the solution and the representative. With this second strategy, it is possible (and recommended) to dynamically update the representatives of each group, before the extraction, if the extreme points vary. In both figures, each point of the Pareto front is linked to its closest representative, which leads to different distributions for each construction.

### 3.3 Extraction and Knowledge Groups

The extraction procedure is presented in Algorithm 1. It is possible to deactivate the extraction until a certain execution time is reached, to balance low-quality and high-quality solutions. In the following, we activate the procedure with no delay, at the beginning of the execution, since an initial front is provided.

For the extraction procedure, a *learning set* $L$ of solutions generated during the execution of the algorithm is provided. However, MO algorithms explore plenty of solutions during their execution (e.g., MOLS), and learning from all of them would scramble the knowledge added to the groups. Consequently, a subset of $L$ that contains only the solutions that undergo the extraction procedure is considered. Here, we suggest to keep only the non-dominated solutions of $L$. In particular, it allows the learning to focus on the most interesting solutions. Please

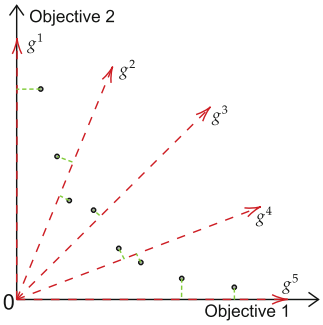



**Fig. 3.** Creation of groups based on weight vectors (denoted WG).

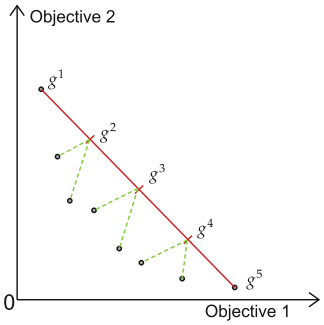



**Fig. 4.** Creation of groups based on extrema points (denoted EG).

note that other possibilities may be taken into account as a random sample or a mix of dominated and non-dominated solutions.

Once $L$ is filtered, knowledge is extracted from each remaining solution $x$. It is then added to the $d_e$ closest groups (function `SelectGroups`, l.4 of Algorithm 1) of $x$, following the evaluation of the proximity between a solution and a group provided in Sect. 3.2. The parameter $d_e$ allows the control of the diversification of the mechanism: smaller values correspond with fewer groups being updated. Then, the elements of knowledge are added to the corresponding groups, and a score (e.g., the frequency of appearance) reflecting the relevance of each element is updated. However, we choose not to allow the same solution to contribute more than once to a group, to avoid the bias induced by local optima. The set $L$ is emptied after updating the groups.

The construction of $L$ and the function `Filter` used in Algorithm 1 are presented in Sect. 4 since they are algorithm-dependent. The functions `Extract` (l.3) and `Update` (l.5), being problem-dependent, are presented in Sect. 5.2.

**Algorithm 1:** Extraction procedure.

**Input:** $\mathcal{A}$ the current archive, $\mathcal{G}$ the knowledge groups, $L$ the learning set, and $d_e$ the number of groups to update.
**Output:** The updated knowledge groups.

1 $\mathcal{S} \leftarrow$ `Filter`$(L)$
2 **for** $x \in \mathcal{S}$ **do**
3 $\quad \mathcal{K} \leftarrow$ `Extract`$(x)$
4 $\quad G = \{G_1, \ldots, G_{d_e}\} \leftarrow$ `SelectGroups`$(\mathcal{G}, d_e, x)$
5 $\quad$ `Update`$(G, \mathcal{K})$
6 $L \leftarrow \emptyset$
7 **return** $\mathcal{G}$

### 3.4 Injection and Knowledge Groups

At that point, all the solutions from the current population undergo the injection procedure presented in Algorithm 2.

First, the knowledge to inject has to be selected. Likewise for the extraction (see the `SelectGroups` function), a subset of $d_i$ groups containing the closest groups to $x$ is created. Again, this parameter controls the diversification of the mechanism. The function `SelectOne` applied to the $d_i$ candidate groups selects the final group, that will produce the knowledge to inject in the solution $x$. It can be done at random, or following a specific criterion if a group is preferred. Then, some knowledge is selected from the resulting group with the `SelectKnowledge` function. In particular, the selection of the knowledge should use the scores of the elements in the group. In that case, it is possible to select the elements with the highest score or by means of a roulette wheel mechanism. Each element $k$ of knowledge is tentatively injected into a solution $x'$ (initially $x$) using the function `Inject`. All solutions accepted (e.g. those non-dominating $x'$) during the injection of $k$ are added to a set $S'$. The next solution $x'$ to undergo the injection can be replaced by taking one of the solutions of $S'$ (function `SelectNextSolution`). For that choice, it is possible to select a solution at random, with a dominance criterion, or with an aggregation when it is defined. Finally, after the injection of all the elements of knowledge, all the accepted solutions are returned.

`SelectOne` (l.2), `SelectKnowledge` (l.3), and `SelectNextSolution` (l.7) used in Algorithm 2 are defined in Sect. 4 since they are algorithm-dependent. The problem-dependent function `Inject` (l.6) is defined in Sect. 5.2.

**Algorithm 2:** Injection procedure.

**Input:** $\mathcal{G}$ the knowledge groups, $x$ the current solution, and $d_i$ the number of candidate groups.
**Output:** Accepted solutions.

```
1  G = {G_1, ..., G_{d_i}} ← SelectGroups(G, d_i, x)
2  G' ← SelectOne(G)
3  K ← SelectKnowledge(G')
4  S ← ∅
5  x' ← x
6  for k ∈ K do
7  |   S' ← Inject(k, x')
8  |   x' ← SelectNextSolution(S', x')
9  |   S ← S ∪ S'
10 return S
```

### 3.5 Integration of SKD Into the Unified View

The Solution-based Knowledge Discovery (SKD) uses knowledge groups and the procedures of extraction and injection suited to MO algorithms. The Unified

View presented in Sect. 2.4 contains successive steps of intensification and diversification. The intensification is usually the core of the MO algorithms where identified regions of the search space are deeply explored using an underlying local knowledge given by the neighborhood. In this section, we integrate the SKD into MO algorithms using our unified view (see Fig. 1). We aim to improve the diversification phase, by exploring larger regions of the search space with the knowledge stored in the groups.

At the beginning of the execution, given the initial front provided, the knowledge groups are created following one strategy presented in Sect. 3.2.

Applying the extraction procedure at every iteration would result in a lot of noise for the knowledge groups. In particular, waiting a few iterations allows the learning set to contain more interesting solutions. Hence, the `Extraction` step should be applied only after the end of a cycle, on a subset of explored solutions.

Any solution can undergo the injection but, like the `Extraction`, applying it to all the explored solutions would waste computational resources. Thus, we consider that the injection should be applied only after the end of a cycle and more precisely after the `Perturbation` if it occurred or after the `Extraction` otherwise. After the injection, a new cycle (i.e., an intensification step) is started by updating the archive and the current population. These remarks lead to the conception of the model presented in Fig. 2.

## 4 SKD for IMOLS and MOEA/D

### 4.1 SKD for IMOLS

We follow the DMLS model originally introduced by Liefooghe et al. [20]. The problem's representation, the solution evaluation, and the neighborhood structure are defined in Sect. 5.1 with the problem. The algorithm starts from an initial front given by the user, integrated into a bounded archive, $\mathcal{A}$, of size $U_a$, representing the current population. The archive is bounded by using the crowding distance [9]. Then, $U_c$ randomly selected solutions from the archive (among the not entirely explored ones) form the set to explore. The DMLS algorithm iteratively explores the selected solutions. During the LS, the neighborhood of a solution $x$ is explored until a non-dominated solution, considering all solutions of $\mathcal{A}$, is found [4]. If no solution is found, $x$ is tagged as explored and is no longer selected during the current cycle, moreover tagged solutions cannot be selected during the LS. If any, the accepted solution is tentatively added to $\mathcal{A}$.

In the iterated variant, we manage a second (unbounded) archive, $\mathcal{A}^*$, containing the best non-dominated solutions found during the execution. After $l_c$ iterations (denoting the length of a cycle), the uHV of $\mathcal{A}$ is evaluated, and the solutions of $\mathcal{A}$ are integrated into $\mathcal{A}^*$. Before starting a new cycle, if all solutions of $\mathcal{A}$ are tagged as explored or the uHV has not been increased by at least $e_{uHV}$ after two consecutive cycles, a perturbation step occurs. During this step, all tags are removed from solutions, all elements from $\mathcal{A}$ are tagged as explored and a new archive $\mathcal{A}$ is created by perturbing solutions from $\mathcal{A}^*$. To perturb a solution $x$, we apply three moves of the LS, with the following acceptance criterion: a

solution $y$ is accepted when $\forall i \in \{1, \ldots, n\}, f_i(y) \leq (1+\epsilon_p) \cdot f_i(x)$, with $\epsilon_p \in \mathbb{R}^+$, allowing a slight relaxation of the objectives of $x$ to test the dominance relation. This version of IMOLS is called $\text{R}_{\text{IMOLS}}$.

Following the steps presented in Sect. 3.5, the extraction and injection procedures are added to $\text{R}_{\text{IMOLS}}$. The variant using the weights (resp. the extrema) to create the groups is called $\text{WG}_{\text{IMOLS}}$ (resp. $\text{EG}_{\text{IMOLS}}$). Concerning the extraction procedure, we have to define how the learning set is managed and how its elements are filtered. Every solution tentatively added to $\mathcal{A}$ after the exploration step should be added to the learning set, since it may produce interesting knowledge to exploit. We only keep non-dominated solutions to filter the solutions of the learning set. Concerning the injection procedure, it is sequentially applied to all the solutions from $\mathcal{A}$. The `SelectOne`, `SelectKnowledge`, and `SelectNextSolution` functions from Algorithm 2 are defined hereafter. The `SelectOne` function chooses the group that gives the knowledge to inject. Here, we choose the group randomly. For the `SelectKnowledge` function, we rely on the scores of the elements learned. We consider $N_i$ elements, randomly selected among the $N_f$ elements with the highest scores, as it was done in [1]. More details are given in Sect. 5.2 in the context of the problem. Finally, for the `SelectNextSolution` function, the initial solution is returned ($x$ in Algorithm 2). Indeed, since we work with a MOLS algorithm, we prefer staying locally around the solution by attempting to inject knowledge into it rather than trying to highly optimize the solution. Finding a better solution is interesting, but could dominate a large part of the archive, resulting in a loss of diversity.

### 4.2 SKD for MOEA/D

Now we provide an instantiation of MOEA/D, called $\text{R}_{\text{MOEAD}}$, following the framework described in Sect. 2.3. We consider scalar problems obtained with a weighted sum of the objectives. Contrary to Tchebycheff decomposition, it does not require a reference point. Given a weight vector $w$, the fitness of a solution is defined as the following quantity: $g(x|w) = \sum_{i=1}^{n} w_i \cdot f_i(x)$. However, all the solutions of the true Pareto front can not be obtained with such aggregations. In the following, we generate $M$ weight vectors uniformly distributed, assuming that is enough to obtain diverse subproblems. A Partially Mapped Crossover (PMX) [16] is applied with probability $p_{pmx}$. Among the two generated solutions, only one is randomly chosen to keep the population's size constant. When the crossover is not applied, the solution associated with the $i$-th subproblem is kept. The mutation is a LS detailed in Sect. 5.1, and applied with probability $p_{ls}$.

Following the steps presented in Sect. 3.5, the extraction and injection procedures are added to $\text{R}_{\text{MOEAD}}$. The variant using the weights (resp. the extrema) to create the groups is called $\text{WG}_{\text{MOEAD}}$ (resp. $\text{EG}_{\text{MOEAD}}$). For the extraction procedure, we keep the idea exposed in Sect. 4.1. Each solution tentatively added to the external archive (`Update` step of Fig. 1), is added to the learning set. Then, the knowledge is extracted from non-dominated solutions of the learning set. The injection procedure is applied to all the solutions of

the current population (i.e., the best solution of each subproblem). The functions `SelectOne` and `SelectKnowledge` are the same as presented in Sect. 4.1, but `SelectNextSolution` differs. The next solution is the best (considering the aggregation of the associated subproblem) accepted during the injection.

## 5 Case Study: Bi-objective VRPTW

### 5.1 Presentation

See [19] for a detailed formalization of the bi-objective VRPTW (bVRPTW) considered. The bVRPTW calls for the determination of routes such that the traveling cost (i.e. the sum of the Euclidean distance between consecutive customers) and the total waiting time (i.e. the sum of the waiting times induced by an early arrival to deliver a customer) are simultaneously minimized. Moreover, each solution of the bVRPTW needs to satisfy the following constraints: each route starts and ends at a specific location (called depot), each customer is visited by exactly one route, the sum of the demands of the customers in any route does not exceed the capacity of the vehicles, and time windows are respected (late arrivals are not allowed).

A solution to the problem is encoded as a customer permutation and evaluated with the split algorithm provided by [26], providing a feasible solution. For this study, we consider the operators `Relocate`, `Swap`, and `2-opt`$^*$. These simple operators are largely used in LS algorithms for routing problems [28] since they can produce a large neighborhood, and allow an easy incremental evaluation. The `Relocate` operator moves one customer from its current position to another location. The `Swap` operator exchanges in the solution the position of two customers. The `2-opt`$^*$ operator generalizes the 2-opt, by involving different routes. In $\mathrm{R}_{\mathrm{MOEAD}}$ a Randomized Variable Neighborhood Descent is applied for exploitation [19,30], where the order of the operators is kept during descent (until a local optimum is reached) but shuffled each time the LS is applied. In $\mathrm{R}_{\mathrm{IMOLS}}$, the order of the operators is randomized too, but the search stops at the first accepting neighbor. Only feasible solutions are considered.

### 5.2 Knowledge Related to a Solution

In this section, the remaining `Extract` and `Update` (resp. `SelectKnowledge` and `Inject`) functions from the Algorithm 1 (resp. Algorithm 2), are defined to suit the bVRPTW context. These functions are inspired by the Pattern Injection LS (PILS) method [1]. It is an optimization method relying on frequent patterns from high-quality solutions to explore vast neighborhoods. PILS has already been integrated into the Hybrid Genetic Search [31] and the Guided LS [2] to solve the Capacitated Vehicle Routing Problem (CVRP).

In routing problems, patterns are defined as sequences of consecutive customers on a route without the depot. Those with a size between 2 and $s_p$ are extracted from generated solutions by `Extract`. In particular, a route

$r = (0, v_1, \ldots, v_{|r|}, 0)$, contains $max(|r| - k + 1, 0)$ patterns of size $k$. Once the patterns are extracted, `Update` adds them to corresponding groups. If the pattern already belongs to the group, its frequency is incremented. Otherwise, it is added with a frequency of one. Different groups may have different frequencies for the same pattern. A pattern becomes *frequent* when its frequency exceeds a threshold $l_f$.

For the injection, `SelectKnowledge` randomly selects a pattern size among $\{2, \ldots, s_p\}$ to not bias the selection towards smaller, more numerous, patterns. Then, $N_i$ patterns are randomly chosen among the $N_f$ most frequent patterns of the corresponding size (without repetition). Only patterns tagged *frequent* can be selected. Given a pattern and a solution $x$, the `Inject` function creates a solution from $x$ containing the pattern provided, as explained in [1] (except that reversed fragments are discarded due to time windows). First, arcs connecting the pattern are removed, thus creating partial routes, which are reconnected (with an exhaustive search) to form feasible solutions. Several solutions may be accepted during the reconstruction step. In IMOLS, all non-dominated solutions are accepted, while solutions with better fitness are accepted in MOEA/D.

## 6 Experimental Study

### 6.1 Choice of Parameters Value

The tuning of the parameters for the $\text{R}_{\text{MOEAD}}$ variant comes from previous tuning with irace [21] and we refer to [19] for a detailed analysis of the parameters. $M = 40$ subproblems are created, with $m = 10$ neighbors. At most 2 neighbors may have their solution replaced during the update step. The crossover is applied with probability $p_{pmx} = 1.00$, and the LS with probability $p_{ls} = 0.10$.

For $\text{R}_{\text{IMOLS}}$, the parameters are chosen to be fair with $\text{R}_{\text{MOEAD}}$. The archive limit is set to $U_a = 40$. Each iteration, $U_c = 1$ solution is explored. The perturbation occurs when the uHV does not increase by at least $e_{uHV} = 10^{-2}$, and during the perturbation, $\epsilon_p = 0.02$. A cycle performs $l_c = 100$ iterations.

The parameters value of $\text{EG}_{\text{MOEAD}}$ and $\text{WG}_{\text{MOEAD}}$ (resp. $\text{EG}_{\text{IMOLS}}$ and $\text{WG}_{\text{IMOLS}}$) are similar and their values follow the recommendation made in [19]. $p_{pmx}$ is set to 0.50. There are $k_G = 20$ knowledge groups. The maximum size $s_p$ of extracted patterns is set to 8 (resp. 5) for instances of class 2 (resp. class 1) since large (resp. short) routes are designed. The knowledge is added to $d_e = 1$ group, and the knowledge to inject is provided by at most $d_i = 1$ group. $N_i = 100$ patterns of the same size are tentatively injected into each solution. They are selected among the $N_f = 250$ most frequent patterns of the corresponding size in the group. The threshold frequency for patterns is set to $l_f = 2$.

### 6.2 Experimental Protocol

The experiments are run on two computers "Intel(R) Xeon(R) CPU E5-2687W v4 @ 3.00 GHz", with 24 cores each. Our framework is implemented in the jMetalPy framework [3]. The source code and our results are available on a Git[1].

[1] https://gitlab.univ-lille.fr/clement.legrand4.etu/skd_integration.

The Solomon [29] and the Gehring and Homberger [12] benchmarks are commonly used to evaluate the performance of MO algorithms. Solomon's benchmark contains instances with up to 100 customers. Customers can be randomly located (R), clustered (C), or mixed (RC). Each category is divided into two classes. Instances of class 1 have tighter time windows than instances of class 2, which are less constrained. Gehring and Homberger's benchmark uses a similar instance generation but considers a larger number of customers.

To fairly compare the algorithms, they are all initialized with the same fronts. Hence, we generate 30 initial fronts (available on Git) for each instance. In IMOLS, the initial front is directly used as the initial population, however, in MOEA/D, each subproblem is initialized with the best solution of the front.

The six algorithms are then executed over 30 seeds on each instance. The termination criterion for each run is set to 10 (resp. 20) minutes for instances of size 100 (resp. 200). The average uHV obtained over the 30 runs is compared with Pairwise Wilcoxon tests with Bonferroni correction.

### 6.3 Results

**Table 1.** Average uHV ($\times 10^3$) of the algorithms on the different categories of instances. $R_{MOEAD}$ and $R_{IMOLS}$ are the reference algorithms. $EG_{MOEAD}$, $WG_{MOEAD}$, $EG_{IMOLS}$, and $WG_{IMOLS}$ are the learning variants. Gray cells are statistically better comparing all algorithms, i.e., the six rows. Bold values represent the best-performing algorithms when MOEA/D (resp. IMOLS) variants are compared together (i.e., three rows each).

| Size | 100 | | | | | | 200 | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Category | C | | R | | RC | | C | | R | | RC | |
| Class | C1 | C2 | R1 | R2 | RC1 | RC2 | C1 | C2 | R1 | R2 | RC1 | RC2 |
| $R_{MOEAD}$ | 833 | 888 | 805 | 773 | 776 | 792 | 703 | 613 | 755 | 668 | 733 | 702 |
| $WG_{MOEAD}$ | **904** | **912** | **834** | **795** | **784** | **808** | **793** | **788** | **800** | **741** | **806** | **792** |
| $EG_{MOEAD}$ | 856 | 902 | 806 | 778 | 762 | 792 | 744 | 740 | 784 | 723 | 774 | 765 |
| $R_{IMOLS}$ | 923 | 966 | 850 | 761 | **837** | 766 | 822 | 746 | 754 | 654 | 758 | 619 |
| $WG_{IMOLS}$ | **970** | **987** | **886** | **814** | **844** | **823** | **885** | **826** | 811 | **761** | **854** | **830** |
| $EG_{IMOLS}$ | 958 | **986** | **885** | 807 | **844** | 814 | **875** | **835** | **814** | 751 | 842 | 814 |

Table 1 summarises the results obtained. Detailed results per instance are available on the Git provided. First, $R_{IMOLS}$ returns better results than $R_{MOEAD}$ except on instances R2 and RC2 of size 100, and RC2 of size 200. Indeed, instances of category 2 are less constrained, leading to a bigger exploration space. In that case, it seems preferable to use MOEA/D rather than IMOLS to intensify the search. However, this consideration does not apply to C2 instances, probably due to the presence of clusters, leading to more local optima.

We can see that using SKD (no matter the strategy used to create the groups) positively impacts $R_{IMOLS}$ in all instances. The same conclusion holds for $R_{MOEAD}$ except on RC1 instances of size 100, with EG groups. Moreover, using SKD is even more beneficial in instances of bigger sizes.

In MOEA/D, using the strategy with the weight vectors to create the groups is statistically better than using the other one. Probably because the algorithm itself uses weight vectors to decompose the search space. Concerning the IMOLS algorithm, both strategies are often equivalent, but using the weight vectors leads to slightly better results. Thus, this strategy should be preferred in general. Additionally, using SKD allows the creation of more diversified Pareto fronts for MOEA/D and IMOLS (see Fig. 5 for comparison).

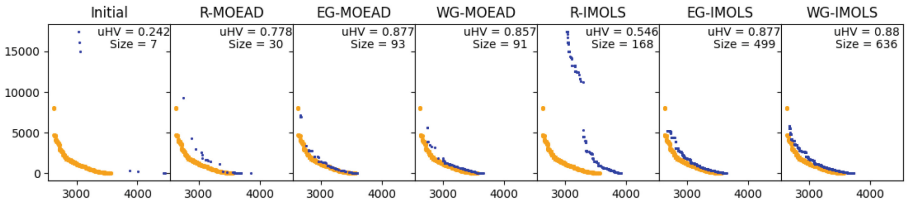



**Fig. 5.** Results of the execution on instance RC2_2_6 (run 6), from the Gehring and Homberger set. The associated hypervolume and size of the final fronts (blue dots) are shown, as well as the reference front (orange dots). (Color figure online)

## 7 Conclusion

In this paper, we proposed to extend the mechanism of [18] to develop a solution-based KD mechanism, called SKD, which extracts knowledge from solutions and injects knowledge to explore new regions of the solution space. The mechanism is mainly based on the creation of knowledge groups, dividing the objective space. Here, two creation strategies for the groups are developed and compared. Any MO algorithm that can be an instantiation of the unified algorithm presented in Fig. 1, can be extended by integrating SKD as shown in Fig. 2. Then, we integrated SKD into two MO algorithms (IMOLS and MOEA/D) to solve a bi-objective routing problem, and we defined accordingly the algorithm-dependent components and the problem-dependent knowledge. Experiments were performed over instances with different characteristics of size and structure. In most cases, using SKD increases the performance of the original algorithm, showing an interest in our developed mechanism. Moreover, creating the groups with the weight vector strategy seems more profitable.

Future works should investigate the impact of the initialization on SKD. More precisely, the time to start the learning may impact the resolution and further analysis may be beneficial. Finally, it could be interesting to investigate the possibility of transferring the knowledge learned from one instance to another

without starting from scratch again. This may be done by detecting similarities in the instances.

**Disclosure of Interests.** The authors have no competing interests to declare that are relevant to the content of this article.

## References


1. Arnold, F., Santana, Í., Sörensen, K., Vidal, T.: PILS: exploring high-order neighborhoods by pattern mining and injection. Pattern Recogn. (2021)
2. Arnold, F., Sörensen, K.: Knowledge-guided local search for the vehicle routing problem. Comput. Oper. Res. **105**, 32–46 (2019)
3. Benitez-Hidalgo, A., Nebro, A.J., Garcia-Nieto, J., Oregi, I., Del Ser, J.: jMetalPy: a Python framework for multi-objective optimization with metaheuristics. Swarm Evol. Comput. **51**, 100598 (2019)
4. Blot, A., Jourdan, L., Kessaci, M.É.: Automatic design of multi-objective local search algorithms: case study on a bi-objective permutation flowshop scheduling problem. In: Proceedings of the Genetic and Evolutionary Computation Conference, pp. 227–234 (2017)
5. Blot, A., Marmion, M., Jourdan, L.: Survey and unification of local search techniques in metaheuristics for multi-objective combinatorial optimisation. J. Heuristics **24**(6), 853–877 (2018). https://doi.org/10.1007/s10732-018-9381-1
6. Bossek, J., Grimme, C., Meisel, S., Rudolph, G., Trautmann, H.: Local search effects in bi-objective orienteering. In: Proceedings of the Genetic and Evolutionary Computation Conference, pp. 585–592 (2018)
7. Castro-Gutierrez, J., Landa-Silva, D., Pérez, J.M.: Nature of real-world multi-objective vehicle routing with evolutionary algorithms. In: 2011 IEEE International Conference on Systems, Man, and Cybernetics, pp. 257–264. IEEE (2011)
8. Coello, C.A.C., Dhaenens, C., Jourdan, L.: Multi-objective combinatorial optimization: problematic and context. In: Coello Coello, C.A., Dhaenens, C., Jourdan, L. (eds.) Advances in Multi-Objective Nature Inspired Computing, vol. 272, pp. 1–21. Springer, Heidelberg (2010). https://doi.org/10.1007/978-3-642-11218-8_1
9. Deb, K., Pratap, A., Agarwal, S., Meyarivan, T.: A fast and elitist multiobjective genetic algorithm: NSGA-II. IEEE Trans. Evol. Comput. **6**(2), 182–197 (2002)
10. Deb, K., Srinivasan, A.: Innovization: innovating design principles through optimization. In: Proceedings of the 8th Annual Conference on Genetic and Evolutionary Computation, pp. 1629–1636 (2006)
11. Fitzpatrick, J., Ajwani, D., Carroll, P.: Learning to prune electric vehicle routing problems. In: Sellmann, M., Tierney, K. (eds.) LION 2023. LNCS, vol. 14286, pp. 378–392. Springer, Cham (2023). https://doi.org/10.1007/978-3-031-44505-7_26
12. Gehring, H., Homberger, J.: A parallel hybrid evolutionary metaheuristic for the vehicle routing problem with time windows. In: Proceedings of EUROGEN99, vol. 2, pp. 57–64. Springer, Berlin (1999)
13. Guijt, A., Luong, N.H., Bosman, P.A., de Weerdt, M.: On the impact of linkage learning, gene-pool optimal mixing, and non-redundant encoding on permutation optimization. Swarm Evol. Comput. **70**, 101044 (2022)
14. Hoos, H.H., Stützle, T.: Stochastic Local Search: Foundations and Applications. Elsevier (2004)

15. Knowles, J.D.: Local-search and hybrid evolutionary algorithms for Pareto optimization. Ph.D. thesis, University of Reading, Reading (2002)
16. Kora, P., Yadlapalli, P.: Crossover operators in genetic algorithms: a review. Int. J. Comput. Appl. **162**(10) (2017)
17. Land, M.W.S.: Evolutionary algorithms with local search for combinatorial optimization. University of California, San Diego (1998)
18. Legrand, C., Cattaruzza, D., Jourdan, L., Kessaci, M.E.: Improving MOEA/D with knowledge discovery. Application to a bi-objective routing problem. In: Emmerich, M., et al. (eds.) EMO 2023. LNCS, vol. 13970, pp. 462–475. Springer, Cham (2023). https://doi.org/10.1007/978-3-031-27250-9_33
19. Legrand, C., Cattaruzza, D., Jourdan, L., Kessaci, M.E.: Improving neighborhood exploration into MOEA/D framework to solve a bi-objective routing problem. Int. Trans. Oper. Res. (2023)
20. Liefooghe, A., Humeau, J., Mesmoudi, S., Jourdan, L., Talbi, E.G.: On dominance-based multiobjective local search: design, implementation and experimental analysis on scheduling and traveling salesman problems. J. Heuristics **18**, 317–352 (2012)
21. López-Ibáñez, M., Dubois-Lacoste, J., Cáceres, L.P., Birattari, M., Stützle, T.: The irace package: iterated racing for automatic algorithm configuration. Oper. Res. Perspect. **3**, 43–58 (2016)
22. Lozano, J.A.: Towards a New Evolutionary Computation: Advances on Estimation of Distribution Algorithms, vol. 192. Springer, Heidelberg (2006). https://doi.org/10.1007/3-540-32494-1
23. Lucas, F., Billot, R., Sevaux, M., Sörensen, K.: Reducing space search in combinatorial optimization using machine learning tools. In: Kotsireas, I.S., Pardalos, P.M. (eds.) LION 2020. LNCS, vol. 12096, pp. 143–150. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-53552-0_15
24. Moradi, B.: The new optimization algorithm for the vehicle routing problem with time windows using multi-objective discrete learnable evolution model. Soft. Comput. **24**(9), 6741–6769 (2020)
25. Paquete, L., Chiarandini, M., Stützle, T.: Pareto local optimum sets in the biobjective traveling salesman problem: an experimental study. In: Gandibleux, X., Sevaux, M., Sörensen, K., T'kindt, V. (eds.) Metaheuristics for Multiobjective Optimisation. LNEMS, vol. 535, pp. 177–199. Springer, Heidelberg (2004). https://doi.org/10.1007/978-3-642-17144-4_7
26. Prins, C.: A simple and effective evolutionary algorithm for the vehicle routing problem. Comput. Oper. Res. **31**(12), 1985–2002 (2004)
27. Riquelme, N., Von Lücken, C., Baran, B.: Performance metrics in multi-objective optimization. In: 2015 Latin American Computing Conference (CLEI), pp. 1–11. IEEE (2015)
28. Schneider, M., Schwahn, F., Vigo, D.: Designing granular solution methods for routing problems with time windows. Eur. J. Oper. Res. **263**(2), 493–509 (2017)
29. Solomon, M.M.: Algorithms for the vehicle routing and scheduling problems with time window constraints. Oper. Res. **35**(2), 254–265 (1987)
30. Subramanian, A., Uchoa, E., Ochi, L.S.: A hybrid algorithm for a class of vehicle routing problems. Comput. Oper. Res. **40**(10), 2519–2531 (2013)
31. Vidal, T., Crainic, T.G., Gendreau, M., Prins, C.: A unified solution framework for multi-attribute vehicle routing problems. Eur. J. Oper. Res. (2014)
32. Wattanapornprom, W., Olanviwitchai, P., Chutima, P., Chongstitvatana, P.: Multi-objective combinatorial optimisation with coincidence algorithm. In: 2009 IEEE Congress on Evolutionary Computation, pp. 1675–1682. IEEE (2009)

33. Xu, Q., Xu, Z., Ma, T.: A survey of multiobjective evolutionary algorithms based on decomposition: variants, challenges and future directions. IEEE Access **8**, 41588–41614 (2020)
34. Zhang, Q., Li, H.: MOEA/D: a multiobjective evolutionary algorithm based on decomposition. IEEE Trans. Evol. Comput. **11**(6), 712–731 (2007)
35. Zitzler, E., Thiele, L., Laumanns, M., Fonseca, C.M., Da Fonseca, V.G.: Performance assessment of multiobjective optimizers: an analysis and review. IEEE Trans. Evol. Comput. **7**(2), 117–132 (2003)

# Innovization for Route Planning Applied to an Uber Movement Speeds Dataset for Berlin

Eva Röper[1(✉)], Jens Weise[2], Christoph Steup[1], and Sanaz Mostaghim[1,3]

[1] Faculty of Computer Science, Otto-von-Guericke-University Magdeburg, Magdeburg, Germany
{eva.roeper,christoph.steup,sanaz.mostaghim}@ovgu.de
[2] Agentur für Innovation in der Cybersicherheit GmbH, Halle, Germany
weise@cyberagentur.de
[3] Fraunhofer Institute for Transportation and Infrastructure Systems IVI, Dresden, Germany

**Abstract.** Multi-objective route planning is a prominent but computationally expensive optimisation problem of everyday life. Reusing knowledge from similar route planning problems could enhance the performance and the sustainability of routing algorithms. The goal of this paper is to adapt the concept of innovization to route planning and in this way extract knowledge from Pareto-optimal solutions. As part of the adaptation, we design a multi-objective evolutionary algorithm for routing and introduce a novel local search for routing problems called Perimeter Mutation Local Search. We evaluate our proposed approach on multi-objective time-dependent routing problems to see what knowledge can be gained and whether this knowledge can improve a multi-objective evolutionary algorithm. Our results show that we can extract knowledge using the introduced innovization for route planning. This knowledge is used to improve a multiobjective evolutionary algorithm by reducing computational effort. With only about 40 % of previously necessary function evaluations, we manage to produce similar optimisation results. This is particularly beneficial for mobile applications with limited available computational resources.

**Keywords:** Innovization · Time-Dependent Route Planning · Multi-objective Evolutionary Algorithms

## 1 Introduction

Route planning is an important optimisation problem of everyday life, affecting many people and various industries. In the past, front-seat passengers were often the ones reading maps and providing direction. Nowadays, this task is mostly performed by routing algorithms of Intelligent Transport Systems (ITSs). Nonetheless, optimising multiple objectives for routes remains a challenging and


M. Affenzeller et al. (Eds.): PPSN 2024, LNCS 15151, pp. 100–116, 2024.
https://doi.org/10.1007/978-3-031-70085-9_7

computationally expensive problem. This especially affects mobile applications like ITSs where few computational resources are available. Ideally, we could reuse knowledge from similar routing problems to enhance the performance of routing algorithms. The innovization methodology [11] gives instructions for extracting knowledge from Pareto-optimal solutions of optimisation problems. The identified, reusable principles could save computation time of algorithms which would make the use of multi-objective algorithms for routing more viable in everyday life. This could result in a quality of life improvement for drivers through additional decision-making support and personalisation options for routes. In addition, reusing knowledge to accelerate routing algorithms increases the sustainability of algorithms and decreases energy consumption.

However, innovization cannot simply be applied to routing problems because such problems has complex decision variables and constraints, and often non-differentiable objective functions. To the best of our knowledge, there are no previous works attempting to use innovization on route planning problems. Therefore, the main goal of this paper and its primary contribution is the development of an innovization for route planning, an adaption of the original innovization. In addition, we assess what knowledge can be gained from a real-world problem using our methodology and whether the gained knowledge can improve an multi-objective evolutionary algorithm (MOEA). Improvement is measured as an increase in efficiency or in normalised hypervolumes of final Pareto fronts, or as a reduction of the total number of function evaluations required. For the innovization for route planning, we construct a MOEA for routing and propose a novel local search method for routing problems called Perimeter Mutation Local Search (PMM-LS). The experiments are executed on multi-objective time-dependent routing problems which optimise travel time, travel time variability and ease of driving. Travel times are time-dependent, and are based on the Uber Movement Speeds dataset [30] for Berlin in January 2020. The proposed method and the evaluation are based on the work in [27].

The rest of this paper is structured as follows. We start by relating this paper to other work in Sect. 2. In Sect. 3, we present our innovization for route planning methodology including our novel local search PMM-LS. The evaluation of the knowledge extraction and algorithm improvement using our proposed approach follows in Sect. 4. Finally, Sect. 5 concludes this paper and presents ideas for future work.

## 2 Related Work

The work most similar to our route planning approach is that of Kanoh and Hara [21]. With time-dependent travel time, route length and ease of driving, they define similar objective functions. Unlike us, they use cellular automata [20] to calculate traffic prediction before the optimisation. Moreover, we utilise Non-dominated Sorting Genetic Algorithm II (NSGA-II) while they propose a combination of the Dijkstra algorithm [12] and a genetic algorithm with different operators than ours. Liu et al. [23] also use NSGA-II with Node Based Crossover (NBX)

but other steps of their evolutionary algorithm (EA) differ from this paper. Furthermore, they have similar objective functions which are total vehicular emission cost, time-dependent travel time, number of turns, and route length. Although the authors utilise real-world data too, they only test their method on a single path planning problem in a much smaller road network than studied in this paper.

To the best of our knowledge, there is no work using innovization for problems in the area of route planning. Coello Coello et al. [6] applied innovization to a related problem, a bi-objective travelling salesman problem. They discovered that some edges are rarely part of Pareto-optimal solutions. As an innovized principle, authors suggest excluding these infrequent edges during optimisation to lower computational effort. In route planning, the value of this type of knowledge is limited since it is not often reusable in complex road networks. There, routes for various route planning problems often use different streets, even when planning in the same city.

## 3 Innovization for Route Planning

Deb and Srinivasan proposed a 6-step innovization methodology [11] which extracts knowledge from Pareto-optimal solutions to potentially accelerate the solving process of similar optimisation problems. The Pareto-optimal solutions are obtained with a multi-objective optimisation method and verified for a high confidence in the results of the innovization. For verification, the authors use Benson's method [2] to determine possible, dominating solutions for some clustered solutions, compute extreme solutions and apply the normal constraint method (NCM) [24]. Finally, similarities, properties of variables or their relation ensuring the Pareto-optimality of solutions are identified as resuable, innovised principles. Benson's method and NCM define auxiliary problems which can also be defined for route planning problems. However, route planning problems usually have disconnected feasible regions due to complex decision variables and constraints such as road networks, as well as often non-differentiable objective functions. Therefore, methods such as the simplex algorithm or gradient-based approaches are not applicable to the auxiliary problems for route planning. The only viable alternative would be to use a metaheuristic, but then guarantees from the verification methods would no longer hold. To the best of our knowledge, there are no other verification methods applicable to routing problems. For this reason, we propose the adapted version of the original innovization method shown in Fig. 1.

In the first part, a Pareto front of a route planning problem is computed using our MOEA explained in Subsect. 3.1. As proposed for the original innovization, the MOEA is reused for the single-objective evolutionary algorithms (SOEAs) by defining only one objective at a time. Since finding extreme points for routing problems is hard due to their deceptiveness, it is possible that the SOEAs return previously undiscovered extreme points instead of verifying existing ones. Trivially, we only keep the non-dominated set of the Pareto front along with any newly found extreme points. Our novel local search PMM-LS, specialised for

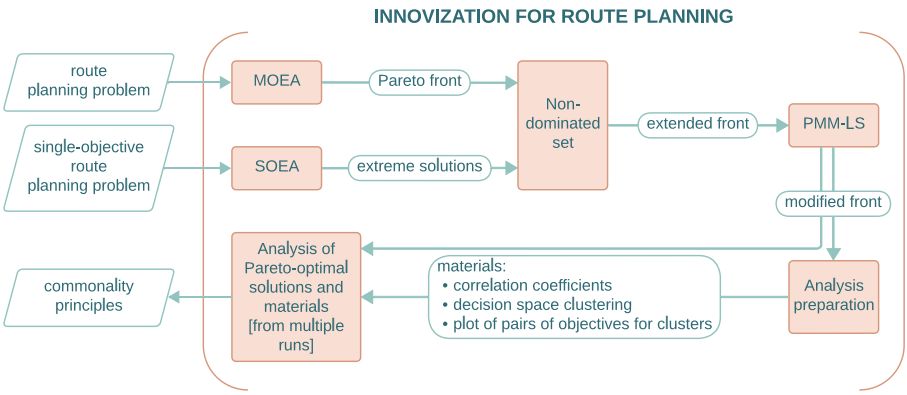



**Fig. 1.** Process of innovization for route planning

route planning applications, is executed on the extended front instead of Benson's method or NCM. For PMM-LS see Subsect. 3.2. Inserting new extreme points from single-objective runs into the front before PMM-LS improves the results of the local search because, due to the deceptiveness of routing problems, new solutions can help reveal entire new parts of the Pareto front.

Based on the obtained, modified front, we perform an analysis preparation step. First, we compute the correlation coefficients between different route characteristics and our objectives. As route characteristics, we choose the number of traffic signals along a route and the percentages of street types. For street types, we distinguish between motorways, main roads, residential streets and other or unclassified streets. We then compute a decision space clustering before plotting pairs of objectives for the clusters. Furthermore, we recommend visualising clusters of routes on a map for the analysis. For the decision space clustering, we apply Ordering Points To Identify the Clustering Structure (OPTICS) [1] with automatic cluster extraction. OPTICS can handle arbitrarily shaped clusters with different densities and noise. Another advantage of this method is that only the $MinPts$ parameter for identifying core points has to be set. We employ the heuristic $MinPts$ = number of objectives + 2 by Ester et al. [14]. As the distance function in decision spaces, we utilise the discrete Fréchet distance [13] with Euclidean distance between node coordinates. The other parameter $\epsilon$ for the neighbourhood size can simply be fixed to infinity. However, lower $\epsilon$ values can limit the runtime of the algorithm. In contrast to the original innovization, we perform the clustering after inserting new extreme points and conducting a local search because PMM-LS often returns dominating solutions which could change the clustering structure. Additionally, we do not cluster in the objective space but in the decision space. Similarities, that can be found for objective space clusters, are often too problem-specific due to the deceptiveness of route planning and the complexity of decision variables.

In the final step, we analyse Pareto-optimal solutions and materials from the analysis preparation step for any commonality principles. For route planning

problems, analysing fronts separately may result in the extraction of knowledge, such as shared edges, which is too problem-specific. Therefore, we propose combining the analysis of modified fronts from multiple runs, for example with different start and end points or with different departure times.

### 3.1 Multi-Objective Evolutionary Algorithm

As in the original innovization, we use NSGA-II [10] as our MOEA. Adapted steps are explained in the following paragraphs. To encode our route planning problems, we want to employ a natural encoding which is suitable for real-world data. Since we represent our road network as a graph, chromosomes are a sequence of node indices from a given start node to a destination node. This way, similar chromosomes have similar fitness values. Feasible individuals must have an edge between each pair of consecutive nodes. Otherwise, we give a penalty of $100,000$ to each objective per missing edge. As paths may have different lengths in reality, our encoding has variable length. To avoid unnecessary function evaluations, population sizes are chosen dynamically for every route planning problem based on its difficulty. We estimate the difficulty by the length of the shortest path $l_{sp}$ between start and end point. Therefore, we compute the population size as $k_N \cdot l_{sp}$ rounded to the nearest multiple of ten, where $k_N$ acts as a scaling coefficient. Concerning our stopping criterion, we follow the recommendation from the original innovization to let the MOEA run for a large number of generations for a high confidence in the innovization results. Since this is problem-specific, we terminate if improvements in the objective space are less than $\Delta_f$ for $\Delta_g$ consecutive generations. This ensures that the algorithm is stopped after diversification and converging of the algorithm are finished [4]. Stopping criteria are evaluated over multiple generations to make the termination more robust. As a fall-back, we stop the algorithm after $g_{max}$ generations.

**The Creation of the Initial Population** is done using guided random walks since standard random walks take too long in complex road networks. Each walk starts at the given start point $n_O$ and ends when the predefined destination $n_D$ is reached. Looking from any current node, we remove visited neighbours from the selection pool unless no other option remains. For simplicity, self-loops are disallowed. We do not disallow revisiting nodes altogether since random walks could otherwise get stuck if all neighbours were already visited. We move in direction of the node that is closest to the direction of our destination with a probability of $p_D$. The other $1 - p_D$ chance is evenly distributed among the rest of the neighbours. This guidance is a trade-off between the diversity of initial populations and runtime. However, the direction towards the goal can sometimes be misleading and the random walk gets stuck in a circle. If the current node has been visited at least ten times and all neighbours have already been visited, we choose a successor uniformly at random from all neighbours to get out of any loops but continue the random walks afterwards for more diverse individuals.

**The Node Based Crossover (NBX)**, to the best of our knowledge, was first described by Munetomo et al. [25] and has only later been termed NBX [5]. When

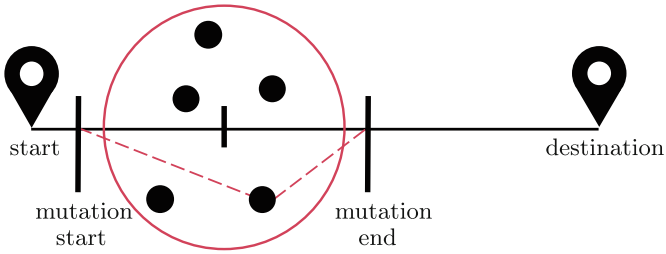



**Fig. 2.** Visualisation of the Perimeter Mutation Operator

two paths share common nodes other than the start and end point, they can be intuitively recombined with a one-point crossover at one of the common nodes. If two parents share no common points, a one-point crossover at a randomly placed cut is performed. The random placement excludes cuts after start and before end points. The resulting individuals are repaired with the shortest path between the nodes before and after the cut.

**The Perimeter Mutation Operator (PMM)** is largely inspired by the Perimeter Mutation Operator by Weise [32]. As visualised in Fig. 2, the idea is to replace a random part of the path $x$ by an alternative route. First, we choose random mutation start and end indices within $ws_{min}$ % to $ws_{max}$ % of the chromosome length $|x|$ apart. This encourages that the mutation is only a small change, but that an alternative route can be found. Since this does not always work, we recommend setting the mutation probability $p_m$ higher than the common 1/number of decision variables [8] or $1/|x|$ in our case. Secondly, we determine the middle node between mutation start and end. We find all nodes within a circle around this middle point with a radius of $m_r$ times the distance between mutation start and end node. Distances are calculated using the great-circle distance. We randomly select one of the nodes which is not already on the path. Finally, we replace the path between mutation start and end node with the shortest path between both nodes via the chosen alternative node.

### 3.2 Perimeter Mutation Local Search

As a local search method, we propose a novel approach called Perimeter Mutation Local Search (PMM-LS) based on our version of PMM. It is more systematic than the small mutations of few individuals during the run of an EA. The goal of PMM-LS is finding solutions dominating individuals in the extended front $P_{res}$ from the second step of our innovization for route planning. The pseudocode of PMM-LS is detailed in Algorithm 1. For every individual in $P_{res}$, alternative routes are created using PMM. The mutation is applied for every possible mutation window of $ws_{max}$ % of the chromosome length. Instead of selecting only one alternative node as for the MOEA, all possible alternative routes are generated. After each set of mutations, the sliding window is moved by step size $\Delta_w$.

After all alternative routes for all individuals have been computed, only the set of non-dominated individuals of $P_{res}$ and alternative routes is kept. These steps are repeated for any newly found individuals until no more new individuals were discovered. Finally, the method returns a locally improved Pareto front.

**Algorithm 1.** PMM-LS($P_{res}$)

$P = P_{res}$; $X_{new} = P_{res}$
**while** $X_{new} \neq \emptyset$ **do**
  $P_{old} = P$
  **for** $x \in X_{new}$ **do**
    $w_{start} = 0$; $w_{end} = w_{start} + \text{ROUND}(|x| \cdot ws_{max})$
    **while** $w_{end} \leq |x|$ **do**
      $centre = \text{MIDDLE\_POINT}(x[w_{start}], x[w_{end}])$
      $radius = \{v \in V \mid \text{DIST}(v, centre) \leq m_r \cdot \text{DIST}(x[w_{start}], x[w_{end}])\}$
      **for** $node_{alt} \in radius$ **do**
        $P = P \cup \text{REPLACE}(x, w_{start}, w_{end}, node_{alt})$
      $w_{start} = w_{start} + \Delta_w$; $w_{end} = w_{end} + \Delta_w$
  $P = \text{NON-DOMINATED\_SET}(P)$
  $X_{new} = P \backslash P_{old}$
**return** $P$

## 4 Analysis

For our experiments, we implement the methodology from Sect. 3 in Python using the pymoo framework [3] for our MOEA. As recommended for the original innovization, our parameter setting in Table 1 encourages a high confidence in the innovization results rather than fast convergence. The three most influential parameters at the top of the table are tuned. Due to time restrictions, the other parameters are set to values which we observed to work well during implementation and preliminary experiments. Results are to be interpreted in the context of the chosen parameter setting. Subsection 4.1 introduces the route planning problems we ran our experiments on. It also covers how time-dependent speeds were calculated based on the Uber Movement Speeds dataset for Berlin in January 2020 [30]. Subsection 4.2 thereafter covers the knowledge extraction from our innovization experiments. Innovized principles from this subsection are used for the experiments for the final Subsect. 4.3 where we examine whether an improvement of our MOEA is achieved. The code, the complete specifications for our experiments and some tabular results are available online[1].

### 4.1 Multi-Objective Time-Dependent Route Planning Problems

Our experiments are run on multi-objective time-dependent route planning problems which we identify by a 6-tuple $\left((f_t, f_v, f_s), G, n_O, n_D, w_0, t_0\right)$. As defined by Eq. 1, the 6-tuples represent specific problems with objective functions $(f_t, f_v, f_s)$, road network $G$, start point $n_O$, destination $n_D$, departure

[1] https://doi.org/10.5281/zenodo.10979116.

**Table 1.** Parameter setting for experiments with our innovization variant. Some values are changed as part of the innovized principles

| parameter | | value |
|---|---|---|
| scaling coefficient for population size | $k_N$ | 4.0 |
| probability of choosing neighbour in direction of destination $n_D$ | $p_D$ | 0.9 |
| mutation probability | $p_m$ | 0.25 |
| difference in indicators required for termination | $\Delta_f$ | 0.0025 |
| number of generations considered for robust termination | $\Delta_g$ | 25 |
| maximum number of generations | $g_{max}$ | 200 |
| minimum percentage of chromosome length for PMM window size | $ws_{min}$ | 10 |
| maximum percentage of chromosome length for PMM window size | $ws_{max}$ | 20 |
| radius coefficient for PMM | $m_r$ | 0.8 |
| step size for PMM-LS sliding window | $\Delta_w$ | 1 |

day $w_0$ and departure time $t_0$. The goal is to find vehicular paths which connect the given $n_O$ and $n_D$ in $G$ and which minimise the three conflicting objective functions explained below. For time-dependent problems, some of the information changes with time, but planning is generally done offline. In our case, the travel times of edges depend on the time and whether it is the weekend ($w = 1$) or not ($w = 0$). The road network is defined as a property graph $G = (V, E)$ which is directed and allows for attributing properties to nodes $V$ and edges $E$ [26]. Nodes are assigned indices as identifiers (IDs). Each edge $(u, v)$ has travel time attributes $\{tt_{uv}^{wh} \,|\, \forall w \in \{0,1\}\ \forall h \in [0,23], tt_{uv}^{wh} \in \mathbb{R}_{\geq 0}\}$. They are based on the length of edges and hourly average speed attributes $s_{uv}^{wh}$ for during the week and on weekends. Additionally, edges are assigned a number of traffic signals $sgn_{uv}$ encountered when traversing an edge in direction from $u$ to $v$.

$$
\begin{aligned}
\min \ \Big( f_t(x, w_0, t_0) \quad &= \sum_{i=0}^{|x|-1} tt_{x[i]x[i+1]}^{w_i t_i} + sgn_{x[i]x[i+1]} \cdot 20, \\
f_v(x, w_0, t_0) \quad &= \sqrt{\frac{1}{13} \sum_{i=-6}^{6} \max\big(0, f_t(x, w_0, t_0 + i \cdot 15) - \overline{tt}\big)^2}, \\
f_s(x) \quad &= \sum_{i=0}^{|x|-2} 180^\circ - \angle\big(x[i], x[i+1], x[i+2]\big) \Big) \\
\text{s.t.} \quad x \in \big\{ (n_0, \ldots, n_l) \ \big| \ & l \in \mathbb{N}_+; n_0, \ldots, n_l \in V; n_0 = n_O; n_l = n_D; \\
& \forall i \in \{0, \ldots, l-1\} \ \ (n_i, n_{i+1}) \in E \big\} \\
w_0 \in \{0, 1\} \\
t_0 \in \{hh : mm : ss \,|\, & hh \in [0,23]; mm, ss \in [0,59]\}
\end{aligned}
\tag{1}
$$

**The Travel Time Objective $f_t$**, unlike the shortest path, depends on multiple factors such as speed limits, varying congestion or traffic signals [22]. We compute the travel time of a route $x$ in seconds using the formula in Eq. 1. $|x|$ denotes the length of route $x$. The variables $w_i$ and $t_i$ calculate the current time and whether it is the weekend when reaching node $i$. Like Kanoh [19], we give a penalty of 20 s for each traffic light along the route.

**The Travel Time Variability Objective $f_v$**, also called travel time reliability, is another important factor for route choice [15]. It is mostly defined as the expected value of travel time, its variance or a combination of both [17,18,28,31]. Unlike these definitions, approaches employing a semi-standard deviation (SSD) of travel times [33,34] only minimise the risk of arriving too late at a destination. Since this is in the interest of most drivers, we use the upper SSD over varying departure times. As defined in Eq. 1, we measure the travel time variability of a route $x$ in the interval $\{t_0 + i \cdot 15 \mid i = -6, -5, \ldots, 6\}$ which is 90 min before and after a planned departure time in 15 min increments. $\overline{tt}$ refers to the mean of all 13 travel times observations. Hourly travel times of an edge are linearly interpolated depending on the minute of each observation. If this was not done, the algorithm would be encouraged to minimise travel time variability by taking detours until the next hour starts and thereby lower or no variability was reached for certain edges.

**The Ease of Driving Objective $f_s$** is inspired by Kanoh and Hara [21]. It prioritises the driving comfort which does not necessarily coincide with the fastest or most reliable path. For example, taking smaller roads through residential areas may avoid congested roads but is strenuous due to more turns. Therefore, our third objective maximises the ease of driving by minimising the sum of degrees of turning needed to drive along a route as specified in Eq. 1. $\angle\big(x[i], x[i+1], x[i+2]\big)$ computes the angle between edges $(x[i], x[i+1])$ and $(x[i+1], x[i+2])$ at node $x[i+1]$. Note that a turn can mean anything between a U-turn (180°) and a straightforward crossing of an intersection (0°). We subtract the turning angle from 180° since the sum of turning angles could simply be maximised by making routes longer. Some aspects such as avoiding congested streets, which also maximise the ease of driving, are already encouraged by our other objectives.

**Speed Attributes $s_{uv}^{wh}$** are exclusively based on historical speed data from the Uber Movement Berlin Speeds dataset [30]. This dataset contains mean hourly speeds and their standard deviation recorded from a sufficient number of Uber trips for Berlin street segments in January 2020. The two main advantages of this dataset are that it is publicly available and in an easily machine-readable CSV format. The biggest limitation of this dataset is the sparseness of the data, especially on smaller roads such as residential streets. Another problem is that the OpenStreetMap (OSM) start and end IDs of street segments provided by Uber

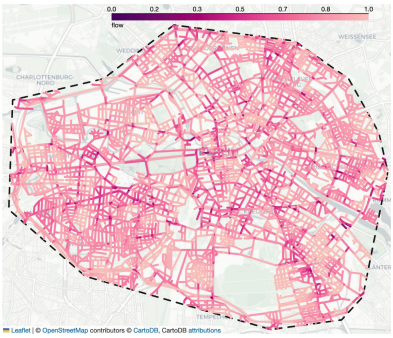



**Fig. 3.** Flows of the interpolated road network for the centre of Berlin (dashed polygon) at 8 pm during the week

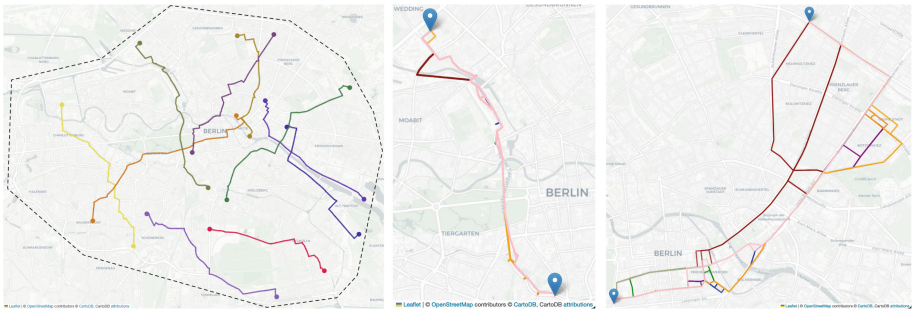
**Table 2.** Classes of innovization experiments for different sets of route planning problems

| length of routes in km | time of day (average flow in network) | weekend? |
|---|---|---|
| medium (3.5, 7] | 4 pm (worst, 75.3 %), 8 pm (medium, 85.5 %), 4 am (best, 97 %) | no |
| medium | 8 pm | yes, no |
| short [0, 3.5], long (7, ∞] | 8 pm | no |

Movements often do not match node IDs in the OSM drive network. Therefore, we preprocess the dataset by matching IDs to the closest edges and averaging speeds where multiple entries exist. Still, around 7 % of the data points cannot be matched. Since the data in the centre of Berlin is a little less sparse and less disconnected, we limit our road network to the polygon that is marked by a dashed line in Fig. 3. Still, only 14 % of edges have data for at least one hour of the day. To improve computation times, we further simplify the graph by removing interstitial nodes, multiples of edges between node pairs and dead ends. Lastly, we calculate speed attributes missing in the dataset by iteratively interpolating the flow from the respective incoming edges of $u$ and the outgoing edges of $v$. We define the flow $\rho_{uv}^{wh}$ of an edge $(u, v)$ at hour $h$ with weekend categorisation $w$ as $\rho_{uv}^{wh} = \frac{s_{uv}^{wh}}{s_{uv}^{\max}}$ where $s_{uv}^{\max}$ is the maximum allowed speed on edge $(u, v)$. Unassigned speed attributes are assumed to have 100 % flow at the beginning of the interpolation. The resulting, interpolated network graph is depicted in Fig. 3 for 8 pm during the week.

### 4.2 Knowledge Extraction

This subsection examines what knowledge can be gained from applying our innovization variant to problems from Subsect. 4.1. To extract representative knowledge, we compare three classes of innovization experiments for different sets of route planning problems, as listed in Table 2. Based on the diameter of the polygon of the centre of Berlin, we categorise routes by length according to the straight line distance between their origin and their destination. We randomly selected a set of ten representative routes for each length category. Medium-length routes are shown in Fig. 4a as an example. To create harder short route planning problems, their start and end points are limited to points of interest in the centre of Berlin based on [29]. Since the median correlation between main roads percentage and number of signals was relatively high across




(a) (b) (c)

**Fig. 4.** (a) Shortest paths of ten medium-length routing problems. (b, c) Decision space clustering for two medium-length problems at rush hour with different colours for different clusters, excluding noise

all experiments, we only consider the former in the analysis. We extracted the following four innovized principles.

**Updated Creation of Initial Populations** Across all experiments, Pareto-optimal routes can usually be separated into two groups. The first closely follows the straight line from start to destination and is often similar to the shortest path, as seen in Fig. 4b. This group exhibits lower travel times but higher turning degrees. The opposite normally holds for the second group like in Fig. 4c which uses a high percentage of main roads and motorways. This is consistent with the median Spearman correlation between the main roads percentage and the travel time of 0.597, respectively the degrees of turning of -0.663. In some experiments, only one of the two groups exists, such as for Fig. 4c, or both groups overlap. An overlap occurs, for example, if the straight line routes mostly use main roads as in Fig. 4b. Otherwise, the routes in the second group might take detours to use main roads and motorways which trivially lead to longer travel times. We observe a positive correlation between maximum detour length and route length.

With this knowledge, we adapt the creation of the initial population by splitting it in half. The first half is still created using guided random walks but $p_D$ is increased to 95 % to follow the straight line path more closely. Nonetheless, we no longer use the shortest path as a replacement when the creation gets stuck. This encouraged the optimisation to get stuck in local optima more easily since the shortest path is initially significantly better than individuals generated by guided random walks, but it can be misleading. Instead, we now finish stuck individuals by inserting the shortest path from the current node to the destination. The other half of individuals prefer main roads and motorways. When choosing the next edge, the outgoing edges are sorted first by street type, and second by their orientation towards the destination. To allow detours correlating with the length of the routes, the probability of choosing the most preferred edge is 85 % for short routes, 80 % for medium-length routes and 75 % for long routes.

However, since some destinations are only reachable via residential roads, we switch to a guided random walk again, when the destination is less than 250 m away.

**Exclusion of the $f_v$ Objective** Since Pareto-optimal solutions from our innovization experiments return relatively low values for the travel time variability, we conducted some additional MOEA runs without $f_v$. The worst recorded function value is a travel time variability of 2.25 seconds with and without the objective. Everyday drivers are unlikely to be concerned about a risk of being a couple of seconds too late. A $p$-value of around 0.5 from a two-tailed Mann-Whitney $U$ test confirms that the impact on the worst-case travel time variability of the Pareto-optimal solutions is insignificant. However, we hypothesise that the small function values are not due to the design of the objective but rather due to the sparseness of the dataset used. Through averaging and interpolation, we likely levelled out peaks in the data.

Interestingly, the additional MOEA runs also show that the travel time variability objective is responsible for small protrusions in routes that are visible, for example at the bottom of Fig. 4b. It appears that it is sometimes more reliable, for instance, to make a right turn and a U-turn to get back on the same road instead of just crossing straight through an intersection.

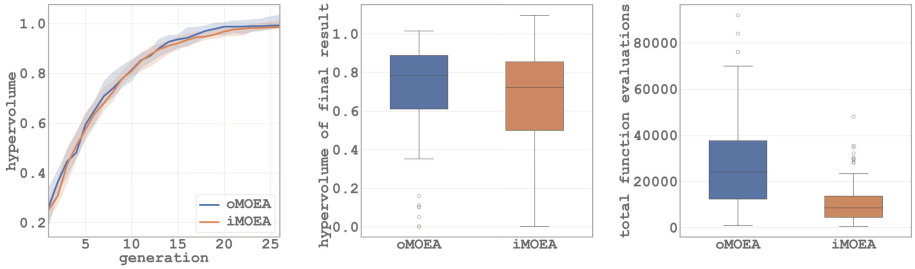
**Decrease in Population Size and Increase in Exploration** Leaving the second objective function out, makes the route planning problems easier to solve. The median number of Pareto-optimal solutions found by our EA is reduced to less than a fifth when optimising without $f_v$. Therefore, we can decrease the scaling coefficient $k_N$ from 4.0 to 2.5 to avoid unnecessary function evaluations.

Exemplary experiments have also shown that optimisations easily get stuck in local optima without the travel time variability objective. Getting stuck in local optima is a known problem for NSGA-II. To mitigate this problem, NSGA-II with controlled elitism [9] is used. This method encourages lateral diversity by keeping individuals from all fronts according to a geometric distribution with reduction rate $r$. In our case, setting $r = 0.2$ and increasing $p_m$ to 35 % for optimisations on the weekend has worked well in exemplary tests.

### 4.3 Algorithm Improvement

We now want to examine whether the innovized principles from Subsect. 4.2 can improve the efficiency or the results of our MOEA. We compare our original route planning MOEA (oMOEA) from Subsect. 3.1 to an adapted version (iMOEA) which implements the four innovized principles. Both algorithms are run on 100 randomly generated route planning problems. For the evaluation, we compute the efficiency, the quality of the final results and the total number of function evaluations for both algorithms.

First, we compare the efficiency of both algorithms. Based on Gupta, Ong and Feng [16], we define the efficiency of an optimisation algorithm on problem




(a) Evolution of median hypervolumes (b) Hypervolumes of final results (c) Total function evaluations

**Fig. 5.** (a) Evolution of the median hypervolume over generations for all experiments. Translucent areas show the respective 95 % confidence intervals. (b) Normalised hypervolumes of final results per algorithm and (c) total number of function evaluations needed to achieve the final results

instance $P$ as the normalised hypervolume $HV_t(P)$ of solutions achieved in $t$ time-steps on a designated computer. Consequently, this metric is independent of any hardware and the optimal Pareto front does not need to be known. In our case, $t$ is 26 because that is the maximum number of generations computed for all runs. Since we test our algorithms on different route planning problems which can return highly different fronts, we normalise all function values $f_m$ for each objective $m$ using $f_m^{norm} = \frac{f_m - z_m^*}{z_m^{nad} - z_m^*}$ with ideal point $z^*$ and nadir point $z^{nad}$ [7]. We use $(1.05, 1.05, 1.05)$ as a reference point. Since we omit the second objective from the optimisation in the iMOEA, it is also disregarded in all hypervolume computations. The oMOEA still uses $f_v$, but we compute the set of non-dominated solutions without the $f_v$ values before any hypervolume calculations. This only excludes irrelevant solutions that have a slightly better travel time variability but are worse for all other objectives. As shown in Fig. 5a, median hypervolumes from all experiments evolve almost identically over generations for both algorithms. The efficiency is virtually the same according to our definition because the median hypervolume of the iMOEA is only about 0.003 worse than that of the oMOEA after the $26^{\text{th}}$ generation. The insignificance of the difference is supported by a $p$-value of approximately 0.4 from a two-tailed Mann-Whitney $U$ test.

Since we allow early termination, we also compare the final results from all runs. In Fig. 5b, we can see that hypervolumes of final results are better when using the oMOEA. The median hypervolume of the oMOEA is around 0.79 while the one for the iMOEA is slightly worse at 0.72. However, a two-tailed Mann-Whitney $U$ test results in a $p$-value of 0.15, indicating that the difference in hypervolume is not significant. Remarkably, Fig. 5c demonstrates that the number of total functions evaluations for runs of the iMOEA is much lower than that of the oMOEA. In total, the iMOEA with 1,090,099 evaluations needs only about 40 % of the oMOEA with 2,813,520 evaluations.

## 5 Conclusion

In this paper, we presented innovization for route planning which is an adapted version of the innovization by Deb and Srinivasan [11]. We introduced PMM-LS, a local search for routing problems which systematically explores the neighbourhoods of routes. Using innovization for route planning, we extracted four innovized principles from multi-objective time-dependent route planning problems that can be reused for other route planning problems in Berlin. One of the major discoveries is that Pareto-optimal routes can typically be separated into two groups. The first group consists of routes that are close to the linear path from start to end point. These solutions usually exhibit lower travel times but higher degrees of turning. The second group is a set of longer routes that take faster roads and that have higher travel times but lower degrees of turning. The other main finding is that the travel time variability objective can be omitted because the differences in values are insignificant for decision makers. The overall small values can be due to limitations of the chosen dataset. In future, the impact of the travel time variability objective should be re-evaluated on a dataset with sufficient speed data, especially for use cases such as emergency services. Alternatively, more sophisticated approaches using estimation models for travel times or real-time traffic information could be integrated.

Lastly, we have shown that we can improve a MOEA using extracted knowledge. While the efficiency remained virtually the same and the quality of the final results was only insignificantly worse, we managed to drastically decrease the number of necessary function evaluations to two fifths. Consequently, similar quality solutions can be produced with far less computational effort using the knowledge extracted with our innovization for route planning methodology. This is particularly valuable for applications, where limited computational resources are available, such as mobile devices which are often utilised for routing.

As for the original innovization method, one limitation of innovization for route planning is its computation speed. Generally, this is not a drawback since innovization is intended to be used only once for knowledge extraction [11]. The benefit of the methodology arises from using extracted knowledge for speeding up optimisations of similar problems. Nevertheless, a trade-off between computation time and completeness of PMM-LS is possible via the parameters $\Delta_w$, $ws_{min}$ and $ws_{max}$. Moreover, only applying PMM-LS to a few well-distributed solutions as in the original innovization is possible. Furthermore, PMM-LS might be improved by search strategies besides the current greedy approach, by the integration into Variable Neighbourhood Search to break out of local optima, or by more exploration through a gradually decreasing mutation window. Future work could also accelerate the innovization process by parallelising some steps or by employing a heuristic for setting problem-specific $\epsilon$-values for OPTICS. A verification method for the Pareto-optimality of solutions, that can handle route planning problems, would also be a beneficial development. The analysis step could be extended by including more route characteristics. In general, we intend to use innovization for route planning with other datasets or for differently

defined path planning problems in future work to also verify the generalisability of algorithm improvements by our methodology.

**Acknowledgments.** This work was funded by the Federal Ministry for Economic Affairs and Climate Action (BMWK) of Germany as part of the "Zentrales Innovationsprogramm Mittelstand" (ZIM) - grant number KK5540301RF3. Speed data retrieved from Uber Movement, (c) 2023 Uber Technologies, Inc., https://movement.uber.com (dataset no longer freely accessible).

**Disclosure of Interests.** The authors have no competing interests to declare that are relevant to the content of this article.

## References

1. Ankerst, M., Breunig, M.M., Kriegel, H.P., Sander, J.: OPTICS: ordering points to identify the clustering structure. ACM SIGMOD Rec. **28**(2), 49–60 (1999)
2. Benson, H.P.: Existence of efficient solutions for vector maximization problems. J. Optim. Theory Appl. **26**, 569–580 (1978)
3. Blank, J., Deb, K.: pymoo: multi-objective optimization in python. IEEE Access **8**, 89497–89509 (2020)
4. Blank, J., Deb, K.: A running performance metric and termination criterion for evaluating evolutionary multi- and many-objective optimization algorithms. In: 2020 IEEE Congress on Evolutionary Computation (CEC), pp. 1–8. IEEE (2020)
5. Chitra, C., Subbaraj, P.: A nondominated sorting genetic algorithm for shortest path routing problem. Int. J. Comput. Inf. Eng. **4**(8), 1227–1235 (2010)
6. Coello Coello, C.A., et al.: Usable knowledge extraction in multi-objective optimization: an analytics and innovization perspective (WG3): personalized multi-objective optimization: an analytics perspective. In: The Dagstuhl Seminar, p. 70 (2018)
7. Deb, K.: Multi-Objective Optimization using Evolutionary Algorithms, Reprised Edition. Wiley, Chichester, England (2004)
8. Deb, K., Deb, D.: Analysing mutation schemes for real-parameter genetic algorithms. Int. J. Artif. Intell. Soft Comput. **4**(1), 1–28 (2014). https://doi.org/10.1504/IJAISC.2014.059280
9. Deb, K., Goel, T.: Controlled elitist non-dominated sorting genetic algorithms for better convergence. In: Zitzler, E., Thiele, L., Deb, K., Coello Coello, C.A., Corne, D. (eds.) Evolutionary Multi-Criterion Optimization, pp. 67–81. Springer, Berlin, Heidelberg (2001). https://doi.org/10.1007/3-540-44719-9_5
10. Deb, K., Pratap, A., Agarwal, S., Meyarivan, T.: A fast and elitist multiobjective genetic algorithm: NSGA-II. IEEE Trans. Evol. Comput. **6**(2), 182–197 (2002)
11. Deb, K., Srinivasan, A.: Innovization: innovating design principles through optimization. In: Proceedings of the 8th Annual Conference on Genetic and Evolutionary Computation, pp. 1629–1636 (2006)
12. Dijkstra, E.W.: A note on two problems in Connexion with graphs. Numerische Mathematik **1**, 269–271 (1959). https://doi.org/10.1145/3544585.3544600
13. Eiter, T., Mannila, H.: Computing discrete fréchet distance, Technical report CD-TR 94/64, Christian Doppler Labor für Expertensysteme (1994)
14. Ester, M., Kriegel, H.P., Sander, J., Xu, X.: A density-based algorithm for discovering clusters in large spatial databases with noise. In: KDD-96 Proceedings, pp. 226–231 (1996)

15. Gan, H., Bai, Y.: The effect of travel time variability on route choice decision: a generalized linear mixed model based analysis. Transportation **41**, 339–350 (2014)
16. Gupta, A., Ong, Y.S., Feng, L.: Insights on transfer optimization: because experience is the best teacher. IEEE Trans. Emerg. Top. Comput. Intell. **2**(1), 51–64 (2017)
17. Hussein, F.F., Süer, G.A., Naik, B.: Travel time modeling using non-linear multi-objective fuzzy optimization approach. In: International Conference on Transportation and Development 2020, pp. 104–118. American Society of Civil Engineers Reston, VA (2020)
18. Ji, Z., Chen, A., Subprasom, K.: Finding multi-objective paths in stochastic networks: a simulation-based genetic algorithm approach. In: Proceedings of the 2004 congress on evolutionary computation, vol. 1, pp. 174–180. IEEE (2004)
19. Kanoh, H.: Dynamic route planning for car navigation systems using virus genetic algorithms. Int. J. Knowl. Based Intell. Eng. Syst. **11**(1), 65–78 (2007)
20. Kanoh, H., Furukawa, T., Tsukahara, S., Hara, K., Nishi, H., Kurokawa, H.: Short-term traffic prediction using fuzzy c-means and cellular automata in a wide-area road network. In: Proceedings. 2005 IEEE Intelligent Transportation Systems, pp. 381–385. IEEE (2005)
21. Kanoh, H., Hara, K.: Hybrid genetic algorithm for dynamic multi-objective route planning with predicted traffic in a real-world road network. In: GECCO '08, pp. 657—664. Association for Computing Machinery, New York, NY, USA (2008). https://doi.org/10.1145/1389095.1389226
22. Lin, H.E., Zito, R., Taylor, M.: A review of travel-time prediction in transport and logistics. Proc. East. Asia Soc. Transp. Stud. **5**, 1433–1448 (2005)
23. Liu, Y.Y., Enayatollahi, F., Thulasiraman, P.: Traffic aware many-objective dynamic route planning. In: 2019 IEEE Symposium Series on Computational Intelligence (SSCI), pp. 1241–1248 (2019). https://doi.org/10.1109/SSCI44817.2019.9002725
24. Messac, A., Mattson, C.A.: Normal constraint method with guarantee of even representation of complete Pareto frontier. AIAA J. **42**(10), 2101–2111 (2004)
25. Munetomo, M., Takai, Y., Sato, Y.: A migration scheme for the genetic adaptive routing algorithm. In: SMC'98 Conference Proceedings, 1998 IEEE International Conference on Systems, Man, and Cybernetics (Cat. No. 98CH36218), vol. 3, pp. 2774–2779. IEEE (1998)
26. Rodriguez, M.A., Neubauer, P.: Constructions from dots and lines. Bull. Am. Soc. Inf. Sci. Technol. **36**(6), 35–41 (2010)
27. Röper, E.: Innovization for multi-objective time-dependent route planning (2024). https://ci.ovgu.de/is_media/Master+und+Bachelor_Arbeiten/MasterThesis_EvaRoeper-p-7664.pdf
28. Sen, S., Pillai, R., Joshi, S., Rathi, A.K.: A mean-variance model for route guidance in advanced traveler information systems. Transp. Sci. **35**(1), 37–49 (2001)
29. State of Berlin: Top 10 Berlin sights and attractions. https://www.berlin.de/en/attractions-and-sights/top-10/. Accessed 6 Oct 2023
30. Uber Technologies, Inc.: Historical speeds, hourly time series. https://movement.uber.com/cities/berlin/downloads/speeds?lang=en-US&tp[y]=2020&tp[q]=1. Accessed 14 Apr 2023
31. Wang, S., Yang, J., Liu, G., Du, S., Yan, J.: Multi-objective path finding in stochastic networks using a biogeography-based optimization method. Simulation **92**(7), 637–647 (2016)
32. Weise, J.: Evolutionary many-objective optimisation for pathfinding problems (2023). http://dx.doi.org/10.25673/101389

33. Zhang, X., Chen, M.: Genetic algorithm-based routing problem considering the travel reliability under asymmetrical travel time distributions. Transp. Res. Rec. **2567**(1), 114–121 (2016)
34. Zhang, X., Chen, M.: Bi-objective routing problem with asymmetrical travel time distributions. J. Intell. Transp. Syst. **22**(2), 87–98 (2018)

# Evolutionary Multi-objective Diversity Optimization

Anh Viet Do(✉), Mingyu Guo, Aneta Neumann, and Frank Neumann

Optimisation and Logistics, The University of Adelaide, Adelaide, Australia
vietanh.do@adelaide.edu.au

**Abstract.** Creating diverse sets of high-quality solutions has become an important problem in recent years. Previous works on diverse solutions problems consider solutions' objective quality and diversity where one is regarded as the optimization goal and the other as the constraint. In this paper, we treat this problem as a bi-objective optimization problem, which is to obtain a range of quality-diversity trade-offs. To address this problem, we frame the evolutionary process as evolving a population of populations, and present a suitable general implementation scheme that is compatible with existing evolutionary multi-objective search methods. We realize the scheme in NSGA-II and SPEA2, and test the methods on various instances of maximum coverage, maximum cut and minimum vertex cover problems. The resulting non-dominated populations exhibit rich qualitative features, giving insights into the optimization instances and the quality-diversity trade-offs they induce.

**Keywords:** Multi-Objective Optimization · Evolutionary Diversity Optimization

## 1 Introduction

Diverse solutions problems, seeking multiple maximally distinct solutions of high-quality instead of a single solution, have been studied for several decades [4,15,16,18–21,23,24]. They aim to fill the gaps in practical considerations left by traditional optimization. A set of diverse solutions provides robustness in order to deal with changes in the problems, which necessitate changes in current solutions. It also gives the users the choices to address the gaps between the problem models and real-world settings, frequently seen in complex applications with factors that are hard to define or measure [40]. Furthermore, diverse solution sets give the decision makers rich information about the problem instance by virtue of being diverse, which helps augment decision making capabilities. While there are methods to enumerate high-quality solutions, having too many overwhelms the decision makers [18], and a small, diverse subset can be more useful. It is also known that k-best enumeration tends to yield highly similar solutions, motivating the use of diversification mechanisms [22,46,49].


M. Affenzeller et al. (Eds.): PPSN 2024, LNCS 15151, pp. 117–134, 2024.
https://doi.org/10.1007/978-3-031-70085-9_8

The diverse solutions problems have been studied as an extension to many important and difficult problems. These include constraint satisfaction and optimization problems [23,35,39], SAT and answer set problem [13,27], and mixed integer programming [9,18,41]. Recent fixed-parameter tractable algorithms for various graph-based vertex problems [4] inspired considerations of other combinatorial structures such as trees, paths [20,21], matching [15], independent sets [16], and linear orders [2]. Furthermore, general frameworks have been proposed for diverse solutions to any combinatorial problem [19,24].

This area of research manifested in Evolutionary Computation literature as *Evolutionary diversity optimization* (EDO). The idea was investigated as early as in the work of Ronald [37] and Zechman and Ranjithan [50,51], motivated by practical concerns in real-world problems. The topic was then studied by Ulrich et al. with emphasis on conceptual frameworks [43,44] and has subsequently gained significant attention within the evolutionary computation community. This represents a shift in perspective on diversity, from a necessity in evolutionary search to an optimization goal. Incidentally, around the same time, aspects of evolutionary searches other than diversity were also investigated to address issues that motivated EDO, such as high-performance regions [33] and solution's robustness [8,42].

Studies on EDO typically involve defining a search space $S$, an objective function $f$ (to be maximized), a quality threshold $T$, a diversity measure $d$, an integer $r$ and applying evolutionary algorithms to solve the optimization problem

$$\max_{P \in 2^S : |P| = r} d(P), \quad s.t. \quad \forall x \in P, f(x) \geq T. \tag{1}$$

Under this paradigm, evolutionary techniques have been investigated in computing diverse Traveling Salesperson Problem (TSP) solutions [11,31,32], knapsack packings [6], minimum spanning trees [7], and submodular optimization solutions [28]. On the other hand, EDO has seen application in generating images with varying features [1], or to compute diverse TSP instances [5,17] useful for automated algorithm selection and configuration [26]. Different indicators for measuring the diversity of sets of solutions in such as the star discrepancy [29] or those from the area of evolutionary multi-objective optimization [30] have been considered in this paradigm as well.

In this work, we address the treatment of objective quality and diversity of solutions as equal optimization goals. In practice, users may not have enough information about the problem instance in order to formulate quality or diversity criteria that would lead to meaningful optimization outcomes. We re-frame the problem as finding a range of quality-diversity trade-offs which hopefully provides such information to an extent, as well as giving a collection of diverse solution sets to choose from. This approach involves an atypical way of looking at the evolutionary process: each population (i.e., set of solutions) is a unit of evolution, with its own fitness. We describe a basic implementation scheme under this paradigm, which can be realized directly in existing evolutionary algorithms. Concrete examples of its realization are given using NSGA-II and SPEA2 in finding diverse solutions to maximum cut, maximum coverage and minimum

vertex cover instances. Experimental investigations give trade-offs of highly varied natures across problems, revealing interesting characteristics of the objective landscapes. Furthermore, these results indicate that the diverse solutions problems can be reasonably addressed under the proposed bi-objective optimization paradigm.

Recently, there have been studies that consider multi-objective optimization within the Quality-Diversity paradigm [36,47]. These investigations look at the integration of multiple objectives into the task of filling the behavioral characteristics space. We remark that these are different from the application of multi-objective optimization within this work, which is set up to tackle the quality-diversity trade-offs directly.

This paper is structured as follows. We include the bi-objective optimization formulation in Sect. 2. The implementation scheme is described in Sect. 3, which is realized in the experimental investigations detailed in Sect. 4. We conclude the paper in Sect. 5.

## 2 Evolutionary Multi-objective Diversity Optimization

Given an objective function $f$ over a space of feasible solution $S$, the classical optimization problem is specified with $\max_{x \in S} f(x)$.[1] In addition, given a diversity measure $d$, an set-aggregating function $F$, and integer $r$, we consider the following bi-objective problem

$$\max_{P \in 2^S : |P| = r} (f_1, f_2), \quad f_1(P) := F\{f(x) : x \in P\}, \quad f_2(P) := d(P).$$

Intuitively, the problem asks to find a set of $r$ solutions representing the best trade-off between diversity and aggregated objective quality. Note this formulation differs from the existing EDO paradigm (Eq. (1)) in that it does not involve a quality threshold. This makes our formulation suitable for opaque instances for which the appropriate thresholds (in relation to a certain goal) are not known.

We consider two aggregating functions in this work: minimum ($F := \min$) and average ($F := \text{avg}$). The former consideration models the worst-case robustness requirement on a solution set, and the latter relaxes this requirement to average-case. As for diversity, we look at distance sum measure, frequently considered in dispersion problems [14,45] and recent works on diverse solutions [4,19–21]. We give precise definitions in later sections, as it depends on the underlying optimization problem (as specified by $f$ and $S$).

As with many multi-objective problems, the actual computational goal is to find a set of non-dominated trade-offs w.r.t. the objectives. Let $f_i$ be the $i$-th objective function, we say solution $x$ dominates solution $y$ ($x \succeq y$) if $\min_i\{f_i(x) - f_i(y)\} \geq 0$, and strong dominance occurs ($x \succ y$) if, in addition, $\max_i\{f_i(x) - f_i(y)\} > 0$. Thus, a set of solutions is non-dominated if it contains no pair that exhibits dominance relation.

[1] This and subsequent formulation also apply to minimization, with trivial differences.

The problem requires searching in the population space $2^S$ rather than the solution space $S$. Most EAs evolve a population of solutions, whereas we are interested in evolving a population of "populations". As such, for the rest of this paper, we modify the classical terminologies in literature to fit the context of the problem. We use "solution" to refer to an element $x \in S$, whose quality is $f(x)$. Furthermore, "individual" refers to a set of solutions, i.e., an element $I \in 2^S$, and "population" refers to a set of individuals[2].

## 3 Implementing Populations as Individuals

As the problem we consider is fundamentally a bi-objective optimization problem, applying existing bi-objective optimizers would be, in principle, sufficient in solving it. In this section, we discuss several considerations regarding the implementation of such algorithms that are unique to this problem.

### 3.1 Individual Representation

In typical EAs, a population is evolved via the selection-variation-replacement paradigm. However, in this work, we treat populations as individuals, so as to apply standard variation operations on the entire set of solutions directly, effectively performing the 3-step procedure in one 1-step procedure. This is to both simplify the overall algorithm, and ease restrictions on the variation neighborhood in the population space. As such, an individual should be represented in a way that both encodes the information about the solution set and allows efficient implementation of variation operators.

In this work, we adopt the simple concatenation scheme: let $x_i$ be the representation of the $i$-th solution in the individual $I$, $I$ is represented by $x_1 x_2 \dots x_r$. For instance, if $x_i$ is a $n$-length bit string, then $I$ is encoded in a $rn$-length bit string, in which each solution is encoded directly as a substring. Having individual representation mirroring solution representation allows each solution to undergo variation when it is applied to the individual, and in the same manner. For example, performing standard bit mutation on the individual essentially does the same to each solution, with the same bit-flip probability. While the scheme is not applicable as-is to non-linear[3] solution representations, it suffices for our applications.

### 3.2 Recombination

We see that the concatenation representation is not unique: the solutions ordering itself encodes redundant information in the context of the problem. While positional-bias-free operations (e.g., standard bit mutation) are unaffected by

[2] "Set" is used loosely: they are technically multisets as self-avoidance is not enforced.

[3] One could extend this scheme to any solution representation with recursive structure (e.g., tree–subtree, string–substring, vector–subspace projection).

such an artefact, the same cannot be said for recombination as a means to transmit parents' genetic information to offspring. Ensuring transmission of such redundant information may limit the flexibility of the operation, so it should be removed from the process.

Here, we choose to remove such information prior to recombination by shuffling the solutions' positions in one of the parents. This allows each solution in one parent individual to be recombined with any solution in the other parent individual with equal probability. Note that it is sufficient to do this to one parent individual since the solution-to-solution mapping between the parents is what matters, not the ordering itself. Additionally, this scheme introduces an additive $\Theta(r)$ overhead, which is small compared to the $\Theta(rn)$ time cost of the recombination.

## 4 Problem Settings and Experiments

Here, we realize our proposed scheme into concrete algorithms, and perform experimental investigations in max coverage, max cut and min vertex cover problems, all of which are NP-hard. These combinatorial problems are formulated in a way that affords us insights on achievable diversity and its interplay with solution quality; this eases contextualizing and interpreting the results. We select instances from standard benchmark suites and, as we will see, observe qualitatively varied results across problems.

For these graph problems, a solution is a subset of vertices, and the standard representation is indicator vector as bit-string: each bit corresponds to a vertex and assumes value 1 if and only if it is a member of the subset. Our scheme implies that each individual is represented as a $rn$-length bit-string where $n$ is the number of vertices. Also, we use Hamming distance to compute diversity among solutions within an individual, which is the size of two sets' symmetric difference: $|A \Delta B|$. The sum of Hamming distances among a set of bit-strings can be computed and updated efficiently without calculating pairwise distances[4]. For consistency, we use set operators with bit-string notations, e.g., $x \in I$ denotes a solution $x$ being in an individual $I$.

The problems impose upper bounds or lower bounds on the sizes of feasible solutions, which we can use to derive upper bounds on diversity. Let $V$ be a ground set, $n := |V|$, $S := \{Z \subseteq V : |Z| \leq b\}$ for some integer $b \geq 0$, and the diversity function $d(I) := \sum_{x,y \in I} |x \Delta y|$ (each pair is counted once), then we have

$$\max_{P \in 2^S : |P| = r} d(P) = g(n, b, r) := nq(r - q) + m(r - 2q - 1), \tag{2}$$

where $h = \min\{b, n/2\}$, and $m \in [0, n), q$ are integers such that $\lceil r/2 \rceil \lceil h \rceil + \lfloor r/2 \rfloor \lfloor h \rfloor = qn + m$. Intuitively, $d(I)$ is maximized when the number of 1 (or 0) values in $I$ are as close to being equal across bits as possible, and as close to $r/2$ as possible. The reader can confirm that such a configuration under the

[4] The former takes $\Theta(rn)$ arithmetic operations, and the latter takes $\Theta(k)$ where $k$ is the number of changed bits.

restriction specified by $S$ yields the right hand side of Eq. (2) when plugged into $d$. We note that the restriction imposed by the graph problems can be explicit as in cardinality constraint, and implicit in the form of optimal objective value.

We use the well-known NSGA-II [10] and SPEA2 [52] algorithms with our proposed scheme and the following setting, unless stated otherwise. Specifically, we use implementations in jMetal 5.11 of these algorithms and their standard components [12], and unlisted parameters are assigned the default values.

- Initialization: Uniform random
- Output set size: $r \in \{10, 20\}$
- Population/Offspring pool size: $N = 20$
- Parent selection: Binary tournament
- Crossover method, rate: Uniform crossover, 80%
- Mutation method, strength: Standard bit mutation[5], $\chi = 0.5/n$
- Evaluation budget: $5rnN$
- Number of independent runs per instance: 20

To handle constraint in max coverage and min vertex cover, we penalize the objectives with violation degree. The constraint violation degree of an individual is the sum of such degrees over its solutions: $C(I) := \sum_{x \in I} C(x)$. We detail the settings in the next sections.

### 4.1 Maximum Cut

Given an undirected graph $G = (V, E)$[6], max cut problem asks to find a solution in

$$\operatorname*{argmax}_{x \subseteq V} f(x) := |\{e \in E : |e \cap x| = 1\}|.$$

We see that measuring diversity in the vertex space is inappropriate due to $f$ being symmetric. By taking any solution $x$, and duplicating it and $V \setminus x$, we get a population of maximum diversity $g(n, n, r)$ where $g$ is given in Eq. (2), and arbitrary quality, effectively reducing the optimal front into a singular point regardless of graph structure. Therefore, we choose to measure diversity in the edge space instead. Let the cut edges of $x$ be $E(x) := \{e \in E : |e \cap x| = 1\}$, we have the following fitness functions:

- Minimum objective setting: $f_1(I) := \min_{x \in I} f(x)$.
- Average objective setting: $f_1(I) := \frac{1}{r} \sum_{x \in I} f(x)$.
- $f_2(I) := \sum_{x,y \in I} |E(x) \Delta E(y)|$.

The aforementioned insight implies we can upper bound diversity by $g(|E|, OPT, r)$ where $OPT$ is the max cut value. We use the five unweighted instances in G-set benchmark [48], containing 800 vertices each, and whose optimal values are known. These values significantly exceed $|E|/2$, meaning for these instances,

[5] We find higher mutation strengths yield lower final diversity.
[6] Each edge is regarded as a set of two vertices that are its endpoints.

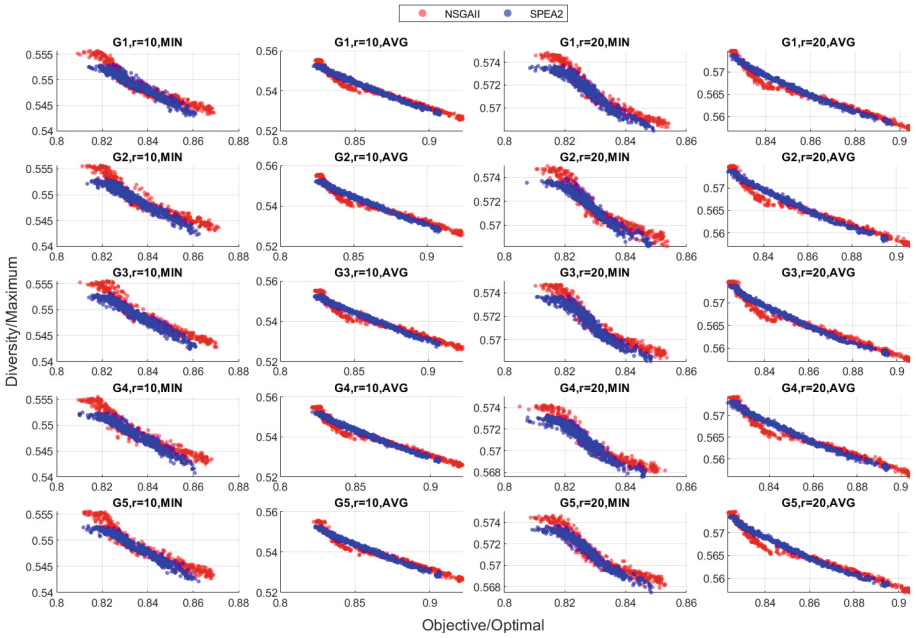



**Fig. 1.** Unions of final populations across all runs on Max Cut instances. "MIN" indicates the minimum objective setting being used for $f_1$ while "AVG" indicates average objective setting. For each run, dominated points are excluded.

as cuts in an individual $I$ approach the optimal, more cut edges become over-represented in $I$, diminishing $f_2(I)$. Thus, if the (collective) quality exceeds $|E|/2$, its correlation to diversity is negative, otherwise it is positive. Such conflict between fitnesses should induce rich non-dominated fronts with clear shapes. Note the bound $g(|E|, OPT, r)$ might not be tight since not every edge subset constitutes a cut.

The final populations are visualized in Fig. 1 and Modified Inverted Generational Distance (IGD+), Hypervolume (HV)[7] and numbers of non-dominated individuals are reported in Table 1. In the table, IGD+ and HV denote values normalized against the extreme objective and diversity values, while IGD+* and HV* are normalized against the best non-dominated fronts aggregated from all runs. Boldface denotes greater medians between two algorithms with statistical significance indicated by Wilcoxon signed-rank tests at 99% confidence level. The same procedure is used in subsequent experiments.

Overall, the algorithms find many individuals on the non-dominated fronts between 81% and 92% of the optimal, consistently across instances. This reveals rich trade-offs, aligning with our intuition regarding quality-diversity correlation. For these instances, 82% of the optimal cut is approximately $|E|/2$, a point

[7] Smaller IGD+ and greater HV indicate better trade-offs. More comprehensive discussions on performance indicators can be found in [3].

**Table 1.** Medians of indicator scores and numbers of non-dominated individuals across runs on Max Cut instances. Boldface denotes greater medians between two algorithms with statistical significance via Wilcoxon signed-rank tests ($\alpha = 1\%$).

| | Inst. | $r$ | NSGA-II | | | | | SPEA2 | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | IGD+ | HV | IGD+* | HV* | # | IGD+ | HV | IGD+* | HV* | # |
| Min objective | G1 | 10 | 0.49044 | **0.46656** | 1.9078e-3 | **0.99578** | 20 | **0.49342** | 0.45972 | **4.4738e-3** | 0.98118 | 20 |
| | | 20 | 0.48715 | **0.46116** | 1.0426e-3 | **0.9969** | 20 | **0.48897** | 0.45734 | **2.3626e-3** | 0.98864 | 20 |
| | G2 | 10 | 0.48982 | **0.46735** | 2.3060e-3 | **0.99428** | 20 | **0.49276** | 0.46058 | **5.2202e-3** | 0.97989 | 20 |
| | | 20 | 0.48601 | **0.46213** | 1.1482e-3 | **0.99718** | 20 | **0.48792** | 0.45845 | **2.2096e-3** | 0.98923 | 20 |
| | G3 | 10 | 0.49041 | **0.46697** | 2.1430e-3 | **0.99505** | 20 | **0.49297** | 0.46036 | **5.3971e-3** | 0.98098 | 20 |
| | | 20 | 0.48658 | **0.46151** | 1.0371e-3 | **0.99761** | 20 | **0.48854** | 0.45783 | **2.3923e-3** | 0.98966 | 20 |
| | G4 | 10 | 0.49431 | **0.46269** | 1.9890e-3 | **0.99442** | 20 | **0.49703** | 0.4562 | **4.8654e-3** | 0.98049 | **20** |
| | | 20 | 0.49365 | **0.45447** | 1.2396e-3 | **0.99538** | 20 | **0.49527** | 0.45151 | **2.3533e-3** | 0.9889 | 20 |
| | G5 | 10 | 0.49179 | **0.46542** | 1.9920e-3 | **0.99573** | 20 | **0.49457** | 0.45868 | **5.2354e-3** | 0.98133 | 20 |
| | | 20 | 0.48932 | **0.45888** | 1.2962e-3 | **0.99697** | 20 | **0.49097** | 0.45546 | **2.2058e-3** | 0.98954 | 20 |
| Average objective | G1 | 10 | 0.49325 | **0.49447** | 3.2364e-3 | **0.99462** | 20 | **0.49434** | 0.48454 | 3.5350e-3 | 0.97463 | **20** |
| | | 20 | 0.48441 | **0.48895** | 2.0149e-3 | **0.99706** | 20 | **0.48512** | 0.48259 | **2.6200e-3** | 0.9841 | 20 |
| | G2 | 10 | 0.49237 | **0.49567** | 3.5111e-3 | **0.99393** | 20 | **0.49372** | 0.48534 | **4.3330e-3** | 0.97323 | 20 |
| | | 20 | 0.4831 | **0.49009** | 2.0846e-3 | **0.99641** | 20 | **0.48407** | 0.48378 | **2.6630e-3** | 0.98358 | 20 |
| | G3 | 10 | 0.49265 | **0.49475** | 3.0740e-3 | **0.99376** | 20 | **0.49398** | 0.48461 | **3.9748e-3** | 0.9734 | 20 |
| | | 20 | 0.48366 | **0.48965** | 2.0681e-3 | **0.99707** | 20 | **0.48436** | 0.48324 | **2.4223e-3** | 0.98401 | 20 |
| | G4 | 10 | 0.49668 | **0.49053** | 3.0533e-3 | **0.99359** | 20 | **0.49812** | 0.48078 | **3.9520e-3** | 0.97384 | 20 |
| | | 20 | 0.49048 | **0.48229** | 2.1208e-3 | **0.99644** | 20 | **0.49158** | 0.47644 | **2.4823e-3** | 0.98434 | 20 |
| | G5 | 10 | 0.49413 | **0.49325** | 3.0788e-3 | **0.99418** | 20 | **0.49542** | 0.48336 | **3.9952e-3** | 0.97424 | 20 |
| | | 20 | 0.48627 | **0.48687** | 1.8234e-3 | **0.99703** | 20 | **0.48706** | 0.48072 | **2.2280e-3** | 0.98443 | 20 |

below which we predict the quality-diversity correlation to be positive. This is supported by the shape of the obtained non-dominated fronts within this range, observed more clearly in minimum objective settings.

We see NSGA-II and SPEA2 perform similarly on these instances. Their output non-dominated fronts largely overlap, which small differences in shapes. Inspecting indicators suggests that NSGA-II consistently produces better trade-offs than SPEA2's with statistically significant differences. However, these differences are small in relative magnitude.

We observe that the algorithms consistently reach plateaus before exhausting the budgets. As the Hamming distance sum is easy to maximize, this gives us confidence that the outputs approximate well the high diversity extreme of the optimal fronts. On the other hand, we know there are individuals on the high-quality (low-diversity) end of the fronts that the algorithms fail to approximate as closely. Doing so requires improving the objective quality of all solutions simultaneously. Therefore, expanding the front toward the high-quality end may call for the use of high-performing white-box heuristics.

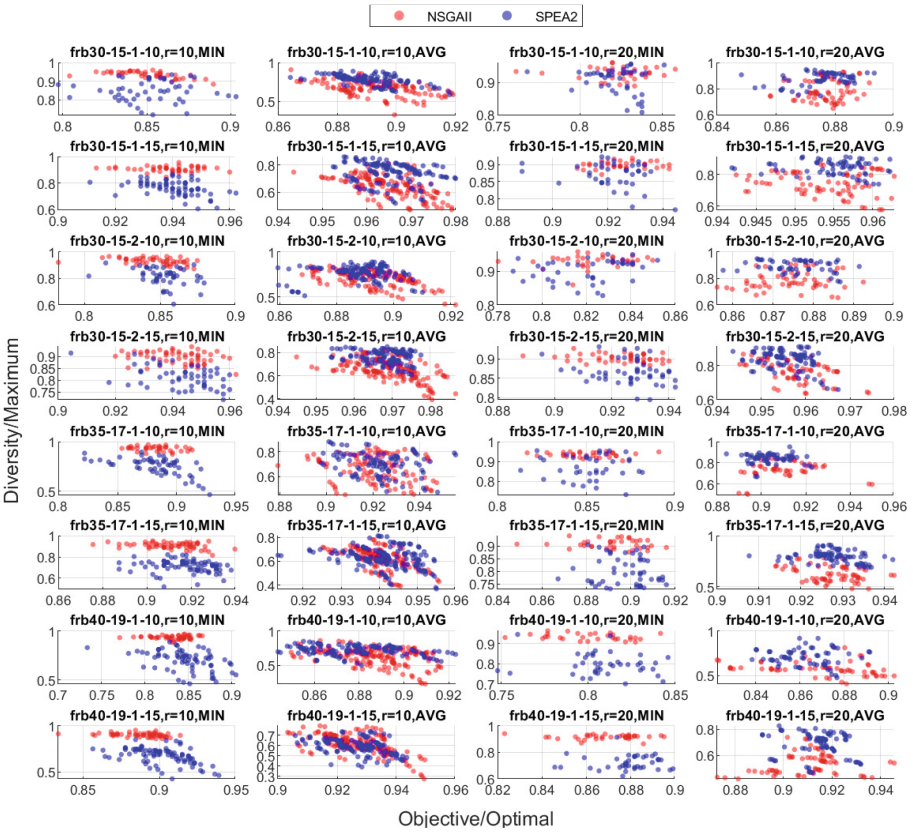



**Fig. 2.** Unions of final populations across all runs on Max Coverage instances. "MIN" indicates the minimum objective setting being used for $f_1$ while "AVG" indicates average objective setting. For each run, dominated points are excluded.

### 4.2 Maximum Coverage

Given an undirected graph $G = (V, E)$ and threshold $B$, max coverage asks to find a solution in

$$\underset{x \subseteq V : |x| \leq B}{\operatorname{argmax}} \; f(x) := |x \cup \{v \in V : \exists u \in x, \{v, u\} \in E\}|.$$

The constraint violation function on a solution is given by $C(x) := max\{|x| - B, 0\}$. We define the fitness functions:

- Minimum objective setting: $f_1(I) := \min_{X \in I} f(X)$ if $I$ is feasible, $f_1(I) := -C(I)$ otherwise.
- Average objective setting: $f_1(I) := \frac{1}{r} \sum_{x \in I : C(x) = 0} f(x) - C(I)$.
- $f_2(I) := \sum_{x,y \in I} |x \Delta y| - r|V|C(I)$.

**Table 2.** Medians of indicator scores and numbers of non-dominated individuals across runs on Max Coverage instances. Boldface denotes greater medians between two algorithms with statistical significance via Wilcoxon signed-rank tests ($\alpha = 1\%$).

| | Inst. | $r$ | NSGA-II | | | | | SPEA2 | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | IGD+ | HV | IGD+* | HV* | # | IGD+ | HV | IGD+* | HV* | # |
| Min objective | frb30-15-1-10 | 10 | 0.15929 | **0.81026** | 0.029592 | **0.93453** | 2 | **0.20494** | 0.74484 | **0.070245** | 0.85907 | 2 |
| | | 20 | 0.1832 | 0.77547 | 0.031759 | 0.94084 | 2 | 0.19708 | 0.76149 | 0.044568 | 0.92388 | 2 |
| | frb30-15-1-15 | 10 | 0.10914 | **0.86323** | 0.025463 | **0.9374** | 2 | **0.20583** | 0.7614 | **0.11083** | 0.82681 | 3 |
| | | 20 | 0.12695 | **0.83641** | 0.02194 | **0.96042** | 1 | **0.14855** | 0.80815 | **0.042557** | 0.92797 | 2 |
| | frb30-15-2-10 | 10 | 0.16417 | **0.80583** | 0.027673 | **0.94845** | 2 | **0.21283** | 0.73197 | **0.079468** | 0.86151 | 2 |
| | | 20 | 0.17935 | 0.7798 | 0.021634 | 0.956 | 2 | 0.19764 | 0.76273 | **0.038031** | 0.93507 | 2 |
| | frb30-15-2-15 | 10 | 0.10548 | **0.87193** | 0.019205 | **0.96324** | 2.5 | **0.17837** | 0.78967 | **0.079967** | 0.87237 | 2.5 |
| | | 20 | 0.12098 | **0.84317** | 0.019826 | **0.96042** | 2 | **0.15161** | 0.80867 | **0.048502** | 0.92112 | 2 |
| | frb35-17-1-10 | 10 | 0.13118 | **0.8378** | 0.03164 | **0.93202** | 2 | **0.2302** | 0.72143 | **0.10349** | 0.80256 | 2 |
| | | 20 | 0.15653 | **0.80693** | 0.024794 | **0.94912** | 1 | **0.19487** | 0.74615 | **0.093325** | 0.87763 | 2 |
| | frb35-17-1-15 | 10 | 0.12243 | **0.84888** | 0.03123 | **0.94851** | 3 | **0.2624** | 0.69466 | **0.1835** | 0.77619 | 4 |
| | | 20 | 0.13673 | **0.81768** | 0.025891 | **0.94968** | 2 | **0.22212** | 0.7338 | **0.12012** | 0.85226 | 3 |
| | frb40-19-1-10 | 10 | 0.17484 | **0.78967** | 0.044668 | **0.90275** | 2 | **0.292** | 0.67112 | **0.097538** | 0.76722 | 3 |
| | | 20 | 0.2006 | **0.75525** | 0.038091 | **0.92657** | 1 | **0.27879** | 0.64764 | **0.14262** | 0.79455 | 2 |
| | frb40-19-1-15 | 10 | 0.14224 | **0.8172** | 0.031981 | **0.91518** | 3 | **0.28658** | 0.66474 | **0.092133** | 0.74444 | **4** |
| | | 20 | 0.14977 | **0.80752** | 0.021691 | **0.9544** | 2 | **0.30992** | 0.62822 | **0.19501** | 0.74249 | 2 |
| Average objective | frb30-15-1-10 | 10 | **0.27011** | 0.67792 | **0.066477** | 0.81149 | 6 | 0.21687 | **0.73174** | 0.035662 | **0.87592** | 5 |
| | | 20 | **0.25191** | 0.68919 | **0.16121** | 0.81487 | 3 | 0.16947 | **0.7829** | 0.048992 | **0.92568** | 2.5 |
| | frb30-15-1-15 | 10 | **0.28312** | 0.69755 | **0.096594** | 0.82766 | 6 | 0.20168 | **0.77957** | 0.027407 | **0.92498** | 6 |
| | | 20 | **0.2467** | 0.72253 | **0.13882** | 0.82469 | 4 | 0.15011 | **0.82141** | 0.041561 | **0.93754** | 3 |
| | frb30-15-2-10 | 10 | 0.25897 | 0.68825 | 0.049847 | 0.84212 | 7 | 0.21594 | 0.73583 | 0.028128 | 0.90033 | 6 |
| | | 20 | **0.24879** | 0.69083 | **0.13418** | 0.8167 | 3 | 0.15323 | **0.79726** | 0.030719 | **0.94252** | 2.5 |
| | frb30-15-2-15 | 10 | **0.26177** | 0.71947 | **0.053042** | 0.85601 | **7** | 0.20224 | **0.77967** | 0.018827 | **0.92764** | 5 |
| | | 20 | **0.22441** | 0.74949 | **0.050041** | 0.84491 | 3 | 0.14728 | **0.82496** | 0.019408 | **0.93** | **4** |
| | frb35-17-1-10 | 10 | 0.26252 | 0.69361 | 0.095864 | 0.82278 | 5 | 0.2524 | 0.70518 | 0.083781 | 0.83649 | 5 |
| | | 20 | **0.27384** | 0.67771 | **0.091109** | 0.75009 | 3 | 0.17247 | **0.78145** | 0.039561 | **0.86492** | 3 |
| | frb35-17-1-15 | 10 | 0.31072 | 0.65554 | 0.061725 | 0.84496 | 6.5 | 0.32269 | 0.64837 | 0.070474 | 0.83571 | 6.5 |
| | | 20 | **0.38841** | 0.57861 | **0.23436** | 0.6746 | 3 | 0.20157 | **0.75954** | 0.046485 | **0.88555** | 4 |
| | frb40-19-1-10 | 10 | 0.29465 | 0.65239 | 0.069837 | 0.82183 | 7 | 0.26244 | 0.68673 | 0.050005 | 0.86509 | 5 |
| | | 20 | **0.44666** | 0.50248 | **0.23174** | 0.60785 | 3 | 0.28296 | **0.65226** | 0.081098 | **0.78904** | 3 |
| | frb40-19-1-15 | 10 | 0.3006 | 0.66013 | 0.046622 | 0.87708 | 7 | 0.35017 | 0.61199 | 0.078987 | 0.81312 | 6.5 |
| | | 20 | **0.45212** | 0.51614 | **0.20746** | 0.65887 | 3 | 0.28312 | **0.67128** | 0.050441 | **0.85692** | 3 |

We find that the algorithm struggles to maintain feasibility of many solutions simultaneously[8], so we use a modified mutation operator. It flips each 1-bit with probability $\chi(C(x)+1)$, 0-bit with probability $\chi$, where $x$ is the solution it belongs to. This reduces generation of infeasible offspring and mitigates stagnation, with small overhead.

We use the complement of BHOSLIB instances obtained from [38], denoted with {graphname}–{threshold}. While these graphs are sparse, the coverage functions they induce exhibit high degrees of multimodality, i.e., there are many distant near-optimal local optima. This means we can expect the optimal non-dominated front to be close to $(OPT, g(n, B, r))$, the former being the maximum

[8] Preliminary runs with standard mutation yield no feasible outputs when $r = 20$.

coverage value and the latter being the diversity upper bound. Note that we can construct a feasible individual $I$ with $f_2(I) = g(n, B, r)$, implying the bound is tight and always meets the optimal front at an individual.

The results are shown in Fig. 2 and Table 2. The algorithms return individuals with high objective values, and relatively small variations along this dimension. This leads to fewer non-dominated individuals from each run, as only few fitness values are occupied. Furthermore, most output individuals have high diversity, reaching above 80% of the upper bound in many instances. High Hypervolumes and small Inverted Generational Distances against $(OPT, g(n, B, r))$ indicate that the output non-dominated fronts occupy a large part of the fitness space; this implies that the optimal front is close to $(OPT, g(n, B, r))$, as we suspect.

The two algorithms seem to perform similarly on these instances. The fronts produced by NSGA-II have slightly higher diversity than those by SPEA2 in the minimum objective settings, while the rest see large overlaps. The achieved indicators show mixed comparisons. In min objective setting, NSGA-II achieves better non-dominated fronts in most cases. In average objective setting, SPEA2 reaches better indicator scores, at greater frequency when $r = 20$.

### 4.3 Minimum Vertex Cover

Given an undirected graph $G = (V, E)$, min vertex cover asks to find a solution in

$$\operatorname*{argmax}_{x \subseteq V} f(x) := |V \setminus x| \quad s.t. \quad \forall e \in E, e \setminus x \neq \emptyset.$$

The constraint violation function on a solution is the number of edges not covered $C(x) := |\{e \in E : e \cap x = \emptyset\}|$. We define the fitness functions:

- Minimum objective setting: $f_1(I) := \min_{X \in I} f(x)$ if $I$ is feasible, $f_1(I) := -C(I)$ otherwise.
- Average objective setting: $f_1(I) := \frac{1}{r} \sum_{x \in I : C(x) = 0} f(x) - C(I)$.
- $f_2(I) := \sum_{x,y \in I} |x \Delta y| - r|E|C(I)$.

In addition, we implement a simple repair heuristic used in [34]. After mutation, infeasible solutions are repaired with the procedure outlined in Algorithm 1. This incurs an additive $\Theta(|E|)$ overhead per infeasible solution, similar to the time cost of feasibility checking[9]. When this heuristic is used, $C(I)$ is not computed.

We choose nine hard BHOSLIB instances and complement of three DIMACS instances for the maximum clique problem [25]. BHOSLIB instances contain only large vertex covers, so maximizing objective correlates with maximizing diversity as both involves minimizing common selected vertices. This also means there are few possible objective values among feasible solutions, so we can expect the non-dominated fronts returned by the algorithms to be sparse. Meanwhile, selected DIMACS graphs are regular, inducing diverse sets of minimum vertex covers. This means we can expect the optimal non-dominated fronts in these instances to be close to $(OPT, g(n, OPT, r))$ where $OPT$ is the optimal objective value.

[9] The cost is further reduced by streamlining repairing with evaluation.

**Algorithm 1:** Repair heuristic for Max Vertex Cover [34]

**Input**: $G = (V, E)$, $x \subseteq V$
**Output**: $x$
**for** *0-bit $i$ in $x$ in random order* **do**
  **if** *$i$ is incident to an edge not covered by $x$* **then**
    $x_i \leftarrow 1$, update covered edges;
**for** *1-bit $j$ in $x$ in random order* **do**
  **if** *$j$ is adjacent to no vertex in $V \setminus x$* **then**
    $x_j \leftarrow 0$;

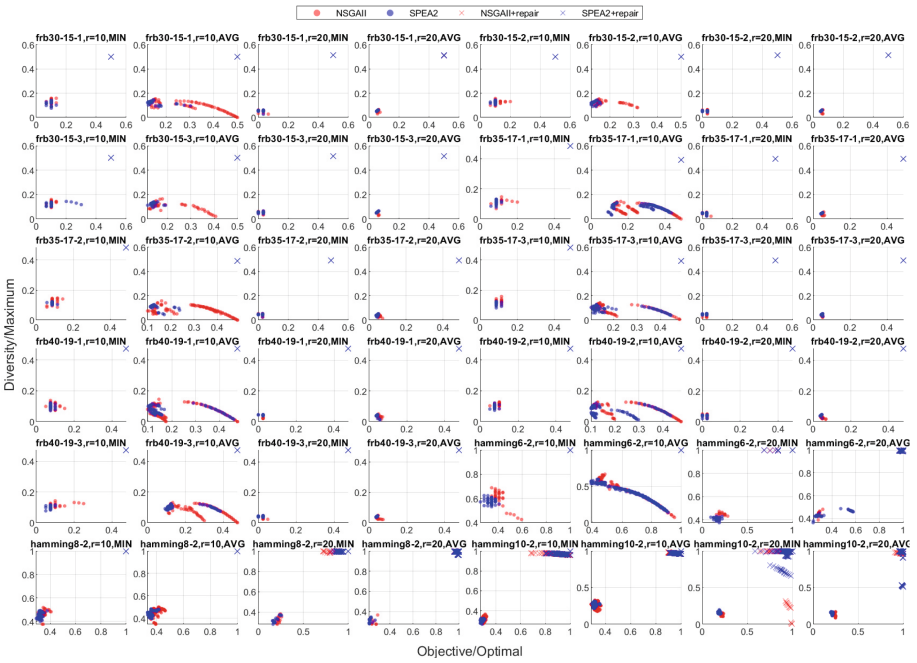



**Fig. 3.** Unions of final populations across all runs on Max Vertex Cover instances. "MIN" indicates the minimum objective setting being used for $f_1$ while "AVG" indicates average objective setting. For each run, dominated points are excluded.

The results are shown in Fig. 3 and Table 3. We see both algorithms with the repair heuristic consistently converge at a single non-dominated individual. In BHOSLIB instances, this individual is mapped to approximately 50% of optimal objective and diversity upper bound, resulting in Hypervolumes at roughly 25% of the entire fitness space. In DIMACS instances, this individual is mapped to the best theoretical trade-off consisting of $(OPT, g(n, OPT, r))$, agreeing with our intuition. The small number of returned non-dominated individuals could be explained by the limited number of distinct fitness values a feasible individual can assume, combined with the positive correlation between maximizing the

**Table 3.** Medians of indicator scores across runs on Max Vertex Cover instances. Medians of IGD+*, HV* and numbers of non-dominated points are 0, 1, 1, respectively, for both algorithm in all instances, with no statistically significant differences.

| | Inst. | $r$ | NSGA-II | | SPEA2 | |
|---|---|---|---|---|---|---|
| | | | IGD+ | HV | IGD+ | HV |
| Min objective | frb30-15-1 | 10 | 0.70711 | 0.25 | 0.70711 | 0.25 |
| | | 20 | 0.69762 | 0.25676 | 0.69762 | 0.25676 |
| | frb30-15-2 | 10 | 0.70711 | 0.25 | 0.70711 | 0.25 |
| | | 20 | 0.69762 | 0.25676 | 0.69762 | 0.25676 |
| | frb30-15-3 | 10 | 0.70711 | 0.25 | 0.70711 | 0.25 |
| | | 20 | 0.69762 | 0.25676 | 0.69762 | 0.25676 |
| | frb35-17-1 | 10 | 0.72731 | 0.23592 | 0.72731 | 0.23592 |
| | | 20 | 0.72182 | 0.2397 | 0.72182 | 0.2397 |
| | frb35-17-2 | 10 | 0.72731 | 0.23592 | 0.72731 | 0.23592 |
| | | 20 | 0.72182 | 0.2397 | 0.72182 | 0.2397 |
| | frb35-17-3 | 10 | 0.72731 | 0.23592 | 0.72731 | 0.23592 |
| | | 20 | 0.72182 | 0.2397 | 0.72182 | 0.2397 |
| | frb40-19-1 | 10 | 0.74246 | 0.22562 | 0.74246 | 0.22562 |
| | | 20 | 0.74069 | 0.22682 | 0.74069 | 0.22682 |
| | frb40-19-2 | 10 | 0.74246 | 0.22562 | 0.74246 | 0.22562 |
| | | 20 | 0.74069 | 0.22682 | 0.74069 | 0.22682 |
| | frb40-19-3 | 10 | 0.74246 | 0.22562 | 0.74246 | 0.22562 |
| | | 20 | 0.74069 | 0.22682 | 0.74069 | 0.22682 |
| | hamming6-2 | 10 | 0 | 1 | 0 | 1 |
| | | 20 | 0 | 1 | 0 | 1 |
| | hamming8-2 | 10 | 0 | 1 | 0 | 1 |
| | | 20 | 0 | 1 | 0 | 1 |
| | hamming10-2 | 10 | 0 | 1 | 0 | 1 |
| | | 20 | 0 | 1 | 0 | 1 |
| Average objective | frb30-15-1 | 10 | 0.70711 | 0.25 | 0.70711 | 0.25 |
| | | 20 | 0.69762 | 0.25676 | 0.69762 | 0.25676 |
| | frb30-15-2 | 10 | 0.70711 | 0.25 | 0.70711 | 0.25 |
| | | 20 | 0.69762 | 0.25676 | 0.69762 | 0.25676 |
| | frb30-15-3 | 10 | 0.70711 | 0.25 | 0.70711 | 0.25 |
| | | 20 | 0.69762 | 0.25676 | 0.69762 | 0.25676 |
| | frb35-17-1 | 10 | 0.72731 | 0.23592 | 0.72731 | 0.23592 |
| | | 20 | 0.72182 | 0.2397 | 0.72182 | 0.2397 |
| | frb35-17-2 | 10 | 0.72731 | 0.23592 | 0.72731 | 0.23592 |
| | | 20 | 0.72182 | 0.2397 | 0.72182 | 0.2397 |
| | frb35-17-3 | 10 | 0.72731 | 0.23592 | 0.72731 | 0.23592 |
| | | 20 | 0.72182 | 0.2397 | 0.72182 | 0.2397 |
| | frb40-19-1 | 10 | 0.74246 | 0.22562 | 0.74246 | 0.22562 |
| | | 20 | 0.74069 | 0.22682 | 0.74069 | 0.22682 |
| | frb40-19-2 | 10 | 0.74246 | 0.22562 | 0.74246 | 0.22562 |
| | | 20 | 0.74069 | 0.22682 | 0.74069 | 0.22682 |
| | frb40-19-3 | 10 | 0.74246 | 0.22562 | 0.74246 | 0.22562 |
| | | 20 | 0.74069 | 0.22682 | 0.74069 | 0.22682 |
| | hamming6-2 | 10 | 0 | 1 | 0 | 1 |
| | | 20 | 0 | 1 | 0 | 1 |
| | hamming8-2 | 10 | 0 | 1 | 0 | 1 |
| | | 20 | 0 | 1 | 0 | 1 |
| | hamming10-2 | 10 | 0 | 1 | 0 | 1 |
| | | 20 | 0 | 1 | 0 | 1 |

two fitnesses. The latter explains the observation that whenever multiple non-dominated individuals are returned, they are all dominated by the one individual the searches seem to converge at in the same instance.

We see NSGA-II and SPEA2 perform similarly under the same configuration across instances. Their achieved indicators are identical in most instances, and no statistically significant difference is detected in the rest. Furthermore, the repair heuristic produces clear improvements over non-repair variants, with roughly 35% run-time overhead. The non-repair variants also fail to produce comparable trade-offs within 150% the evaluation budgets.

We observe similar output individuals across instances within the same class (e.g., prefixed by "frb30-15"), resulting in identical median indicator scores. This indicates the similarity in the objective landscapes induced by these graphs. In addition, the outputs are similar between min objective setting and average objective setting. The choice of $f_1$ does not seem to influence the behaviors of the algorithms on these instances.

## 5 Conclusions

In this work, we study a bi-objective optimization formulation of the diverse solutions problem, where different trade-offs between solutions objective quality and diversity are evolved. This formulation requires that the output be a collection of solution sets, in exchange for eliminating the need to set quality or diversity criteria. We present an implementation scheme that treats a set of solutions as an individual, and handles the inherent symmetry in diversity measures. We realize the scheme in NSGA-II and SPEA2, and test the methods on various maximum cut, maximum coverage and minimum vertex cover instances. The results reveal insights on the optimization instances to which diverse solutions are computed, and confirm that the bi-objective optimization paradigm can be used to address the diverse solutions problem.

We remark that moving from solution spaces to populations spaces introduces extra complexity to the search processes in a way that seems to frustrate black-box approaches. It appears that this additional difficulty can be overcome by problem-specific heuristics, as observed in our investigation with min vertex cover. We speculate that state-of-the-art for this problem will involve memetic algorithms and hybrid approaches.

**Acknowledgments.** This work has been supported by the Australian Research Council (ARC) through grants DP190103894 and FT200100536.

## References


1. Alexander, B., Kortman, J., Neumann, A.: Evolution of artistic image variants through feature based diversity optimisation. In: GECCO, pp. 171–178. ACM, New York (2017). https://doi.org/10.1145/3071178.3071342
2. Arrighi, E., Fernau, H., Lokshtanov, D., de Oliveira Oliveira, M., Wolf, P.: Diversity in kemeny rank aggregation: A parameterized approach. In: IJCAI, pp. 10–16. International Joint Conferences on Artificial Intelligence Organization (Aug 2021). https://doi.org/10.24963/ijcai.2021/2
3. Audet, C., Bigeon, J., Cartier, D., Digabel, S.L., Salomon, L.: Performance indicators in multiobjective optimization. Eur. J. Oper. Res. **292**(2), 397–422 (2021). https://doi.org/10.1016/j.ejor.2020.11.016
4. Baste, J., et al.: Diversity of solutions: an exploration through the lens of fixed-parameter tractability theory. Artif. Intell. **303**, 103644 (2022). https://doi.org/10.1016/j.artint.2021.103644
5. Bossek, J., Kerschke, P., Neumann, A., Wagner, M., Neumann, F., Trautmann, H.: Evolving diverse TSP instances by means of novel and creative mutation operators. In: FOGA 2019. pp. 58–71. ACM Press, New York (2019).https://doi.org/10.1145/3299904.3340307
6. Bossek, J., Neumann, A., Neumann, F.: Breeding diverse packings for the knapsack problem by means of diversity-tailored evolutionary algorithms. In: GECCO, pp. 556–564. ACM, New York (Jun 2021).https://doi.org/10.1145/3449639.3459364
7. Bossek, J., Neumann, F.: Evolutionary diversity optimization and the minimum spanning tree problem. In: GECCO, pp. 198–206. ACM, New York (Jun 2021).https://doi.org/10.1145/3449639.3459363
8. Branke, J.: Creating robust solutions by means of evolutionary algorithms, pp. 119-128. Springer, Berlin (1998). https://doi.org/10.1007/bfb0056855
9. Danna, E., Fenelon, M., Gu, Z., Wunderling, R.: Generating multiple solutions for mixed integer programming problems. In: Fischetti, M., Williamson, D.P. (eds.) IPCO 2007. LNCS, vol. 4513, pp. 280–294. Springer, Heidelberg (2007). https://doi.org/10.1007/978-3-540-72792-7_22
10. Deb, K., Pratap, A., Agarwal, S., Meyarivan, T.: A fast and elitist multiobjective genetic algorithm: NSGA-II. IEEE Trans. Evol. Comput. **6**(2), 182–197 (2002). https://doi.org/10.1109/4235.996017
11. Do, A., Guo, M., Neumann, A., Neumann, F.: Analysis of evolutionary diversity optimization for permutation problems. ACM Trans. Evolutionary Learn. Optimizat. **2**(3), 1–27 (2022). https://doi.org/10.1145/3561974
12. Durillo, J.J., Nebro, A.J.: jMetal: a java framework for multi-objective optimization. Adv. Eng. Softw. **42**(10), 760–771 (2011). https://doi.org/10.1016/j.advengsoft.2011.05.014
13. Eiter, T., Erdem, E., Erdoğan, H., Fink, M.: Finding similar or diverse solutions in answer set programming. In: Hill, P.M., Warren, D.S. (eds.) ICLP 2009. LNCS, vol. 5649, pp. 342–356. Springer, Heidelberg (2009). https://doi.org/10.1007/978-3-642-02846-5_29
14. Erkut, E.: The discrete p-dispersion problem. Eur. J. Oper. Res. **46**(1), 48–60 (1990). https://doi.org/10.1016/0377-2217(90)90297-o
15. Fomin, F.V., Golovach, P.A., Jaffke, L., Philip, G., Sagunov, D.: Diverse pairs of matchings. In: ISAAC pp. 26:1–26:12. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, Dagstuhl, Germany (2020).https://doi.org/10.4230/LIPICS.ISAAC.2020.26

16. Fomin, F.V., Golovach, P.A., Panolan, F., Philip, G., Saurabh, S.: Diverse collections in matroids and graphs. In: STACS, pp. 31:1–31:14. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, Dagstuhl, Germany (2021).https://doi.org/10.4230/LIPICS.STACS.2021.31
17. Gao, W., Nallaperuma, S., Neumann, F.: Feature-based diversity optimization for problem instance classification. Evol. Comput. **29**(1), 107–128 (2021). https://doi.org/10.1162/evco_a_00274
18. Glover, F., LøKketangen, A., Woodruff, D.L.: Scatter search to generate diverse MIP solutions. In: Operations Research/Computer Science Interfaces Series, pp. 299–317. Springer US, Boston (2000).https://doi.org/10.1007/978-1-4615-4567-5_17
19. Hanaka, T., Kiyomi, M., Kobayashi, Y., Kobayashi, Y., Kurita, K., Otachi, Y.: A framework to design approximation algorithms for finding diverse solutions in combinatorial problems. In: AAAI 2022, pp. 3758–3766. AAAI Press (2022), https://ojs.aaai.org/index.php/AAAI/article/view/20290
20. Hanaka, T., Kobayashi, Y., Kurita, K., Lee, S.W., Otachi, Y.: Computing diverse shortest paths efficiently: A theoretical and experimental study. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 36(4), pp. 3758–3766 (Jun 2022).https://doi.org/10.1609/aaai.v36i4.20290
21. Hanaka, T., Kobayashi, Y., Kurita, K., Otachi, Y.: Finding diverse trees, paths, and more. In: AAAI 2021, vol. 35, pp. 3778–3786. AAAI Press (May 2021). https://doi.org/10.1609/aaai.v35i5.16495
22. Hao, F., Pei, Z., Yang, L.T.: Diversified top-k maximal clique detection in social internet of things. Future Generat. Comput. Syst. **107**, 408–417 (Jun 2020).https://doi.org/10.1016/j.future.2020.02.023
23. Hebrard, E., Hnich, B., O'Sullivan, B., Walsh, T.: Finding diverse and similar solutions in constraint programming. In: AAAI, pp. 372–377. AAAI Press / The MIT Press (2005)
24. Ingmar, L., de la Banda, M.G., Stuckey, P.J., Tack, G.: Modelling diversity of solutions. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 34(02), pp. 1528–1535 (2020). https://doi.org/10.1609/aaai.v34i02.5512
25. Johnson, D.J., Trick, M.A.: Cliques, Coloring, and Satisfiability: Second DIMACS Implementation Challenge, Workshop, October 11–13, 1993. American Mathematical Society, USA (1996)
26. Kerschke, P., Hoos, H.H., Neumann, F., Trautmann, H.: Automated algorithm selection: survey and perspectives. Evol. Comput. **27**(1), 3–45 (2019). https://doi.org/10.1162/evco_a_00242
27. Nadel, A.: Generating diverse solutions in SAT. In: Sakallah, K.A., Simon, L. (eds.) SAT 2011. LNCS, vol. 6695, pp. 287–301. Springer, Heidelberg (2011). https://doi.org/10.1007/978-3-642-21581-0_23
28. Neumann, A., Bossek, J., Neumann, F.: Diversifying greedy sampling and evolutionary diversity optimisation for constrained monotone submodular functions. In: GECCO, pp. 261–269. ACM, New York (Jun 2021). https://doi.org/10.1145/3449639.3459385
29. Neumann, A., Gao, W., Doerr, C., Neumann, F., Wagner, M.: Discrepancy-based evolutionary diversity optimization. In: GECCO, pp. 991–998. ACM, New York (Jul 2018). https://doi.org/10.1145/3205455.3205532
30. Neumann, A., Gao, W., Wagner, M., Neumann, F.: Evolutionary diversity optimization using multi-objective indicators. In: GECCO, pp. 837–845. ACM, New York (Jul 2019). https://doi.org/10.1145/3321707.3321796

31. Nikfarjam, A., Bossek, J., Neumann, A., Neumann, F.: Computing diverse sets of high quality TSP tours by EAX-based evolutionary diversity optimisation. In: FOGA 2021. ACM, New York (Sep 2021).https://doi.org/10.1145/3450218.3477310
32. Nikfarjam, A., Bossek, J., Neumann, A., Neumann, F.: Entropy-based evolutionary diversity optimisation for the traveling salesperson problem. In: GECCO, pp. 600–608. ACM, New York (Jun 2021).https://doi.org/10.1145/3449639.3459384
33. Parmee, I.C., Bonham, C.R.: Improving Cluster Oriented Genetic Algorithms for High-performance Region Identification, pp. 189-202. Springer, London (2002). https://doi.org/10.1007/978-1-4471-0675-3_16
34. Pelikan, M., Kalapala, R., Hartmann, A.K.: Hybrid evolutionary algorithms on minimum vertex cover for random graphs. In: GECCO, pp. 547–554. ACM, New York (Jul 2007). https://doi.org/10.1145/1276958.1277073
35. Petit, T., Trapp, A.C.: Finding diverse solutions of high quality to constraint optimization problems. In: IJCAI 2015, pp. 260—266. AAAI Press (2015)
36. Pierrot, T., Richard, G., Beguir, K., Cully, A.: Multi-objective quality diversity optimization. In: Proceedings of the Genetic and Evolutionary Computation Conference, GECCO 2022, ACM (Jul 2022). https://doi.org/10.1145/3512290.3528823
37. Ronald, S.: Finding multiple solutions with an evolutionary algorithm. In: Proceedings of 1995 IEEE International Conference on Evolutionary Computation, vol. 2, pp. 641–646. IEEE (1995). https://doi.org/10.1109/icec.1995.487459
38. Rossi, R.A., Ahmed, N.K.: The network data repository with interactive graph analytics and visualization. In: AAAI (2015). https://networkrepository.com
39. Ruffini, M., Vucinic, J., de Givry, S., Katsirelos, G., Barbe, S., Schiex, T.: Guaranteed diversity & quality for the weighted CSP. In: ICTAI, pp. 18–25. IEEE (Nov 2019). https://doi.org/10.1109/ictai.2019.00012
40. Schittekat, P., Sörensen, K.: Supporting 3pl decisions in the automotive industry by generating diverse solutions to a large-scale location-routing problem. Oper. Res. **57**(5), 1058–1067 (2009)
41. Trapp, A.C., Konrad, R.A.: Finding diverse optima and near-optima to binary integer programs. IIE Trans. **47**(11), 1300–1312 (2015). https://doi.org/10.1080/0740817x.2015.1019161
42. Tsutsui, S., Ghosh, A.: Genetic algorithms with a robust solution searching scheme. IEEE Trans. Evol. Comput. **1**(3), 201–208 (1997). https://doi.org/10.1109/4235.661550
43. Ulrich, T., Bader, J., Zitzler, E.: Integrating decision space diversity into hypervolume-based multiobjective search. In: GECCO, pp. 455–462. ACM Press, New York (2010).https://doi.org/10.1145/1830483.1830569
44. Ulrich, T., Thiele, L.: Maximizing population diversity in single-objective optimization. In: GECCO, pp. 641–648. ACM Press, New York (2011).https://doi.org/10.1145/2001576.2001665
45. Wang, D., Kuo, Y.S.: A study on two geometric location problems. Inf. Process. Lett. **28**(6), 281–286 (1988). https://doi.org/10.1016/0020-0190(88)90174-3
46. Wang, J., Cheng, J., Fu, A.W.C.: Redundancy-aware maximal cliques. In: Proceedings of the 19th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD 2013, pp. 122–130. ACM (Aug 2013). https://doi.org/10.1145/2487575.2487689

47. Wang, R.J., Xue, K., Shang, H., Qian, C., Fu, H., Fu, Q.: Multi-objective optimization-based selection for quality-diversity by non-surrounded-dominated sorting. In: Proceedings of the Thirty-Second International Joint Conference on Artificial Intelligence. IJCAI-2023, International Joint Conferences on Artificial Intelligence Organization (Aug 2023). https://doi.org/10.24963/ijcai.2023/482
48. Ye, Y.: Gset max-cut problem set (2003). https://web.stanford.edu/~yyye/yyye/Gset/
49. Yuan, L., Qin, L., Lin, X., Chang, L., Zhang, W.: Diversified top-k clique search. VLDB J. **25**(2), 171–196 (2015). https://doi.org/10.1007/s00778-015-0408-z
50. Zechman, E.M., Ranjithan, S.R.: An evolutionary algorithm to generate alternatives (EAGA) for engineering optimization problems. Eng. Optim. **36**(5), 539–553 (2004). https://doi.org/10.1080/03052150410001704863
51. Zechman, E.M., Ranjithan, S.R.: Generating alternatives using evolutionary algorithms for water resources and environmental management problems. J. Water Resour. Plan. Manag. **133**(2), 156–165 (2007). https://doi.org/10.1061/(asce)0733-9496(2007)133:2(156)
52. Zitzler, E., Laumanns, M., Thiele, L.: Spea 2: Improving the strength pareto evolutionary algorithm. Tech. Rep. (2001). https://doi.org/10.3929/ETHZ-A-004284029

# Bayesian Forward-Inverse Transfer for Multiobjective Optimization

Tingyang Wei[1], Jiao Liu[1(✉)], Abhishek Gupta[2], Puay Siew Tan[3], and Yew-Soon Ong[1,4]

[1] College of Computing and Data Science, Nanyang Technological University, Singapore, Singapore
TINGYANG001@e.ntu.edu.sg, {jiao.liu,asysong}@ntu.edu.sg
[2] School of Mechanical Sciences, Indian Institute of Technology, Goa, India
abhishekgupta@iitgoa.ac.in
[3] Singapore Institute of Manufacturing Technology (SIMTech), Agency for Science, Technology and Research, Singapore, Singapore
pstan@simtech.a-star.edu.sg
[4] Centre for Frontier AI Research (CFAR), Agency for Science, Technology and Research, Singapore, Singapore

**Abstract.** We present an evolutionary optimizer incorporating knowledge transfer through forward and inverse surrogate models for solving multiobjective problems, within a stringent computational budget. Forward knowledge transfer is employed to fully exploit solution-evaluation datasets from related tasks by building Bayesian forward multitask surrogate models that map points from decision to objective space. Inverse knowledge transfer via Bayesian inverse multitask models makes possible the creation of high-quality solution populations in decision space by mapping back from preferred points in objective space. In contrast to prior work, the proposed method can improve the overall convergence performance to multiple Pareto sets by fully exploiting information available for diverse multiobjective problems. Empirical studies conducted on benchmark and real-world multitask multiobjective optimization problems demonstrate the faster convergence rate and enhanced inverse modeling accuracy of our algorithm compared to state-of-the-art algorithms.

**Keywords:** Evolutionary algorithms · Bayesian optimization · multiobjective optimization · inverse models

## 1 Introduction

In various real-world scenarios, decision-makers (DMs) confront the challenges of simultaneously considering multiple conflicting criteria when choosing a specific optimal solution [27,33,34]. Such problems are commonly formulated as multiobjective optimization problems (MOPs) [4]. The defining characteristic of MOPs is the absence of a single solution that universally outperforms others across all criteria. Thus, the primary aim in addressing an MOP is to identify


M. Affenzeller et al. (Eds.): PPSN 2024, LNCS 15151, pp. 135–152, 2024.
https://doi.org/10.1007/978-3-031-70085-9_9

Pareto optimal solutions, mapping to the Pareto front (PF) of optimal performance trade-offs in the objective space. Evolutionary algorithms (EAs), with the inherent population-based nature, are well suited for MOPs [9], through evolving and maintaining a solution set iteratively. The obtained solution set can approximate the PF with well-distributed solutions, enabling DMs to assess performance trade-offs and subsequently select the preferred solution(s) *a posteriori* [29,39].

While EAs have effectively addressed MOPs, a notable drawback is their data-hungry nature, necessitating massive evaluations to converge upon a set of nondominated solutions. This characteristic renders EAs unsuitable to directly tackle expensive MOPs, especially those demanding time-consuming computer simulations or intricate real-world experiments. To overcome this challenge, leveraging inexpensive surrogate models can aid algorithmic evolution, reducing the budget for expensive evaluations. Notable algorithms in this domain include ParEGO [28], MOEA/D-EGO [48], EHVI [15], and SMS-EGO [37]. However, these algorithms still require a sufficient number of evaluated samples to construct informative surrogate models, thus necessitating a generous function evaluation budget. This necessity has been termed as the *cold start* issue.

Integrating knowledge transfer capabilities into algorithms has emerged as a promising strategy to address cold starts. By allowing algorithms to learn from optimization experiences across related tasks, they often exhibit faster convergence [10,21]. Notably, transfer evolutionary optimization and evolutionary multitasking have attracted significant attention as effective approaches in this regard [2,17,21,45]. These methodologies recognize that real-world problems rarely exist in isolation, allowing knowledge from various tasks to be adaptively reused, thereby enhancing convergence. The efficacy of these algorithms is heavily influenced by the strategy employed for knowledge transfer. Over the past decades, several knowledge transfer techniques have been proposed and incorporated into the EA to enhance performance. This paper focuses on a recent approach termed *inverse transfer*, which has shown promise in addressing expensive MOPs [32]. This technique leverages Bayesian inverse models that map points on the PF back to their corresponding nondominated solutions in the decision space to achieve knowledge transfer. For instance, Liu et al. [32] introduced invTrEMO, a novel transfer optimization algorithm showcasing the capability to expedite convergence by harnessing implicit knowledge inherent in the source data. Moreover, invTrEMO generates accurate inverse models as a byproduct, facilitating the generation of nondominated solutions tailored to user preferences on demand. However, it is crucial to note that Bayesian inverse models typically assume one-to-one mappings from the PF to Pareto optimal solutions, grounded in the Karush-Kuhn-Tucker conditions [7]. This assumption constrains Bayesian inverse models to be trained solely based on nondominated samples, limiting the utilization of the information embedded in dominated samples across different tasks, and potentially impeding the overall efficacy of the approach.

To overcome the limitations of invTrEMO, we introduce a novel extension, *forward-inverse transfer evolutionary multiobjective optimizer* (F-invTrEMO). This enhanced method integrates both forward and inverse transfer techniques

to maximize information utilization. By combining these two approaches, F-invTrEMO harnesses the strengths of both forward and inverse transfer, enabling more comprehensive knowledge transfer across multiple optimization tasks. The forward transfer component leverages all evaluated samples to capture correlations in function landscapes across all task domains, enhancing the accuracy of Bayesian forward models to effectively guide the optimization process. Meanwhile, the inverse transfer aspect collaboratively utilizes nondominated samples across related tasks, enabling the Bayesian inverse models to map preference vectors from the objective space to the search distribution [24,40]. This distribution facilitates the generation of promising offspring solutions by harnessing knowledge from multiple tasks, thus guiding the evolutionary process. This forward-inverse transfer mechanism can hopefully upgrade the ability to exploit inter-task information across related MOPs [25] including composites manufacturing [23], path planning [1], model compression [8] and hyperspectral unmixing [46].

In this paper, we primarily focus on addressing the more challenging multitask multiobjective optimization problems with F-invTrEMO, rather than the sequential transfer optimization problems [22]. This broader focus aims to further validate the effectiveness of the proposed method. The main contributions of this paper can be summarized as follows:

- We present a novel algorithm named F-invTrEMO. A distinctive feature of F-invTrEMO is its incorporation of both forward and inverse transfer techniques to steer the optimization process. This approach allows F-invTrEMO to capitalize on information from both dominated and nondominated evaluated samples across diverse tasks, thereby maximizing the utilization of available data for improved optimization outcomes.
- We demonstrate that integrating both forward and inverse transfer mechanisms not only accelerates convergence but also yields more precise Bayesian inverse models for approximating the Pareto fronts of individual tasks.
- We validate the superiority of our method across a spectrum of multitask optimization problems, encompassing benchmark problems, a set of engineering optimization problems, and hyperparameter optimization problems.

The remainder of this paper is organized as follows. Section 2 introduces the basic concepts and the related work. Section 3 introduces the proposed F-invTrEMO framework. The effectiveness of the proposed framework is justified by experimental analysis in Sect. 4, and Sect. 5 concludes the paper.

## 2 Preliminaries

### 2.1 Multitask Multiobjective Optimization

Considering the synergies inherent in MOPs, multitask multiobjective optimization concurrently solves a set of MOPs by leveraging information across these problem domains [16,22]. Let $f_{k,i}(\cdot)$, where $k \in \{1, \ldots, K\}$ and $i \in \{1, \ldots, m\}$,

be the $i$th objective on the $k$th optimization task, without loss of generality, the multitask MOP is defined as follows:

$$\begin{aligned} &\forall T_k, k \in \{1, \ldots, K\}, \\ &\min : \mathbf{F}_k(\mathbf{x}_k) = (f_{k,1}(\mathbf{x}_k), \ldots, f_{k,m}(\mathbf{x}_k)), \quad \text{s.t. } \mathbf{x}_k \in \Omega_k \subset \mathbb{R}^d \end{aligned} \tag{1}$$

where $T_k$ denotes the $k$th task setting, $\mathbf{F}_k(\cdot)$ represents the objective function vector corresponding to the $k$th task setting, $\mathbf{x}_k = (x_{k,1}, \ldots, x_{k,d})$ denotes the solution corresponding to the $k$th task setting, and $\Omega_k$ signifies the decision space corresponding to the $k$th task setting. In this paper, we assume that each objective function is expensive to evaluate, imposing a constraint on the evaluation budget allocated for each task. The key concepts [4] associated with the formulation in (1) are explained as follows:

- (**Pareto Dominance**) Solution $\mathbf{x}_k^{(a)}$ is said to Pareto dominate another solution $\mathbf{x}_k^{(b)}$ on the $k$th task setting, if $\forall i \in \{1, 2, \ldots, m\}$, $f_{k,i}(\mathbf{x}_k^{(a)}) \leq f_{k,i}(\mathbf{x}_k^{(b)})$ and $\exists i' \in \{1, 2, \ldots, m\}$ such that $f_{k,i'}(\mathbf{x}_k^{(a)}) < f_{k,i'}(\mathbf{x}_k^{(b)})$.
- (**Pareto Optimal Solutions**) Solution $\mathbf{x}_k^*$ is said to be Pareto optimal on the $k$th task setting if there are no other candidate solutions that can dominate $\mathbf{x}_k^*$.
- (**Pareto Set**) Pareto set (PS) consists of all the Pareto optimal solutions.
- (**Pareto Front**) The image of the Pareto set in the objective space is referred to as the Pareto front (PF).

The result of (1) can be represented as a set of Pareto optimal solution sets:

$$PS_k = \{\mathbf{x}_k^{(1)*}, \mathbf{x}_k^{(2)*}, \mathbf{x}_k^{(3)*}, \ldots\}, \quad \mathcal{S} = \bigcup_{k=1}^{K} PS_k, \tag{2}$$

where $\mathbf{x}_k^{(\cdot)*}$ is a Pareto optimal solution of the $k$th task, $PS_k$ is a Pareto optimal solution set of the $k$th task setting, and $\mathcal{S}$ is the optimal set of solution sets.

### 2.2 Decomposition-Based Multiobjective Optimization

The fundamental concept behind decomposition-based multiobjective optimization is to decompose a MOP into a series of single-objective subproblems through the utilization of a specific scalarization function [47]. By solving these subproblems individually, a finite set of Pareto optimal solutions can be obtained. In this paper, we concentrate on augmented Tchebycheff scalarization [28,30], selected for its simplicity and suitability for non-convex Pareto fronts. This technique combines objective functions using the following expression, which remains consistent across tasks:

$$f^{tch}(\mathbf{x}|\mathbf{w}) = \max_{1 \leq i \leq m} \{w_i \cdot (f_i(\mathbf{x}) - (z_i^* - \epsilon))\} + \rho \sum_{i=1}^{M} w_i f_i(\mathbf{x}), \tag{3}$$

where $f_i(\mathbf{x})$ is the $i$th objective of a specific MOP, $\mathbf{w}$ is known as the *preference vector* sampled from a $(m-1)$ dimensional simplex (i.e., $\mathbf{w} \in \mathcal{W} = \{(w_1, w_2, ..., w_m)^T | \sum_{i=1}^{m} w_i = 1, w_i \in [0,1]\}$), $z_i^*$ is the ideal objective function value (i.e., minimal possible value) for the $i$th objective, the coupled term $(z_i^* - \epsilon)$ is the utopia objective value by maintaining $\epsilon$ as a small positive value for the $i$th objective, and $\rho$ is a sufficiently small scalar weight.

### 2.3 Gaussian Process

To address expensive objective functions, Gaussian Processes (GPs) [38] are commonly employed as the surrogate to facilitate a data-efficient problem-solving process. This method has been widely applied in various multiobjective optimization techniques, including ParEGO [28], MOEA/D-EGO [48], EHVI [15], and SMS-EGO [37]. In this paper, we also opt for GP as the surrogate model, where we assume $f \sim \mathcal{GP}(\mu, \kappa(\mathbf{x}, \mathbf{x}'))$, where $\mu$ represents the mean function with a zero prior, and $\kappa$ is a kernel function that encapsulates the correlation between input data points. Given sampled data $\mathcal{D} = \{(\mathbf{x}^{(i)}, y^{(i)})\}_{i=1}^{N}$ with noise $\epsilon^{(i)} \sim \mathcal{N}(0, \sigma_\epsilon^2)$, the predicted posterior distribution of the GP model, $\mathcal{N}(\mu(\mathbf{x}^{(*)}), \sigma^2(\mathbf{x}^{(*)}))$, for a query $\mathbf{x}^{(*)}$ can be computed as follows:

$$\mu(\mathbf{x}^{(*)}) = \mathbf{k}_*^\intercal (\boldsymbol{\Sigma} + \sigma_\epsilon^2 \mathbf{I}_N)^{-1} \mathbf{y}, \tag{4}$$

$$\sigma(\mathbf{x}^{(*)}) = \kappa_{**} - \mathbf{k}_*^\intercal (\boldsymbol{\Sigma} + \sigma_\epsilon^2 \mathbf{I}_N)^{-1} \mathbf{k}_*. \tag{5}$$

where $\kappa_{**} = \kappa(\mathbf{x}^{(*)}, \mathbf{x}^{(*)})$, $\mathbf{k}_*$ is the kernel vector between $\mathbf{x}^{(*)}$ and the data in $\mathcal{D}$, $\boldsymbol{\Sigma}$ is an $N \times N$ matrix with elements $\Sigma_{p,q} = \kappa(\mathbf{x}^{(p)}, \mathbf{x}^{(q)})$, $p, q \in \{1, \ldots, N\}$, $\mathbf{I}_N$ is a $N \times N$ identity matrix, and $\mathbf{y}$ is the vector of the noisy observations.

### 2.4 Inverse Modeling in Multiobjective Optimization

Inverse modeling has emerged as a powerful tool in the domain of multiobjective optimization, offering innovative strategies for navigating optimization and enabling DMs to generate preferred trade-off solutions on demand. Under the premise that the Karush-Kuhn-Tucker conditions hold, PS and PF are both $(m-1)$-dimensional piecewise manifolds under mild conditions [14], enabling inverse models to facilitate the mapping of solutions from the objective space back into the decision space. It is noteworthy that the input for inverse models can encompass not only objective function values [7,24] but also preference vectors [20,30,31]. In this paper, we specifically explore the mapping from preference vectors to decision variables. This approach allows DMs to articulate their preferences more easily without delving deeply into the intricacies of the PF topology [40]. GP models are utilized to map a preference vector $\mathbf{w}$ to the $j$th decision variable as $\Psi_j^{-1}(\mathbf{w})$, where $\Psi_j^{-1} \sim \mathcal{GP}(\mu_j, \kappa_j(\mathbf{w}, \mathbf{w}'))$. By amalgamating all $d$ mappings, the inverse prediction $[\Psi_1^{-1}(\mathbf{w}), \Psi_2^{-1}(\mathbf{w}), \ldots, \Psi_d^{-1}(\mathbf{w})]$ is obtained. To train the inverse GP models, a dataset $\mathcal{D}^{inv} = \{(\mathbf{w}^{(l)}, \mathbf{x}^{(l)})\}_{l=1}^{N^{nd}}$ should be provided first, where $\mathbf{w}^{(l)}$ is the preference vector corresponding to the

nondominated solution $\mathbf{x}^{(l)}$. It's worth noting that the multiobjective optimizer used may not be decomposition-based, and therefore, well-defined subproblems with associated preference vectors may not be specified beforehand. As a general strategy, the following fomulation can be employed to derive $\mathbf{w}^{(l)}$ for $\mathbf{x}^{(l)}$ based on objective function values:

$$w_i^{(l)} = \frac{c_i^{(l)}}{\sum_{i=1}^{M} c_i^{(l)}}, i \in \{1, \ldots, M\}, \tag{6}$$

where $c_i^{(l)} = \frac{\sum_{v=1}^{m}(f_v(\mathbf{x}^{(l)}) - (z_v^* - \epsilon))}{(f_i(\mathbf{x}^{(l)}) - (z_i^* - \epsilon))}$, ensuring the preference vector $\mathbf{w}^{(l)}$ can correspond to the aggregated solution in the objective space.

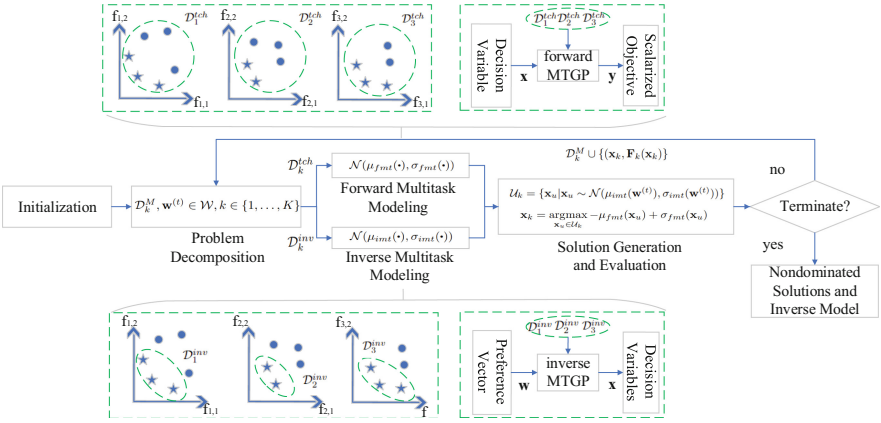



**Fig. 1.** Workflow of the proposed F-invTrEMO. Inverse modeling considers only nondominated solutions while forward modeling considers all the solutions.

## 3 Forward-Inverse Transfer for Multitask Multiobjective Optimization

This section elucidates the methodology of the proposed F-invTrEMO. We commence by delineating the workflow of the proposed method and demonstrate how to incorporate forward and inverse transfer into the optimization process. Subsequently, we introduce multitask Gaussian Process (MTGP) models [3] including forward MTGP and inverse MTGP that enable forward and inverse transfer.

### 3.1 General Framework of F-InvTrEMO

The general workflow of the F-invTrEMO can be detailed in Fig. 1 and **Algorithm 1**, and the steps are explained as follows:

- **Initialization**: In this step, $N_{init}$ solutions are generated and evaluated for each task, thus forming the initial datasets $\{\mathcal{D}_1^M, \ldots, \mathcal{D}_K^M\}$, where $\mathcal{D}_k^M = \{(\mathbf{x}_k^{(l)}, \mathbf{F}_k(\mathbf{x}_k^{(l)}))\}_{l=1}^{N_k}$ and $N_k = N_{init}$.
- **Problem Decomposition**: In each iteration, we sample a randomly selected preference vector $\mathbf{w}^{(t)} \in \mathcal{W}$ and scalarize the MOPs into single-objective optimization problems based on (3). The resultant dataset for task $k$ can be denoted as $\mathcal{D}_k^{tch} = \{\mathbf{X}_k, \mathbf{y}_k\} = \{(\mathbf{x}_k^{(l)}, y_k^{(l)})\}_{l=1}^{N_k}$, and $y_k^{(l)} = f_k^{tch}(\mathbf{x}_k^{(l)}) + \epsilon^{(l)}$ represents a noisy observation of the scalarized output corresponding to the $k$-th task, with $\epsilon^{(l)}$ being additive Gaussian noise with zero mean.
- **Forward Multitask Modeling**: In each iteration, a forward MTGP model is constructed to guide the optimization process for each task. Notably, in the context of F-invTrEMO, the forward model prediction for each task is designed to approximate the augmented Tchebycheff function as defined in (3) and the training set for the forward MTGP model is $\mathcal{D}_k^{tch}, k \in \{1, \ldots .K\}$. *It is important to note that the datasets for the forward MTGP models contain both dominated and nondominated samples, enabling the algorithm to fully utilize information from all the data.*
- **Inverse Multitask Modeling**: To construct inverse models, the dataset $\mathcal{D}_k^{inv}$ is generated for the $k$th task by only leveraging the nondominated samples within $\mathcal{D}_k^M$. To be specific, given multitask MOPs with $d$-dimensional decision variable for each task, $d$ datasets are included in dataset $\mathcal{D}_k^{inv}$ as $\mathcal{D}_{k,j}^{inv} = \{\mathbf{W}_k, \mathbf{X}_{k,j}\} = \{(\mathbf{w}_k^{(l)}, \mathbf{x}_{k,j}^{(l)})\}_{l=1}^{N_k^{nd}}$, $j \in \{1, \ldots, d\}$, containing pairs of nondominated solutions $\mathbf{x}_k^{(l)}$ and corresponding preference vectors $\mathbf{w}_k^{(l)}$ obtained according to (6). Subsequently, $d$ independent inverse MTGP models are trained based on $\{\mathcal{D}_1^{inv}, \ldots \mathcal{D}_K^{inv}\}$ to map the preference vectors to nondominated solutions in the decision space.
- **Evolutionary Solution Generation**: For each solving task, a candidate solution is generated, and the corresponding dataset is updated. For the $k$th task, given $\mathbf{w}^{(t)}$, the inverse MTGP provides a search distribution characterized by a Gaussian distribution. Subsequently, a set of offspring solutions $\mathcal{U}_k$ is produced by sampling according to this search distribution. We utilize the upper confidence bound (UCB) acquisition function to select the most promising solution $\mathbf{x}_k^{(t)}$ from $\mathcal{U}_k$ for the $k$th task. In the case of a minimization problem, the UCB function is formulated as $-\mu_{fmt}(\mathbf{x}_u) + \beta \cdot \sigma_{fmt}(\mathbf{x}_u)$, where $\beta$ is a predefined parameter balancing exploration and exploitation, $\mu_{fmt}(\mathbf{x}_u)$ and $\sigma_{fmt}(\mathbf{x}_u)$ are the predictive mean and standard deviation for a given query $\mathbf{x}_u$ by the forward MTGP. The selected $\mathbf{x}_k^{(t)}$ is then evaluated by the objective function of the $k$th task, denoted as $\mathbf{F}_k(\mathbf{x}_k^{(t)})$. Finally, the dataset $\mathcal{D}_k^M$ is updated by appending $\{(\mathbf{x}_k^{(t)}, \mathbf{F}_k(\mathbf{x}_k^{(t)}))\}$ to it.

Steps 5 to 20 in **Algorithm 1** are repeated until the termination condition is met. The algorithm then returns the inverse MTGP models and the nondominated solutions corresponding to each optimization task.

**Algorithm 1:** The workflow of F-invTrEMO

**Input**: Task size $K$, Initialization budgets $N_{init}$, Dimension size $d$, Sample size of offspring solutions $N_S$, Objective functions $\mathbf{F}_k$ for each task $k$, Predefined preference vector set $\mathcal{W}$.

**Output**: Pareto optimal solutions for each task, Inverse model for each dimension

**foreach** *task* $k$ **do**
  Initialize $N_{init}$ solutions for the $k$th task, thus forming dataset $\mathcal{D}_k^M = \{(\mathbf{x}_k^{(l)}, \mathbf{F}_k(\mathbf{x}_k^{(l)}))\}_{l=1}^{N_k=N_{init}}$
**end**
$t \leftarrow 0$
**while** $t < t_{max}$ **do**
  Sample a preference vector $\mathbf{w}^{(t)}$ randomly from $\mathcal{W}$
  **foreach** *task* $k$ **do**
    Scalarize $\mathcal{D}_k^M$ based on (3) and $\mathbf{w}^{(t)}$ to obtain dataset $\mathcal{D}_k^{tch} = \{(\mathbf{x}_k^{(l)}, y_k^{(l)})\}_{l=1}^{N_k}$
  **end**
  Build a forward MTGP model $\mathcal{N}(\mu_{fmt}(\mathbf{x}^{(*)}), \sigma_{fmt}^2(\mathbf{x}^{(*)}))$ using $\cup_{k=1}^{K}\mathcal{D}_k^{tch}$
  **foreach** *dimension* $j$ **do**
    Generate $k$ datasets $\mathcal{D}_{k,j}^{inv} = \{(\mathbf{w}_k^{(l)}, \mathbf{x}_{k,j}^{(l)})\}_{l=1}^{N_k^{nd}}, (k \in \{1, \ldots, K\})$ based on nondominated samples in $\mathcal{D}_k^M$ and (6)
    Build an inverse MTGP model $\mathcal{N}(\mu_{imt,j}(\mathbf{w}_{k^*}^{(*)}), \sigma_{imt,j}^2(\mathbf{w}_{k^*}^{(*)}))$ using $\{\mathcal{D}_{1,j}^{inv}, \ldots, \mathcal{D}_{K,j}^{inv}\}$ for the $j$-th decision variable
  **end**
  **foreach** *task* $k$ **do**
    Let $\mathbf{w}_k^{(*)} = \mathbf{w}^{(t)}$, then sampling a set $\mathcal{U}_k$ of $N_S$ offspring solutions based on the prediction provided by the inverse MTGP models
    Select solution $\mathbf{x}_k^{(t)} = \text{argmax}_{\mathbf{x}_u \in \mathcal{U}_k} -\mu_{fmt}(\mathbf{x}_u) + \sigma_{fmt}(\mathbf{x}_u)$
    $\mathcal{D}_k^M \leftarrow \mathcal{D}_k^M \cup \{(\mathbf{x}_k^{(t)}, \mathbf{F}_k(\mathbf{x}_k^{(t)}))\}$
  **end**
  $t \leftarrow t + 1$
**end**

### 3.2 Multitask Gaussian Process as Forward and Inverse Models

Within the F-invTrEMO framework, the incorporation of MTGP models in both forward and inverse modeling[1] aims to thoroughly exploit sampled data across multiple tasks, thus enhancing information utilization. The iterative refinement of both forward and inverse modeling mutually benefits each other, resulting in improved approximation of the PF during and after optimization.

Distinguishing itself from canonical GP models outlined in equations (4) and (5), the primary innovation of MTGP lies in the formulation of a multitask kernel function [3], as expressed below:

[1] The training process of the MTGP model can be found in [19].

$$\kappa_{mt}((\mathbf{x}_{input}, a), (\mathbf{x}'_{input}, b)) = \kappa_{tasks}(a, b) \cdot \kappa(\mathbf{x}_{input}, \mathbf{x}'_{input}) \tag{7}$$

Here, $\mathbf{x}_{input}$ and $\mathbf{x}'_{input}$ represent arbitrary input vectors, $\kappa_{tasks}$ assesses the similarity among tasks, and $\kappa$ mirrors the kernel functions in standard GPs, as delineated in equation (5). Within the context of this paper, the task indices $a$ and $b$ refer to the $a$th and $b$th optimization tasks, respectively. Subsequently, we will elucidate on the construction of the forward and inverse MTGP models.

- **Forward MTGP Building**: In each iteration, we construct a forward MTGP for the optimization tasks, mapping decision variables to aggregated objective values based on (3). The training set for task $k$ during forward modeling is $\mathcal{D}_k^{tch}$. Subsequently, the posterior distribution of the forward MTGP can be formulated as follows:

$$\mu_{fmt}(\mathbf{x}_{k^*}^{(*)}) = \bar{\mathbf{k}}_{fmt,*}^{\intercal}(\bar{\boldsymbol{\Sigma}}_{fmt} + \boldsymbol{\Lambda}_{fmt})^{-1}\bar{\mathbf{y}} \tag{8}$$

$$\begin{aligned}\sigma_{fmt}(\mathbf{x}_{k^*}^{(*)}) = \kappa_{fmt}((\mathbf{x}_{k^*}^{(*)}, k^*), (\mathbf{x}_{k^*}^{(*)}, k^*)) - \\ \bar{\mathbf{k}}_{fmt,*}^{\intercal}(\bar{\boldsymbol{\Sigma}}_{fmt} + \boldsymbol{\Lambda}_{fmt})^{-1}\bar{\mathbf{k}}_{fmt,*}\end{aligned} \tag{9}$$

  where $\bar{\mathbf{k}}_{fmt,*}$ is the multitask kernel vector between the input $\mathbf{x}_{k^*}^{(*)}$ and all the inputs in $\{\mathbf{X}_1, \ldots, \mathbf{X}_K\}$, $\boldsymbol{\Sigma}_{fmt}$ is the overall multitask kernel matrix corresponding to the forward MTGP, $\bar{\mathbf{y}}$ is the vector with all the weighted outputs in $\{\mathbf{y}_1, \ldots, \mathbf{y}_K\}$ where every single output can be obtained with the preference vector $\mathbf{w}$ and (3), and $\boldsymbol{\Lambda}_{fmt}$ is the additive noise variance matrix of the forward MTGP.
- **Inverse MTGP Building**: For inverse modeling, we establish an inverse MTGP model for each dimension of decision variables across all tasks. Given the training dataset $\{\mathcal{D}_1^{inv}, \ldots \mathcal{D}_K^{inv}\}$, the posterior distribution for a query preference vector $\mathbf{w}_{k^*}^{(*)}$, $\mathcal{N}(\mu_{imt,j}(\mathbf{w}_{k^*}^{(*)}), \sigma_{imt,j}(\mathbf{w}_{k^*}^{(*)}))$ can be computed using the formula (10) and (11) with the kernel function (7).

$$\mu_{imt,j}(\mathbf{w}_{k^*}^{(*)}) = \bar{\mathbf{k}}_{imt,*j}^{\intercal}(\bar{\boldsymbol{\Sigma}}_{imt,j} + \boldsymbol{\Lambda}_{imt,j})^{-1}\bar{\mathbf{X}}_j \tag{10}$$

$$\begin{aligned}\sigma_{imt,j}(\mathbf{w}_{k^*}^{(*)}) = &\kappa_{imt,j}((\mathbf{w}_{k^*}^{(*)}, k^*), (\mathbf{w}_{k^*}^{(*)}, k^*)) - \\ &\bar{\mathbf{k}}_{imt,*j}^{\intercal}(\bar{\boldsymbol{\Sigma}}_{imt,j} + \boldsymbol{\Lambda}_{imt,j})^{-1}\bar{\mathbf{k}}_{imt,*j}\end{aligned} \tag{11}$$

  where $\bar{\mathbf{k}}_{imt,*j}$ is the multitask kernel vector between the input $\mathbf{w}_{k^*}^{(*)}$ and all the inputs in $\{\mathbf{W}_1, \ldots, \mathbf{W}_K\}$, $\boldsymbol{\Sigma}_{imt,j}$ is the multitask kernel matrix corresponding to the $j$th inverse MTGP, $\bar{\mathbf{X}}_j$ contains all the outputs in $\{\mathbf{X}_{1,j}, \ldots, \mathbf{X}_{K,j}\}$ and $\boldsymbol{\Lambda}_{imt,j}$ is the additive noise variance matrix corresponding to the $j$th inverse MTGP.

**Table 1.** IGD results on benchmark problems and real-world problems.

| Problems | Tasks | ParEGO | PSL-MOBO | F-invTrEMO |
|---|---|---|---|---|
| mDTLZ-1 | Task-1 | 0.0897 (0.0214) – | 0.1307 (0.0512) – | **0.0611 (0.0051)** |
| | Task-2 | 0.1538 (0.0780) ≈ | 0.1877 (0.0699) ≈ | **0.1528 (0.1200)** |
| | Task-3 | 0.1282 (0.0281) – | 0.1876 (0.0955) – | **0.0731 (0.0099)** |
| | Task-4 | 0.1630 (0.0331) – | 0.2571 (0.0812) – | **0.1182 (0.0718)** |
| mDTLZ-2 | Task-1 | 0.1305 (0.0113) – | 0.1708 (0.0149) – | **0.1135 (0.0083)** |
| | Task-2 | 0.1656 (0.0207) – | 0.1798 (0.0185) – | **0.1489 (0.0174)** |
| | Task-3 | 0.1598 (0.0184) – | 0.2537 (0.0836) – | **0.1328 (0.0150)** |
| | Task-4 | 0.3054 (0.0590) – | 0.4187 (0.1235) – | **0.1439 (0.0154)** |
| mDTLZ-3 | Task-1 | 0.4371 (0.1679) – | 0.7681 (0.0535) – | **0.2726 (0.0277)** |
| | Task-2 | 0.5621 (0.1276) – | 0.8203 (0.0347) – | **0.3059 (0.0399)** |
| | Task-3 | 0.4907 (0.1474) – | 0.8142 (0.0624) – | **0.2833 (0.0356)** |
| | Task-4 | 0.5843 (0.1082) – | 0.8922 (0.0571) – | **0.3241 (0.0305)** |
| EO | Task-1 | 0.0152 (0.0024) – | 0.0549 (0.0100) – | **0.0095 (0.0011)** |
| | Task-2 | 0.0159 (0.0020) – | 0.0518 (0.0132) – | **0.0102 (0.0017)** |
| | Task-3 | 0.0142 (0.0026) – | 0.0526 (0.0090) – | **0.0104 (0.0013)** |
| HPO-1 | Task-1 | 0.0208 (0.0030) – | 0.0180 (0.0013) – | **0.0170 (0.0028)** |
| | Task-2 | 0.0222 (0.0023) – | 0.0264 (0.0022) – | **0.0169 (0.0012)** |
| | Task-3 | **0.0418 (0.0041)** ≈ | 0.0606 (0.0226) – | 0.0420 (0.0134) |
| HPO-2 | Task-1 | 0.0277 (0.0043) – | 0.0338 (0.0021) – | **0.0220 (0.0053)** |
| | Task-2 | 0.0227 (0.0027) – | 0.0226 (0.0026) – | **0.0160 (0.0016)** |

## 4 Experimental Studies

In this section, we assess the effectiveness of F-invTrEMO using various multitask multiobjective optimization problems, encompassing benchmark problems [32,40], an engineering optimization problem [41], and hyperparameter optimization problems [36]. We compare F-invTrEMO with the classical ParEGO, which only incorporates single-task forward GP models, and the state-of-the-art PSL-MOBO [30], which includes both single-task forward and inverse models. For all the methods, the evaluation budget for each task in optimizing the multitask MOPs is set to 100 evaluations, with an initialization budget $N_{init}$ of 20. During scalarization in each algorithm iteration, the weight $\rho$ is fixed at 0.05. For the UCB function, the parameter $\beta$ is set to 0.5. Due to the page limit, we introduce more experiments for comparison with a recent multiobjective Bayesian optimization method [35] and experiments as per advanced performance indicators including IGD$^+$ [26] and hypervolume [49] in the supplementary file [44].

### 4.1 Problem Settings

**Benchmark Problems.** Three base problems are formulated based on the well-established DTLZ benchmark [11], following the methodology outlined in prior research [40]. By adjusting the parameters in these base problems, three sets of multitask modified DTLZ (referred to as mDTLZ) problems can be generated [32,40]. In our experiment, the dimension of the decision space for all mDTLZ optimization tasks is set to six. Additional detailed information about the base problems can be found in the supplementary file [44], with insights into the corresponding parameters provided in [32,40].

**Real-World Problems.** Our real-world problems include an engineering optimization problem (*EO*) and two hyperparameter optimization problems (*HPO*).

It is commonplace to seek optimal engineering designs for components under diverse conditions, such as varying load cases or production materials. In such scenarios, obtaining high-quality solutions corresponding to each environment is imperative. Our case study centers on an engineering optimization problem known as the four-bar truss design problem [6]. We endeavor to identify a set of optimal designs for the four-bar truss under three distinct load cases. Further details regarding the parameter settings of the design problem are available in the supplementary file [44], while the formulation of objectives can be found in [41].

Hyperparameter optimization is a classical topic in the field of machine learning [18]. The configuration of hyperparameters for machine learning models can impact the model performance, computational resources, and interpretability to decision-makers. In this paper, we apply F-invTrEMO to generate optimal hyperparameter sets concurrently across diverse related tasks. We consider the following two scenarios. The first problem set (HPO-1) contains three hyperparameter optimization problems. The three optimization problems adjust the hyperparameters of the same machine-learning model, XGBoost [5], but on three distinct classification tasks including credit approval, medical diagnosis, and speech classification problems. The second problem set (HPO-2) contains two hyperparameter optimization problems. The two optimization problems adjust the hyperparameters of distinct but related machine learning models (i.e., XGBoost in 'gbtree' mode and 'dart' mode.) on the same classification task, medical diagnosis problem [36].

All the hyperparameter optimization case studies are implemented using the framework of YAHPO Gym [36]. Each task for a problem set is a tri-objective hyperparameter tuning, targetting at classification accuracy, model RAM, and model interpretability. Further details can be found in the supplementary file [44] and OpenML problems [43].

### 4.2 Performance Metrics

We involve two metrics to assess the performance of the algorithms, including the inverted generational distance (IGD) [42] and root mean square error (RMSE).

The Inverted Generational Distance (IGD) is a commonly utilized metric in multiobjective optimization. It quantifies the Euclidean distance between the objective function values of the obtained nondominated solution sets and the true PF. For the $k$th task, the corresponding IGD value is computed as follows:

$$IGD_k = \frac{1}{N^r} \sum_{r=1}^{N^r} \min\{||\mathbf{F}_k(\mathbf{x}_{opt,k}^{(r)}) - \mathbf{F}_k(\mathbf{x}_k^{(1)})||_2, \ldots, ||\mathbf{F}_k(\mathbf{x}_{opt,k}^{(r)}) - \mathbf{F}_k(\mathbf{x}_k^{(N_k^{nd})})||_2\} \tag{12}$$

**Table 2.** RMSE results on benchmark problems and real-world problems.

| Problems | Tasks | PSL-MOBO | F-invTrEMO |
|---|---|---|---|
| mDTLZ-1 | Task-1 | 0.3021 (0.0744) – | **0.0721 (0.0173)** |
| | Task-2 | **0.3828 (0.0770)** ≈ | 0.4250 (0.2263) |
| | Task-3 | 0.3790 (0.0837) – | **0.1592 (0.0780)** |
| | Task-4 | 0.4019 (0.0787) – | **0.3568 (0.1691)** |
| mDTLZ-2 | Task-1 | 0.3501 (0.1321) – | **0.0679 (0.0295)** |
| | Task-2 | 0.2997 (0.0146) – | **0.1436 (0.0219)** |
| | Task-3 | 0.7010 (0.1819) – | **0.1700 (0.0454)** |
| | Task-4 | 0.9420 (0.1361) – | **0.3248 (0.1767)** |
| mDTLZ-3 | Task-1 | 1.0219 (0.0310) – | **0.6946 (0.0392)** |
| | Task-2 | 1.0430 (0.0429) – | **0.7412 (0.0325)** |
| | Task-3 | 1.0562 (0.0457) – | **0.7830 (0.0361)** |
| | Task-4 | 1.1166 (0.0747) – | **0.8301 (0.0423)** |
| EO | Task-1 | 0.2211 (0.0242) ≈ | **0.2173 (0.0054)** |
| | Task-2 | 0.2290 (0.0261) – | **0.2144 (0.0067)** |
| | Task-3 | 0.2225 (0.0372) ≈ | **0.2046 (0.0053)** |
| HPO-1 | Task-1 | 0.1818 (0.1143) – | **0.0943 (0.0721)** |
| | Task-2 | 0.1663 (0.1558) – | **0.0626 (0.0125)** |
| | Task-3 | 0.1881 (0.0545) – | **0.1273 (0.0411)** |
| HPO-2 | Task-1 | 0.1855 (0.1075) – | **0.0811 (0.0476)** |
| | Task-2 | 0.1572 (0.1612) – | **0.0694 (0.0260)** |

where $\{\mathbf{x}_{opt,k}^{(1)}, \ldots, \mathbf{x}_{opt,k}^{(N^r)}\}$ represents a set of true Pareto optimal solutions that correspond to well-distributed points along the Pareto front, and $\mathbf{x}_k^{(1)}, \ldots, \mathbf{x}_k^{(N_k^{nd})}$

denote the nondominated solutions obtained in task $k$ by the optimizer. In this paper, $N^r$ is set to 2000.[2]

The RMSE aims to measure the accuracy of the obtained inverse models. Given a test set $\mathcal{D}_{opt,k} = \{(\mathbf{w}_{opt,k}^{(r)}, \mathbf{x}_{opt,k}^{(r)})\}_{r=1}^{N^{test}}$, where $\mathbf{w}_{opt,k}^{(r)}$ is the preference vector corresponding to $\mathbf{x}_{opt,k}^{(r)}$. Since the goal of the inverse model is to satisfy DM's preferences articulated in the objective space, we calculate the RMSE of the $k$th task in the objective space as:

$$RMSE_k = \sqrt{\frac{\sum_{r=1}^{N^r} ||\mathbf{F}_k(\mathbf{x}_{opt,k}^{(r)}) - \mathbf{F}_k(\mathbf{x}_{pred,k}^{(r)})||_2^2}{N^r}} \tag{13}$$

where $\mathbf{x}_{pred,k}^{(r)}$ is the prediction of the inverse models with respect to $\mathbf{w}_{opt,k}^{(r)}$. The predicted means of the inverse models are taken as the prediction $\mathbf{x}_{pred,k}^{(r)}$.

### 4.3 Results

In Tables 1 to 3, numbers with indicators (+), (−) and (≈) imply that the compared algorithm is better than, worse than, or similar to F-invTrEMO at 95% confidence level as per the Wilcoxon signed-rank test.

**Table 3.** IGD and RMSE results on benchmark problems for F-invTrEMO with and without forward multitask GP (invTrEMO).

| Problems | Tasks | IGD | | RMSE | |
|---|---|---|---|---|---|
| | | invTrEMO | F-invTrEMO | invTrEMO | F-invTrEMO |
| mDTLZ-1 | Task-1 | 0.0820 (0.0126) − | **0.0611 (0.0051)** | 0.1174 (0.0358) − | **0.0721 (0.0173)** |
| | Task-2 | **0.1264 (0.0699)** ≈ | 0.1528 (0.1200) | 0.4857 (0.2244) ≈ | **0.4250 (0.2263)** |
| | Task-3 | 0.1203 (0.0301) − | **0.0731 (0.0099)** | 0.3181 (0.1851) − | **0.1592 (0.0780)** |
| | Task-4 | 0.1457 (0.0427) ≈ | **0.1182 (0.0718)** | 0.4911 (0.2010) − | **0.3568 (0.1691)** |
| mDTLZ-2 | Task-1 | 0.1157 (0.0074) ≈ | **0.1135 (0.0083)** | **0.0637 (0.0244)** ≈ | 0.0679 (0.0295) |
| | Task-2 | **0.1284 (0.0118)** + | 0.1489 (0.0174) | 0.1466 (0.0484) ≈ | **0.1436 (0.0219)** |
| | Task-3 | 0.1350 (0.0120) ≈ | **0.1328 (0.0150)** | **0.1556 (0.0273)** ≈ | 0.1700 (0.0454) |
| | Task-4 | 0.2192 (0.1015) − | **0.1439 (0.0154)** | 0.3360 (0.1605) ≈ | **0.3248 (0.1767)** |
| mDTLZ-3 | Task-1 | 0.3090 (0.1115) − | **0.2726 (0.0277)** | 0.7355 (0.0633) − | **0.6946 (0.0392)** |
| | Task-2 | 0.4589 (0.1682) − | **0.3059 (0.0399)** | 0.8288 (0.1004) − | **0.7412 (0.0325)** |
| | Task-3 | 0.3103 (0.0743) − | **0.2833 (0.0356)** | 0.7837 (0.0650) ≈ | **0.7830 (0.0361)** |
| | Task-4 | 0.3941 (0.1475) − | **0.3241 (0.0305)** | 0.8403 (0.0803) ≈ | **0.8301 (0.0423)** |

[2] For benchmark problems, the ground truth solutions serve as the reference solution set. For real-world problems, bi-objective and tri-objective reference solution sets are obtained using NSGA-II [13] and NSGA-III [12] with population size 500 and generation size 1000, similar to [40].

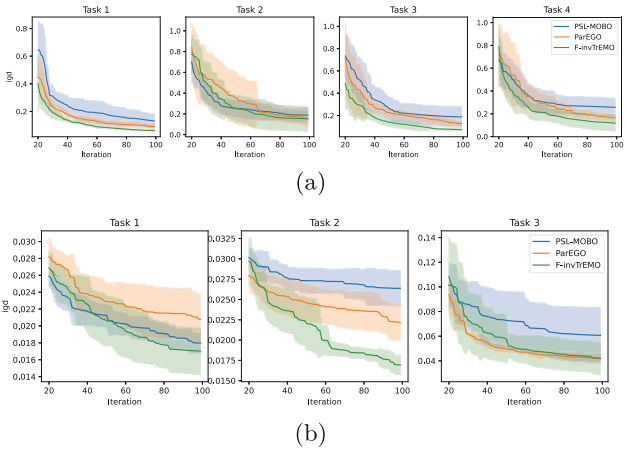



**Fig. 2.** The IGD convergence trends of the mDTLZ-1 problem with four tasks and the HPO-1 problem with three tasks provided by ParEGO, PSL-MOBO, and F-invTrEMO (a) The mDTLZ-1 problem. (b) The HPO-1 problem.

**Comparison with ParEGO and PSL-MOBO.** In Table 1, we present a summary of the IGD (mean and standard deviation) obtained by ParEGO, PSL-MOBO, and F-invTrEMO after 100 evaluations with 10 independent runs. Notably, F-invTrEMO outperforms the other methods in terms of IGD results for 18 out of the 20 tasks, indicating a better convergence of F-invTrEMO than its two competitors. Additionally, in Fig. 2, we illustrate the convergence trends of ParEGO, PSL-MOBO, and F-invTrEMO on mDTLZ-1 and HPO-1 problems. Figure 2 demonstrates that F-invTrEMO exhibits faster convergence compared to both ParEGO and PSL-MOBO. These results underscore the capability of F-invTrEMO to achieve superior convergence outcomes.

In addition to evaluating convergence performance, we also assess the accuracy of the obtained inverse models. Since only PSL-MOBO provides inverse models, Table 2 presents the RMSE results obtained by PSL-MOBO and F-invTrEMO. The findings indicate that, for 17 out of the 20 tasks, F-invTrEMO achieves superior RMSE values, suggesting that F-invTrEMO can produce more accurate inverse models through knowledge transfer across multiple tasks.

**Ablation Study.** In F-invTrEMO, we incorporate the forward transfer to maximize the utilization of information from both dominated and nondominated samples across distinct tasks based on the inverse transfer model. This study verifies the forward transfer's effectiveness by comparing F-invTrEMO that fully exploits the interplay between forward transfer and inverse transfer models with invTrEMO that is solely based on inverse transfer models. The obtained IGD and RMSE results are presented in Table 3. Our analysis reveals that the convergence of F-invTrEMO is significantly improved with the inclusion of a forward MTGP model. In terms of both IGD and RMSE results, F-invTrEMO outperforms

invTrEMO in 7 out of the 12 tasks on IGD and 5 of 12 tasks on RMSE. These findings suggest that by leveraging information from both dominated and nondominated samples corresponding to different tasks, the algorithm enhances both the convergence of the optimization and the accuracy of the inverse models.

## 5 Conclusion

In this paper, we incorporate knowledge transfer in both forward and inverse surrogate modelling for multiobjective optimization. It's argued that both forward and inverse models encounter the challenge of data scarcity. Both forward and inverse knowledge transfer are employed to alleviate this problem. Forward knowledge transfer improves convergence performance, while inverse knowledge transfer enhances the quality of approximating PF, both by utilizing the available information across distinct tasks through multitask modeling. The results are compared with those using only surrogate models without forward or inverse knowledge transfer across benchmark problems and real-world problems. These studies indicate that the proposed mechanism can enhance both the convergence performance and PF approximation results in multitask multiobjective optimization under strict computational constraints.

**Acknowledgement.** This research is partly supported by the National Research Foundation, Singapore and DSO National Laboratories under the AI Singapore Programme (AISG Award No.: AISG2-GC-2023-010, "Design Beyond What You Know": Material-Informed Differential Generative AI (MIDGAI) for Light-Weight High-Entropy Alloys and Multi-functional Composites (Stage 1a)", the Distributed Smart Value Chain programme which is funded under the Singapore RIE2025 Manufacturing, Trade and Connectivity (MTC) Industry Alignment Fund-Pre-Positioning, (Award No: M23L4a0001), and the Centre for Frontier AI Research (CFAR) under Agency for Science, Technology and Research (A*STAR), and the College of Computing and Data Science, Nanyang Technological University.

**Disclosure of Interests.** The authors have no competing interests to declare that are relevant to the content of this article.

## References

1. Bali, K.K., Gupta, A., Ong, Y.S., Tan, P.S.: Cognizant multitasking in multiobjective multifactorial evolution: MO-MFEA-II. IEEE Trans. Cybern. **51**(4), 1784–1796 (2021)
2. Bali, K.K., Ong, Y.S., Gupta, A., Tan, P.S.: Multifactorial evolutionary algorithm with online transfer parameter estimation: MFEA-II. IEEE Trans. Evol. Comput. **24**(1), 69–83 (2019)
3. Bonilla, E.V., Chai, K., Williams, C.: Multi-task gaussian process prediction. Adv. Neural Inf. Process. Syst. **20** (2007)
4. Branke, J., Deb, K., Miettinen, K., Słowiński, R. (eds.): Multiobjective Optimization: Interactive and Evolutionary Approaches. Springer, Berlin, Heidelberg (2008)

5. Chen, T., Guestrin, C.: XGBoost: a scalable tree boosting system. In: Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 785–794. KDD '16, Association for Computing Machinery, New York, NY, USA (2016). https://doi.org/10.1145/2939672.2939785
6. Cheng, F.Y., Li, X.S.: Generalized center method for multiobjective engineering optimization. Eng. Optim. **31**(5), 641–661 (1999).https://doi.org/10.1080/03052159908941390
7. Cheng, R., Jin, Y., Narukawa, K., Sendhoff, B.: A multiobjective evolutionary algorithm using Gaussian process-based inverse modeling. IEEE Trans. Evol. Comput. **19**(6), 838–856 (2015). https://doi.org/10.1109/TEVC.2015.2395073
8. Choong, H.X., Ong, Y.S., Gupta, A., Chen, C., Lim, R.: Jack and masters of all trades: one-pass learning sets of model sets from large pre-trained models. IEEE Comput. Intell. Mag. **18**(3), 29–40 (2023). https://doi.org/10.1109/MCI.2023.3277769
9. Coello, C.A.C.: Evolutionary Algorithms for Solving Multi-objective Problems. Springer, Cham (2007). https://doi.org/10.1007/978-0-387-36797-2
10. Da, B., Gupta, A., Ong, Y.S.: Curbing negative influences online for seamless transfer evolutionary optimization. IEEE Trans. Cybern. **49**(12), 4365–4378 (2018)
11. Deb, K., Thiele, L., Laumanns, M., Zitzler, E.: Scalable multi-objective optimization test problems. In: Proceedings of the 2002 Congress on Evolutionary Computation, CEC'02, Cat. No.02TH8600, vol. 1, pp. 825–830 (2002). https://doi.org/10.1109/CEC.2002.1007032
12. Deb, K., Jain, H.: An evolutionary many-objective optimization algorithm using reference-point-based nondominated sorting approach, part i: solving problems with box constraints. IEEE Trans. Evol. Comput. **18**(4), 577–601 (2014). https://doi.org/10.1109/TEVC.2013.2281535
13. Deb, K., Pratap, A., Agarwal, S., Meyarivan, T.: A fast and elitist multiobjective genetic algorithm: NSGA-II. IEEE Trans. Evol. Comput. **6**(2), 182–197 (2002)
14. Dellnitz, M., Schütze, O., Hestermeyer, T.: Covering Pareto sets by multilevel subdivision techniques. J. Optim. Theory Appl. **124**, 113–136 (2005)
15. Emmerich, M., Giannakoglou, K., Naujoks, B.: Single and multiobjective evolutionary optimization assisted by Gaussian random field metamodels. IEEE Trans. Evol. Comput. **10**(4), 421–439 (2006). https://doi.org/10.1109/TEVC.2005.859463
16. Feng, L., Gupta, A., Tan, K.C., Ong, Y.S.: Evolutionary Multi-Task Optimization: Foundations and Methodologies. Springer, Singapore (2023). https://doi.org/10.1007/978-981-19-5650-8
17. Feng, L., Zhou, L., Zhong, J., Gupta, A., Ong, Y.S., Tan, K.C.: Evolutionary multitasking via explicit autoencoding. IEEE Trans. Cybern. **49**(9), 3457–3470 (2018)
18. Feurer, M., Hutter, F.: Hyperparameter optimization. In: Hutter, F., Kotthoff, L., Vanschoren, J. (eds.) Automated Machine Learning: Methods, Systems, Challenges, pp. 3–33. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-05318-5_1
19. Gardner, J., Pleiss, G., Weinberger, K.Q., Bindel, D., Wilson, A.G.: GPyTorch: blackbox matrix-matrix Gaussian process inference with GPU acceleration. In: Bengio, S., Wallach, H., Larochelle, H., Grauman, K., Cesa-Bianchi, N., Garnett, R. (eds.) Advances in Neural Information Processing Systems, vol. 31. Curran Associates, Inc. (2018)
20. Giagkiozis, I., Fleming, P.J.: Pareto front estimation for decision making. Evol. Comput. **22**(4), 651–678 (2014)

21. Gupta, A., Ong, Y.S., Feng, L.: Multifactorial evolution: toward evolutionary multitasking. IEEE Trans. Evol. Comput. **20**(3), 343–357 (2016)
22. Gupta, A., Ong, Y.S., Feng, L.: Insights on transfer optimization: because experience is the best teacher. IEEE Trans. Emerg. Topics Comput. Intell. **2**(1), 51–64 (2017)
23. Gupta, A., Ong, Y.S., Feng, L., Tan, K.C.: Multiobjective multifactorial optimization in evolutionary multitasking. IEEE Trans. Cybern. **47**(7), 1652–1665 (2016)
24. Gupta, A., Ong, Y.S., Shakeri, M., Chi, X., NengSheng, A.Z.: The blessing of dimensionality in many-objective search: an inverse machine learning insight. In: 2019 IEEE International Conference on Big Data (Big Data), pp. 3896–3902 (2019). https://doi.org/10.1109/BigData47090.2019.9005525
25. Gupta, A., Zhou, L., Ong, Y.S., Chen, Z., Hou, Y.: Half a dozen real-world applications of evolutionary multitasking, and more. IEEE Comput. Intell. Mag. **17**(2), 49–66 (2022). https://doi.org/10.1109/MCI.2022.3155332
26. Ishibuchi, H., Masuda, H., Tanigaki, Y., Nojima, Y.: Modified distance calculation in generational distance and inverted generational distance. In: Gaspar-Cunha, A., Henggeler Antunes, C., Coello, C.C. (eds.) Evolutionary Multi-Criterion Optimization, pp. 110–125. Springer, Cham (2015). https://doi.org/10.1007/978-3-319-15892-1_8
27. Kim, Y., Pan, Z., Hauser, K.: MO-BBO: multi-objective bilevel Bayesian optimization for robot and behavior co-design. In: 2021 IEEE International Conference on Robotics and Automation (ICRA), pp. 9877–9883 (2021).https://doi.org/10.1109/ICRA48506.2021.9561846
28. Knowles, J.: Parego: a hybrid algorithm with on-line landscape approximation for expensive multiobjective optimization problems. IEEE Trans. Evol. Comput. **10**(1), 50–66 (2006). https://doi.org/10.1109/TEVC.2005.851274
29. Lai, G., Liao, M., Li, K.: Empirical studies on the role of the decision maker in interactive evolutionary multi-objective optimization. In: 2021 IEEE Congress on Evolutionary Computation (CEC), pp. 185–192 (2021).https://doi.org/10.1109/CEC45853.2021.9504980
30. Lin, X., Yang, Z., Zhang, X., Zhang, Q.: Pareto set learning for expensive multiobjective optimization. In: Advances in Neural Information Processing Systems. vol. 35, pp. 19231–19247. Curran Associates, Inc. (2022)
31. Lin, X., Zhen, H.L., Li, Z., Zhang, Q.F., Kwong, S.: Pareto multi-task learning. Adv. Neural Inf. Process. Syst. **32** (2019)
32. Liu, J., Gupta, A., Ong, Y.S.: Inverse transfer multiobjective optimization. arXiv preprint arXiv:2312.14713 (2023)
33. Ma, J., Zhao, Z., Yi, X., Chen, J., Hong, L., Chi, E.H.: Modeling task relationships in multi-task learning with multi-gate mixture-of-experts. In: Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 1930–1939. KDD '18, Association for Computing Machinery, New York, NY, USA (2018). https://doi.org/10.1145/3219819.3220007
34. Ma, P., Du, T., Matusik, W.: Efficient continuous pareto exploration in multi-task learning. In: III, H.D., Singh, A. (eds.) Proceedings of the 37th International Conference on Machine Learning, Proceedings of Machine Learning Research, vol. 119, pp. 6522–6531. PMLR (2020)
35. Paria, B., Kandasamy, K., Póczos, B.: A flexible framework for multi-objective Bayesian optimization using random scalarizations. In: Adams, R.P., Gogate, V. (eds.) Proceedings of The 35th Uncertainty in Artificial Intelligence Conference. Proceedings of Machine Learning Research, vol. 115, pp. 766–776. PMLR (2020). https://proceedings.mlr.press/v115/paria20a.html

36. Pfisterer, F., Schneider, L., Moosbauer, J., Binder, M., Bischl, B.: YAHPO gym - an efficient multi-objective multi-fidelity benchmark for hyperparameter optimization. In: Proceedings of the First International Conference on Automated Machine Learning. Proceedings of Machine Learning Research, vol. 188, pp. 31–39. PMLR (2022)
37. Ponweiser, W., Wagner, T., Biermann, D., Vincze, M.: Multiobjective optimization on a limited budget of evaluations using model-assisted s-metric selection. In: Rudolph, G., Jansen, T., Beume, N., Lucas, S., Poloni, C. (eds) Parallel Problem Solving from Nature – PPSN X. PPSN 2008. Lecture Notes in Computer Science, vol. 5199. Springer, Berlin, Heidelberg (2008). https://doi.org/10.1007/978-3-540-87700-4_78
38. Seeger, M.: Gaussian processes for machine learning. Int. J. Neural Syst. **14**(02), 69–106 (2004)
39. Sinha, A., Korhonen, P., Wallenius, J., Deb, K.: An interactive evolutionary multiobjective optimization algorithm with a limited number of decision maker calls. Eur. J. Oper. Res. **233**(3), 674–688 (2014). https://doi.org/10.1016/j.ejor.2013.08.046
40. Tan, C.S., Gupta, A., Ong, Y.S., Pratama, M., Tan, P.S., Lam, S.K.: Pareto optimization with small data by learning across common objective spaces. Sci. Rep. **13**(1), 7842 (2023). https://doi.org/10.1038/s41598-023-33414-6
41. Tanabe, R., Ishibuchi, H.: An easy-to-use real-world multi-objective optimization problem suite. Appl. Soft Comput. **89**, 106078 (2020). https://doi.org/10.1016/j.asoc.2020.106078
42. Van Veldhuizen, D.A., Lamont, G.B.: Multiobjective evolutionary algorithm research: a history and analysis. Technical report, Citeseer (1998)
43. Vanschoren, J., van Rijn, J.N., Bischl, B., Torgo, L.: Openml: Networked science in machine learning. SIGKDD Explor. Newsl. **15**(2), 49–60 (2014). https://doi.org/10.1145/2641190.2641198
44. Wei, T., Liu, J., Gupta, A., Tan, P.S., Ong, Y.S.: Bayesian forward-inverse transfer for multiobjective optimization - supplementary materials (2024). https://doi.org/10.5281/zenodo.11665260 https://doi.org/10.5281/zenodo.11665260
45. Wei, T., Wang, S., Zhong, J., Liu, D., Zhang, J.: A review on evolutionary multitask optimization: trends and challenges. IEEE Trans. Evol. Comput. **26**(5), 941–960 (2022). https://doi.org/10.1109/TEVC.2021.3139437
46. Wei, T., Zhong, J.: Towards generalized resource allocation on evolutionary multitasking for multi-objective optimization. IEEE Comput. Intell. Mag. **16**(4), 20–37 (2021). https://doi.org/10.1109/MCI.2021.3108310
47. Zhang, Q., Li, H.: MOEA/D: a multiobjective evolutionary algorithm based on decomposition. IEEE Trans. Evol. Comput. **11**(6), 712–731 (2007)
48. Zhang, Q., Liu, W., Tsang, E., Virginas, B.: Expensive multiobjective optimization by MOEA/D with Gaussian process model. IEEE Trans. Evol. Comput. **14**(3), 456–474 (2010). https://doi.org/10.1109/TEVC.2009.2033671
49. Zitzler, E., Thiele, L.: Multiobjective optimization using evolutionary algorithms — a comparative case study. In: Eiben, A.E., Bäck, T., Schoenauer, M., Schwefel, H.P. (eds.) PPSN V, pp. 292–301. Springer, Berlin, Heidelberg (1998). https://doi.org/10.1007/BFb0056872

# Near-Tight Runtime Guarantees for Many-Objective Evolutionary Algorithms

Simon Wietheger[1(✉)] and Benjamin Doerr[2]

[1] Algorithms and Complexity Group, Technische Universität Wien, Vienna, Austria
swietheger@ac.tuwien.ac.at
[2] Laboratoire d'Informatique (LIX), CNRS, École Polytechnique, Institut Polytechnique de Paris, Palaiseau, France
doerr@lix.polytechnique.fr

**Abstract.** Despite significant progress in the field of mathematical runtime analysis of multi-objective evolutionary algorithms (MOEAs), the performance of MOEAs on discrete many-objective problems is little understood. In particular, the few existing bounds for the SEMO, global SEMO, and SMS-EMOA algorithms on classic benchmarks are all roughly quadratic in the size of the Pareto front.

In this work, we prove near-tight runtime guarantees for these three algorithms on the four most common benchmark problems OneMinMax, CountingOnesCountingZeros, LeadingOnesTrailingZeros, and OneJumpZeroJump, and this for arbitrary numbers of objectives. Our bounds depend only linearly on the Pareto front size, showing that these MOEAs on these benchmarks cope much better with many objectives than what previous works suggested. Our bounds are tight apart from small polynomial factors in the number of objectives and length of bitstrings. This is the first time that such tight bounds are proven for many-objective uses of these MOEAs. While it is known that such results cannot hold for the NSGA-II, we do show that our bounds, via a recent structural result, transfer to the NSGA-III algorithm.

**Keywords:** evolutionary multi-objective optimization · runtime analysis · SMS-EMOA · NSGA · theory

## 1 Introduction

Evolutionary algorithms such as the SMS-EMOA, NSGA-II, or MOEA/D are among the most successful approaches to tackle optimization problems with several conflicting objectives [7,33]. Despite the challenges stemming from the often complex population dynamics in multi-objective evolutionary algorithms (MOEAs), the analysis of MOEAs via theoretical means has made considerable progress in the last twenty years. Starting with simplistic algorithms such as the *simple evolutionary multi-objective optimizer (SEMO)* [21] and the global


M. Affenzeller et al. (Eds.): PPSN 2024, LNCS 15151, pp. 153–168, 2024.
https://doi.org/10.1007/978-3-031-70085-9_10

SEMO [16], this line of research has now reached the maturity to deal with state-of-the-art algorithms such as the MOEA/D, NSGA-II, NSGA-III, and SMS-EMOA [4,22,26,28].

However, this progress so far was mostly restricted to the analysis of MOEAs on bi-objective optimization problems. In particular, the few existing mathematical runtime analyses for the SEMO, global SEMO, and SMS-EMOA [3,20,31] for problems with general number $m$ of objectives show runtime guarantees roughly quadratic in the size of the Pareto front; the recent work [29] proves that the NSGA-II cannot optimize the OneMinMax problem in time better than exponential in the size of the Pareto front when the number of objectives is three or more. These results could give the impression that MOEAs have significant difficulties in dealing with larger numbers of objectives, clearly beyond the mere increase of the size of the Pareto front with increasing numbers of objectives.

In this work, we revisit this twenty years old question and prove significantly stronger performance guarantees, which give a different impression. More specifically, we analyze the runtimes of the SEMO, global SEMO, SMS-EMOA, and NSGA-III on the OneMinMax (OMM), CountingOnesCountingZeros (COCZ), LeadingOnesTrailingZeros (LOTZ), and OneJumpZeroJump (OJZJ) problems for a general (even) number $m$ of objectives. We prove runtime guarantees showing that these algorithms compute the Pareto fronts of these problems in an expected time (number of function evaluations) that is linear in the size of the largest set of incomparable solutions (apart from small polynomial factors in the number $m$ of objectives and the bitstring length $n$). For all benchmarks except LOTZ and OJZJ with large jump size, this number coincides with or is close to the size of the Pareto front. Since naturally the size of the Pareto front is a lower bound for these runtimes, these guarantees are tight apart from the small factors. In the bi-objective case ($m = 2$) our results match the state-of-the-art bounds (apart from constant factors).

Together with the parallel and independent work on the NSGA-III [25], these are the first that tight runtime guarantees for these MOEAs for general numbers of objectives, and they improve significantly over the previous results with their quadratic dependence on the Pareto front size.

We are optimistic that our methods can also be applied to other MOEAs. We discuss some sufficient conditions for transferring the obtained bounds. We argue that they are fulfilled by SMS-EMOA and NSGA-III, but we believe that our arguments can also be applied to variants of the NSGA-II that do not suffer from the problems detected in [29], e.g., the NSGA-II with the tie-breaker proposed in [15], and to the $(\mu + 1)$ SIBEA [5].

## 2 Previous Work

The mathematical runtime analysis of randomized search heuristics is an active area of research for more than 30 years now, see [1,13,19,24,34]. Since around 20 years ago, also the runtime of multi-objective evolutionary algorithms (MOEAs) has been analyzed with mathematical means. Starting with simple toy algorithms

like the *simple evolutionary multi-objective optimizer (SEMO)* [21] or the *global SEMO (GSEMO)* [16], the field has steadily progressed and is now able to also predict the runtimes of state-of-the-art algorithms such as the MOEA/D [22], NSGA-II [32], NSGA-III [26], and the SMS-EMOA [4].

Looking closer at the results obtained, we note that the vast majority of the runtime analyses of MOEAs consider only bi-objective problems. The sporadic results regarding more than two objectives appear less mature, and the runtime guarantees are far from the (mostly trivial) existing lower bounds.

One natural reason for the additional difficulty of runtime analyses for many-objective problems, visible from comparing the proofs of results for two and for more objectives, is the richer structure of the Pareto front. In bi-objective problems, the Pareto front has a one-dimensional structure. Hence the typical runtime analysis first estimates the time to find some solution on the Pareto front and then regards how the MOEA progresses along the Pareto front in the only two directions available. For problems with more objectives, the Pareto front is higher-dimensional, and hence there are many search trajectories from the first solution on the Pareto front to a particular solution.

The main runtime results for more than two objectives are the following. Already the journal version [20, Section V] of the first runtime analysis work on MOEAs [21] contains two many-objective runtime results, namely proofs that the SEMO computes the Pareto front of $m$COCZ and $m$LOTZ (which are the $m$-objective analogues of the classic COCZ and LOTZ benchmarks) with problem size $n$ and even number $m \geq 4$ of objectives in an expected number of $O(n^{m+1})$ function evaluations. While the result for $m$LOTZ naturally extends the $O(n^3)$ bound for the bi-objective LOTZ problem, the same is not true for $m$COCZ, where the bi-objective runtime guarantee is $O(n^2 \log n)$. Considerably later, the bounds for $m$COCZ were slightly improved in [3], namely to $O(n^m)$ for $m > 4$ and to $O(n^3 \log n)$ for the special case $m = 4$. As often in the runtime analysis of MOEAs, the complicated population dynamics prevented the proof of any interesting lower bounds, so only the trivial bound $\Omega(n^{m/2}\Theta(m)^{-m/2})$, which is the size of the Pareto front for both problems, is known.

Huang, Zhou, Luo and Lin [18] analyzed how the MOEA/D [27] optimizes the benchmarks $m$COCZ and $m$LOTZ. As the MOEA/D decomposes the multi-objective problem into several single-objective subproblems and solves these in a co-evolutionary way, this framework is fundamentally different from the MOEAs regarded in this work, so we do not discuss these results in more detail.

Surprisingly, the NSGA-II [9] has enormous difficulties with discrete many-objective problems. Zheng and Doerr [29] showed that this prominent algorithm with any population size that is linear in the Pareto front size cannot optimize the OMM problem in polynomial time (in expectation) when the number of objectives is three or more. The proof of this result suggests that this is an intrinsic problem of the crowding distance, and that similar negative results hold for the other benchmark problems in this work.

The NSGA-III [8] might cope better with more objectives, however, this was proven only for the 3-objective OMM problem, where a runtime guarantee of

$O(Nn \log n)$ function evaluations was shown in [26] (when the population size $N$ is at least the size $(\frac{n}{2}+1)^2$ of the Pareto front). In a very recent parallel and independent work, [25] generalized the property of the NSGA-III used in the analyses of the 3-objective OMM problem to arbitrary benchmarks and numbers of objectives. They used this property to prove the runtime of the NSGA-III (for a proper choice of reference points and population size $\mu$) to be $O(\mu n \log n)$ on $m$OMM and $m$COCZ, and $O(\mu n^2)$ on $m$LOTZ, for arbitrary but constant and even $m$. Our results on the NSGA-III heavily rely on their proven property and extend their results to arbitrary, non-constant numbers of objectives.

Very recently [31], the runtime of the GSEMO and the SMS-EMOA [2] on the $\text{OJZJ}_k$ problem for arbitrary (even) numbers $m$ of objective was shown to be $O(M^2 n^k)$ and $O(\mu M n^k)$, respectively, where $M = (2\frac{n}{m} - 2k + 3)^{m/2}$ is the size of the Pareto front of this problem and $\mu \geq M$ denotes the size of the population of the SMS-EMOA.

We note that all bounds discussed in this section except for the ones of the NSGA-III are quadratic in the Pareto front size, times some small polynomial in $m$ and $n$. As our results will show, this quadratic dependence is not necessary and merely stems from the difficulty to analyze many-objective MOEAs.

## 3 Preliminaries

Let $\mathbb{N} = \{1, 2, \ldots\}$ and for $n \in \mathbb{N}$, let $[n] = \{1, \ldots, n\}$. Whenever we are speaking of how close one bitstring is to another, we are referring to their Hamming distance. Many of our proofs rely on two well-known bounds in probability theory, which we give here for completeness and reference in our proofs by name. First, a *union bound* states for some finite set of events $\{E_1, \ldots, E_\ell\}$ that the probability of any of the events happening is at most the sum over the individual probabilities, that is, $\Pr[\bigcup_{i\in[\ell]} E_i] \leq \sum_{i\in[\ell]} \Pr[E_i]$. Second, we employ a variant of the *Chernoff bound*. Let $X_1, \ldots, X_\ell$ be independent random variables with values in $\{0, 1\}$ and let $X = \sum_{i\in[\ell]} X_i$. Then for any $0 < \delta < 1$ we have

$$\Pr[X \leq (1-\delta)E[X]] \leq \exp\big(-\tfrac{1}{2}\delta^2 E[X]\big).$$

In particular, $\Pr[X \leq \frac{1}{2}E[X]] \leq \exp(-\frac{1}{8}E[X])$.

### 3.1 Multi-objective Optimization

Let $m \in \mathbb{N}$. An $m$-objective function $f$ is a tuple $(f_1, \ldots, f_m)$ such that $f_i \colon \Omega \to \mathbb{R}$ for some search space $\Omega$, for all $i \in [m]$. We define the objective value of $x \in \Omega$ to be $f(x) = (f_1(x), \ldots, f_m(x))$. There is usually no solution that maximizes all $m$ objective functions at the same time. For $x, y \in \Omega$ we write $x \succeq y$ if and only if $f_i(x) \geq f_i(y)$ for all $i \in [m]$ and say that $x$ *dominates* $y$. If additionally $f_j(x) > f_j(y)$ for some $j \in [m]$, we say that $x$ *strictly dominates* $y$ and write $x \succ y$. A solution $x \in \Omega$ is Pareto-optimal if it is not strictly dominated by any other solution. The *Pareto front* is the set of objective values of Pareto-optimal

solutions. Given an algorithm and an objective function, we are interested in the number of function evaluations until the population covers the Pareto front, that is, until for all values $p$ on the Pareto front there is a solution $x$ in the population such that $f(x) = p$. We note that all algorithms analyzed in this work in each iteration create one new individual and thus only require one function evaluation per iteration. Thus we simply analyze the number of iterations until the Pareto front is covered and remark here that the initial population also has to be evaluated (1 evaluation for the SEMO and GSEMO, $\mu$ iterations for the SMS-EMOA with population size $\mu$).

All the objective functions we consider are defined on the search space of bitstrings of length $n$ for $n \in \mathbb{N}$. For $m = 2m'$ objectives, they are obtained by partitioning individuals into $m'$ blocks of size $b = \frac{n}{m'}$ and applying a bi-objective function to each block. For some individual $x \in \{0,1\}^n$ and $i \in [m']$, we define $x^i$ to be the $i$th block of $x$, that is the substring from $x_{b(i-1)+1}$ to $x_{bi}$. We define $|x^i|_1 = \sum_{j=(i-1)b+1}^{ib} x_j$ (and $|x^i|_0 = b - |x^i|_1$) to denote the number of 1-bits (and 0-bits) in the $i$th block.

### 3.2 Benchmarks

We evaluate the algorithms on four established multi-objective benchmarks.

*$m$OneMinMax ($m$*OMM*).* The objective function $m$OMM translates the well-established OneMax benchmark in a setting with $m = 2m'$ objectives for some $m' \in \mathbb{N}$. Intuitively, the bitstring is divided into $m'$ equally sized blocks that each contribute two objectives, the number of 1-bits and the number of 0-bits in that block. The bi-objective case of $m' = 1$ was proposed by [17] and later generalized to arbitrary $m'$ [29]. Let $b, m' \in \mathbb{N}$ and $n = bm'$. For all $x \in \{0,1\}^n$, define $m\text{OMM}(x) = (f_1(x), \ldots, f_m(x))$ where for all $i \in [m']$

$$f_{2i}(x) = |x^i|_1 \quad \text{and} \quad f_{2i-1}(x) = b - |x^i|_1.$$

The benchmark $m$OMM is special in the sense that each of the $S_m^{\text{OMM}} := \left(\frac{n}{m'} + 1\right)^{m'}$ possible objective values lies on the Pareto front.

*$m$CountingOnesCountingZeros ($m$*COCZ*).* The COCZ benchmark and its multi-objective variant $m$COCZ [20] are closely related to OMM and $m$OMM. However, the objectives cooperate on the first half of the bitstring and only the second half is evaluated just like for $m$OMM. Formally, let $m' \in \mathbb{N}$, $b \in 4\mathbb{N}$, $n = 2bm'$, and $m = 2m'$. Then for all $x \in \{0,1\}^n$, define $\text{mCOCZ}(x) = (f_1(x), \ldots, f_m(x))$ where for all $i \in [m']$

$$f_{2i}(x) = \sum_{j=1}^{bm'} x_j + \sum_{j=ib+1}^{ib} x_{bm'+j} \quad \text{and} \quad f_{2i-1}(x) = \sum_{j=1}^{bm'} x_j + \sum_{j=ib+1}^{ib} 1 - x_{bm'+j}.$$

Observe that $S_m^{\text{COCZ}} = (\frac{n}{2m'} + 1)^{m'}$ is the size of Pareto front for the $m$COCZ problem and also the maximum size of any set of pairwise non-dominating individuals.

$m$LeadingOnesTrailingZeros *($m$LOTZ).* The objective function $m$LOTZ is the many objective variant of the bi-objective LOTZ benchmark [20]. Intuitively, it has two objectives per block, one being the number 1-bits up to the first 0-bit and the other being the number of 0-bits behind the last 1-bit. Formally, let $b, m' \in \mathbb{N}$, $n = bm'$, and $m = 2m'$. Then for all $x \in \{0,1\}^n$, define $m\text{LOTZ}(x) = (f_1(x), \dots, f_m(x))$ where for all $i \in [m']$

$$f_{2i}(x) = \sum_{j=(i-1)b+1}^{ib} \prod_{j'=1}^{j} x_{j'} \quad \text{and} \quad f_{2i-1}(x) = \sum_{j=(i-1)b+1}^{ib} \prod_{j'=j}^{ib} 1 - x_{j'}.$$

Observe that $\bar{S}_m^{\text{LOTZ}} = S_m^{\text{OMM}} = (\frac{n}{m'}+1)^{m'}$ is the size of Pareto front for the $m$LOTZ problem. However, the size $S_m^{\text{LOTZ}}$ of the largest incomparable set is asymptotically tightly bounded by $S_m^{\text{LOTZ}} \leq (\frac{n}{m'}+1)^{2m'-1}$ [25] and thus almost quadratic in $\bar{S}_m^{\text{LOTZ}}$.

$m$OneJumpZeroJump$_k$ *($m$OJZJ$_k$).* The objective function $m\text{OJZJ}_k$ is the recently introduced many objective variant [31] of the bi-objective OJZJ benchmark [14]. It has two objectives per block, one for the number of 1-bits and one for the number of 0-bits. However, it has a fitness valley with decreasing objective value if the number of 0-bits or 1-bits is in $[k]$, where $k$ is a parameter of the benchmark. Formally, let $b, m' \in \mathbb{N}$, $n = bm'$ and $m = 2m'$. Then for all $x \in \{0,1\}^n$, define $m\text{OJZJ}_k(x) = (f_1(x), \dots, f_m(x))$ where for all $i \in [m']$

$$f_{2i}(x) = \text{Jump}_k(x^i) \quad \text{and} \quad f_{2i-1}(x) = \text{ZeroJump}_k(x^i) \quad \text{with}$$

$$\text{Jump}_k(x) = \begin{cases} |x|_1 + k, & \text{if } |x|_1 \leq b-k \text{ or } |x|_1 = b; \\ b - |x|_1, & \text{else;} \end{cases}$$

$$\text{ZeroJump}_k(x) = \begin{cases} |x|_0 + k, & \text{if } |x|_0 \leq b-k \text{ or } |x|_0 = b; \\ b - |x|_0, & \text{else.} \end{cases}$$

We assume $2 \leq k \leq \frac{n}{2m'}$. The 2-objective OJZJ benchmark with jumps of size $k$ has a Pareto front of size $n - 2k + 3$ [14]. Thus, the Pareto front of $m\text{OJZJ}_k$, which corresponds to OJZJ in $m'$ individual blocks of size $\frac{n}{m'}$, is of size $\bar{S}_m^{\text{OJZJ}} = \left(\frac{n}{m'} - 2k + 3\right)^{m'}$. As the objective value is defined by the number of 1-bits in each block, the size $S_{m,k}^{\text{OJZJ}}$ of a largest set of pairwise non-dominating individuals is upper bounded by $S_{m,k}^{\text{OJZJ}} \leq \left(\frac{n}{m'}+1\right)^{m'} = S_m^{\text{OMM}}$. Note that $S_{m,k}^{\text{OJZJ}} \approx \bar{S}_m^{\text{OJZJ}}$ for small values of $k$, which is typically assumed.

### 3.3 SEMO and GSEMO

The SEMO and GSEMO start the first generation with a single, random individual in the population. In each iteration, they uniformly at random choose an individual from the current population and mutate it to a new solution $x'$. Now they remove all solutions from the population that are dominated by $x'$ and add $x'$ to the population if and only if it is not strictly dominated by a

solution in the population, see Algorithm 1. This way, the population stores exactly one individual for each encountered objective value that is not strictly dominated by any other encountered objective value. Observe that by evaluating the objective function only once after a new individual is created and storing the value for future comparisons, the number of evaluations is exactly the number of generations (plus 1 for the initial individual). The only difference between the SEMO and the GSEMO is the mutation step. While the SEMO uniformly at random selects one bit of the parent and flips that bit to create an offspring, the GSEMO independently flips each bit of the parent with some probability $p$. Here we assume the conventional mutation rate $p = \frac{1}{n}$.

**Algorithm 1:** (Global) SEMO

**Input** : objective function $f = (f_1, \dots, f_m)$,
length of bitstrings $n$ with $\frac{n}{m} \in \mathbb{N}$

Generate $x_0 \in \{0,1\}^n$ uniformly at random and let $P_0 := \{x_0\}$
**for** $t = 1, 2, \dots$ **do**
  Select $x$ from $P_{t-1}$ uniformly at random and let $x' := \text{mutate}(x)$
  $P_t := \{p \in P_{t-1} \mid x' \not\succeq p\}$
  **if** *there is no* $p \in P_t$ *such that* $p \succ x'$ **then**
    $P_t := P_t \cup \{x'\}$

## 4 Mathematical Analyses of GSEMO

We start our contribution by giving upper bounds on the optimization time of the GSEMO on the four benchmarks. Afterward, in Sect. 5, these results are transferred to other MOEAs as well. To start our analysis, we observe that for these benchmarks the size of any set of incomparable solutions is at most $S_m^{\text{OMM}}, S_m^{\text{COCZ}}, S_m^{\text{LOTZ}}$, or $S_{m,k}^{\text{OJZJ}}$, respectively. Consequently, these are upper bounds on the population size in any iteration of the GSEMO.

### 4.1 *m*OneMinMax

If a bitstring has $a_i$ bits of value 1 in the $i$th block for all $i \in [m']$, we call it an $(a_1, a_2, \dots, a_{m'})$-bitstring. We abbreviate the notation of vectors of the Pareto front from $(n-a_1, a_1, n-a_2, a_2, \dots, n-a_{m'}, a_{m'})$ to $(a_1, a_2, \dots, a_{m'})$. Define the set $C_m$ to contain all bitstrings for which each of the $m'$ blocks consists either only of 1-bits or only of 0-bits ("corners"). Observe that $|C_m| = 2^{m'}$ and that each bitstring in $C_m$ is the unique individual of the respective objective value. For the upper bound, we separately bound the time until all individuals in $C_m$ are in the population, see Lemma 1, and then analyze how from this point on all other individuals are generated, see Lemma 2. The following theorem combines these results to bound the total optimization time.

**Theorem 1.** *Let $m' \in \mathbb{N}$ and $m = 2m'$. Consider the GSEMO optimizing $m$OMM. Let $T$ denote the number of iterations until the population matches the complete Pareto front and let $t$ be*

$$\left(\frac{\ln(2)m'+2}{\ln(n)} + 16\frac{m'^2+2\,m'}{n} + 2\right) eS_m^{\mathrm{OMM}} n\ln(n) + 1.$$

*Then $T \leq t$ with high probability and $E[T] \leq (1-\frac{1}{n})^{-2}t$.*

*Proof.* Let $t_1$ and $t_2$ be the optimization times in Lemmas 1 and 2. We have $t \geq t_1 + \lceil t_2 \rceil$ by observing $\ln(n) \geq \ln(m')$ and $\ln(n)+1 \geq \ln(n+1) \geq \ln\left(\frac{n}{m'}+1\right)$. Thus, $T \leq t$ with a high probability of at least $(1-\frac{1}{n})^2$. We employ a simple restart argument to obtain an upper bound on the expected value of $T$, analyzing the success of each sequence of $t$ iterations separately. Each sequence of $t$ iterations fails to cover the Pareto front with probability at most $1-(1-\frac{1}{n})^2$. Due to the convergence of the geometric series we have

$$E[T] \leq \sum_{i=0}^{\infty}\left(1-\left(1-\frac{1}{n}\right)^2\right)^i t = \left(1-\frac{1}{n}\right)^{-2} t. \qquad \square$$

**Lemma 1.** *Let $m' \in \mathbb{N}$ and $m = 2m'$. Consider the GSEMO optimizing $m$OMM and let $T$ denote the number of iterations until the population contains $C_m$. Then*

$$T \leq \left(\ln(2)\frac{m'}{\ln(n)} + 2\right) eS_m^{\mathrm{OMM}} n\ln(n)$$

*with probability at least $1-\frac{1}{n}$.*

*Proof.* We first prove a tail bound for the time $T_C$ until the population contains the corner bitstring $1^n$. By the symmetry of $m$OMM, this bound applies to all other elements in $C_m$ as well. Hence, applying a union bound over the tail bounds for the individual elements in $C_m$ yields a bound on the time until all elements are covered.

Let $x$ be a member of the population with maximum number of 1-bits. Let $i = n - |x|_1$ be the number of 0-bits of $x$. Then the probability of sampling an individual that has $i-1$ many 0-bits in the next iteration is at least $\frac{1}{S_m^{\mathrm{OMM}}} \cdot i \cdot \frac{1}{n} \cdot (1-\frac{1}{n})^{n-1} \geq \frac{i}{enS_m^{\mathrm{OMM}}} = p_i$, by choosing $x$ as parent, flipping any one of its 0-bits, and not flipping any other bit. Hence, $E[T_C] \leq \sum_{i=1}^{n}\frac{1}{p_i}$.

For $1 \leq i \leq n$, let $X_i$ be independent geometric random variables, each with success probability $p_i$, and let $X = \sum_{i=1}^{n} X_i$. Then $X$ stochastically dominates $T_C$, and thus a tail bound for $X$ also applies to $T_C$. By Theorem 1.10.35 in [11], a tail bound for sums of geometric random variables with harmonic success probabilities, for all $\delta \geq 0$ we have

$$\Pr[T_C \geq (1+\delta)eS_m^{\mathrm{OMM}} n\ln(n)] \leq \Pr[X \geq (1+\delta)eS_m^{\mathrm{OMM}} n\ln(n)] \leq n^{-\delta}.$$

By the symmetry of the problem and operators, the bound also holds for all other elements in $C_m$. Let $\delta = m' \log_n(2) + 1$. A union bound helps to give a tail bound on the time $T$ until the population contains all elements in $C_m$. The probability that $T \leq (m' \log_n(2) + 2) e S_m^{\mathrm{OMM}} n \ln(n)$ is at least

$$1 - |C_m| \cdot \Pr[T_C \geq (m' \log_n(2) + 2) e S_m^{\mathrm{OMM}} n \ln(n)] \geq 1 - 2^{m'} n^{-m' \log_n(2) - 1} = 1 - \tfrac{1}{n}.$$

We note that this applies for arbitrary starting configurations, as all that we assumed about the initial population was that it is non-empty. □

**Lemma 2.** *Let $m' \in \mathbb{N}$ and $m = 2m'$. Consider the GSEMO optimizing $m$OMM starting with a population that contains at least all individuals in $C_m$. Let $T$ denote the number of iterations until the population matches the complete Pareto front and let*

$$t = \max\left\{1, 8\frac{m'(m' \ln\left(\frac{n}{m'} + 1\right) + \ln(m') + \ln(n))}{n}\right\} \cdot 2e S_m^{\mathrm{OMM}} n.$$

*Then $T \leq \lceil t \rceil$ with probability at least $1 - \frac{1}{n}$.*

*Proof.* Consider any objective value $v = (a_1, a_2, \ldots, a_{m'})$ on the Pareto front. Let $c_0 \in C_m$ be any closest corner to an $(a_1, a_2, \ldots, a_{m'})$-bitstring. We bound the time until an $(a_1, a_2, \ldots, a_{m'})$-bitstring is generated by bounding the time until a marked individual $c$ becomes an $(a_1, a_2, \ldots, a_{m'})$-bitstring. Let initially $c = c_0$. Whenever the individual $c$ creates an offspring $c'$ by flipping exactly one bit and $c'$ is closer to any $(a_1, a_2, \ldots, a_{m'})$-bitstring than $c$, we update $c$ to be $c'$. We also replace $c$ by $c'$ whenever an individual $c'$ with the same objective value replaces $c$ in the population. The time until the population contains an $(a_1, a_2, \ldots, a_{m'})$-bitstring is at most the time until $c$ is an $(a_1, a_2, \ldots, a_{m'})$-bitstring. We first bound the probability that after $t$ iterations there are exactly $a_i$ bits of value 1 in the $i$th block of $c$, for any fixed $1 \leq i \leq m'$. By symmetry, suppose without loss of generality that $a_i \leq \frac{n}{2m'}$ and the $i$th block of $c_0$ to be $0^{n/m'}$. All $c$ we will encounter have between 0 and $a_i$ bits with value 1. The probability of creating an offspring of $c$ in the next iteration that flips one of the at least $\frac{n}{2m'}$ bits of value 0 in the $i$th block and no other bit is at least

$$\frac{1}{S_m^{\mathrm{OMM}}} \cdot \frac{n}{2\,m'} \cdot \frac{1}{n} \cdot \left(1 - \frac{1}{n}\right)^{n-1} \geq \frac{1}{2em' S_m^{\mathrm{OMM}}} =: p.$$

After $a_i \leq \frac{n}{2m'}$ such iterations, $c$ contains the correct number of 1-bits in the $i$th block. Thus, for the time $T_i$ until the $i$th block of $c$ is correct we have $E[T_i] \leq \frac{a_i}{p}$. For $j \in [\lceil t \rceil]$, let $X_j$ be independent random variables, each with a Bernoulli distribution with success probability $p$. Let $X = \sum_{j=1}^{\lceil t \rceil} X_j$. Then, due to stochastic domination, $\Pr[T_i > \lceil t \rceil] \leq \Pr[X < a_i] \leq \Pr[X \leq a_i]$. By observing $E[X] = p\lceil t \rceil \geq \frac{n}{m'}$ we have

$$\Pr[X \leq a_i] \leq \Pr\left[X \leq \frac{n}{2\,m'}\right] \leq \Pr\left[X \leq \frac{1}{2}E[X]\right].$$

Applying a Chernoff bound yields

$$\Pr[T_i > \lceil t \rceil] \leq \exp\left(-\frac{1}{8}E[X]\right) \quad \leq \exp\left(-\ln(m') - m'\ln\left(\frac{n}{m'}+1\right) - \ln(n)\right).$$

Using a union bound over all blocks gives that any fixed objective value on the Pareto front is not sampled in $t$ iterations with probability at most

$$\exp\left(-\ln(m') - m'\ln\left(\frac{n}{m'}+1\right) - \ln(n)\right)m' = \exp\left(-m'\ln\left(\frac{n}{m'}+1\right) - \ln(n)\right).$$

Let $E$ denote the event that after $\lceil t \rceil$ iterations there is still an objective value $(a_1, a_2, \ldots, a_{m'})$ such that the respective individual $c$ does not contain the correct number 1-bits in any block. By applying a union bound we have

$$\Pr[E] \leq \exp\left(-m'\ln\left(\frac{n}{m'}+1\right) - \ln(n)\right)S_m^{\mathrm{OMM}} = \exp(-\ln(n)) = \frac{1}{n}$$

by observing $S_m^{\mathrm{OMM}} = \exp(m'\ln(\frac{n}{m'}+1))$. □

### 4.2 *m*CountingOnesCountingZeros

Due to the similarity between $m$OMM and $m$COCZ, our previous proofs can be adapted to also work for $m$COCZ. We first show that with probability at least $1-\frac{1}{n}$ the population after $2eS_m^{\mathrm{COCZ}}n\ln(n)$ iterations contains an individual such that the cooperative, first half is maximized. From this point on, we employ the same ideas as for Theorem 1 by only considering individuals with maximum cooperative part. More details are placed in the supplementary material.[1]

**Theorem 2.** *Let $m' \in \mathbb{N}$ and $m = 2m'$. Consider the GSEMO optimizing $m$COCZ. Let $T$ denote the number of iterations until the population matches the complete Pareto front and let $t$ be*

$$\left(\frac{\ln(2)m'+2}{\ln(n)} + 16\frac{{m'}^2+2\,m'}{n} + 4\right)eS_m^{\mathrm{COCZ}}n\ln(n) + 1.$$

*Then $T \leq t$ with high probability and $E[T] \leq (1-\frac{1}{n})^{-3}t$.*

Comparing the bounds for $m$OMM and $m$COCZ on bitstrings of the same length, we note that the bound for $m$COCZ is smaller than the one on $m$OMM as $S_m^{\mathrm{COCZ}} \approx 2^{-m/2}S_m^{\mathrm{OMM}}$ .

### 4.3 *m*LeadingOnesTrailingZeros

In contrast to $m$OMM and $m$COCZ, for $m$LOTZ the probability to transform a solution of a certain objective value into one where one of the values changed by 1 is less depending on the objective value itself. In fact, this probability is at least

[1] Supplementary material is available at https://arxiv.org/abs/2404.12746.

$\frac{1}{eS_m^{\mathrm{LOTZ}}n}$ for all objective values. Employing a similar strategy as in the proof of Lemma 2, but starting from the initial individual gives the following Theorem 3. This strategy is not compromised by the fact that some intermediate solutions might vanish from the population when strictly dominated. In that case, the dominating solutions actually are at least as close to a desired solution as the dominated one. Details are given in the supplementary material.

**Theorem 3.** *Let $m' \in \mathbb{N}$ and $m = 2m'$. Consider the GSEMO optimizing $m$LOTZ. Let $T$ denote the number of iterations until the population matches the complete Pareto front and let*

$$t = \max\left\{1, \frac{4m'^2 \ln\left(\frac{n}{m'}+1\right) + 8\,m' \ln(n)}{n}\right\} 2eS_m^{\mathrm{LOTZ}} \frac{n^2}{m'}.$$

*Then $T \leq \lceil t \rceil$ with high probability and $E[T] \leq (1-\frac{1}{n})^{-1}\lceil t \rceil$.*

While, unlike our other bounds for the GSEMO, this result does not improve over the existing $O(n^{2m'-1})$ bound [20], we note that our bound applies to all choices for the numbers of objectives while the previous one assumed it to be constant.

### 4.4 $m$OneJumpZeroJump$_k$

We define the set $K_{m,k} = \{(a_1, \ldots, a_{m'}) \mid a_i \in \{k, \frac{n}{m'} - k\} \text{ for all } i \in [m']\}$ to contain all objective values of individuals that in each block have either exactly $k$ bits of value 0 or exactly $k$ bits of value 1. Further, we define the set $C_{m,k} = \{(a_1, \ldots, a_{m'}) \mid a_i \in \{0, k, \frac{n}{m'} - k, \frac{n}{m'}\} \text{ for all } i \in [m']\}$ to contain all objective values of individuals that in each block have either only 1-bits, only 0-bits, exactly $k$ bits of value 0, or exactly $k$ bits of value 1.

For the $m$OJZJ$_k$ benchmark, we separately consider three phases: the time until $K_{m,k}$ is covered, the time until $C_m$ is covered, and the time until the remaining Pareto front is covered. While the first and third phase roughly compare to Lemmas 1 and 2, the second phase dominates the running time. There, progress is made by jumping over the valley of low fitness, which for any fixed block takes time in $eS_{m,k}^{\mathrm{OJZJ}}n^k$ as the only way is to simultaneously flip the $k$ bits. The bounds we obtain for the $m$OJZJ$_k$ benchmark are only applicable if $m' \geq 2$. For the case $m' = 1$, we thus refer to previous results in the literature, which show that the expected number of iterations until the GSEMO solves 2OJZJ$_k$ is at most $eS_{2,k}^{\mathrm{OJZJ}}(\frac{3}{2}n^k + 2n\ln(\lceil \frac{n}{2} \rceil) + 3)$ [30]. Details on the proofs are placed in the supplementary material.

**Theorem 4.** *Let $m' \in \mathbb{N}_{\geq 2}$ and $m = 2m'$. Consider the GSEMO optimizing $m$OJZJ$_k$. Let $T$ denote the number of iterations until the population matches the complete Pareto front and let*

$$t = \left(\frac{\ln(4)m' + \ln(n)}{\ln(m')} + 1\right) 3e\ln(m') S_{m,k}^{\mathrm{OJZJ}} n^k.$$

*Then $T \leq t$ with high probability . Further,*

$$E[T] \leq \left(1 - \frac{1}{m'}\right)^{-1} \left(\frac{\ln(4)m'}{\ln(m')} + 2\right) 3e \ln(m') S^{\mathrm{OJZJ}}_{m,k} n^k.$$

## 5 Runtime Results for the SEMO, SMS-EMOA, and NSGA-III

We started our mathematical runtime analysis of many-objective MOEAs with an analysis of the GSEMO, the most prominent MOEA in theoretical works. We are very optimistic that our general methods apply to many other MOEAs as well. We discuss sufficient conditions to extend our results to other MOEAs and demonstrate this on the SEMO, an algorithm prominent in theory, as well as the SMS-EMOA and NSGA-III, algorithms often used in practical applications.

Note that all our proofs for upper bounds for the GSEMO only rely on three properties of the algorithm:

1. Once a solution $x$ is generated, all future populations contain a solution $y$ such that $y \succeq x$.
2. The chance to select an individual from the population for mutation is at least $\frac{1}{S}$, where $S$ is typically an upper bound on the size of sets of incomparable solutions. The bounds on the runtime will include a factor of $S$.
3. The employed mutation operator is bitwise mutation with a chance of $\frac{1}{n}$ to flip a bit. Other mutation operators are possible but might affect the bounds and ability to solve some benchmarks at all, as we discuss for the SEMO.

While the latter two properties enable progress to be made, the first one is responsible for not loosing already made progress, that is, not loosing any desired solutions and intermediate solutions on the path to a desired solution.

### 5.1 SEMO

The only difference between the SEMO and GSEMO is the mutation step. While the GSEMO flips each bit independently with probability $\frac{1}{n}$, the SEMO uniformly at random selects any bit and flips it. This makes it impossible for the SEMO to solve $m\mathrm{OJZJ}_k$, as argued for the bi-objective case [14]. By the same reasoning, the SEMO cannot solve $m\mathrm{OJZJ}_k$ for any $m$. Nevertheless, the SEMO is able to solve all other benchmarks discussed in this work. In fact, we note that in all proofs, except for the ones concerning $m\mathrm{OJZJ}_k$, our arguments exclusively build on improvements by flipping exactly one bit. Thus, not only do all proven results immediately transfer, but actually improve by a factor $e$, that previously accounted for the probability of no other bit than the desired one flipping.

**Theorem 5.** *Consider the SMS-EMOA optimizing $m$OMM, $m$COCZ, or $m$LOTZ. Then the respective bounds for the GSEMO as given by Theorems 1, 2, and 3, divided by e, also hold for the SEMO.*

**Algorithm 2:** SMS-EMOA with population size $\mu$

**Input :** objective function $f = (f_1, \dots, f_m)$, length of bitstrings $n$ with $\frac{n}{m} \in \mathbb{N}$

1 Let $P_0 \coloneqq \{x_1, \dots, x_\mu\}$, where each $x_i \in \{0,1\}^n$ is generated independently and uniformly at random

2 **for** $t = 1, 2, \dots$ **do**

3 Select $x$ from $P_{t-1}$ uniformly at random and let $x' \coloneqq \text{mutate}(x)$

4 Divide $R_t \coloneqq P_t \cup \{x'\}$ into fronts $F_1, \dots, F_{i^*}$, by fast-non-dominated-sort() [9].

5 Pick $z' \in \arg\min_{z \in F_{i^*}} \Delta_r(z, F_{i^*})$ uniformly at random and let $P_t \coloneqq R_t \setminus z'$

### 5.2 SMS-EMOA

The SMS-EMOA works with a population of fixed size $\mu$. Similar to the (G)SEMO, it produces one offspring solution in each generation. We consider the SMS-EMOA using bit-wise mutation just like employed by the GSEMO. While the (G)SEMO only relies on the (strict) domination relation to select the surviving solutions for the next generation, the SMS-EMOA sorts solutions into fronts $F_1, \dots, F_{i^*}$, where each front contains all pairwise not strictly dominating solutions that are not yet represented in an earlier front. Then one element of $F_{i^*}$ is removed to reduce the population size back to $\mu$, namely one with smallest hypervolume contribution. For some set $S$ and reference point $r$, the hypervolume of $S$ is $\text{HV}_r(S) = \mathcal{L}\big(\bigcup_{u \in S}\{h \in \mathbb{R}^m \mid r \leq h \leq f(u)\}\big)$, where $\mathcal{L}$ is the Lebesgue measure. The hypervolume contribution of an individual of an individual $x \in F$ is $\Delta_r(x, F) = \text{HV}_r(F) - \text{HV}_r(F \setminus \{x\})$. Since we only regard maximization problems with non-negative objective values, as common, we use the reference point $r = (-1, \dots, -1)$. Algorithm 2 states the SMS-EMOA in pseudocode.

Our results for the GSEMO transfer to the SMS-EMOA if $\mu$ is at least as the largest set of incomparable solutions. The central observation is that then, once the population of the SMS-EMOA contains an individual $x$, all future generations will contain at least one individual $x'$ such that $f(x') \succeq f(x)$ [31, Lemma 4], yielding the first of the three above mentioned properties. For the second and third one, we note that the chance to select an individual for mutation is simply $\frac{1}{\mu}$ and bitwise mutation is applied. This yields the results as for the GSEMO though $\mu$ replaces the factor $S$ in all runtime bounds.

**Theorem 6.** *Consider the SMS-EMOA optimizing one of* $m$OMM, $m$COCZ, $m$LOTZ, *or* $m\text{OJZJ}_k$, *let* $S$ *denote the size of the largest incomparable set of solutions, and let the population size be* $\mu \geq S$. *Then the respective bounds for the GSEMO given by Theorems 1-4, multiplied by* $\frac{\mu}{S}$, *also hold for the SMS-EMOA.*

### 5.3 NSGA-II and NSGA-III

We briefly summarize the NSGA-II and NSGA-III and refer to [26] for details. The algorithms work with a population of fixed size $\mu$. Each iteration, every

individual produces an offspring, here we assume by bitwise mutation. The combined population of size $2\mu$ is sorted into ranks, each containing all solutions only dominated by those with lower rank. The new generation of size $\mu$ is selected by prioritizing lower rank solutions. As a tiebreaker, the NSGA-II employs crowding distance while the NSGA-III uses reference points in the solution space.

As discussed in the introduction, the NSGA-II struggles for $m > 2$ objectives. Our proofs do not transfer as, even with a large population, the NSGA-II can lose non-dominated objective values as proven for $m$OMM in [29].

In contrast, the NSGA-III preserves non-dominated solutions for reasonable choices of the population size and reference points [25]. Each individual is mutated with probability $\frac{1}{S} = 1$ in each generation, eliminating the factor $S$ from the running time. However, every generation requires $\mu \geq S$ fitness evaluations.

**Theorem 7.** *Consider the NSGA-III optimizing* $m$OMM, $m$COCZ,$m$LOTZ, *or* $m\text{OJZJ}_k$, *let* $S$ *denote the size of the largest incomparable set of solutions and* $f_{\max}$ *denote the largest fitness value over all solutions and objectives. Assume* $\mu \geq S$ *and the NSGA-III to employ a set of reference points* $\mathcal{R}_p$ *as defined in [25] with* $p \geq 2m^{3/2} f_{\max}$. *Then the respective bounds for the GSEMO as given by Theorems 1-4 multiplied by* $\frac{\mu}{S}$, *also hold for the NSGA-III.*

## 6 Conclusion

In this work, we revisited the problem of proving performance guarantees for MOEAs dealing with more than two objectives. In the first major progress after the initial work on this question twenty years ago [20], we proved runtime guarantees for three classic algorithms on four classic benchmarks that are all (except for $m$LOTZ) linear in the size of the Pareto front (apart from small factors polynomial in $n$ and $m$), in contrast to the previous bounds, which were all quadratic in the size of the Pareto front. Our results thus suggest that MOEAs cope much better with many-objective problems than known. In fact, our work hints at that the performance loss observed with increasing number of objectives in experiments is caused by the increasing size of the Pareto front rather than by particular algorithmic difficulties of many-objective optimization.

An obvious next step in this research direction would be to advance from synthetic benchmarks to classic combinatorial optimization problems, e.g., the multi-objective minimum spanning tree problem regarded so far only for two objectives [6,10,23]. A technical challenge would be to determine the runtimes for the many-objective LOTZ problem, where we do not have the obvious lower bound of the Pareto front size. We note, though, that lower bounds for LOTZ are already very difficult in the bi-objective setting [12].

**Acknowledgments.** Simon Wietheger acknowledges the support by the Austrian Science Fund (FWF, START Project Y1329). Benjamin Doerr was supported by a public grant as part of the Investissements d'avenir project, reference ANR-11-LABX-0056-LMH, LabEx LMH.

**Disclosure of Interests.** The authors have no competing interests to declare.

## References


1. Auger, A., Doerr, B. (eds.): Theory of Randomized Search Heuristics. World Scientific Publishing (2011)
2. Beume, N., Naujoks, B., Emmerich, M.: SMS-EMOA: Multiobjective selection based on dominated hypervolume. Eur. J. Oper. Res. **181**, 1653–1669 (2007)
3. Bian, C., Qian, C., Tang, K.: A general approach to running time analysis of multi-objective evolutionary algorithms. In: International Joint Conference on Artificial Intelligence, IJCAI 2018, pp. 1405–1411. IJCAI (2018)
4. Bian, C., Zhou, Y., Li, M., Qian, C.: Stochastic population update can provably be helpful in multi-objective evolutionary algorithms. In: International Joint Conference on Artificial Intelligence, IJCAI 2023, pp. 5513–5521 (2023). ijcai.org
5. Brockhoff, D., Friedrich, T., Neumann, F.: Analyzing hypervolume indicator based algorithms. In: Rudolph, G., Jansen, T., Beume, N., Lucas, S., Poloni, C. (eds.) PPSN 2008. LNCS, vol. 5199, pp. 651–660. Springer, Heidelberg (2008). https://doi.org/10.1007/978-3-540-87700-4_65
6. Cerf, S., Doerr, B., Hebras, B., Kahane, J., Wietheger, S.: The first proven performance guarantees for the Non-Dominated Sorting Genetic Algorithm II (NSGA-II) on a combinatorial optimization problem. In: International Joint Conference on Artificial Intelligence, IJCAI 2023, pp. 5522–5530 (2023). ijcai.org
7. Coello, C.A.C., Lamont, G.B., van Veldhuizen, D.A.: Evolutionary Algorithms for Solving Multi-Objective Problems. Springer, 2nd edn. (2007). https://doi.org/10.1007/978-0-387-36797-2
8. Deb, K., Jain, H.: An evolutionary many-objective optimization algorithm using reference-point-based nondominated sorting approach, part I: solving problems with box constraints. IEEE Trans. Evol. Comput. **18**, 577–601 (2014)
9. Deb, K., Pratap, A., Agarwal, S., Meyarivan, T.: A fast and elitist multiobjective genetic algorithm: NSGA-II. IEEE Trans. Evol. Comput. **6**, 182–197 (2002)
10. Do, A.V., Neumann, A., Neumann, F., Sutton, A.M.: Rigorous runtime analysis of MOEA/D for solving multi-objective minimum weight base problems. In: Advances in Neural Information Processing Systems, NeurIPS 2023 (2023)
11. Doerr, B.: Probabilistic tools for the analysis of randomized optimization heuristics. In: Theory of Evolutionary Computation. NCS, pp. 1–87. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-29414-4_1
12. Doerr, B., Kodric, B., Voigt, M.: Lower bounds for the runtime of a global multi-objective evolutionary algorithm. In: Congress on Evolutionary Computation, CEC 2013, pp. 432–439. IEEE (2013)
13. Doerr, B., Neumann, F. (eds.): Theory of Evolutionary Computation—Recent Developments in Discrete Optimization. Springer (2020). http://www.lix.polytechnique.fr/Labo/Benjamin.Doerr/doerr_neumann_book.html
14. Doerr, B., Zheng, W.: Theoretical analyses of multi-objective evolutionary algorithms on multi-modal objectives. In: Conference on Artificial Intelligence, AAAI 2021, pp. 12293–12301. AAAI Press (2021)
15. Fortin, F., Parizeau, M.: Revisiting the NSGA-II crowding-distance computation. In: Genetic and Evolutionary Computation Conference, GECCO 2013, pp. 623–630. ACM (2013)
16. Giel, O.: Expected runtimes of a simple multi-objective evolutionary algorithm. In: Congress on Evolutionary Computation, CEC 2003, pp. 1918–1925. IEEE (2003)

17. Giel, O., Lehre, P.K.: On the effect of populations in evolutionary multi-objective optimisation. Evol. Comput. **18**, 335–356 (2010)
18. Huang, Z., Zhou, Y., Luo, C., Lin, Q.: A runtime analysis of typical decomposition approaches in MOEA/D framework for many-objective optimization problems. In: International Joint Conference on Artificial Intelligence, IJCAI 2021, pp. 1682–1688 (2021)
19. Jansen, T.: Analyzing Evolutionary Algorithms – The Computer Science Perspective. Springer (2013). https://doi.org/10.1007/978-3-642-17339-4
20. Laumanns, M., Thiele, L., Zitzler, E.: Running time analysis of multiobjective evolutionary algorithms on pseudo-Boolean functions. IEEE Trans. Evol. Comput. **8**, 170–182 (2004)
21. Laumanns, M., Thiele, L., Zitzler, E., Welzl, E., Deb, K.: Running time analysis of multi-objective evolutionary algorithms on a simple discrete optimization problem. In: Guervós, J.J.M., Adamidis, P., Beyer, H.-G., Schwefel, H.-P., Fernández-Villacañas, J.-L. (eds.) PPSN 2002. LNCS, vol. 2439, pp. 44–53. Springer, Heidelberg (2002). https://doi.org/10.1007/3-540-45712-7_5
22. Li, Y.L., Zhou, Y.R., Zhan, Z.H., Zhang, J.: A primary theoretical study on decomposition-based multiobjective evolutionary algorithms. IEEE Trans. Evol. Comput. **20**, 563–576 (2016)
23. Neumann, F.: Expected runtimes of a simple evolutionary algorithm for the multi-objective minimum spanning tree problem. Eur. J. Oper. Res. **181**, 1620–1629 (2007)
24. Neumann, F., Witt, C.: Bioinspired Computation in Combinatorial Optimization – Algorithms and Their Computational Complexity. Springer (2010). https://doi.org/10.1007/978-3-642-16544-3
25. Opris, A., Dang, D.C., Neumann, F., Sudholt, D.: Runtime analyses of NSGA-III on many-objective problems. In: Genetic and Evolutionary Computation Conference, GECCO 2024. ACM (2024). to appear
26. Wietheger, S., Doerr, B.: A mathematical runtime analysis of the Non-dominated Sorting Genetic Algorithm III (NSGA-III). In: International Joint Conference on Artificial Intelligence, IJCAI 2023, pp. 5657–5665 (2023). ijcai.org
27. Zhang, Q., Li, H.: MOEA/D: a multiobjective evolutionary algorithm based on decomposition. IEEE Trans. Evol. Comput. **11**, 712–731 (2007)
28. Zheng, W., Doerr, B.: Mathematical runtime analysis for the non-dominated sorting genetic algorithm II (NSGA-II). Artif. Intell. **325**, 104016 (2023)
29. Zheng, W., Doerr, B.: Runtime analysis for the NSGA-II: proving, quantifying, and explaining the inefficiency for many objectives. IEEE Trans. Evol. Comput. (2023). in press, https://doi.org/10.1109/TEVC.2023.3320278
30. Zheng, W., Doerr, B.: Theoretical analyses of multiobjective evolutionary algorithms on multimodal objectives. Evol. Comput. **31**, 337–373 (2023)
31. Zheng, W., Doerr, B.: Runtime analysis of the SMS-EMOA for many-objective optimization. In: Conference on Artificial Intelligence, AAAI 2024, pp. 20874–20882. AAAI Press (2024)
32. Zheng, W., Liu, Y., Doerr, B.: A first mathematical runtime analysis of the Non-Dominated Sorting Genetic Algorithm II (NSGA-II). In: Conference on Artificial Intelligence, AAAI 2022, pp. 10408–10416. AAAI Press (2022)
33. Zhou, A., Qu, B.Y., Li, H., Zhao, S.Z., Suganthan, P.N., Zhang, Q.: Multiobjective evolutionary algorithms: a survey of the state of the art. Swarm Evol. Comput. **1**, 32–49 (2011)
34. Zhou, Z.H., Yu, Y., Qian, C.: Evolutionary Learning: Advances in Theories and Algorithms. Springer (2019). https://doi.org/10.1007/978-981-13-5956-9

# Multi-objective Random Bit Climbers with Weighted Permutation on Large Scale Binary MNK-Landscapes

Felipe Honjo Ide(✉), Hernan Aguirre, and Kiyoshi Tanaka

Shinshu University, Nagano, Japan
{22hs204k,ahernan,ktanaka}@shinshu-u.ac.jp

**Abstract.** Multi-Objective Evolutionary Algorithms have proven to be very effective when solving Multi-Objective Optimization Problems. However, their performance decreases significantly when solving large scale problems, which can have hundreds or thousands of variables. Although several algorithms have been proposed to tackle this problem in the recent years, most of them are designed for continuous problems, and only a few focus on binary ones. In this paper, we propose a modification to multi-objective random one-bit climbers that achieves better performance in large scale binary problems by learning the trend of the values of the decision variables from previously found solutions and applying that information to decide which ones to focus on when executing the bit climb. We present the implemented algorithm, compare its performance to other well known evolutionary algorithms and study some of its properties.

**Keywords:** Multi-objective optimization · Multi-objective bit climbers · Evolutionary algorithms · Large scale binary problems · Decision space reduction · MNK-Landscapes

## 1 Introduction

In real-world applications, optimization problems often have multiple concurring objectives that need to be optimized simultaneously, in which the improvement of one objective is often accompanied by the detriment of others [11]. These problems, known as Multi-Objective Optimization Problems (MOPs), can often be solved by Multi-Objective Evolutionary Algorithms (MOEAs). MOEAs use methods analogous to natural evolution, such as selection, mutation and crossover, to iterate over candidate solutions, until a satisfactory set of solutions are found [10,11].

Various types of MOEAs have been proposed following four main approaches to implement selection. Namely, Pareto dominance [10,11], decomposition [16,28], performance indicators [7,9,31], and relaxations of Pareto dominance [1,13,18]. These algorithms have been applied successfully in a broad range of


M. Affenzeller et al. (Eds.): PPSN 2024, LNCS 15151, pp. 169–185, 2024.
https://doi.org/10.1007/978-3-031-70085-9_11

application domains. However, despite all the progress we have seen in algorithm development, MOEAs require further improvements to perform an efficient optimization on problems of increased difficulty induced by variable interactions [4], large-scale search spaces [19], many objective functions [5,25], and various shapes of the Pareto optimal front [27].

In this work, our interest is in improving MOEAs for large-scale multi-objective optimization problems (LSMOPs), where usually the number of variables is in the order of hundreds or thousands. LSMOPs have proven to be very challenging for traditional MOEAs to solve, whether the decision space is continuous or discrete [23].

Several MOEAs for LSMOPs have been proposed in recent years, tackling this challenge with different approaches [15,23]. The methods used to solve them can be classified in two major categories. Variable grouping splits the solutions, either randomly or heuristically, into multiple parts and optimizes them separately, as used in CCGDE3 [6], LMEA [29], and others. Decision space reduction reduces the dimension of the problems by using a transformation function that converts the original problem into a smaller representation, such as WOF [30], or by using some techniques such as unsupervised learning to decrease the number of decision variables, such as MOEA/PSL [22]. In addition, other novel approaches have been proposed, such as DGEA [14], which uses an adaptive offspring generation method to create superior solutions, and SparseEA [21], which guides the operators based on the non-dominated scores of variables. Most of these MOEAs developed for LSMOPs focus on continuous search spaces and only a few, such as LMEA, SparseEA, and MOEA/PSL, have been applied with relative success to discrete binary search spaces. Further research is required to understand better the principles that lead to an efficient search on binary LSMOPs and the development of effective algorithms.

From this standpoint, in this paper, we focus on bi-objective binary optimization and present a Multi-Objective Random One-Bit Climber with a Weighted Permutation (moRBC-WP), aiming to improve the performance of a conventional moRBC when solving LSMOPs. Conventional moRBCs have proven to be very effective MOEAs for solving MOPs with up to 100 variables [3]. The modified moRBC-WP aims to achieve better performance by learning the trend of the value of each decision variable from previously found solutions and applying this information to selectively climb the bits estimated to be the most impactful to finding better solutions.

The remainder of this paper is organized as follows. Section 2 introduces the basic concepts of MOPs and presents MNK-Landscapes, the benchmark problems that we use in our study. Then, Sect. 3 explains moRBC and its shortcomings when solving LSMOPs. Next, Sect. 4 introduces the proposed algorithm, moRBC-WP, detailing the changes done to moRBC. Section 5 shows the experimental results and discussions related to the performance comparison with other well known MOEAs, NSGA-II [12], MOEA/D [28], SparseEA and moRBC, takes a deeper look at some of moRBC-WP's properties, and studies the change in

performance of moRBC-WP when varying its added parameter. Finally, Sect. 6 presents the conclusions of our study and some ideas for future works.

## 2 MNK-Landscapes

Binary MOPs can be mathematically represented as (1):

$$\begin{aligned} &\text{maximize} \quad \mathbf{f} = (f_1(\mathbf{x}), f_2(\mathbf{x}), \ldots, f_M(\mathbf{x})) \\ &\text{subject to} \quad \mathbf{x} = (x_1, x_2, \ldots, x_N),\ x_i \in \{0, 1\}\ , \end{aligned} \tag{1}$$

where $M$ functions $f_i$ need to be simultaneously optimized, and the solutions are represented by an array of binary values of size $N$. A popular binary MOP that is used as benchmark problem is the MNK-Landscape, which is a multi-objective version of an NK-Landscape.

NK-Landscapes are mathematical models with flexible number of variables and landscape ruggedness, configurable via their input parameters $N$ and $K$, respectively [17]. A multi-objective version was then introduced with the addition of a third input parameter, $M$, that details the number of objectives [2]. These MNK-Landscapes can be mathematically represented as (2):

$$f_i(x) = \frac{1}{N} \sum_{j=1}^{N} f_{i,j}(x_j, \underbrace{z_1^{(i,j)}, z_2^{(i,j)}, \ldots, z_{K_i}^{(i,j)}}_{K_i \text{ bits interacting with } x_j}), \quad i = 1, 2, \ldots, M\ , \tag{2}$$

where $f_{i,j} : B^{K_i+1} \rightarrow \mathbb{R}$ gives the fitness contribution of bit $x_j$ to $f_i(\cdot)$, and $z_1^{(i,j)}, \ldots, z_{K_i}^{(i,j)}$ are the $K_i$ bits interacting with bit $x_j$ in the string $\mathbf{x}$. This means that the fitness value of the bit $x_j$ is depending not only on its own value, but also on the other $K$ bits that it interacts with. A higher $K$ creates a more rugged landscape, which makes the problem more difficult to solve [2,20]. Note that a different model of epistatic interactions among variables is randomly specified for each fitness function.

We have chosen to use MNK-Landscapes as the benchmark problem for our experiments since its flexibility allow us to create large scale problems by setting $N$ to large values, as well as to control the degree of interaction between variables by adjusting the epistatic interactions $K$.

## 3 moRBC

The Multi-Objective Random One-Bit Climber (moRBC) is an MOEA that has proven to be very effective at solving MOPs [3]. moRBC is a (1+1) algorithm that creates one offspring from one parent solution at each iteration by applying a 1-bit mutation operator according to a permutation $\boldsymbol{PO}$ that determines the exploration order of the decision variables. moRBC decides which solution

survives among the parent and offspring based on their Pareto dominance[1]. The algorithm starts its execution with a randomly created parent solution and a randomly generated permutation order $\boldsymbol{PO}$ of size $N$. Then, a child is created by flipping the first variable given by $\boldsymbol{PO}$. If the child dominates the parent, the parent is deleted, the child replaces the parent (the algorithm climbs to a better solution), and exploration continues with the new parent. If the parent dominates the child, then the child is deleted and the exploration continues with the current parent. If they are non-dominated, the child is tagged as not-climbed, added to an archive of non-dominated solutions, and the search continues with the current parent. Each time an offspring is created, the variable subject to mutation is given by the next element in $\boldsymbol{PO}$. When the algorithm has searched through all the decision variables of the same parent, it means that it has found a dominance 1-bit local optima solution, which is tagged as climbed, saved to the non-dominated archive, and the search is restarted. The restart is done by selecting a new solution as a parent from the not-climbed non-dominated solutions in the archive and randomly generating a new $\boldsymbol{PO}$. If no solutions are available, then it is restarted with a randomly generated solution. The archive is kept to a maximum size by non-dominated sorting and crowding distance [11]. For more details regarding moRBC, in-depth analysis on its performance and comparison with other MOEAs, refer to [3].

To illustrate the performance of moRBC, we compare it to other well-known algorithms, NSGA-II and MOEA/D when solving MNK-Landscapes with $M = 2$ objectives, $N = \{100, 200, \ldots, 1000\}$ variables and $K = 5$ epistatic interactions. Figure 2a shows the final Hypervolume (HV) after 2,000,000 fitness evaluations of NSGA-II, MOEA/D and moRBC, among other MOEAs studied in this paper. The parameters used in this experiment are detailed in Sect. 5.1. We can see that moRBC performs significantly better than the other two MOEAs in problems with $N > 300$, but its performance rapidly decreases with higher $N$, showing its limitations when solving LSMOPs.

## 4 MoRBC-WP

Here, we present the developed algorithm, moRBC-WP. It introduces two major changes to the original moRBC. First, it learns the trend of the values of previously found solutions by updating a weight array every time a new solution is created. Second, these weights are use to decide which bits should be flipped or skipped by the algorithm during the climb. Since the developed algorithm assigns weights that are used alongside the original moRBC's permutation order, we name it "moRBC-Weighted Permutation", or moRBC-WP. The overall framework of the algorithm is shown in Algorithm 1, and detailed as follows.

[1] Given two solutions $s_1$ and $s_2$ and assuming a maximization problem, $s_1$ is said to Pareto dominate $s_2$, denoted $s_1 \succ s_2$, if and only if $\forall f_i(s_1) \geq f_i(s_2)$ and $\exists f_j(s_1) > f_j(s_2)$, where $i$ and $j \in \{1, 2, \ldots, M\}$. If neither $s_1 \succ s_2$ nor $s_2 \succ s_1$, they are said to be non-dominated between each other.

**Algorithm 1:** General framework of moRBC-WP.

```
   Data: N, δ
 1 i ← 0; // Index of the elements in PO
 2 steps ← 0; // Counts the variables visited (flipped or skipped)
 3 archive ← ∅; // Initializes an empty archive
 4 parent ← New random parent;
 5 PO ← Create new permutation order randomly;
 6 W ← Initialize weights at their default value of 0.5;
 7 while termination criteria not fulfilled do
 8     flip ← FlipDecision(parent, W, PO[i]); // Fig. 1a and Algorithm 2
 9     if flip = TRUE then
10         child ← mutate(parent, PO[i]);
11         if child ≻ parent then // Child dominates parent
12             parent ← child;
13             steps ← 0;
14             W ← UpdateWeights(δ, child, W, N); // Algorithm 3
15         else if parent ≻ child then// Parent dominates child
16             delete child;
17             W ← UpdateWeights(δ, parent, W, N); // Algorithm 3
18         else // Child and parent are non-dominated
19             archive ← UpdateArchive(child);
20             W ← UpdateWeights(δ/2, child, W, N); // Algorithm 3
21     steps + +;
22     i ← (i + 1)%N; // Circularly iterate to the next position in PO
23     W ← DecayWeights(W, N);
24     if steps = N then // All bits in the parent have been visited
25         archive ← UpdateArchive(parent);
26         parent ← restart(); // Restart strategy from moRBC
27         i ← 0;
28         steps ← 0;
29         PO ← Create new permutation order randomly;
30         W ← Reset weights to 0.5;
31 return archive;
```

To learn from the previously found solutions and apply that information to decide which bits to climb, we add a new array $\boldsymbol{W}$ of size $N$ (the number of variables of the problem being solved), that stores weights corresponding to each decision variable of the solutions. They are real numbers that start at a default value of 0.5 and can vary between 0 and 1. The $i$-th weight $\boldsymbol{W}[i]$ defines the tendency of the value of the $i$-th decision variable, where a weight closer to 0 or 1 indicates a higher chance for that variable value to be 0 or 1, respectively.

moRBC-WP initialization is the same as moRBC, creating a starting random parent and a randomized permutation order (lines 1 to 5).

On a given $i$-th iteration, moRBC-WP decides whether to flip or skip the $\boldsymbol{PO}[i]$-th bit based on $\boldsymbol{W}$ (line 8), as described in Fig. 1a, Algorithm 2 and as follows. A random real number *rand*, from 0 to 1, is generated and compared to

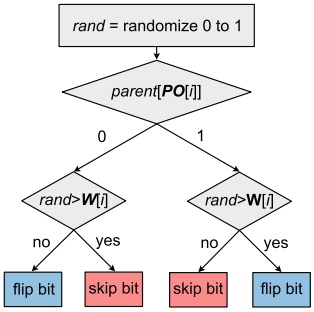
**Algorithm 2:** *FlipDecision* ($parent$, $\boldsymbol{W}$, $\boldsymbol{PO}[i]$)

```
   Data: parent, W, PO, i
1  rand ← random(0, 1);
2  if parent[PO[i]] = 0 then
3  |  if rand > W[PO[i]] then
4  |  |  return FALSE;
5  |  else
6  |  |  return TRUE;
7  else
8  |  if rand > W[PO[i]] then
9  |  |  return TRUE;
10 |  else
11 |  |  return FALSE;
```

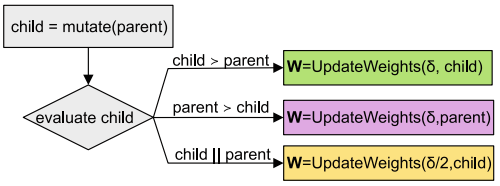
**Algorithm 3:** *UpdateWeights* ($w$, $\boldsymbol{x}$, $\boldsymbol{W}$, $N$)

```
   Data: w, x[N], W[N], N
1  for i ← 0 to N do
2  |  if x[i] = 0 then
3  |  |  if W[i] ≥ w then
4  |  |  |  W[i] ← W[i] − w;
5  |  |  else
6  |  |  |  W[i] ← 0;
7  |  else
8  |  |  if W[i] ≤ w then
9  |  |  |  W[i] ← W[i] + w;
10 |  |  else
11 |  |  |  W[i] ← 1;
12 return W;
```

the corresponding weight of that bit, $\boldsymbol{W}[\boldsymbol{PO}[i]]$. If the current bit is 0 and $rand > \boldsymbol{W}[\boldsymbol{PO}[i]]$, then the algorithm skips that bit, otherwise it flips it. Analogously, if the bit is 1 and $rand > \boldsymbol{W}[\boldsymbol{PO}[i]]$, then it will flip it, or else it will be skipped.




(a) Whether to either skip or flip the $i$-th bit, given $\boldsymbol{W}[N]$ and $rand$.




(b) The amount of weight added based on Pareto dominance between the child and the parent.

**Fig. 1.** Flowchart for bit climbs and weight updates.

Then, if the bit is flipped, a child is generated (line 10) and its Pareto dominance is compared with its parent to select the surviving individual, as done in moRBC (lines 11-13, 15-16 and 18-19). In addition, $\boldsymbol{W}$ is updated accordingly, as illustrated in Fig. 1b. If the child dominates the parent, the weight is updated by a value $\delta$ in relation to the decision variables of the child (line 14), if it is dominated, it is updated by a value $\delta$ in relation to the parent (line 17), and if it is non-dominated, then it is updated by a value $\frac{\delta}{2}$ in relation to the child (line 20). Note that, at a given iteration, only one bit is flipped, but all weights

are updated. If the decision variable is 0, then its corresponding weight will be decremented by a value $w$ ($\delta$ or $\frac{\delta}{2}$), or else it will be incremented by that value. In addition, moRBC-WP ensures that the weights stay within their range of 0 to 1 by truncating them if they would go over these values. A more detailed descriptions is shown in Algorithm 3.

Since constantly adding or subtracting values to the weights would ultimately make their values reach their lower and upper limits of 0 or 1, we also add a decay factor, $0.1 \times \delta$, which makes all weights decay towards the default value of 0.5 in every fitness evaluation loop (decrementing if it is higher than 0.5, or incrementing if it is lower than 0.5) (line 23). This ensures that the algorithm does not get stuck in local optima and can move to higher ranked solutions, as past weight values are forgotten over time.

When all variables of the parent have been visited following $\boldsymbol{PO}$ (line 24), moRBC-WP updates the archive, restarts the search and creates a new $\boldsymbol{PO}$ as done in morBC (lines 25-29). In addition, it resets $\boldsymbol{W}$ to its default values of 0.5 (line 30).

## 5 Experiments, Results and Discussions

### 5.1 Experimental Setup

To study and compare the performance of the proposed moRBC-WP, we also experiment with four other algorithms. Namely, NSGA-II [12], MOEA/D [28], SparseEA [21], and moRBC [3]. SparseEA is an MOEA proposed for solving sparse LSMOPs and is one of the few MOEAs in the literature that the authors explicitly claim it can be used to solve binary LSMOPs. Although MNK-Landscapes are not sparse problems, we use SparseEA to gain a better understanding of the proposed method and of the algorithm itself. It is implemented using the selection framework of NSGA-II, in which their main differences lie in the way SparseEA creates its initial population and the mechanism it uses to guide recombination and mutation. NSGA-II does not specialize in LSMOPs but serves as a reference to assess the performance of SparseEA. Similarly, moRBC serves as a reference to assess moRBC-WP's performance. MOEA/D was not proposed for LSMOPs, but its decomposition approach in objective space may help focus the search around smaller promising regions in decision space, which could translate into better performance solving LSMOPs.

For all MOEAs we use a population size of 100 individuals. In NSGA-II and MOEA/D we apply two-point crossover with probability of 1.0 and one-bit mutation with probability $1/N$. In MOEA/D we set the neighborhood size to 10 and the weights are generated as described in [26]. These parameter values were chosen as they are commonly used in the literature. It would interesting to investigate their effect on the performance of their respective MOEAs on future works. moRBC-WP is configured with $\delta = 0.002$ when solving MNK-Landscapes with $N \leq 600$, and $\delta = 0.02$ for $N > 600$. Additional fine tuning of this parameter can be done to obtain better performance in more specific settings of the problem being solved, but we have obtained satisfactory results with those

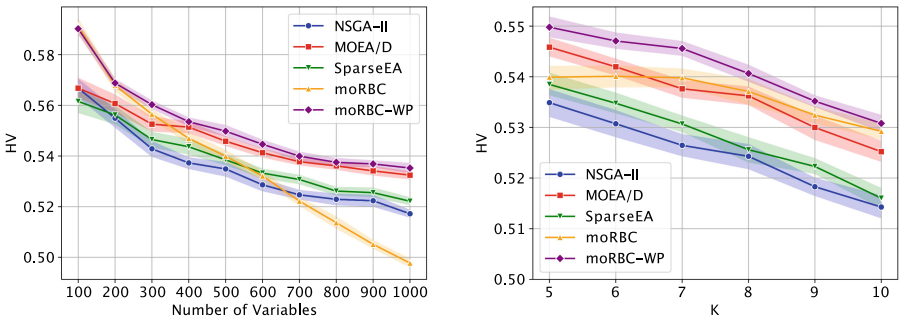
values. We do a more in-depth analysis on the impact of $\delta$ in a later section in this paper. The algorithms were executed for 2,000,000 fitness evaluations, or 20,000 generations.

We use MNK-Landscapes with $M = 2$ objectives, $N = \{100, 200, \ldots, 1000\}$ variables and $K = 5$ as benchmark problems to study the scalability of the algorithms in the number of variables. In addition, we solve MNK-Landscapes with $M = 2$, $N = 500$ and $K = \{5, 6, 7, 8, 9, 10\}$ to analyze the performance of the MOEAs when solving problems with higher interaction between variables. The variables are randomly correlated between each other and independent between objectives.

The performance of the MOEAs studied in this paper are evaluated by the Hypervolume metric [32] with reference point at $[0.0, \ldots, 0.0]$, as MNK-Landscapes used here are maximization problems. In addition, we use the Wilcoxon rank sum test with a significance level of 0.05 to perform statistical analysis between the HV results obtained by the algorithms. All results are obtained from 30 randomly generated MNK-Landscape instances per configuration. Overall, we use 450 problem instances in our study.

### 5.2 Performance Comparison

In this section, we first compare the performance of the algorithms increasing the number of decision variables $N$ while keeping constant the number of epistatic interactions $K$ between variables. Figure 2a shows the HV of the MOEAs when solving MNK-Landscapes with $N = \{100, 200, \ldots, 1000\}$ and fixed $K = 5$. We can see that in problems with $N = 100$, moRBC still performs better than any other algorithm, as previously described in Sect. 3. However, for higher values of $N$, moRBC-WP outperforms all the MOEAs studied. This shows the increase in performance from the original moRBC that can be achieved when selectively climbing only a fraction of the decision variables.




(a) $N = \{100, 200, \ldots, 1000\}$ and $K = 5$. (b) $N = 500$ and $K = \{5, 6, 7, 8, 9, 10\}$.

**Fig. 2.** HV of NSGA-II, MOEA/D, SparseEA, moRBC and moRBC-WP after 2,000,000 fitness evaluations when solving MNK-Landscapes with different $N$ and $K$.

**Table 1.** HV of all MOEAs when solving 30 instances of each MNK-Landscape configuration, where the best results are highlighted in gray.

| Problem | NSGA-II | MOEA/D | SparseEA | moRBC | moRBC-WP |
|---|---|---|---|---|---|
| N = 100 K = 5 | 5.6681e-01 − (1.073e-02) | 5.6680e-01 − (1.038e-02) | 5.6157e-01 − (1.311e-02) | 5.9143e-01 + (7.189e-03) | 5.9024e-01 (5.861e-03) |
| N = 200 K = 5 | 5.5493e-01 − (9.877e-03) | 5.6072e-01 − (9.739e-03) | 5.5621e-01 − (9.951e-03) | 5.6788e-01 ≈ (6.762e-03) | 5.6881e-01 (4.900e-03) |
| N = 300 K = 5 | 5.4283e-01 − (8.132e-03) | 5.5257e-01 − (7.784e-03) | 5.4650e-01 − (8.389e-03) | 5.5650e-01 − (5.717e-03) | 5.6026e-01 (4.194e-03) |
| N = 400 K = 5 | 5.3728e-01 − (6.251e-03) | 5.5141e-01 ≈ (6.131e-03) | 5.4365e-01 − (8.556e-03) | 5.4696e-01 − (6.326e-03) | 5.5354e-01 (4.431e-03) |
| N = 500 K = 5 | 5.3488e-01 − (7.584e-03) | 5.4584e-01 − (5.014e-03) | 5.3849e-01 − (5.842e-03) | 5.3992e-01 − (6.148e-03) | 5.4979e-01 (5.797e-03) |
| N = 600 K = 5 | 5.2862e-01 − (6.5614e-03) | 5.4130e-01 − (4.9126e-03) | 5.3326e-01 − (4.5502e-03) | 5.3210e-01 − (5.1718e-03) | 5.4464e-01 (4.8374e-03) |
| N = 700 K = 5 | 5.2466e-01 − (4.8207e-03) | 5.3775e-01 − (3.4875e-03) | 5.3072e-01 − (4.7178e-03) | 5.2219e-01 − (4.8080e-03) | 5.3994e-01 (4.5470e-03) |
| N = 800 K = 5 | 5.2290e-01 − (5.9765e-03) | 5.3610e-01 ≈ (4.3873e-03) | 5.2615e-01 − (4.3766e-03) | 5.1367e-01 − (6.1600e-03) | 5.3751e-01 (4.6279e-03) |
| N = 900 K = 5 | 5.2234e-01 − (6.2188e-03) | 5.3413e-01 − (4.6546e-03) | 5.2553e-01 − (4.9707e-03) | 5.0514e-01 − (4.3371e-03) | 5.3687e-01 (3.4072e-03) |
| N = 1000 K = 5 | 5.1720e-01 − (3.8922e-03) | 5.3245e-01 − (4.0543e-03) | 5.2215e-01 − (4.2035e-03) | 4.9773e-01 − (4.5120e-03) | 5.3525e-01 (5.2929e-03) |
| N = 500 K = 6 | 5.3072e-01 − (7.773e-03) | 5.4197e-01 − (4.537e-03) | 5.3477e-01 − (5.421e-03) | 5.4009e-01 − (5.924e-03) | 5.4707e-01 (4.197e-03) |
| N = 500 K = 7 | 5.2646e-01 − (6.104e-03) | 5.3763e-01 − (4.783e-03) | 5.3068e-01 − (4.347e-03) | 5.3980e-01 − (5.054e-03) | 5.4559e-01 (4.047e-03) |
| N = 500 K = 8 | 5.2429e-01 − (6.572e-03) | 5.3623e-01 − (5.110e-03) | 5.2560e-01 − (6.567e-03) | 5.3709e-01 ≈ (6.869e-03) | 5.4065e-01 (4.594e-03) |
| N = 500 K = 9 | 5.1829e-01 − (4.879e-03) | 5.2999e-01 − (6.684e-03) | 5.2229e-01 − (4.132e-03) | 5.3244e-01 − (4.638e-03) | 5.3519e-01 (3.119e-03) |
| N = 500 K = 10 | 5.1428e-01 − (6.244e-03) | 5.2521e-01 − (5.671e-03) | 5.1603e-01 − (5.775e-03) | 5.2923e-01 ≈ (4.337e-03) | 5.3080e-01 (4.437e-03) |

**Table 2.** Time taken in seconds to execute all 2,000,000 fitness evaluations when solving an MNK-Landscape with $N = 500$ and $K = 5$.

| | NSGA-II | MOEA/D | SparseEA | moRBC | moRBC-WP |
|---|---|---|---|---|---|
| Runtime (s) | 198.120 (5.110) | 193.156 (3.654) | 187.429 (5.312) | 143.902 (4.542) | **172.568 (1.881)** |

Although MOEA/D is not designed for LSMOPs, it performs very well for problems with high $N$. This is likely due to the fact that MNK-Landscapes present a very uniform distribution in the search space, which is advantageous for the decomposition method adopted by MOEA/D.

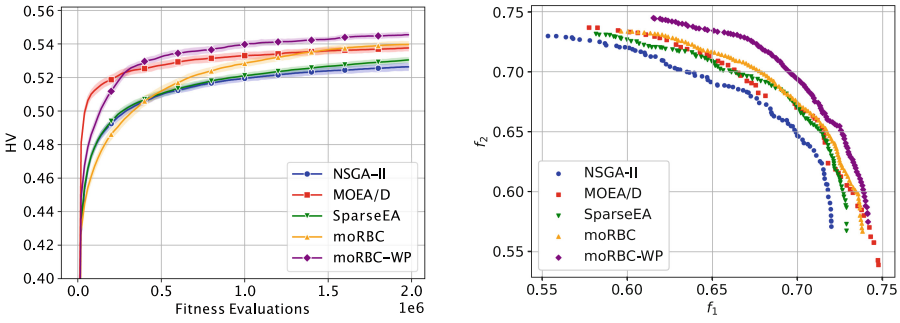
On the other hand, SparseEA, although designed for LSMOPs, shows low HV values in all MNK-Landscape configurations. This is likely due to two factors. First, MNK-Landscapes are not sparse problems, which is the focus of SparseEA. Second, we analyze problems with high level of interactions between variables,

which is not taken into account by SparseEA when ranking and scoring the reference solutions, used to create the initial population and to guide crossover and mutation. However, note that SparseEA performs better than NSGA-II for problems with 300 or more variables.

Next we compare the algorithms keeping constant the number of variables while increasing the number of epistatic interactions between variables. Figure 2b shows the HV of the MOEAs when solving MNK-Landscapes with fixed $N = 500$ and $K = \{5, 6, 7, 8, 9, 10\}$. It is possible to note that the performance of moRBC-WP remains higher than all other MOEAs for all configurations studied here, showing that morBC-WP can maintain its performance even with problems with higher interaction between variables. it is worth noting that moRBC approaches moRBC-WP as $K$ increases. Further investigation with problems with $K > 10$ can be interesting to better understand the balance between exploration and exploitation of the decision variables in problems with high espistasis.

Table 1 summarizes all the results obtained by showing the mean HV and standard deviation in parenthesis after 2,000,000 evaluations of the 5 MOEAs studied. "−", "+" and "≈" symbols next to the mean HV values indicate whether they are significantly worse, better or statistically similar to the HV of moRBC-WP according to the Wilcoxon rank sum test. It can be seen that apart from the problems with $N = 100$, morBC-WP outperforms all other MOEAs.

Figure 3a shows the HV transition of NSGA-II, MOEA/D, SparseEA, moRBC and moRBC-WP over fitness evaluations when solving MNK-Landscapes with $M = 2$, $N = 500$ and $K = 7$. First, we can note that the increase in HV of moRBC is very slow throughout its run, which reflects its nature of exploring all bits one at a time until it finds a local optima and restarts the search, and highlights its low performance when solving LSMOPs. However, when applying the weighted permutation of moRBC-WP, we can see that its HV increases much more quickly, surpassing and remaining above all other MOEAs after about 300,000 fitness evaluations.



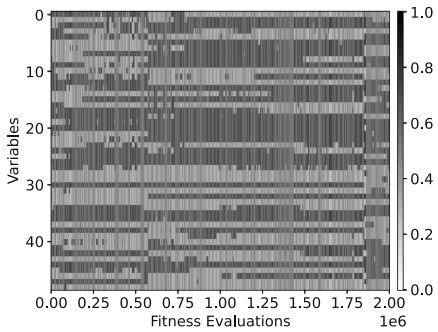

(a) HV over fitness evaluations for all MOEAs studied.

(b) Final population pool of the MOEAs after 2,000,000 fitness evaluations.

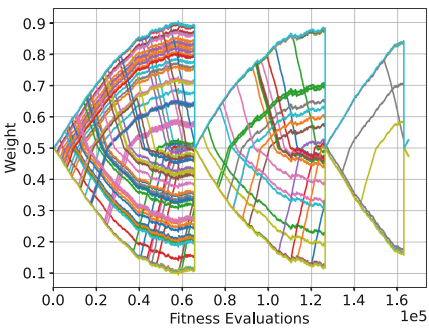
**Fig. 3.** Results when solving MNK-Ladscapes with $N = 500$ $K = 7$.

Figure 3b shows the population found by each MOEA at the end of their execution, after 2,000,000 fitness evaluations, in the 2-dimensional search space, when solving one instance of an MNK-Landscape with $N = 500$ and $K = 7$. It can be clearly seen that the solutions in moRBC-WP have higher fitness values than the other MOEAs. They also show good distribution over the objective space, although not as spread as in MOEA/D, as this algorithm can focus on solutions at the extreme points of the objective space with some of its weights [28].

Another important factor when solving LSMOPs is the computational cost of the solvers, as it might not be feasible to wait for the solver to reach high HV values if they take too long to finish their execution. Here we compare the execution time of the MOEAs studied when run in a MacBook Pro (2021) Apple M1 Pro Chip with 10-Core CPU and 32GB of RAM. Table 2 shows the execution time in seconds of all MOEAs when solving the MNK-Landscapes with $N = 500$ and $K = 5$ after 2,000,000 evaluations. Here, we can emphasize the weakness of NSGA-II and SparseEA, as they show worse HV performance while also having high execution time. MOEA/D also shows high execution time, which is due to the fact that it has to update the neighboring solutions on every generation. moRBC is faster than all other solvers, as it simply mutates solutions bit by bit, without any crossover operator or other complicated sorting techniques. However, as previously shown, it does not perform well in LSMOPs. Finally, we can see that moRBC-WP finishes its executions faster than NSGA-II, MOEA/D and SparseEA while still presenting higher HV values. When compared to the original moRBC, the execution time of moRBC-WP is impacted by the updates of $\boldsymbol{W}$ and the decision making to whether to flip or skip a given bit.

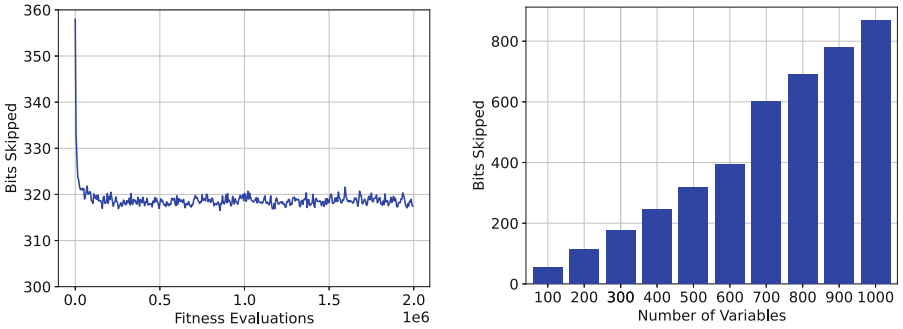



(a) Change in the first 50 weights over all fitness evaluations.




(b) Change in all weights until the first 3 restarts.

**Fig. 4.** Change in the weights $\boldsymbol{W}$ of moRBC-WP when solving MNK-Landscapes with $N = 500$ and $K = 5$.




(a) Skipped bits when solving MNK-Landscapes with $N = 500$ and $K = 5$.

(b) Skipped bits when solving MNK-Landscapes with $N = \{100, \ldots, 1000\}$.

**Fig. 5.** Number of skipped bits by moRBC-WP.

### 5.3 moRBC-WP Analysis

In this section, we further understand how moRBC-WP works by analyzing the features added to the original moRBC. More specifically, we study the change in the weights $\boldsymbol{W}$, the number of skipped bits over the fitness evaluations and the effect of the parameter $\delta$ in the performance of the algorithm.

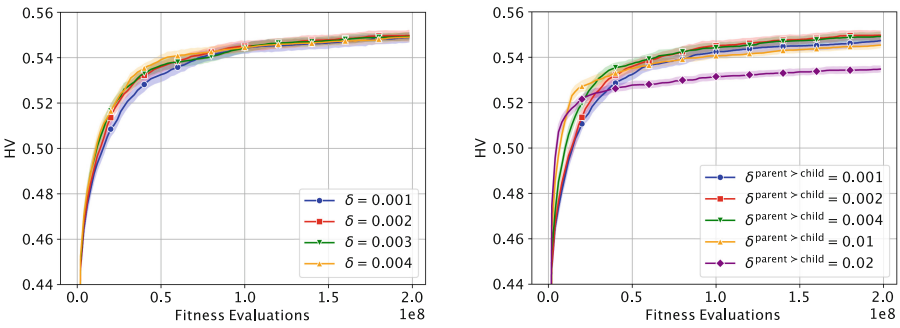
First, we study the added weight array $\boldsymbol{W}$ by analyzing the change of each weight value over the run of the algorithm. We solve one instance of an MNK-Landscape with $N = 500$ and $K = 5$, and show in Fig. 4a the value of the first 50 weights of $\boldsymbol{W}$, in the vertical axis, through all 2,000,000 fitness evaluations, in the horizontal axis, in which a white block means that the weight value in that position is 0 and a black block means that it is 1. We can see that no weight value reaches the extremes of either 0 or 1, indicating that the ideal values for their respective variables have not been found yet. In addition, most of the bits switch between darker and lighter colors over time, meaning that the algorithm is constantly finding better values for these variables, likely when it escapes local optima to move towards higher ranked ones.

Figure 4b shows the weights of the entire $\boldsymbol{W}$ array over the first 160,000 fitness evaluations, which is when the algorithm has completed 3 restarts. Note that weights with the same values are overlapped. We can see that they all start at the default weight value of 0.5, and rapidly move towards the range limits of 0 or 1. As it advances through the fitness evaluations, the algorithm gains more confidence in the value of the decision variables, which is reflected by the weights getting closer to either 0 or 1. It is also possible to see more clearly the moments when the weights are moving towards opposite directions in the y-axis as the algorithm finds more suitable values to their corresponding decision variables. Additionally, in this problem instance, note that the weights are reset to the default value of 0.5 around the 70,000th, 125,000th and 160,000th fitness evaluations, which are when all variables of the parent have been visited

without finding a child that dominates the parent, and the algorithm restarts the search and resets $\boldsymbol{W}$. As more restarts happen, the weights values are less spread, indicating that the algorithm is gaining more confidence in the values of the variables. Nevertheless, they are still allowed to switch from 0 to 1 and vice-versa, as we can see the lines crossing across the y-axis of the graph.

Next, we study the bits that are skipped by moRBC-WP during its run. Figure 5a shows the average number of skipped bits for each non-dominated solution found by moRBC-WP over fitness evaluations when solving MNK-Landscapes with $N = 500$ and $K = 5$. We can see that moRBC-WP starts by skipping around 360 bits per solution, then quickly decreases and plateaus to around 320 bits until it finishes its execution, with small fluctuations throughout the entire run. This highlights the importance of the learning portion of the algorithm, as it allows morBC-WP to dynamically vary the number of bits that are skipped in order to obtain good performance when solving the MNK-Landscapes.

Then, we analyze the number of skipped bits when varying $N$. We solve MNK-Landscapes with $N = \{100, 200, \ldots, 1000\}$ and $K = 5$, and plot the average number of skipped bits over $N$ in Fig. 5b. For problems with $N \leq 600$, the number of skipped bits grows linearly as $N$ increases, which explains why moRBC-WP can, to some extent, perform well as $N$ increases with a fixed $\delta = 0.002$. However, it does not skip enough bits to maintain its performance for problems with $N > 600$, which is why we set $\delta = 0.02$ in those problems, which is reflected by the sudden increase in the number of skipped bits in the plot.




(a) Varying $\delta$. (b) Varying $\delta^{parent \succ child}$.

**Fig. 6.** HV of moRBC-WP when solving MNK-Landscapes with $N = 500$ and $K = 5$ with different parameter values.

Next, we investigate the impact of the parameter $\delta$ in the robustness and performance of moRBC-WP. First, we analyze the robustness of the parameter $\delta$, i.e., the impact a change on its value has on the performance of moRBC-WP [8]. Figure 6a shows the HV of moRBC-WP solving MNK-Landscapes with $N = 500$ and $K = 5$, varying $\delta = \{0.001, 0.002, 0.003, 0.004\}$. Note that, apart from a slightly faster increase in HV at earlier fitness evaluation in configurations with

higher $\delta$ values, the performance is similar between all configurations, specially when comparing the final HV after 2,000,000 fitness evaluations. This shows that the algorithm is not very sensitive to small changes in $\delta$ and indicating that, when fine tuning $\delta$, satisfactorily good results can be achieved, even though its ideal value for the current problem's configuration have not been found.

Lastly, we analyze the individual impact of changing one rate of weight change of $\mathbf{W}$. Here, we keep $\delta = 0.002$, as done in all other experiments, but detach the rate of weight change when a dominated solution is found (line 17 of Algorithm 1), here referred as $\delta^{parent \succ child}$. Figure 6b shows the HV of moRBC-WP when solving MNK-Landscapes with $N = 500$ and $K = 5$, varying $\delta^{parent \succ child} = \{0.002, 0.004, 0.01, 0.02\}$. We can see that higher $\delta^{parent \succ child}$ cause moRBC-WP to quickly increase the HV in earlier fitness evaluations. However, the algorithm converges to lower levels of HV in later fitness evaluations. Although not shown here, similar behaviors are observed when independently varying the rate of weight changes when a dominant or non-dominated solution is found and when the weights are decayed. This shows that there is a trade-off between quicker convergence and higher final HV values when independently configuring these parameters. As a consequence, it also shows that a practitioner can independently fine tune the rates of weight changes attached to each Pareto dominance condition based on the their need for a faster HV convergence at early fitness evaluations or higher HV values in more time consuming runs.

## 6 Conclusions

In this paper, we have presented an improvement to the traditional moRBC, named moRBC-WP, that boosts its performance when solving binary LSMOPs by implementing a method that learns from previously found solutions, and applies that information to decide which variables should be focused on when executing the bit climbs.

To study the developed algorithm, we have compared them to well known binary MOEAs, NSGA-II, MOEA/D, SparseEA and moRBC, and shown that moRBC-WP has overall better performance than all the MOEAs compared, and fast execution time in relation to the MOEAs studied. Then, we further analyzed the proposed algorithm by studying the weight array added, the number of bits that are skipped over its run, and studied the effect of the newly introduced parameter by analyzing its effect on the algorithm's performance and the robustness of the algorithm to changes in the parameter's value.

In future works, we would like to improve moRBC-WP by implementing a method to automatically determine the optimal values of its parameter, as well as making it dynamic, so its value can change over time according to trends in the solutions found. In addition, we believe that we can further improve the performance of moRBC-WP by changing its restart strategy and its truncation method. Finally, we would like to solve problems with more objectives and

different correlation between variables, including higher number of interactions and different types of interactions, such as correlation between objectives [24], as well as solving different problem, such as real-world ones.

**Disclosure of Interests.** The authors have no competing interests to declare that are relevant to the content of this article.

## References


1. Aguirre, H., Oyama, A., Tanaka, K.: Adaptive $\epsilon$-sampling and $\epsilon$-hood for evolutionary many-objective optimization. In: Purshouse, R.C., Fleming, P.J., Fonseca, C.M., Greco, S., Shaw, J. (eds.) EMO 2013. LNCS, vol. 7811, pp. 322–336. Springer, Heidelberg (2013). https://doi.org/10.1007/978-3-642-37140-0_26
2. Aguirre, H., Tanaka, K.: Insights on properties of multiobjective mnk-landscapes. In: Proceedings of the 2004 Congress on Evolutionary Computation, vol. 1, pp. 196–203 (2004). https://doi.org/10.1109/CEC.2004.1330857
3. Aguirre, H., Tanaka, K.: Random bit climbers on multiobjective mnk-landscapes: Effects of memory and population climbing. IEICE Trans. **88-A**, 334–345 (2005). https://doi.org/10.1093/ietfec/E88-A.1.334
4. Aguirre, H., Tanaka, K.: Working principles, behavior, and performance of moeas on mnk-landscapes. Eur. J. Oper. Res. **181**(3), 1670–1690 (2007). https://doi.org/10.1016/j.ejor.2006.08.004
5. Aguirre, H., Zapotecas, S., Liefooghe, A., Verel, S., Tanaka, K.: Approaches for many-objective optimization: analysis and comparison on mnk-landscapes. In: Bonnevay, S., Legrand, P., Monmarché, N., Lutton, E., Schoenauer, M. (eds.) EA 2015. LNCS, vol. 9554, pp. 14–28. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-31471-6_2
6. Antonio, L., Coello, C.: Use of cooperative coevolution for solving large scale multiobjective optimization problems. In: IEEE Congress on Evolutionary Computation, pp. 2758–2765. IEEE (2013). https://doi.org/10.1109/CEC.2013.6557903
7. Beume, N., Naujoks, B., Emmerich, M.: Sms-emoa: multiobjective selection based on dominated hypervolume. Eur. J. Oper. Res. **181**(3), 1653–1669 (2007). https://doi.org/10.1016/j.ejor.2006.08.008
8. Brewer, K., Carraway, L., Ingram, D.: Forward selection as a candidate for constructing nonregular robust parameter designs. Arkansas State University, Tech. rep. (2010)
9. Brockhoff, D., Wagner, T., Trautmann, H.: R2 indicator-based multiobjective search. Evol. Comput. **23**(3), 369–395 (2015). https://doi.org/10.1162/EVCO_a_00135
10. Coello, C., Van Veldhuizen, D., Lamont, G.: Evolutionary Algorithms for Solving Multi-Objective Problems. Genetic Algorithms and Evolutionary Computation, Springer, US, USA (2013)
11. Deb, K.: Multi-Objective Optimization Using Evolutionary Algorithms. Wiley Interscience Series in Systems and Optimization, Wiley
12. Deb, K., Pratap, A., Agarwal, S., Meyarivan, T.: A fast and elitist multiobjective genetic algorithm: Nsga-ii. IEEE Trans. Evol. Comput. **6**(2), 182–197 (2002). https://doi.org/10.1109/4235.996017

13. Hadka, D., Reed, P.: Borg: an auto-adaptive many-objective evolutionary computing framework. Evol. Comput. **21**(2), 231–259 (2013). https://doi.org/10.1162/EVCO_a_00075
14. He, C., Cheng, R., Yazdani, D.: Adaptive offspring generation for evolutionary large-scale multiobjective optimization. IEEE Trans. Systems Man Cybernet. Syst. **52**(2), 786–798 (2022). https://doi.org/10.1109/TSMC.2020.3003926
15. Hong, W., Yang, P., Tang, K.: Evolutionary computation for large-scale multi-objective optimization: A decade of progresses. Int. J. Autom. Comput. **18**, 155–169 (2021). https://doi.org/10.1007/s11633-020-1253-0
16. Jain, H., Deb, K.: An evolutionary many-objective optimization algorithm using reference-point based nondominated sorting approach, part ii: Handling constraints and extending to an adaptive approach. IEEE Trans. Evol. Comput. **18**(4), 602–622 (2014). https://doi.org/10.1109/TEVC.2013.2281534
17. Kauffman, S.: The Origins of Order: Self-Organization and Selection in Evolution. Oxford University Press, England (1993)
18. Laumanns, M., Thiele, L., Deb, K., Zitzler, E.: Combining convergence and diversity in evolutionary multiobjective optimization. Evol. Comput. **10**(3), 263–282 (2002). https://doi.org/10.1162/106365602760234108
19. Mahdavi, S., Shiri, M.E., Rahnamayan, S.: Metaheuristics in large-scale global continues optimization: a survey. Inf. Sci. **295**, 407–428 (2015). https://doi.org/10.1016/j.ins.2014.10.042
20. Pelikan, M.: Nk landscapes, problem difficulty, and hybrid evolutionary algorithms. In: Proceedings of the 12th Annual Conference on Genetic and Evolutionary Computation, pp. 665-672. Association for Computing Machinery, New York (2010). https://doi.org/10.1145/1830483.1830606
21. Tian, Y., Wang, C., Zhang, X., Jin, Y.: An evolutionary algorithm for large-scale sparse multi-objective optimization problems. IEEE Trans. Evol. Comput. **24**(2), 380–393 (2019). https://doi.org/10.1109/TEVC.2019.2918140
22. Tian, Y., Lu, C., Zhang, X., Tan, K.C., Jin, Y.: Solving large-scale multiobjective optimization problems with sparse optimal solutions via unsupervised neural networks. IEEE Trans. Cybernet. **51**(6), 3115–3128 (2021). https://doi.org/10.1109/TCYB.2020.2979930
23. Tian, Y., Si, L., Zhang, X., Cheng, R., He, C., Tan, K.C., Jin, Y.: Evolutionary large-scale multi-objective optimization: a survey. ACM Comput. Surv. **54**(8), 1–34 (2021). https://doi.org/10.1145/3470971
24. Verel, S., Liefooghe, A., Jourdan, L., Dhaenens, C.: On the structure of multiobjective combinatorial search space: Mnk-landscapes with correlated objectives. Eur. J. Oper. Res. **227**(2), 331–342 (2013). https://doi.org/10.1016/j.ejor.2012.12.019
25. Von Lücken, C., Brizuela, C., Baran, B.: An overview on evolutionary algorithms for many-objective optimization problems. Wiley Interdisc. Rev. Data Mining Knowl. Dis. **9**(1), e1267 (2018). https://doi.org/10.1002/widm.1267
26. Zapotecas-Martínez, S., Aguirre, H.E., Tanaka, K., Coello, C.A.C.: On the low-discrepancy sequences and their use in moea/d for high-dimensional objective spaces. In: IEEE Congress on Evolutionary Computation, pp. 2835–2842. IEEE (2015). https://doi.org/10.1109/CEC.2015.7257241
27. Zapotecas-Martínez, S., Coello, C.A.C., Aguirre, H.E., Tanaka, K.: Challenging test problems for multi- and many-objective optimization. Swarm Evol. Comput. **81**, 101350 (2023). https://doi.org/10.1016/j.swevo.2023.101350
28. Zhang, Q., Li, H.: Moea/d: a multiobjective evolutionary algorithm based on decomposition. IEEE Trans. Evol. Comput. **11**(6), 712–731 (2007). https://doi.org/10.1109/TEVC.2007.892759

29. Zhang, X., Tian, Y., Cheng, R., Jin, Y.: A decision variable clustering-based evolutionary algorithm for large-scale many-objective optimization. IEEE Trans. Evol. Comput. **22**(1), 97–112 (2018). https://doi.org/10.1109/TEVC.2016.2600642
30. Zille, H., Ishibuchi, H., Mostaghim, S., Nojima, Y.: Weighted optimization framework for large-scale multi-objective optimization. In: Proceedings of the 2016 on Genetic and Evolutionary Computation Conference Companion, pp. 83-84. Association for Computing Machinery, New York (2016). https://doi.org/10.1145/2908961.2908979
31. Zitzler, E., Künzli, S.: Indicator-based selection in multiobjective search. In: Yao, X., et al. (eds.) PPSN 2004. LNCS, vol. 3242, pp. 832–842. Springer, Heidelberg (2004). https://doi.org/10.1007/978-3-540-30217-9_84
32. Zitzler, E.: Evolutionary algorithms for multiobjective optimization: methods and applications. Springer, US, USA (1999)

# An Unbounded Archive-Based Inverse Model in Evolutionary Multi-objective Optimization

Rongguang Ye, Longcan Chen, Jinyuan Zhang(✉), and Hisao Ishibuchi(✉)

Guangdong Provincial Key Laboratory of Brain-inspired Intelligent Computation, Department of Computer Science and Engineering, Southern University of Science and Technology, Shenzhen 518055, China
{yerg2023,12250061}@mail.sustech.edu.cn, {zhangjy,hisao}@sustech.edu.cn

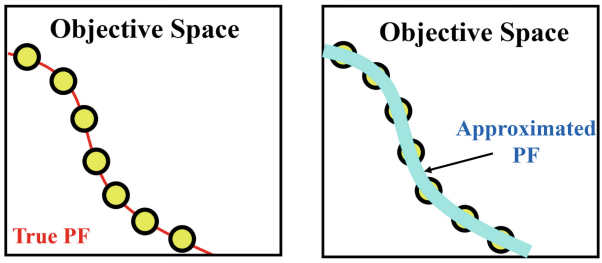
**Abstract.** Inverse model (IM) is a method for tailoring solutions to decision-makers based on their preferences. Existing approaches are often trained by the final solution set (as training data) obtained from a multi-objective evolutionary algorithm (MOEA). The final solution set obtained by MOEA usually has a limited number of samples. However, model training will perform poorly when there are few samples in the final solution set. To further improve the performance of the model, we propose an unbounded archive-based inverse model (UAIM) to enhance the quality of the trained inverse model. We first create an unbounded archive to collect all non-dominated solutions during the execution of MOEA. Unlike IM, UAIM is trained using all solutions in the archive. Moreover, for a decision maker's preference, an alternative solution from the archive is considered if the suggested solution is inferior to the alternative solution in the archive. UAIM thus may provide more reliable suggested solutions for decision-makers. To better evaluate algorithms, we propose two indicators that can measure the matching degree between the suggested solution and the decision maker's preference. We demonstrate that the proposed UAIM is superior to IM on ten problems.

**Keywords:** Decision Maker · Inverse Model · Multi-Objective Optimization

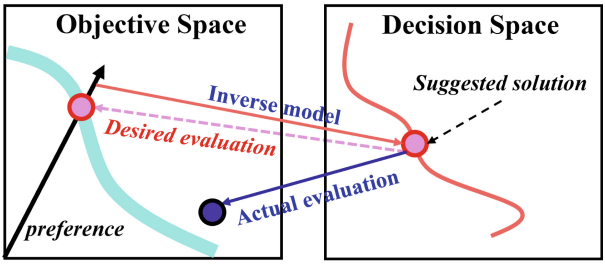
## 1 Introduction

In multi-objective optimization problems (MOPs), conflicting objectives need to be optimized and no single solution can optimize all objectives simultaneously. Multi-objective evolutionary algorithms (MOEAs) are one of the most effective ways to solve MOPs. Various MOEAs have been proposed to find solutions with different optimal trade-offs among all objectives [1–3]. These solutions are called Pareto optimal solutions in the decision space. The set comprising all Pareto optimal solutions is termed the Pareto set, and their image in objective space is known as the Pareto front. MOEAs iteratively optimize all solutions in the population by setting a certain population size. With the development of


M. Affenzeller et al. (Eds.): PPSN 2024, LNCS 15151, pp. 186–201, 2024.
https://doi.org/10.1007/978-3-031-70085-9_12




(a) The training stage of the inverse model. The red curve is the Pareto front (PF).




(b) The inference stage of the inverse model. The purple point is objective values of the suggested solution provided by the inverse model.

**Fig. 1.** Diagram of inverse model training and inference. The yellow points are the training data, and the black vector is the preference vector. (Color figure online)

MOEAs, they can already obtain uniform solutions on different shapes of Pareto fronts (e.g., irregular, disconnected, etc.) [4–6].

However, the continuous Pareto front typically exhibits an infinite number of solutions. The limited number of solutions obtained by a multi-objective evolutionary algorithm (MOEA) can not fully represent the continuous Pareto front. A crucial challenge in MOEAs lies in approximating the continuous Pareto front. To address this challenge, some researchers use a hyperplane or polynomial function to fit the Pareto front [7,8]. In other words, they use the objective values of the solutions obtained by the MOEA to estimate the parameters of the hyperplane or polynomial function. For example, let $\alpha$ be a hyper-parameter and $f_i$ be the $i$-th objective. $\sum_{i=1}^{m} {f_i}^{\alpha} = 1$ is used to approximate the Pareto front. This kind of approach aims to fit a continuous Pareto front in the objective space. However, the continuous Pareto front obtained in this way loses its relationship with the decision space (i.e., the Pareto set might not be obtained).

Recently, the inverse model has been proposed to learn continuous Pareto front [9]. In the literature, Suresh et al. [9] proposed an inverse model to map the preference vectors in the objective space to the values of decision variables. In this way, the inverse model could learn the mapping from the decision-maker's

preferences to corresponding solutions, i.e., the decision-maker can get an exclusive solution by simply providing a preference. The current inverse model usually uses the final solution set obtained from the MOEA as training data. However, The MOEA generally can only obtain a fixed number (population size) of solutions, and the small amount of training data makes training the inverse model difficult. The solutions predicted by the inverse model may deviate from the decision-maker's expectations if the inverse model is poorly trained. As shown in Fig. 1, Fig. 1(a) presents the inverse model fitting the training data. Figure 1(b) shows the inference stage of the inverse model. When the model outputs a suggested solution given a preference vector, we expect the evaluated solution (pink point) to lie on the preference vector and close to the Pareto front. However, when the model is not trained well, the evaluated solution (purple point) may be located far from the preference and the Pareto front.

To further improve the quality of solutions provided by the inverse model to decision-makers, we propose an Unbounded Archive-based Inverse Model (UAIM). By storing all non-dominated solutions in an unbounded external archive [10,11], we improve the quality of the inverse model in two key aspects. First, instead of utilizing only the final solution set for training, we incorporate all non-dominated solutions that have been evaluated to train the inverse model. This approach enhances the reliability of the inverse model's predictions, as the training data size can be increased by hundreds or even thousands of times. Second, incorporating the unbounded archive is beneficial during the inference stage. Sometimes, the solution stored in the archive may outperform the suggested solution. Hence, we treat the solution in the archive as an alternative solution. If the performance of the suggested solution is worse than the solution in the archive, we use the solution from the archive. These dual strategies collectively enhance the quality of solutions provided to decision-makers.

Our contributions are summarized as follows:

- We propose a novel unbounded archive-based inverse model (UAIM). UAIM uses all solutions in the archive training the inverse model, which improves the predictive ability of the model. In the inference phase, we combine solutions from the archive to replace inferior suggested solutions.
- We design two new indicators to evaluate the matching degree of solutions obtained by decision-makers. The proposed indicators couple the preferences of decision-makers.
- The experimental results validate that the proposed method is significantly better than IM on ten benchmarks.

## 2 Related Work

There are many MOEAs [12–15] are designed to solving multi-objective optimization problems. Unfortunately, these methods only generate a limited number of solutions, thereby the Pareto front obtained by those methods cannot be fully approximated. For example, IM-MOEA [16] utilizes the Gaussian process to estimate the objective values of some random samples from objective

space. They create a Gaussian model for each objective and each cluster of population. In a continuous MOP with $m$ objectives, the Pareto set usually is a $m-1$-dimensional manifold [17]. Although this kind of machine learning-based method can get a better final solution set, they can only obtain a small part of the Pareto front. In addition, some studies directly learn a model using the obtained solution set in the objective space. For example, pa$\lambda$-MyDE [7] and RIB-EMOA [8] use a polynomial function to fit the solution set in the objective space. MMEA [18] uses a hyperplane with scaling parameters to estimate the Pareto front. GFM-MOEA [19] is an improvement of MMEA, which makes the power term of each objective variable as a parameter. Although these methods can accurately estimate the Pareto front, they cannot obtain solutions in the corresponding decision space. Therefore, this method for estimating the Pareto front is of limited use.

Recently, MORM [16] uses the inverse model to map the objective space to the decision space. Then, objective values of the final population are input to the inverse model and further evaluate the predicted solution. In this way, MORM can more accurately estimate the Pareto front by generating more solutions. The method of Anirudh et al. [9] replaces the input of the inverse model with pseudo-weights, which can provide a suggested solution for each point on the Pareto front. Although it can potentially estimate solutions on the entire Pareto front, each preference mapped by the inverse model is not perfect (i.e., the solution deviates significantly from the decision maker's expectations).

## 3 Background

### 3.1 Multi-objective Optimization

A multi-objective optimization problem (MOP) with $m$ objectives and $d$ decision variables can be defined as follows:

$$\min_{\boldsymbol{x}\in\mathcal{X}\subseteq\mathbb{R}^d} \boldsymbol{f}(\boldsymbol{x}) = (f_1(\boldsymbol{x}), f_2(\boldsymbol{x}), ..., f_m(\boldsymbol{x})), \tag{1}$$

where $\boldsymbol{x}$ is the decision variables in the decision space $\mathcal{X}$ and $\boldsymbol{f}(\boldsymbol{x})$ is the objective functions: $\mathbb{R}^d \to \mathbb{R}^m$. No single solution can simultaneously optimize all objective functions. The goal of solving MOP is to find all Pareto optimal solutions with different trade-offs on the Pareto front. Often, objective vectors are compared using Pareto domination. The widely used definitions of MOP are described as follows:

**Definition 1.** ***(Pareto Domination).*** *A solution $\boldsymbol{x_1}$ is said to dominate another solution $x_2$, denoted as $\boldsymbol{x_1} \prec \boldsymbol{x_2}$, if and only if $\boldsymbol{f}_i(x_1) \leq \boldsymbol{f}_i(x_2), \forall i \in \{1, \ldots, m\}$ and $\exists\ j \in \{1, ..., m\}$ such that $\boldsymbol{f}_j(x_1) < \boldsymbol{f}_j(x_2)$.*

**Definition 2.** ***(Pareto Optimal Solution).*** *Solution $\boldsymbol{x_1}$ is called a Pareto optimal solution if there does not exist a solution $\boldsymbol{x_2}$ such that $\boldsymbol{x_2} \prec \boldsymbol{x_1}$.*

**Definition 3.** ***(Pareto Set and Pareto Front).*** *The set of all Pareto optimal solutions is called Pareto set (denoted as $\mathcal{M}_{ps}$). The Pareto front $\mathcal{P}(\mathcal{M}_{ps})$ is the image of the Pareto Set.*

$$\mathcal{P}(\mathcal{M}_{ps}) = \{\boldsymbol{f}(\boldsymbol{x}) : \boldsymbol{x} \in \mathcal{M}_{ps}\}$$

Many indicators have been designed to measure the performance of evolutionary multi-objective algorithms in approximating the Pareto front. For example, when the actual Pareto front is known, we can use indicators such as GD [20] and IGD [21] to measure the algorithm's performance. Hypervolume [22] can be used when the actual Pareto front is unknown.

**Definition 4.** ***(Hypervolume).*** *The hypervolume (HV) indicator, $HV(\mathcal{P}(X), \boldsymbol{r})$ is m-dimensional Lebesgue measure $\lambda_M$ of the region dominated by $\mathcal{P}(X)$ and bounded from below by a reference point $r \in \mathbb{R}^m$.*

### 3.2 Inverse Model for Decision Making

The inverse model is a method that maps the decision space from the objective space. For a set of preference vectors $\Lambda$ (it can be arbitrary), Inverse model $\phi$ takes the preference vector set $\Lambda$ as the input and the decision variables $X$ as the output, i.e., $\hat{X} = \phi(\boldsymbol{f}(\Lambda), \boldsymbol{\theta})$, where $\boldsymbol{\theta}$ is the parameter of the inverse model. The inverse model can approximate the entire Pareto set by inputting continuous preference vectors. In addition, another role of the inverse model is to use the inverse model to provide solutions for decision makers [9].

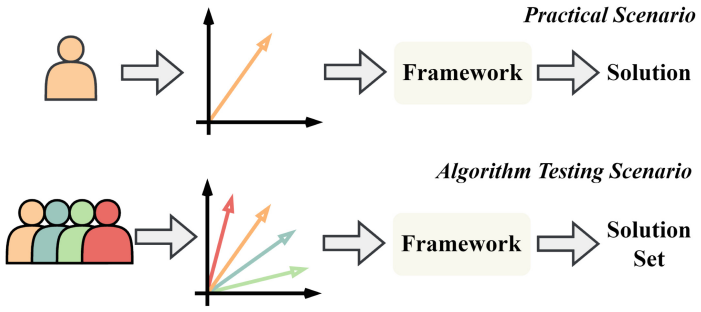



**Fig. 2.** The process of providing a solution for a given decision maker and testing algorithm.

As shown in Fig. 2, this work focuses on providing a solution for arbitrary decision-makers. In the practical scenario, the framework can return a solution (independently) for the preferences of any decision-maker. In the scenario of testing the framework's performance, we evaluate all algorithms using uniform preferences.

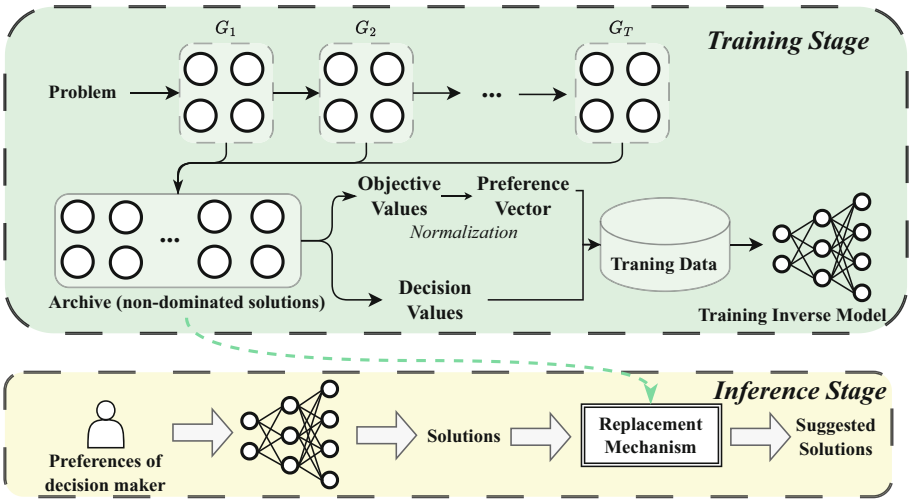



**Fig. 3.** Framework of UAIM. The green and yellow regions denote the training and inference stages, respectively. (Color figure online)

## 4 Unbounded Archive-Based Inverse Model

This section presents an unbounded archive-based inverse model (UAIM), an advanced inverse model used to provide the solution to any decision-maker. As shown in Fig. 3, UAIM first stores all non-dominated solutions in the process of MOEA for training rather than only training the inverse model on final solutions. Then, we design a mechanism to replace inferior solutions suggested by the trained inverse model. In the inference stage, some of the inaccurate solutions suggested by the inverse model will be replaced using solutions from the archive. For the remainder of this section, we describe the core components of UAIM and propose suitable performance indicators.

### 4.1 Archive Construction

We can obtain a final solution set with size $N$ (population size) when using an evolutionary algorithm to solve the problem in Equation (1). Then, we can train an inverse model on the final solution set. However, MOEAs generally can only obtain solutions with a predefined population size. The number of solutions in the final solution set is usually a few hundred or dozens. Learning an inverse model using such a limited number of solutions is usually insufficient. The inverse model can have better performance if there is sufficient training data.

As shown in Fig. 3, an empty archive ($\mathcal{A}$) is initialized at the beginning to collect training data, which will collect all non-dominated solutions. Next, the evolutionary algorithm is used for optimization. In each generation, when a new solution set $G_i (i = 1, 2, ..., T)$ is obtained by the evolution algorithm,

the non-dominated solution set ($G_i^{'}$) is selected from $G_i$. After that, we select non-dominated ($ND$) solution set $\mathcal{A}_{i-1}$ from the combination of $G_i^{'}$ and $\mathcal{A}_{i-1}$, i.e., $\mathcal{A}_i = ND(G_i^{'} \cup \mathcal{A}_{i-1})$. $\mathcal{A}$ can collect all non-dominated solutions when the evolutionary algorithm terminates. All collected non-dominated solutions are used to train the inverse model. Usually, the number of solutions in $\mathcal{A}$ is much larger than $N$, so the inverse model may perform better for suggesting solutions to decision-makers.

### 4.2 Unbounded Archive Training

Since the scale of the target value in the solution set cannot be determined, we need to normalize it to an applicable scale. To design an inverse model that can provide solutions to any preferences given by decision-makers, we need to construct a mapping from preference vectors to decision vectors. First, we use normalization to change the minimum value of the objective vector of solutions in archive to zeros:

$$\boldsymbol{p}'(x) = \frac{\boldsymbol{f}(\boldsymbol{x}) - \boldsymbol{f}_{\min}}{\boldsymbol{f}_{\max} - \boldsymbol{f}_{\min}}, \tag{2}$$

where $x \subseteq \mathcal{A}$, $\boldsymbol{f}_{min}$, and $\boldsymbol{f}_{max}$ are the minimum and maximum objective value of each objective in the unbounded archive $\mathcal{A}$. To provide a solution for the decision-maker's preference more conveniently, we set the sum of the preference vector to one. The final preference vector $\boldsymbol{p}$ is:

$$\boldsymbol{p}(\boldsymbol{x}) = \frac{\boldsymbol{p}^{'}(\boldsymbol{x})}{||\boldsymbol{p}'(\boldsymbol{x})||_1}, \tag{3}$$

where $||\boldsymbol{p}^{'}(\boldsymbol{x})||_1$ represents the $L_1$ norm. The inverse model $\boldsymbol{\beta} \in \mathcal{B}$ can map preference vectors to a solution in the decision space, i.e., $\hat{\boldsymbol{x}} = \phi(\boldsymbol{p}(\boldsymbol{x}); \boldsymbol{\beta})$. The mean squared error $J(\boldsymbol{\beta})$ is used as loss function of the inverse model:

$$J(\boldsymbol{\beta}) = \mathbb{E}_{(\boldsymbol{p},\boldsymbol{x})\sim(\Lambda,\mathcal{P}_{ps})}(\phi(\boldsymbol{p};\boldsymbol{\beta}) - \boldsymbol{x})^2, \tag{4}$$

where $\Lambda$ is the preference vector set and $\mathcal{P}_{ps}$ is the Pareto set distribution. We approximate $\Lambda$ and $\mathcal{P}_{ps}$ and minimize the mean squared error based on the collected training data, as follows:

$$\arg\min_{\boldsymbol{\beta}\in\mathcal{B}} \frac{1}{K}\sum_{i=1}^{K}(\phi(\boldsymbol{p}_i;\boldsymbol{\beta}) - \boldsymbol{x_i})^2, \tag{5}$$

where $\mathcal{A}$ is the unbounded archive obtained in Sect. 4.1 and $\widetilde{P}$ is calculated by Eq. (2) and (3). $K$ is the number of training data in the unbounded archive. Then, we calculate the gradient of Equation (5) for optimizing the model.

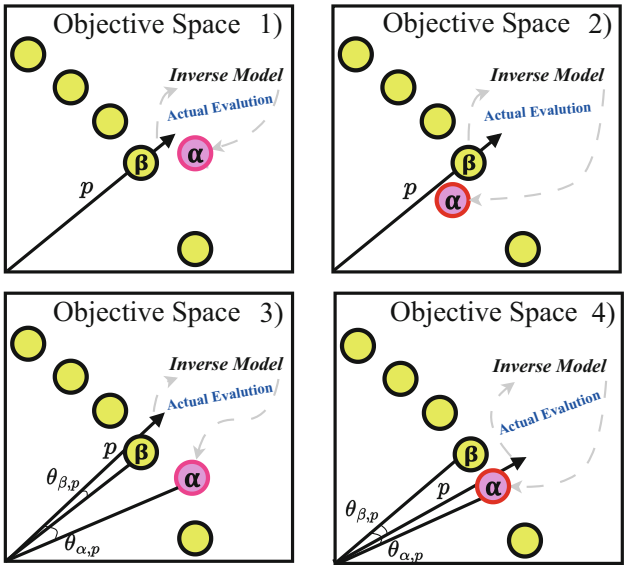



**Fig. 4.** Four situations in the replacement mechanism. The solution $\boldsymbol{\alpha}$ is the output of the inverse model given the preference vector $\boldsymbol{p}$, and the yellow point is the solution in the archive. Solution $\boldsymbol{\beta}$ is the solution closest to preference vector $\boldsymbol{p}$. The purpose of the replacement mechanism is to determine whether $\boldsymbol{\beta}$ is more suitable for the preference vector $p$ than $\boldsymbol{\alpha}$.

### 4.3 Unbounded Archive Replacement Mechanism

A common phenomenon is that the inverse model can provide perfect suggested solutions for some preference vectors but not for others. There is a possibility that some other solutions in training data are better than the suggested solutions provided by the inverse model. Therefore, we propose a replacement mechanism to improve the quality of suggested solutions when the solution given by the inverse model is worse than the training data. As shown in Fig. 4, the suggested solution $\boldsymbol{\alpha}$ is provided by the inverse model for the preference vector $\boldsymbol{p}$. The solution $\boldsymbol{\beta}$ closest to the preference vector in the training data is selected for comparison with the solution $\boldsymbol{\alpha}$. Here are four possible situations:

1) The solution $\boldsymbol{\beta}$ dominates solution $\boldsymbol{\alpha}$, it means that $\boldsymbol{\beta}$ is no worse than $\boldsymbol{\alpha}$ each objective, therefore, we provide $\boldsymbol{\beta}$ to preference vector $\boldsymbol{p}$.
2) The solution $\boldsymbol{\alpha}$ dominates solution $\boldsymbol{\beta}$, it means that $\boldsymbol{\alpha}$ is worse than $\boldsymbol{\beta}$ each objective. Therefore, we provide $\boldsymbol{\alpha}$ to the preference vector $\boldsymbol{p}$.
3) Solution $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$ do not dominate each other, we believe that $\boldsymbol{\beta}$ is the best for the preference vector $\boldsymbol{p}$ if $\theta_{\boldsymbol{\alpha},\boldsymbol{p}} > \theta_{\boldsymbol{\beta},\boldsymbol{p}}$. In this case, $\boldsymbol{\beta}$ is provided to the preference vector $\boldsymbol{p}$.
4) Solution $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$ do not dominate each other, we believe that $\boldsymbol{\alpha}$ is the best for the preference vector $\boldsymbol{p}$ if $\theta_{\boldsymbol{\alpha},\boldsymbol{p}} < \theta_{\boldsymbol{\beta},\boldsymbol{p}}$. In this case, $\boldsymbol{\beta}$ is provided to the preference vector $\boldsymbol{p}$.

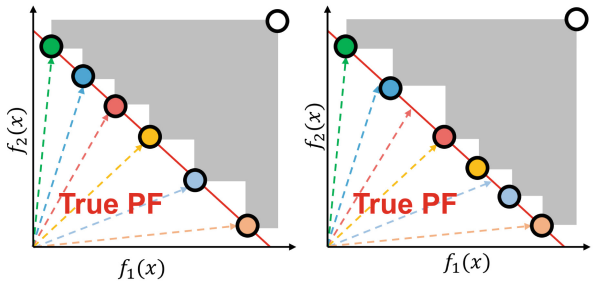



**Fig. 5.** Calculation of HV under different decision-makers. Different color vectors represent the specific preferences, and the corresponding color points represent the solutions suggested to decision-makers. The gray is the HV value. The red line is the true Pareto front, and the white point is the reference point. (Color figure online)

### 4.4 Performance Indicators

HV and IGD are the two most commonly used performance indicators in multi-objective optimization. They are used to evaluate the quality of the obtained solution set. However, when providing a solution set for a group of decision-makers, we mainly consider the suitability of each solution for the corresponding decision-maker. Currently, the evaluation of the solution set by HV and IGD has a shortcoming in our case. As shown in Fig. 5, it gives a simple example to illustrate why HV is insufficient for evaluation. The two subgraphs plot the solutions provided to decision-makers by different algorithms. The figures show that the first algorithm is more suitable for the decision-maker's preferences because solutions of the first algorithm falls on preference vectors. However, we can see that the HV values obtained by the solutions in the two figures are the same (the gray areas are the same).

To address the bove mentioned issue, we couple the calculation of HV with the decision-maker's preferences. The matching degree of each solution to the preference vector in HV is considered. Specifically, we calculate the angle between each preference vector $\boldsymbol{p}$ and the objective value of the corresponding solution $\phi_{\boldsymbol{\theta}}(\boldsymbol{p})$, For a given set of preference vectors $\Lambda$ and $F = \boldsymbol{f}(\phi_{\boldsymbol{\theta}}(\Lambda))$, the average deviation angle is:

$$\theta = \frac{\sum_{\boldsymbol{f} \in F, \boldsymbol{p} \in \Lambda} \arccos\left(\frac{\boldsymbol{f} \cdot \boldsymbol{p}}{\|\boldsymbol{f}\| \cdot \|\boldsymbol{p}\|}\right)}{|\Lambda|}, \tag{6}$$

where $|\Lambda|$ is the number of decision-makers. Then, the new performance indicator is $HV_\theta = \frac{\frac{\pi}{2}-\theta}{\frac{\pi}{2}} HV$. The newly designed performance indicator contains the penalty of each solution for the decision-maker. We also propose another decision-maker-based average distance (AD) metric to evaluate the quality of the solution set where the Pareto front is known:

$$AD = \frac{\sum_{\boldsymbol{f} \in F, \boldsymbol{f}_t \in \mathcal{P}^b} \|\boldsymbol{f} - \boldsymbol{f}^b\|_2}{|F|}, \tag{7}$$

where $\mathcal{P}^b$ is the best solution set in the true Pareto front for the given preference $\Lambda$. Different from IGD [21], we first pre-select Pareto optimal solutions from the Pareto set according to the preference vector set $\mathcal{P}^b$. Then, we calculate the average distance corresponding to $\mathcal{P}^b$ and $F$. $AD$ is calculated based on the distance after preference vector matching. As a result, $AD$ is a more suitable indicator than IGD in the decision-making scenario.

## 5 Experiments

We examine the performance of UAIM on extensive benchmarks with various parameters, including the number of decision variables ($d$), the number of objectives ($m$), and the shape of the Pareto front. We consider five types of Pareto front as follows:

- **Linear** [3]: DTLZ1 ($d$=5, $m$=3).
- **Convex** [23,24]: ZDT2 ($d$=30, $m$=2), Convex-DTLZ2 ($d$=10, $m$=3) and Convex-DTLZ4 ($d$=10, $m$=3).
- **Concave** [3,23]: ZDT1 ($d$=30, $m$=3), DTLZ2 ($d$=10, $m$=3) and DTLZ4 ($d$=10, $m$=3).
- **Disconnected** [3,23]: ZDT3 ($d$=30, $m$=2) and DTLZ5 ($d$=10, $m$=3).
- **Degenerated** [3]: DTLZ5 ($d$=10, $m$=3).

We use NSGA-II to solve the multi-objective optimization problems. In the setting of NSGA-II, generation ($T$) and population size ($N$) are set to 200 and 55, respectively. UAIM is compared with NSGA-II and the inverse model (IM). To fairly compare the learning-based methods (IM and UAIM) and the evolution-based method NSGA-II, 495 uniform preference vectors are used as inputs to IM and UAIM for obtaining corresponding solutions. Then, we use the $HV_\theta$ (proposed in Sect. 4.4) and $AD$ to measure the performance of all algorithms and run each algorithm three times to calculate the mean and standard deviation of $HV_\theta$ and $AD$. The larger the $HV_\theta$ value, the better the performance; the smaller the $AD$ value, the better the performance. UAIM is implemented in pymoo [25] and pytorch [26]. All experiments used one NVIDIA GeForce RTX 2080 Ti GPU (11GB RAM). The code is available at https://github.com/rG223/UAIM.

### 5.1 Results

In Table 1, we evaluate all methods using $HV_\theta$ and $AD$. Among the two objective problems (ZDT1-3), IM has advantages over NSGA-II in ZDT1 and ZDT2 problems but has no advantage in ZDT3 problems. This is because the Pareto front of ZDT3 is disconnected, some preference vectors have no intersection with the Pareto front. The proposed method UAIM achieves the best performance in

**Table 1.** Comparison of the mean and standard deviation of algorithms in terms of $HV_\theta$ and $AD$ on ten benchmarks.

| Problem | Indictor | Method | | |
|---|---|---|---|---|
| | | NSGA-II | IM | UAIM |
| ZDT1 (D=30) | $HV_\theta \uparrow$ | $0.8546_{\pm 0.0003}$ | $0.8649_{\pm 0.0010}$ | $\mathbf{0.8677_{\pm 0.0001}}$ |
| | $AD \downarrow$ | $0.0091_{\pm 0.0002}$ | $0.0049_{\pm 0.0006}$ | $\mathbf{0.0012_{\pm 0.0002}}$ |
| ZDT2 (D=30) | $HV_\theta \uparrow$ | $0.5280_{\pm 0.0009}$ | $0.5327_{\pm 0.0008}$ | $\mathbf{0.5354_{\pm 0.0000}}$ |
| | $AD \downarrow$ | $0.0080_{\pm 0.0005}$ | $0.0034_{\pm 0.0009}$ | $\mathbf{0.0011_{\pm 0.0001}}$ |
| ZDT3 (D=30) | $HV_\theta \uparrow$ | $1.0186_{\pm 0.0005}$ | $1.0026_{\pm 0.0139}$ | $\mathbf{1.0208_{\pm 0.0002}}$ |
| | $AD \downarrow$ | $0.0080_{\pm 0.0004}$ | $0.0144_{\pm 0.0017}$ | $\mathbf{0.0011_{\pm 0.0001}}$ |
| DTLZ1 (D=5) | $HV_\theta \uparrow$ | $0.1234_{\pm 0.0005}$ | $0.0813_{\pm 0.0266}$ | $\mathbf{0.1324_{\pm 0.0018}}$ |
| | $AD \downarrow$ | $0.0356_{\pm 0.0008}$ | $0.1253_{\pm 0.0630}$ | $\mathbf{0.0287_{\pm 0.0036}}$ |
| DTLZ2 (D=10) | $HV_\theta \uparrow$ | $0.6165_{\pm 0.0109}$ | $0.6554_{\pm 0.0079}$ | $\mathbf{0.7122_{\pm 0.0021}}$ |
| | $AD \downarrow$ | $0.0965_{\pm 0.0045}$ | $0.0408_{\pm 0.0024}$ | $\mathbf{0.0263_{\pm 0.0178}}$ |
| DTLZ4 (D=10) | $HV_\theta \uparrow$ | $0.6267_{\pm 0.0052}$ | $0.6061_{\pm 0.0204}$ | $\mathbf{0.7075_{\pm 0.0037}}$ |
| | $AD \downarrow$ | $0.0921_{\pm 0.0016}$ | $0.1032_{\pm 0.0124}$ | $\mathbf{0.0226_{\pm 0.0023}}$ |
| DTLZ5 (D=10) | $HV_\theta \uparrow$ | $0.2912_{\pm 0.0026}$ | $\mathbf{0.2942_{\pm 0.0092}}$ | $0.2918_{\pm 0.0002}$ |
| | $AD \downarrow$ | $\mathbf{0.0085_{\pm 0.0001}}$ | $0.0162_{\pm 0.0065}$ | $0.0034_{\pm 0.0005}$ |
| DTLZ7 (D=10) | $HV_\theta \uparrow$ | $2.0292_{\pm 0.0092}$ | $1.9741_{\pm 0.0220}$ | $\mathbf{2.2222_{\pm 0.0082}}$ |
| | $AD \downarrow$ | $0.1407_{\pm 0.0041}$ | $0.1467_{\pm 0.0009}$ | $\mathbf{0.0601_{\pm 0.0109}}$ |
| Convex-DTLZ2 (D=10) | $HV_\theta \uparrow$ | $1.0709_{\pm 0.0690}$ | $1.1831_{\pm 0.0104}$ | $\mathbf{1.2523_{\pm 0.0026}}$ |
| | $AD \downarrow$ | $0.0841_{\pm 0.0156}$ | $0.0421_{\pm 0.0060}$ | $\mathbf{0.0232_{\pm 0.0107}}$ |
| Convex-DTLZ4 (D=10) | $HV_\theta \uparrow$ | $1.1197_{\pm 0.0128}$ | $0.8463_{\pm 0.0794}$ | $\mathbf{1.2404_{\pm 0.0031}}$ |
| | $AD \downarrow$ | $0.0693_{\pm 0.0054}$ | $0.2137_{\pm 0.0434}$ | $\mathbf{0.0189_{\pm 0.0012}}$ |

**Table 2.** The rate of using archive solution on ten problems.

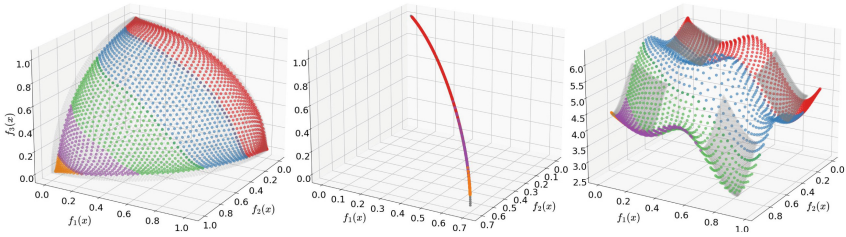
| Problem | The rate of using archive solution |
|---|---|
| ZDT1 | $0.3758_{\pm 0.0966}$ |
| ZDT2 | $0.3697_{\pm 0.0309}$ |
| ZDT3 | $0.5636_{\pm 0.0647}$ |
| DTLZ1 | $0.7091_{\pm 0.1653}$ |
| DTLZ2 | $0.2364_{\pm 0.0786}$ |
| DTLZ4 | $0.6182_{\pm 0.1178}$ |
| DTLZ5 | $0.7576_{\pm 0.0600}$ |
| DTLZ7 | $0.5818_{\pm 0.0535}$ |
| Convex-DTLZ2 | $0.4182_{\pm 0.0514}$ |
| Convex-DTLZ4 | $0.7939_{\pm 0.0562}$ |

**Table 3.** The impact of two components in UAIM. UAT and UARM represent the use of the archive to train the inverse model and replace suggested solutions, respectively.

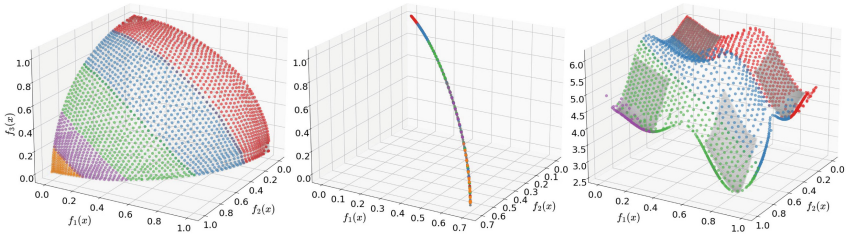
| Problem | UAT | UARM | $HV_\theta \uparrow$ | $AD \downarrow$ |
|---|---|---|---|---|
| DTLZ1 | ✗ | ✗ | $0.0813_{\pm 0.0266}$ | $0.2658_{\pm 0.0991}$ |
| | ✓ | ✗ | $0.0961_{\pm 0.0248}$ | $0.2583_{\pm 0.0250}$ |
| | ✓ | ✓ | $\mathbf{0.1324_{\pm 0.0018}}$ | $\mathbf{0.0287_{\pm 0.0036}}$ |
| DTLZ2 | ✗ | ✗ | $0.6554_{\pm 0.0079}$ | $0.0408_{\pm 0.0024}$ |
| | ✓ | ✗ | $\mathbf{0.7123_{\pm 0.0015}}$ | $\mathbf{0.0256_{\pm 0.0183}}$ |
| | ✓ | ✓ | $0.7122_{\pm 0.0021}$ | $0.0266_{\pm 0.0182}$ |
| DTLZ4 | ✗ | ✗ | $0.6061_{\pm 0.0204}$ | $0.2318_{\pm 0.0349}$ |
| | ✓ | ✗ | $0.6989_{\pm 0.0097}$ | $0.0370_{\pm 0.0119}$ |
| | ✓ | ✓ | $\mathbf{0.7075_{\pm 0.0037}}$ | $\mathbf{0.0226_{\pm 0.0023}}$ |
| DTLZ5 | ✗ | ✗ | $0.2942_{\pm 0.0092}$ | $0.0439_{\pm 0.0085}$ |
| | ✓ | ✗ | $\mathbf{0.3054_{\pm 0.0015}}$ | $\mathbf{0.0052_{\pm 0.0009}}$ |
| | ✓ | ✓ | $0.2918_{\pm 0.0002}$ | $0.0116_{\pm 0.0020}$ |
| DTLZ7 | ✗ | ✗ | $1.9741_{\pm 0.0220}$ | $0.3559_{\pm 0.0336}$ |
| | ✓ | ✗ | $2.2044_{\pm 0.0141}$ | $\mathbf{0.0739_{\pm 0.0217}}$ |
| | ✓ | ✓ | $\mathbf{2.2222_{\pm 0.0082}}$ | $0.0741_{\pm 0.0217}$ |
| Convex-DTLZ2 | ✗ | ✗ | $1.1831_{\pm 0.0104}$ | $0.1054_{\pm 0.0478}$ |
| | ✓ | ✗ | $1.2512_{\pm 0.0031}$ | $\mathbf{0.0356_{\pm 0.0321}}$ |
| | ✓ | ✓ | $\mathbf{1.2523_{\pm 0.0026}}$ | $0.0389_{\pm 0.0330}$ |
| Convex-DTLZ4 | ✗ | ✗ | $0.8463_{\pm 0.0794}$ | $0.6106_{\pm 0.0712}$ |
| | ✓ | ✗ | $1.0967_{\pm 0.0200}$ | $0.0467_{\pm 0.0016}$ |
| | ✓ | ✓ | $\mathbf{1.2404_{\pm 0.0031}}$ | $\mathbf{0.0189_{\pm 0.0012}}$ |

ZDT1-3. The strengths of UAIM are that the model is more confident than IM because it has enough training data. In addition, UAIM can replace bad solutions predicted by the model using solutions in the archive, which sometimes ensures that the final suggested solutions are not too bad. In the DTLZ1 problem, IM's $HV_\theta$ and $AD$ are 0.0813 and 0.1253, which are much worse than NSGA-II. Therefore, the solution quality obtained by the inverse model on DTLZ1 is poor. In DTLZ5, UAIM performs slightly worse than IM because the solution set is closer to the Pareto front (Fig. 6(a)). This is because UAIM chooses solutions closer to the preference vector but farther from the Pareto front in the replacement mechanism. IM also performed poorly on the disconnected DTLZ7 problem, while UAIM performed significantly better than IM and NSGA-II.

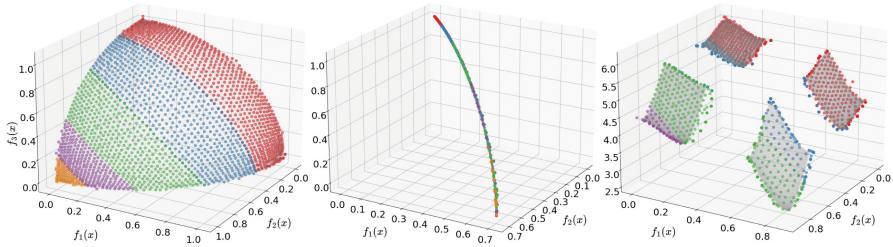
In Table 2, we present the rate of the number of archive solutions used by UAIM in each problem. We can find them between 20% and 80%. Therefore, the proposed method combines the solutions given by the inverse model and the archive.



(a) The suggested solutions provided by IM for decision-makers on DTLZ2, DTLZ5 and DTLZ7 problems.



(b) The suggested solutions provided by archived IM (trained using the archive) for decision-makers on DTLZ2, DTLZ5 and DTLZ7 problems.



(c) The suggested solutions provided by UAIM for decision-makers on DTLZ2, DTLZ5 and DTLZ7 problems.

**Fig. 6.** Visual results obtained by different kinds of the preference of decision makers. The color coding represents different preference ranges for the first objective: orange (0 to 0.2), purple (0.2 to 0.4), green (0.4 to 0.6), blue (0.6 to 0.8), and red (0.8 to 1). (Color figure online)

### 5.2 Ablation Study

In this subsection, we examine the effectiveness of unbounded archive training (UAT) and the replacement mechanism (UARM). As shown in Table 3, UARM and UAT have greatly improved the performance of IM on DTLZ1, DTLZ4, DTLZ7, and Convex-DTLZ4. In DTLZ5, using UARM will make the value of $HV_\theta$ slightly worse. This is because most preference vectors have no intersection with the Pareto front in the degenerated problem. Then, UARM usually chooses solutions closer to the preference vector but farther from the Pareto front. Finally, some solutions close to the Pareto front may be sacrificed in

exchange for being closer to the preference vector. When the improvement of UARM is obvious, it means that the solutions returned by IM on many preference vectors are not ideal, and the solution in the training data is better in this case. When the improvement of UAT is obvious, it means that the solutions returned by IM on many preference vectors are ideal. Still, there will be greater improvement after expanding the training data.

### 5.3 Decision-Makers with Different Preferences

The advantage of the inverse model is that it can provide specific solutions for any group of decision-makers. To visualize the solutions obtained by different decision-makers, we tested the solutions obtained by IM and UAIM on the DTLZ2, DTLZ5, and DTLZ7 problems under about 1800 uniform preferences. As shown in Fig. 6, the shape of the solution set provided by IM is smooth in DTLZ2, but for some preferences on the edge of the Pareto front, e.g. (0.05, 0.05, 0.9), IM cannot provide corresponding solutions. The archived inverse model (trained using the archive) can improve the estimation accuracy for each decision maker's preferences. Therefore, the archived inverse model can perform better than IM in DTLZ2. The solution set provided by UAIM can also cover the entire Pareto front for different decision-makers. In the second DTLZ5 problem, IM covers most of the region on the Pareto front by one type of preference (red dots), which shows that IM does not have excellent discrimination ability in different preferences. The archived inverse model can better distinguish different preferences and return appropriate solutions. UAIM combines solutions in the archive to further improve the quality of the suggested solution set. In the last DTLZ7 problem, some preference vectors do not intersect with the Pareto front. IM tends to provide dominated solutions, which is terrible for those decision-makers. For any preference vector, UAIM gives non-dominated solutions near the Pareto front.

## 6 Conclusion and Future Work

In this paper, we proposed an unbounded archive-based inverse model to improve the performance of the original inverse model for decision-making. UAIM has made two improvements. On the one hand, UAIM uses an archive to train the inverse model instead of the final population. Secondly, UAIM contains a replacement mechanism. Specifically, when the solution suggested by the model is worse than the solution in the archive, we use a better solution in the archive instead. Through these two aspects, UAIM improves the quality of solutions for decision-makers. Besides, two new indicators are designed to evaluate the algorithm's performance. We demonstrate in extensive experiments that the proposed method significantly outperforms the original inverse model.

This paper mainly focuses on the amount of training data for the inverse model and a replacement mechanism. Future research on using the inverse model can be considered as follows:

1. The impact of training data quality on the inverse model. In some problems, the training data collected may contain noise, or the data distribution is unbalanced (i.e., the data shifts toward a preference for a specific objective).
2. We can further consider using the inverse model to improve the performance of multi-objective evolutionary algorithms. For example, in many-objective optimization problems, it is challenging to find the entire Pareto front. We can use the inverse model to explore those sparse areas. As a result, we can improve the quality of the solution set obtained by multi-objective evolutionary algorithms.

**Acknowledgements.** This work was supported by National Natural Science Foundation of China (Grant No. 62106099, 62250710163, 62376115), Guangdong Basic and Applied Basic Research Foundation (Grant No. 2023A1515011380), Guangdong Provincial Key Laboratory (Grant No. 2020B121201001).

**Disclosure of Interest.** The authors declare that they have no conflict of interest.

## References

1. Deb, K.: Multi-objective optimisation using evolutionary algorithms: an introduction. In: Wang, L., Ng, A., Deb, K. (eds.) Multi-objective Evolutionary Optimisation for Product Design and Manufacturing. Springer, London (2011). https://doi.org/10.1007/978-0-85729-652-8_1
2. Miettinen, K.: Nonlinear multiobjective optimization, vol. 12. Springer Science & Business Media (1999)
3. Deb, K., Thiele, L., Laumanns, M., Zitzler, E.: Scalable test problems for evolutionary multiobjective optimization. In: Abraham, A., Jain, L., Goldberg, R. (eds.) Evolutionary Multiobjective Optimization. Advanced Information and Knowledge Processing. Springer, London (2005). https://doi.org/10.1007/1-84628-137-7_6
4. Li, M., Yao, X.: What weights work for you? adapting weights for any pareto front shape in decomposition-based evolutionary multiobjective optimisation. Evol. Comput. **28**(2), 227–253 (2020)
5. Burachik, R.S., Kaya, C.Y., Rizvi, M.M.: A new scalarization technique to approximate pareto fronts of problems with disconnected feasible sets. J. Optimizat. Theory Appli. **162**, 428–446 (2014)
6. Jiang, S., Yang, S.: An improved multiobjective optimization evolutionary algorithm based on decomposition for complex pareto fronts. IEEE Trans. Cybernet. **46**(2), 421–437 (2015)
7. Tian, Y., Zhang, X., Cheng, R., He, C., Jin, Y.: Guiding evolutionary multiobjective optimization with generic front modeling. IEEE Trans. Cybernet. **50**(3), 1106–1119 (2018)
8. Zapotecas Martínez, S., Sosa Hernández, V.A., Aguirre, H., Tanaka, K., Coello Coello, C.A.: Using a family of curves to approximate the pareto front of a multi-objective optimization problem. In: Bartz-Beielstein, T., Branke, J., Filipič, B., Smith, J. (eds.) PPSN 2014. LNCS, vol. 8672, pp. 682–691. Springer, Cham (2014). https://doi.org/10.1007/978-3-319-10762-2_67
9. Suresh, A., Deb, K.: Machine learning based prediction of new pareto-optimal solutions from pseudo-weights. IEEE Trans. Evolutionary Comput. (2023) (Early Access)

10. Li, M., López-Ibáñez, M., Yao, X.: Multi-objective archiving. IEEE Trans. Evolutionary Comput. (2023) (Early Access)
11. Ishibuchi, H., Pang, L.M., Shang, K.: A new framework of evolutionary multi-objective algorithms with an unbounded external archive, pp. 283–290 (2020)
12. Zhang, Q., Li, H.: MOEA/D: a multiobjective evolutionary algorithm based on decomposition. IEEE Trans. Evol. Comput. **11**(6), 712–731 (2007)
13. Li, X., Li, K., Zeng, T., Ye, T., Zhang, L., Wang, H.: Artificial bee colony with multiple search strategies and a new updating mechanism. Int. J. Comput. Sci. Math. **18**(1), 44–53 (2023)
14. Li, K., et al.: A new artificial bee colony algorithm based on modified search strategy. Int. J. Comput. Sci. Math. **15**(4), 387–395 (2022)
15. Li, K., Wang, H., Wang, W., Wang, F., Cui, Z.: Improving artificial bee colony algorithm using modified nearest neighbor sequence. J. King Saud Univ.-Comput. Inform. Sci. **34**(10), 8807–8824 (2022)
16. Bidgoli, A.A., et al.: Machine learning-based framework to cover optimal pareto-front in many-objective optimization. Complex Intell. Syst. **8**(6), 5287–5308 (2022)
17. Hillermeier, C.: Generalized homotopy approach to multiobjective optimization. J. Optim. Theory Appl. **110**(3), 557–583 (2001)
18. Zhou, A., Zhang, Q., Jin, Y.: Approximating the set of pareto-optimal solutions in both the decision and objective spaces by an estimation of distribution algorithm. IEEE Trans. Evol. Comput. **13**(5), 1167–1189 (2009)
19. Zhu, F., et al.: A coordinated optimization framework for long-term complementary operation of a large-scale hydro-photovoltaic hybrid system: Nonlinear modeling, multi-objective optimization and robust decision-making. Energy Convers. Manage. **226**, 113543 (2020)
20. Van Veldhuizen, D.A.: Multiobjective evolutionary algorithms: classifications, analyses, and new innovations. Air Force Institute of Technology (1999)
21. Coello Coello, C.A., Reyes Sierra, M.: A study of the parallelization of a coevolutionary multi-objective evolutionary algorithm. In: Monroy, R., Arroyo-Figueroa, G., Sucar, L.E., Sossa, H. (eds.) MICAI 2004. LNCS (LNAI), vol. 2972, pp. 688–697. Springer, Heidelberg (2004). https://doi.org/10.1007/978-3-540-24694-7_71
22. Fonseca, C.M., Paquete, L., López-Ibánez, M.: An improved dimension-sweep algorithm for the hypervolume indicator. In: Proceedings of IEEE International Conference on Evolutionary Computation, pp. 1157–1163. IEEE (2006)
23. Zitzler, E., Deb, K., Thiele, L.: Comparison of multiobjective evolutionary algorithms: Empirical results. Evol. Comput. **8**(2), 173–195 (2000)
24. Deb, K., Jain, H.: An evolutionary many-objective optimization algorithm using reference-point-based nondominated sorting approach, part i: Solving problems with box constraints. IEEE Trans. Evol. Comput. **18**(4), 577–601 (2013)
25. Blank, J., Deb, K.: Pymoo: multi-objective optimization in python. IEEE Access **8**, 89497–89509 (2020)
26. Paszke, A., et al.: Pytorch: An imperative style, high-performance deep learning library, vol. 32 (2019)

# Reinvestigating the R2 Indicator: Achieving Pareto Compliance by Integration

Lennart Schäpermeier[1,2(✉)] and Pascal Kerschke[1,2]

[1] Big Data Analytics in Transportation, TU Dresden, Dresden, Germany
{lennart.schaepermeier,pascal.kerschke}@tu-dresden.de
[2] ScaDS.AI Dresden/Leipzig, Leipzig, Germany

**Abstract.** In multi-objective optimization, set-based quality indicators are a cornerstone of benchmarking and performance assessment. They capture the quality of a set of trade-off solutions by reducing it to a scalar number. One of the most commonly used set-based metrics is the R2 indicator, which describes the expected utility of a solution set to a decision-maker under a distribution of utility functions. Typically, this indicator is applied by discretizing this distribution of utility functions, yielding a weakly Pareto-compliant indicator. In consequence, adding a nondominated or dominating solution to a solution set *may* – but does not have to – improve the indicator's value.

In this paper, we reinvestigate the R2 indicator under the premise that we have a continuous, uniform distribution of (Tchebycheff) utility functions. We analyze its properties in detail, demonstrating that this continuous variant is indeed Pareto-compliant – that is, any beneficial solution will improve the metric's value. Additionally, we provide an efficient computational procedure to compute this metric for bi-objective problems in $\mathcal{O}(N \log N)$. As a result, this work contributes to the state-of-the-art Pareto-compliant unary performance metrics, such as the hypervolume indicator, offering an efficient and promising alternative.

**Keywords:** Performance assessment · Multi-objective optimization · R2 indicator · Benchmarking · Utility functions · Pareto compliance

## 1 Introduction

When optimizing any system, there is often not just one objective, but multiple criteria required to assess the quality of a solution. Rather than aggregating these different optimization objectives into one, e.g., by means of a linear combination of the individual objectives, the domain of multi-objective (MO) optimization aims to find a set of (Pareto-)optimal trade-off solutions to present to a decision-maker [13]. However, to benchmark MO optimizers and facilitate algorithm design, parameter tuning, and automated algorithm selection, quantifying


M. Affenzeller et al. (Eds.): PPSN 2024, LNCS 15151, pp. 202–216, 2024.
https://doi.org/10.1007/978-3-031-70085-9_13

the quality of trade-off solutions in a unary set-based performance indicator is often necessary.

To be reasonably interpretable, it is recommended that a MO performance indicator fulfills a property that is known as Pareto compliance [8]. More precisely, a set-based performance indicator is called Pareto-compliant, if and only if its indicator value for set A is better than for set B when set A dominates set B. In addition, an indicator is called weakly Pareto-compliant if its value for set A is not worse than for set B.

Up to now, the hypervolume (HV) indicator, and variants thereof, are the only set-based performance indicators that are recognized as truly Pareto-compliant [2,9,17]. The HV indicator computes the $m$-dimensional hypervolume dominated by the solution set w.r.t. to a user-specified anti-optimal reference point.

In contrast, there are multiple families of weakly Pareto-compliant indicators. Exemplary and widely used representatives are the IGD+ indicator [11], requiring an (approximated or known) reference Pareto front, or the R2 indicator [5,10], requiring an ideal/utopian reference point as well as a sample of aggregation (or: utility) functions.

The R2 indicator, in particular, may be an attractive choice as it requires an ideal rather than an anti-optimal reference point. In many problems, the ideal point is easier to find, e.g., in multi-objective machine learning problems where optimal, but not always anti-optimal, values for loss functions are available. Also, solutions that do not dominate a chosen reference point may not contribute to the dominated hypervolume, and a reference point far away from the Pareto front (PF) tends to put a high weight on solutions at the PF's boundary, which is also often undesirable. Another benefit arises when constructing test problems from objectives with known single-objective optima, which lack a natural upper bound for the reference point or a clear "region of interest".

While the unary R2 indicator was initially defined as an integral over a continuum of utility functions by [10], it is usually only discussed and applied in an approximate manner for which the distribution of utility functions is discretized [5,14]. The latter comes with the benefit of being flexible regarding the involved utility functions. It also provides a convenient way to compute the indicator as an average of multiple utility functions, however, sacrificing Pareto compliance in the process. In this paper, we consider the most common R2 indicator definition with a uniform distribution of Tchebycheff utility functions [5,10]. Our main contribution is the methodology to compute this indicator exactly for a set of solutions in the bi-objective case, thereby preserving its Pareto compliance. We achieve this by shifting perspective away from averaging over predefined utility functions towards computing the R2 indicator contributions of each individual nondominated point, thereby eliminating weaknesses in the R2 indicator's properties. Additionally, we demonstrate the exact R2 indicator values for individual points and linear Pareto fronts, as well as the approximate nature of the discretization-based approach commonly used so far.

This paper is structured as follows: We introduce multi-objective optimization and set-based performance assessment in Sect. 2. Then, in Sect. 3, we derive

the methodology for computing exact R2 indicator values, first for a single solution and then for a set of solutions. Section 4 presents some exemplary results regarding characteristics of the exact R2 indicator, and Sect. 5 concludes the paper with an outlook on future research avenues.

## 2 Background

We begin by introducing some fundamental aspects of multi-objective optimization and dominance relationships of multi-objective solutions. Then, we will cover core aspects of set-based performance assessment in multi-objective optimization and its best known representative, the hypervolume indicator. Finally, we introduce the R2 indicator with its most important properties for the discretized and the exact variant.

### 2.1 Multi-objective Optimization

In multi-objective (MO) optimization, we aim to (w.l.o.g.) minimize multiple conflicting objectives. Commonly, a MO optimization problem (MOP) with $m$ objectives is given by an objective function $F : \mathcal{X} \mapsto \mathbb{R}^m$ where $\mathcal{X}$ represents the decision space. The individual objectives are denoted as $f_i : \mathcal{X} \mapsto \mathbb{R}, i = 1, \ldots, m$ in this work. Further, we are primarily considering the bi-objective setting ($m = 2$). Depending on the particular problem and decision space, there may be further constraints on admissible solutions.

A particular challenge posed by MOPs pertains to solution comparison. While in single-objective optimization, solutions can be compared directly (either they have identical objective values or one is better than another), such immediate comparisons are not possible for all solutions of a MOP. To solve this, we need the concept of dominance. A solution $x$ *dominates* another solution $y$ ($x \prec y$), iff $f_i(x) \leq f_i(y)$ for all $i$ and $f_i(x) < f_i(y)$ for at least one $i$. A solution $x$ *strongly dominates* another solution $y$ if the stronger condition $f_i(x) < f_i(y)$ holds for all $i$. A solution that dominates, but does not strongly dominate, another solution is also called *weakly dominant*. Finally, two solutions can be incomparable, that is, *mutually nondominated*, if either fulfills some objective better than the other.

Definitions of dominance can also be extended to sets of solutions. A set of solutions $A$ (weakly) dominates another set $B$, if each member of $B$ is (weakly) dominated by a solution in $A$, written as $A \preceq B$ and $A \prec B$, respectively [8,18].

The set of all nondominated solutions $P = \{x \in \mathcal{X} \mid \nexists y \in \mathcal{X} : y \prec x\}$ is known as the Pareto set, and its image under $F$ is known as the Pareto front. The Pareto set contains the optimal trade-off solutions regarding the objectives, and aiming to obtain a close approximation to it is the prevalent approach of solving MOPs when no further constraints or preferences on the objectives are known, i.e., under black-box assumptions. Evolutionary algorithms are the most widespread approach for finding good Pareto set approximations in these conditions.

Finally, we call the vector of the optimal, individual function values *ideal point*, and refer to the best vector dominated by all Pareto optimal points as

*nadir point*. Often, before computing indicator values and if the ideal and nadir points are available, the region between them is normalized to the $[0,1]^m$ box in objective space as a normalization technique.

### 2.2 Set-Based Performance Assessment

As the Pareto set generally contains more than one solution, set-based performance measures are the norm in assessing the overall quality of an archive of evaluated points. This need to quantify Pareto set approximations has led to numerous performance measures being introduced. For a recent survey on MO performance indicators, we refer to [1].

The most prominent set-based performance measure is the dominated hypervolume (HV) indicator (or: $S$-metric) that measures the space dominated by the set of solutions w.r.t. an anti-optimal reference point [2,9,17]. An illustration of the HV indicator for two objectives is given in Fig. 1.

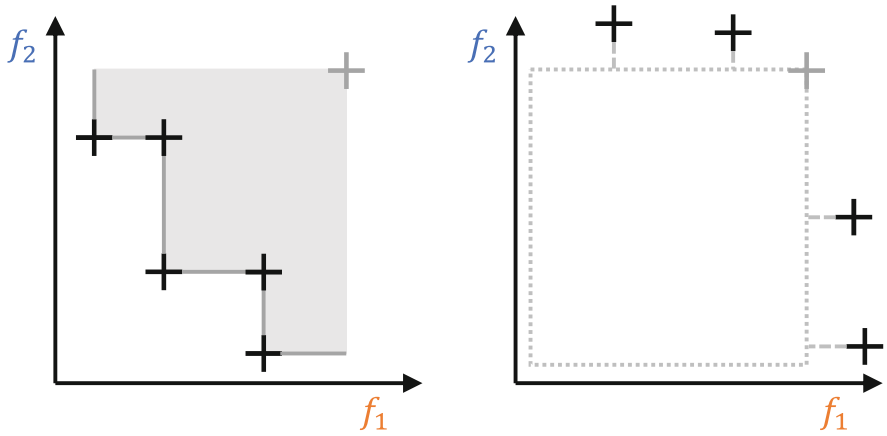



**Fig. 1.** Left: Illustration of the HV indicator. Right: If no point dominating the HV is found, the minimal distance to the region of interest (dotted) can be used as an additional indicator. The combined indicator is, however, not Pareto-compliant anymore.

An important property of a set-based performance measure $I : \mathbb{R}^m \to \mathbb{R}$ is *Pareto compliance*: If $A \preceq B$ and $B \not\preceq A$, then $I(A) < I(B)$ [18]. That is, a performance measure should improve if new non-dominated or (weakly) dominating solutions are added to a set of solutions. If only $I(A) \leq I(B)$ can be guaranteed under the same circumstances, $I$ is called *weakly Pareto-compliant*. Only the HV indicator and other indicators based on it are established to be Pareto-compliant [9]. The selection of weakly Pareto-compliant indicators is somewhat larger, including, for example, the (discretized) R2 [5] and the IGD+ [11] measures.

The necessity of the anti-optimal reference point can, however, be a hindrance to achieving Pareto compliance in practice. Setting the reference point so far back that it is dominated by every feasible solution introduces a bias towards the edges of the PF, while a reference point close to the nadir point fails to consider all solutions outside of such a defined region of interest, cf. Figure 1.

### 2.3 The R2 Indicator

In contrast to the HV indicator, the unary R2 indicator is ordinarily defined as the expected utility of the point set w.r.t. a distribution of utility functions $U$ [10]. In the most general case, for a set of solutions $Y$ of a MOP, we can define it as follows:

$$R2(Y) := \int_{u \in U} \min_{y \in Y} u(y) du.$$

The most common choice of a utility function is a Tchebycheff aggregation, which allows to reach all Pareto-optimal points depending on the chosen parametrization. For a weight vector $w \in [0,1]^m$ with $\sum_{i=1}^{m} w_i = 1$ and a utopian vector $y^*$, it is given by

$$\begin{aligned} u_w(y) &= \max_{i=1,\dots,m} w_i(y_i - y_i^*) \\ &= \max_{i=1,\dots,m} w_i y_i' \end{aligned}$$

using $y_i' = y_i - y_i^*$ to shift the utopian point w.l.o.g. to the origin. The distribution of utility functions is then usually chosen as uniform on the weight simplex.

In the bi-objective case, where $w_2 = w_1 - 1$ holds, this yields the following formula for $R2$:

$$\begin{aligned} R2(Y) &= \int_0^1 \min_{y \in Y} u_w(Y) dw \\ &= \int_0^1 \min_{y \in Y} \{\max(w y_1', (1-w) y_2')\} dw. \end{aligned}$$

As there is no apparent way to calculate this property directly, it is generally approximated in a discrete manner by discretizing $U$ using $n = |W|$ weight vectors $w \in W$:

$$R2(Y) \approx \frac{1}{|W|} \sum_{w \in W} \min_{y \in Y} u_w(y).$$

As an example, the uniform weight distribution with size $n$ for the bi-objective case is given by [5]

$$W = \left\{ (0,1), \left( \frac{1}{n-1}, \frac{n-2}{n-1} \right), \dots, \left( \frac{n-2}{n-1}, \frac{1}{n-1} \right), (1,0) \right\}.$$

The discretization is simultaneously a blessing and a curse: On the one hand, it provides an effective method of approximating the underlying exact $R2$ value

with high precision. On the other hand, this weakens the indicators' properties. To optimize the discretized R2 indicator, one can consider at most $|W|$ points on the Pareto front, which optimize $u_w$ for each $w \in W$, respectively. Additional nondominated solutions cannot contribute to the indicator value. Further, an individual utility function $u_w$ may be optimized by a point that is only weakly Pareto optimal, but has the same utility (for this set of weights) as another, Pareto optimal point. See, for example, the top left image of Fig. 2, where each point along the vertical lines would have identical utility values. This places the discretized R2 indicator among the weakly Pareto-compliant indicators.

A final property of the R2 indicator is that each solution $y$ from a nondominated set of solutions $Y$ is optimal in terms of utility compared to all other solutions from $Y$ for a particular weight $(w^*, 1 - w^*)$ such that [5]

$$\begin{aligned} w^* y'_1 &= (1 - w^*) y'_2 \\ \Rightarrow w^* &= \frac{y'_2}{y'_1 + y'_2}. \end{aligned}$$

Discussions around the R2 indicator and its application focus (almost) exclusively on its discrete variant [5,16] rather than on the original continuous definition of $R2$ [10]. The remainder of this paper is dedicated to a better understanding of this original definition, analyzing its properties, and detailing methods for computing it.

## 3 The R2 Indicator for Continuous Utility Distributions

In this section, we derive how the R2 indicator can be computed under the assumption of a continuous distribution of Tchebycheff utility functions for bi-objective problems. We will start by analyzing the scenario for a single solution point before extending the analysis to sets of solutions. At last, we derive the computational complexity of the presented approach.

### 3.1 R2 for a Single Solution

Without loss of generality, let $y^* = (0, 0)$ be the utopian point and $Y = \{y\}$ be the set containing only one solution $y = (y_1, y_2) > (0, 0)$. Further, let $w = (w^*, 1 - w^*)$ be the weight vector such that $w^* y_1 = (1 - w^*) y_2$. Then, for $w_1 < w^*$, $w_1 y_1 < (1 - w_1) y_2$ and for $w_1 > w^*$, $w_1 y_1 > (1 - w_1) y_2$. Based on this observation, we can compute the exact R2 indicator of $Y$ for a uniform

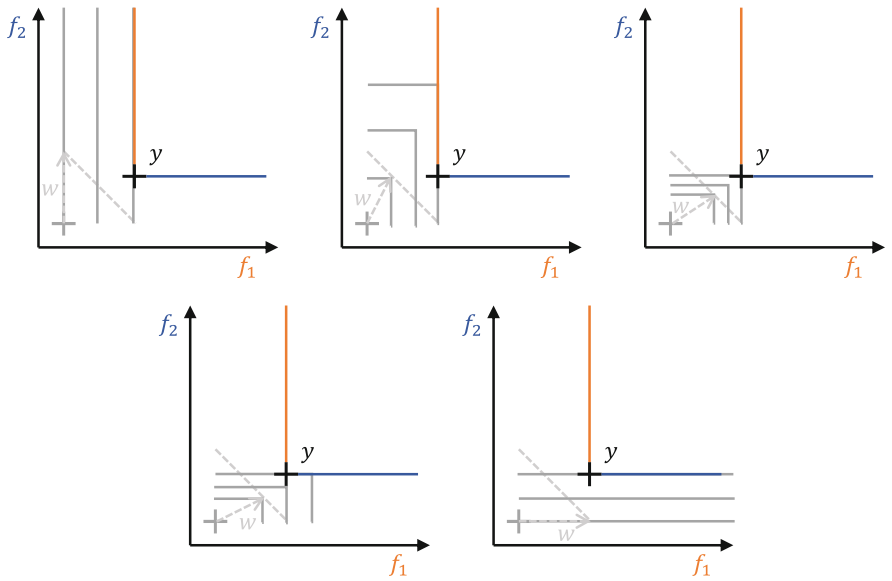



**Fig. 2.** Illustration of level sets of the Tchebycheff utility for five different weight vectors $w$. The utility value $u_w(y)$ is determined by the surface that the level set touches first: At vertical surfaces (see left and center image in the top row), the $u_w(y)$ is determined by the $f_1$ value while at horizontal surfaces (see bottom row) $u_w(y)$ depends on the $f_2$ value of $y$. At $y$ (the top right figure) both $w_1 y_1$ and $w_2 y_2$ are identical. Note: Weight vectors are illustrated to point towards their equilibrium between both objectives, i.e., $w_1 f_1 = w_2 f_2$.

distribution of Tchebycheff utility functions by splitting the integral along $w^*$:

$$\begin{aligned} R2(Y) &= \int_0^1 \max(y_1 w, y_2(1-w))dw \\ &= \int_0^{w^*} y_2(1-w)dw + \int_{w^*}^1 y_1 w dw \\ &= \left[-\frac{1}{2}y_2(1-w)^2\right]_0^{w^*} + \left[\frac{1}{2}y_1 w^2\right]_{w^*}^1 \\ &= \frac{1}{2}y_2\left(1-(1-w^*)^2\right) + \frac{1}{2}y_1\left(1-(w^*)^2\right). \end{aligned}$$

This simple case demonstrates that computing the exact R2 indicator for a set of solutions consists of the following steps: First, we need to identify areas in which the utility value does not vary w.r.t. $y$ and is only sensitive to $w$. Then, we compute the contribution of each of these areas to the final $R2$ value using the corresponding integral and finally sum everything up.

We can build on these observations to derive a general procedure to compute $R2(Y)$ for arbitrary solution sets.

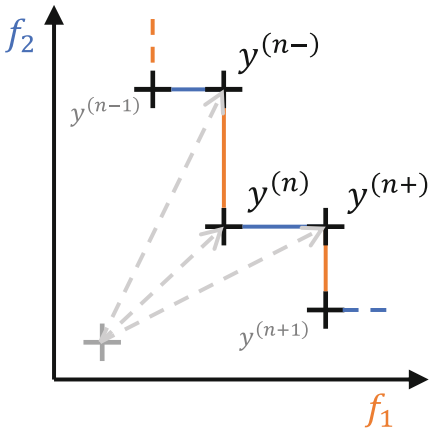



**Fig. 3.** Schematic illustration of the integration ranges of a solution $y^{(n)}$. $y^{(n-1)}$, $y^{(n)}$, and $y^{(n+1)}$ are consecutive points in the solution set. $y^{(n-)}$ and $y^{(n+)}$ correspond to the corners in the PF that are adjacent to $y^{(n)}$. Their corresponding weight vectors $w^{(n-)}$ and $w^{(n+)}$ indicate the boundaries in which $y^{(n)}$ locally determines the PF. At $w^{(n)}$, the objective switches between the $f_1$- (vertical PF segment) and $f_2$-values (horizontal PF segment) of solution $y^{(n)}$.

### 3.2 R2 for a Set of Solutions

Let us now consider what happens when our solution set contains $N > 1$ solutions. Let $Y = \{y^{(1)}, \ldots, y^{(N)}\}$ be the set of *nondominated* solutions ordered by ascending $y_1$ value. Analogous to $w^*$ above, let $w^{(n)} = (w_1^{(n)}, 1 - w_1^{(n)})$ be the weight vector such that $w_1^{(n)} y_1^{(n)} = (1 - w_1^{(n)}) y_2^{(n)}$ for all $n = 1, \ldots, N$.

The weights $w^{(n)}$ indicate when the utility of solution $y^{(n)}$ is optimal and identical w.r.t. both individual objectives. Slightly increasing (decreasing) $w_1^{(n)}$ lets the term of the first (second) objective dominate, i.e., the $w^{(n)}$ values indicate a switch in the relevant objective.

In addition, we need to identify the weight ranges for which the utility value is determined by a given solution $y^{(n)}$ rather than by one of its adjacent nondominated solutions, $y^{(n-1)}$ or $y^{(n+1)}$. As a visual aid for the following explanations, Fig. 3 provides a sketch of the involved solutions, weight vectors, and their relationships. For this purpose, let $w^{(n+)}$ be the weight vector such that $w_1^{(n+)} y_1^{(n+1)} = (1 - w_1^{(n+)}) y_2^{(n)}$. $w^{(n+)}$ represents the weight vector pointing to the intersection between $y^{(n)}$ and $y^{(n+1)}$, marked as $y^{(n+)}$ in Fig. 3, where the relevant solution switches between the two solution vectors. On the other side, $w^{(n-)}$ analogously defines the boundary between $y^{(n-1)}$ and $y^{(n)}$, which is indicated by $y^{(n-)}$ in Fig. 3. Given these definitions, a solution $y^{(n)} = (y_1^{(n)}, y_2^{(n)})$ determines the Tchebycheff utility in the interval $[w_1^{(n-)}, w_1^{(n+)}]$ for $w_1$, with the subinterval $[w_1^{(n-)}, w_1^{(n)}]$ being dependent on the $y_2$-value and $[w_1^{(n)}, w_1^{(n+)}]$ depending on its $y_1$-value.

Note also that, similarly to the HV indicator, the solution $y^{(n)}$ has an exclusive contribution corresponding to the box spanned by it with $y^{(n-)}$ and $y^{(n+)}$: The utilities $u_{w^{(n-)}}$ to $u_{w^{(n+)}}$ would worsen if $y^{(n)}$ was removed. This will still be true if we consider the case that $y^{(n)}$ would only weakly dominate one of its neighbors in the set, re-establishing Pareto compliance for the R2 indicator as it was described in its original publication [10].

Now we can compute the partial R2 contribution of $y^{(n)}$ in a similar way to the single solution set by splitting along $w_1^{(n)}$. It only differs by replacing the overall integration boundaries by $y^{(n)}$'s relevant weight range, $w_1^{(n-)}$ and $w_1^{(n+)}$, which gives:

$$
\begin{aligned}
R2(y^{(n)}) &= \int_{w_1^{(n-)}}^{w_1^{(n+)}} \max(y_1^{(n)} w, y_2^{(n)}(1-w))dw \\
&= \int_{w_1^{(n-)}}^{w_1^{(n)}} y_2^{(n)}(1-w)dw + \int_{w_1^{(n)}}^{w_1^{(n+)}} y_1^{(n)} w dw \\
&= \left[-\frac{1}{2} y_2^{(n)}(1-w)^2\right]_{w_1^{(n-)}}^{w_1^{(n)}} + \left[\frac{1}{2} y_1^{(n)} w^2\right]_{w_1^{(n)}}^{w_1^{(n+)}} \\
&= \frac{1}{2} y_2^{(n)} \left((1-w_1^{(n-)})^2 - (1-w_1^{(n)})^2\right) + \frac{1}{2} y_1^{(n)} \left((w_1^{(n+)})^2 - (w_1^{(n)})^2\right).
\end{aligned}
$$

For the special cases that $n = 1$ or $n = N$, where there is no neighbor in one direction, we can define $w_1^{(1-)} = 0$ and $w_1^{(N+)} = 1$, respectively.

Given the individual contributions $R2(y^{(n)})$ for all $n = 1, \dots N$, we can construct the exact $R2$ value of the whole solution set $Y$ simply as their sum:

$$
R2(Y) = \sum_{n=1}^{N} R2(y^{(n)}).
$$

In consequence, computing the exact R2 indicator for two objectives can essentially be achieved with a single pass along the $N$ solutions from $Y$, as will be described next.

### 3.3 Computational Complexity

The computational complexity of this approach is determined mostly by the condition that we require $Y$ to contain only nondominated solutions and be sorted by $y_1$. For this, we can first sort an archive of solutions (including dominated points) by $y_1$ and pass over this list once, removing any dominated points and duplicates. This takes $\mathcal{O}(N \log N)$ time with standard sorting algorithms.

The computation of the indicator itself is then just a matter of another pass over the sorted list of nondominated points, which requires linear time: All required $w$ and $y$ values can be obtained on the fly based on any given point $y^{(n)}$ and its immediate neighbors. To summarize, the computation of the exact $R2$

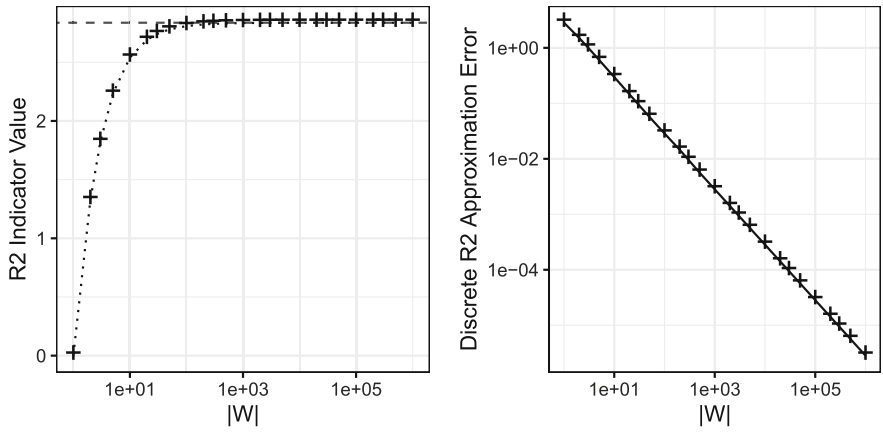



**Fig. 4.** Comparison of the approximated R2 indicator values using $|W|$ weight vectors and the exact R2 indicator value computed by our methodology. Left: The exact value is shown by the dashed line. Right: The corresponding approximation error of the discretized approach. The test set of solutions is given by $10^5$ randomly evaluated points on a bi-sphere problem from the bi-objective BBOB [4] with parameters {FID: 1, IID: 1, DIM: 2}.

indicator on an archive of $N$ points in bi-objective space requires a complexity of $\mathcal{O}(N \log N)$. This improves upon the $\mathcal{O}(N|W|)$ complexity of the discretized R2 for precise indicator values and large sets of solutions, i.e., large $|W|$ and $N$ values.

## 4 Properties of the Exact R2 Indicator

In this section, we present some empirical and theoretical results on our proposed exact R2 indicator. We start by demonstrating the approximation behaviour of the discrete R2 in relation to the exact computation described in the previous sections. Then, we will calculate the optimal $R2$ indicator values for simple front shapes and demonstrate the convergence behaviour of the indicator w.r.t. increasingly large nondominated sets.

We implemented the exact R2 indicator in the statistical software `R`. For computations of the discrete R2 indicator, we utilize the `unary_r2_indicator` function from the `emoa` package [12]. The scripts to reproduce these experimental results are published at https://github.com/schaepermeier/r2-revisited.

### 4.1 Comparison of Discrete and Exact R2 Values

We start by comparing the exact R2 indicator as computed using our method (see Sect. 3) with the discretized version found throughout the literature. As a basis for the comparison, we rely on the nondominated points found by evaluating $10^5$

solutions on a bi-sphere problem from the bi-objective BBOB [4]. We evaluated the discretized R2 with uniformly distributed weights, and the number of weights ($|W|$) ranging from one to one million. The value of the discretized indicator as well as the approximation error are visualized in Fig. 4.

We can see that, in this example, the R2 indicator seems sufficiently well approximated at around 1,000 weights. Still, more weights yield a more accurate approximation: Empirically, there seems to be an exponential relationship between the number of weights chosen and the approximation error. This reflects analyses by [5] on the behavior of the discrete R2 with an increasing number of weights, albeit missing the exact R2 values for comparison.

### 4.2 Exact R2 Indicator Values

For the exact R2 indicator, we can provide the optimal indicator values for simple solution sets and corresponding test functions. This contributes to a better understanding of the R2 indicator values, as a geometric interpretation like with the hypervolume indicator is not possible.

**Nadir and Ideal Points.** Assuming we have normalized the objective space ($(0,0)$ being the ideal point and $(1,1)$ being the nadir point), we can compute the R2 indicator for the worst possible solution within the region of interest $[0,1] \times [0,1]$ by inserting the nadir point $(1,1)$ into the equation. According to the previous results, and with $w^* = 0.5$ to fulfill $w^* y_1 = (1-w^*)y_2$, we can compute this value as follows:

$$\begin{aligned} R2(\{(1,1)\}) &= \frac{1}{2} y_2 (1 - (1-w^*)^2) + \frac{1}{2} y_1 (1 - (w^*)^2) \\ &= 0.5 \cdot 1 (1 - (1-0.5)^2) + 0.5 \cdot 1 (1 - 0.5^2) \\ &= 0.5 \cdot 0.75 + 0.5 \cdot 0.75 = \mathbf{0.75}. \end{aligned}$$

This result is independent of the particular problem or PF, as it is only dependent on the normalized nadir and ideal points. Analogously, we can derive the R2 value for the ideal point as $R2(\{(0,0)\}) = \mathbf{0}$, as all utilities equal zero at the ideal point. For comparison, the HV w.r.t. the nadir point as the reference point is $HV(\{(1,1)\}) = 0$, while the HV of the ideal point is $HV(\{(0,0)\}) = 1$ in this situation.

**Linear Front.** Let us now consider a linear PF where $y_2 = 1 - y_1$ and $y_1 \in [0,1]$ with ideal point $(0,0)$. When computing R2, for each weight vector $w$, we can find the optimal solution $y$ on the PF, which we can derive as follows:

$$w_1 y_1 = w_2 y_2 \Rightarrow w_1 y_1 = (1-w_1)(1-y_1) \Leftrightarrow y_1 = 1 - w_1.$$

Note that when we use this, it does not matter which of the objectives we consider in the R2 computation, as they will always yield the same utility value.

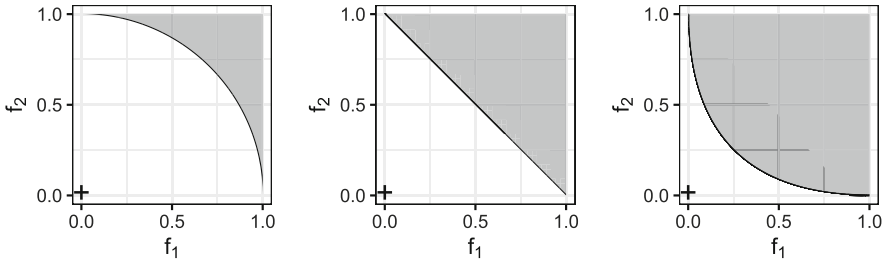



**Fig. 5.** Schematic illustration of a concave ($\frac{3}{4} < R2 < \frac{1}{6}$), linear ($R2 = \frac{1}{6}$), and convex ($0 < R2 < \frac{1}{6}$) PF, respectively. The ideal point at the origin is denoted by +, and the gray area indicates the dominated area.

Integrating over the weights for this set $Y_{\texttt{lin}}$, we get:

$$\begin{aligned} R2(Y_{\texttt{lin}}) &= \int_0^1 \min_{y \in Y_{\texttt{lin}}} \{\max(wy_1, (1-w)y_2)\} dw \\ &= \int_0^1 wy_1 dw = \int_0^1 w(1-w)dw = \int_0^1 (w - w^2)dw \\ &= \left[\frac{1}{2}w^2 - \frac{1}{3}w^3\right]_0^1 \\ &= \frac{1}{2} - \frac{1}{3} = \mathbf{\frac{1}{6}} \approx \mathbf{0.1667}. \end{aligned}$$

Based on this result, we can derive that the optimal R2 indicator value for the DTLZ1 problem [7], which possesses a linear PF with ideal point $(0, 0)$ and nadir point $(0.5, 0.5)$, is $\frac{1}{12} \approx 0.0833$.

**Convex and Concave Fronts.** Additionally, we can derive value ranges for general concave and convex PFs: A concave front $Y_{\texttt{conc}}$ will always achieve worse utility values than a linear function, and a convex front $Y_{\texttt{conv}}$ will always have better utility. Again considering the normalized objective space, a general concave front has an ideal R2 value between the value of the linear front and the nadir point's value, i.e., $\frac{1}{6} < R2(Y_{\texttt{conc}}) < \frac{3}{4}$. Analogously, a convex front $Y_{\texttt{conv}}$ will always fall between the R2 values of the ideal point and the linear front, i.e., $0 < R2(Y_{\texttt{conv}}) < \frac{1}{6}$. All cases are illustrated in Fig. 5. If none of the discussed conditions apply, the ideal R2 indicator value of the normalized objective space may lie anywhere between 0 and 0.75.

As shown above, exact R2 indicator values can be derived for certain analytical PF shapes. Following the same pattern, that is, resolving $w_1 y_1 = (1 - w_1)y_2$ and integrating the utility function, we can compute the exact indicator values also for more complex PF shapes, albeit in a less straightforward manner. We limit ourselves to reporting the results for simple quadratic PF functions, which correspond to the PFs in Fig. 5:

- Convex PF with $y_2 = (1 - \sqrt{y_1})^2$: $\frac{3\pi - 8}{16} \approx 0.0890$
- Concave PF with $y_2 = \sqrt{1 - y_1^2}$: $\frac{1}{8}\left(3\sqrt{2}\sinh^{-1}(1) - 2\right) \approx 0.2174$

The result for the convex PF applies, e.g., for the classical bi-sphere problem, while the concave PF corresponds to DTLZ2 [7].

To summarize, we can compute exact R2 indicator values for different simple front shapes and individual points. While these values do not seem to have an intuitive (geometric) interpretation, they are rather given meaning by the expected utility to a decision-maker.

## 5 Conclusion

In this paper, we introduce a procedure to compute the exact R2 indicator value for a given solution set. In contrast to its widely-known and commonly used discrete counterpart, the exact R2 indicator is not just weakly Pareto-compliant, but a proper Pareto-compliant indicator. This is achieved by foregoing the discretization of the distribution of utility functions usually performed in the calculation of the indicator and taking a continuous, uniform distribution of Tchebycheff utility functions as the basis. Pareto compliance of the R2 indicator with this utility distribution was already described when it was introduced by [10], but missing instructions for its exact calculation. Further, we show how this indicator is implemented efficiently with a running time of $\mathcal{O}(N \log N)$ for bi-objective solution sets of size $N$. This positions the exact R2 indicator as a promising Pareto-compliant alternative to the hypervolume indicator, especially when a utopian rather than an anti-optimal reference point is naturally available.

We demonstrate the approximation behaviour of the commonly used, discretized R2 indicator in comparison with the exact computation, and provide optimal R2 indicator values for the ideal and nadir points, as well as a linear front in normalized objective space. From this, we derive R2 indicator value ranges for general convex and concave PFs as well.

We expect that the exact R2 indicator offers multiple directions for further theoretical and empirical research. So far, we have only demonstrated how the R2 indicator is computed for bi-objective problems. A natural further research direction pertains to the computation of the indicator for more than two objectives. We do not expect that the R2 indicator will provide a runtime advantage over the hypervolume indicator, however, schemes to approximate it are until now the state-of-the-art in its computation and therefore very accessible compared to HV approximations. We also believe that an incremental variant of the indicator can be designed, which would make it ideal for benchmarking applications, where an indicator should be computed iteratively for the whole archive of solutions.

From a theoretical point of view, we see potential in analyzing the approximation quality of the discretized R2 depending on the number of weights used. This could be particularly interesting for giving quality guarantees for R2 in higher-dimensional objective spaces, where exact indicator computations may

become computationally intractable. A deeper theoretically supported analysis of its properties, along the lines of [5], would also be interesting. Further, the effects of different utility functions as well as the integration of (decision-maker) preferences can be examined, as both directions have been studied in detail for the discretized R2 indicator [15]. Finally, connections between the R2 indicator and the integrated preference functional [3,6], a parallel development of an indicator very similar to R2 in the operations research community, should be further investigated, and could yield improvements in understanding the R2 indicator's properties and computation.

Differences and similarities in the preferred distributions between the exact R2 indicator and the hypervolume indicator when applying them in a benchmarking context could present a promising research direction. Likewise, integrating the exact R2 in optimization heuristics, similar to R2-EMOA [14], may be the subject of future studies.

**Acknowledgments.** The work of Lennart Schäpermeier and Pascal Kerschke was supported by the Center for Scalable Data Analytics and Artificial Intelligence (ScaDS.AI) Dresden/Leipzig.

## References

1. Audet, C., Bigeon, J., Cartier, D., Le Digabel, S., Salomon, L.: Performance indicators in multiobjective optimization. Eur. J. Oper. Res. **292**(2), 397–422 (2021)
2. Beume, N., Fonseca, C.M., López-Ibáñez, M., Paquete, L., Vahrenhold, J.: On the complexity of computing the hypervolume indicator. IEEE Trans. Evol. Comput. **13**(5), 1075–1082 (2009). https://doi.org/10.1109/TEVC.2009.2015575
3. Bozkurt, B., Fowler, J.W., Gel, E.S., Kim, B., Köksalan, M., Wallenius, J.: Quantitative comparison of approximate solution sets for multicriteria optimization problems with weighted Tchebycheff preference function. Oper. Res. **58**(3), 650–659 (2010)
4. Brockhoff, D., Auger, A., Hansen, N., Tusar, T.: Using well-understood single-objective functions in multiobjective black-box optimization test suites. Evol. Comput. **30**(2), 165–193 (2022). https://doi.org/10.1162/EVCO_A_00298
5. Brockhoff, D., Wagner, T., Trautmann, H.: On the properties of the R2 indicator. In: Proceedings of the 14th Annual Conference on Genetic and Evolutionary Computation, pp. 465–472 (2012)
6. Carlyle, W.M., Fowler, J.W., Gel, E.S., Kim, B.: Quantitative comparison of approximate solution sets for bi-criteria optimization problems. Decis. Sci. **34**(1), 63–82 (2003)
7. Deb, K., Thiele, L., Laumanns, M., Zitzler, E.: Scalable test problems for evolutionary multiobjective optimization. In: Abraham, A., Jain, L., Goldberg, R. (eds.) Evolutionary Multiobjective Optimization, pp. 105–145. Springer, London (2005). https://doi.org/10.1007/1-84628-137-7_6
8. Grimme, C., et al.: Peeking beyond peaks: challenges and research potentials of continuous multimodal multi-objective optimization. Comput. Oper. Res. **136**, 105489 (2021)
9. Guerreiro, A.P., Fonseca, C.M., Paquete, L.: The hypervolume indicator: computational problems and algorithms. ACM Comput. Surv. **54**(6), 119:1–119:42 (2022). https://doi.org/10.1145/3453474

10. Hansen, M.P., Jaszkiewicz, A.: Evaluating the quality of approximations to the non-dominated set, Department of Mathematical Modelling, Technical University of Denmark, IMM (1998)
11. Ishibuchi, H., Masuda, H., Tanigaki, Y., Nojima, Y.: Modified distance calculation in generational distance and inverted generational distance. In: Gaspar-Cunha, A., Henggeler Antunes, C., Coello, C.C. (eds.) Evolutionary Multi-Criterion Optimization, pp. 110–125. Springer, Cham (2015). https://doi.org/10.1007/978-3-319-15892-1_8
12. Mersmann, O.: emoa: evolutionary multiobjective optimization algorithms, R package version 0.5-0.2 (2023). https://CRAN.R-project.org/package=emoa
13. Miettinen, K.: Nonlinear Multiobjective Optimization. Springer, Boston, MA (1998). https://doi.org/10.1007/978-1-4615-5563-6
14. Trautmann, H., Wagner, T., Brockhoff, D.: R2-EMOA: focused multiobjective search using R2-indicator-based selection. In: Nicosia, G., Pardalos, P. (eds.) Learning and Intelligent Optimization, pp. 70–74. Springer, Berlin, Heidelberg (2013). https://doi.org/10.1007/978-3-642-44973-4_8
15. Purshouse, R.C., Fleming, P.J., Fonseca, C.M., Greco, S., Shaw, J. (eds.): EMO 2013. LNCS, vol. 7811. Springer, Heidelberg (2013). https://doi.org/10.1007/978-3-642-37140-0
16. Zitzler, E., Knowles, J., Thiele, L.: Quality assessment of Pareto set approximations. In: Branke, J., Deb, K., Miettinen, K., Słowiński, R. (eds.) Multiobjective Optimization, pp. 373–404. Springer, Berlin, Heidelberg (2008). https://doi.org/10.1007/978-3-540-88908-3_14
17. Zitzler, E., Thiele, L.: Multiobjective optimization using evolutionary algorithms — a comparative case study. In: Eiben, A.E., Bäck, T., Schoenauer, M., Schwefel, H.-P. (eds.) PPSN V, pp. 292–301. Springer, Berlin, Heidelberg (1998). https://doi.org/10.1007/BFb0056872
18. Zitzler, E., Thiele, L., Laumanns, M., Fonseca, C.M., da Fonseca, V.G.: Performance assessment of multiobjective optimizers: an analysis and review. IEEE Trans. Evol. Comput. **7**(2), 117–132 (2003). https://doi.org/10.1109/TEVC.2003.810758

# Influence Maximization in Hypergraphs Using Multi-Objective Evolutionary Algorithms

Stefano Genetti, Eros Ribaga, Elia Cunegatti, Quintino F. Lotito, and Giovanni Iacca(✉)

University of Trento, Trento, Italy
{stefano.genetti,eros.ribaga}@studenti.unitn.it,
{elia.cunegatti,quintino.lotito,giovanni.iacca}@unitn.it

**Abstract.** The Influence Maximization (IM) problem is a well-known NP-hard combinatorial problem over graphs whose goal is to find the seed set of nodes in a network that spreads influence at most. Among the various methods for solving the IM problem, evolutionary algorithms (EAs) have been shown to be particularly effective. While the literature on the topic is particularly ample, only a few attempts have been made at solving the IM problem over higher-order networks, namely extensions of standard graphs that can capture interactions that involve more than two nodes. Hypergraphs are a valuable tool for modeling complex interaction networks in various domains; however, they require rethinking of several graph-based problems, including IM. In this work, we propose a multi-objective EA for the IM problem over hypergraphs, aiming at minimizing the seed set size while maximizing influence. Smart initialization and hypergraph-aware mutation operators are utilized to facilitate algorithm convergence. While the existing methods rely on greedy or heuristic methods, to our best knowledge this is the first attempt at applying EAs to this problem. Our results over nine real-world datasets and three propagation models, compared with five baseline algorithms, reveal that our method achieves in most cases state-of-the-art results in terms of hypervolume and solution diversity.

**Keywords:** Influence Maximization · Hypergraphs · Evolutionary Algorithm · Multi-Objective Optimization · Higher-order Networks

## 1 Introduction

Networks provide a valuable framework to model and analyze systems of interacting unities. Networks are typically represented as a graph, namely, a collection of nodes (the units of the system) connected via edges (the interactions between those units). Given their flexibility, networks have found applications in several domains, from the study of human behavior and cellular interactions to the

S. Genetti and E. Ribaga—Equal contribution.


M. Affenzeller et al. (Eds.): PPSN 2024, LNCS 15151, pp. 217–235, 2024.
https://doi.org/10.1007/978-3-031-70085-9_14

assessment of the resilience and efficiency of technological systems [12]. However, conventional graph models may fail at capturing the full complexity and heterogeneity characterizing real-world networks. In fact, while empirical interactions can be described by many complex features (e.g., direction, weight, temporality, etc.), standard graphs usually associate only one feature with each edge. Moreover, graphs can only encode pairwise interactions, oversimplifying systems characterized by higher-order interactions, i.e., group interactions among three or more units [7,8]. Examples of such systems are scientific collaborations [52], people's face-to-face encounters [18], and the brain [54]. In order to model such higher-order interactions, hypergraphs [11], rather than standard graphs, are needed. Hypergraphs are a generalization of graphs in which interactions are encoded into sets of arbitrary size, i.e., the hyperedges.

At the interplay between network structure and dynamics, a popular problem over graphs is the so-called Influence Maximization (IM). In this problem, the aim is to select a set of nodes from which the influence can be spread at most over the network [36]. Solving this problem exactly has been proven to be NP-hard. IM has been mainly studied in single-objective formulation, i.e., given a predefined number of nodes to be picked as starting seeds, the only objective is to maximize the spread (i.e., the number of influenced nodes). On the other hand, recent works [14,25] deviated from this formulation, showing that setting the seed set size cardinality *a priori* inherently restricts the solution space explored by the optimization process, therefore proposing multi-objective settings in which the seed size has to be minimized while maximizing influence propagation.

IM has been extensively studied in the context of standard graphs, yet, its application to higher-order networks is still limited. In hypergraphs, influence can spread through groups, impacting multiple units simultaneously and leading to non-linear behaviors. Hence, dynamical processes on hypergraphs are a more accurate model for many complex real-world dynamics, such as social influence in groups of friends, that are oversimplified by graph representations [5,8]. However, the expressive power of hypergraphs comes at the cost of having to generalize traditional graph problems and algorithms to the higher-order case. In this direction, solving the IM problem in hypergraphs would allow the analysis of data that inherently represent a hypergraph (e.g., scientific collaborations), for which propagation models are directly defined as higher-order, and that cannot be run on top of standard graphs. Moreover, IM on hypergraphs would also allow for better modeling systems previously studied with graph approximation of higher-order dynamics, aiming for better identification of influential nodes.

In this work, we propose a Higher-Order Network Multi-Objective Evolutionary Algorithm (in short, HN-MOEA), the first algorithm that employs Evolutionary Computation to solve the IM problem over higher-order networks. We also increase the problem complexity by designing a bi-objective formulation where the influence spread (to maximize) and the number of nodes in the starting seed set (to minimize) are jointly optimized. We design our method by adapting, for higher-order networks, a state-of-the-art Evolutionary Algorithm (EA) for IM [15,24], also including IM-specific techniques such as smart

initialization [37] and graph-aware mutations [24] to further boost the evolutionary process. We compare HN-MOEA w.r.t. the most recent baselines for IM on higher-order networks over three different propagation models, showing how our proposed method always shows comparable or better performance both in terms of hypervolume and solution diversity. In summary:

- We propose HN-MOEA, a multi-objective EA designed to solve the IM problem over higher-order networks;
- We adapt smart initialization and hypergraph-aware mutations usually designed for standard graphs to be hypergraph-dependent;
- We test our approach w.r.t. to two standard (i.e., non-hypergraph-specific) baseline methods as well as three recent IM algorithms specifically designed for higher-order networks, over three propagation models;
- We show how our approach not only provides, in general, higher hypervolumes, but also finds sets of non-dominated solutions that are inherently more diverse than those found by the compared methods.

## 2 Background

Recently, there has been a growing interest in characterizing hypergraphs, from micro-scale patterns [39,46,47], to core-periphery organization [60], community structure [21,48,56], backboning [49], and centrality measures [9]. This is mainly motivated by the fact that hypergraphs can encode, without losing information, systems that display higher-order interactions, thus providing new insights into those systems' behavior. Formally, a *hypergraph* is an ordered pair $H(V, E)$, where $V = \{v_1, \ldots, v_n\}$ is the set of nodes and $E = \{e_1, \ldots, e_m\}$ is the set of *hyperedges*. Each hyperedge $e \in E$ is a set of nodes with a cardinality of at least 2, i.e., $e \subseteq V$ and $|e| \geq 2$. For any given node $v \in V$, the set $E(v) \in E$ refers to the collection of hyperedges containing $v$. Additionally, a node $u$ belongs to the set of neighbors $\mathcal{N}(v)$ of a node $v$ if there exists at least one $e \in E$ such that $e$ contains both $v$ and $u$ ($E(v) \cap E(u) \neq \emptyset$). The *degree* $d$ of a node $v$ corresponds to the cardinality of its set of neighbors, expressed as $d(v) = |\mathcal{N}(v)|$. Whereas, the *hyperdegree* $d^H$ of a node $v$ is the number of hyperedges to which $v$ belongs.

### 2.1 Propagation Models

Higher-order interactions in complex systems can significantly alter propagation dynamics [7,8]. Therefore, understanding how higher-order interactions affect different dynamical processes (e.g., contagion [20,34] or synchronization [29]) previously studied in the traditional graph setting is attracting interest. In this work, our focus is on influence propagation.

In standard graphs, influence propagation is typically modeled as an iterative process in which, given a graph $(V, E)$, at each iteration (timestep $t$), each node in $V$ can be either *active* (i.e., it has been influenced) or *inactive*. The influence spread starts at $t_0$ from a set of *seed nodes* $S \subseteq V$, which are all active, while all the other nodes in the graph are not. Then, at each timestep, each of the

active nodes can influence one or more of its neighbors, according to different logics (e.g., based on a certain probability) that depend on the given *propagation model*. It is typically assumed that, once a node becomes active, it cannot become inactive anymore. Hence, the set of active nodes in $V$ increases monotonously over the timesteps, until the spreading process ends.

This general propagation process applies also to the case of hypergraphs of the form $H(V, E)$: what changes, is only the propagation model under which such process occurs. However, while propagation models devised for standard graphs are well-established, the literature on propagation models in higher-order networks is still limited. Developing propagation models tailored for hypergraphs that account for the complexity of hyperedges is nevertheless crucial for understanding and predicting influence propagation in diverse real-world scenarios that would be oversimplified with a traditional network representation and lower-order dynamics [7,8].

In our experiments, we consider three different propagation models, namely the Weighted Cascade (WC) [36], a model commonly adopted in the case of standard graphs and generalized to the higher-order domain in [69], as well as two recently-introduced hypergraph-specific models, referred to as Susceptible-Infected Contact Process (SICP) [63] and Linear Threshold (LT) [67].

① **Weighted Cascade (WC).** At each timestep $t \geq 1$, each node $n$ active at time $t-1$ may activate some of its inactive neighbors $m$ with non-uniform probability inversely proportional to the number of neighbors of $m$, i.e., the probability of $a \rightarrow b$ is given by $\frac{1}{d(b)}$.

② **Susceptible-Infected Contact Process (SICP).** At each timestep $t$, for each active node $n$, we consider all the hyperedges $E_i = \{e_{i1}, e_{i2}, \ldots e_{iq}\}$ in which node $n$ participates. At this point, a hyperedge $e$ is sampled from $E_i$ uniformly at random. Then, each of the inactive nodes in $e$ is influenced by node $n$ with probability $p$, i.e., the probability of $a \rightarrow b$ is given by $p$.

③ **Linear Threshold (LT).** Let us consider a hyperedge $e$ with $A$ be the set of nodes in active state. In this scenario, a hyperedge $e$ becomes activated if the fraction of activated nodes $\frac{|A|}{|e|}$ is greater than or equal to the configured threshold value $\theta \in (0, 1)$. Upon activation of $e$, all nodes $v$ belonging to $e$ also transition to an active state in the subsequent step of the propagation process. For instance, let $e_i = \{v_{i1}, v_{i2}, v_{i3}, v_{i4}\}$ denote a hyperedge comprising four nodes. Suppose that, at time $t$, the set $A_t$ includes $v_{i1}, v_{i3}$. Then, at time $t+1$, if the fraction of active nodes in $e_i$ exceeds $\theta$, $e_i$ will be activated. Consequently, all currently inactive vertices of $e_i$, namely $v_{i2}$ and $v_{i4}$, will also become active.

We illustrate the considered propagation models in Fig. 1. In all these models, the propagation of influence terminates either upon reaching convergence, indicated by no further nodes being activated in the last timestep, or upon reaching a specified maximum number of timesteps $\tau$ (maximum number of hops). Due to the stochasticity that characterizes the WC and SICP propagation models, the influence propagation is computed through multiple Monte Carlo simulations. On the other hand, the LT model has been designed to provide a deterministic execution [67], without requiring multiple simulations.

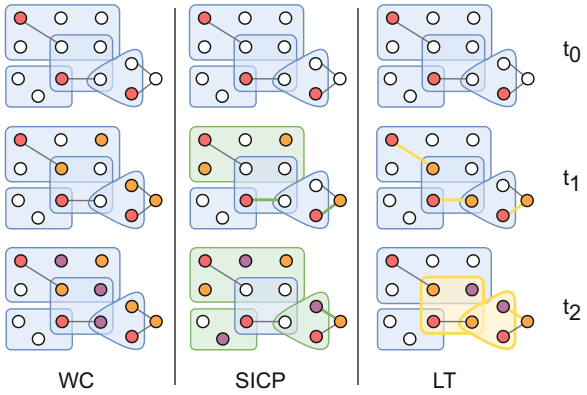



**Fig. 1.** Graphical representation of the WC, SICP, and LT models. Each colored polygon indicates a hyperedge. The first row represents the hypergraph at time $t_0$, where only the nodes of the seed set (Red) are activated. The second row shows the nodes that are activated at timestep $t_1$ (Orange), while the bottom row depicts the nodes that are activated at timestep $t_2$ (Purple). The second column displays the hyperedges that SICP randomly selects to spread the influence (Green). The third column highlights the activation of hyperedges based on the LT propagation model, assuming a threshold value of 0.5 (Yellow). (Color figure online)

### 2.2 Influence Maximization Problem

Given a seed set $S$, its influence, denoted as $\sigma(S)$, is the (expected, in the case of stochastic propagation models) size of the set of active nodes at the end of the influence propagation process. Introduced in [27] and further formalized as a combinatorial optimization problem in [36], the IM problem aims to identify the seed set of nodes $S$ in the network that maximizes the number of influenced nodes, i.e., $S = argmax_S\{\sigma(S)\}$. In the traditional formulation, the number of nodes $|S|$ in the seed set is predefined. As detailed in Sect. 4, in this work we highlight the importance of including $|S|$ as an additional objective of the optimization problem, to direct the search towards valuable trade-offs between *effort* (number of seed nodes) and *effect* (final influence over the whole network). Of note, such bi-objective formulation also leads to higher solution diversity among different seed set sizes, as we will demonstrate in the Results section.

## 3 Related Work

The existing literature on IM is mainly focused on standard graphs [32,43,50,55]. Over the years, researchers have explored various algorithms to address this problem [6,41,42,53]. Among the proposed solutions, EAs and other forms of metaheuristics [23,28,38,44,59,66] have demonstrated remarkable effectiveness in tackling this kind of combinatorial optimization problem [1]. For instance, [31] proposed the local influence estimation (LIE) function, which considers the influence within the 2-hop neighborhood of seed nodes, and optimized it using the

Discrete Particle Swarm Optimization (DPSO) algorithm. In [35], the authors introduced the expected diffusion value (EDV) evaluation function and utilized Simulated Annealing (SA) to identify the most influential nodes. Other works have showcased the efficacy of single-objective [13] and multi-objective EAs [14,15] in outperforming alternative approaches both in terms of quality and execution times [25]. Lastly, [24] proposed a many-objective formulation aimed at maximizing the influence spread while minimizing the size of the seed set, along with other objectives such as the propagation time, the influence fairness, or the cost of propagation.

While the IM problem has been extensively studied in the context of lower-order interactions, IM in the higher-order case remains relatively unexplored. In principle, the IM problem in higher-order networks can be solved by applying existing IM algorithms designed for standard graphs on the hypergraph's clique-expanded graph, i.e., a graph representation in which edges connect vertices that are part of the same hyperedge in the original hypergraph. Although executing an IM algorithm on clique-expanded graphs is a viable strategy for identifying influential sources, it is important to recognize that this process inevitably sacrifices several crucial topological features of the original hypergraph data structure [62]. In more depth, in a higher-order network, multiple interactions could potentially be shared by two neighboring nodes. On the other hand, in its low-order graph counterpart, each pair of nodes can only have one pairwise interaction. This distinction significantly impacts the spread of the influence across the network [8]. Moreover, the insofar proposed propagation models for hypergraphs are specifically tailored for higher-order networks, and may not perform well when applied to networks with only dyadic interactions.

Given that IM is NP-hard also on higher-order networks [69], existing works rely on greedy [5,61,68] or heuristic strategies [64,67] to explore the search space within reasonable computational time. Overall, these methods evaluate the suitability of each node as a source of influence spread by assigning it a score, and incrementally add nodes with the highest marginal benefit to the seed set. These techniques have effectively addressed the IM problem across various real-world higher-order network datasets. However, as we will show in our experiments, they are characterized by limited exploration capabilities and resulting solution diversity. In contrast to standard graphs, no prior work has yet explored the resolution of the IM problem in hypergraphs using EAs, which instead can provide better exploration and diversity.

## 4 Methodology

In agreement with the methodology originally suggested in [14], our formulation of the IM problem does not enforce any *a priori* constraint on the cardinality of the seed set. Instead, the multi-objective formulation in this study aims to maximize the influence spread while minimizing the size of the seed set $|S|$. Following this methodology, given an input hypergraph $H(V, E)$, the genotype of an individual $x$ generated throughout the evolutionary process encodes a set of

nodes $S \subseteq V$ of variable size in $\{1, 2, \ldots, k\}$, representing the seeds of influence in the network. Each node is indicated by its id, i.e., an integer in $\{0, 1, \ldots, |V|-1\}$. The fitness of a candidate solution $x$ is a tuple containing: (i) the influence $\sigma(x)$ of the seed set, calculated as described in Sect. 2.2, to be maximized $\uparrow$; (ii) the size $k$ of the seed set $S$, to be minimized $\downarrow$. Both values are normalized w.r.t. the network size, to allow comparisons between networks of different sizes.

The multi-objective EA of choice in this work is NSGA-II [26], which has been proven to be successful on the IM problem in standard graphs, outperforming in most cases the alternative heuristics [25]. Moreover, this method can be easily extended to incorporate additional objective functions into the optimization process, as shown in [24]. We also use the smart initialization strategies proposed in [19,24,37], aiming at accelerating algorithm execution and guiding population convergence towards prominent regions of the solution space. In line with the strategy adopted in [14,15,25], parent solutions are selected with fixed-size tournament and elitism. The offspring solutions are generated by standard one-point crossover, while for mutation we took inspiration from the graph-based mutation presented in [24,37]. For individual replacement, we rely on the standard NSGA-II replacement mechanism consisting of non-dominated sorting followed by crowding distance preference.

**Smart Initialization.** Generating an initial population situated within prominent regions of the solution space is a practice commonly adopted in Evolutionary Computation [16,17,33] in order to facilitate convergence towards profitable fitness landscape regions. This approach has been proven to be effective also on the IM problem over standard graphs [24,37]. Hence, we decided to adopt the same approach also on the hypergraphs handled in this paper. The rationale behind this approach is rooted in the observation that nodes characterized by high centrality are likely to be effective sources of influence spread.

Our smart initialization over higher-order networks works as follows: initially, given a hypergraph $H(V, E)$, we sort the nodes in the set $V$ based on their degree. Subsequently, we select the top $\lambda$ percentage of nodes with the highest degree, hence obtaining a filtered set of nodes $\bar{V}$ (with $\lambda$ being a hyperparameter). Half of the initial population consists of a set of nodes sampled from the filtered set $\bar{V}$, with probabilities proportional to their degrees. In order to favor diversity in the population, the other half comprises seed sets of nodes chosen uniformly at random from the entire node set $V$. To further promote diversity, each individual's genotype in the initial population is initialized with a randomly selected number of nodes, ranging from $k_{\min}$ to $k_{\max}$ (both are hyperparameters).

**Hypergraph-Aware Mutation.** While the use of random mutation and one-point crossover leads to remarkable performance on the IM problem [15,25], introducing hypergraph-aware mutation operators can guide the evolutionary process towards even better results [24,37]. Hence, we rely on a combination of stochastic and hypergraph-aware mutation operators with complementary effects, in order to strike a balance between exploration and exploitation (as found in preliminary experiments not reported for brevity). Each individual is mutated according to one of the two mutation operators, selected uniformly at random.

① **Stochastic mutation.** Given an individual $x$ (i.e., a seed set), with a genotype consisting of $l$ genes, this mutation performs either: (1) *node replacement*, which generates a new individual $x'$ by randomly replacing one of the genes of $x$ with a node $n \notin x$; (2) *node insertion*, which generates a new individual $x'$ with $l+1$ genes by adding to $x$ a new node $n \notin x$; or (3) *node removal*, which generates a new individual $x'$ with $l-1$ genes, by randomly removing from $x$ a node $n \in x$. The three strategies are chosen at random with uniform probability.

② **Hypergraph-aware mutation.** This mutation leverages the intrinsic characteristics of nodes. In this case, we only consider *node replacement*. Given again an individual $x$ of size $l$, first, we select the gene to be replaced with a probability inversely proportional to its corresponding node degree. Then, we choose (with equal probability) the new node $n \notin x$ either from the entire collection of nodes $V$, or from the neighbors of the gene selected for replacement. In both cases, the new node is chosen with probability proportional to its degree.

## 5 Experimental Setup

In this section, we provide an overview of the experimental design employed to evaluate the efficacy of HN-MOEA, as well as the tested baselines.

**Computational Setup.** We performed our experiments on two Ubuntu 20.04 workstations, respectively with a 28-core Intel i9-7940X CPU @ 3.10 GHz and 64 GB RAM, and a 36-core Intel i9-10980XE CPU @ 3.00 GHz and 128 GB RAM. The total execution time of our experiments was in the order of (approximately) 400 CPU core hours. The code implementing our methods is completely written in Python and is made publicly available[1]. Furthermore, it has been integrated into the Hypergraphx library [45].

**Datasets.** In order to evaluate the effectiveness of our proposed method, and to allow for a direct comparison with other methods for IM on hypergraphs, we performed an experimental analysis on nine publicly available real-world higher-order networks[2] used in related works. The selected datasets represent empirical hypergraphs from three heterogeneous domain categories spanning social networks, online reviews, and email communication. These datasets cover a wide range of different topological properties, i.e., number of nodes, number of hyperedges, and density. Each dataset has been properly pre-processed to remove duplicated hyperedges, duplicated nodes within the same higher-order interaction, and relations populated by less than two entities. Summary statistics of the datasets, after pre-processing, are available in Table 1.

**Baselines.** We compare our proposed HN-MOEA with five other algorithms for IM in higher-order networks. It is important to remark that: (1) no other multi-objective approaches exist for this problem, and (2) the only existing baselines are inherently single-objective and, by construction, cannot be turned into

[1] https://github.com/DIOL-UniTN/hn-moea-im.git.

[2] https://www.cs.cornell.edu/~arb/data/.

**Table 1.** Datasets tested in our experimental setup, divided by category: social (Algebra, Geometry, MAG-10), online reviews (Restaurant, Music, Bars), and email communication (Email-eu, Email-enron, Email-w3c).

| Dataset | Nodes | Hyperedges | Hyperdegree | | | Degree | | | Source |
|---|---|---|---|---|---|---|---|---|---|
| | | | Avg. | Std. | Max. | Avg. | Std. | Max. | |
| Algebra | 423 | 980 | 17.52 | 29.93 | 328 | 78.89 | 68.38 | 303 | [4] |
| Geometry | 580 | 888 | 19.90 | 32.69 | 227 | 164.79 | 121.62 | 474 | [4] |
| MAG-10 | 80198 | 51889 | 2.25 | 4.56 | 187 | 5.91 | 9.19 | 335 | [3,58] |
| Restaurant | 565 | 594 | 8.11 | 7.17 | 59 | 79.75 | 59.82 | 310 | [4] |
| Music | 1106 | 686 | 9.47 | 10.72 | 127 | 167.87 | 107.92 | 865 | [51] |
| Bars | 1234 | 1188 | 9.60 | 7.36 | 146 | 174.30 | 145.02 | 818 | [4] |
| Email-eu | 1000 | 78919 | 259.12 | 340.84 | 2386 | 280.44 | 217.53 | 755 | [10,40,65] |
| Email-enron | 4423 | 5734 | 6.80 | 32.05 | 1139 | 25.34 | 43.96 | 934 | [10] |
| Email-w3c | 14317 | 19821 | 3.07 | 23.90 | 958 | 4.06 | 23.98 | 959 | [2,22] |

multi-objective. Among our baselines, two are non-hypergraph-specific (RANDOM and HIGH-DEGREE), while the other three (HDD, HCI-1, and HCI-2) are specific for hypergraphs. Furthermore, four of the compared methods are deterministic (HIGH-DEGREE, HDD, HCI-1, and HCI-2) while RANDOM is stochastic. In the following, we consider a hypergraph $H(V, E)$ and a maximum seed set size $k_{\max}$.

The RANDOM algorithm simply generates $k_{\max}$ seed sets, with sizes ranging from $k_{\min}$ to $k_{\max}$, by iteratively adding a node randomly sampled from $V$.

In the HIGH-DEGREE approach [63], nodes in $V$ are sorted according to their degree. The output Pareto Front (i.e., the set of non-dominated solutions) comprises candidate seed sets $S_1, \ldots, S_{k_{\max}}$ of increasing sizes from $k_{\min}$ to $k_{\max}$. For each set size $i$, the top $k_i$ nodes with the highest degree are selected ($S_i = argmax_{S' \subseteq V, |S'|=k_i} \sum_{v \in S'} d(v)$).

Along with these two non-hypergraph-specific baselines, we include in our experiments three of the most recent IM algorithms for hypergraphs proposed in the literature. Specifically, we consider HDD, proposed in [63], as well as the HCI-1 and HCI-2 algorithms introduced in [67]. As said, all these methods are meant for single-objective optimization. Hence, to compare them with HN-MOEA we executed them for every value of seed set size $k$ within the interval $[k_{\min}, k_{\max}]$.

**Hyperparameter Setting.** To strike a balance between computational efficiency and accurately capturing the influence spread dynamics, we set the maximum number of hops within which influence is propagated to $\tau = 5$ as in [24,30], which reflects a fair compromise between the commonly adopted 2-hop approximation and an unbounded spread process (i.e., $\tau = \infty$).

To limit the hyperparameter dependency for the SICP model, rather than relying on a constant value for the probability $p$, we sample $p$ at each timestep uniformly at random within $[0.005, 0.02]$ (values commonly used in [63]). Regard-

ing the LT propagation model, the spread of influence is extremely sensitive to the threshold $\theta$ and it is not feasible to identify a value for this parameter that is suitable for every dataset. Therefore, we opted for a different threshold for each network. Specifically, we adopted the values utilized in [67]: $\theta = 0.8$ for Algebra and Geometry, $\theta = 0.5$ for MAG-10, $\theta = 0.6$ for Music and Bars. For the remaining datasets, we employed the approach outlined in [67], wherein the parameter $\theta$ is tailored to the specific characteristics of each dataset. Following this strategy, through empirical investigation, we determined $\theta = 0.7$ for Restaurant and Email-enron, $\theta = 0.6$ for Email-w3c, and $\theta = 0.8$ for Email-eu.

Concerning the EA parameters, as proposed in [15,24] we set the minimum and the maximum seed set size of an individual to $k_{\min} = 1$ and $k_{\max} = 100$ respectively. The parameter $\lambda$ used in our smart initialization strategy has been set to $\lambda = 30\%$. For all the experiments, the evolutionary hyperparameters have been kept fixed, setting the population size to 100, number of offspring to 100, number of elites to 2, tournament size to 5, and generations to 100 (as in [24]). When evaluating the fitness of the HN-MOEA solutions w.r.t. WC and SICP, we conduct 100 Monte Carlo simulations of the propagation model, while the LT model [67] being deterministic does not require multiple evaluations. To deal with the inherent stochasticity of HN-MOEA and the RANDOM baseline, the results presented below for these two algorithms have been aggregated from 5 independent runs, providing a more robust and reliable assessment of outcomes. Conversely, due to their deterministic nature, the results for the other baselines (HIGH-DEGREE, HDD, HCI-1, HCI-2) are based on a single execution.

The hyperparameters of the HCI-1 and HCI-2 algorithms [67] have been carefully fine-tuned. Indeed, due to the specific characteristics of HCI-1 and HCI-2, the selection of certain parameters significantly impacts their ability to identify seed sets for different values of $k$ within the range of interest, namely $[1, 100]$. In more depth, we adjusted the hyperedge threshold parameter in their source code to 0.85, which ensures fair results across all datasets analyzed in our study.

## 6 Results

We analyze the performance of HN-MOEA w.r.t. the compared algorithms for IM both in terms of hypervolume and solution diversity.

**Performance in Terms of Hypervolume.** We compare our proposed approach and the baselines both qualitatively and quantitatively. Figure 2 displays a qualitative representation of the performance attained by the evaluated IM algorithms on two selected datasets, considering all three influence propagation models examined in this study. Remarkably, in most cases the Pareto Fronts generated by HN-MOEA demonstrate superior performance compared to the solutions obtained by other algorithms, achieving more favorable trade-offs between seed set size and percentage of influenced nodes. However, in consonance with the observations from [14], for some datasets, NSGA-II struggles in populating the Pareto Front for solutions with node counts approaching the upper bound of the

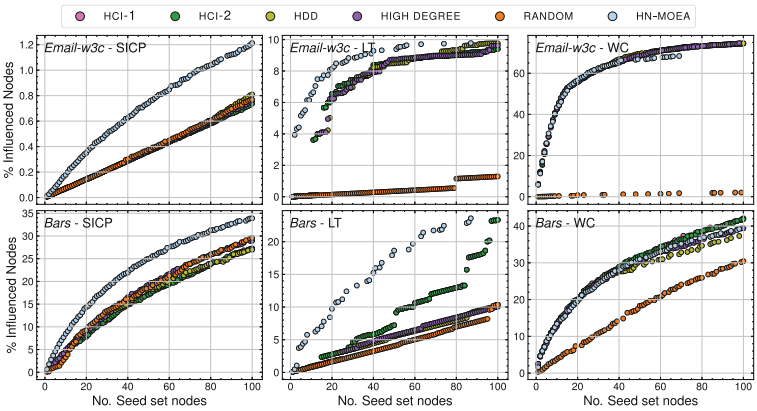



**Fig. 2.** Results obtained by the compared IM algorithms on the Email-w3c and Bars datasets, using WC, SICP, and LT as influence propagation models.

seed set size. This might be due to the fact that, while for larger $k$ there exist less possible combinations, finding them becomes harder.

Table 2 provides a quantitative comparison of the algorithmic solutions by computing the hypervolume [57] (i.e., the area under the curve of the Pareto Front found by the various methods on each dataset and propagation model). Here, it can be seen that HN-MOEA excels particularly within propagation models tailored for the higher-order domain (SICP and LT), often outperforming competitors by a significant margin. On the other hand, in the case of a propagation model initially intended for standard graphs (WC), our algorithm's performance does not exhibit a notable enhancement compared to the evaluated baselines. Nevertheless, in these instances, the quality of the solutions proposed by our method aligns closely with that of other algorithms. These observations hold considerable importance in practical applications, as real-world scenarios often encompass diverse influence propagation patterns. Consequently, it is essential to develop algorithmic solutions capable of producing valuable outcomes across a wide range of propagation models. Furthermore, HN-MOEA demonstrates remarkable versatility not only with respect to the propagation model but also across the diverse datasets analyzed in this work, being able to effectively converge towards profitable solutions regardless of the peculiarities inherent to the different network domains. However, it is worth noting that its performance on the MAG-10 dataset is slightly less satisfactory. This can be attributed to the large number of nodes and hyperedges present in this dataset, resulting in a significantly expanded solution space. Using more generations or larger population sizes could potentially yield better outcomes.

**Performance in Terms of Solution Diversity.** As introduced before, the bi-formulation of IM problem leads to avoiding having an incremental Pareto Front (as usual in single-objective formulation), providing more diversity in the solution seed sets of the final Pareto Front. Hence, we further enhance our analysis

**Table 2.** Hypervolumes achieved by the compared algorithms. Results for RANDOM and HN-MOEA are cross 5 independent runs (mean ± std. dev.). The boldface indicates the highest hypervolume per dataset and propagation model. We highlight the cases where our method achieves the highest hypervolume.

| Dataset | Algorithm | Higher-Order Propagation | | Standard Propagation |
|---|---|---|---|---|
| | | SICP | LT | WC |
| Algebra | HIGH-DEGREE | 1.79e−01 | 4.67e−01 | 4.96e−01 |
| | HDD | 1.84e−01 | 1.65e−01 | 4.41e−01 |
| | HCI-1 | 1.67e−01 | 4.73e−01 | 4.69e−01 |
| | HCI-2 | 1.72e−01 | 3.35e−01 | 4.37e−01 |
| | RANDOM | 2.15e−01 ± 5.85e−03 | 1.29e−01 ± 1.58e−02 | 3.53e−01 ± 1.10e−02 |
| | HN-MOEA | **2.96e−01 ± 7.86e−04** | **5.28e−01 ± 9.10e−03** | **5.01e−01 ± 1.12e−03** |
| Geometry | HIGH-DEGREE | 2.39e−01 | 2.04e−01 | **4.34e−01** |
| | HDD | 2.28e−01 | 1.10e−01 | 3.73e−01 |
| | HCI-1 | 2.22e−01 | 3.06e−01 | 4.24e−01 |
| | HCI-2 | 2.25e−01 | 2.50e−01 | 4.08e−01 |
| | RANDOM | 3.29e−01 ± 7.09e−03 | 8.74e−02 ± 7.31e−04 | 3.06e−01 ± 6.04e−03 |
| | HN-MOEA | **4.59e−01 ± 1.86e−03** | **3.06e−01 ± 1.04e−02** | 4.32e−01 ± 8.67e−04 |
| MAG-10 | HIGH-DEGREE | 7.44e−04 | 2.63e−01 | 3.73e−02 |
| | HDD | 7.45e−04 | 2.65e−01 | **3.81e−02** |
| | HCI-1 | 7.33e−04 | **2.76e−01** | 3.72e−02 |
| | HCI-2 | 7.33e−04 | **2.76e−01** | 3.71e−02 |
| | RANDOM | 7.42e−04 ± 9.25e−06 | 2.18e−02 ± 1.43e−02 | 2.31e−03 ± 1.33e−04 |
| | HN-MOEA | **1.30e−03 ± 1.65e−05** | 2.62e−01 ± 2.73e−03 | 2.76e−02 ± 8.97e−04 |
| Restaurant | HIGH-DEGREE | 1.62e−01 | 1.38e−01 | 4.27e−01 |
| | HDD | 1.69e−01 | 8.95e−02 | 3.97e−01 |
| | HCI-1 | 1.51e−01 | 1.98e−01 | **4.33e−01** |
| | HCI-2 | 1.51e−01 | 1.89e−01 | 4.32e−01 |
| | RANDOM | 1.72e−01 ± 2.93e−03 | 8.98e−02 ± 2.78e−04 | 3.06e−01 ± 8.39e−03 |
| | HN-MOEA | **2.18e−01 ± 1.30e−03** | **2.54e−01 ± 1.35e−02** | 4.31e−01 ± 1.00e−03 |
| Music | HIGH-DEGREE | 1.65e−01 | 1.80e−01 | 3.10e−01 |
| | HDD | 1.98e−01 | 6.98e−02 | 2.89e−01 |
| | HCI-1 | 1.38e−01 | **3.38e−01** | 2.98e−01 |
| | HCI-2 | 1.38e−01 | 3.37e−01 | 2.99e−01 |
| | RANDOM | 2.10e−01 ± 6.88e−03 | 5.92e−02 ± 1.74e−02 | 1.95e−01 ± 6.97e−03 |
| | HN-MOEA | **2.94e−01 ± 9.13e−04** | 3.21e−01 ± 2.16e−02 | **3.10e−01 ± 5.28e−04** |
| Bars | HIGH-DEGREE | 1.77e−01 | 5.40e−02 | 2.81e−01 |
| | HDD | 1.67e−01 | 5.25e−02 | 2.70e−01 |
| | HCI-1 | 1.62e−01 | 8.79e−02 | **2.94e−01** |
| | HCI-2 | 1.62e−01 | 8.79e−02 | 2.93e−01 |
| | RANDOM | 1.72e−01 ± 9.54e−03 | 4.52e−02 ± 3.78e−03 | 1.73e−01 ± 6.29e−03 |
| | HN-MOEA | **2.28e−01 ± 4.04e−04** | **1.54e−01 ± 1.44e−02** | 2.82e−01 ± 1.16e−03 |
| Email-eu | HIGH-DEGREE | 6.03e−02 | 7.90e−01 | **3.48e−01** |
| | HDD | 6.47e−02 | 7.91e−01 | 2.93e−01 |
| | HCI-1 | 5.87e−02 | 8.05e−01 | 2.91e−01 |
| | HCI-2 | 5.79e−02 | 8.22e−01 | 2.50e−01 |
| | RANDOM | 6.25e−02 ± 5.57e−04 | 5.28e−02 ± 4.10e−03 | 2.19e−01 ± 8.82e−03 |
| | HN-MOEA | **7.25e−02 ± 5.21e−04** | **8.36e−01 ± 1.29e−02** | 3.40e−01 ± 1.21e−03 |
| Email-enron | HIGH-DEGREE | 1.58e−02 | 2.41e−01 | 3.63e−01 |
| | HDD | 1.69e−02 | 2.45e−01 | **3.95e−01** |
| | HCI-1 | 1.59e−02 | **2.60e−01** | 3.85e−01 |
| | HCI-2 | 1.69e−02 | 2.41e−01 | 3.56e−01 |
| | RANDOM | 1.94e−02 ± 5.38e−04 | 1.14e−02 ± 0.00e+00 | 5.26e−02 ± 5.49e−03 |
| | HN-MOEA | **2.85e−02 ± 2.15e−04** | 2.29e−01 ± 4.12e−03 | 3.91e−01 ± 4.03e−03 |
| Email-w3c | HIGH-DEGREE | 3.72e−03 | 7.13e−02 | 6.32e−01 |
| | HDD | 3.80e−03 | 7.21e−02 | **6.35e−01** |
| | HCI-1 | 3.67e−03 | 7.32e−02 | 6.31e−01 |
| | HCI-2 | 3.67e−03 | 7.32e−02 | 6.31e−01 |
| | RANDOM | 3.76e−03 ± 7.57e−05 | 3.53e−03 ± 4.34e−19 | 1.14e−02 ± 8.84e−04 |
| | HN-MOEA | **7.03e−03 ± 4.29e−05** | **8.79e−02 ± 1.94e−03** | 6.04e−01 ± 8.58e−03 |

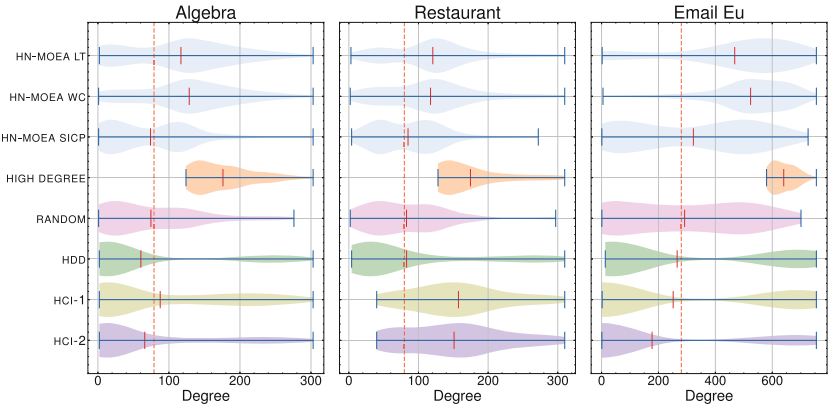



**Fig. 3.** Violin plot w.r.t the Pareto Front of HN-MOEA and baselines. The dotted red lines correspond to the avg. degree of the hypergraph. Red lines inside the violin plots correspond to the avg. degree of the solutions in the Pareto Front. (Color figure online)

by comparing the characteristics of the nodes comprising the seed sets proposed by the IM algorithms under investigation. In this regard, Fig. 3 highlights the *topological diversity* in terms of the degree distribution of the nodes included in solutions found by each algorithm on three selected datasets (one per category). In the case of HN-MOEA, we report the results for each of the three propagation models, while for the baselines the results do not depend on the propagation model as the seed set is built only based on the properties of the hypergraph. Upon examining the figure, we notice that HN-MOEA incorporates nodes spanning a wide spectrum of degree values, which suggests that the EA benefits from the ability to explore the solution space without rigid adherence to specific greedy properties, as done in some of the compared heuristics. This usually leads to better results, although the effectiveness of node properties can vary depending on the peculiarities of the network at hand. In this regard, a node metric that works well as an indicator of a promising influence source in one dataset or region of a hypergraph may perform poorly in other contexts. For instance, high centrality does not invariably denote optimal influence propagation sources. As an example, bridge nodes linking distinct communities might have few neighbors, yet they are crucial for propagating influence.

In addition, Table 3 reports the *population diversity* and *node diversity* within the solutions in the Pareto Front found by each algorithm. Note again that the solutions obtained by the baselines do not depend on the propagation model and as such they are all characterized by the same diversity. Hence, a single value is sufficient to represent all the baselines.

*Population diversity* refers to the extent to which individuals in the Pareto Front exhibit distinct genotypes. In our context, achieving a high level of population diversity relates to having minimal overlap between the seed sets of different

individuals. Given a Pareto Front $\mathcal{P}$, where each individual $x_i \in \mathcal{P}$ is a seed set $x_i = \{v_1, v_2, \ldots, v_{|x|}\}$, the population diversity $D : \mathcal{P} \to \mathcal{R}$ is computed as:

$$D(\mathcal{P}) = 1 - \frac{1}{|\mathcal{P}|(|\mathcal{P}|-1)} \sum_{x_i \in \mathcal{P}} \sum_{x_j \in \mathcal{P}, i \neq j} \frac{|x_i \cap x_j|}{|x_i|}. \tag{1}$$

*Node diversity*, instead, measures the percentage of unique nodes within the seed sets in the Pareto Front. A high node diversity indicates that the Pareto Front does not comprise nodes confined to a few regions of the network; rather, the proposed solutions represent a wide array of possible spread sources from various hypergraph locations. The node diversity $ND : \mathcal{P} \to \mathcal{R}$ is calculated as:

$$ND(\mathcal{P}) = \frac{\left|\bigcup_{x_i \in \mathcal{P}} x_i\right|}{\sum_{x_i \in \mathcal{P}} |x_i|} \tag{2}$$

As shown in the table, the inherent superior ability of HN-MOEA to effectively explore the complex search space leads to a Pareto Front that not only encompasses nodes with heterogeneous topological features but also demonstrates higher population and node diversity compared to the solutions obtained by algorithms that construct seed sets by iteratively adding nodes.

**Table 3.** Population and node diversity achieved by the compared algorithms. Algorithms that incrementally construct seed sets by adding nodes to maximize a specific objective function inherently generate Pareto Fronts that exhibit the same values of node and population diversity. The boldface indicates the highest population and node diversity per dataset and propagation model.

| Dataset | Diversity | HN-MOEA- LT | HN-MOEA- WC | HN-MOEA- SICP | Baselines |
|---|---|---|---|---|---|
| Algebra | Population | **5.10e−01 ± 4.00e−02** | 3.43e−01 ± 9.59e−03 | 3.89e−01 ± 5.06e−03 | 2.50e−01 |
| | Node | **9.30e−02 ± 2.66e−03** | 4.39e−02 ± 1.39e−03 | 4.79e−02 ± 1.08e−03 | 2.00e−02 |
| Geometry | Population | **5.54e−01 ± 3.50e−02** | 3.74e−01 ± 9.62e−03 | 3.68e−01 ± 1.10e−02 | 2.50e−01 |
| | Node | **9.78e−02 ± 1.60e−02** | 4.71e−02 ± 3.13e−03 | 5.59e−02 ± 4.78e−03 | 2.00e−02 |
| MAG-10 | Population | 3.50e−01 ± 2.45e−02 | 3.36e−01 ± 1.64e−02 | **3.99e−01 ± 5.23e−03** | 2.50e−01 |
| | Node | **9.41e−02 ± 7.18e−03** | 7.99e−02 ± 4.81e−03 | 5.80e−02 ± 6.23e−03 | 2.00e−02 |
| Restaurant | Population | **5.16e−01 ± 3.71e−02** | 3.48e−01 ± 2.53e−03 | 4.43e−01 ± 1.85e−02 | 2.50e−01 |
| | Node | **9.11e−02 ± 7.34e−03** | 5.40e−02 ± 3.86e−03 | 4.77e−02 ± 3.33e−03 | 2.00e−02 |
| Music | Population | **5.79e−01 ± 3.93e−02** | 3.61e−01 ± 5.58e−03 | 4.52e−01 ± 3.87e−02 | 2.50e−01 |
| | Node | **1.47e−01 ± 1.90e−02** | 5.61e−02 ± 4.44e−03 | 5.04e−02 ± 4.56e−03 | 2.00e−02 |
| Bars | Population | **5.66e−01 ± 2.80e−02** | 3.85e−01 ± 1.42e−02 | 5.20e−01 ± 1.70e−02 | 2.50e−01 |
| | Node | **1.61e−01 ± 4.58e−02** | 6.90e−02 ± 2.84e−03 | 6.02e−02 ± 4.83e−03 | 2.00e−02 |
| Email-eu | Population | **6.32e−01 ± 6.14e−02** | 4.41e−01 ± 1.36e−02 | 4.75e−01 ± 1.63e−02 | 2.50e−01 |
| | Node | **2.58e−01 ± 3.60e−02** | 5.21e−02 ± 3.13e−03 | 5.89e−02 ± 4.36e−03 | 2.00e−02 |
| Email-enron | Population | **4.72e−01 ± 1.90e−02** | 3.11e−01 ± 7.30e−03 | 4.57e−01 ± 2.11e−02 | 2.50e−01 |
| | Node | **1.37e−01 ± 1.78e−02** | 7.83e−02 ± 9.24e−03 | 5.59e−02 ± 2.61e−03 | 2.00e−02 |
| Email-w3c | Population | **4.30e−01 ± 1.87e−02** | 2.87e−01 ± 1.06e−02 | 3.94e−01 ± 2.10e−02 | 2.50e−01 |
| | Node | **1.25e−01 ± 2.97e−02** | 7.95e−02 ± 4.50e−03 | 5.02e−02 ± 1.71e−03 | 2.00e−02 |

## 7 Conclusions

In this paper, we presented HN-MOEA, a multi-objective EA for IM in higher-order networks. To the best of our knowledge, this work marks the first attempt at employing EAs in this domain. Our method aims to minimize the seed set size $|S|$ while maximizing the expected influence, and includes smart initialization and hypergraph-aware mutations to improve convergence and performance.

The method has been evaluated on nine different real-world datasets characterized by heterogeneous properties, with three different propagation models. Our experimental analysis demonstrates that HN-MOEA overall outperforms current state-of-the-art algorithms for IM in higher-order networks. In line with previous findings [14,15,25], these results confirm that EAs are particularly well-suited for solving discrete optimization problems of this nature.

One notable advantage of HN-MOEA lies in its flexibility to maximize influence across various propagation models. Moreover, in contrast to other heuristic methods, HN-MOEA explores the solution space without any bias towards specific node metrics and greedy properties. As a result, the evolutionary process better explores the search space that characterizes the IM problem. Because of the bi-objective formulation, the resulting population also exhibits higher individual diversity, with candidate solutions comprising nodes of varied properties.

Future research could focus on many-objective IM, as recently done in [24] in the context of standard graphs. Moreover, in this study, we employed hypergraph-aware mutations chosen at random. However, the selection of the optimal mutation operator may vary depending on the characteristics of the dataset and the evolutionary stage. Hence, future research could investigate the use of adaptive mutation operators to select the most suitable mutation operator based on feedback from the search process.

**Disclosure of Interests.** The authors have no competing interests to declare that are relevant to the content of this article.

## References

1. Aghaee, Z., Ghasemi, M.M., Beni, H.A., Bouyer, A., Fatemi, A.: A survey on meta-heuristic algorithms for the influence maximization problem in the social networks. Computing **103**, 2437–2477 (2021)
2. Amburg, I., Kleinberg, J., Benson, A.R.: Planted hitting set recovery in hypergraphs. J. Phys. Complex. **2**(3), 035004 (2021)
3. Amburg, I., Veldt, N., Benson, A.R.: Clustering in graphs and hypergraphs with categorical edge labels. In: Proceedings of the Web Conference (2020)
4. Amburg, I., Veldt, N., Benson, A.R.: Fair clustering for diverse and experienced groups. arXiv arXiv:2006.05645 (2020)
5. Antelmi, A., Cordasco, G., Spagnuolo, C., Szufel, P.: Social influence maximization in hypergraphs. Entropy **23**(7), 796 (2021)
6. Banerjee, S., Jenamani, M., Pratihar, D.K.: A survey on influence maximization in a social network. Knowl. Inf. Syst. **62**, 3417–3455 (2020)

7. Battiston, F., et al.: The physics of higher-order interactions in complex systems. Nat. Phys. **17**(10), 1093–1098 (2021)
8. Battiston, F., et al.: Networks beyond pairwise interactions: structure and dynamics. Phys. Rep. **874**, 1–92 (2020)
9. Benson, A.R.: Three hypergraph eigenvector centralities. SIAM J. Math. Data Sci. **1**(2), 293–312 (2019)
10. Benson, A.R., Abebe, R., Schaub, M.T., Jadbabaie, A., Kleinberg, J.: Simplicial closure and higher-order link prediction. Proc. Natl. Acad. Sci. **115**(48), E11221–E11230 (2018)
11. Berge, C.: Graphs and Hypergraphs. North-Holland Publishing, Co. (1973)
12. Boccaletti, S., Latora, V., Moreno, Y., Chavez, M., Hwang, D.U.: Complex networks: structure and dynamics. Phys. Rep. **424**(4–5), 175–308 (2006)
13. Bucur, D., Iacca, G.: Influence maximization in social networks with genetic algorithms. In: Squillero, G., Burelli, P. (eds.) EvoApplications 2016. LNCS, vol. 9597, pp. 379–392. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-31204-0_25
14. Bucur, D., Iacca, G., Marcelli, A., Squillero, G., Tonda, A.: Multi-objective evolutionary algorithms for influence maximization in social networks. In: Squillero, G., Sim, K. (eds.) EvoApplications 2017. LNCS, vol. 10199, pp. 221–233. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-55849-3_15
15. Bucur, D., Iacca, G., Marcelli, A., Squillero, G., Tonda, A.: Improving multi-objective evolutionary influence maximization in social networks. In: Sim, K., Kaufmann, P. (eds.) EvoApplications 2018. LNCS, vol. 10784, pp. 117–124. Springer, Cham (2018). https://doi.org/10.1007/978-3-319-77538-8_9
16. Caraffini, F., Iacca, G., Neri, F., Picinali, L., Mininno, E.: A CMA-ES super-fit scheme for the re-sampled inheritance search. In: 2013 IEEE Congress on Evolutionary Computation, pp. 1123–1130 (2013)
17. Caraffini, F., et al.: Super-fit multicriteria adaptive differential evolution. In: 2013 IEEE Congress on Evolutionary Computation, pp. 1678–1685 (2013)
18. Cencetti, G., Battiston, F., Lepri, B., Karsai, M.: Temporal properties of higher-order interactions in social networks. Sci. Rep. **11**(1), 1–10 (2021)
19. Chawla, A., Cheney, N.: Neighbor-hop mutation for genetic algorithm in influence maximization. In: Proceedings of the Companion Conference on Genetic and Evolutionary Computation, pp. 187–190 (2023)
20. Chowdhary, S., Kumar, A., Cencetti, G., Iacopini, I., Battiston, F.: Simplicial contagion in temporal higher-order networks. J. Phys. Complex. **2**(3), 035019 (2021)
21. Contisciani, M., Battiston, F., De Bacco, C.: Inference of hyperedges and overlapping communities in hypergraphs. Nat. Commun. **13**(1), 1–10 (2022)
22. Craswell, N., de Vries, A.P., Soboroff, I.: Overview of the TREC 2005 enterprise track. In: TREC, vol. 5, pp. 199–205 (2005)
23. Cui, L., et al.: DDSE: a novel evolutionary algorithm based on degree-descending search strategy for influence maximization in social networks. J. Netw. Comput. Appl. **103**, 119–130 (2018)
24. Cunegatti, E., Custode, L.L., Iacca, G.: Many-objective evolutionary influence maximization: Balancing spread, budget, fairness, and time. arXiv arXiv:2403.18755 (2024)
25. Cunegatti, E., Iacca, G., Bucur, D.: Large-scale multi-objective influence maximisation with network downscaling. In: Rudolph, G., Kononova, A.V., Aguirre, H., Kerschke, P., Ochoa, G., Tušar, T. (eds.) Parallel Problem Solving from Nature – PPSN XVII, PPSN 2022. LNCS, vol. 13399. Springer, Cham (2022). https://doi.org/10.1007/978-3-031-14721-0_15

26. Deb, K., Pratap, A., Agarwal, S., Meyarivan, T.: A fast and elitist multiobjective genetic algorithm: NSGA-II. IEEE Trans. Evol. Comput. **6**(2), 182–197 (2002)
27. Domingos, P., Richardson, M.: Mining the network value of customers. In: Proceedings of the Seventh ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 57–66 (2001)
28. Du, Y.H., Qiu, W.J., Chen, W.N.: Influence maximization with reverse influence sampling and evolutionary algorithm. In: 2023 IEEE International Conference on Systems, Man, and Cybernetics (SMC), pp. 783–790. IEEE (2023)
29. Gambuzza, L.V., et al.: Stability of synchronization in simplicial complexes. Nat. Commun. **12**(1), 1–13 (2021)
30. Gong, H., Guo, C.: Influence maximization considering fairness: a multi-objective optimization approach with prior knowledge. Expert Syst. Appl. **214**, 119138 (2023)
31. Gong, M., Yan, J., Shen, B., Ma, L., Cai, Q.: Influence maximization in social networks based on discrete particle swarm optimization. Inf. Sci. **367**, 600–614 (2016)
32. Goyal, A., Lu, W., Lakshmanan, L.V.: CELF++ optimizing the greedy algorithm for influence maximization in social networks. In: Proceedings of the 20th International Conference Companion On World Wide Web, pp. 47–48 (2011)
33. Iacca, G., Mallipeddi, R., Mininno, E., Neri, F., Suganthan, P.N.: Super-fit and population size reduction in compact differential evolution. In: 2011 IEEE Workshop on Memetic Computing (MC), pp. 1–8 (2011)
34. Iacopini, I., Petri, G., Barrat, A., Latora, V.: Simplicial models of social contagion. Nat. Commun. **10**(1), 1–9 (2019)
35. Jiang, Q., Song, G., Gao, C., Wang, Y., Si, W., Xie, K.: Simulated annealing based influence maximization in social networks. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 25, pp. 127–132 (2011)
36. Kempe, D., Kleinberg, J., Tardos, É.: Maximizing the spread of influence through a social network. In: Proceedings of the Ninth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 137–146 (2003)
37. Konotopska, K., Iacca, G.: Graph-aware evolutionary algorithms for influence maximization. In: Genetic and Evolutionary Computation Conference (GECCO) Companion, July 2021, pp. 1467–1475. ACM, New York, NY, USA (2021)
38. Krömer, P., Nowaková, J.: Guided genetic algorithm for information diffusion problems. In: 2018 IEEE Congress on Evolutionary Computation (CEC), pp. 1–8. IEEE (2018)
39. Lee, G., Ko, J., Shin, K.: Hypergraph motifs: concepts, algorithms, and discoveries. Proc. VLDB Endow. **13**(12), 2256–2269 (2020)
40. Leskovec, J., Kleinberg, J., Faloutsos, C.: Graph evolution: densification and shrinking diameters. ACM Trans. Knowl. Discov. Data **1**(1), 2 (2007)
41. Li, Y., Gao, H., Gao, Y., Guo, J., Wu, W.: A survey on influence maximization: from an ML-based combinatorial optimization. ACM Trans. Knowl. Discov. Data **17**(9), 1–50 (2023)
42. Li, Y., Fan, J., Wang, Y., Tan, K.L.: Influence maximization on social graphs: a survey. IEEE Trans. Knowl. Data Eng. **30**(10), 1852–1872 (2018)
43. Ling, C., et al .: Deep graph representation learning and optimization for influence maximization. In: International Conference on Machine Learning, pp. 21350–21361. PMLR (2023)
44. Lotf, J.J., Azgomi, M.A., Dishabi, M.R.E.: An improved influence maximization method for social networks based on genetic algorithm. Phys. A **586**, 126480 (2022)

45. Lotito, Q.F., et al.: Hypergraphx: a library for higher-order network analysis. J. Complex Netw. **11**(3), cnad019 (2023)
46. Lotito, Q.F., Musciotto, F., Battiston, F., et al.: Exact and sampling methods for mining higher-order motifs in large hypergraphs. Computing **106**, 475–494 (2024). https://doi.org/10.1007/s00607-023-01230-5
47. Lotito, Q.F., Musciotto, F., Montresor, A., Battiston, F.: Higher-order motif analysis in hypergraphs. Commun. Phys. **5**(1), 79 (2022)
48. Lotito, Q.F., Musciotto, F., Montresor, A., Battiston, F.: Hyperlink communities in higher-order networks. J. Complex Netw. **12**(2), cnae013 (2024)
49. Musciotto, F., Battiston, F., Mantegna, R.N.: Detecting informative higher-order interactions in statistically validated hypergraphs. Commun. Phys. **4**(1), 1–9 (2021)
50. Nguyen, H.T., Thai, M.T., Dinh, T.N.: Stop-and-stare: Optimal sampling algorithms for viral marketing in billion-scale networks. In: Proceedings of the 2016 International Conference on Management of Data, pp. 695–710 (2016)
51. Ni, J., Li, J., McAuley, J.: Justifying recommendations using distantly-labeled reviews and fine-grained aspects. In: Conference on Empirical Methods in Natural Language Processing and the International Joint Conference on Natural Language Processing (EMNLP-IJCNLP), pp. 188–197. Association for Computational Linguistics, Hong Kong, China (2019)
52. Patania, A., Petri, G., Vaccarino, F.: The shape of collaborations. EPJ Data Sci. **6**, 1–16 (2017)
53. Peng, S., Zhou, Y., Cao, L., Yu, S., Niu, J., Jia, W.: Influence analysis in social networks: a survey. J. Netw. Comput. Appl. **106**, 17–32 (2018)
54. Petri, G., et al.: Homological scaffolds of brain functional networks. J. R. Soc. Interface **11**(101), 20140873 (2014)
55. Rahimkhani, K., Aleahmad, A., Rahgozar, M., Moeini, A.: A fast algorithm for finding most influential people based on the linear threshold model. Expert Syst. Appl. **42**(3), 1353–1361 (2015)
56. Ruggeri, N., Contisciani, M., Battiston, F., De Bacco, C.: Community detection in large hypergraphs. Sci. Adv. **9**(28), eadg9159 (2023)
57. Shang, K., Ishibuchi, H., He, L., Pang, L.M.: A survey on the hypervolume indicator in evolutionary multiobjective optimization. IEEE Trans. Evol. Comput. **25**(1), 1–20 (2020)
58. Sinha, A., et al .: An overview of Microsoft Academic Service (MAS) and applications. In: International Conference on World Wide Web (WWW). ACM, New York, NY, USA (2015)
59. Tang, X., Liu, X.: Improved evolution algorithm that guides the direction of individual mutation for influence maximization in social networks. In: Qiu, H., Zhang, C., Fei, Z., Qiu, M., Kung, S.-Y. (eds.) KSEM 2021. LNCS (LNAI), vol. 12817, pp. 532–545. Springer, Cham (2021). https://doi.org/10.1007/978-3-030-82153-1_44
60. Tudisco, F., Higham, D.J.: Core-periphery detection in hypergraphs. SIAM J. Math. Data Sci. **5**(1), 1–21 (2023)
61. Wang, H., Pan, Q., Tang, J.: HEDV-Greedy: an advanced algorithm for influence maximization in hypergraphs. Mathematics **12**(7), 1041 (2024)
62. Wang, Y., Kleinberg, J.: From graphs to hypergraphs: hypergraph projection and its remediation (2024)
63. Xie, M., Zhan, X.X., Liu, C., Zhang, Z.K.: Influence maximization in hypergraphs (2022)
64. Xie, M., Zhan, X.X., Liu, C., Zhang, Z.K.: An efficient adaptive degree-based heuristic algorithm for influence maximization in hypergraphs. Inf. Process. Manage. **60**(2), 103161 (2023)

65. Yin, H., Benson, A.R., Leskovec, J., Gleich, D.F.: Local higher-order graph clustering. In: ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD), pp. 555–564. ACM, New York, NY, USA (2017)
66. Zhang, K., Du, H., Feldman, M.W.: Maximizing influence in a social network: improved results using a genetic algorithm. Phys. A **478**, 20–30 (2017)
67. Zhang, R., Qu, X., Zhang, Q., Xu, X., Pei, S.: Influence maximization based on threshold models in hypergraphs. Chaos Interdisc. J. Nonlinear Sci. **34**(2), 023111 (2024)
68. Zheng, H., Wang, N., Wu, J.: Non-submodularity and approximability: influence maximization in online social networks. In: 2019 IEEE 20th International Symposium on "A World of Wireless, Mobile and Multimedia Networks" (WoWMoM), pp. 1–9. IEEE (2019)
69. Zhu, J., Zhu, J., Ghosh, S., Wu, W., Yuan, J.: Social influence maximization in hypergraph in social networks. IEEE Trans. Netw. Sci. Eng. **6**(4), 801–811 (2018)

# Biased Pareto Optimization for Subset Selection with Dynamic Cost Constraints

Dan-Xuan Liu[1,2] and Chao Qian[1,2(✉)]

[1] National Key Laboratory for Novel Software Technology, Nanjing University, Nanjing, China
{liudx,qianc}@lamda.nju.edu.cn
[2] School of Artificial Intelligence, Nanjing University, Nanjing, China

**Abstract.** Subset selection with cost constraints aims to select a subset from a ground set to maximize a monotone objective function without exceeding a given budget, which has various applications such as influence maximization and maximum coverage. In real-world scenarios, the budget, representing available resources, may change over time, which requires that algorithms must adapt quickly to new budgets. However, in this dynamic environment, previous algorithms either lack theoretical guarantees or require a long running time. The state-of-the-art algorithm, POMC, is a Pareto optimization approach designed for static problems, lacking consideration for dynamic problems. In this paper, we propose BPODC, enhancing POMC with biased selection and warm-up strategies tailored for dynamic environments. We focus on the ability of BPODC to leverage existing computational results while adapting to budget changes. We prove that BPODC can maintain the best known $(\alpha_f/2)(1-e^{-\alpha_f})$-approximation guarantee when the budget changes. Experiments on influence maximization and maximum coverage show that BPODC adapts more effectively and rapidly to budget changes, with a running time that is less than that of the static greedy algorithm.

**Keywords:** Subset selection · Dynamic cost constraints · Pareto optimization · Multi-objective evolutionary algorithms

## 1 Introduction

The subset selection problem is a general NP-hard problem, which has a wide range of applications, such as influence maximization [12], maximum coverage [8] and sensor placement [13], to name a few. The goal is to select a subset $X$ from a ground set $V$ to optimize a given function $f$, subject to a cost constraint. Specifically, the cost of the subset $X$ must not exceed a given budget $B$. This can be formally expressed as follows:

$$\arg\max_{X\subseteq V} f(X) \quad \text{s.t.} \quad c(X)\leq B, \tag{1}$$

where the objective function $f: 2^V \to \mathbb{R}$ and the cost function $c: 2^V \to \mathbb{R}$ are both monotone, meaning they do not decrease with the addition of elements to the set $X$.

This work was supported by the National Science and Technology Major Project (2022ZD0116600) and National Science Foundation of China (62276124). Chao Qian is the corresponding author.


M. Affenzeller et al. (Eds.): PPSN 2024, LNCS 15151, pp. 236–251, 2024.
https://doi.org/10.1007/978-3-031-70085-9_15

However, these functions are not necessarily submodular; a set function $f$ is submodular if $\forall X \subseteq Y, v \notin Y : f(X \cup \{v\}) - f(X) \geq f(Y \cup \{v\}) - f(Y)$, implying the diminishing returns property [15]. We will introduce two applications of this problem in the next section. For the general constraint presented in Eq. (1), the Generalized Greedy Algorithm (GGA) iteratively selects an item with the largest ratio of marginal gain on $f$ and cost $c$, achieving the best known $(\alpha_f/2)(1-e^{-\alpha_f})$-approximation ratio [21,31], where $\alpha_f$ measures the degree to which $f$ is nearly submodular.

In real-world scenarios, the resource budget $B$ in Eq. (1) for subset selection problems may vary over time. For example, in influence maximization, the market investment budget changes based on company outcomes and strategies, leading to a dynamic cost constraint. After each change of the budget $B$, it is feasible to treat the problem as a static problem with the new budget, and run static algorithms (e.g., GGA) from scratch, which, however, may lead to a long running time. When problem constraints change frequently, static algorithms might not be able to provide new solutions in time. For subset selection with dynamic cost constraints, Roostapour *et al.* [28] proposed an Adaptive Generalized Greedy Algorithm named AGGA, adjusting the current solution to fit the new budget. If the budget is reduced, it removes items with smallest ratio of the marginal gain on $f$ and cost $c$. Conversely, if the budget increases, it adds items like the static GGA. However, AGGA cannot maintain an $(\alpha_f/2)(1-e^{-\alpha_f})$-approximation for the new budget and may even perform arbitrarily poorly [28].

Unlike memoryless deterministic greedy algorithms, Evolutionary Algorithms (EAs) are capable of leveraging existing computational results while adapting to changes in the budget $B$. EAs can continually search for new solutions by using the genetic information from the parent individuals in the population. Qian *et al.* [21] proposed a Pareto Optimization approach for maximizing a Monotone function with a monotone Cost constraint, called POMC [9,26]. It reformulates the original subset selection problem as a bi-objective optimization problem, which maximizes the objective $f$ and minimizes the cost $c$ simultaneously, and then employs a multi-objective EA to solve it. For the static setting of a fixed budget $B$, POMC achieves the best known $(\alpha_f/2)(1-e^{-\alpha_f})$-approximation ratio using at most $O(nBP_{max}/\delta_{\widehat{c}})$ expected running time[1], where $n$ is the size of the ground set $V$, $P_{max}$ is the largest size of population during the run of POMC, $\widehat{c}$ is an approximation of the cost function $c$ in case the exact computation of $c$ is impractical in real-world scenarios, and $\delta_{\widehat{c}} = \min\{\widehat{c}(X \cup v) - \widehat{c}(X) | X \subseteq V, v \notin X\}$. When the budget $B$ decreases, POMC has already achieved the $(\alpha_f/2)(1-e^{-\alpha_f})$-approximation ratio; when the budget $B$ increases to $B'$, POMC can regain the $(\alpha_f/2)(1-e^{-\alpha_f})$-approximation ratio using at most $O(n\Delta_B P_{max}/\delta_{\widehat{c}})$ expected running time [28], where $\Delta_B = B' - B$. However, the running time of POMC may not be polynomial because the population size $P_{max}$ can grow exponentially [28]. Bian et al. [2] proposed a single-objective EA for maximizing a Monotone function with a monotone Cost constraint, called EAMC, which maximizes a surrogate objective $g(X) = f(X)/(1-e^{-\alpha_f \widehat{c}(X)/B})$, and maintains at most two solutions for each possible size in the population. For the static setting, EAMC ensures the best known approximation ratio using at most $O(n^3)$ expected running time, which is

[1] The expected running time refers to the expected number of evaluations of the objective function, as evaluations are usually the most expensive part of the process.

**Table 1.** Summary of algorithms with approximation guarantees and running time for the subset selection problem with static and dynamic cost constraints. The results obtained in this paper are highlighted in boldface. A "-" denotes the algorithm is not suitable for the corresponding case, while a "✗" means the algorithm is applicable but its performance is unknown.

| Algorithm | Static | | Dynamic | |
|---|---|---|---|---|
| | Guarantee | Running time | Guarantee | Running time |
| GGA [31] | $(\alpha_f/2)(1-e^{-\alpha_f})$ | $O(n^2)$ | - | - |
| AGGA [28] | - | - | ✗ | $O(n^2)$ |
| POMC [21,28] | $(\alpha_f/2)(1-e^{-\alpha_f})$ | $O(nBP_{max}/\delta_{\hat{c}})$ | $(\alpha_f/2)(1-e^{-\alpha_f})$ | $O(n\Delta_B P_{max}/\delta_{\hat{c}})$ |
| EAMC [2] | $(\alpha_f/2)(1-e^{-\alpha_f})$ | $O(n^3)$ | ✗ | ✗ |
| FPOMC [3] | $(\alpha_f/2)(1-e^{-\alpha_f})$ | $O(n^2K_B)$ | $(\alpha_f/2)(1-e^{-\alpha_f})$ | $O(nK_{B'}(K_{B'}-K_B))$ |
| **BPODC** | $\boldsymbol{(\alpha_f/2)(1-e^{-\alpha_f})}$ | $\boldsymbol{O(nB/(p_{min}\delta_{\hat{c}}))}$ | $\boldsymbol{(\alpha_f/2)(1-e^{-\alpha_f})}$ | $\boldsymbol{O(n\Delta_B/(p_{min}\delta_{\hat{c}}))}$ |

polynomial. However, the surrogate objective $g$ of EAMC changes with the budget $B$, rendering previous solutions potentially ineffective for the new $B$. Bian et al. [3] then proposed the Fast Pareto Optimization algorithm for maximizing a Monotone function with a monotone Cost constraint, called FPOMC, which is modified from POMC by introducing a greedy selection strategy and estimating the goodness of a solution to be selected. For the static setting, FPOMC obtains the best known $(\alpha_f/2)(1-e^{-\alpha_f})$-approximation ratio using at most $O(n^2K_B)$ expected running time, which is also polynomial, where $K_B$ denotes the largest size of a subset satisfying the constraint $c(X) \leq B$. If the budget decreases, FPOMC maintains the best known approximation ratio; if the budget increases to $B'$, FPOMC can regain the same ratio within an expected running time of $O(nK_{B'}(K_{B'}-K_B))$, where $K_{B'}$ is the maximum subset size satisfying the new budget constraint $c(X) \leq B'$. We summarize the related works in Table 1.

Among the algorithms suitable for dynamic environments (AGGA, POMC, EAMC and FPOMC), POMC empirically performs the best [28]. However, it was originally designed for static problems and lacks considerations for dynamic environments. A natural question is whether it is possible to design a more advanced algorithm for dynamic environments that can quickly adapt its solutions when the budget $B$ changes. If so, how fast can this algorithm adapt, and can it surpass the speed of the static GGA?

In this paper, we introduce BPODC based on Biased Pareto Optimization for maximizing a monotone function under Dynamic Cost constraints. BPODC is a modification of the POMC algorithm, enhanced with a biased selection mechanism and a warm-up strategy. POMC selects a solution uniformly at random from the population, often resulting in the choice of an "unnecessary" solution and leading to inefficiency. In contrast, BPODC employs a biased selection strategy that favors solutions with cost values closer to the current budget $B$, which are selected with a higher probability. During the initial phase of one run, BPODC employs uniform selection temporarily as a warm-up to obtain a diverse population, which will lead the biased selection strategy to be more efficient. As in [5,7], we focus on the ability of BPODC to leverage existing computational results while adapting to budget changes. We prove that BPODC maintains the best-known $(\alpha_f/2)(1-e^{-\alpha_f})$-approximation ratio when the budget $B$ decreases.

When the budget increases to $B'$, BPODC can regain the same ratio within an expected running time of $O(n\Delta_B/(p_{min}\delta_{\widehat{c}}))$, where $\Delta_B = B' - B$, and $p_{min}$ is the minimum probability of selecting a solution during the run of BPODC. Through empirical evaluation on the applications of influence maximization and maximum coverage, we show that BPODC can find better solutions than previous algorithms while requiring less running time than the static greedy algorithm GGA, offering an alternative for solving subset selection problems with dynamic cost constraints.

## 2 Subset Selection with Cost Constraints

Let $\mathbb{R}$ and $\mathbb{R}^+$ denote the set of reals and non-negative reals, respectively. The set $V = \{v_1, v_2, \ldots, v_n\}$ denotes a ground set. A set function $f : 2^V \to \mathbb{R}$ is monotone if $\forall X \subseteq Y : f(X) \leq f(Y)$. A set function $f$ is submodular if $\forall X \subseteq Y, v \notin Y : f(X \cup \{v\}) - f(X) \geq f(Y \cup \{v\}) - f(Y)$, which intuitively represents the diminishing returns property [15], i.e., adding an item to a set $X$ gives a larger benefit than adding the same item to a superset $Y$ of $X$. The submodularity ratio in Definition 1 characterizes how close a set function is to submodularity. When $f$ satisfies the monotone property, we have $0 \leq \alpha_f \leq 1$, and $f$ is submodular iff $\alpha_f = 1$.

**Definition 1 (Submodularity Ratio [24,31]).** *The submodularity ratio of a non-negative set function $f$ is defined as* $\alpha_f = \min_{X \subseteq Y, v \notin Y} \frac{f(X \cup \{v\}) - f(X)}{f(Y \cup \{v\}) - f(Y)}$.

As presented in Definition 2, the subset selection problem with static cost constraints is to maximize a monotone objective function $f$ such that a monotone cost function $c$ is no larger than a budget $B$. We assume w.l.o.g. that monotone functions are normalized, i.e., $f(\emptyset) = 0$ and $c(\emptyset) = 0$. Since the exact computation of $c(X)$ may be unsolvable in polynomial time in some real-world applications [21,31], we assume that only an $\psi(n)$-approximation $\widehat{c}$ can be obtained, where $\forall X \subseteq V : c(X) \leq \widehat{c}(X) \leq \psi(n) \cdot c(X)$. If $\psi(n) = 1$, $\widehat{c}(X) = c(X)$.

**Definition 2 (Subset Selection with Static Cost Constraints).** *Given a monotone objective function $f : 2^V \to \mathbb{R}^+$, a monotone cost function $c : 2^V \to \mathbb{R}^+$ and a budget $B$, to find*

$$\arg\max_{X \subseteq V} f(X) \quad s.t. \quad c(X) \leq B. \tag{2}$$

The static problem in Definition 2 assumes that the budget $B$ is fixed. However, in real-world scenarios, the resources often change over time, and thus the budget $B$ may change dynamically. For example, in influence maximization, the market investment budget changes based on company outcomes and strategies, leading to dynamic cost constraints, that is, the budget $B$ in Eq. (2) may change over time. In this paper, we focus on the subset selection problem with dynamic cost constraints, given in Definition 3. Whenever the budget $B$ changes, the problem can be treated as a new static problem using the updated budget, and static algorithms can be applied from the beginning. However, this may lead to a long running time.

**Definition 3 (Subset Selection with Dynamic Cost Constraints).** *Given a monotone objective function $f : 2^V \to \mathbb{R}^+$, a monotone cost function $c : 2^V \to \mathbb{R}^+$, and a sequence of changes on the budget $B$, to find a subset optimizing Eq. (2) for each new $B$.*

**Influence Maximization.** Influence maximization in Definition 4 is to identify a set of influential users in social networks [12]. A social network can be represented by a directed graph $G = (V, E)$, where each node represents a user and each edge $(u, v) \in E$ has a probability $p_{u,v}$ representing the influence strength from user $u$ to $v$. Given a budget $B$, influence maximization is to find a subset $X$ of $V$ such that the expected number of nodes activated by propagating from $X$ is maximized, while not violating the cost constraint $c(X) \leq B$. Here we use the fundamental propagation model called Independence Cascade (IC). Starting from a seed set $X$, it uses a set $A_t$ to record the nodes activated at time $t$, and at time $t+1$, each inactive neighbor $v$ of $u \in A_t$ becomes active with probability $p_{u,v}$; this process is repeated until no nodes get activated at some time. The set of nodes activated by propagating from $X$ is denoted as $IC(X)$, which is a random variable. The objective $\mathbb{E}[IC(X)]$ denotes the expected number of nodes activated by propagating from $X$, which is monotone and submodular.

**Definition 4 (Influence Maximization).** *Given a directed graph $G = (V, E)$, edge probabilities $p_{u,v}$ where $(u, v) \in E$, a monotone cost function $c : 2^V \to \mathbb{R}^+$ and a budget $B$, to find*

$$\arg\max_{X \subseteq V} \mathbb{E}[|IC(X)|] \quad \textit{s.t.} \quad c(X) \leq B.$$

**Maximum Coverage.** Given a family of sets that cover a universe of elements, maximum coverage as presented in Definition 5 is to select some sets whose union is maximal under a cost budget. It is easy to verify that $f$ is monotone and submodular.

**Definition 5 (Maximum Coverage).** *Given a set $U$ of elements, a collection $V = \{S_1, S_2, \ldots, S_n\}$ of subsets of $U$, a monotone cost function $c : 2^V \to \mathbb{R}^+$ and a budget $B$, to find*

$$\arg\max_{X \subseteq V} f(X) = |\bigcup\nolimits_{S_i \in X} S_i| \quad \textit{s.t.} \quad c(X) \leq B.$$

### 2.1 Previous Algorithms

We now introduce five algorithms capable of solving the subset selection problem with dynamic cost constraints in Definition 3.

**GGA.** The Generalized Greedy Algorithm (GGA) proposed in [31] selects one item maximizing the ratio of the marginal gain on $f$ and $\widehat{c}$ in each iteration. After examining all items, the found subset is compared with the best single item (i.e., $v^* \in \arg\max_{v \in V : \widehat{c}(\{v\}) \leq B} f(\{v\})$), and the better one is returned. Let

$$f(\widetilde{X}) = \max\left\{ f(X) \mid c(X) \leq B \cdot \frac{\alpha_{\widehat{c}}(1 + \alpha_c^2(K_B - 1)(1 - \kappa_c))}{\psi(n) K_B} \right\}, \tag{3}$$

where $\alpha_c$ and $\alpha_{\widehat{c}}$ are the submodularity ratios of the cost function $c$ and its approximation $\widehat{c}$, respectively, $\kappa_c = 1 - \min_{v \in V : c(\{v\}) > 0} \frac{c(V) - c(V \setminus \{v\})}{c(\{v\})}$ is the total curvature of

$c$, and $K_B = \max\{|X| \mid c(X) \leq B\}$, i.e., the largest size of a subset satisfying the constraint. As $1 - \kappa_c \leq 1/\alpha_c$, $0 \leq \alpha_{\widehat{c}}, \alpha_c \leq 1$ and $\psi(n) \geq 1$, it holds that

$$\frac{\alpha_{\widehat{c}}(1 + \alpha_c^2(K_B - 1)(1 - \kappa_c))}{\psi(n)K_B} \leq 1.$$

Thus, $\widetilde{X}$ is actually an optimal solution of Eq. (2) with a slightly smaller budget constraint. GGA can solve only the static problem in Definition 2. Qian *et al.* [21] proved that GGA can obtain a subset $X$ satisfying $f(X) \geq (\alpha_f/2) \cdot (1 - e^{-\alpha_f}) \cdot f(\widetilde{X})$. The dynamic problem in Definition 3 can be viewed as a series of static problems, each with a different budget $B$. After the budget changes, GGA can be applied to solve the subsequent static problem with the updated budget from scratch.

**AGGA.** To face the dynamic changes of $B$, Roostapour *et al.* [28] introduced a natural Adaptive version of the static GGA, named AGGA. When $B$ increases, AGGA continues to iteratively add an item with the largest ratio of the marginal gain on $f$ and cost $c$ to the current solution, just like GGA; when $B$ decreases, it iteratively deletes one item with the smallest ratio of the marginal gain on $f$ and cost $c$, until the solution no longer violates the new budget. However, AGGA cannot maintain an approximation for the new budget and may even perform arbitrarily badly [28].

**POMC.** Qian *et al.* [21] proposed POMC, a Pareto Optimization method for maximizing a Monotone function with a monotone Cost constraint, which reformulates the original problem Eq. (2) as a bi-objective maximization problem that maximizes the objective function $f$ and minimizes the approximate cost function $\widehat{c}$ simultaneously [9,26]. To solve the bi-objective problem, POMC employs a simple multi-objective EA, i.e., GSEMO [14,25], which uses uniform selection and bit-wise mutation to generate an offspring solution and keeps the non-dominated solutions generated-so-far in the population. After terminated, it returns the best feasible solution with the largest $f$ value in the population. For the static setting of a fixed budget $B$, POMC can achieve the best known $(\alpha_f/2)(1-e^{-\alpha_f})$-approximation ratio, and also regain this approximation ratio at most $O(n\Delta_B P_{max}/\delta_{\widehat{c}})$ expected running time when the budget changes [21,28].

**EAMC.** Bian *et al.* [2] proposed a single-objective EA for maximizing a Monotone function with a monotone Cost constraint, called EAMC. It tries to maximize a surrogate objective $g$ which considers both the original objective $f$ and the cost $\widehat{c}$. For $|X| \geq 1$, $g(X) = f(X)/(1 - e^{-\alpha_f \widehat{c}(X)/B})$, while for $|X| = 0$, $g(X) = f(X)$. The submodularity ratio $\alpha_f$, used to calculate the surrogate objective $g$, may require exponential time to compute accurately, so a lower bound on $\alpha_f$ is often used instead. EAMC also applies uniform selection and bit-wise mutation to generate an offspring solution as POMC. For each subset size $i$, EAMC contains the solutions with the largest $g$ or $f$ values, bounding the maximum population size. After terminated, EAMC returns the feasible solution with the largest $f$ value in the population. EAMC can achieve the best-known $(\alpha_f/2)(1-e^{-\alpha_f})$-approximation in a static setting. However, in a dynamic setting, the $g$ function, based on the old budget, cannot characterize the new problem well, potentially leading to poor performance with the new budget.

**FPOMC.** Bian et al. [3] then proposed the Fast Pareto Optimization algorithm for maximizing a Monotone function with a monotone Cost constraint, called FPOMC, which

is modified from POMC. The main difference between FPOMC and POMC is that FPOMC applies a greedy selection strategy, while POMC applies uniform selection. The greedy selection strategy uses a function $h_Z(X)$ to estimate the goodness of a solution $X$ w.r.t. a reference point $Z$, which is defined as

$$h_Z(X) = \begin{cases} (f(X) - f(Z))/(\widehat{c}(X) - \widehat{c}(Z)) & \widehat{c}(X) > \widehat{c}(Z), \\ (f(X) - f(Z)) \cdot C + \widehat{c}(Z) - \widehat{c}(X) & \widehat{c}(X) \leq \widehat{c}(Z), \end{cases}$$

where $C$ is a large enough number. Intuitively, $h_Z(X)$ measures the goodness of $X$ by the marginal gain on $f$ and $c$ w.r.t. a reference point $Z$, and the solution with the largest $h$ value is selected with a high probability. For more detailed design of FPOMC, please refer to [3]. For the static setting, FPOMC can obtain the best known $(\alpha_f/2)(1 - e^{-\alpha_f})$-approximation ratio, and also regain the same approximation ratio at most $O(nK_{B'}(K_{B'} - K_B))$ expected running time for the new budget.

AGGA and the static GGA are greedy algorithms. POMC, EAMC and FPOMC are anytime algorithms that can find better solutions using more time. Among them, POMC performs the best empirically in dynamic environments [28]; however, it was initially designed for static settings and lacks considerations for dynamic environments. This work focuses on designing an advanced algorithm tailored for dynamic environments, aiming to quickly adapt its solutions to budget changes and potentially exceed the speed of the static GGA.

## 3 The BPODC Algorithm

In this section, we propose an algorithm based on Biased Pareto Optimization [4,7,9, 16–20,22,23,25–28,32] for maximizing a monotone function with Dynamic Cost constraints, called BPODC, which can quickly adapt its solutions when budget changes. It represents a subset $X \subseteq V$ by a Boolean vector $\boldsymbol{x} \in \{0,1\}^n$, where the $i$-th bit $x_i = 1$ iff $v_i \in X$. We will not distinguish $\boldsymbol{x} \in \{0,1\}^n$ and its corresponding subset $\{v_i \in V | x_i = 1\}$ for notational convenience. BPODC reformulates the original problem Eq. (2) as a bi-objective maximization problem

$$\arg\max_{\boldsymbol{x} \in \{0,1\}^n} \big(f_1(\boldsymbol{x}),\ f_2(\boldsymbol{x})\big), \tag{4}$$

$$\text{where } f_1(\boldsymbol{x}) = \begin{cases} -\infty, & \widehat{c}(\boldsymbol{x}) > B + 1 \\ f(\boldsymbol{x}), & \text{otherwise} \end{cases}, \quad f_2(\boldsymbol{x}) = -\widehat{c}(\boldsymbol{x}).$$

That is, BPODC maximizes the objective function $f$ and the negative of the approximate cost function $\widehat{c}$ simultaneously. Solutions with cost values over $B + 1$ (i.e., $\widehat{c}(\boldsymbol{x}) > B + 1$) are excluded by setting their $f_1$ values to $-\infty$. We use the value $B + 1$ to give the algorithm a slight look ahead for larger constraint bounds without making the population size too large. The introduction of the second objective $f_2$ can naturally bring a diverse population, which may lead to better optimization performance.

Note that the objective vector $(f_1(\boldsymbol{x}), f_2(\boldsymbol{x}))$ is calculated only when the solution $\boldsymbol{x}$ is generated. This means that any subsequent changes to $B$ do not trigger an update of the objective vector. Thus, solutions exceeding cost $B' + 1$ for a new budget $B'$

**Algorithm 1.** BPODC Algorithm

**Input**: a monotone objective function $f$, a monotone approximate cost function $\widehat{c}$, a ground set with $n$ items, and a sequence of changes on the budget B
**Output**: a solution $\boldsymbol{x} \in \{0,1\}^n$ with $\widehat{c}(\boldsymbol{x}) \leq B$
**Process**:

1: Let $\boldsymbol{x} = 0^n$, $P = \{\boldsymbol{x}\}$;
2: **repeat**
3: **if** warm-up stage **then**
4: Select $\boldsymbol{x}$ from $P$ uniformly at random
5: **else**
6: $\boldsymbol{x} =$ Biased Selection$(P, B)$
7: **end if**
8: Generate $\boldsymbol{x}'$ by flipping each bit of $\boldsymbol{x}$ with probability $1/n$;
9: **if** $\nexists \boldsymbol{z} \in P$ such that $\boldsymbol{z} \succ \boldsymbol{x}'$ **then**
10: $P = (P \setminus \{\boldsymbol{z} \in P \mid \boldsymbol{x}' \succeq \boldsymbol{z}\}) \cup \{\boldsymbol{x}'\}$
11: **end if**
12: **until** some criterion is met
13: **return** $\arg\max_{\boldsymbol{x} \in P, \widehat{c}(\boldsymbol{x}) \leq B} f(\boldsymbol{x})$

are still retained in the population. However, for any new solutions exceeding $B' + 1$, the $f_1$ value is set to $-\infty$. As the two objectives may be conflicting, the domination relationship in Definition 6 is often used for comparing two solutions. A solution is Pareto optimal if no other solution dominates it. The collection of objective vectors of all Pareto optimal solutions is called the Pareto front.

**Definition 6 (Domination).** *For two solutions $\boldsymbol{x}$ and $\boldsymbol{x}'$,*

- $\boldsymbol{x}$ weakly dominates $\boldsymbol{x}'$, *denoted as* $\boldsymbol{x} \succeq \boldsymbol{x}'$, *if* $f_1(\boldsymbol{x}) \geq f_1(\boldsymbol{x}') \wedge f_2(\boldsymbol{x}) \geq f_2(\boldsymbol{x}')$;
- $\boldsymbol{x}$ dominates $\boldsymbol{x}'$, *denoted as* $\boldsymbol{x} \succ \boldsymbol{x}'$, *if* $\boldsymbol{x} \succeq \boldsymbol{x}'$ *and either* $f_1(\boldsymbol{x}) > f_1(\boldsymbol{x}')$ *or* $f_2(\boldsymbol{x}) > f_2(\boldsymbol{x}')$;
- *they are* incomparable *if neither* $\boldsymbol{x} \succeq \boldsymbol{x}'$ *nor* $\boldsymbol{x}' \succeq \boldsymbol{x}$.

After constructing the bi-objective problem in Eq. (4), BPODC solves it by a process of multi-objective EAs, as described in Algorithm 1. EAs, inspired by Darwin's theory of evolution are general-purpose randomized heuristic optimization algorithms [1,33], mimicking variational reproduction and natural selection, which have become the most popular tool for multi-objective optimization [6,11,30]. It starts from the empty set $0^n$ (line 1), and repeatedly improves the quality of solutions in population $P$ (lines 2–12). At the start of the process, BPODC applies uniform selection to select a parent solution $\boldsymbol{x}$ for a period of time, which is referred to the warm-up stage (lines 3–4). Our aim is to uniformly explore the $(0, B+1]$ area initially to obtain a population with good diversity. After the warm-up stage, BPODC selects a parent solution $\boldsymbol{x}$ in $P$ according to the BIASED SELECTION subroutine in Algorithm 2 (line 6). Then, a solution $\boldsymbol{x}'$ is generated by applying bit-wise mutation on $\boldsymbol{x}$ (line 8), which is used to update the population $P$ (line 9–10). If $\boldsymbol{x}'$ is not dominated by any solution in $P$ (line 9), it will be added into $P$, and meanwhile, those solutions weakly dominated by $\boldsymbol{x}'$ will be deleted (line 10). This updating procedure makes the population $P$ always

**Algorithm 2.** Biased Selection($P$, $B$): Subroutine of BPODC

**Input**: the population $P$ and the budget $B$
**Output**: a solution in $P$ for mutation
**Process**:
1: $\epsilon \leftarrow 1 \times 10^{-10}$;
2: **for** $i = 1$ to $|P|$ **do**
3: $\quad \boldsymbol{x} \leftarrow$ the $i$-th solution in population $P$;
4: $\quad probs[i] \leftarrow 1/(|c(\boldsymbol{x}) - B| + \epsilon)$
5: **end for**
6: $probs \leftarrow Normalization(probs)$;
7: Select $\boldsymbol{x}$ from $P$ according to the probabilities *probs*
8: **return** $\boldsymbol{x}$

contain incomparable solutions. After running a number of iterations, the best feasible solution with the largest $f$ value in the population $P$ is output (line 13). Note that the aim of BPODC is to find a good solution of the original problem in Definition 3, rather than the Pareto front of the reformulated bi-objective problem in Eq. (4). That is, the bi-objective reformulation is an intermediate process.

The BIASED SELECTION subroutine in Algorithm 2 first computes a selection probability of each solution $\boldsymbol{x} \in P$ iteratively, which is inversely proportional to the difference between the cost value $c(\boldsymbol{x})$ and the given budget $B$ (lines 2–5). The $\epsilon$ is added to avoid division by zero (line 4). The subroutine then normalizes the probabilities, and selects a solution $\boldsymbol{x}$ from $P$ based on these normalized probabilities (lines 6–7). Algorithm 2 exhibits a bias whereby solutions with cost values close to the budget $B$ have a higher probability of selection. This enables BPODC to quickly regain high-quality solutions upon budget changes, thereby meeting the demands of dynamic environments.

Note that BPODC uses the warm-up strategy only for the initial budget, starting from the zero solution; if the budget changes, it continues from the current population using biased selection.

## 4 Theoretical Analysis

In this section, we prove the general approximation bound of BPODC in Theorem 1, implying that BPODC can achieve the best known $(\alpha_f/2)(1 - e^{-\alpha_f})$-approximation guarantee for the static problem in Definition 2. When facing a dynamic budget change, BPODC still can regain the $(\alpha_f/2)(1 - e^{-\alpha_f})$-approximation ratio (Theorem 2).

Let $p_{min}$ denote the minimum probability of selecting a solution from the population during the run of BPODC and $\delta_{\widehat{c}} = \min\{\widehat{c}(X \cup v) - \widehat{c}(X) | X \subseteq V, v \notin X\}$. We assume that $\delta_{\widehat{c}} > 0$. The proof of Theorem 1 is based on the approach used in Theorem 2 of [21], mainly analyzing the expected number of iterations of BPODC required to obtain an $(\alpha_f/2)(1 - e^{-\alpha_f})$-approximation solution.

**Theorem 1.** *For the static problem in Definition 2, BPODC using at most* $O(nB/(p_{min} \cdot \delta_{\widehat{c}}))$ *expected number of iterations finds a subset* $X \subseteq V$ *with*

$$f(X) \geq (\alpha_f/2) \cdot (1 - e^{-\alpha_f}) \cdot f(\widetilde{X}),$$

*where $f(\widetilde{X})$ is defined in Eq. (3).*

*Proof.* We follow the proof of POMC in Theorem 2 of [21], which analyzes the increase of a quantity $J_{max}$ during the process of POMC, where $J_{max} = \max\{j \in [0, B) | \exists \boldsymbol{x} \in P, \widehat{c}(\boldsymbol{x}) \leq j \wedge f(\boldsymbol{x}) \geq (1-(1-\alpha_f \frac{j}{Bk})^k) \cdot f(\widetilde{X}) \text{ for some } k\}$. Let $\boldsymbol{x} \in P$ be a solution corresponding to the current value of $J_{max}$. It is pointed out that $J_{max}$ can increase at least $\delta_{\widehat{c}}$ by adding a specific item to $\boldsymbol{x}$. Furthermore, they proved that POMC can find a solution with the $f$ value at least

$$\max\{f(\boldsymbol{p}), f(\boldsymbol{q})\} \geq (\alpha_f/2) \cdot (1 - e^{-\alpha_f}) \cdot f(\widetilde{X}),$$

where $\boldsymbol{p}$ is the solution that results from increasing $J_{max}$ at least $B/\delta_{\widehat{c}}$ times starting from $J_{max} = 0$, and $\boldsymbol{q}$ is defined as $\boldsymbol{q} = \arg\max_{v \in V: \widehat{c}(\boldsymbol{q}) \leq B} f(v)$.

Our proof finishes by analyzing the expected number of iterations until BPODC contains two solutions $\boldsymbol{p}$ and $\boldsymbol{q}$. We first analyze the expected number of iterations of BPODC to generate the solution $\boldsymbol{p}$. The initial value of $J_{max}$ is 0 as BPODC starts from $\{0\}^n$. Assume the current value of $J_{max}$ is $i$, and $\boldsymbol{x} \in P$ is a corresponding solution. According to the population update mechanism described in lines 9-10 of Algorithm 1, $J_{max}$ cannot decrease because $\boldsymbol{x}$ can be deleted from $P$ (line 10) only when the newly included solution $\boldsymbol{x}'$ weakly dominates $\boldsymbol{x}$, i.e., $f(\boldsymbol{x}') \geq f(\boldsymbol{x})$ and $\widehat{c}(\boldsymbol{x}') \leq \widehat{c}(\boldsymbol{x})$, which makes $J_{max} \geq i$. As mentioned above, to increase $J_{max}$ from $i$ to at least $i + \delta_{\widehat{c}}$, we can add a specific item to $\boldsymbol{x}$ to generate a new solution $\boldsymbol{x}'$. Upon generating $\boldsymbol{x}'$, it will be included into $P$; otherwise, $\boldsymbol{x}'$ must be dominated by one solution in $P$ (line 9 of Algorithm 1), and this implies that $J_{max}$ has already been larger than $i$, which contradicts with the assumption $J_{max} = i$. In each iteration, the probability of successfully increasing $J_{max}$ is at least $p_{min} \cdot \frac{1}{n} \cdot (1 - \frac{1}{n})^{n-1} \geq \frac{p_{min}}{en}$, where $p_{min}$ is a lower bound on the probability of selecting $\boldsymbol{x}$ in line 4 or line 6 of Algorithm 1 and $\frac{1}{n} \cdot (1 - \frac{1}{n})^{n-1}$ is the probability of flipping a specific bit of $\boldsymbol{x}$ while keeping other bits unchanged in line 8. Then, it needs at most $en/p_{min}$ expected number of iterations to increase $J_{max}$ by at least $\delta_{\widehat{c}}$. After at most $enB/(p_{min} \cdot \delta_{\widehat{c}})$ expected number of iterations, $\boldsymbol{p}$ will be generated and included into $P$; otherwise, $P$ has already contained a solution $\boldsymbol{z} \succeq \boldsymbol{p}$, i.e., $\widehat{c}(\boldsymbol{z}) \leq \widehat{c}(\boldsymbol{p}) \leq B$ and $f(\boldsymbol{z}) \geq f(\boldsymbol{p})$.

We now analyze the expected number of iterations to generate and contain the solution $\boldsymbol{q}$. Since $\{0\}^n$ has the largest $f_2$ value (i.e., the smallest $\widehat{c}$ value), no other solution can dominate it, ensuring that $\{0\}^n$ will always be included in $P$. Thus, $\boldsymbol{q}$ can be generated in one iteration by selecting $\{0\}^n$ in line 4 or 6 of Algorithm 1 and flipping only the corresponding 0-bit in line 8, whose probability is at least $p_{min}/en$. That is, $\boldsymbol{q}$ will be generated using at most $en/p_{min}$ expected number of iterations.

Taking the expected number of iterations for generating $\boldsymbol{p}$ and $\boldsymbol{q}$ together, BPODC using at most $O(nB/(p_{min} \cdot \delta_{\widehat{c}}))$ expected number of iterations finds a solution with the $f$ value at least $\max\{f(\boldsymbol{p}), f(\boldsymbol{q})\} \geq (\alpha_f/2) \cdot (1 - e^{-\alpha_f}) \cdot f(\widetilde{X})$. □

**Theorem 2.** *For the dynamic problem in Definition 3, assume that BPODC has achieved an $(\alpha_f/2)(1-e^{-\alpha_f})$-approximation ratio for the current budget $B$ after running at most $O(nBP_{max}/\delta_{\widehat{c}})$ expected number of iterations:*

*(1) when $B$ decreases to $B'$, BPODC has already achieved the $(\alpha_f/2)(1 - e^{-\alpha_f})$-approximation ratio for the new budget;*

*(2) when $B$ increases to $B'$, BPODC using at most $O(n(B'-B)/(p_{min}\cdot\delta_{\widehat{c}}))$ expected number of iterations, can regain the $(\alpha_f/2)(1-e^{-\alpha_f})$-approximation ratio.*

*Proof.* By analyzing the increasing process of $J_{max}$ from 0 to $B$ as shown in the proof of Theorem 1, BPODC using $O(nB/(p_{min}\cdot\delta_{\widehat{c}}))$ expected number of iterations has achieved an $(\alpha_f/2)\cdot(1-e^{-\alpha_f})$-approximation ratio for the problem with any budget $B'\leq B$. When the budget $B$ increases to $B'$, in order to achieve the desired approximation guarantee, it is sufficient to continuously increase $J_{\max}$ from $B$ to $B'$. The expected number of iterations required for the increase is $O(n(B'-B)/(p_{min}\cdot\delta_{\widehat{c}}))$. □

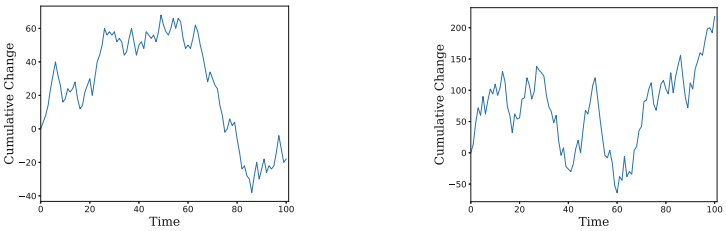
## 5 Empirical Study

In this section, we empirically examine the performance of BPODC on dynamic variants of subset selection problems, specifically focusing on influence maximization and maximum coverage tasks where the cost constraint varies over time. We compare BPODC with several competitive algorithms: the static greedy algorithm GGA, the dynamic greedy algorithm AGGA, and the EA-based methods, POMC, EAMC, and FPOMC.

POMC and FPOMC use the same bi-objective as BPODC, as shown in Eq. (4), which is calculated only when the solution $\boldsymbol{x}$ is generated, and any subsequent changes to the budget $B$ do not trigger an update, that is, algorithms do not delete solutions from the population that become infeasible under a new budget. For EAMC, a budget change influences the value of surrogate function $g(X)=f(X)/(1-e^{-\alpha_f\widehat{c}(X)/B})$. Thus, the value of function $g$ for solutions in the population is updated after each dynamic change. Note that the parameter $\alpha_f$ equals 1 because the objective functions of influence maximization and maximum coverage are submodular. The static greedy algorithm GGA requires $n(n+1)/2$ objective evaluations, denoted as $T_G$. For each budget change, the number of objective evaluations for EAs (BPODC, POMC, EAMC, and FPOMC) is set to $t=\{0.25T_G, 0.5T_G\}$, which is less than that required by GGA. Note that for the $t$ evaluations of the initial budget, BPODC uses the warm-up strategy to create a diverse population; for subsequent $t$ evaluations of a changed budget, it employs biased selection on the current population without a warm-up. For randomized algorithms (BPODC, POMC, EAMC and FPOMC), we independently run 30 times and report the average results. The source code is available at https://github.com/lamda-bbo/BPODC.

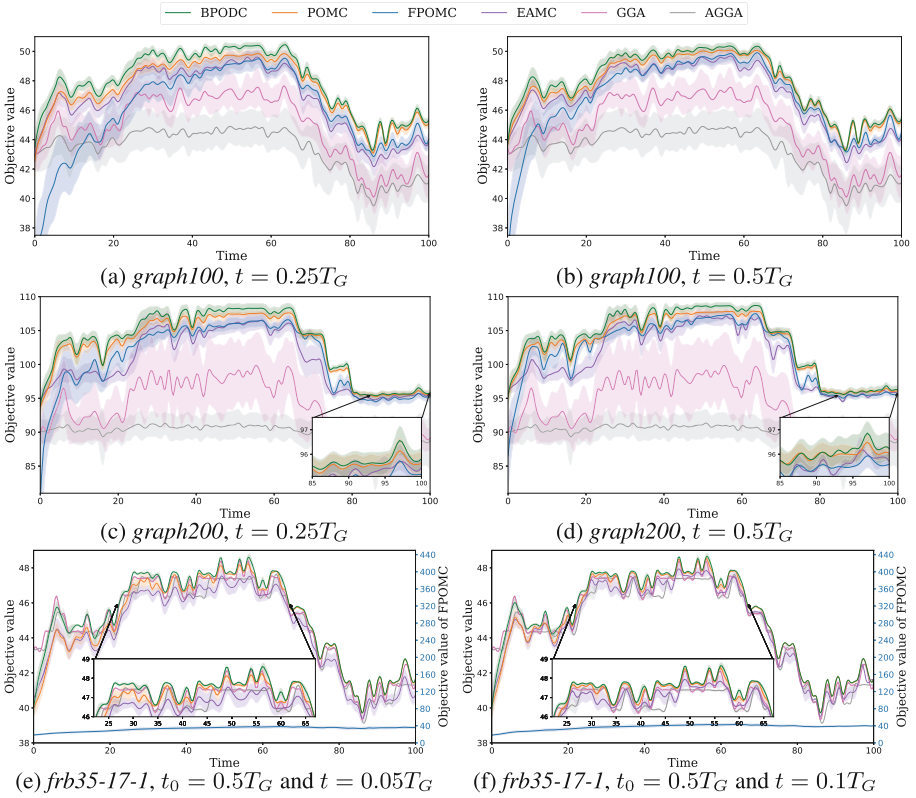
The experiments are mainly to answer two questions: Can BPODC perform the best among all algorithms under dynamic environments? Can BPODC adapt its solution within a shorter running time than the static GGA?

**Influence Maximization.** We use the same two datasets as social networks in influence maximization as in [2], called *graph100* (100 vertices, 3,465 edges) and *graph200* (200 vertices, 9,950 edges), respectively. The probability of each edge is set to 0.05. Besides, we use a larger real-world dataset, *frb35-17-1* (595 vertices, 27,856 edges), with each edge's probability 0.01. We consider the linear cost constraint, where $c(X)=\sum_{v\in X}c_v$. The cost of each item is calculated based on its out-degree $d(v)$, i.e., $c_v=1+(1+|\xi|)\cdot d(v)$, where $\xi$ is a random number drawn from the normal distribution $\mathcal{N}(0,0.5^2)$. The original budget is set to 300, and stays within the interval $[100,500]$.




(a) Influence Maximization (b) Maximum Coverage

**Fig. 1.** The cumulative budget changes relative to the initial budget.



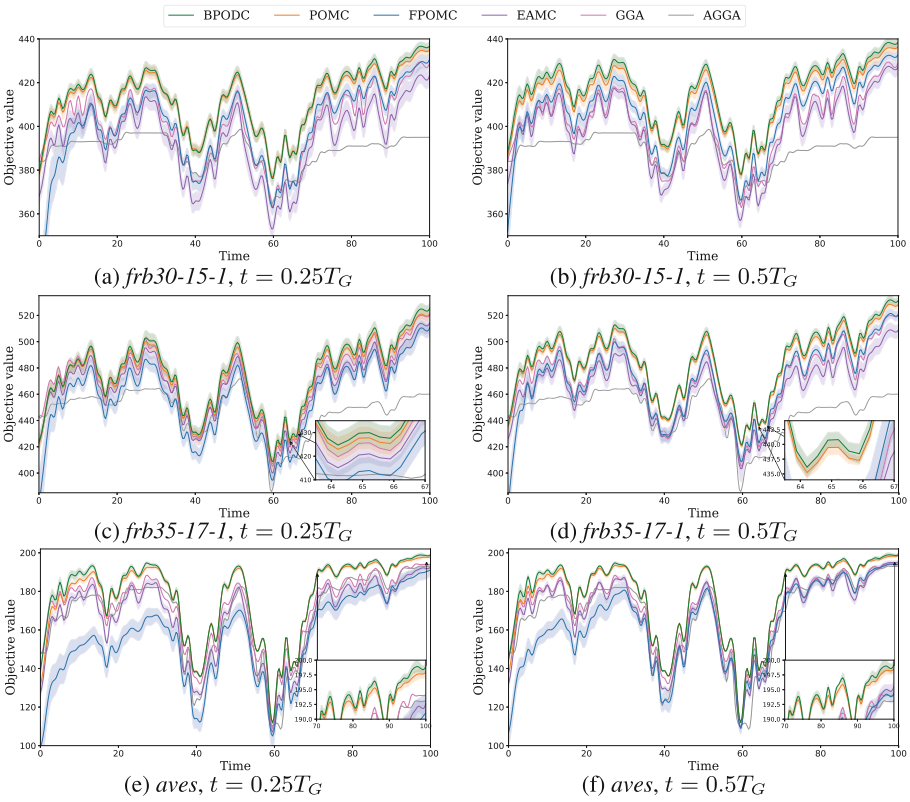

(a) *graph100*, $t = 0.25T_G$ (b) *graph100*, $t = 0.5T_G$

(c) *graph200*, $t = 0.25T_G$ (d) *graph200*, $t = 0.5T_G$

(e) *frb35-17-1*, $t_0 = 0.5T_G$ and $t = 0.05T_G$ (f) *frb35-17-1*, $t_0 = 0.5T_G$ and $t = 0.1T_G$

**Fig. 2.** The average value $\pm$ std for influence maximization under each budget change.

We consider a sequence of 100 budget changes obtained by randomly changing the current budget $B$ by a value of $[-10, 10]$. The cumulative changes relative to the initial budget are depicted in Fig. 1(a). The current budget at each time is calculated by adding the $y$-value at that point to the initial budget. To calculate $\mathbb{E}[|IC(X)|]$ in our experiments, we simulate the random propagation process starting from the solution $X$ for 500 times independently, and use the average as an estimation. Due to time constraints,




(a) *frb30-15-1*, $t = 0.25T_G$ (b) *frb30-15-1*, $t = 0.5T_G$

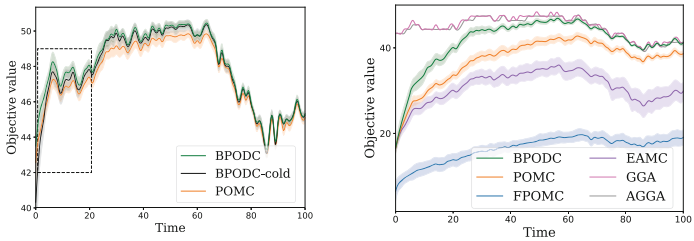
(c) *frb35-17-1*, $t = 0.25T_G$ (d) *frb35-17-1*, $t = 0.5T_G$

(e) *aves*, $t = 0.25T_G$ (f) *aves*, $t = 0.5T_G$

**Fig. 3.** The average value $\pm$ std for maximum coverage under each budget change.

we limit the number of objective evaluations of EAs (BPODC, POMC, EAMC, and FPOMC) on the *frb35-17-1* dataset, i.e., $t_0 = 0.5T_G$ evaluations for the initial budget and $t = \{0.05T_G, 0.1T_G\}$ evaluations for upcoming budgets. This setting is to ensure that the study could be completed within the available timeframe while still providing meaningful insights into the algorithm's performance trends. Since the behavior of the greedy algorithm GGA or AGGA is randomized under noise, we also repeat its run 30 times independently and report the average results. The curves of average results over each time of change are plotted in Fig. 2, where the shaded areas indicate the standard deviation around the mean.

Figure 2(a)-(d) clearly shows that BPODC, POMC and EAMC consistently outperform GGA and AGGA and are more stable, exhibiting a smaller std. FPOMC initially obtains a lower value during the first few changes, and then exceeds GGA and AGGA. These observations highlight the superior ability of EAs to leverage existing computational results while adapting to changes in budget constraints. Among all the algorithms, BPODC performs the best, regardless of the settings at $t = 0.25T_G$ or $t = 0.5T_G$. In Fig. 2(e)-(f), we set the running time allowed for EAs after each dynamic change to be much shorter, specifically $0.05T_G$ and $0.1T_G$, respectively. BPODC is the first to




(a) *graph100*, $t = 0.25T_G$ (b) *frb35-17-1*, $t = 1000$

**Fig. 4.** Additional results for influence maximization under dynamic environment.

surpass GGA and AGGA and continues to consistently perform the best. Note that the curve of FPOMC is plotted on the secondary $y$-axis due to its poor performance.

**Maximum Coverage.** We use three real-world graph datasets for maximum coverage: *frb30-15-1* (450 vertices, 17,827 edges) and *frb35-17-1* (595 vertices, 27,856 edges), both used in [2,28], and *aves* (202 vertices, 4,658 edges) [29]. For each vertex, we generate a set which contains the vertex itself and its adjacent vertices. We still use the linear cost constraint $c(X) = \sum_{v \in X} c_v$. The cost of each vertex is $c_v = 1 + \max\{d(v) - q, 0\}$ as in [10], where $d(v)$ is the out-degree of vertex $v$ and $q$ is a constant (which is set to 6 in our experiment). For datasets *frb30-15-1* and *frb35-17-1*, the original budget is set to 500 and ranges from 300 to 700; for dataset *aves*, it is set to 50 and ranges from 100 to 300. The sequence of 100 budget changes, randomly varying the current budget $B$ by $[-40, 40]$, is shown in Fig. 1(b) relative to the initial budget. The curves of average results over each time of change are plotted in Fig. 3.

Figure 3 shows that BPODC consistently performs the best across three datasets, although POMC sometimes matches its performance. However, the performance of other algorithms is less stable, for example, FPOMC performs moderately on *frb30-15-1* (Fig. 3(a)-(b)) but poorly on *aves* (Fig. 3(e)-(f)). EAMC underperforms GGA at both $t = 0.25T_G$ and $t = 0.5T_G$.

**Ablation Study.** As mentioned in Sect. 3, the warm-up stage of BPODC initially applies uniform selection for a period, enhancing the effectiveness of subsequent biased selection. We test this by running BPODC-cold on the *graph100* dataset with $t = 0.25T_G$, using only biased selection without a warm-up. Figure 4(a) shows that BPODC-cold has a performance gap compared to BPODC during the first 20 changes, yet achieves similar performance afterward. Despite solely using biased selection, BPODC-cold still outperforms POMC. This implies the effectiveness of both biased selection and warm-up.

Next, we conduct experiments on the dataset *frb35-17-1* for influence maximization in an extreme scenario. The number of objective evaluations for BPODC, POMC, EAMC, and FPOMC is set to 1000, including for the warm-up stage of BPODC. The results are plotted in Fig. 4(b). As expected, the EAs do not surpass GGA, given that the available time resources are extremely limited-merely 0.006 times that of the greedy algorithm GGA. However, BPODC demonstrates a significantly faster adaptation speed compared to other EAs and maintains a performance level similar to GGA in the later stages (at 80th-100th changes).

## 6 Conclusion

This paper proposes BPODC, an enhancement of POMC that uses biased selection and warm-up strategies for subset selection with dynamic cost constraints. We prove that BPODC can maintain the best known $(\alpha_f/2)(1-e^{-\alpha_f})$-approximation guarantee when the budget changes. Experiments on influence maximization and maximum coverage show that BPODC adapts faster and more effectively to budget changes, utilizing a running time that is less than that of the greedy algorithm GGA. BPODC always achieves the best performance empirically. In the future, it is interesting to examine the performance of BPODC in more applications.

## References

1. Bäck, T.: Evolutionary Algorithms in Theory and Practice: Evolution Strategies, Evolutionary Programming. Genetic Algorithms. Oxford University Press, Oxford, UK (1996)
2. Bian, C., Feng, C., Qian, C., Yu, Y.: An efficient evolutionary algorithm for subset selection with general cost constraints. In: Proceedings of the 34th AAAI Conference on Artificial Intelligence (AAAI 2020), pp. 3267–3274. New York, NY (2020)
3. Bian, C., Qian, C., Neumann, F., Yu, Y.: Fast Pareto optimization for subset selection with dynamic cost constraints. In: Proceedings of the 30th International Joint Conference on Artificial Intelligence (IJCAI 2021), pp. 2191–2197. Montreal, Canada (2021)
4. Bian, C., Zhou, Y., Qian, C.: Robust subset selection by greedy and evolutionary Pareto optimization. In: Proceedings of the 31st International Joint Conference on Artificial Intelligence (IJCAI 2022), pp. 4726–4732. Vienna, Austria (2022)
5. Bossek, J., Neumann, F., Peng, P., Sudholt, D.: Runtime analysis of randomized search heuristics for dynamic graph coloring. In: Proceedings of the 21st ACM Conference on Genetic and Evolutionary Computation Conference (GECCO 2019), pp. 1443–1451. Prague, Czech Republic (2019)
6. Coello, C.A.C., Lamont, G.B., van Veldhuizen, D.A.: Evolutionary Algorithms for Solving Multi-Objective Problems. Springer, New York, NY (2007)
7. Do, A.V., Neumann, F.: Pareto optimization for subset selection with dynamic partition matroid constraints. In: Proceedings of the 35th AAAI Conference on Artificial Intelligence (AAAI 2021), pp. 12284–12292. Virtual (2021)
8. Feige, U.: A threshold of ln n for approximating set cover. J. ACM **45**(4), 634–652 (1998)
9. Friedrich, T., Neumann, F.: Maximizing submodular functions under matroid constraints by evolutionary algorithms. Evol. Comput. **23**(4), 543–558 (2015)
10. Harshaw, C., Feldman, M., Ward, J., Karbasi, A.: Submodular maximization beyond non-negativity: guarantees, fast algorithms, and applications. In: Proceedings of the 36th International Conference on Machine Learning (ICML 2019), pp. 2634–2643. Long Beach, California (2019)
11. Hong, W.J., Yang, P., Tang, K.: Evolutionary computation for large-scale multi-objective optimization: a decade of progresses. Int. J. Autom. Comput. **18**(2), 155–169 (2021)
12. Kempe, D., Kleinberg, J., Tardos, É.: Maximizing the spread of influence through a social network. In: Proceedings of the 9th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD 2003), pp. 137–146. Washington, DC (2003)
13. Krause, A., Singh, A., Guestrin, C.: Near-optimal sensor placements in Gaussian processes: theory, efficient algorithms and empirical studies. J. Mach. Learn. Res. **9**, 235–284 (2008)
14. Laumanns, M., Thiele, L., Zitzler, E.: Running time analysis of multiobjective evolutionary algorithms on pseudo-Boolean functions. IEEE Trans. Evol. Comput. **8**(2), 170–182 (2004)

15. Nemhauser, G.L., Wolsey, L.A., Fisher, M.L.: An analysis of approximations for maximizing submodular set functions - I. Math. Program. **14**(1), 265–294 (1978)
16. Qian, C.: Distributed Pareto optimization for large-scale noisy subset selection. IEEE Trans. Evol. Comput. **24**(4), 694–707 (2020)
17. Qian, C.: Multi-objective evolutionary algorithms are still good: maximizing monotone approximately submodular minus modular functions. Evol. Comput. **29**(4), 463–490 (2021)
18. Qian, C., Bian, C., Feng, C.: Subset selection by Pareto optimization with recombination. In: Proceedings of the 34th AAAI Conference on Artificial Intelligence (AAAI 2020), pp. 2408–2415. New York, NY (2020)
19. Qian, C., Liu, D., Feng, C., Tang, K.: Multi-objective evolutionary algorithms are generally good: maximizing monotone submodular functions over sequences. Theoret. Comput. Sci. **943**, 241–266 (2023)
20. Qian, C., Liu, D., Zhou, Z.: Result diversification by multi-objective evolutionary algorithms with theoretical guarantees. Artif. Intell. **309**, 103737 (2022)
21. Qian, C., Shi, J., Yu, Y., Tang, K.: On subset selection with general cost constraints. In: Proceedings of the 26th International Joint Conference on Artificial Intelligence (IJCAI 2017), pp. 2613–2619. Melbourne, Australia (2017)
22. Qian, C., Shi, J., Yu, Y., Tang, K., Zhou, Z.: Optimizing ratio of monotone set functions. In: Proceedings of the 26th International Joint Conference on Artificial Intelligence (IJCAI 2017), pp. 2606–2612. Melbourne, Australia (2017)
23. Qian, C., Shi, J., Yu, Y., Tang, K., Zhou, Z.: Subset selection under noise. In: Advances in Neural Information Processing Systems (NeurIPS 2017), vol. 30, pp. 3560–3570. Long Beach, CA (2017)
24. Qian, C., Yu, Y., Tang, K.: Approximation guarantees of stochastic greedy algorithms for subset selection. In: Proceedings of the 27th International Joint Conference on Artificial Intelligence (IJCAI 2018), pp. 1478–1484. Stockholm, Sweden (2018)
25. Qian, C., Yu, Y., Tang, K., Yao, X., Zhou, Z.: Maximizing submodular or monotone approximately submodular functions by multi-objective evolutionary algorithms. Artif. Intell. **275**, 279–294 (2019)
26. Qian, C., Yu, Y., Zhou, Z.: Subset selection by Pareto optimization. In: Advances in Neural Information Processing Systems (NeurIPS 2015), vol. 28, pp. 1765–1773. Montreal, Canada (2015)
27. Qian, C., Zhang, Y., Tang, K., Yao, X.: On multiset selection with size constraints. In: Proceedings of the 32nd Conference on Artificial Intelligence (AAAI 2018), pp. 1395–1402. New Orleans, LA (2018)
28. Roostapour, V., Neumann, A., Neumann, F., Friedrich, T.: Pareto optimization for subset selection with dynamic cost constraints. Artif. Intell. **302**, 103597 (2022)
29. Rossi, R., Ahmed, N.: The network data repository with interactive graph analytics and visualization. In: Proceedings of the 29th AAAI Conference on Artificial Intelligence (AAAI 2015), pp. 4292–4293. Austin,Texas (2015)
30. Wu, T., Qian, H., Liu, Z., Zhou, J., Zhou, A.: Bi-objective evolutionary Bayesian network structure learning via skeleton constraint. Front. Comp. Sci. **17**(6), 176350 (2023)
31. Zhang, H., Vorobeychik, Y.: Submodular optimization with routing constraints. In: Proceedings of the 30th Conference on Artificial Intelligence (AAAI 2016), pp. 819–826. Phoenix, AZ (2016)
32. Zhang, L., Sun, X., Yang, H., Cheng, F.: Sparsity preserved Pareto optimization for subset selection. IEEE Transactions on Evolutionary Computation (2023)
33. Zhou, Z., Yu, Y., Qian, C.: Evolutionary Learning: Advances in Theories and Algorithms. Springer, Singapore (2019). https://doi.org/10.1007/978-981-13-5956-9

# Reaching Pareto Front Shape Invariance with a Continuous Multi-objective Ant Colony Optimization Algorithm

Rodolfo Humberto Tamayo[1], Jesús Guillermo Falcón-Cardona[1(✉)], and Carlos A. Coello Coello[2]

[1] Tecnologico de Monterrey, School of Engineering and Sciences, Ave. Eugenio Garza Sada 2501 Sur, Col: Tecnológico, 64700 Monterrey, N.L., Mexico
A01282633@tec.mx, jfalcon@tec.mx

[2] CINVESTAV-IPN, Departamento de Computación, Av. IPN No. 2508, Col. San Pedro Zacatenco, 07360 Ciudad de México, Mexico
carlos.coellocoello@cinvestav.mx

**Abstract.** Generating Pareto Front Approximations with good convergence, uniformity, and spread regardless of the geometry of the Pareto Front remains as an open problem. Many Multi-Objective Evolutionary Algorithms (MOEAs) have been proposed for this aim achieving remarkable results. However, the utilization of Swarm Intelligence algorithms such as Multi-Objective Ant Colony Optimization Algorithms (MOACOs) has been scarcely studied. In this paper, we propose a Geometric-Invariant MOACO$_{\mathbb{R}}$ (GI-MOACO$_{\mathbb{R}}$) designed to tackle multi-objective optimization problems with a continuous decision space. According to our experimental results, GI-MOACO$_{\mathbb{R}}$ outperforms the existing MOACOs for continuous search spaces and it is competitive with respect to state-of-the-art MOEAs on several test suites with regular and irregular Pareto Front geometries. To the best of the author's knowledge, GI-MOACO$_{\mathbb{R}}$ is the first Pareto-Front-Shape invariant MOACO.

**Keywords:** Multi-Objective Ant Colony Optimization · Pareto Front Shape Invariance · Continuous Decision Space · Pair-Potential Energy

## 1 Introduction

A Pareto Front (PF) is the image of the solution to a Multi-Objective Optimization Problem (MOP). A PF is a manifold of dimension at most $m-1$ that represents the trade-offs among the $m \geq 2$ conflicting objectives $f_i : \Omega \subseteq \mathbb{R}^n \rightarrow \mathbb{R}$ [22].

The first author acknowledges support given by Tecnológico de Monterrey and Consejo Nacional de Humanidades, Ciencias y Tecnologías (CONAHCYT) to pursue graduate studies under the CVU number 1238924. Carlos A. Coello Coello is a member of the Faculty of Excellence at the School of Engineering and Sciences of Tecnologico de Monterrey as part of a sabbatical leave.


M. Affenzeller et al. (Eds.): PPSN 2024, LNCS 15151, pp. 252–267, 2024.
https://doi.org/10.1007/978-3-031-70085-9_16

In benchmark problems and real-world applications, the geometric characteristics of a PF may drastically change. In other words, a PF could be convex, linear, concave, disconnected, degenerate, or mixed. Hence, Multi-Objective Evolutionary Algorithms (MOEAs), which are metaheuristic methods to approximate the solution to a MOP, should be capable of generating an accurate and finite representation of a PF, regardless of its geometry [2]. However, Ishibuchi *et al.* pointed out that the performance of some MOEAs depends on the PF geometry [16].

In recent years, two main design strategies have been proposed to alleviate the performance dependence of MOEAs. On the one hand, MOEAs with adaptive reference point sets transform a set of objective vectors throughout the evolutionary process, aiming to generate a Pareto Front Approximation (PFA) that properly represents the PF [19–21,24]. On the other hand, Multi-Indicator MOEAs leverage the strengths of multiple Quality Indicators[1] (QIs) to construct selection mechanisms that increase the selection pressure towards the PF, exhibiting a specific distribution of points [10,11,14,17]. All these MOEAs have shown promising results when tackling MOPs with different PF shapes [15,16,26]. However, Swarm Intelligence metaheuristics, such as Multi-Objective Ant Colony Optimization algorithms (MOACOs) [12], have been scarcely studied with the aim of a PF shape invariance behavior.

MOACO$_{\mathbb{R}}$ [13] and the Indicator-based MOACO$_{\mathbb{R}}$ (iMOACO$_{\mathbb{R}}$) [7] were proposed to tackle MOPs with a continuous decision space. They have obtained competitive results against MOEAs at tackling MOPs with degenerate and disconnected PF shapes [7,13]. Hence, there may be specific MOPs where MOACOs can potentially discover and exploit complex relations among the variables that are difficult to MOEAs [7,13]. It is worth noting that both algorithms use the mechanisms of ACO$_{\mathbb{R}}$ to explore the decision space and to create new solutions [23]. Furthermore, MOACO$_{\mathbb{R}}$ and iMOACO$_{\mathbb{R}}$ use an archive with $k$ decision vectors as a pheromone matrix to model the decision space using $n$ Gaussian-kernel Probability Density Functions (GKPDFs). Both MOACOs differ in the selection mechanisms adopted to update the pheromone matrix. On the one hand, MOACO$_{\mathbb{R}}$ uses the selection mechanisms of NSGA-II, thus, it loses selection pressure in high-dimensional objective spaces [3]. On the other hand, iMOACO$_{\mathbb{R}}$ replaces the selection operators of NSGA-II by a survival selection mechanism based on the $R2$ indicator [1,14]. Unlike MOACO$_{\mathbb{R}}$, iMOACO$_{\mathbb{R}}$ can solve MOPs with more than three objectives and it generates uniform PFAs if the geometry of the PF is correlated with an $m$-dimensional simplex. Thus, the performance of iMOACO$_{\mathbb{R}}$ depends on the PF shape.

In addition to the issues mentioned above of MOACO$_{\mathbb{R}}$ and iMOACO$_{\mathbb{R}}$, the use of ACO$_{\mathbb{R}}$ raises other problems. ACO$_{\mathbb{R}}$ shows a strong dependence on a parameter $\xi > 0$ that controls the evaporation of the pheromones and, thus, the convergence behavior. As a result, ACO$_{\mathbb{R}}$ has shown an extremely poor performance on multi-modal single-objective optimization problems. However, if $\xi$ could take large values without reducing its convergence capability, solutions

[1] A unary Quality Indicator is a set function that assigns a real value to a PFA depending on its degree of proximity to the PF, spread, or uniformity [18].

with high diversity would be reached. Addressing these issues would enhance the performance of $ACO_{\mathbb{R}}$ and, possibly, that of $MOACO_{\mathbb{R}}$ and $iMOACO_{\mathbb{R}}$.

In this paper, we enhance $ACO_{\mathbb{R}}$ to improve its exploration capabilities. Then, we propose the Geometric-Invariant $MOACO_{\mathbb{R}}$ that uses the enhanced $ACO_{\mathbb{R}}$ and a diversity-preserving mechanism based on pair-potential energy [6] to deal with MOPs regardless of their PF geometry. We tested GI-$MOACO_{\mathbb{R}}$ on several test suites with regular and irregular PF shapes on multiple QIs. Our experimental results indicate that GI-$MOACO_{\mathbb{R}}$ outperforms $MOACO_{\mathbb{R}}$ and $iMOACO_{\mathbb{R}}$, and it is competitive with state-of-the-art MOEAs designed to tackle MOPs with regular and irregular PF geometries. Thus, to the authors' best knowledge, GI-$MOACO_{\mathbb{R}}$ is the first PF-shape-invariant MOACO, highlighting that this work shows that Swarm Intelligence-based algorithms can have this property.

The structure of this paper is organized as follows. Section 2 presents the background to make the paper self-contained. Section 3 outlines our proposed GI-$MOACO_{\mathbb{R}}$. Section 4 describes our experimental settings and Sect. 5 discusses the results. Finally, Sect. 6 provides our main conclusions as well as some possible paths for future research.

## 2 Background

In this section, we first describe $ACO_{\mathbb{R}}$ which is the search engine of GI-$MOACO_{\mathbb{R}}$. Then, we defined a MOP and some terms important in Multi-Objective Optimization. Finally, we briefly introduce Pair-Potential Energy.

### 2.1 $ACO_{\mathbb{R}}$

$ACO_{\mathbb{R}}$ is an ACO for continuous search spaces proposed in 2008 [23]. A distinctive characteristic of $ACO_{\mathbb{R}}$ is the utilization of a pheromone matrix $\mathcal{T}$ that stores the best $k$ solutions sorted in ascending order by the objective function. Hence, each solution $\vec{x}_l \in \mathcal{T}$ has a rank $l$ according to its position in the ordering. For each dimension $i = 1, \ldots, n$ of $\Omega$, where $\Omega \subseteq \mathbb{R}^n$ is the decision space[2], $ACO_{\mathbb{R}}$ models the promising search region with a Gaussian-kernel Probability Density Function ($G^i(z)$), using the solutions in $\mathcal{T}$. Thus, the promising region is defined as $G^i(z) = \sum_{l=1}^{k} w_l g_l^i(z, \mu_l^i, \sigma_l^i)$, where $w_l \geq 0$ is a weight factor and $g_l^i(z, \mu_l^i, \sigma_l^i)$ is a Gaussian function with mean $\mu_l^i = x_{li}$ (the $i^{\text{th}}$ decision variable of $\vec{x}_l$) and the standard deviation $\sigma_l^i$ is defined as follows:

$$\sigma_l^i = \xi \sum_{r=1}^{k} \frac{|x_{ri} - x_{li}|}{k-1}, \tag{1}$$

where $\xi \geq 0$, known as evaporation rate, is a user-defined parameter that controls convergence. Small values of $\xi$ increase the convergence behavior of $ACO_{\mathbb{R}}$. To

[2] $\Omega = [l_1, u_1] \times [l_2, u_2] \times \cdots \times [l_n, u_n]$, where $l_i$ and $u_i$ define the lower and upper bounds, respectively, of the $i^{\text{th}}$ decision variable.

generate a new candidate solution ($\vec{x}_{\text{new}}$), each $\text{ant}_j \in \mathcal{C} = \{\text{ant}_1, \ldots, \text{ant}_M\}$ first selects a guiding pheromone $\vec{x}_l \in \mathcal{T}$ with probability $p_l$. Then, $\text{ant}_j$ samples all $G_l^i(z), i = 1, \ldots, n$ to construct a new candidate solution. The probability $p_l$ of selecting $\vec{x}_l$ is directly proportional to its weight $w_l$ that is defined as follows:

$$w_l = \frac{1}{qk\sqrt{2\pi}} \cdot e^{-\frac{(l-1)^2}{2q^2k^2}}, \tag{2}$$

where $q \geq 0$ is a user-defined parameter that controls the diversification process of the search, where large values of $q$ promote a higher degree of elitism.

### 2.2 Multi-objective Optimization

In this paper, we solve unconstrained MOPs[3] assuming, without loss of generality, the minimization of all the objectives:

$$\min_{\vec{x} \in \Omega} \left\{ f(\vec{x}) := [f_1(\vec{x}), f_2(\vec{x}), ..., f_m(\vec{x})]^{\mathsf{T}} \right\}, \tag{3}$$

where $\vec{x} = [x_1, x_2, ..., x_n]^{\mathsf{T}}$ is an $n$-dimensional decision vector and $\Lambda = f(\Omega) \subseteq \mathbb{R}^m$ is the objective space. $f : \Omega \to \Lambda$ is a vector-valued function based on $m(\geq 2)$ objectives $f_i : \Omega \to \mathbb{R}, i = 1, 2, \ldots, m$. In MOPs, the definition of optimality is commonly based on the Pareto dominance relation. Given $\vec{x}, \vec{y} \in \Omega$, we say that $\vec{x}$ dominates $\vec{y}$ (denoted as $\vec{x} \prec \vec{y}$) if $\forall i = 1, 2, \ldots, m, f_i(\vec{x}) \leq f_i(\vec{y})$ and there exists at least an index $j \in \{1, 2 \ldots, m\}$ such that $f_j(\vec{x} < f_j(\vec{y})$. Then, $\vec{x}_* \in \Omega$ is Pareto optimal if there is no other $\vec{x} \in \Omega$ such that $\vec{x} \prec \vec{x}_*$. The solution to a MOP is the Pareto Set PS $= \{\vec{x}_* \in \Omega \mid \vec{x}_*$ is Pareto optimal$\}$ and its image $f(\text{PS})$ is the Pareto Front. A PFA is a set of solutions that approximate a PF where the solutions are mutually non-dominated, i.e., given $\vec{x}, \vec{y} \in \mathcal{A}$, $\vec{x} \not\prec \vec{y}$ and $\vec{y} \not\prec \vec{x}$.

### 2.3 Potential Energy

A Pair-Potential Energy function (PPF), denoted as $\mathcal{K}$, measures the interaction between particles at positions $\vec{u}, \vec{v} \in \mathbb{R}^m$. In physics, there is a wide range of PPFs but just a few of them have been utilized in EMOO [8]. Two representative PPFs are the Coulomb's law $\mathcal{K}^{\text{COU}}(\vec{u}, \vec{v}) = \frac{q_1 q_2}{4\pi\epsilon_0} \cdot \|\vec{u} - \vec{v}\|^{-2}$, where $\frac{1}{4\pi\epsilon_0}$ is the Coulomb's constant, being $\epsilon_0 \approx 8.8542 \times 10^{-12}$[F/m] the vacuum permittivity, and the Riesz $s$-kernel, $\mathcal{K}^{\text{RSE}}(\vec{u}, \vec{v}) = \|\vec{u} - \vec{v}\|^{-s}$, where $s > 0$. In the context of EMOO, Falcón-Cardona *et al.* [8] proposed to set $q_1 = \|\vec{u}\|$ and $q_2 = \|\vec{v}\|$. Given $\mathcal{K} : \mathbb{R}^m \times \mathbb{R}^m \to \mathbb{R}$, the total potential energy ($U$) of an $N$-point set $\mathcal{A} = \{\vec{a}_1, \vec{a}_2, \ldots, \vec{a}_N\}$, with $N \geq 2$, where the lower the $U$ of $\mathcal{A}$ the higher the diversity of the set [8], is given by:

[3] In the Evolutionary Multi-Objective Optimization (EMOO) community, the term Many-Objective Optimization Problem (MaOP) is used to denote problems having more than three objectives.

$$U^{\mathcal{K}}(\mathcal{A}) = \sum_{i=1}^{N} \sum_{\substack{j=i \\ j \neq i}}^{N} \mathcal{K}(\vec{a}_i, \vec{a}_j). \tag{4}$$

In recent years, $U$ has been used in MOEAs to increase the diversity of PFAs, regardless of the PF geometry. MOEAs use selection mechanisms based on heuristic algorithms to approximate $\mathcal{A}^* = \arg\min_{\substack{\mathcal{A} \subset \mathcal{S} \\ |\mathcal{A}|=N}} U(\mathcal{A})$, where $N << |\mathcal{S}|$. These selection mechanisms have been used as pruning policies in external archives [9], density estimators [14], and for the generation of reference point sets [6]. Mainly, these selection mechanisms are based on a fast greedy removal algorithm that iteratively deletes from $\mathcal{S}$ the point $\vec{a}_{\text{worst}} = \arg\max_{\vec{a} \in \mathcal{S}} \Delta(\mathcal{S}, \vec{a})$, where $\Delta(\mathcal{S}, \vec{a}) = U(\mathcal{S}) - U(\mathcal{S} \setminus \{\vec{a}\})$ is the so-called contribution to $U$ [6].

## 3 Our Proposal: GI-MOACO$_{\mathbb{R}}$

This section describes our proposed GI-MOACO$_{\mathbb{R}}$, which aims to tackle regular and irregular MOPs. GI-MOACO$_{\mathbb{R}}$ utilizes an enhanced version of ACO$_{\mathbb{R}}$ that solves the dependency to $\xi$ and the lack of diversification when selecting solutions from the pheromone matrix. GI-MOACO$_{\mathbb{R}}$ also employs a PPF-based selection mechanism to promote diversity regardless of the PF geometry by performing a subset selection on $\mathcal{T}$. In the following sections, we describe three mechanisms (namely, **Classification, Construction,** and **Diversification**) to improve the performance of ACO$_{\mathbb{R}}$. Finally, we introduce our proposed GI-MOACO$_{\mathbb{R}}$.

**Algorithm 1.** Classification

**Input:** $\mathcal{T}$: Pheromone matrix; $\overline{\sigma}$: Deviation threshold
**Output:** CHOICES, STAGNATED.
1: CHOICES, STAGNATED $\leftarrow \emptyset, \emptyset$
2: **for** $i = 1$ to $n$ **do**
3: $\quad \sigma'_{x_i} \leftarrow$ Standard-deviation$(\mathcal{T}, i)$
4: $\quad \sigma'_{x_i} \leftarrow (\sigma'_{x_i} - l_i)/(u_i - l_i)$
5: $\quad$ **if** $\sigma'_{x_i} > \overline{\sigma}$ **then**
6: $\quad\quad$ CHOICES $\leftarrow$ CHOICES $\cup \{i\}$
7: $\quad$ **else**
8: $\quad\quad$ STAGNATED $\leftarrow$ STAGNATED $\cup \{i\}$
9: $\quad$ **end if**
10: **end for**
11: **return** CHOICES, STAGNATED

### 3.1 Detection of Stagnated Variables

In ACO$_{\mathbb{R}}$, each ant$_j \in \mathcal{C}$ selects a guiding pheromone $\vec{x}_l \in \mathcal{T}$ to generate a new candidate solution ($\vec{x}_{\text{new}}$) by sampling $g_l^i(z), i = 1 \ldots, n$. The sampling is done without dispersion information of each dimension in the pheromone matrix. For some problems, especially multi-modal ones, pheromones in $\mathcal{T}$ may converge to

a single solution. Consequently, $\sigma_l^i \approx 0$ for all $l = 1, \ldots, k$ and $i = 1, \ldots, n$ which causes a lack of exploration. To improve the exploration ability of ACO$_{\mathbb{R}}$, we propose a mechanism (see Algorithm 1) to classify the decision variables in $\mathcal{T}$ into stagnated or not. This mechanism requires a threshold $\overline{\sigma} \in (0, 1)$ that indicates the minimum permissible deviation value. Line 1 initializes the sets `CHOICES` and `STAGNATED`. For each dimension $i$, we calculate the standard deviation ($\sigma_{x_i}$) of the $i^{\text{th}}$ decision variable in $\mathcal{T}$ and we normalize it using the lower and upper bounds, $l_i$ and $u_i$, respectively, of this decision variable. In case that $\sigma'_{x_i} \geq \overline{\sigma}$, the dimension $i$ is assigned to `CHOICES`, otherwise $i$ is assigned to `STAGNATED`. Hence, `STAGNATED` will contain the stagnated decision variables and `CHOICES` the rest. The smaller the value of $\overline{\sigma}$, the stricter is the constraint to consider a variable as stagnated. According to our experimental observations, well-spread values have $0.2 \leq \sigma'_{x_i} \leq 0.4$. Hence, we recommend setting $\overline{\sigma} \leq 0.15$, as larger values might prematurely assign non-stagnated decision variables to set `STAGNATED`, compromising convergence.

### 3.2 Constructing New Candidate Solutions

To improve the exploration and exploitation ability of our approach, we propose here a new method to construct a candidate solution $\vec{x}_{\text{new}} \in \Omega$. To this aim, ant$_j$ needs to provide as input a guiding pheromone $\vec{x}_l$. In contrast to the original ACO$_{\mathbb{R}}$ where an ant samples each $g_l^i(z, \mu_l^i, \sigma_l^i)$, in our proposal only $\overline{n}$ decision variables from `CHOICES` are selected to construct $\vec{x}_{\text{new}}$. Algorithm 2 sets the value of $\overline{n}$ by iteratively calculating $\overline{n} = \min(\overline{n}, \sim \mathcal{U}_{\mathbb{N}}[1, |\texttt{CHOICES}|])$, where $\sim \mathcal{U}_{\mathbb{N}}[1, |\texttt{CHOICES}|]$ generates a random natural number in the range $[1, |\texttt{CHOICES}|$. Then, in line 5, $\overline{n}$ random and distinct variables from `CHOICES` are assigned to $\mathcal{I}$. To construct $\vec{x}_{\text{new}} \in \Omega$, the Gaussian kernels associated with each $i \in \mathcal{I}$ are sampled. The remaining variables[4] are copied directly from the guiding pheromone $\vec{x}_l$. It is worth emphasizing that modifying fewer decision variables allows the use of larger values of $\xi$. As a result, it maintains convergence while increasing diversity. In consequence, this reduces the dependency of ACO$_{\mathbb{R}}$ to $\xi$.

**Algorithm 2.** Construction

**Input:** $\vec{x}_l$: Guiding pheromone; $\xi$: Evaporation rate; $q$: Diversification rate; `CHOICES`: Set of non-stagnated variables
**Output:** $\vec{x}_{\text{new}}$.
1: $\overline{n} \leftarrow n$
2: **for** objective $j = 1$ to $m$ **do**
3: $\quad \overline{n} \leftarrow \min(\overline{n}, \sim \mathcal{U}_{\mathbb{N}}[1, |\texttt{CHOICES}|])$
4: **end for**
5: $\mathcal{I} \leftarrow$ Randomly select $\overline{n}$ distinct variables from `CHOICES`
6: Calculate $\sigma_l^i$ using $\xi$ and $q$
7: $x_{\text{new},i} \leftarrow$ Sample $g_l^i(z, \mu_l^i, \sigma_l^i), \forall i \in \mathcal{I}$
8: $x_{\text{new},i} \leftarrow x_{li}, \forall i \in \{1, 2, \ldots, n\} \setminus \mathcal{I}$
9: **return** $\vec{x}_{\text{new}}$

[4] Note that each ant$_j \in \mathcal{C}$ might modify a different number $\overline{n}$ of decision variables.

**Algorithm 3.** Diversification

**Input:** $\mathcal{K}$: PPF; $\mathcal{M}$: External archive; $\mathcal{T}$: Pheromone matrix; $\alpha$: Percentage of pheromone replacement; `STAGNATED`: Set of stagnated decision variables
**Output:** $\mathcal{T}$: Updated pheromone matrix
1: $k \leftarrow |\mathcal{T}|$
2: **if** $|\mathcal{M}| > k$ **then**
3: $\quad \mathcal{T} \leftarrow U^{\mathcal{K}}$**-Based Selection Mechanism**$(\mathcal{M}, \mathcal{K}, k)$
4: **else**
5: $\quad \mathcal{N} \leftarrow$ Randomly select $k - |\mathcal{M}|$ guiding pheromones $\vec{x} \in \mathcal{M}$
6: $\quad \mathcal{T} \leftarrow \mathcal{M} \cup \mathcal{N}$
7: **end if**
8: $i \leftarrow$ Select a random decision variable from `STAGNATED`
9: Randomly permute the pheromones of $\mathcal{T}$
10: **for** $j = 1$ to $\lceil k \times \alpha \rceil$ **do**
11: $\quad \mathcal{T}_{ji} \leftarrow\sim \mathcal{U}_{\mathbb{R}}[l_i, u_i]$
12: **end for**
13: **return** $\mathcal{T}$

### 3.3 Avoiding Stagnation of Decision Variables

The original ACO$_{\mathbb{R}}$ suffers from stagnation because the best solutions are stored in the pheromone matrix. When the solutions in $\mathcal{T}$ are similar, $\sigma_l^i \approx 0$ (see Eq. (1)) for all $l = 1, \ldots, k$ and $i = 1, \ldots, n$. In consequence, an ant cannot explore more regions of $\Omega$. Algorithm 3 avoids stagnation by increasing the diversity of solutions in $\mathcal{T}$, using an external archive $\mathcal{M}$. First, if $\mathcal{M}$ has more than $k = |\mathcal{T}|$ solutions, the content of $\mathcal{T}$ is replaced by a subset of $k$ solutions from $\mathcal{M}$ using a PPF-based subset selection as in [6]. Otherwise, $k - |\mathcal{M}|$ solutions are randomly selected from $\mathcal{M}$ to be inserted in $\mathcal{T}$. Lines 8 and 9 randomly permute the pheromones of $\mathcal{T}$ and randomly select a decision variable $i$ from `STAGNATED`. Finally, $\lceil k \times \alpha \rceil$ solutions $\in \mathcal{T}$ are perturbed by assigning a random real number $\sim \mathcal{U}_{\mathbb{R}}[l_i, u_i]$ to the $i^{\text{th}}$ component of the solutions. Thus, this injection of randomness aims to increase the exploration ability and diversification of ACO$_{\mathbb{R}}$. As a result, the $i^{\text{th}}$ component of $\mathcal{T}$ will be added to the set `CHOICES` by Algorithm 1.

### 3.4 GI-MOACO$_{\mathbb{R}}$

Algorithm 4 outlines the main loop of GI-MOACO$_{\mathbb{R}}$, which uses a bounded external archive $\mathcal{M}$ of maximum size $3\mu$ to store non-dominated solutions found during the search process. The size of the set $\mathcal{M}$ is limited because larger sizes require more computational and time resources to update, with a complexity of $\mathcal{O}(|\mathcal{M}|^2)$ [3]. Additionally, previous experimentation has shown that performance is not compromised once the size reaches $3\mu$. In case that $\mathcal{M}$ has more than $3\mu$ solutions, we apply a $U^{\mathcal{K}}$-based subset selection using the algorithm provided in [6]. Lines 7 to 24 sketch the main loop of GI-MOACO$_{\mathbb{R}}$. First, Algorithm 1 classifies the decision variables into `CHOICES` and `STAGNATED`. Then, each $\text{ant}_j \in \mathcal{C}, j = 1, \ldots, k$ generates a new candidate solution using Algorithm 2, adding it to $\mathcal{T}$. In line 14, $\mathcal{M}$ and $\mathcal{T}$ are joined and, then, $\mathcal{T}$ is sorted with the Non-Dominated Sorting Algorithm [3] and pruned if necessary. In line 17, we ensure that $\mathcal{M}$ has at most $3\mu$ solutions. Since applying Algorithm 3 at each iteration

**Algorithm 4.** GI-MOACO$_{\mathbb{R}}$

**Input:** $\xi_0$: Initial evaporation rate; $q$: Diversification rate; $\mu$: Approximation size; $\overline{\sigma}$: Deviation threshold; $\alpha$: Percentage of pheromone replacement; $T_w$: Replacement time window; $\mathcal{K}$: PPF
**Output:** $\mathcal{M}$: Pareto front approximation

$k \leftarrow \lceil \mu/4 \rceil$
Randomly initialize pheromone matrix $\mathcal{T}$ of size $k$
Initialize external archive $\mathcal{M}$ with non-dominated solutions from $\mathcal{T}$
$\mathcal{M}_{max} \leftarrow 3\mu$
$\mathcal{T} \leftarrow U^{\mathcal{K}}$**-Based Selection Mechanism** $(\mathcal{T}, \mathcal{K}, k)$
$g, \xi \leftarrow 0, \xi_0$
**while** termination condition is not fulfilled **do**
    CHOICES, STAGNATED $\leftarrow$ **Classification** $(\mathcal{T}, \overline{\sigma})$
    **for** $j = 1$ to $k$ **do**
        ant$_j$ selects a guiding pheromone $\vec{x}_l \in \mathcal{T}$
        $\vec{x}_{\text{new}} \leftarrow$ **Construction** $(\vec{x}_l, \xi, q,$ CHOICES$)$
        $\mathcal{T} \leftarrow \mathcal{T} \cup \{\vec{x}_l\}$
    **end for**
    $\mathcal{M} \leftarrow \mathcal{M} \cup \mathcal{T}$
    $\mathcal{T} \leftarrow$ **Non-Dominated Sorting**$(\mathcal{T})$
    $\mathcal{T} \leftarrow U^{\mathcal{K}}$**-Based Selection Mechanism** $(\mathcal{T}, \mathcal{K}, k)$
    Prune $\mathcal{M}$ if $|\mathcal{M}| > \mathcal{M}_{max}$
    **if** $g \mod T_w == 0$ and |STAGNATED| $> 0$ **then**
        $\mathcal{T} \leftarrow$ **Diversification** $(\mathcal{K}, \mathcal{M}, \mathcal{T}, \alpha,$ STAGNATED$)$
    **end if**
    **if** 80% of process is completed **then**
        $\xi \leftarrow \xi_0/2$
    **end if**
    $g \leftarrow g + 1$
**end while**
$\mathcal{M} \leftarrow U^{\mathcal{K}}$**-Based Selection Mechanism** $(\mathcal{M}, \mathcal{K}, \mu)$
**return** $\mathcal{M}$

may produce a lack of convergence, we execute it every $T_w$ iterations only if |STAGNATED| $> 0$. Finally, $\xi$ is updated if 80% of the search process has been completed to encourage an exploitation behavior. GI-MOACO$_{\mathbb{R}}$ returns a subset of $\mu$ solutions from $\mathcal{M}$ using the $U^{\mathcal{K}}$-based subset selection.

## 4 Experimental Settings

This section is devoted to test the performance of GI-MOACO$_{\mathbb{R}}$[5]. We compared our proposal against MOACO$_{\mathbb{R}}$ [13] and iMOACO$_{\mathbb{R}}$[6] [7], and five state-of-the-art MOEAs[7] (designed to tackle both regular and irregular PF shapes): AdaW [19], AR-MOEA [24], SPEA2+SDE [17], RVEA-iGNG [20], and Two_Arch2 [28]. We performed 30 independent executions of each algorithm per test instance.

### 4.1 Benchmark Problems

We adopted the test suites Deb-Thiele-Laumanns-Zitzler (DTLZ) [4], Walking-Fish-Group (WFG) [15], their inverted versions DTLZ$^{-1}$ and WFG$^{-1}$ [16],

[5] The source code of GI-MOACO$_{\mathbb{R}}$ is available at https://github.com/Humberto-Tamayo/GI-MOACOR.git.
[6] MOACO$_{\mathbb{R}}$ and iMOACO$_{\mathbb{R}}$ were coded in Python 3.10.12. Their source code is available at https://github.com/Humberto-Tamayo/MOACOR-iMOACOR.git.
[7] We used the implementations from the PlatEMO platform [25].

**Table 1.** Characteristics of the PF geometries of the selected MOPs and the reference point for the HV calculation.

| MOP | PF geometry | Simplex-like | Reference point (HV) |
|---|---|---|---|
| DTLZ1 | Triangular (linear) | Yes | $(1, \ldots, 1)$ |
| DTLZ2 - DTLZ4 | Concave | Yes | $(2, \ldots, 2)$ |
| DTLZ5 & DTLZ6 | Concave ($m = 2$)<br>Degenerate ($m = 3$)<br>Unknown ($m > 3$) | Yes No No | $(2, \ldots, 2)$ |
| DTLZ7 | Disconnected | No | $(1, \ldots, 1, 21)$ |
| DTLZ1$^{-1}$ | Inverted triangular | No | $(1, \ldots, 1)$ |
| DTLZ2$^{-1}$ - DTLZ6$^{-1}$ | Convex | No | $(1, \ldots, 1)$ |
| DTLZ7$^{-1}$ | Disconnected | No | $(0.1, \ldots, 0.1, -10)$ |
| WFG1 | Mixed | Yes | $(3, 5, 7, \ldots, 2m + 1)$ |
| WFG2 | Disconnected | Yes | $(3, 5, 7, \ldots, 2m + 1)$ |
| WFG3 | Linear ($m = 2$)<br>Degenerate ($m \geq 3$) | Yes No | $(3, 5, 7, \ldots, 2m + 1)$ |
| WFG4 - WFG9 | Concave | Yes | $(3, 5, 7, \ldots, 2m + 1)$ |
| WFG1$^{-1}$ | Mixed | No | $(1, \ldots, 1)$ |
| WFG2$^{-1}$, WFG4$^{-1}$ - WFG9$^{-1}$ | Convex | No | $(1, \ldots, 1)$ |
| WFG3$^{-1}$ | Inverted triangular | No | $(1, \ldots, 1)$ |
| VIE1 | Convex | No | $(4, 5, 4)$ |
| VIE2 | Mixed | No | $(5, -15, -11)$ |
| VIE3 | Mixed | No | $(10, 18, 1)$ |
| IMOP1 | Convex | Yes | $(1.2, 1.2)$ |
| IMOP2 | Concave | Yes | $(1.2, 1.2)$ |
| IMOP3 | Disconnected | No | $(1.5, 1.2)$ |
| IMOP4 | Degenerate | No | $(1.2, 1.2, 1.2)$ |
| IMOP5 | Disconnected | No | $(1, 1, 2)$ |
| IMOP6 | Degenerate | No | $(1.2, 1.2, 1.2)$ |
| IMOP7 | Concave | No | $(1.2, 1.2, 1.2)$ |
| IMOP8 | Degenerate | No | $(1.2, 1.2, 3.2)$ |

respectively, with 2, 3, 5, and 7 objectives. Additionally, we employed the Irregular MOPs (IMOP) [26] and the Viennet problems (VIE) [27]. We selected these test suites because they cover a wide range of PF geometries and search difficulties.

Table 1 presents the main characteristics of the PF geometries of the selected MOPs, indicating if the geometry correlates with the simplex shape. For DTLZ and DTLZ$^{-1}$ problems, the number of variables was set to $n = m + K - 1$, where $m$ denotes the number of objectives, $K = 5$ for DTLZ1, $K = 10$ for DTLZ2-DTLZ6, and $K = 20$ for DTLZ7. The inverted versions share the same values of $K$. Regarding WFG problems, we set the tuples $(n, m, k_{\text{position}})$, where $k_{\text{position}}$ is the number of position-related parameters, to $(24, 2, 4)$, $(26, 3, 4)$, $(30, 5, 8)$, and $(34, 7, 12)$. For the IMOP test suite, the parameters $K, L, a_1, a_2$, and $a_3$ are set to 5, 5, 0.05, 0.05, and 10 as indicated by its authors [26].

### 4.2 Parameter Settings

For a fair comparison, all the algorithms use the same population size ($\mu$) and they consume the same number of function evaluations ($\text{FE}_{\max}$). Hence, we set $(\mu, m, \text{FE}_{\max})$ to $(120, 2, 40 \times 10^3)$, $(120, 3, 50 \times 10^3)$, $(126, 5, 70 \times 10^3)$, and

$(210, 7, 90 \times 10^3)$. These values are selected as they are standard values utilized in the EMOO literature [5]. Regarding GI-MOACO$_{\mathbb{R}}$, the parameters $\xi_0$, $q$, $\overline{\sigma}$, $\alpha$, and $T_w$, are set to 1.8, 0.5, 0.1, 0.5, and 4, respectively. The selection of values is based on previous experimentation. Additionally, $\mathcal{K}$ is set to $\mathcal{K}^{\text{COU}}$ for all the processes to ensure convergence and diversity as suggested by Falcón-Cardona et al. [6], except on Algorithm 3, where is set to $\mathcal{K}^{\text{RSE}}$ to take advantage of its theoretical properties. In MOACO$_{\mathbb{R}}$ and iMOACO$_{\mathbb{R}}$ the parameters $q$ and $\xi$ are set to 0.1 and 0.5, respectively, as suggested by their authors [7,13]. In MOACO$_{\mathbb{R}}$, the number of ants is set to 2. For all the selected MOEAs, the parameter values are set to their default values as in the PlatEMO platform [25]. The crossover and mutation probabilities are set to 1.0 and $1/n$, respectively, while both crossover and mutation distribution indexes are set to 20.

### 4.3 Quality Indicators

We adopted the Hypervolume indicator (HV) and the Inverted Generational Distance plus (IGD$^+$) to assess convergence towards the PF [18]. To measure the diversity of the PFAs, we used the Riesz $s$-energy $E_s = U^{\mathcal{K}^{\text{RSE}}}$ (see Eq. 4), where $s > 0$ is a user-supplied parameter. Table 1 lists the reference points used to calculate HV for each test problem. The calculation of IGD$^+$ requires a reference point set, constructed by merging all the PFAs generated by all algorithms for a given test instance and filtering out the non-dominated solutions. Then, we performed an $E_s$-based subset selection to obtain $100 \times m$ solutions with $s = m+1$ [6]. For $E_s$ assessment, we set $s = m+1$ to evaluate the normalized PFAs.

## 5 Discussion of Results

The main goal of this Section is to present the numerical comparison of GI-MOACO$_{\mathbb{R}}$ based on HV, IGD$^+$, and $E_s$. However, we first show how Algorithms 2 and 3 help to enhance the performance of the original ACO$_{\mathbb{R}}$. Then, we discuss here the comparison of GI-MOACO$_{\mathbb{R}}$ with MOACO$_{\mathbb{R}}$ and iMOACO$_{\mathbb{R}}$. Then, we focus on comparing GI-MOACO$_{\mathbb{R}}$ with the selected MOEAs. Due to space constraints, the complete numerical results for HV are provided in Tables SM-1 to SM-6 of the Supplementary Material (SM) available at https://github.com/Humberto-Tamayo/GI-MOACOR.git. Tables SM-7 to SM-12 show the results for IGD$^+$, and Tables SM-13 to SM-18 show the results for E$_s$. Table 2 presents a summary of HV-based results in Tables SM-1 to SM-6 from the SM. Additionally, Fig. 2 shows PFAs according to the first, second, third, and last rank in the comparison from Tables SM-1 to SM-6.

### 5.1 Discussion of the Enhanced ACO$_{\mathbb{R}}$

To show the effect of Algorithms 1 and 3 in ACO$_{\mathbb{R}}$, we measured the performance of ACO$_{\mathbb{R}}$ when turning on and off these algorithms. Moreover, we used

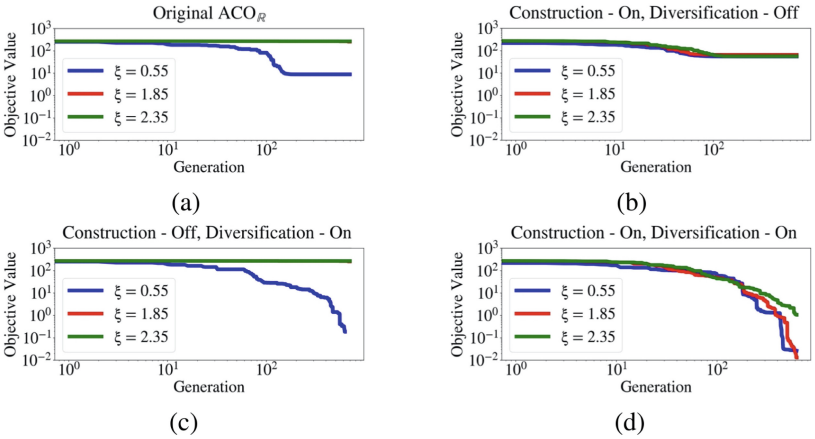



**Fig. 1.** Effect of Construction and Diversification algorithms in ACO$_{\mathbb{R}}$ on the Rastrigin Function. (a) Original ACO$_{\mathbb{R}}$, (b) Construction on and Diversification off, (c) Construction off and Diversification on, (d) Construction On and Diversification off.

$\xi = 0.55, 1.85, 2.35$ to identify its effect. We tested the algorithm with the multimodal Rastrigin Function with 20 decision variables with 35 thousand function evaluations. Figure 1(a) shows the performance of the original ACO$_{\mathbb{R}}$ when Algorithms 2 and 3 are turned off[8]. This illustrates that ACO$_{\mathbb{R}}$ fails to converge with high values of $\xi$ and gets stuck with $\xi = 0.55$. The effect of Algorithm 2 is shown in Fig. 1(b) for $\xi = 0.55, 1.85, 2.25$. In this case, all the algorithms have a similar behavior, showing a clear reduction in the dependency on $\xi$. On the other hand, the effect of Algorithm 3 is illustrated in Fig. 1(c) for $\xi = 0.55$, obtaining better results than in Fig. 1(a). Finally, Fig. 1(d) shows the effectiveness of combining Algorithms 2 and 3, outperforming the original ACO$_{\mathbb{R}}$. The enhancements of ACO$_{\mathbb{R}}$ increase convergence and reduce the dependency to $\xi$, thus improving diversity which is beneficial for MaOPs.

### 5.2 Comparison Against MOACOs

Table 2 shows that GI-MOACO$_{\mathbb{R}}$ outperforms MOACO$_{\mathbb{R}}$ and iMOACO$_{\mathbb{R}}$ at tackling multi-frontal MOPs such as DTLZ1 and WFG4. Additionally, the results associated with DTLZ4 and WFG8 show that GI-MOACO$_{\mathbb{R}}$ performs competitively against iMOACO$_{\mathbb{R}}$ in regular MOPs and MaOPs. In this regard, Tables SM-3, SM-5, SM-9, and SM-11 indicate that GI-MOACO$_{\mathbb{R}}$ outperforms iMOACO$_{\mathbb{R}}$ in most regular MaOPs from the DTLZ and WFG test suites. According to HV and IGD$^{+}$, GI-MOACO$_{\mathbb{R}}$ obtained better approximations in 17 out of 26 regular MaOPs with 5 and 7 objectives.

[8] Note that for Algorithms 2 and 3 to work, Algorithm 1 must be turned on.

**Table 2.** Mean and standard deviation (in parentheses) of HV values. The two best values are highlighted in gray scale, where the darker tone corresponds to the best one. The rank in the comparison for each value is shown in parentheses. The symbol # is placed where the best-ranked algorithm performs statistically better according to a one-tailed Wilcoxon rank-sum test using a significance level of 0.05.

| MOP | m | GI-MOACO$_{\mathbb{R}}$ | MOACO$_{\mathbb{R}}$ | iMOACO$_{\mathbb{R}}$ | AdaW | AR-MOEA | SPEA2+SDE | Two_Arch2 | RVEA-iGNG |
|---|---|---|---|---|---|---|---|---|---|
| VIE1 | 3 | 2.316e+01(4#) (1.995e-02) | 2.306e+01(5#) (3.042e-02) | 2.254e+01(8#) (6.831e-02) | 2.320e+01(3) (1.655e-02) | 2.289e+01(6) (7.504e-02) | 2.289e+01(7#) (1.425e-01) | 2.323e+01(1) (1.355e-02) | 2.322e+01(2) (1.989e-02) |
| VIE3 | 3 | 3.161e+01(1) (1.509e-03) | 3.152e+01(7#) (1.809e-02) | 3.149e+01(8#) (3.691e-02) | 3.160e+01(5) (1.624e-03) | 3.161e+01(2) (1.366e-03) | 3.160e+01(3) (7.785e-03) | 3.160e+01(4) (5.184e-03) | 3.159e+01(6) (4.069e-03) |
| IMOP6 | 3 | 9.959e-01(6#) (9.440e-03) | 9.786e-01(8#) (1.035e-02) | 9.842e-01(7#) (3.743e-03) | 1.053e+00(1) (1.460e-03) | 1.014e+00(4) (1.049e-01) | 1.011e+00(5#) (1.067e-01) | 1.051e+00(2#) (7.691e-04) | 1.021e+00(3#) (1.277e-01) |
| IMOP7 | 3 | 9.662e-01(2#) (3.304e-02) | 9.607e-01(4#) (9.753e-02) | 9.623e-01(3#) (1.887e-02) | 1.017e+00(1) (1.456e-01) | 4.016e-01(7) (2.769e-01) | 4.819e-01(6#) (3.449e-01) | 3.798e-01(8#) (2.370e-01) | 8.907e-01(5#) (2.059e-01) |
| DTLZ1 | 2 | 8.629e-01(6#) (8.770e-03) | 0.000e+00(7#) (0.000e+00) | 0.000e+00(8#) (0.000e+00) | 8.736e-01(3) (3.530e-04) | 8.736e-01(2) (4.751e-04) | 8.733e-01(5#) (5.185e-04) | 8.734e-01(4) (9.117e-04) | 8.736e-01(1) (2.328e-04) |
| | 3 | 9.663e-01(6#) (5.766e-03) | 0.000e+00(7#) (0.000e+00) | 0.000e+00(8#) (0.000e+00) | 9.742e-01(2#) (1.546e-04) | 9.743e-01(1) (1.254e-04) | 9.700e-01(5#) (1.818e-03) | 9.739e-01(4#) (2.098e-04) | 9.741e-01(3#) (9.345e-05) |
| | 5 | 9.946e-01(5#) (1.098e-02) | - | 0.000e+00(7#) (0.000e+00) | 9.987e-01(2#) (7.072e-05) | 9.987e-01(1) (1.657e-05) | 9.941e-01(6#) (1.837e-03) | 9.984e-01(4#) (5.122e-05) | 9.985e-01(3#) (8.291e-05) |
| | 7 | 9.558e-01 (6#) (8.0076e-02) | - | 0.000e+00 (7#) (0.000e+00) | 9.999e-01 (2#) (2.530e-05) | 9.999e-01 (1) (1.266e-06) | 9.983e-01 (5#) (5.641e-04) | 9.998e-01 (4#) (9.268e-06) | 9.999e-01 (3#) (3.129e-05) |
| DTLZ4 | 2 | 3.211e+00(2) (2.201e-04) | 3.207e+00(4) (4.573e-03) | 3.211e+00(1) (2.653e-05) | 3.209e+00(3) (4.143e-03) | 2.848e+00(7) (5.644e-01) | 3.090e+00(5) (3.694e-01) | 2.888e+00(6) (5.449e-01) | 2.767e+00(8) (5.936e-01) |
| | 3 | 7.408e+00(2) (2.165e-03) | 7.370e+00(5) (1.471e-02) | 7.420e+00(1) (3.813e-04) | 7.382e+00(3) (1.953e-01) | 6.960e+00(8) (9.104e-01) | 7.288e+00(7) (3.457e-01) | 7.307e+00(6) (6.246e-01) | 7.375e+00(4) (1.839e-01) |
| | 5 | 3.163e+01(4#) (2.249e-02) | - | 3.164e+01(3#) (3.938e-03) | 3.165e+01(1) (1.195e-02) | 3.161e+01(5) (1.663e-01) | 3.158e+01(6#) (2.018e-01) | 3.152e+01(7#) (6.123e-02) | 3.165e+01(2#) (1.147e-02) |
| | 7 | 1.277e+02 (6#) (1.134e-02) | - | 1.277e+02 (5#) (4.751e-03) | 1.277e+02 (4#) (1.006e-02) | 1.278e+02 (1) (4.495e-04) | 1.278e+02 (2#) (2.997e-03) | 1.268e+02 (7#) (4.048e-01) | 1.278e+02 (3#) (5.284e-03) |
| DTLZ7 | 2 | 1.773e+01(1) (9.318e-06) | 1.770e+01(5#) (7.212e-02) | 1.772e+01(2#) (3.993e-04) | 1.771e+01(4) (2.985e-02) | 1.766e+01(8) (1.414e-01) | 1.771e+01(3) (1.479e-02) | 1.768e+01(7) (1.203e-01) | 1.769e+01(6) (1.060e-01) |
| | 3 | 1.637e+01(1) (1.761e-03) | 1.624e+01(7#) (3.756e-02) | 1.624e+01(8#) (7.939e-03) | 1.637e+01(2) (1.179e-02) | 1.629e+01(6) (1.282e-01) | 1.631e+01(5) (7.380e-02) | 1.633e+01(4) (1.020e-01) | 1.636e+01(3) (5.546e-02) |
| | 5 | 1.290e+01(3#) (2.086e-02) | - | 1.177e+01(7#) (8.988e-02) | 1.299e+01(2) (3.210e-02) | 1.279e+01(4) (2.556e-02) | 1.279e+01(5#) (2.630e-01) | 1.274e+01(6#) (9.724e-02) | 1.305e+01(1) (3.280e-02) |
| | 7 | 9.180e+00 (2#) (2.972e-02) | - | 7.934e+00 (7#) (1.534e-01) | 9.055e+00 (3) (1.926e-01) | 8.847e+00 (5) (4.012e-02) | 8.947e+00 (4#) (3.131e-01) | 8.771e+00 (6#) (4.031e-01) | 9.423e+00 (1) (7.067e-02) |
| DTLZ5$^{-1}$ | 2 | 1.758e+01(3#) (2.108e-04) | 1.755e+01(6#) (1.113e-02) | 1.754e+01(8#) (2.610e-02) | 1.758e+01(4) (4.175e-04) | 1.757e+01(5) (5.013e-05) | 1.755e+01(7#) (1.259e-02) | 1.758e+01(1) (3.801e-04) | 1.758e+01(2) (5.557e-04) |
| | 3 | 5.888e+01(1) (3.682e-02) | 5.637e+01(7#) (4.488e-01) | 5.635e+01(8#) (3.805e-01) | 5.886e+01(2) (4.535e-02) | 5.816e+01(5) (3.628e-02) | 5.703e+01(6) (3.938e-01) | 5.869e+01(3) (8.140e-02) | 5.859e+01(4) (7.385e-02) |
| | 5 | 4.033e+02(3#) (7.404e-01) | - | 2.929e+02(7#) (4.177e+00) | 4.135e+02(1) (1.278e+00) | 3.889e+02(5) (4.687e+00) | 3.826e+02(6#) (5.860e+00) | 4.040e+02(2#) (2.210e+00) | 4.020e+02(4#) (2.224e+00) |
| | 7 | 1.788e+03 (5#) (6.238e+00) | - | 1.191e+03 (7#) (2.625e+01) | 2.031e+03 (1) (1.112e+01) | 1.783e+03 (6) (2.315e+01) | 1.978e+03 (3#) (2.773e+01) | 1.970e+03 (4#) (2.638e+01) | 1.981e+03 (2#) (1.663e+01) |
| WFG4 | 2 | 8.524e+00(6#) (2.342e-02) | 8.071e+00(8#) (6.935e-02) | 8.114e+00(7#) (2.326e-02) | 8.571e+00(5) (3.198e-02) | 8.583e+00(4) (3.405e-02) | 8.638e+00(2#) (1.965e-02) | 8.649e+00(1) (1.148e-02) | 8.614e+00(3) (2.573e-02) |
| | 3 | 7.533e+01(4#) (2.368e-01) | 6.595e+01(8#) (7.651e-01) | 6.942e+01(7#) (2.834e-01) | 7.508e+01(6) (2.495e-01) | 7.517e+01(5) (2.184e-01) | 7.612e+01(1) (1.738e-01) | 7.589e+01(3) (1.925e-01) | 7.598e+01(2) (1.647e-01) |
| | 5 | 8.915e+03(1) (3.170e+01) | - | 7.829e+03(7#) (1.599e+02) | 8.240e+03(6) (6.735e+01) | 8.536e+03(4) (4.927e+01) | 8.656e+03(2#) (5.303e+01) | 8.288e+03(5) (5.584e+01) | 8.593e+03(3) (3.763e+01) |
| | 7 | 1.838e+06 (1) (9.627e+03) | - | 1.588e+06 (6#) (2.822e+04) | 1.496e+06 (7) (3.291e+04) | 1.704e+06 (4) (1.347e+04) | 1.739e+06 (2) (1.287e+04) | 1.592e+06 (5) (1.822e+04) | 1.726e+06 (3) (1.149e+04) |
| WFG8 | 2 | 7.489e+00(5#) (4.485e-02) | 7.452e+00(8#) (1.650e-01) | 7.480e+00(7#) (2.399e-02) | 7.652e+00(2) (3.484e-02) | 7.540e+00(4) (4.556e-02) | 7.624e+00(3#) (2.706e-02) | 7.664e+00(1) (3.290e-02) | 7.483e+00(6) (6.448e-02) |
| | 3 | 6.763e+01(6#) (6.951e-01) | 5.848e+01(8#) (1.092e+00) | 6.471e+01(7#) (5.244e-01) | 6.957e+01(4) (4.102e-01) | 6.944e+01(5) (4.053e-01) | 7.040e+01(1) (1.555e-01) | 6.993e+01(3) (1.895e-01) | 7.010e+01(2) (2.292e-01) |
| | 5 | 7.640e+03(3#) (1.331e+02) | - | 5.268e+03(7#) (3.124e+02) | 7.239e+03(5#) (9.265e+01) | 7.824e+03(1) (4.568e+01) | 7.766e+03(2#) (5.206e+01) | 7.172e+03(6#) (5.764e+01) | 7.558e+03(4#) (7.764e+01) |
| | 7 | 1.532e+06 (3#) (2.850e+04) | - | 9.383e+05 (7#) (5.933e+04) | 1.148e+06 (6#) (4.279e+04) | 1.577e+06 (1) (1.491e+04) | 1.560e+06 (2#) (1.153e+04) | 1.231e+06 (5#) (2.321e+04) | 1.441e+06 (4#) (1.701e+04) |
| WFG7$^{-1}$ | 2 | 2.226e+01(1) (7.493e-04) | 2.218e+01(8#) (1.163e-01) | 2.223e+01(6#) (2.277e-02) | 2.225e+01(3) (5.147e-03) | 2.224e+01(4) (1.979e-03) | 2.220e+01(7) (1.877e-02) | 2.225e+01(2) (7.343e-03) | 2.224e+01(5) (1.325e-02) |
| | 3 | 1.441e+02(2#) (3.066e-01) | 1.364e+02(8#) (1.170e+00) | 1.384e+02(7#) (1.204e+00) | 1.441e+02(3) (2.361e-01) | 1.430e+02(5) (2.325e-01) | 1.409e+02(6#) (9.857e-01) | 1.444e+02(1) (1.135e-01) | 1.440e+02(4) (2.331e-01) |
| | 5 | 5.917e+03(6#) (4.959e+01) | - | 3.748e+03(7#) (6.122e+01) | 6.005e+03(5) (7.140e+01) | 6.055e+03(3) (5.859e+01) | 6.039e+03(4#) (2.122e+02) | 6.357e+03(2#) (3.209e+01) | 6.359e+03(1) (5.074e+01) |
| | 7 | 1.946e+05 (6#) (2.881e+03) | - | 1.512e+05 (7#) (3.547e+03) | 2.196e+05 (5) (5.651e+03) | 2.488e+05 (4) (7.363e+03) | 2.847e+05 (3#) (5.079e+03) | 2.888e+05 (2#) (3.445e+03) | 2.958e+05 (1) (4.001e+03) |

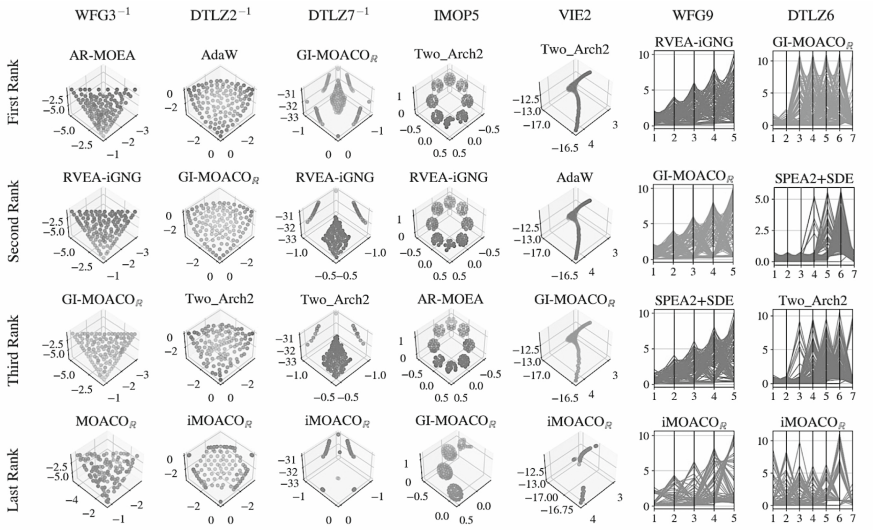



**Fig. 2.** PFAs generated by GI-MOACO$_\mathbb{R}$ and the selected MOACOs and MOEAs.

Figure 3 shows a set of heat maps comparing GI-MOACO$_\mathbb{R}$ with the selected MOACOs and MOEAs. The heat maps show the number of times an algorithm is ranked first or second in the comparison based on HV, IGD$^+$, and $E_s$, based on Tables SM-1 to SM-18. On the one hand, Fig. 3(a) shows that the PFAs generated by GI-MOACO$_\mathbb{R}$ are the best compared to MOACO$_\mathbb{R}$ and iMOACO$_\mathbb{R}$ according to HV, IGD$^+$, and E$_s$ in most of the adopted MOPs and MaOPs. GI-MOACO$_\mathbb{R}$ obtains the first rank in 80.89% of the MOPs and 89.58% of the MaOPs[9]. Additionally, Fig. 2 shows several irregular MOPs such as WFG3$^-$1 DTLZ2$^-$1, DTLZ7$^-$1, and VIE2, where GI-MOACO$_\mathbb{R}$ outperformed MOACO$_\mathbb{R}$ and iMOACO$_\mathbb{R}$, according to HV. For DTLZ2$^-$1, DTLZ7$^-$1, and VIE2, iMOACO$_\mathbb{R}$ shows difficulties to converge. These results indicate that GI-MOACO$_\mathbb{R}$ outperforms MOACO$_\mathbb{R}$ and iMOACO$_\mathbb{R}$.

### 5.3 Comparison Against State-of-the-Art MOEAs

The goals of comparing GI-MOACO$_\mathbb{R}$ with state-of-the-art MOEAs are threefold: 1) determining if GI-MOACO$_\mathbb{R}$ is competitive against MOEAs designed to tackle regular and irregular PF geometries, 2) identifying if GI-MOACO$_\mathbb{R}$ outperforms these MOEAs in MOPs with specific properties, and 3) pointing out improvement areas for GI-MOACO$_\mathbb{R}$.

Table 2 and Fig. 2 show that GI-MOACO$_\mathbb{R}$ is competitive with the selected MOEAs in multiple irregular MOPs such as DTLZ2$^{-1}$, DTLZ7$^{-1}$, VIE3, WFG7$^{-1}$ for 4 objectives, DTLZ6 for 7 objectives, and DTLZ7 regardless of

[9] Note that MOACO$_\mathbb{R}$ does not take part of the comparison of MaOPs since it does not properly scale.

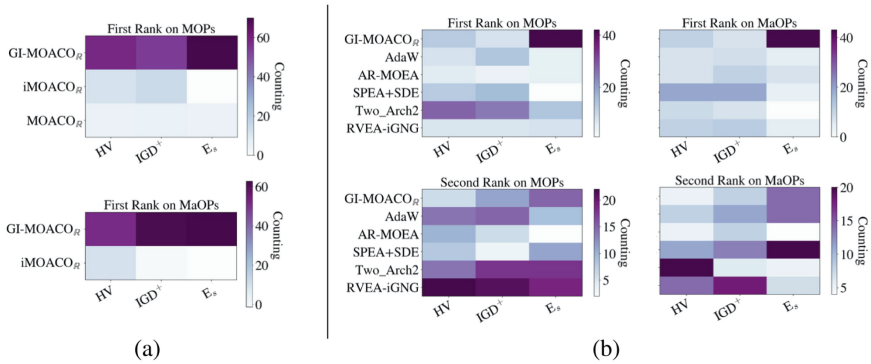



**Fig. 3.** (a) Heat map that shows the number of times a MOACO was ranked first according to HV, IGD$^+$, and E$_s$. (b) Heat map that shows the number of times GI-MOACO$_{\mathbb{R}}$ or a MOEA was ranked first or second according to the three selected QIs.

the number of objectives, according to HV. Additionally, Tables SM-5 and SM-11 indicate that GI-MOACO$_{\mathbb{R}}$ obtains competitive results according to HV and IGD$^+$ in WFG2, WFG6, WFG7, and WFG9 for 5 and 7 objectives. These MaOPs are non-separable and simplex-like. These results show that GI-MOACO$_{\mathbb{R}}$ does not have a preference for specific PF geometries. However, it has a good performance on disconnected, non-separable, and deceptive problems. We believe that this behavior is due to the probabilistic mechanisms of ACO$_{\mathbb{R}}$. In contrast to MOEAs where crossover might not easily generate offspring that jump across disconnected regions, or in deceptive landscapes, ACO$_{\mathbb{R}}$ is less constrained by the need for solutions to be adjacent or directly related in the search space.

Figure 3(b) compares GI-MOACO$_{\mathbb{R}}$ with selected MOEAs. GI-MOACO$\mathbb{R}$ shows superior PFA diversity (in terms of $E_s$) due to its use of PPFs. Two_Arch2 provides the best approximations for most MOPs, and SPEA+SDE excels in most MaOPs according to HV and IGD$^+$. Despite some challenges in covering the entire PF in the IMOP test suite, GI-MOACO$_{\mathbb{R}}$ achieves top ranks in many MOPs and MaOPs, demonstrating its competitiveness. The issue likely stems from the sensitivity of ACO$_{\mathbb{R}}$ to the pheromone matrix composition, suggesting room for improving its probabilistic sampling and pheromone updating mechanisms.

## 6 Conclusions and Future Work

In this paper, we proposed GI-MOACO$_{\mathbb{R}}$ which is a Pareto-Front-Shape Invariant MOACO for solving continuous MOPs. GI-MOACO$_{\mathbb{R}}$ uses an enhanced ACO$_{\mathbb{R}}$ to improve its search capabilities and it has a diversity-preserving mechanism based on PPFs. According to our experimental results using several benchmark

problems with regular and irregular Pareto Front geometries, it outperforms both $\text{MOACO}_{\mathbb{R}}$ and $\text{iMOACO}_{\mathbb{R}}$ and it is competitive concerning state-of-the-art MOEAs. It also showed significant performance in disconnected, non-separable, and deceptive MOPs. For future work, we want to develop a hybrid model and test PDFs to model more difficult and large-scale search spaces.

## References


1. Brockhoff, D., Wagner, T., Trautmann, H.: On the properties of the $R2$ indicator. In: 2012 Genetic and Evolutionary Computation Conference (GECCO'2012), pp. 465–472. ACM Press, Philadelphia, USA (2012). iSBN: 978-1-4503-1177-9
2. Coello Coello, C.A., Lamont, G.B., Veldhuizen, D.A.V.: Evolutionary algorithms for solving multi-objective problems (Genetic and Evolutionary Computation). Springer-Verlag, Berlin, Heidelberg (2006). https://doi.org/10.1007/978-0-387-36797-2
3. Deb, K., Pratap, A., Agarwal, S., Meyarivan, T.: A fast and elitist Multiobjective genetic algorithm: NSGA-II. IEEE Trans. Evol. Comput. **6**(2), 182–197 (2002)
4. Deb, K., Thiele, L., Laumanns, M., Zitzler, E.: Scalable test problems for evolutionary multiobjective optimization. In: Abraham, A., Jain, L., Goldberg, R. (eds.) Evolutionary Multiobjective Optimization. Advanced Information and Knowledge Processing. Springer, London (2005). https://doi.org/10.1007/1-84628-137-7_6
5. Falcón-Cardona, J.G.: New Findings on Indicator-based Multi-Objective Evolutionary Algorithms. Ph.D. thesis, Centro de Investigación y de Estudios Avanzados del Instituto Politécnico Nacional, Mexico, Mexico City (2020)
6. Falcón-Cardona, J., Covantes Osuna, E., Coello, C., Ishibuchi, H.: On the utilization of pair-potential energy functions in multi-objective optimization. Swarm Evol. Comput. **79**, 101308 (2023). https://doi.org/10.1016/j.swevo.2023.101308
7. Falcón-Cardona, J.G., Coello Coello, C.A.: A new indicator-based many-objective ant colony optimization optimizer for continuous search spaces. Swarm Intell. **11**, 71–100 (2017). https://doi.org/10.1007/s11721-017-0133-x
8. Falcón-Cardona, J.G., Covantes Osuna, E., Coello Coello, C.A.: An overview of pair-potential functions for multi-objective optimization. In: Ishibuchi, H., et al. (eds.) EMO 2021. LNCS, vol. 12654, pp. 401–412. Springer, Cham (2021). https://doi.org/10.1007/978-3-030-72062-9_32
9. Falcón-Cardona, J.G., Emmerich, M.T., Coello Coello, C.A.: On the cooperation of multiple indicator-based multi-objective evolutionary algorithms. In: 2019 IEEE Congress on Evolutionary Computation (CEC), pp. 2050–2057 (2019). https://doi.org/10.1109/CEC.2019.8790315
10. Falcón-Cardona, J.G., Ishibuchi, H., Coello, C.A.C.: Exploiting the Trade-off between Convergence and Diversity Indicators. In: 2020 IEEE Symposium Series on Computational Intelligence (SSCI), pp. 141–148 (2020). https://doi.org/10.1109/SSCI47803.2020.9308469
11. Falcón-Cardona, J.G., Ishibuchi, H., Coello Coello, C.A., Emmerich, M.: On the effect of the cooperation of indicator-based multi-objective evolutionary algorithms. IEEE Trans. Evol. Comput. **25**(4), 681–695 (2021). https://doi.org/10.1109/TEVC.2021.3061545
12. Multi-objective ant colony optimization: an updated review of approaches and applications. In: Dehuri, S., Chen, YW. (eds.) Advances in Machine Learning for Big Data Analysis. Intelligent Systems Reference Library, vol. 218. Springer, Singapore (2022). https://doi.org/10.1007/978-981-16-8930-7_1

13. García Nájera, A., Bullinaria, J.: Extending ACO$_{\mathbb{R}}$ to solve multi-objective problems. In: Proceedings of the 2007 UK Workshop on Computation Intelligence (UKCI) (2007)
14. Hernández Gómez, R., Coello Coello, C.A.: A hyper-heuristic of scalarizing functions. In: Proceedings of the Genetic and Evolutionary Computation Conference, pp. 577–584. GECCO'17, Association for Computing Machinery (2017). https://doi.org/10.1145/3071178.3071220
15. Huband, S., Hingston, P., Barone, L., While, L.: A review of multiobjective test problems and a scalable test problem toolkit. IEEE Trans. Evol. Comput. **10**(5), 477–506 (2006). https://doi.org/10.1109/TEVC.2005.861417
16. Ishibuchi, H., Setoguchi, Y., Masuda, H., Nojima, Y.: Performance of decomposition-based many-objective algorithms strongly depends on pareto front shapes. IEEE Trans. Evol. Comput. **21**(2), 169–190 (2017)
17. Li, M., Yang, S., Liu, X.: Shift-based density estimation for pareto-based algorithms in many-objective optimization. IEEE Trans. Evol. Comput. **18**(3), 348–365 (2014). https://doi.org/10.1109/TEVC.2013.2262178
18. Li, M., Yao, X.: Quality evaluation of solution sets in multiobjective optimisation: a survey. ACM Comput. Surv. **52**(2), 26:1–26:38 (2019)
19. Li, M., Yao, X.: What weights work for you? Adapting weights for any pareto front shape in decomposition-based evolutionary multiobjective optimisation. Evol. Comput. **28**(2), 227–253 (2020). https://doi.org/10.1162/evco_a_00269
20. Liu, Q., Jin, Y., Heiderich, M., Rodemann, T., Yu, G.: An adaptive reference vector-guided evolutionary algorithm using growing neural gas for many-objective optimization of irregular problems. IEEE Trans. Cybern. **52**(5), 2698–2711 (2022). https://doi.org/10.1109/TCYB.2020.3020630
21. Márquez-Vega, L.A., Falcón-Cardona, J.G., Covantes Osuna, E.: On the adaptation of reference sets using niching and pair-potential energy functions for multi-objective optimization. Swarm Evol. Comput. **83**, 101408 (2023). https://doi.org/10.1016/j.swevo.2023.101408
22. Miettinen, K.: Nonlinear Multiobjective Optimization. Springer New York, NY (1998). https://doi.org/10.1007/978-1-4615-5563-6
23. Socha, K., Dorigo, M.: Ant colony optimization for continuous domains. Eur. J. Oper. Res. **185**(3), 1155–1173 (2008). https://doi.org/10.1016/j.ejor.2006.06.046
24. Tian, Y., Cheng, R., Zhang, X., Cheng, F., Jin, Y.: An indicator-based multiobjective evolutionary algorithm with reference point adaptation for better versatility. IEEE Trans. Evol. Comput. **22**(4), 609–622 (2018). https://doi.org/10.1109/TEVC.2017.2749619
25. Tian, Y., Cheng, R., Zhang, X., Jin, Y.: PlatEMO: a MATLAB platform for evolutionary multi-objective optimization [educational forum]. IEEE Comput. Intell. Mag. **12**(4), 73–87 (2017). https://doi.org/10.1109/MCI.2017.2742868
26. Tian, Y., Cheng, R., Zhang, X., Li, M., Jin, Y.: Diversity assessment of multi-objective evolutionary algorithms: performance metric and benchmark problems [research frontier]. IEEE Comput. Intell. Mag. **14**(3), 61–74 (2019). https://doi.org/10.1109/MCI.2019.2919398
27. Veldhuizen, D.A.V.: Multiobjective Evolutionary Algorithms: Classifications, Analyses, and New Innovations. Ph.D. thesis, Department of Electrical and Computer Engineering. Graduate School of Engineering. Air Force Institute of Technology, Wright-Patterson AFB, Ohio, USA (1999)
28. Wang, H., Jiao, L., Yao, X.: Two_Arch2: an improved two-archive algorithm for many-objective optimization. IEEE Trans. Evol. Comput. **19**(4), 524–541 (2015). https://doi.org/10.1109/TEVC.2014.2350987

# Scalable Quantum Approximate Optimiser for Pseudo-Boolean Multi-objective Optimisation

Zakaria Abdelmoiz Dahi[1(✉)], Francisco Chicano[2], Gabriel Luque[2], Bilel Derbel[1], and Enrique Alba[2]

[1] Univ. Lille, Inria, CNRS, Centrale Lille, UMR 9189 CRIStAL, F-59000 Lille, France
{abdelmoiz-zakaria.dahi,bilel.derbel}@inria.fr
[2] ITIS Software, University of Malaga, Malaga, Spain
{chicano,gluque,eat}@uma.es

**Abstract.** Quantum computation uses quantum mechanical principles to reach beyond-classical computational power. This has endless applications, especially in optimisation-problems' solving. Most of today's quantum optimisers, more specifically, Quantum Approximate Optimisation Algorithm (QAOA), were originally designed to solve single-objective problems, although real-life scenarios include generally dealing with multiple objectives. Very preliminary literature with design/implementation limitations has been done in this sense. This makes dealing with such limitations and expanding the QAOA applicability to multi-objective optimisation an important step towards advancing quantum computation. To do so, this work presents a decomposition-based Multi-Objective QAOA (MO-QAOA) able to solve multi-objective problems. The proposal's design explores QAOA's features considering the error-prone and limited nature of today's quantum computers as well as the costly quantum simulation. This work's contributions stand in designing both, `(I)` sequential and parallel MO-QAOA, based on `(II)` weighted-sum and Tchebycheff scalarisation, by `(III)` exploring the QAOA's parameters' transference. The validation has been done using 2, 3 and 4-objectives problems of several sizes/complexities/types, using up to 2000 slaves/jobs running quantum computer simulators, as well as three real IBM 127-qubits' quantum computers. The results show up to 89% execution-time decrease, which supports the applicability/reliability of the proposal in today's time-constrained and error-prone quantum computers.

**Keywords:** Quantum Computing · Multi-objective Optimisation

## 1 Introduction

Quantum Computation (QC), based on the principles of quantum physics (e.g. superposition, entanglement, etc.), is able to provide a computational speed-up over classical computing [11]. This has numerous applications, especially


M. Affenzeller et al. (Eds.): PPSN 2024, LNCS 15151, pp. 268–284, 2024.
https://doi.org/10.1007/978-3-031-70085-9_17

in optimisation-problem solving [2]. Regarding the former, two QC paradigms exist: quantum annealing and gate-based quantum computing. This work investigates optimisers designed for discrete gate-based quantum computers, considering their wider applicability, investigation and support by manufacturers. Although quantum technology is gaining more interest, it is still in a Noisy Intermediate Scale Quantum era (NISQ), with limited qubits' number and noise robustness. So, this work investigates a relatively-recent quantum solver; the Quantum Approximate Optimisation Algorithm (QAOA) [9] knowing that it can still provide promising performances regardless of the NISQ nature of the quantum technology, and can, theoretically, guarantee solving-optimality under certain conditions.

QAOA was devised originally to solve single-objective problems, where its performances have been actively investigated. This being said, most real-life problems usually imply considering several conflicting objectives [1]. To this end, rethinking QAOA to be also applicable on such type of problems is crucial towards exploring the advantages of such algorithms, or even quantum computation, in real-world applications. To the best of the authors' knowledge, only three works have investigated Variational Quantum Algorithms (VQAs) in a multi-objective context [3,7,8], including only one researching specifically the QAOA [3]. All the aforementioned works present some design or implementation shortfalls which might be a handicap towards their applicability and efficiency. Considering specifically the work in [3] addressing the QAOA, from a design point of view: `(I)` the decomposition's design might be an issue, `(II)` it does not explore some interesting features of the QAOA such as parameters' distribution concentration and transference that might lead to a more efficient design [13]. Now, from an implementation aspect, the proposal does not consider two very important facts: `(I)` quantum computers are in their NISQ era with quite delaying queuing systems, and `(II)` quantum simulation requirements increase exponentially with respect to the problems' size.

This work proposes both *sequential* and *parallel* approaches to research the limitations in [3], where the *sequential* proposal investigates the design shortfalls by exploring a weighted-sum scalarisation and, for the first time in the literature, Tchebycheff decomposition as well as the QAOA's parameters' concentration/transference. The *parallel* approach deals with the implementation issues by leveraging the parallel use of quantum computers/simulators. The validation has been done on large and diverse benchmarks (2-4 min-max objectives, 4-127 variables), using the largest IBM real gate-based quantum computers (127 qubits) and quantum simulators (up to 2000). It is worth stating that our claims are with regard to the VQAs' literature in Multi-Objective Optimisation (MOO). One should note that this work's purpose is not proving quantum advantage neither outperforming the state-of-the-art classical solvers, as this would be hardly achievable considering today's small-scale/NISQ nature of quantum machines. Instead, we aim to `(I)` prepare the foundations to do so when scalable/stable quantum hardware will be available, and `(II)` explore the strength/weaknesses of nowadays' quantum optimisers/computers regarding classical ones.

The rest of the paper presents some background on quantum computing, QAOA, and MOO in Sect. 2. Afterwards, Sect. 3 thoroughly analyses the literature. In Sect. 4, the proposed approach is introduced, while Sect. 5 assesses its efficiency. Finally, Sect. 6 concludes the paper.

## 2 Preliminary Concepts

This section presents fundamentals of QC, QAOA and MOO.

### 2.1 Quantum Computing and Approximate Optimisation

Quantum computing in a discrete gate-based model describes computation as quantum circuits. The former consists in a set of quantum gates acting on quantum bits (or qubits) [11]. Unlike classical bits, which can take either value 0 or 1, the qubits can be in a superposition of both states. Formally, the quantum state $|\psi\rangle = \alpha\,|0\rangle + \beta\,|1\rangle$ of a qubit is the superposition of both basis state $|0\rangle$ and $|1\rangle$, having $\alpha$ and $\beta$ as probability amplitudes. Based on the Born rule, the probability of measuring 0 and 1 is $\mathcal{P}_z(\langle 0|\psi\rangle) = |\alpha|^2$ and $\mathcal{P}_z(\langle 1|\psi\rangle) = |\beta|^2$, respectively, where $\mathcal{P}_z(\langle 1|\psi\rangle) + \mathcal{P}_z(\langle 0|\psi\rangle) = 1$. Considering the quantum gates, they are unitary operators $\mathcal{U}$, with the property $\mathcal{U}\mathcal{U}^\dagger = \mathcal{U}^\dagger\mathcal{U} = \mathcal{I}$, where $\mathcal{U}^\dagger$ is the adjoint of $\mathcal{U}$ (conjugate transpose) and $\mathcal{I}$ is the identity matrix. The state of $|\psi\rangle$ evolves in a complex Hilbert space $\mathcal{H}$ and the quantum state of $n$ qubits $|\psi_*\rangle$ lives in the tensor product of $n$ complex Hilbert spaces $\mathcal{H}^{\otimes n}$.

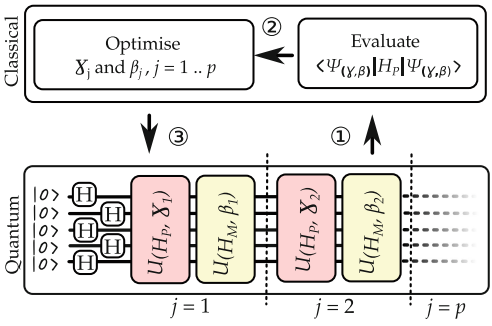



**Fig. 1.** Execution workflow of the QAOA

QAOA is a hybrid quantum algorithm represented by a layered application of unitary transformations called *ansatz*. Concretely, the QAOA's building blocks are the problem unitary and the mixer unitary. The first one is a unitary transformation that depends on the problem Hamiltonian $\mathcal{H}_\mathcal{P}$, while the second depends on a mixer Hamiltonian $\mathcal{H}_\mathcal{M}$. QAOA stands in two components: a quantum and a classical one. The quantum component is an ansatz that produces a state $\left|\psi_{(\gamma,\beta)}\right\rangle$

after applying $p$ pairs of the problem unitary and the mixer unitary applied to $n$ qubits prepared in a superposition state using the Walsh-Hadamard transform $\mathcal{H}^{\otimes n}$ (see Eq. (1)). The $i^{th}$ pair of layers $(\mathcal{H}_\mathcal{P}, \mathcal{H}_\mathcal{M})$ is parameterised with two parameters $\gamma_i$ and $\beta_i$, where $p$ layers use $2p$ parameters. The classical component optimises the sets of parameters $\gamma = \{\gamma_1, \ldots, \gamma_p\}$ and $\beta = \{\beta_1, \ldots, \beta_p\}$ to minimise the expectation value $\mathcal{F}_{(\gamma,\beta)}$ defined in Eq. (2).

$$\mathcal{U}_{(\gamma,\beta)} = \bigotimes_{j=1}^{p} \left(e^{-i\beta_j \mathcal{H}_\mathcal{M}} e^{-i\gamma_j \mathcal{H}_\mathcal{P}}\right) \mathcal{H}^{\otimes n} = \bigotimes_{j=1}^{p} \left(e^{-i\beta_j \sum_{i=1}^{n} \mathcal{X}_i} e^{-i\gamma_j \mathcal{H}_\mathcal{P}}\right) \mathcal{H}^{\otimes n}, \tag{1}$$

$$\mathcal{F}_{(\gamma,\beta)} = \left\langle \psi_{(\gamma,\beta)} \right| \mathcal{H}_P \left| \psi_{(\gamma,\beta)} \right\rangle. \tag{2}$$

### 2.2 Multi-objective and Pseudo-Boolean Optimisation

Solving a multi-objective problem $\mathcal{F}_*$ stands in solving simultaneously a set of $\mathcal{K}$ min/max optimisation problems $\{\mathcal{F}_1, \ldots, \mathcal{F}_\mathcal{K}\}$, where $\mathcal{F}_i : x \to \mathbb{R}$, $i = 1, \ldots, \mathcal{K}$, and $\mathcal{F}_* : x \to \mathbb{R}^\mathcal{K}$. A solution $x_1$ is said to weakly dominate another one $x_2$, denoted $x_1 \preceq x_2$, when $\mathcal{F}_i(x_1) \leq \mathcal{F}_i(x_2)$, for $i = 1, \ldots, \mathcal{K}$. Also, $x_1$ is said to dominate $x_2$, denoted as $x_1 \prec x_2$, when $x_1 \preceq x_2$ and $\exists i \in \{1, \ldots, \mathcal{K}\}$ where $\mathcal{F}_i(x_1) < \mathcal{F}_i(x_2)$. Finally, $x_1$ is said to strictly dominate $x_2$, and denoted $x_1 \lll x_2$, when $\forall i \in \{1, \ldots, \mathcal{K}\}$, $\mathcal{F}_i(x_1) < \mathcal{F}_i(x_2)$ [1]. It is worth noting that normal and strict dominance are more likely to be considered in this work.

Optimisation problems to be solved in MOO can be of different types, although in this present work, a special interest is given to pseudo-Boolean problems. In general, an $n$-dimensional pseudo-Boolean problem can be defined as

$$\mathcal{F}(x) = \sum_{\mathcal{S} \subseteq [n]} \mathcal{C}_\mathcal{S} \prod_{j \in \mathcal{S}} x_j, \tag{3}$$

where we define $[n] = \{1, 2, \ldots, n\}$, $\mathcal{S}$ is a subset of variables from $[n]$, $\mathcal{C}_\mathcal{S}$ is the coefficient of the term involving the product of variables in $\mathcal{S}$ and $x \in \{0, 1\}^n$ is a binary string $x = (x_1, \ldots, x_n)$. In this work, we focus on quadratic and higher-order problems where $2 \leq |\mathcal{S}|$. Quadratic Unconstrained Binary Optimisation (QUBO) can be defined using Eq. (4), where $x$ is an $n$-dimensional column vector, $x^T$ its transpose, and $\mathcal{Q}$ is an $n \times n$ interaction matrix [10].

$$\mathcal{F}(x) = x^T \mathcal{Q} x \tag{4}$$

## 3 Review of Quantum Multi-objective Optimisation

To the best of our knowledge, three works have studied VQAs in the context of MOO [3,7,8], including only one that focused on QAOA. Each of the approaches had shortfalls in some aspects. First, in [7] a Variational Quantum Eigen Solver (VQE) is used to tackle constrained problems as bi-objective problems, where

the problem is the first objective and its constraints are the second one. During optimisation, a non-dominated front is generated, although only solution(s) that satisfy all the constraints and minimise the best energy function are picked. This approach does not guarantee the multi-objective nature (i.e. conflicting) between both objectives. Besides, the multi-objectivity is handled in the algorithm's classical part where NSGA-II [6] optimises the VQE ansatz parameters. Actually, it is not clear which Hamiltonian is being optimised, although as described, the VQE does not handle the MOO problem using its quantum routine.

The authors in [8] devise a VQA that solves a combination of the multi/many-objectives (up to 5) unconstrained minimisation problems. The VQA's parameters optimisation is done with regard to the hypervolume indicator [1] using non-gradient-based methods, `COBYLA` and `FGBSC`. Doing so poses some shortcomings regarding other metrics such as the Inverted Generational Distance and its variant (IGD and IGD$^+$) (i.e. the high computational cost, nadir point selection, etc.) [1]. The unitary transformation describing the algorithm is composed of $\mathcal{L}$ layers. Each layer is composed of $\mathcal{K}$ blocks. Each block combines the mixer and phase unitaries representing one of the $\mathcal{K}$ problems to be solved. The $\mathcal{N}$ most sampled solutions are considered as a non-dominated set. Such design leads to high-depth ansatz, which might make the computation unfeasible knowing the NISQ nature of today's quantum machines. Also, having $\mathcal{K}$ problems will involve optimising $2\mathcal{L}\mathcal{K}$ parameters which hardens the optimisation process compared to a vanilla VQA of $\mathcal{K}$ factor less parameters. In addition, the proposed approach is based on qudits which is more computationally expensive to simulate, unstable and not widely available. Actually, the authors mention the use of qudits, but do not indicate which quantum system/simulator has been used. If any simulation was done, simulating the quantum state of $\mathcal{N}$ qudits will require $d^{\mathcal{N}}$ memory.

The work in [3] proposes a QAOA-based approach to tackle the multi-objective formulation of some small instances of the network routing problem. The proposal solves the weighted aggregation of up-to 4 conflicting QUBOs. This being said, weighted-sum scalarisation might have some shortfalls compared to other scalarisation techniques such as Tchebycheff that finds solutions in non-convex parts of the Pareto front [4]. The authors consider the weighted sum of quadratic polynomials expressed by both the problems and their constraints (as penalty functions). Using the weighted sum to solve problems of this type would require giving a different weight to each QUBO representing both the problem and penalties together. However, the authors dissociated the problems and penalties by assigning them different weights, which impacts differently the penalisation of feasible/unfeasible solutions. Similarly to [8], the authors solve the Hamiltonian representing the weighted sum of objectives and penalties, then extracts a set of non-dominated fronts during sampling. Alternatively, they solve the weighted sum using different weight aggregations, where a non-dominated front is extracted when solving each Hamiltonian. The final front is extracted from the union of all the non-dominated fronts using a non-dominated sorting routine that is quadratic in terms of the number of sampled solutions $\mathcal{N}$ and objectives $\mathcal{M}$ ($\mathcal{O}(\mathcal{M}\mathcal{N}^2)$ [6]). Such complexity increases to $\mathcal{O}(\mathcal{Z}\mathcal{M}\mathcal{N}^2)$ when

having $\mathcal{Z}$ aggregation-weights' combinations. The first approach was mainly researched, while the second has been explored with just 3 weight combinations.

## 4 The Proposed Approach

The proposed sequential approach investigates the design issues in [3,7,8], while its parallelisation researches the implementation limitations. In the following, we will start describing first the sequential approach, then move to the parallel one.

**Algorithm 1.** The Proposed Sequential/Parallel MO-QAOA

**Require:** $\mathcal{Z}$ aggregation-weight configurations
1: **for** $i = 1 \ldots \mathcal{Z}$ **do** ▷ **Parallel Execution in Parallel MO-QAOA**
2: Solve $\mathcal{F}_*$ using $i^{th}$ weight configuration
3: Extract the $i^{th}$ non-dominated front $\mathcal{S}'_i$
4: Set $\gamma$ and $\beta$ values as seed for solving the $(i+1)^{th}$ instance of $\mathcal{F}_*$
5: $\mathcal{S}'_* = \mathcal{S}'_* \cup \mathcal{S}'_i$
6: **end for**
7: Extract the non-dominated front $\mathcal{S}'_*$

### 4.1 Sequential Multi-objective QAOA

This approach goes along with the a *priori* proposal discussed in [3]. In this sense, the QAOA's has some interesting features such as `(I)` its parameters' transference and `(II)` theoretical guarantee of optimality when $p$ tends to $\infty$. So, the core of our proposal relies on keeping the vanilla QAOA unitary transformation $\mathcal{U}_{(\gamma,\beta)}$ (see Eq. (1)) to explore the usefulness of some of the above-mentioned QAOA's features when designing our proposal.

**Weighted-Sum Scalarisation.** We tackle a given multi-objective problem as the aggregation of its multiple objective functions $\{\mathcal{F}_1, \ldots, \mathcal{F}_\mathcal{K}\}$, where $\mathcal{F}_i : x \rightarrow \mathbb{R}$, and $i = 1, \ldots, \mathcal{K}$. This work focuses mainly on QUBOs, although, it is applicable on higher order pseudo-Boolean problems as indicated in Sect. 4.3. Unlike the work done in [3], where the penalty functions are dissociated from the aggregation of the problem (see Eq. (5)), here the penalty functions are considered as part of the QUBO to be solved and, therefore, they will receive the same aggregation coefficient as the original QUBO. This is done because the penalty coefficient is tailored to have the sought impact on how to avoid feasible solutions to be penalised and unfeasible solutions to be kept. The proposed aggregation is done using Eq. (6), where $x$ is the vector of the problem's variables, $\mathcal{Q}$ and $\mathcal{P}$ are the upper triangular matrices of the QUBO's and penalty function coefficients, respectively, which translates to a new QUBO to be solved.

$$\mathcal{F}_* = \sum_{i=1}^{\mathcal{L}} w_i \left(x^T \mathcal{Q}_i x\right) + w_p \left(\sum_{j=1}^{k} x^T \mathcal{P}_j x\right), \quad w_p = 1, \tag{5}$$

$$\mathcal{F}_* = \sum_{i=1}^{\mathcal{L}} w_i \left( x^T \mathcal{Q}_i x + x^T \mathcal{P}_i x \right), \quad \sum_{i=1}^{\mathcal{L}} w_i = 1. \tag{6}$$

The problem defined by Eq. (6) is solved using the vanilla QAOA (see Sect. 2.1), and a set $\mathcal{S}$ of solutions are sampled. A subset $\mathcal{S}' \subset \mathcal{S}$ of non-dominated solutions are extracted in $\mathcal{O}(\mathcal{L}|\mathcal{S}|^2)$. Having $\mathcal{Z}$ aggregation-weight configurations will induce repeating this process $\mathcal{Z}$ times. The final non-dominated front $\mathcal{S}'_*$ is extracted from the union of all $\mathcal{Z}$ non-dominated fronts resulting by solving $\mathcal{F}_*$ using a given weight configuration (see Eq. (7)).

$$\mathcal{S}'_* = \left\{ \mathcal{D} \subset \bigcup_{i=1}^{\mathcal{Z}} \mathcal{S}'_i \;\middle|\; \forall\, s \in \mathcal{D}, \nexists\, s' \prec s, s' \in \bigcup_{i=1}^{\mathcal{Z}} \mathcal{S}'_i \right\}. \tag{7}$$

**Tchebycheff Scalarisation.** We extend our proposal to Tchebycheff decomposition [15] since it is said to be robust when approaching peculiar fronts such as non-convex ones (see Eq. (8)). We propose both an algorithmic and mathematical implementation, which as far as our knowledge, is the first Tchebycheff implementation applicable to QAOA on quantum machines.

$$\min_x \mathcal{F}_*(x|\mathcal{W}, \mathcal{R}^*) = \max_{1 \le i \le \mathcal{L}} \left\{ w_i \left| \left( x^T \mathcal{Q}_i x + x^T \mathcal{P}_i x \right) - r_i^* \right| \right\}, \quad \sum_{i=1}^{\mathcal{L}} w_i = 1. \tag{8}$$

The algorithmic approach computes the `min-max` in $\mathcal{O}(\mathcal{L}^2)$, where $\mathcal{R}^*$ is the chosen reference point (see Algorithm 2). Although this approach can reproduce the `min-max` relation, if the stochastic optimisation process of $\gamma$ and $\beta$ is removed, it is likely to reproduce the same solution to $\left( x^T \mathcal{Q}_i x + x^T \mathcal{P}_i x \right) - r_i^*$, regardless of the aggregation weights (like in randomised algorithms using the same seed).

**Algorithm 2.** Tchebycheff Algorithmic Approach

**Require:** $\mathcal{R}^*$: reference point, $\mathcal{Z}$: aggregation-weight configurations
$\mathcal{S}'_* = \{\}$ ▷ **Non-dominated front**
**for** $i = 1 \ldots \mathcal{Z}$ **do**
  $\mathcal{T}_* = \{\}$ ▷ **Sub-problems' solutions**
  **for** $j = 1 \ldots \mathcal{L}$ **do**
    $t_i =$ **optimise** $\left( (x^T \mathcal{Q}_j x + x^T \mathcal{P}_j x) - r_j^* \right)$
    $\mathcal{T}_* = \mathcal{T}_* \cup \{t_i\}$
  **end for**
  $m_g = \infty$ **and** $t_g = \{\}$
  **for** $k = 1 \ldots \mathcal{L}$ **do**
    $m_d = 0$ **and** $t_d = \{\}$
    **for** $l = 1 \ldots |\mathcal{T}|$ **do**
      $f =$ **evaluate** $w_{i,k} |(x^T \mathcal{Q}_k x + x^T \mathcal{P}_k x) - r_k^*|$ using $t_l$
      **if** $f > m_d$ **then** $m_d = f$ **and** $t_d = t_l$
    **end for**
    **if** $f < m_g$ **then** $m_g = m_d$ **and** $t_g = t_d$
  **end for**
  $\mathcal{S}'_* = \mathcal{S}'_* \cup \{t_g\}$
**end for**

To cope with the drawback of the algorithmic approach, we provide a mathematical approach to the Tchebycheff scalarisation defined by Eqs. (9) and (10). For a detailed justification, one can see theorem and proof in the Appendix [5]. Knowing that $f_i(x)$ is a QUBO, one can rewrite $f_i(x) = \sum_{\mathcal{S}\subseteq[n]} \mathcal{C}_{\mathcal{S}} \prod_{j\in\mathcal{S},b\in\mathcal{E}} x_j^b = \sum_{\mathcal{S}\subseteq[n]} \mathcal{C}_{\mathcal{S}} \prod_{j\in\mathcal{S}} x_j$, $x_j \in \{0,1\}$ and $\mathcal{E}$ is the powers' subset. It can be guaranteed that the Tchebycheff scalarisation will result in a polynomial of degree higher than 2, which can be solved by constructing the corresponding ansatz using Eq. (10) [2], where `CNOT` and $\mathcal{R}_{\mathcal{Z}}$ are the controlled-not and $\mathcal{Z}$ rotation gates, respectively. Although, one should note that in this work we will experimentally explore the algorithmic Tchebycheff approach considering that the present work focuses on QUBO, besides that higher-order polynomials might induce extra transpilation time and simulation resources that might jeopardize comparisons/feasibility (see Sect. 5.2). So, we leave the experimental assessment of the mathematical demonstration for future works especially considering our findings in Sect. 5.2.

$$\min_x \mathcal{F}_*(x|\mathcal{W},\mathcal{R}^*) = \left(\sum_{i=1}^{\mathcal{L}} w_i^p \sum_{j=0}^{p} \binom{p}{j} \left(x^T\mathcal{Q}_i x + x^T\mathcal{P}_i x\right)^j (-r_i^*)^{p-j}\right)^{\frac{1}{p}}, p = 2k, k \in \mathbb{Z}, k \to \infty \quad (9)$$

$$e^{-\gamma \bigotimes_{\mathcal{S}\subseteq[n],j\in\mathcal{S}} \mathcal{Z}_j} = \prod_{i=1}^{|\mathcal{S}|-1} \texttt{CNOT}_{(d_i,d_{i+1})} \mathcal{R}_{\mathcal{Z}_{|\mathcal{S}|}}(2\gamma) \prod_{i=|\mathcal{S}|-1}^{1} \texttt{CNOT}_{(d_i,d_{i+1})}, \mathcal{S} = \{d_1,\ldots,d_{\mathcal{N}}\} \quad (10)$$

**QAOA Parameters' Transference.** The work in [12] proved that for different instances of the `MAX-Cut` problem, optimal QAOA's parameters ($\gamma$ and $\beta$) can be sampled from the same distribution, and another work [13] showed that they can be transferable from one instance to another. Here, the QAOA's parameters transference is explored in its simplest form, by passing the optimised QAOA's parameters values when solving $\mathcal{F}_*$ using the $i^{th}$ weight configuration $i = 1,\ldots,\mathcal{Z}$, as a seed to the optimiser when solving $\mathcal{F}_*$ using the $(i+1)^{th}$ weight configuration. Algorithm 1 depicts the sequential MO-QAOA workflow.

### 4.2 Parallel Multi-objective QAOA

Nowadays quantum computers are in their NISQ era, so most of today's manufacturers provide access to quantum computers for a limited time (usually upon fee), and also long queuing when executing sequential computation. Using the devised approach in [3] or the sequential version devised in this work might pose a bottleneck towards the applicability of the proposed approaches. On the other hand, the same manufacturers that provide limited time of computation, provide access to several machines. So, neglecting to some degree QAOA's transference (Sect. 4.1), one can take advantage of this fact by solving in parallel, the $\mathcal{Z}$ instances of $\mathcal{F}_*$ on $\mathcal{Z}$ quantum computers, and then extracting the global non-dominated front $\mathcal{S}'_*$ on the classical machine. When using Tchebycheff scalarisation, the parallelism is applied for computing $w_i \left|\left(x^T\mathcal{Q}_i x + x^T\mathcal{P}_i x\right) - r_i^*\right|$ (see Pseudo-code of Algorithm 1). This approach turns out to be beneficial also when

using quantum computer simulators considering the availability of several classical machines. It stands in solving $\mathcal{F}_*$ by using a quantum computer simulator on $\mathcal{Z}$ classical machines, where each machine will run a given weight configuration. This can overcome the bottleneck of time-limited QC and take advantage of classical computation to facilitate quantum simulation (see Fig. 2). The communication between the server and quantum simulators/machines is used to dispatch the $\mathcal{Z}$ instances of the weighted problem. So, communication complexity grows linearly $\mathcal{O}(\mathcal{Z})$ considering the number of weights' aggregations.

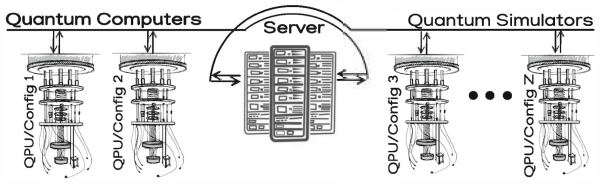



**Fig. 2.** The proposed parallel MO-QAOA

### 4.3 Consequences of the Proposed MO-QAOA Design

Regarding our proposal's design, two consequences must be highlighted: `(I)` it can be used to solve higher-order polynomial problems. Considering the findings in [14], we expect our approach to be more efficient on higher-order polynomials. `(II)` the current design of our approach allows applying other decomposition methods which enables dealing with more complex Pareto fronts.

## 5 Experimental Results and Analysis

The implementation has been done using Python 3.10.12 and bash scripting. The execution has been done on a server using Ubuntu 22.04.3 LTS 64 bits (7 CPUs and 16 GB of RAM) and a cluster with the configuration given in Table 1 using Linux Enterprise Server 15 SP4 15.4 OS. IBM Qiskit version 1.0.2 has been used for quantum execution/simulation. The QAOA has been run using `COBYLA` optimiser and sampled 128 times with a depth $p = \log(\mathcal{N})$ following findings in [12], where $\mathcal{N}$ is the problem's size. Comparisons have been made against the Non-Dominated Sorting Genetic Algorithm (NSGA-II) [6] and the Multi-Objective Evolutionary Algorithm based on Decomposition (MOEA/D) [15]. The former have been run using 100 solutions over 500 iterations ($50,000\mathcal{M}$ fitness evaluations) which is (on average) much larger than the solutions' sampled by the MO-QAOA. The experiments were repeated over 32 execution and comparison metrics such as `Median` and the Median Absolute Deviation (`MAD`) have been considered. Also, null-hypothesis tests with 5% significance have been applied.

The benchmarks have been generated to assess the proposal's scalability when dealing with different complexities in terms of type and number of objectives.

**Table 1.** Hardware components of the cluster used.

| Nodes | CPU / GPU | RAM | InfiniBand | Localscratch |
|---|---|---|---|---|
| 126 × `SD530` | 56 × Intel Xeon Gold 6230R @ 2.10GHz | 200GB | HDR100 | 950 GB |
| 24 × `Bull R282-Z90` | 128 × AMD EPYC 7H12 @ 2.6GHz | 2TB | HDR200 | 3.5TB |
| 168 × `IBM dx360 M4` | 16 × Intel E5-2670 @ 2.6GHz | 32GB | FDR40 | 400GB |
| 4 × `DGX-A100` | 8 × A100 Tensor Core | 1TB | | 14 TB |

Problems of 2-4 objectives, where $\mathcal{Q}_{ij}$, $\mathcal{P}_{ij} \in [-1000, 1000]$, of 4-127 variables have been generated and tackled. Table 2 summarises the benchmarks features including the problems' number, size, type and sparsity percentage (i.e. variables with no interactions). Finally, the benchmarks have been solved using 2-2000 equally-spaced aggregation-weights' configurations. A large literature exists on weight-aggregations selection, however, this is not the focus of the present paper and left to be explored in future works. It is to be indicated also that the size of the benchmarks solved on quantum simulators has been fixed to 29 variables representing more than 536 millions combinations. Implementing the approach for 29 variables would require using/simulating 29 qubits. Knowing that the simulation requirements grows exponentially as the quantum calculation grows, increases the simulation or real execution requirements beyond the ones that are currently available. This being said, this is an effect universal to all quantum calculation and admissible knowing that the goal of our work is a proof of concept rather than an attempt to outperform state-of-the-art classical solvers.

**Table 2.** Benchmark problems used in the experimental evaluation.

| # Objectives | Size | Type | Q | # Aggreg. Config. |
|---|---|---|---|---|
| 2 | 9, 4, 16, 32, 64 127 | {min, min} | {10,20} % | {2,5,11, 44, 100, 500, 1000, 2000} |
| 3 | 19 | {min, max} | {10,20,30} % | {2,5,11, 44, 100, 500, 1000, 2000} |
| 4 | 29 | {max, max} | {10,20,30,40} % | {2,5,11} |

### 5.1 Obtained Results and Discussion

The first set of experiments has been done to assess the sequential approach, especially when using (or not) parameters' transference, while the second set evaluates the parallel approach. The best results are bold/yellow-shaded. Figures 3 and 4 display the `Median` and `MAD` of $\mathcal{F}_*$ during 32 executions, when solving 2 and 3-objectives problems using 2000 aggregation-weights' configurations. It can be observed that the parameters' transference has small impact on the solving efficiency. This turns out to be similar in problems with 2-2000 aggregations. This observation is confirmed in Figs. 5 and 6, where using (or not) parameter transference yields similar approximated Pareto front for bi and three-objective problems. This encourages the use of the parallel approach of the MO-QAOA,

where no parameter transference is possible. One should note that Figs. 3,4, 5 and 6 are obtained without using parameters' transference, but have been used since they are quite representative to both scenarios; whether using or not transference.

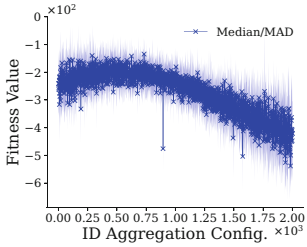



**Fig. 3.** MO-QAOA with/out trans.

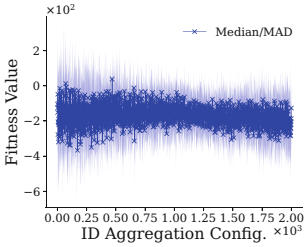



**Fig. 4.** MO-QAOA with/out trans.

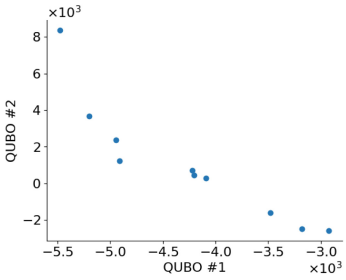



**Fig. 5.** MO-QAOA with/out trans.

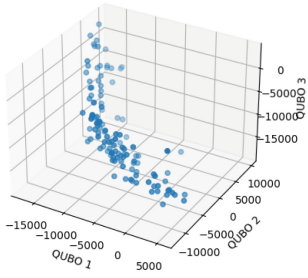



**Fig. 6.** MO-QAOA with/out trans.

Figures 7 and 8 display the Hypervolume (H) and IGD$^+$ values for 3-objectives problems with regard to the number of weight-aggregations considered, where as the number of weight-aggregation configurations increases the Hypervolume/IGD$^+$ value becomes better. This was expected and demonstrates the relevance of our proposal, since it can scale easily when the number of aggregations increases. The chosen figures are of an execution that is representative of the same conclusion for both MO-QAOA with and without parameters' transference.

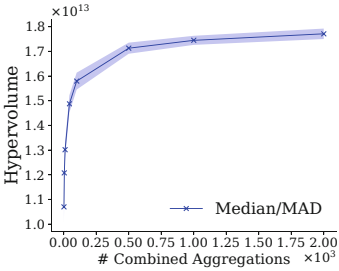



**Fig. 7.** Seq. MO-QAOA without trans.

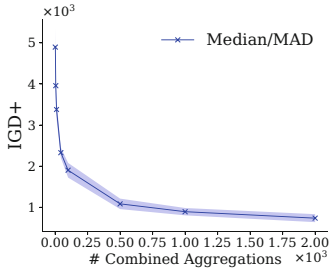



**Fig. 8.** Seq. MO-QAOA without trans.

**Table 3.** Median Hypervolume/IGD$^+$: Seq. MO-QAOA with vs. without transf.

| # $\mathcal{W}$ | Alg. | Par. Transf. | # Obj. | 2 | | | | 3 | | | | 4 | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | H | | IGD+ | | H | | IGD+ | | H | | IGD+ | |
| | | | | Median | MAD | Median | MAD | Median | MAD | Median | MAD | Median | MAD | Median | MAD |
| 2 | WS | Yes | | 97.8E+6 | 1.4E+6 | 198.4E+0 | 98.3E+0 | 10.7E+12 | 445.4E+9 | 4.9E+3 | 308.5E+0 | 1.2E+18 | 47.2E+15 | 12.5E+3 | 549.6E+0 |
| | | No | | 97.1E+6 | 1.7E+6 | 218.5E+0 | 105.6E+0 | 10.7E+12 | 621.5E+9 | 4.9E+3 | 447.0E+0 | 1.2E+18 | 52.4E+15 | 12.6E+3 | 493.6E+0 |
| | Tcheby. | No | | 49.3E+6 | 7.9E+6 | 3.2E+3 | 866.2E+0 | 3.5E+12 | 789.6E+9 | 12.4E+3 | 1.8E+3 | 374.8E+15 | 83.5E+15 | 21.6E+3 | 2.1E+3 |
| 5 | WS | Yes | | 99.6E+6 | 919.1E+3 | 79.9E+0 | 43.1E+0 | 12.0E+12 | 351.0E+9 | 4.2E+3 | 266.0E+0 | 1.4E+18 | 70.8E+15 | 10.9E+3 | 538.8E+0 |
| | | No | | 99.8E+6 | 922.5E+3 | 72.1E+0 | 37.9E+0 | 12.1E+12 | 361.7E+9 | 4.0E+3 | 282.5E+0 | 1.4E+18 | 58.9E+15 | 11.2E+3 | 482.4E+0 |
| | Tcheby. | No | | 58.6E+6 | 6.4E+6 | 2.5E+3 | 467.7E+0 | 5.0E+12 | 752.0E+9 | 10.5E+3 | 1.3E+3 | 465.8E+15 | 144.5E+15 | 20.3E+3 | 2.8E+3 |
| 11 | WS. | Yes | | 100.9E+6 | 182.8E+3 | 11.6E+0 | 11.6E+0 | 13.1E+12 | 340.7E+9 | 3.3E+3 | 251.3E+0 | 1.3E+18 | 10.6E+15 | 10.1E+3 | 190.0E+0 |
| | | No | | 100.9E+6 | 219.5E+3 | 25.1E+0 | 23.6E+0 | 13.0E+12 | 335.3E+9 | 3.4E+3 | 230.1E+0 | 1.2E+18 | 8.6E+15 | 11.3E+3 | 106.1E+0 |
| | Tcheby. | No | | 73.7E+6 | 7.6E+6 | 1.6E+3 | 459.1E+0 | 6.0E+12 | 713.0E+9 | 9.0E+3 | 941.2E+0 | 646.0E+15 | 110.3E+15 | 17.6E+3 | 1.6E+3 |
| SOTA | | NSGA-II | | **101.1E+6** | 0.0E+0 | **0.0E+0** | 0.0E+0 | **18.9E+12** | 56.1E+9 | **404.4E+0** | 28.2E+0 | **3.0E+18** | 67.8E+15 | **3.5E+3** | 308.8E+0 |
| | | MOEAD | | 100.2E+6 | 0.0E+0 | 32.9E+0 | 0.0E+0 | 10.9E+12 | 0.0E+0 | 3.8E+3 | 0.0E+0 | 1.5E+18 | 1.2E+15 | 8.7E+3 | 80.2E+0 |

**Table 4.** Median Hypervolume/IGD$^+$: Seq. MO-QAOA vs. SOTA

| # $\mathcal{W}$ | Alg. | Par. Transf. | # Obj. | 2 | | | | 3 | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | H | | IGD+ | | H | | IGD+ | |
| | | | | Median | MAD | Median | MAD | Median | MAD | Median | MAD |
| 44 | WS. | Yes | | **101.1E+6** | 0.0E+0 | **0.0E+0** | 0.0E+0 | 14.7E+12 | 277.5E+9 | 2.4E+3 | 223.1E+0 |
| | | No | | **101.1E+6** | 0.0E+0 | **0.0E+0** | 0.0E+0 | 14.9E+12 | 348.6E+9 | 2.3E+3 | 117.6E+0 |
| 100 | WS. | Yes | | **101.1E+6** | 0.0E+0 | **0.0E+0** | 0.0E+0 | 15.7E+12 | 286.2E+9 | 1.8E+3 | 120.1E+0 |
| | | No | | **101.1E+6** | 0.0E+0 | **0.0E+0** | 0.0E+0 | 15.8E+12 | 342.6E+9 | 1.9E+3 | 178.5E+0 |
| 500 | WS. | Yes | | **101.1E+6** | 0.0E+0 | **0.0E+0** | 0.0E+0 | 16.9E+12 | 326.8E+9 | 1.1E+3 | 99.1E+0 |
| | | No | | **101.1E+6** | 0.0E+0 | **0.0E+0** | 0.0E+0 | 17.1E+12 | 224.8E+9 | 1.1E+3 | 127.5E+0 |
| 1000 | WS. | Yes | | **101.1E+6** | 0.0E+0 | **0.0E+0** | 0.0E+0 | 17.5E+12 | 256.4E+9 | 867.7E+0 | 97.5E+0 |
| | | No | | **101.1E+6** | 0.0E+0 | **0.0E+0** | 0.0E+0 | 17.4E+12 | 184.0E+9 | 894.3E+0 | 89.3E+0 |
| 2000 | WS. | Yes | | **101.1E+6** | 0.0E+0 | **0.0E+0** | 0.0E+0 | 17.7E+12 | 253.7E+9 | 745.2E+0 | 91.4E+0 |
| | | No | | **101.1E+6** | 0.0E+0 | **0.0E+0** | 0.0E+0 | 17.7E+12 | 211.7E+9 | 736.8E+0 | 97.1E+0 |
| SOTA | | NSGA-II | | **101.1E+6** | 0.0E+0 | **0.0E+0** | 0.0E+0 | **18.9E+12** | 56.1E+9 | **404.4E+0** | 28.2E+0 |
| | | MOEAD | | 100.2E+6 | 0.0E+0 | 32.9E+0 | 0.0E+0 | 10.9E+12 | 0.0E+0 | 3.8E+3 | 0.0E+0 |

Tables 3 and 4 show the results of comparing the MO-QAOA's sequential implementation using the Weighted Sum (WS) as well Tchebycheff (Tcheby.) against the NSGA-II and MOEA/D. It can be noted that `(I)` as the number of aggregations increases the MO-QAOA efficiency becomes better, which is quite expected. `(II)` the Tchebycheff-based MO-QAOA as well as the MOEA/D

approachs are worst than the remaining solvers. This can be explained by the need of appropriate reference-point choice, which has been done *ad hoc* in this work for proof-of-concept purposes. (III) the NSGA-II is performing slightly better than the MO-QAOA. This is not alarming considering that the number of aggregations used in the MO-QAOA can grow, although it already yields very similar results to the NSGA-II. Also, many parameters such as the depth of the ansatz, the classical optimiser have a big impact on the efficiency of the proposal. On overall, we expect that increasing slightly the aggregations' number or ansatz depth will be sufficient to completely outperform the NSGA-II (see Table 4).

**Table 5.** Execution time in seconds: sequential vs. parallel MO-QAOA

| # Obj. | # $\mathcal{W}$ | Sequential | | Parallel Tcheby. | | Time Gain (") | Parallel WS | | Time Gain (") |
|---|---|---|---|---|---|---|---|---|---|
| | | Median | MAD | Median | MAD | % | Median | MAD | % |
| 2 | 2 | 5.4E+0 | 212.8E-3 | 4.8E+0 | 80.6E-3 | 11.91 | 3.2E+0 | 266.2E-3 | **41.32** |
| | 5 | 13.0E+0 | 415.7E-3 | 4.8E+0 | 54.2E-3 | 63.35 | 3.9E+0 | 651.1E-3 | **70.18** |
| | 11 | 29.6E+0 | 745.8E-3 | 5.2E+0 | 80.3E-3 | 82.47 | 5.3E+0 | 682.2E-3 | **82.17** |
| 3 | 2 | 50.4E+0 | 1.8E+0 | 81.9E+0 | 5.3E+0 | -62.33 | 34.7E+0 | 927.1E-3 | **31.20** |
| | 5 | 126.8E+0 | 2.7E+0 | 83.8E+0 | 5.7E+0 | 33.93 | 33.5E+0 | 1.1E+0 | **73.61** |
| | 11 | 282.3E+0 | 5.4E+0 | 84.3E+0 | 1.4E+0 | 70.13 | 28.5E+0 | 1.4E+0 | **89.92** |
| 4 | 2 | 111.9E+3 | 3.4E+3 | 226.5E+3 | 10.4E+3 | -102.41 | 92.5E+3 | 2.9E+3 | **17.34** |
| | 5 | 287.9E+3 | 6.1E+3 | 224.4E+3 | 3.6E+3 | 22.04 | 92.1E+3 | 2.1E+3 | **68.01** |
| | 11 | 507.1E+3 | 8.4E+3 | 224.9E+3 | 2.5E+3 | 55.7E+0 | 64.4E+3 | 4.9E+3 | **87.30** |

For assessing the parallel approach, we use the same experiment configurations (benchmarks, seeding, etc.), so we are not interested in the results knowing they will be similar to the sequential. Instead, we seek time reduction. So, in Table 5, it can be seen that for 2, 3, and 4-objectives problems, the execution time (in seconds) has been divided by 8 in some cases. Also, it can be noted that for each size of the problem, the time complexity remains constant, while the sequential one (similar to the literature) increases linearly in terms of the number of weight aggregations. This also applies to non-dominated front extraction, where each slave machine in the parallel approach performs a $\mathcal{O}(\mathcal{MN}^2)$ complexity routine, while the sequential one $\mathcal{O}(\mathcal{LMN}^2)$. However, one can observe that the Tchebycheff parallel implementation does not yield enhancements regarding the sequential variant. This can be explained by the $\mathcal{O}(\mathcal{M}^2)$ additional comparisons and fitness evaluations to compute the `min-max` in Tchebycheff scalarisation.

### 5.2 Results on Real 127-Qubits IBM Quantum Computers

As a proof of concept, we execute our proposal on the largest real gate-based quantum computers available nowadays. We used three 127-qubits IBM quantum computers: `ibm_brisbane`, `ibm_osaka` and `ibm_kyoto` (see Fig. 9). We have combined three IBM quantum-experience accounts to sum-up 30 min of Quantum Processing Unit (QPU) computation (IBM limits 10 min QPU per account).

Parallel execution has been done using all accounts simultaneously where the *least-busy* machine is taken first. Still, reproducing the experiments we did on quantum simulators goes way beyond 30 min of real QPU. Although, our count-based approach provides never-explored-before ideas to pass-by IBM QPU time limitations. In addition to that, since job execution of IBM quantum computers is subject to a queuing, it is not trivial to provide fair/accurate comparisons with the results obtained on quantum simulators. Indeed, following performing some experiments on April $9^{th}$, we found high sparsity in the queuing time: $1880.77'' \pm 2610.53''$. So, the goal of this Section is to provide a proof of applicability of the proposal and eventually identify its strength/weaknesses on the current quantum computers. So, experimentation has been done only on bi-objective problems with two weight-aggregation sets on problems requiring 4, 16, 32, 64 and 127 qubits. Also, the `COBYLA` optimiser has been limited to only 1 iteration. The reported results are of experiments performed on $11^{th}$-$16^{th}$ of April 2024 at three periods of the day to draw a rough landscape of the possible queuing times: `20:00-22:00`, `3:00-7:00` and `9:00-14:00`.

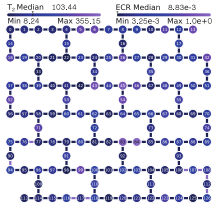



**Fig. 9.** IBM QPU

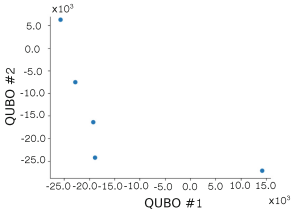



**Fig. 10.** 64 variables

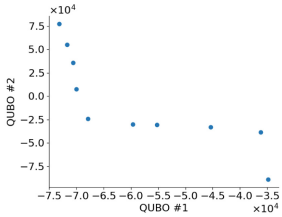



**Fig. 11.** 127 variables

Based on the theoretical findings in [12], $p$ has been set to $\log(\mathcal{N})$. It can be inferred from Table 6 and [12] that the original ansatz depth follows a $\mathcal{O}(\mathcal{N}\log\mathcal{N})$. However, transpilation takes increasingly larger time to produce ansatz that are increasingly more complex. This is a bottleneck considering that it will induce a larger execution time on quantum machines. As a matter of facts, the quantum computers are not able to run a $\log(\mathcal{N})$ layered ansatz for problems requiring 64 and 127 qubits, so $p$ has been set to 1 for those benchmarks. Also, as more complex ansatz requires larger execution time, as in most classical HPC, the probability of hardware error increases with time. Indeed, in our experiments, websocket cancels after machines goes off for daily calibration/maintenance. This happened for instance at `3-5AM` the $9^{th}$ of April 2024. Also, the failure error increases proportionally to the ansatz's depth. When tackling problems requiring 4-32 qubits, the execution success rate is 1. It decreases to $\frac{2}{9}$, when using 64 qubits and falls to $\frac{3}{22}$ when using 127 qubits. Also, on $12^{th}$ April, jobs got stuck/frozen in an endless queuing loop. In addition, to that, it can be noted the sparsity of the queuing time that spans from minutes to several hours.

**Table 6.** Results on real 127-qubits IBM quantum computers

| Size/Metric | Orig. Depth | Transp. Depth | Transp. Time (s) | Queuing Time (s) | Total Ex. Time (s) |
|---|---|---|---|---|---|
| **4 variables** | 15 ± 0 | 603.5 ± 30.5 | 0.1213 ± 0.0083 | 106 ± 5.5 | 762.55 |
| **16 variables** | 87 ± 0 | 14349.5 ± 1254.5 | 6.4142 ± 0.0756 | 144 ± 26.5 | 655.36 |
| **32 variables** | 201 ± 0 | 48787 ± 6977.0 | 38.4706 ± 0.3846 | 3682.5 ± 4672.3 | 14,910.04 |
| **64 variables** | 129 ± 0 | 28384.5 ± 3205.5 | 32.1697 ± 0.6087 | 2978 ± 890.5 | 17,365.12 |
| **127 variables** | 255 ± 0 | 103871.5 ± 580.5 | 187.2764 ± 1.7365 | 5071.5 ± 693.0 | 22,678.94 |

Table 6 presents the `Median` and `MAD` of the original and transpiled ansatz's depth, the transpilation time, queuing time, and the overall execution time, while Figs. 10 and 11 show the obtained non-dominated front for bi-objective problems requiring 64 and 127 qubits. It is worth noting that the queuing time considers both the sampling and expectation value estimation phases. The results in Table 6 and Figs. 10 and 11 prove that our aggregation-based parallel MO-QAOA is feasible on real-quantum computers and optimises their use by leveraging simultaneously all the QPUs. Although, to enhance its practicality, the IBM QPU-time limitation needs to be relaxed. Also, today's IBM transpilation would need to be enhanced to be faster and produce lighter ansatz. In addition, IBM hardware would need to be enhanced to execute more complex ansatz and calibration of machines would need to be less pervasive and avoid cancelling users' jobs. So, to enhance our proposal's practicality on today's quantum computers, we plan as next step to: `(I)` design an ansatz that combines and optimises all the sub-problems simultaneously. `(II)` Design proposals that can perform well with a reduced layering to produce lighter transpiled ansatz.

## 6 Conclusions and Research Perspectives

This work proposes a sequential and a parallel multi-objective QAOA using weighted-sum and Tchebycheff decomposition. The experiments have been done using diverse 2-4 objectives benchmarks of different types and sparsity. Experiments have been made using 2-2000 weight-aggregation configurations running on 2000 slave machines/jobs where an IBM quantum computer simulator as well as three 127-qubits' real quantum computers have been used. The results have shown a decrease up to 89% of the execution time, which is more suitable for today's NISQ quantum machines, as well as taking advantage of the existing classical machines' facilities to optimise quantum simulation. Our proposal turns out to be feasible on real quantum computers, but would be more practical if manufacturers remove QPU time limitations. This being said, as a next step, we plan to `(I)` implement our mathematical Tchebycheff approach, `(II)` test our approach on higher-order polynomials, and `(III)` design an approach that will combine all the problems' objectives into one shallow quantum ansatz.

**Acknowledgments.** This research is partially funded by (I) the PID 2020-116727RB-I00 (HUmove) funded by MCIN/AEI/10.13039/501100011033, (II) TAILOR ICT-48 Network (No 952215) funded by EU Horizon 2020 research and innovation programme, and (III) the Junta de Andalucia (Spain), under contract QUAL21 010UMA. The authors thank the Supercomputing and Bioinnovation Center (SCBI) of the University of Malaga for their provision of computational resources/technical support. Enrique Alba declares that the views expressed are purely those of the writer and may not in any circumstances be regarded as stating an official position of the European Commission.

**Disclosure of Interests.** The authors have no competing interests to declare that are relevant to the content of this article.

## References

1. Audet, C., Bigeon, J., Cartier, D., Le Digabel, S., Salomon, L.: Performance indicators in multiobjective optimization. Eur. J. Oper. Res. **292**(2), 397–422 (2021). https://doi.org/10.1016/j.ejor.2020.11.016
2. Chicano, F., Dahi, Z., Luque, G.: An efficient QAOA via a polynomial QPU-needless approach. In: Proceedings of the Companion Conference on Genetic and Evolutionary Computation, pp. 2187-2194. GECCO '23 Companion, Association for Computing Machinery, New York, NY, USA (2023). https://doi.org/10.1145/3583133.3596409, https://doi.org/10.1145/3583133.3596409
3. Chiew, S.H., et al.: Multi-objective optimization and network routing with near-term quantum computers (2023)
4. Chugh, T.: Scalarizing functions in Bayesian multiobjective optimization. In: 2020 IEEE Congress on Evolutionary Computation (CEC), pp. 1–8 (2020). https://doi.org/10.1109/CEC48606.2020.9185706
5. Dahi, Z.A., Chicano, F., Luque, G., Derbel, B., Alba, E.: Scalable Quantum Approximate Optimiser for Pseudo-Boolean Multi-objective Optimisation (2024). https://doi.org/10.5281/zenodo.12404574
6. Deb, K., Pratap, A., Agarwal, S., Meyarivan, T.: A fast and elitist multiobjective genetic algorithm: NSGA-II. IEEE Trans. Evol. Comput. **6**(2), 182–197 (2002). https://doi.org/10.1109/4235.996017
7. Díez-Valle, P., et al.: Multiobjective variational quantum optimization for constrained problems: an application to cash handling. Quantum Sci. Technol. **8**(4), 045009 (2023). https://doi.org/10.1088/2058-9565/ace474
8. Ekstrom, L., Wang, H., Schmitt, S.: Variational quantum multi-objective optimization (2024)
9. Farhi, E., Goldstone, J., Gutmann, S.: A quantum approximate optimization algorithm (2014)
10. Moraglio, A., Georgescu, S., Sadowski, P.: AutoQubo: data-driven automatic qubo generation. In: Proceedings of the Genetic and Evolutionary Computation Conference Companion, pp. 2232–2239. GECCO '22, Association for Computing Machinery, New York, NY, USA (2022). https://doi.org/10.1145/3520304.3533965, https://doi.org/10.1145/3520304.3533965
11. Nielsen, M.A., Chuang, I.L.: Quantum Computation and Quantum Information. Cambridge University Press (2000)

12. Shaydulin, R., Galda, A.: Error mitigation for deep quantum optimization circuits by leveraging problem symmetries. In: 2021 IEEE International Conference on Quantum Computing and Engineering (QCE). IEEE (Oct 2021). https://doi.org/10.1109/qce52317.2021.00046
13. Shaydulin, R., Lotshaw, P.C., Larson, J., Ostrowski, J., Humble, T.S.: Parameter transfer for quantum approximate optimization of weighted maxcut. ACM Trans. Quant. Comput. **4**(3) (2023). https://doi.org/10.1145/3584706
14. Stein, J., Chamanian, F., Zorn, M., Nüßlein, J., Zielinski, S., Kölle, M., Linnhoff-Popien, C.: Evidence that PUBO outperforms QUBO when solving continuous optimization problems with the QAOA. In: Proceedings of the Companion Conference on Genetic and Evolutionary Computation, pp. 2254–2262. GECCO '23 Companion, Association for Computing Machinery, New York, NY, USA (2023). https://doi.org/10.1145/3583133.3596358
15. Zhang, Q., Li, H.: Moea/d: a multiobjective evolutionary algorithm based on decomposition. IEEE Trans. Evol. Comput. **11**(6), 712–731 (2007). https://doi.org/10.1109/TEVC.2007.892759

# Reliability of Indicator-Based Comparison Results of Evolutionary Multi-objective Algorithms

Lie Meng Pang, Hisao Ishibuchi(✉), Yang Nan, and Cheng Gong

Guangdong Provincial Key Laboratory of Brain-inspired Intelligent Computation, Department of Computer Science and Engineering, Southern University of Science and Technology, Shenzhen 518055, China
{panglm,hisao}@sustech.edu.cn, {12132350,12150059}@mail.sustech.edu.cn

**Abstract.** In evolutionary multi-objective optimization (EMO), performance indicators are often used to measure the quality of non-dominated solution sets obtained by EMO algorithms. However, the reliability of the performance indicators has not been well studied. In this paper, we compare the quality of non-dominated solution sets using four performance indicators: hypervolume (HV), inverted generational distance (IGD), inverted generational distance$^+$ (IGD$^+$), and additive epsilon ($\epsilon_+$). Our experimental results show that different performance indicators produce similar results when they are applied to commonly-used benchmark test problems such as DTLZ1 and DTLZ2. However, for real-world problems, we obtained significantly different comparison results from these indicators. Even when we use the same HV indicator, we obtain significantly different results depending on the reference point specifications. These observations suggest the importance of the choice of an indicator for performance comparison of EMO algorithms on real-world problems. When the HV indicator is used, the choice of a reference point is also important. Moreover, our observations suggest the necessity of using multiple indicators (including the HV indicator with multiple reference points) to obtain reliable performance comparison results.

**Keywords:** Evolutionary multi-objective optimization · performance comparisons · performance indicators · reliability

## 1 Introduction

In many real-world applications, it is often necessary to solve problems with multiple conflicting objectives [16]. These problems, known as multi-objective optimization problems, do not have a single solution that optimizes all objectives simultaneously. Typically, a set of Pareto optimal solutions is obtained to

**Supplementary Information** The online version contains supplementary material available at https://doi.org/10.1007/978-3-031-70085-9_18.


M. Affenzeller et al. (Eds.): PPSN 2024, LNCS 15151, pp. 285–298, 2024.
https://doi.org/10.1007/978-3-031-70085-9_18

represent the trade-offs among the objectives. A Pareto front is formed when all Pareto optimal solutions are projected onto the objective space [2,4].

In the evolutionary multi-objective optimization (EMO) field, a major focus is on developing effective and efficient EMO algorithms that search for a set of non-dominated solutions to approximate the entire Pareto front. EMO algorithms, known for their population-based search mechanism, naturally offer the advantage of obtaining multiple solutions in a single run, thus eliminating the necessity to repeatedly apply an algorithm to obtain such a solution set [2]. Therefore, the goal of EMO algorithms is to obtain a set of non-dominated solutions with good convergence (i.e., close to the Pareto front) and large diversity (i.e., a wide range of solutions covering the entire Pareto front). Over the years, numerous EMO algorithms have been proposed with the aim of achieving this goal.

When a new EMO algorithm is proposed, it is typically evaluated by comparing its performance with other state-of-the-art EMO algorithms. Comparisons are usually conducted on benchmark test problems and/or real-world problems. Then, the performance of each algorithm is evaluated using performance indicators. The results of these evaluations are often presented in comprehensive tables to demonstrate the superior performance of the newly proposed algorithm over the compared EMO algorithms based on some selected performance indicators [10]. Although it is common practice to draw conclusions about the superiority of the proposed EMO algorithm on the basis of numerical indicator values in comparison with other EMO algorithms, the reliability of these values in accurately representing the true performance of each EMO algorithm has not been carefully investigated.

For evaluating the quality of non-dominated solution sets obtained by EMO algorithms, more than 60 performance indicators have been proposed in the literature [1,12]. Among them, the most frequently-used indicators are the hypervolume (HV) [15,25] and inverted generational distance (IGD) [3] indicators. The HV indicator is commonly used because of its Pareto compliant property. However, as the number of objectives increases (e.g., more than 10), the computational time of HV increases exponentially. While IGD is not Pareto compliant, it has been extensively used in many studies. This is primarily due to its computational efficiency, especially in many-objective optimization. A modified version of IGD, i.e., $\text{IGD}^+$ [9], has received increasing attention as a weakly Pareto compliant indicator and has been used in some recent studies for performance comparison of EMO algorithms. The additive epsilon ($\epsilon_+$) indicator [22] is also a weakly Pareto compliant indicator, and it is useful for measuring the convergence of a solution set.

For the purpose of understanding the reliability of performance indicators in the EMO field, in this study, we focus on the four aforementioned indicators: HV, IGD, $\text{IGD}^+$, and $\epsilon_+$. It is critical to know how reliable these indicators are, since they are extensively used in EMO studies for performance comparisons. Using these four indicators, we compare different non-dominated solution sets generated by EMO algorithms. In addition, we use different specifications of

the reference point for the HV indicator in order to understand the effect of its specifications on performance comparison results.

This paper is organized as follows. First, Sect. 2 provides background information on the four performance indicators. Then, Sect. 3 presents experimental results and analysis. Finally, Sect. 4 concludes the paper.

## 2 Background

In general, a multi-objective optimization problem can be formulated as follows:

$$\text{Minimize } (f_1(\mathbf{x}), f_2(\mathbf{x}), ..., f_m(\mathbf{x})), \text{ subject to } \mathbf{x} \subseteq \mathbf{X}, \tag{1}$$

where $m$ is the number of objectives, $\mathbf{x} = (x_1, x_2, ..., x_D)$ is a $D$-dimensional decision vector, $\mathbf{X}$ is the feasible region (search space) of $\mathbf{x}$, and $f_i(\mathbf{x})$ is the $i$-th objective to be minimized ($i = 1, 2, ..., m$).

Let us consider two solutions $\mathbf{a} = (a_1, a_2, ..., a_m)$ and $\mathbf{b} = (b_1, b_2, ..., b_m)$ in the objective space. $\mathbf{a}$ is said to Pareto dominate $\mathbf{b}$ iff $a_i \leq b_i$ for all $i \in \{1, 2, ..., m\}$ and $a_j < b_j$ for at least one $j \in \{1, 2, ..., m\}$. Let $Z$ be a set of solutions with $|Z|$ objective vectors, i.e., $Z = \{\mathbf{z}_1, \mathbf{z}_2, ..., \mathbf{z}_{|Z|}\}$. The set $Z$ is referred to as a non-dominated set when no objective vector in $Z$ is dominated by any other objective vector in $Z$. A performance indicator allows quantitative comparisons by mapping non-dominated sets into real numbers. The following subsections provide a brief explanation of each performance indicator: HV, IGD, IGD$^+$ and $\epsilon_+$.

### 2.1 Hypervolume (HV)

Given a non-dominated solution set $Z \subset \mathbf{R}^m$ and a user-specified reference point $\mathbf{r} = (r_1, r_2, ..., r_m) \in \mathbf{R}^m$, the HV indicator calculates the volume of the region enclosed by the solution set $Z$ and the reference point $\mathbf{r}$. Formally, the HV of $Z$ is defined as follows:

$$\text{HV}(Z, \mathbf{r}) = \mathcal{L}\left(\bigcup_{\mathbf{z} \in Z} [f_1(\mathbf{z}), r_1] \times ... \times [f_m(\mathbf{z}), r_m]\right), \tag{2}$$

where $\mathcal{L}(\cdot)$ is the Lebesgue measure. The larger the HV value, the better the quality of the non-dominated solution set $Z$ in (2).

### 2.2 Inverted Generational Distance (IGD)

For a given non-dominated solution set $Z \subset \mathbf{R}^m$ and a reference point set $Q = \{\mathbf{q}_1, \mathbf{q}_2, ..., \mathbf{q}_{|Q|}\} \subset \mathbf{R}^m$, the IGD indicator calculates the average distance from each reference point to its nearest solution in $Z$ as follows:

$$\text{IGD}(Z) = \frac{1}{|Q|} \sum_{j=1}^{|Q|} \min_{\mathbf{z}_i \in Z} dist(\mathbf{z}_i, \mathbf{q}_j), \tag{3}$$

where $dist(\mathbf{z}_i, \mathbf{q}_j)$ is the Euclidean distance between $\mathbf{z}_i$ and $\mathbf{q}_j$ in the objective space. The smaller the IGD value, the better the quality of the solution set $Z$ in (3).

### 2.3 Inverted Generational Distance$^+$ (IGD$^+$)

The IGD$^+$ indicator is a modified version of the IGD indicator. The distance calculation of IGD$^+$ takes into account the Pareto dominance relation between a reference point and an objective vector. Thus, the IGD$^+$ indicator is weakly Pareto-compliant. As in the IGD indicator, the calculation of the IGD$^+$ value also requires a set of reference points. Given a non-dominated solution set $Z$ and a reference point set $Q$, the IGD$^+$ indicator value is calculated as follows:

$$\text{IGD}^+(Z) = \frac{1}{|Q|} \sum_{j=1}^{|Q|} \min_{\mathbf{z}_i \in Z} dist^+(\mathbf{z}_i, \mathbf{q}_j), \tag{4}$$

where $dist^+(\mathbf{z}_i, \mathbf{q}_j) = \sqrt{\sum_{k=1}^{m} (\max\{z_{ik} - q_{jk}, 0\})^2}$. A solution set with a lower IGD$^+$ value is considered to be of higher quality.

### 2.4 Additive Epsilon ($\epsilon_+$)

Considering a non-dominated solution set $Z \subset \mathbf{R}^m$ and a reference point set $Q \subset \mathbf{R}^m$, the $\epsilon_+$ indicator calculates the minimum shift of $Q$ such that each point in $Q$ is weakly dominated by at least one solution in $Z$. It can be calculated as follows:

$$\epsilon_+(Z) = \max_{\mathbf{q}_j \in Q} \min_{\mathbf{z}_i \in Z} \max_{k \in \{1,\ldots,m\}} (z_{ik} - q_{jk}) \tag{5}$$

The smaller the value of $\epsilon_+$, the better the convergence of a solution set.

## 3 Experiments

To examine the reliability of each performance indicator, we begin by comparing the quality of various non-dominated solution sets using HV, IGD, IGD$^+$ and $\epsilon_+$. For HV, five different reference point specifications are used, i.e., $\mathbf{r} = (r, ..., r)$ where $r = 1.1$, 1.2, 1.5, 100, and 1000 in the normalized objective space with the ideal point $(0, 0, ..., 0)$ and the nadir point $(1, 1, ..., 1)$. Therefore, a total of eight indicators are used in our experiments. Usually, the reference point $\mathbf{r} = (r, ..., r)$ is specified as $1 \leq r \leq 1.5$ in many studies. The other two specifications (i.e., $r = 100$ and 1000) is to examine the effect of inappropriate specifications.

In our experimental setup, we use 50 solution sets, which are the current populations from the 1$^{\text{st}}$ to the 50$^{\text{th}}$ generations of a single run of an EMO algorithm on a test problem. That is, we generate 50 solution sets from a single run of an EMO algorithm with the termination condition of 50 generations. The following three-objective problems are used in our experiments:

– Artificial test problems: DTLZ1, DTLZ2, DTLZ3, and DTLZ4 [7].
– Real world problems: RWA2, RWA3, RWA4, RWA5, RWA6, and RWA7 [20].

We use three-objective optimization problems in our experiments so that we can visually examine the quality of a solution set and its performance indicator values.

Three EMO algorithms are used in this experiment: MOEA/D-PBI (with $\theta = 5$) [21], PREA [18], and NSGA-III [5]. Each algorithm generates 50 solution sets from its single run for each problem. All experiments are conducted on the PlatEMO platform [17]. In order to compute the IGD, IGD$^+$ and $\epsilon_+$ indicator values, a reference point set is required. For the artificial test problems (i.e., DTLZ problems), we use the reference point sets provided in PlatEMO, which contains about 10,000 solutions sampled from the true Pareto front of each problem. This setting is commonly used in many studies. For the real-world problems (i.e., RWA2-7), we use the reference point sets provided by the authors in [20]. The population size is specified as 91 for all three-objective problems. We run 31 independent runs for each EMO algorithm on each problem.

For examining the reliability of each indicator, we rank the 50 solution sets obtained from a single run of each EMO algorithm on each test problem using each indicator. That is, we obtain eight different rankings for the 50 solution sets (i.e., for each run of each algorithm on each test problem). Among the 50 solution sets, a solution set with the best indicator value receives a rank of 1. Conversely, a solution set receives a rank of 50 if it has the worst indicator value among the 50 solution sets. If two solution sets have the same best indicator value, they are each assigned an average rank of 1.5. As a result of this approach, a ranking of the 50 solution sets is obtained for each indicator, which allows a visual comparison of the eight indicators for different solution sets. In this study, the reliability of indicators refers to consistency among the indicators in rankings of 50 solution sets.

Since the average performance of multiple runs is usually used in performance comparison of EMO algorithms in the literature, we also calculate the ranking for the average performance over 31 runs in the following manner. Using 31 runs of each EMO algorithm on each test problem, we first calculate the average indicator value of the 31 solution sets at the 1$^{\text{st}}$ generation for each indicator. Next, we calculate the average indicator value at the 2$^{\text{nd}}$ generation. In this manner, we have 50 average indicator values of each indicator for each EMO algorithm on each test problem. Then, we create the ranking of those 50 average indicator values. This ranking can be viewed as the average ranking of the 50 generations by each indicator for each EMO algorithm on each test problem.

Figure 1 shows the average ranking of the 50 generations by each of the eight performance indicators on the DTLZ1, DTLZ2, and DTLZ3 problems. In each sub-figure of Fig. 1, the X-axis corresponds to the generation ID, which can be also viewed as solution set ID. For example, in the first sub-figure of Fig. 1, the results at ID 30 indicates that they are obtained at the 30$^{\text{th}}$ generation by MOEA/D for the DTLZ1 problem. The Y-axis in each sub-figure represents the rank based on the corresponding indicator. For example, the average performance

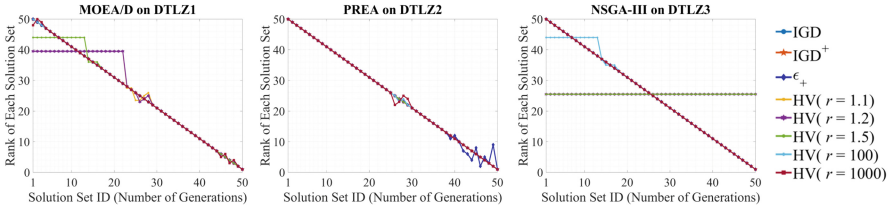



**Fig. 1.** Average ranking of the 50 generations by each performance indicator over 31 independent runs for DTLZ1, DTLZ2 and DTLZ3 problems.

at the 30th generation with ID 30 for MOEA/D on DTLZ1 has a consistent rank of 21 across all performance indicators. It is therefore reasonable to assume that the performance indicator values at the 30th generation are reliable. In fact, it can be observed that (similar) consistent rankings are obtained for the artificial test problems by the eight performance indicators in Fig. 1. When NSGA-III is used on DTLZ3 in the third sub-figure of Fig. 1, all generations have the same rank by the HV indicator with $r = 1.1, 1.2, 1.5$. This is because no solutions dominate the reference point in the first 50 generations. That is, the HV values of all solution sets are zero. When the reference point is unusually large (i.e., $r =$ 100 and 1000), we can see that the performance is improved at every generation. This observation show that the frequently-used reference point specifications are not always appropriate especially when we evaluate the performance of early generations.

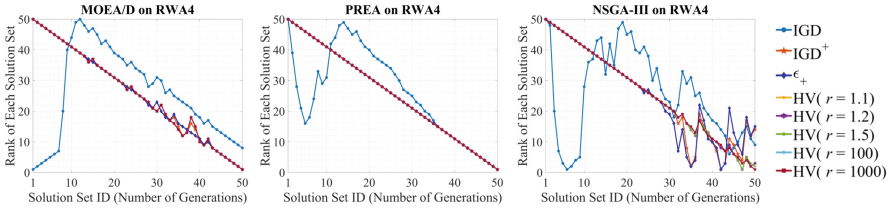



**Fig. 2.** Average ranking of the 50 generations (obtained by MOEA/D, PREA, and NSGA-III) over 31 independent runs by each performance indicator on RWA4 problem.

In contrast, Fig. 2 on RWA4 highlights the inconsistency in the average rankings between the IGD indicator and the other seven performance indicators. Interestingly, in the first sub-figure in Fig. 2 on the average performance of MOEA/D, the best rank is obtained at the $1^{st}$ generation (i.e., ID 1) when we use the IGD indicator. The average IGD performance is quickly deteriorated by MOEA/D from the $1^{st}$ generation to the $12^{th}$ generation. After that, the average IGD performance is gradually improved by MOEA/D. The IGD performance at the $50^{th}$ generation is worse than that in the first seven generations.

Whereas these observations are obtained from the blue results by IGD in the first sub-figure in Fig. 2, it is not likely that the IGD results show the true search behavior of MOEA/D on RWA4 since usually the initial population is gradually improved in the first 50 generations as shown by all the other indicators in the first sub-figure in Fig. 2. Somewhat similar observations are obtained from the other sub-figures in Fig. 2. In the middle sub-figure on the average performance of PREA on RWA4, the average IGD performance starts to deteriorate at the $5^{\text{th}}$ generation, and start to improve at the $14^{\text{th}}$ generation. In the last sub-figure on the average performance of NSGA-III on RWA4, the best average IGD performance is obtained at the $6^{\text{th}}$ generation. The average IGD performance in the last sub-figure shows frequent ups and downs over the 50 generations. In this figure, the average $\text{IGD}^+$ and $\epsilon_+$ performance shows some ups and downs in the last 20 generations, and the average HV performance also show some minor ups and downs. One clear observation in Fig. 2 is that the average IGD performance is totally different from the average performance evaluated by the other indicators.

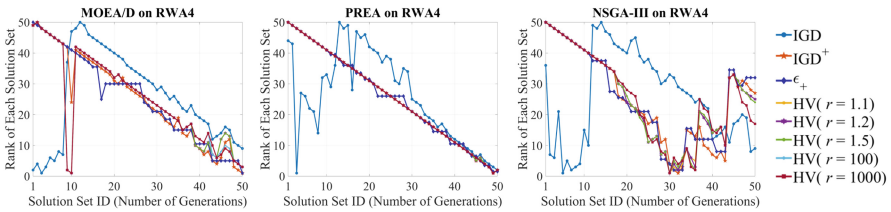



**Fig. 3.** Ranking of 50 solution sets of a single run (obtained by MOEA/D, PREA, and NSGA-III) by each performance indicator on RWA4 problem.

To understand why the average IGD performance shows such a suspicious behavior on the RWA4 problem, we select for each algorithm a single run with the median IGD value at the $50^{\text{th}}$ generation among 31 runs. Figure 3 shows the rank of the 50 solution sets of the median run of each algorithm. In each sub-figure, we can obtain the following observations from the IGD-based ranking results (blue lines):

- MOEA/D: The solution set at the $3^{\text{rd}}$ generation with ID 3 has the best rank. The solution set at the $12^{\text{th}}$ generation with ID 12 has the worst rank.
- PREA: The solution set at the $3^{\text{rd}}$ generation with ID 3 has the best rank. The solution set at the $13^{\text{th}}$ with ID 13 has the worst rank.
- NSGA-III: The solution set with ID at the $5^{\text{th}}$ generation with ID 5 has the best rank. The solution set at the $14^{\text{th}}$ generation with ID 14 has the worst rank.

For analyzing these observations, Figs. 4, 5 and 6 show the solution sets corresponding to the best rank, worst rank, and the final population (Generation 50)

in the objective space for each algorithm. Compared to later generations (e.g., Generation 50), solutions in early generations (e.g., Generation 3 in MOEA/D and PREA, and Generation 5 in NSGA-III) have not yet converged to the Pareto front. Despite this, the IGD indicator tends to assign high evaluations to solution sets in the early generations because their distribution closely resembles the reference point set (i.e., the blue points in each figure). This observation indicates that the IGD indicator may not be reliable. In order to avoid misleading comparison results, multiple indicators should be used in addition to the IGD indicator in performance evaluation.

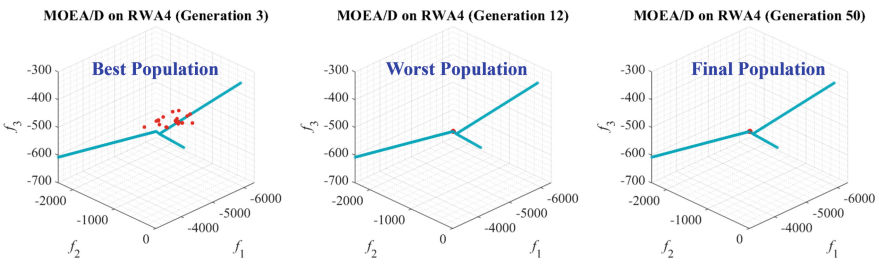



**Fig. 4.** Solution sets corresponding to the best rank, worst rank, and the final population (Generation 50) in the objective space for MOEA/D. The red points are the solution set, and the blue points are the reference point set (i.e., approximated Pareto front). (Color figure online)

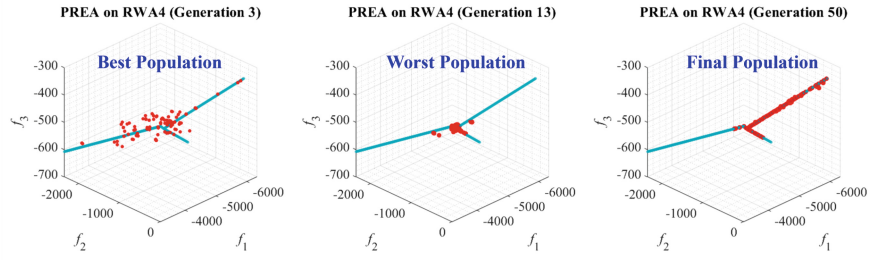



**Fig. 5.** Solution sets corresponding to the best rank, worst rank, and the final population (Generation 50) in the objective space for PREA. The red points are the solution set, and the blue points are the reference point set (i.e., approximated Pareto front). (Color figure online)

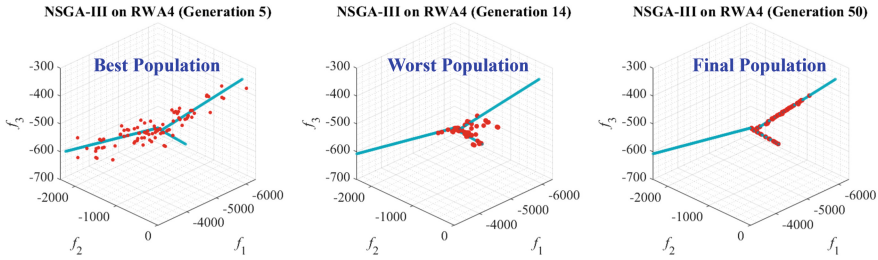



**Fig. 6.** Solution sets corresponding to the best rank, worst rank, and the final population (Generation 50) in the objective space for NSGA-III. The red points are the solution set, and the blue points are the reference point set (i.e., approximated Pareto front). (Color figure online)

Figure 3 shows that the solution sets in early generations have bad evaluations results by all the other indicators. The IGD best solution sets in Figs. 4, 5 and 6 are ranked within the worst five ranks by all the other indicators in Fig. 3. This is because the solution sets in early generations are not close to the Pareto front. As a result, they do not have large HV values. Whereas $IGD^+$ has a very similar formulation to IGD, $IGD^+$ shows much similar behavior to HV than IGD. This is because $IGD^+$ calculates the average distance from the reference point sets to the dominated region by the solution set, which is closely related to the volume of the dominated region by the solution set. In Fig. 3, we can observe similar behaviors of $IGD^+$ and HV independent of the reference point specification. In general, the reference point specification has a large effect on HV-based performance comparison results. This is because boundary solutions have large HV contributions for a large reference point (i.e., far away from the Pareto front) but almost zero contributions for a small reference point (i.e., close to the nadir point). However, since the Pareto front of RWA4 is a combination of lines (i.e., degenerate Pareto front) as shown in Figs. 4, 5 and 6, such a boundary solution effect disappears (since there is no inside solutions on the Pareto front). As a result, almost the same HV-based comparison results are obtained in Fig. 3 independent of the reference point specifications from $r = 1.1$ to $r = 1000$.

In the previous experiments, we examined the reliability of performance indicators for different solution sets obtained by a single EMO algorithm at different generations. In the following experiments, we create the average ranking of ten EMO algorithms based on their final populations using each indicator on each test problem. This experiment is to examine the dependency of the algorithm comparison results on the choice of an indicator. We use ten algorithms in our experiments: SPEA2 [24], NSGA-II [6], IBEA [23], MOEA/D [21], MOEA/D-DE [11], NSGA-III [5], $\theta$-DEA [19], onebyoneEA [13], DEA-GNG [14], and PREA [18]. This set of ten algorithms includes five algorithms proposed in 2001–2010, and five algorithms proposed in 2011–2020. Two termination conditions are con-

sidered in our experiments: 50 generations and 500 generations. All algorithm configurations follow PlatEMO's default specifications. We perform 31 independent runs for each EMO algorithm on each test problem.

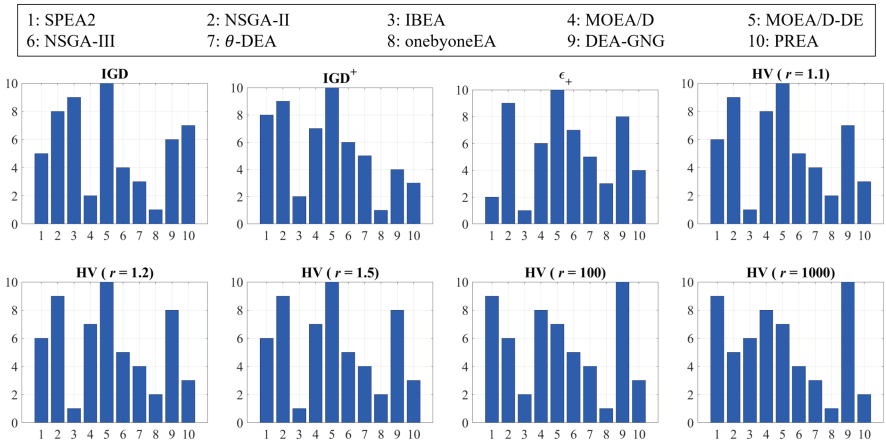



**Fig. 7.** Average ranking of the ten EMO algorithms by each indicator on the DTLZ2 problem under the termination condition of 50 generations.

Figure 7 presents the comparison results for DTLZ2 under the termination condition of 50 generations. The X-axis of Fig. 7 denotes each algorithm ID, e.g., 1 represents SPEA2, 2 represents NSGA-II, and so on. On each sub-figure, the Y-axis represents an algorithm's average rank using the corresponding performance indicator. For example, in the first sub-figure at the upper left, SPEA2 (Algorithm 1) receives an average rank of 5 based on the IGD indicator. The average ranking of the 10 algorithms is based on the average indicator value over 31 runs. It should be noted that all the eight sub-figures in Fig. 7 evaluate exactly the same 31 runs of each EMO algorithm (i.e., exactly the same 31 solution sets obtained by each EMO algorithm).

From Fig. 7, we can observe some similarity from the eight sub-figures whereas they are different. From more careful examination of Fig. 7, we can see that the five sub-figures by $IGD^+$, $\epsilon_+$, and HV with $r = 1.1, 1.2$ and 1.5 are similar. The sub-figure by IGD is different from those figures (e.g., IBEA with ID 3 is rank 9 whereas IBEA is rank 2 by $IGD^+$ and rank 1 by $\epsilon_+$ and HV with $r = 1.1, 1.2, 1.5$). We can also see that a small difference of the reference point specification between $r = 1.1$ and $r = 1.2$ leads to a small difference in the performance comparison results, and a large difference between $r = 1.1$ and $r = 1000$ leads to a large difference in the performance comparison results.

For DTLZ2 (and other DTLZ test problems), the difference of performance comparison results by the eight indicators becomes small (i.e., similar comparison results are obtained from different indicators) when the termination condition is large. For example, Fig. 8 shows the performance comparison results for DTLZ2

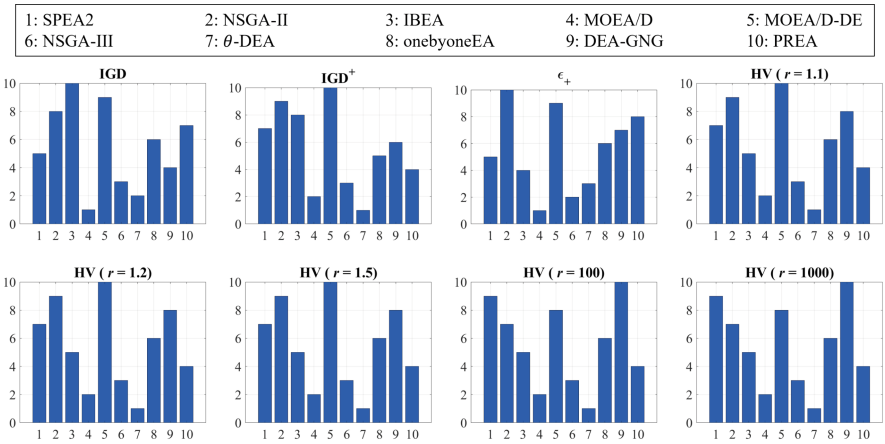



**Fig. 8.** Average ranking of the ten EMO algorithms by each indicator on the DTLZ2 problem under the termination condition of 500 generations.

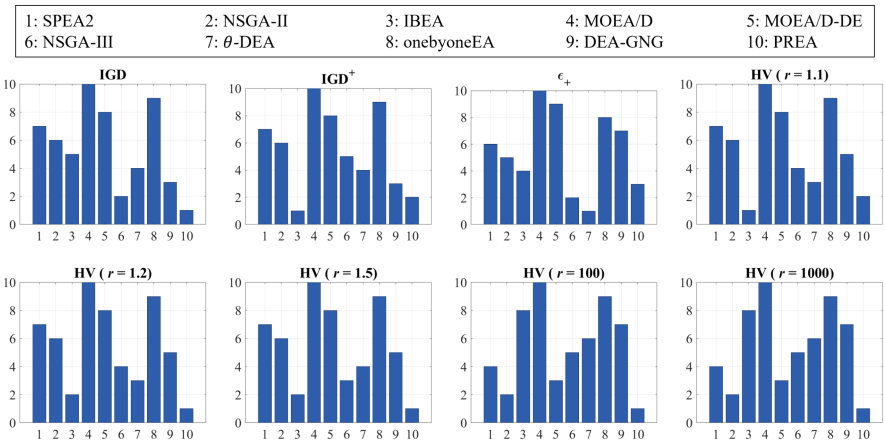



**Fig. 9.** Average ranking of the ten EMO algorithms by each indicator on the RWA3 problem under the termination condition of 500 generations.

under the termination condition of 500 generations. On DTLZ2, all indicators agree that MOEA/D, NSGA-III, and $\theta$-DEA are the top three performing algorithms. However, for real-world problems, different performance indicators generate totally different results. Figure 9 shows the comparison results for RWA3 under the termination condition of 500 generations. It can be seen that IBEA is the best when evaluated by $\text{IGD}^+$ and HV with $r = 1.1$ (the performance comparison results by $\text{IGD}^+$ and HV with $r = 1.1$ are very similar). However, IBEA ranks fourth and fifth when it is evaluated by $\epsilon_+$ and IGD, respectively. Furthermore, with different reference point specifications for HV calculation, the

HV indicator gives totally different results. For example, while IBEA is the best when HV is used with $r = 1.1$, it is the third worst when HV is used with $r = 100$ and $r = 1000$. Overall observations from Fig. 9 are similar to those from Fig. 7: Similar performance comparison results are obtained from $IGD^+$ and HV with $r = 1.1, 1.2, 1.5$. Those similar comparison results are clearly different from IGD-based and $\epsilon_+$-based performance comparison results and HV-based comparison results with $r = 100$ and 1000.

Experimental results on the other test problems are available in a supplementary file (https://github.com/HisaoLabSUSTC/PPSN2024_Performance Indicators). We can obtain similar observations from the results in the supplementary file. Our observations clearly suggest the need of using multiple performance indicators when we evaluate the performance of EMO algorithms. If we use only the IGD indicator for some reasons (e.g., exact HV calculation is unrealistic due to very long computation time), some misleading conclusions can be obtained about the performance of compared EMO algorithms. The use of the IGD indicator alone should be avoided in any case. Since $IGD^+$ show similar performance comparison results with HV-based results with a reasonable specification of the reference point (i.e., $r = 1.1, 1.2, 1.5$), it is advisable to use both IGD and $IGD^+$ when the use of the HV indicator is unrealistic. This suggestion is aligned with the results reported in [8], where HV and $IGD^+$ have similar near-optimal solution distributions. Whereas IGD and $IGD^+$ use exactly the same reference point set for distance calculation, totally different performance comparison results can be obtained as shown in Fig. 9.

## 4 Conclusions

In this paper, we examined the reliability of indicator-based comparison results of EMO algorithms. Experimental results showed that when we used frequently-used test problems such as DTLZ, we obtained very similar performance comparison results from all the examined indicators (e.g., Fig. 8). This means that the choice of an indicator and the implementation of the selected indicator are not very important for such test problems. However, when real-world problems were used, we obtained clearly different comparison results from some indicators (e.g., Fig. 9). Even from the same HV indicator, we obtained clearly different comparison results depending on the specification of the reference point (Fig. 9). This observation suggests the importance of the choice of an indicator and the implementation of the selected indicator. In many cases, IGD-based performance comparison results are clearly different from the other indicator-based comparison results. In almost all cases, similar comparison results are obtained from $IGD^+$ and HV with an appropriate reference point specification. For $\epsilon_+$-based comparison results, there are cases where it performs similarly to IGD, while other cases are more aligned with HV and $IGD^+$. These observations strongly suggest the necessity of using multiple indicators to ensure reliable performance evaluation. When IGD is used, it should be complemented with other (weakly) Pareto compliant indicators such as $IGD^+$ (when the use of HV is very time consuming) and $\epsilon_+$.

More indicators will be used in future work to further examine performance comparison reliability. It should also be noted that for IGD, $IGD^+$, and $\epsilon_+$, the specification of the reference point sets might influence their results. In the future, we will also examine the performance comparison reliability of these indicators with different reference point set specifications. Through these investigations, comprehensive guidelines about the use of performance indicators for evaluating the performance of EMO algorithms will be established. This will ensure high reliability of performance comparison results and preventing creating misleading comparison results.

**Acknowledgments.** This work was supported by National Natural Science Foundation of China (Grant No. 62250710163, 62376115), Guangdong Provincial Key Laboratory (Grant No. 2020B121201001).

**Disclosure of Interests.** The authors have no competing interests to declare that are relevant to the content of this article.

## References

1. Audet, C., Bigeon, J., Cartier, D., Le Digabel, S., Salomon, L.: Performance indicators in multiobjective optimization. Eur. J. Oper. Res. **292**(2), 397–422 (2021)
2. Coello Coello, C.A.: Evolutionary multi-objective optimization: a historical view of the field. IEEE Comput. Intell. Mag. **1**(1), 28–36 (2006)
3. Coello Coello, C.A., Reyes Sierra, M.: A study of the parallelization of a coevolutionary multi-objective evolutionary algorithm. In: Monroy, R., Arroyo-Figueroa, G., Sucar, L.E., Sossa, H. (eds.) MICAI 2004. LNCS (LNAI), vol. 2972, pp. 688–697. Springer, Heidelberg (2004). https://doi.org/10.1007/978-3-540-24694-7_71
4. Deb, K.: Multi-Objective Optimization Using Evolutionary Algorithms. Wiley, New York (2001)
5. Deb, K., Jain, H.: An evolutionary many-objective optimization algorithm using reference-point-based nondominated sorting approach, part I: Solving problems with box constraints. IEEE Trans. Evol. Comput. **18**(4), 577–601 (2014)
6. Deb, K., Pratap, A., Agarwal, S., Meyarivan, T.: A fast and elitist multiobjective genetic algorithm: NSGA-II. IEEE Trans. Evol. Comput. **6**(2), 182–197 (2002)
7. Deb, K., Thiele, L., Laumanns, M., Zitzler, E.: Scalable test problems for evolutionary multiobjective optimization. In: Abraham, A., et al. (eds.) Evolutionary Multiobjective Optimization. Advanced Information and Knowledge Processing, pp. 105–145. Springer, London (2005). https://doi.org/10.1007/1-84628-137-7_6
8. Ishibuchi, H., Imada, R., Masuyama, N., Nojima, Y.: Comparison of Hypervolume, IGD and $IGD^+$ from the viewpoint of optimal distributions of solutions. In: Deb, K., Goodman, E., Coello Coello, C.A., Klamroth, K., Miettinen, K., Mostaghim, S., Reed, P. (eds.) EMO 2019. LNCS, vol. 11411, pp. 332–345. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-12598-1_27
9. Ishibuchi, H., Masuda, H., Tanigaki, Y., Nojima, Y.: Modified distance calculation in generational distance and inverted generational distance. In: Gaspar-Cunha, A., et al. (eds.) Evolutionary Multi-Criterion Optimization. Lecture Notes in Computer Science, vol. 9019, pp. 110–125. Springer, Cham (2015)

10. Ishibuchi, H., Pang, L.M., Shang, K.: Difficulties in fair performance comparison of multiobjective evolutionary algorithms [research frontier]. IEEE Comput. Intell. Mag. **17**(1), 86–101 (2022)
11. Li, H., Zhang, Q.: Multiobjective optimization problems with complicated Pareto sets, MOEA/D and NSGA-II. IEEE Trans. Evol. Comput. **13**(2), 284–302 (2009)
12. Li, M., Yao, X.: Quality evaluation of solution sets in multiobjective optimisation: a survey. ACM Comput. Surv. **52**(2), 1–38 (2019)
13. Liu, Y., Gong, D., Sun, J., Jin, Y.: A many-objective evolutionary algorithm using a one-by-one selection strategy. IEEE Trans. Cybern. **47**(9), 2689–2702 (2017)
14. Liu, Y., Ishibuchi, H., Masuyama, N., Nojima, Y.: Adapting reference vectors and scalarizing functions by growing neural gas to handle irregular Pareto fronts. IEEE Trans. Evol. Comput. **24**(3), 439–453 (2020)
15. Shang, K., Ishibuchi, H., He, L., Pang, L.M.: A survey on the hypervolume indicator in evolutionary multiobjective optimization. IEEE Trans. Evol. Comput. **25**(1), 1–20 (2021)
16. Stewart, T., et al.: Real-world applications of multiobjective optimization. In: Branke, J., et al. (eds.) Multiobjective Optimization. Lecture Notes in Computer Science, vol. 5252, pp. 285–327. Springer, Heidelberg (2008). https://doi.org/10.1007/978-3-540-88908-3_11
17. Tian, Y., Cheng, R., Zhang, X., Jin, Y.: PlatEMO: a matlab platform for evolutionary multi-objective optimization [educational forum]. IEEE Comput. Intell. Mag. **12**(4), 73–87 (2017)
18. Yuan, J., Liu, H.L., Gu, F., Zhang, Q., He, Z.: Investigating the properties of indicators and an evolutionary many-objective algorithm using promising regions. IEEE Trans. Evol. Comput. **25**(1), 75–86 (2021)
19. Yuan, Y., Xu, H., Wang, B., Yao, X.: A new dominance relation-based evolutionary algorithm for many-objective optimization. IEEE Trans. Evol. Comput. **20**(1), 16–37 (2016)
20. Zapotecas-Martínez, S., García-Nájera, A., Menchaca-Méndez, A.: Engineering applications of multi-objective evolutionary algorithms: a test suite of box-constrained real-world problems. Eng. Appl. Artif. Intell. **123**, 106192 (2023)
21. Zhang, Q., Li, H.: MOEA/D: a multiobjective evolutionary algorithm based on decomposition. IEEE Trans. Evol. Comput. **11**(6), 712–731 (2007)
22. Zitzler, E., Thiele, L., Laumanns, M., Fonseca, C., Da Fonseca, V.: Performance assessment of multiobjective optimizers: an analysis and review. IEEE Trans. Evol. Comput. **7**(2), 117–132 (2003)
23. Zitzler, E., Künzli, S.: Indicator-based selection in multiobjective search. In: Yao, X., Burke, E.K., Lozano, J.A., Smith, J., Merelo-Guervós, J.J., Bullinaria, J.A., Rowe, J.E., Tiňo, P., Kabán, A., Schwefel, H.-P. (eds.) PPSN 2004. LNCS, vol. 3242, pp. 832–842. Springer, Heidelberg (2004). https://doi.org/10.1007/978-3-540-30217-9_84
24. Zitzler, E., Laumanns, M., Thiele, L.: SPEA2: improving the strength Pareto evolutionary algorithm. In: Proceedings of the Conference on Evolutionary Methods for Design, Optimization and Control with Applications to Industrial Problems, pp. 95–100 (2001)
25. Zitzler, E., Thiele, L.: Multiobjective optimization using evolutionary algorithms — a comparative case study. In: Eiben, A.E., Bäck, T., Schoenauer, M., Schwefel, H.-P. (eds.) PPSN 1998. LNCS, vol. 1498, pp. 292–301. Springer, Heidelberg (1998). https://doi.org/10.1007/BFb0056872

# Pareto Landscape: Visualising the Landscape of Multi-objective Optimisation Problems

Zimin Liang, Zhiji Cui, and Miqing Li(✉)

University of Birmingham, Birmingham B15 2TT, UK
m.li.8@bham.ac.uk

**Abstract.** Fitness landscape is a valuable framework to understand optimisation problems. In single-objective optimisation, by displaying fitness landscape in a 3D space with the "height" representing the fitness (objective function value) of solutions, one can easily comprehend a variety of problem characteristics (optimality, multi-modality, level of ruggedness, etc.) and spatial features of the search space (basin, ridge, funnel, etc.). However, such straightforward visualisation cannot be directly extended to the multi-objective optimisation case in which a solution corresponds to a vector of values on multiple objective functions. In this paper, we make an attempt to address this issue. Instead of objective function values, we use the Pareto dominance relation to stratify solutions, introducing a method we term *Pareto landscape* for visualising multi-objective problem landscape. We compare Pareto landscape with well-established fitness landscape visualisation methods, including *cost landscape*, *gradient field heatmap* and *PLOT*, and show that Pareto landscape can capture problem characteristics that the other methods cannot do. Lastly, we present the Pareto landscapes of commonly used benchmark problems (ZDT, DTLZ, WFG and BBOB) in the domain, and discuss their features and characteristics.

## 1 Introduction

Fitness landscape plots serve as a valuable framework for understanding optimisation problems. They facilitate an intuitive grasp of complex search spaces (also known as decision spaces) and challenges confronted by search heuristics, inspiring potential improvements and development of new optimisation algorithms.

In the context of multi-objective optimisation, there is a growing interest in developing visualisation tools for presenting fitness landscapes [3,28]. This ranges from simply integrating landscape information (e.g., local Pareto optimal sets) into the objective space [28] to designing networks that graphically represent optimal solutions [21,22] and sets [7], as well as the behaviour of search algorithms [24]. While these visualisation tools effectively capture the structure of (local) optimal solutions/sets and their connectedness, the compression and

Z. Liang and Z. Cui—These authors contributed equally to this work.


M. Affenzeller et al. (Eds.): PPSN 2024, LNCS 15151, pp. 299–315, 2024.
https://doi.org/10.1007/978-3-031-70085-9_19

abstract representation involved make it difficult for people to envision the spatial location of solutions/sets and their associated adjacent structures.

Another approach, which can overcome the above issue, is to directly display the decision space of an optimisation problem with an additional element (e.g., colour) representing solution quality [28]. This encompasses early attempts like cost landscape (aka dominance ratio) [11] and recent efforts such as multi-contour plot [18], gradient field heatmaps [17], line cuts [2,30], local dominance landscape [8], and PLOT [26]. These methods are similar to the fitness landscape plots in single-objective optimisation, where the native decision space is considered, allowing them to present important problem characteristics such as location, shape and size of attraction basins. However, they may not be very accurate in some cases due to the effect of the behaviour of the search algorithm used to locate local optima (e.g., in gradient field heatmaps [17] and PLOT [26]), or the criterion considered to measure the "fitness" of solutions (e.g., in cost landscape [11]), which we will show later on.

In this paper, we attempt to develop a fitness landscape visualisation method for multi-objective optimisation, termed Pareto landscape, that can accurately represent problem characteristics and spatial features. The idea of Pareto landscape is simple. It considers the level of the Pareto non-domination to which a solution belongs when determining its fitness. To evaluate Pareto landscape, we first compare it with its counterpart in single-objective optimisation to identify their similarities and differences. Then, by utilising a class of test functions flexible for embedding rich features [9,15,19], we demonstrate that Pareto landscape can capture various problem characteristics (e.g., multi-modality and neutrality), as well as spatial features of the decision space (e.g., valleys, plateaus, ridges and funnels). Moreover, we compare Pareto landscape with three relevant landscape visualisation methods and discuss their differences. Lastly, we use Pareto landscape to visualise commonly-used multi-objective benchmark functions in the domain, including ZDT [32], DTLZ [6], WFG [13] and BBOB (bbob-biobj) [2], to delineate their features and characteristics, particularly the challenges they may pose for search heuristics.

## 2 Preliminaries

### 2.1 Terminology in Multi-objective Optimisation

For multi-objective optimisation problems (MOPs), without loss of generality, we seek to simultaneously minimise $m$ objective functions $f(x) = (f_1(x), ..., f_m(x))$, where $f(x)$ $(x \in X)$ is a mapping of an $n$-dimensional variable vector to an $m$-dimensional objective vector; namely, $f : X \to Z$ where $X \subseteq \mathbb{R}^n$ and $Z \subseteq \mathbb{R}^m$.

Unlike single-objective optimisation, where there are only three relations between solutions with respect to their quality, *better*, *worse* and *equal*, in multi-objective optimisation there is a situation that two solutions are not comparable; namely, one solution is better on some objective(s) and the other is better on some other objective(s). This situation is termed as Pareto non-domination. Formally, a decision vector $x$ is said to (Pareto) dominate another $x'$, denoted by $x \prec x'$, iff

$$f_i(x) \leq f_i(x') \qquad \forall i \in \{1, ..., m\} \wedge f(x) \neq f(x') \tag{1}$$

For a solution set $S$, a solution $x$ ($x \in S$) is called a (*Pareto*) *nondominated solution* if there does not exist any $x' \in S$ such that $x' \prec x$. The set of such nondominated solutions is called the *nondominated set* of $S$, and its mapping in the objective space is called the *nondominated front*. For an MOP, in the case that $S$ represents all possible feasible solutions (i.e., $S = X$), the nondominated set is called the *Pareto optimal set* (or *global efficient set*). The mapping of the Pareto optimal set in the objective space is called the *Pareto optimal front*.

Besides the global efficient set, there usually exist some local optima in the search space, namely the *local efficient sets*, each consisting of a set of *local efficient points*. In continuous multi-objective optimisation, a solution $x$ is said to be local efficient if there does not exist $y \in B_\epsilon$ that $y \prec x$ [4], where $B_\epsilon \subset X$ represents an $\epsilon$ ball for a sufficiently small $\epsilon > 0$, meaning that all neighbours of $x$ within a sufficiently small distance do not dominate $x$.

An important property in continuous multi-objective optimisation is the *multi-objective gradient* (MOG) [10]. MOG is a vector that is oriented towards a descent direction by the sum of its normalised single-objective components. It has a desired characteristic that MOG $= 0$ if a solution is a local or global efficient solution. Formally, MOG is defined as

$$\nabla f(x) = \sum_{i=1}^{m} \alpha_i^* \nabla f_i(x) \tag{2}$$

where $\alpha^*$ is a weight vector determined based on

$$\alpha^* = \arg\min_{\alpha} \left\{ \left\| \sum_{i=1}^{m} \alpha_i \nabla f_i(x) \right\| \;\middle|\; \alpha_i \geq 0, \sum_{i=1}^{m} \alpha_i = 1 \right\} \tag{3}$$

### 2.2 Related Work

Amongst various fitness landscape visualisation methods in multi-objective optimisation [3,28], we consider the ones that work on the native decision space (rather than a compressed, abstract space). Specifically, we consider three visualisation methods, cost landscape [11], gradient field heatmap [17] and PLOT [26], which are relevant to the proposed method. We now briefly introduce them (in Sect. 4 we will compare our method against them).

**Cost Landscape** [11]**.** Proposed in Carlos M. Fonseca's PhD thesis, cost landscape is a useful visualisation tool for observing the global structure of a multi-objective problem. In some visualisation platforms (e.g., BBOB visualisation website [1]), it is also called dominance ratio. In cost landscape, the fitness (which can be reflected by colour or height in the plot) of a point is the number of points that dominate it in the point set considered (i.e., *Pareto rank* [11]).

**Gradient Field Heatmap (GFH)** [17]**.** GFH, proposed by Kerschke and Grimme, is the first attempt to employ gradient to visualise the landscape of

multi-objective problems. In GFH, the fitness (represented by colour) of a point is obtained by the cumulative lengths of MOGs (Eq. (2), the sum of normalised single objective gradients) on the path to a locally efficient point. More specifically, starting from a point and a variable to store the cumulative MOGs, GFH performs a search that moves to one of its 8 neighbouring points according to its MOG, and at the same time, accumulates the MOG of each step. The move is iterated until it hits a local efficient point. Then the fitness of the starting point is the cumulative length of the MOGs. GFH is considered a good tool to capture the local structures of the problem [28].

**Plot of Landscapes with Optimal Trade-offs (PLOT) [26].** More recently, Schäpermeier et al. developed PLOT, which can be seen as an improved version of GFH in combination with cost landscape. Similar to GFH, PLOT employs the cumulative length of MOGs as its fitness, but with a grey-scale colour scheme. The global and local efficient points are highlighted by the colour on the basis of the Pareto rank considered in cost landscape. As such, it is very clear to see global/local optimality structure of the problem.

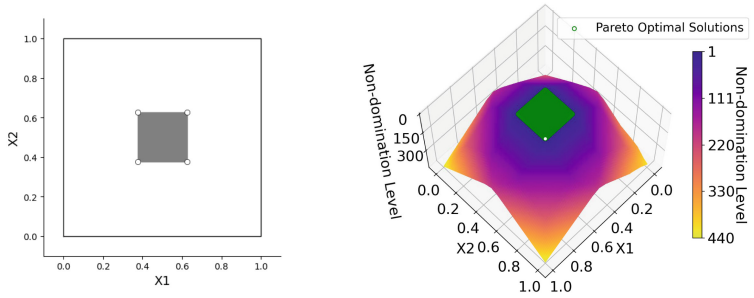
## 3 The Proposed Pareto Landscape

Classical fitness landscape, introduced by the geneticist Sewall Wright in 1930s [31], is a 3D "mountainous" landscape in which the genotypes (decision variables) of solutions are organised in the $x$-$y$ plane and the phenotype (fitness) is plotted as the height (i.e., on the $z$ axis). Fitness landscape plots have been widely used in optimisation and evolutionary computation [16,25]. They enable an intuitive understanding of diverse problem characteristics and spatial features, encompassing peaks, valleys, plateaus, ridges, and funnels, as well as optimality-related aspects like multi-modality, level of ruggedness, and the shape/location of (local) optimal solutions along with their attraction basins.

Yet, such straightforward visualisation cannot be directly extended to the multi-objective case in which a solution corresponds to a vector consisting of values of multiple objective functions. A possible solution to this issue is to utilise the Pareto dominance relation to sort and rank the solutions and then plot their ranks as the height, instead of objective function values, in the landscape.

The non-dominated sorting procedure [12] is a well-established way to sort solutions in multi-objective optimisation, e.g., used in NSGA-II [5]. It stratifies a set of solutions into many levels based on their Pareto dominance relation, with level 1 meaning nondominated solutions of the set, level 2 meaning new nondominated solutions after removing the level-1 solutions, and so on. In the proposed Pareto landscape, the "fitness" of a solution is its level after the non-dominated sorting procedure, termed the non-domination level. Figure 1 gives the Pareto landscape[1] of a 4-objective optimisation problem called multi-point distance minimisation problems (MP-DMPs). MP-DMPs, introduced by [19] and

[1] In this paper, the points to plot Pareto landscape are those evenly distributed on a $500 \times 500$ grid unless specifically stated otherwise.




(a) Decision space (b) Pareto landscape

**Fig. 1.** Pareto landscape of a four-objective multi-point distance minimisation problem (MP-DMP) [9,15] whose four objectives of a point are the Euclidean distances of a point in the 2D space to the four vertices of a square, respectively. The decision space of the MP-DMP is $x_1, x_2 \in [0, 1]$, and the Pareto optimal set of the problem is the square (shaded region in Fig. 1(a)).

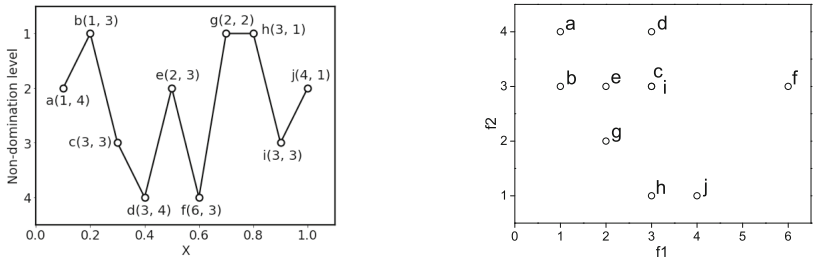
extended and generalised by [9,15], are class of increasingly popular test functions which are capable of being embedded with a variety of features (see [9] for a survey). One prominent feature of MP-DMPs is that they inherently have a 2D decision space (despite easily accommodating an arbitrary number of objectives), thus allowing the Pareto optimal solutions to be visually identified.

MP-DMPs, in its basic form, minimise the Euclidean distance of a solution to the vertices of a given regular polygon, where the distance to one of these vertices is treated as an individual objective. It can be easily derived that the Pareto optimal region of an MP-DMP is the regular polygon.

Figure 1(b) plots the Pareto landscape of the MP-DMP in Fig. 1(a), where the Pareto optimal solutions are highlighted in green for facilitating observation. As can be seen in the figure, Pareto landscape can well reflect the problem's landscape – points being farther away from the optimal region, gradually and symmetrically, take lower altitude (i.e., higher non-domination levels), with the four corners of the decision space having the lowest altitude touching the $x_1$-$x_2$ plane. Next, we will use a toy example to explain the characteristics of Pareto landscape, in comparison with fitness landscape in single-objective optimisation.

### 3.1 Properties and Characteristics of Pareto Landscape

Let us consider a set of 10 solutions ($\mathbf{a} - \mathbf{j}$) with one decision variable ($x$) and two minimisation objectives ($f_1, f_2$). In the format of $(x|f_1, f_2)$, these solutions are $\mathbf{a} = (0.1|1, 4)$, $\mathbf{b} = (0.2|1, 3)$, $\mathbf{c} = (0.3|3, 3)$, $\mathbf{d} = (0.4|3, 4)$, $\mathbf{e} = (0.5|2, 3)$, $\mathbf{f} = (0.6|6, 3)$, $\mathbf{g} = (0.7|2, 2)$, $\mathbf{h} = (0.8|3, 1)$, $\mathbf{i} = (0.9|3, 3)$, and $\mathbf{j} = (1.0|4, 1)$. Figure 2 gives their Pareto landscape as well as the scatter plot in the objective space. As can be seen in the figure, the 10 solutions can be divided into four non-domination levels. Solutions $\mathbf{b}, \mathbf{g}, \mathbf{h}$ form the first level, solutions $\mathbf{a}, \mathbf{e}, \mathbf{j}$ the second level, solutions $\mathbf{c}, \mathbf{i}$ the third level, and solutions $\mathbf{d}, \mathbf{f}$ the last level. Looking




(a) Pareto landscape (with objective values $(f_1, f_2)$) (b) Objective space

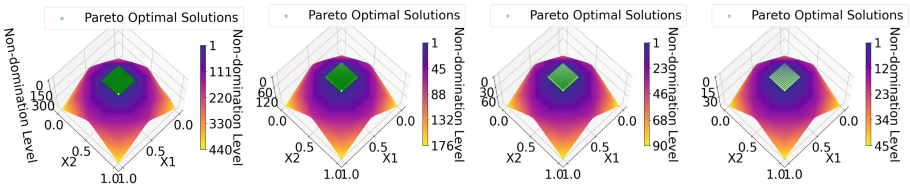
**Fig. 2.** A toy example of a set of 10 solutions ($\mathbf{a}-\mathbf{j}$) with one decision variable ($x$) and two minimisation objectives ($f_1, f_2$), shown by Pareto landscape. They, in the format of $(x|f_1, f_2)$, are $\mathbf{a} = (0.1|1, 4)$, $\mathbf{b} = (0.2|1, 3)$, $\mathbf{c} = (0.3|3, 3)$, $\mathbf{d} = (0.4|3, 4)$, $\mathbf{e} = (0.5|2, 3)$, $\mathbf{f} = (0.6|6, 3)$, $\mathbf{g} = (0.7|2, 2)$, $\mathbf{h} = (0.8|3, 1)$, $\mathbf{i} = (0.9|3, 3)$, $\mathbf{j} = (1.0|4, 1)$.

closer at the figure, one may find the following properties of Pareto landscape, in comparison with fitness landscape in single-objective optimisation.

- Like fitness landscape, optimal solutions (i.e., Pareto optimal solutions in the context of multi-objective optimisation) are always located in the highest peak(s), such as solutions $\mathbf{b}, \mathbf{g}, \mathbf{h}$ in Fig. 2.
- Like fitness landscape, it is straightforward to see the characteristics of the problem through Pareto landscape, e.g., local optimality and connectedness of Pareto optimal solutions. For example, in Fig. 2 solution **e** is a local optimal solution; optimal solution **b** is disconnected from the other two optimal solutions **g** and **h**.
- Like fitness landscape, in Pareto landscape, for any two solutions, if one is superior to the other (i.e., Pareto dominating), then the better one is always in a higher location, such as solutions **b** versus **a**, and solutions **g** versus **e** in Fig. 2.
- Like fitness landscape, in Pareto landscape if two solutions have identical objective vectors, they have the same altitude, such as solutions **c** and **i** in the figure.
- Like fitness landscape, in Pareto landscape the slope indicates the degree of change of a solution to its neighbours (with respect to non-domination levels). For example, in Fig. 2 the slope from solution **b** to solution **c** is steeper than that from solution **c** to solution **d** since the level difference between **b** and **c** is greater than that between **c** and **d** (i.e., $2 > 1$).

The above are properties of Pareto landscape that are analogous to fitness landscape in single-objective optimisation. Nevertheless, Pareto landscape does have its own characteristics.

- Unlike fitness landscape, higher altitude in Pareto landscape always means being superior, irrespective of a minimisation or maximisation problem.




(a) $500 \times 500$ (b) $200 \times 200$ (c) $100 \times 100$ (d) $50 \times 50$

**Fig. 3.** Pareto landscape of the four-objective MP-DMP problem with different sizes of the considered solution set in the space, where the solutions are uniformly chosen from $500 \times 500$, $200 \times 200$, $100 \times 100$ and $50 \times 50$ grids, respectively.

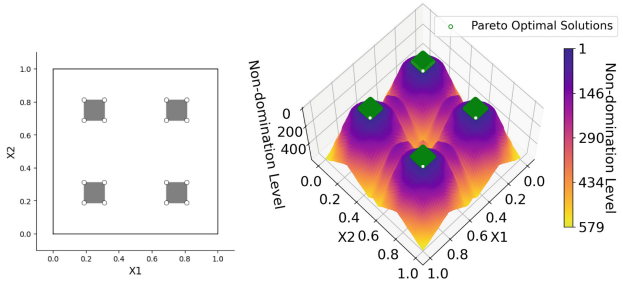
- Since Pareto landscape is based on the Pareto dominance relation rather than objective values, solutions with significantly different objective disparities may exist within the same non-domination level provided that they are nondominated to each other. For example, in Fig. 2 for the same-level solutions **d** and **f**, on the first objective, **d** is significantly better than **f**, whereas on the second objective, **f** is only slightly better than **d**.
- Solutions that are nondominated to each other *may or may not* be in the same level; it depends on other solutions in the set. For example, in Fig. 2 solutions **e** and **j** are in the same level, whereas solutions **i** and **j** are not in the same level as solution **e** (who is in the same level as **j**) dominates **i**, thus "pushing" **i** down to the next level.

The reason the proposed Pareto landscape exhibits the last two characteristics is its reliance on the Pareto dominance relation between solutions rather than their native objective values. Pareto landscape is a "relative" landscape, and unlike fitness landscape, the levels of solutions can change depending on the considered solution set. That being said, the change in levels is unlikely to affect the spatial location (i.e., height) of solutions in the landscape, provided that a set of well-distributed solutions is sampled in the space. Figure 3 gives the Pareto landscape of the four-objective MP-DMP problem with different sizes of the considered set. As can be seen in the figure, despite having different scale of non-domination levels, the four landscapes look identical.

In the following section, we will look at more examples of the MP-DMP problem to understand how Pareto landscape reflects other characteristics of multi-objective problems, such as multi-modality, neutrality, and scalability.

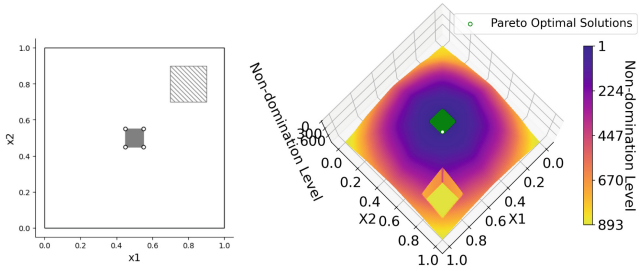
### 3.2 On Different MP-DMP Problem Instances

The MP-DMP problem has been frequently used to examine the behaviours of evolutionary algorithms in the area [14,20,23,29], due to its flexibility of equipping with diverse features [9], such as (global) multi-modality [15] and neutrality [8]. Multi-modality means multiple Pareto optimal solutions in the decision space mapping to the same point in the objective space, and neutrality means solutions in a particular region having the same objective values. The




(a) Decision space (b) Pareto landscape

**Fig. 4.** An example of the four-objective multi-modal MP-DMP with four Pareto optimal regions (shaded).



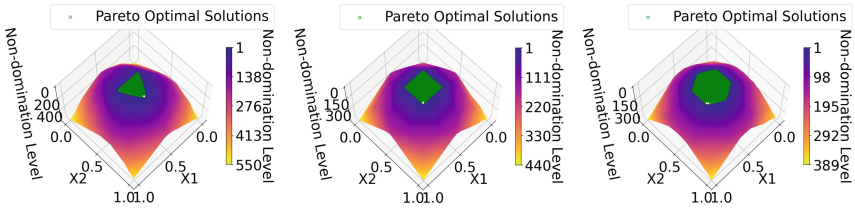

(a) Decision space (b) Pareto landscape

**Fig. 5.** An example of the four-objective MP-DMP with a Pareto optimal region (shaded) and a neutral region (dashed) of $x \in [0.7, 0.9]^2$, in which $f_1(x) = f_2(x) = f_3(x) = f_4(x) = 2$.

multi-modal MP-DMP is constructed by multiple polygons having the same shape and size, where an objective is the minimisation of distances of a solution to a set of counterpart vertices in the multiple polygons. Figure 4 gives a four-objective multi-modal MP-DMP instance, where the four congruent squares are Pareto optimal regions, each mapping to the whole Pareto front. As shown in the figure, Pareto landscape can well reflect this multi-modality feature, with the four Pareto optimal regions sitting on the four squares.

The neutrality feature of the MP-DMP problem can be easily generated by assigning a constant value to all solutions in a region [8]. Figure 5 presents an MP-DMP instance with a neural region of $x \in [0.7, 0.9]^2$, in which $f_1(x) = f_2(x) = f_3(x) = f_4(x) = 2$. Clearly, Pareto landscape can reflect this feature. All solutions in the neutral region have the same worst level as they are dominated by any solution in the rest of the decision space.

Another important feature of MP-DMP is its scalability in terms of the number of objectives, which is determined by the number of the vertices of the polygon. Figure 6 gives the Pareto landscape of three MP-DMP instances with three, four and six objectives. As can be seen in the figure,




(a) Tri-objective instance (b) Four-objective instance (c) Six-objective instance

**Fig. 6.** Pareto landscape of the MP-DMP instances with different numbers of objectives.

although different MP-DMP problems have different scales of non-domination levels, their landscapes look very similar, indicating that the level of difficulty they pose for search algorithms may be very similar.

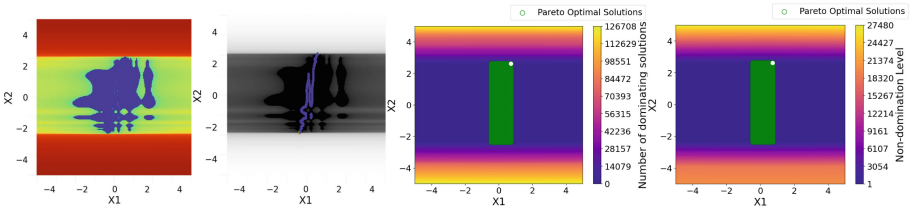
## 4 Comparison with Existing Methods

In this section, we compare Pareto landscape with three landscape visualisation methods, gradient field heatmap (GFH) [17], PLOT [26] and cost landscape [11]. The first two are recently proposed state-of-the-arts using a search algorithm to locate global/local efficient sets. Their plots were obtained directly from the moPLOT platform [27] developed by their authors. The third one is similar to ours – the only difference is that cost landscape considers Pareto rank (i.e., the number of solutions that dominate a solution), whereas our method considers the non-domination level a solution is located in. Here, due to their similarity, we use the same colour to represent their "fitness". Note that since existing methods are usually presented in a 2D space, for facilitating comparison, we present Pareto landscape in a 2D space as well (i.e., the height being removed).

A landscape visualisation method is deemed helpful if it can accurately reflect some important features in multi-objective optimisation. For example, we may hope that it can accurately identify Pareto optimal solutions of the problem. We may also hope that it is aligned with the Pareto dominance relation. That is, if two solutions (especially when they sit closely) are comparable in terms of dominance relation, it is desirable they are visibly distinguishable (i.e., having distinct colours/height). Equally, if two solutions are nondominated to each other, it is desirable that they have the same or similar colour/height.

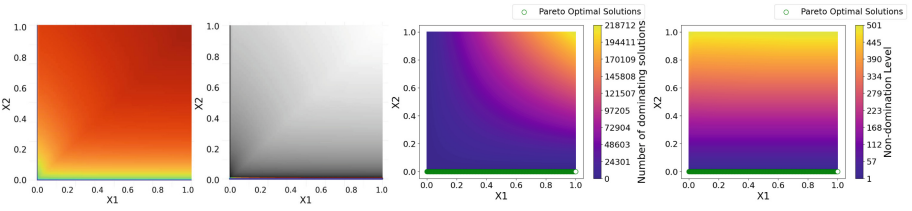
### 4.1 Comparison with Search-Based Visualisation Methods

As search-based methods, GFH [17] and PLOT [26] first sample a large number of points uniformly in a grid (e.g., a grid of $500 \times 500$ cells), and then for each cell, it considers its adjacent cells according to the steepest multiobjective gradient (MOG) direction (see Sect. 2.1) until the MOG of the cell's point is zero. However, in practice, in many cases locating a point whose MOG is precisely zero




(a) GFH (b) PLOT (c) Cost landscape (d) Pareto landscape

**Fig. 7.** GFH, PLOT, cost landscape and Pareto landscape of function 11 in bbob-biobj [2].




(a) GFH (b) PLOT (c) Cost landscape (d) Pareto landscape

**Fig. 8.** GFH, PLOT, cost landscape and Pareto landscape of ZDT1 [32].

may not be possible. As such, one needs to set a small number as a threshold $\delta$. In GFH [17] and PLOT [26], this parameter was set to $\delta = 1.0e-4$. On the other side, there may be non-optimal points with a very small MOG that is less than the threshold. Therefore, GFH and PLOT may not be able to accurately identify the Pareto optimal set.

Figure 7 shows the result of GFH and PLOT, along with the two non-search-based visualisation methods cost landscape and Pareto landscape, on the 2D bbob-biobj test function 11 [2], denoted by BBOB11. The Pareto optimal set of BBOB11 is a rectangle in the middle of the decision space. As can be seen in the figure, GFH and PLOT fail to identify the Pareto optimal set of the problem; the blue colour, which is supposed to represent Pareto optimal solutions, misses the target. In contrast, cost landscape and Pareto landscape can accurately reflect the Pareto optimal set (represented by the green points).

In addition, the way that search-based methods perform the search also plays a role in their accuracy. In GFH and PLOT, the search is conducted on the pixel of the grid. For each step, the move can only be taken in 8 possible directions: up, down, left, right, and the 4 diagonals, which may not be aligned with the actual steepest direction of MOG. Moreover, the fitness of a point is calculated based on the length of the path to a local efficient set. Points which have the same quality may have different path lengths. For example, Fig. 8 shows the result of GFH, PLOT, cost landscape and Pareto landscape on the well-known problem ZDT1 [32]. Looking closely at Fig. 8(a) and (b), one may see that there is a diagonal line with inferior fitness than points on either side. This does not

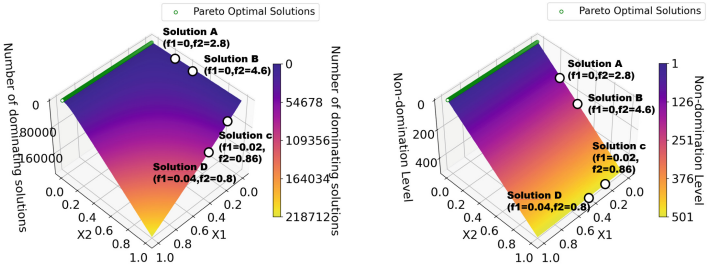
mean the points in the diagonal line are worse than others in quality, but means they have longer length than others. As such, the search-based methods, which well reflect the gradient direction, may not be very accurate to reflect the quality of solutions.

In Fig. 8, one may also find differences between cost landscape and the proposed Pareto landscape, which is not like the example in Fig. 7 where the plots obtained by the two methods are virtually the same. For example, in cost landscape (Fig. 8(c)), the worst area is the top-left, whereas in Pareto landscape (Fig. 8(d)), the worst is the top. In the next section, we will discuss differences between cost landscape and Pareto landscape.

### 4.2 Comparison with Cost Landscape

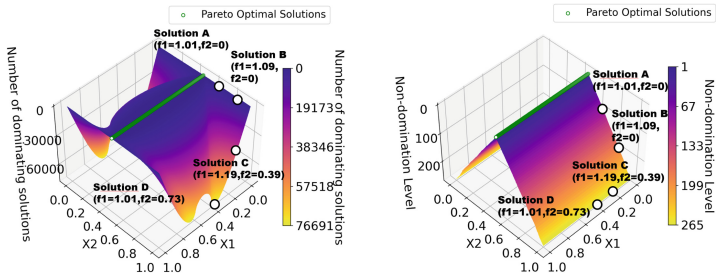
Cost landscape [11] measures the fitness of a point based on the number of points in the whole set that dominate the point. Like Pareto landscape, it provides "relative" fitness of the set considered. However, cost landscape is often much more sensitive than Pareto landscape to points in the set considered. Adding every new point that dominates the concerned point will change its fitness in cost landscape. This, however, is not the case for Pareto landscape since it considers the level of a Pareto optimal front. For example, in the example of Fig. 2, adding any nondominated points between **g**=(2,2) and **h**=(3,1) will not affect the level of any point in the set. However, for cost landscape, adding such a point will affect the fitness of points **d**, **c**, **i** and **f**. This characteristic of Pareto landscape can make a point's fitness more steady and robust to other points.

Figure 9 gives such an example, where the 3D plots of cost landscape and Pareto landscape on ZDT1 are displayed. For ZDT1, when it has only two variables, X1 and X2, one can easily derive two properties. That is, 1) for a solution, with X1 being fixed, increasing its value on X2 will always generate a new solution that is dominated by it; and 2) with X2 being fixed, changing its value on X1 will always generate a new solution that is nondominated to it (the proofs can be found in https://github.com/zxc990/Pareto-landscape). However, cost



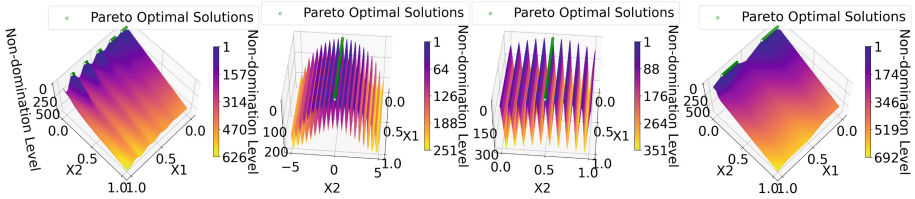

(a) Cost landscape (b) Pareto landscape

**Fig. 9.** 3D cost landscape and Pareto landscape of ZDT1, where the coordinates (X1, X2) of points **A**–**D** are **A**(0, 0.2), **B**(0, 0.4), **C**(0.2, 0) and **D**(0.4, 0).




(a) Cost landscape (b) Pareto landscape

**Fig. 10.** 3D cost landscape and Pareto landscape of DTLZ2, where the coordinates (X1, X2) of points **A**–**D** are **A**(0, 0.6), **B**(0, 0.8), **C**(0.2, 1) and **D**(0.4, 1).




(a) ZDT3 (b) ZDT4 (c) DTLZ1 (d) DTLZ7

**Fig. 11.** Pareto landscape of ZDT3, ZDT4, DTLZ1 and DTLZ7.

landscape cannot reflect that. As can be seen from Fig. 9(a), solutions **A** and **B** have the same value on X1 (hence **A** dominating solution **B**); however, they are virtually on the same height. On the other hand, solutions **C** and **D** have the same value on X2 (hence being non-dominated to each other); however, solution **C** is significantly higher than **D** by cost landscape. The same misleading result happens for the problem DTLZ2 [6], shown in Fig. 10(a). From the figure, it is clear that, by cost landscape, solution **A** dominates **B** but they have the same height; solutions **C** and **D** are nondominated to each other but their heights are highly different. In contrast, Pareto landscape is in line with the problem's properties (Fig. 9(b) and Fig. 10(b)) – solution **A** is always higher than **B**, and solutions **C** and **D** are always on the same level.

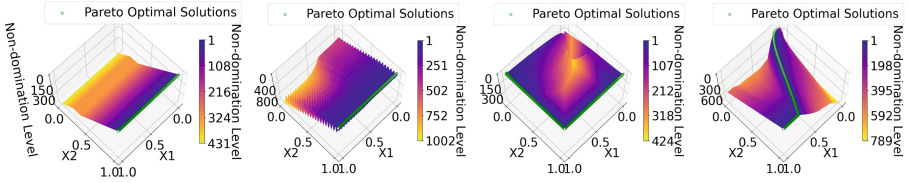
## 5 Pareto Landscape on Commonly Used Problems

In this section, we use Pareto landscape to plot several widely used problem suites in the domain. They are ZDT [32], DTLZ [6], WFG [13] and BBOB (i.e., bbob-biobj) [2]. For each BBOB problem, there are multiple instances; here the first instance was considered. Due to space limitations, we do not show all test problems for a suite but several representatives. For visualisation, all the problems were set to be in a 2D decision space and with two objectives.

### 5.1 ZDT and DTLZ

Figure 11(a) and (b) give the Pareto landscape of two ZDT problems, ZDT3 and ZDT4. A characteristic of the ZDT suite is that the Pareto optimal solutions are located in a region (or regions) where all decision variables except $x_1$ are zero [32], as can be seen from the Pareto landscape in Fig. 11(a) and (b). As shown, ZDT3 has five disconnected Pareto optimal regions; note that the problem is not multi-modal, these disconnected regions corresponding to different parts of the Pareto front in the objective space. This feature poses somewhat of a challenge for search algorithms to locate all the optimal regions, but the challenge is mild since the overall landscape is smooth and there are no local optimal regions that can trap search algorithms. ZDT4 is a challenging problem in the suite. From Fig. 11(b), we can see that it has many local optima, which can certainly trap the algorithms if their search step is not appropriate (e.g., too small).

The DTLZ suite [6] generally has similar problem features and behaviours to the ZDT suite. Figure 11(c) and (d) give the Pareto landscape of two DTLZ problems, DTLZ1 and DTLZ7. As can be seen from Fig. 11(c), DTLZ1, similar to ZDT4, has a number of equally-spaced local optimal regions (despite with deeper valleys). DTLZ7 is similar to ZDT3 and has disconnected Pareto optimal regions when $x_2 = 0$ (see Fig. 11(d)), but the gap between the optimal regions is bigger, hence more challenging for search algorithms to jump to the other when they find one of the local optimal regions.




(a) WFG1 (b) WFG4 (c) WFG7 (d) WFG8

**Fig. 12.** Pareto landscape of several WFG problems.

### 5.2 WFG

Figure 12 gives the Pareto landscape of four WFG problems, WFG1, WFG4, WFG7 and WFG8. As can be seen from the figure, a difference of WFG1 from the previous problems is that it has a neural area (around $x_2 = 0.2$), hence posing challenges for search algorithms, particularly for local search ones, to make progress. For WFG4, there are many local optimal peaks along both $x_1$ and $x_2$ directions, which contrasts those only along $x_2$ direction on ZDT4 and DTLZ1. As shown in Figs. 12(c) and (d), WFG7 and WFG8 have two piecewise Pareto optimal lines since both contain a transition function where one of the decision variables is mapped onto two regions [13]. It is worth mentioning that the

variables of WFG7 on the two Pareto optimal lines can be optimised separately, whereas the variables of WFG8 on one Pareto optimal line are non-separable.

### 5.3 BBOB

A common feature amongst the previously considered problem suites ZDT, DTLZ and WFG is that they all inherently have some "man-made patterns" (so that the Pareto optimal region is known and controllable). This includes Pareto optimal solutions clustered at regions where certain variables are constants, and local optimal regions that are equally spaced. This may make them easily exploitable by search algorithms; for example, by separately searching for optimum on different variables or employing specific search step-sizes during the search process. This is not the case in the BBOB suite (i.e., bbob-biobj), where the two objectives are made up of well-known single-objective functions (so that the Pareto optimal region is not known nor controllable).

Figure 13 shows the Pareto landscape of eight bbob-biobj problems. As can be seen from the figure, their landscapes have a great variety. There are problems of uni-modality (BBOB1 and BBOB29), high ruggedness (BBOB10, BBOB47 and BBOB55), global structure (BBOBF47), big mountain-like local optima (BBOB10 and BBOB55), and small spike-like local optima (BBOB18, BBOB39 and BBOB53). Moreover, the Pareto optimal regions of the problems are rather irregular. They include the optimal region as a simple straight line (BBOB1), a broken line (BBOB29), several separate lines (BBOB10 and BBOB55), disconnected points (BBOB47), and a mix of lines and points (BBOB18, BBOB39 and BBOB53).

Note that the disconnectedness of Pareto optimal regions in many BBOB problems represents an important feature that challenges search algorithms. To find the whole Pareto optimal region, search algorithms need to maintain

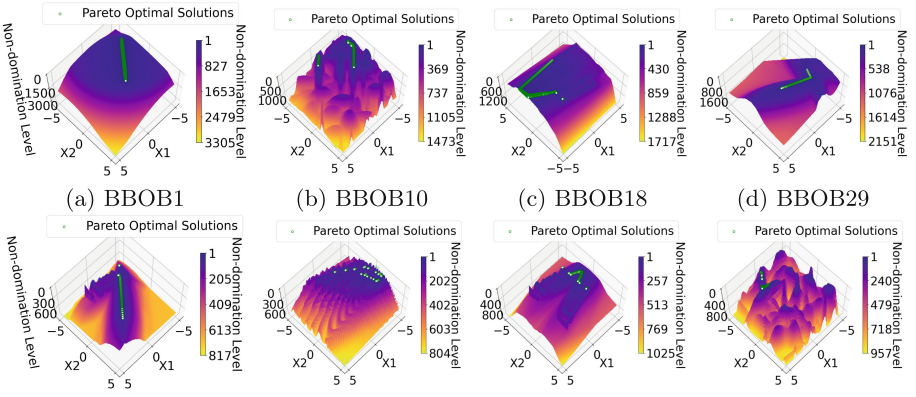



**Fig. 13.** Pareto landscape of several BBOB (namely bbob-biojb) problems.

diversity during the search – they cannot first search along one direction and, upon reaching an optimal solution, then move around within the neighbourhood. Unfortunately, in contrast to BBOB, most ZDT, DTLZ and WFG problems do not have this feature, making search algorithms easier to find the whole Pareto optimal region.

## 6 Conclusions

In this work, we attempted to develop a tool to display the "fitness" of solutions in multi-objective optimisation, termed Pareto landscape. We presented the similarity and dissimilarity of the Pareto landscape compared with fitness landscape in single-objective optimisation. Pareto landscape is different from conventional fitness landscape in the sense that it provides "relative" fitness of solutions in the set considered, but like fitness landscape, it can effectively capture a range of problem characteristics (e.g., optimality, multi-modality, neutrality, level of ruggedness, and location of attraction basins), and spatial features of the decision space (e.g., peaks, valleys, plateaus, ridges and funnels). We also compared Pareto landscape with other popular landscape visualisation methods and presented that it can reflect important features that other methods fail to. Note that like other methods which directly display the decision space, the proposed method can only work on problems with less than three variables.

Lastly, we employed Pareto landscape to visualise commonly used multi-objective benchmark problem suites in the domain and discussed their features and characteristics. We found that the ZDT, DTLZ and WFG problems exhibit some man-made features which can be easily exploited by specific search configurations, whereas the BBOB problems may better represent challenging real-world optimisation scenarios with diverse, natural features.

## References

1. Brockhoff, D., Auger, A., Hansen, N., Tušar, T.: BBOB biobj Visualizations (2021). https://numbbo.github.io/bbob-biobj/vis
2. Brockhoff, D., Auger, A., Hansen, N., Tušar, T.: Using well-understood single-objective functions in multiobjective black-box optimization test suites. Evol. Comput. **30**(2), 165–193 (2022)
3. Brockhoff, D., Tušar, T.: GECCO 2023 tutorial on benchmarking multiobjective optimizers 2.0. In: Proceedings of the Companion Conference on Genetic and Evolutionary Computation, pp. 1183–1212 (2023)
4. Custódio, A.L., Madeira, J.F.A.: MultiGLODS: global and local multiobjective optimization using direct search. J. Global Optim. **72**(2), 323–345 (2018)
5. Deb, K., Pratap, A., Agarwal, S., Meyarivan, T.: A fast and elitist multiobjective genetic algorithm: Nsga-ii. IEEE Trans. Evol. Comput. **6**(2), 182–197 (2002)
6. Deb, K., Thiele, L., Laumanns, M., Zitzler, E.: Scalable test problems for evolutionary multiobjective optimization. In: Abraham, A., Jain, L., Goldberg, R. (eds.) Evolutionary Multiobjective Optimization. Theoretical Advances and Applications, pp. 105–145. Springer, Berlin (2005). https://doi.org/10.1007/1-84628-137-7_6

7. Fieldsend, J.E., Alyahya, K.: Visualising the landscape of multi-objective problems using local optima networks. In: Proceedings of the Genetic and Evolutionary Computation Conference Companion, pp. 1421–1429 (2019)
8. Fieldsend, J.E., Chugh, T., Allmendinger, R., Miettinen, K.: A feature rich distance-based many-objective visualisable test problem generator. In: Proceedings of the Genetic and Evolutionary Computation Conference, pp. 541–549 (2019)
9. Fieldsend, J.E., Chugh, T., Allmendinger, R., Miettinen, K.: A visualizable test problem generator for many-objective optimization. IEEE Trans. Evol. Comput. **26**(1), 1–11 (2021)
10. Fliege, J., Svaiter, B.F.: Steepest descent methods for multicriteria optimization. Math. Methods Oper. Res. **51**(3), 479–494 (2000)
11. Fonseca, C.M.M.D.: Multiobjective genetic algorithms with application to control engineering problems. Ph.D. thesis, University of Sheffield (1995)
12. Goldberg, D.: Genetic Algorithms in Search, Optimization, and Machine Learning. Addison-Wesley (1989)
13. Huband, S., Hingston, P., Barone, L., While, L.: A review of multiobjective test problems and a scalable test problem toolkit. IEEE Trans. Evol. Comput. **10**(5), 477–506 (2006)
14. Ishibuchi, H., Akedo, N., Ohyanagi, H., Nojima, Y.: Behavior of EMO algorithms on many-objective optimization problems with correlated objectives. In: Proceedings of the IEEE Congress Evolutionary Computation, pp. 1465–1472 (2011)
15. Ishibuchi, H., Hitotsuyanagi, Y., Tsukamoto, N., Nojima, Y.: Many-objective test problems to visually examine the behavior of multiobjective evolution in a decision space. In: Proceedings of the International Conference on Parallel Problem Solving from Nature (PPSN), pp. 91–100 (2010)
16. Jones, T., et al.: Evolutionary algorithms, fitness landscapes and search. Ph.D. thesis, Citeseer (1995)
17. Kerschke, P., Grimme, C.: An expedition to multimodal multi-objective optimization landscapes. In: Trautmann, H., et al. (eds.) EMO 2017. LNCS, vol. 10173, pp. 329–343. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-54157-0_23
18. Kerschke, P., et al.: Towards analyzing multimodality of continuous multiobjective landscapes. In: Handl, J., Hart, E., Lewis, P.R., López-Ibáñez, M., Ochoa, G., Paechter, B. (eds.) PPSN 2016. LNCS, vol. 9921, pp. 962–972. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-45823-6_90
19. Köppen, M., Yoshida, K.: Substitute distance assignments in NSGA-II for handling many-objective optimization problems. In: Obayashi, S., Deb, K., Poloni, C., Hiroyasu, T., Murata, T. (eds.) Substitute distance assignments in NSGA-II for handling many-objective optimization problems. LNCS, vol. 4403, pp. 727–741. Springer, Heidelberg (2007). https://doi.org/10.1007/978-3-540-70928-2_55
20. Li, M., Yang, S., Liu, X., Shen, R.: A comparative study on evolutionary algorithms for many-objective optimization. In: Proceedings of the 7th International Conference on Evolutionary Multi-criterion Optimization (EMO), pp. 261–275 (2013)
21. Liefooghe, A., Derbel, B., Verel, S., López-Ibáñez, M., Aguirre, H., Tanaka, K.: On pareto local optimal solutions networks. In: Auger, A., Fonseca, C.M., Lourenço, N., Machado, P., Paquete, L., Whitley, D. (eds.) PPSN 2018. LNCS, vol. 11102, pp. 232–244. Springer, Cham (2018). https://doi.org/10.1007/978-3-319-99259-4_19
22. Liefooghe, A., Ochoa, G., Verel, S., Derbel, B.: Pareto local optimal solutions networks with compression, enhanced visualization and expressiveness. In: Proceedings of the Genetic and Evolutionary Computation Conference, pp. 713–721 (2023)

23. Liu, Y., Ishibuchi, H., Nojima, Y., Masuyama, N., Shang, K.: A double-niched evolutionary algorithm and its behavior on polygon-based problems. In: Auger, A., Fonseca, C.M., Lourenço, N., Machado, P., Paquete, L., Whitley, D. (eds.) PPSN 2018. LNCS, vol. 11101, pp. 262–273. Springer, Cham (2018). https://doi.org/10.1007/978-3-319-99253-2_21
24. Ochoa, G., Liefooghe, A., Lavinas, Y., Aranha, C.: Decision/objective space trajectory networks for multi-objective combinatorial optimisation. In: Pérez Cáceres, L., Stützle, T. (eds.) EvoCOP 2023. LNCS, vol. 13987, pp. 211–226. Springer, Cham (2023). https://doi.org/10.1007/978-3-031-30035-6_14
25. Ochoa, G., Malan, K.: Recent advances in fitness landscape analysis. In: Proceedings of the Genetic and Evolutionary Computation Conference Companion, pp. 1077–1094 (2019)
26. Schäpermeier, L., Grimme, C., Kerschke, P.: One PLOT to show them all: visualization of efficient sets in multi-objective landscapes. In: Bäck, T., et al. (eds.) PPSN 2020. LNCS, vol. 12270, pp. 154–167. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-58115-2_11
27. Schäpermeier, L., Grimme, C., Kerschke, P.: moPLOT website (2021). https://schaepermeier.shinyapps.io/moPLOT/
28. Schäpermeier, L., Grimme, C., Kerschke, P.: Plotting impossible? surveying visualization methods for continuous multi-objective benchmark problems. IEEE Trans. Evol. Comput. **26**(6), 1306–1320 (2022)
29. Singh, H.K., Isaacs, A., Ray, T., Smith, W.: A study on the performance of substitute distance based approaches for evolutionary many objective optimization. In: Li, X., et al. (eds.) SEAL 2008. LNCS, vol. 5361, pp. 401–410. Springer, Heidelberg (2008). https://doi.org/10.1007/978-3-540-89694-4_41
30. Volz, V., Naujoks, B., Kerschke, P., Tušar, T.: Single-and multi-objective gamebenchmark for evolutionary algorithms. In: Proceedings of the Genetic and Evolutionary Computation Conference, pp. 647–655 (2019)
31. Wright, S., et al.: The roles of mutation, inbreeding, crossbreeding, and selection in evolution pp. 356–366 (1932)
32. Zitzler, E., Deb, K., Thiele, L.: Comparison of multiobjective evolutionary algorithms: empirical results. Evol. Comput. **8**(2), 173–195 (2000)

# Real-World Applications

# Improving Continuous Monte Carlo Tree Search for Identifying Parameters in Hybrid Gene Regulatory Networks

Romain Michelucci[1(✉)], Denis Pallez[1], Tristan Cazenave[2], and Jean-Paul Comet[1]

[1] Université Côte d'Azur, CNRS, I3S, Sophia Antipolis, France
{romain.michelucci,denis.pallez,jean-paul.comet}@univ-cotedazur.fr
[2] LAMSADE, Université Paris Dauphine - PSL, CNRS, Paris, France
tristan.cazenave@lamsade.dauphine.fr

**Abstract.** Monte-Carlo Tree Search (MCTS) is largely responsible for the improvement not only of many computer games, including Go and General Game Playing (GPP), but also of real-world continuous Markov decision process problems. MCTS initially uses the Upper Confidence bounds applied to Trees (UCT), but the Rapid Action Value Estimation (RAVE) heuristic has rapidly taken over in the discrete and continuous domains. Recently, generalized RAVE (GRAVE) outperformed such heuristics in the discrete domain. This paper is concerned with extending the GRAVE heuristic to continuous action and state spaces (cGRAVE). To enhance its performance, we suggest an action decomposition strategy to break down multidimensional actions into multiple unidimensional actions, and we propose a selective policy based on constraints that bias the playouts and select promising actions in the search tree. The approach is experimentally validated on a real-world biological problem: the goal is to identify the continuous parameters of gene regulatory networks (GRNs).

**Keywords:** MCTS · continuous GRAVE · constraints-based selective policy · action decomposition · chronotherapy · hybrid GRN

## 1 Introduction

MCTS is a general decision-time planning algorithm that was initially designed for the improvement of computer Go [13]. The MCTS core idea is to incrementally build a search tree whose nodes represent the states of the environment and edges represent the actions taken from one state to a successor state. MCTS has proved to be effective in a wide variety of settings, including General Game Playing (GGP) [15,23] but is not limited to games [5,26]: it can be effective for single-agent sequential decision problems if there is an environment model simple enough for fast multistep simulation. The most popular MCTS algorithm is Upper Confidence bounds applied to Trees (UCT) [19], which addresses the


M. Affenzeller et al. (Eds.): PPSN 2024, LNCS 15151, pp. 319–334, 2024.
https://doi.org/10.1007/978-3-031-70085-9_20

exploration *versus* exploitation trade-off in each state of the tree search using the Upper Confidence Bound [1]. The Rapid Action Value Estimate [16,17] is a simple yet powerful improvement. The RAVE algorithm combines UCT and the *all-moves-as-first* (AMAF) heuristic [4,6] to provide knowledge sharing between related nodes, resulting in a rapid but biased estimate of the action values. A generalization of the RAVE heuristic [8] has been proposed to gather more accurate estimates near the leaves: the resulting algorithm outperformed RAVE on multiple games such as Go, Atarigo, Knightthrough, and Domineering, without any specific knowledge.

Since the striking success of decision-time planning by MCTS in discrete action spaces, existing methods try to mitigate the requirement of enumerating all actions to deal with large-scale and continuous domains. Progressive Widening (PW) [11,14], also known as progressive unpruning [10], increases the number of child actions of a tree node based on its visitation count. cRAVE [12] adopts the RAVE heuristic using Gaussian convolution-based smoothing, reinforcing information sharing between similar states and actions in each node's sub-tree. Other variants of MCTS in the continuous domain abound. Kernel Regression-UCT [25] generalizes the value estimation between similar actions in a node through kernel regression. Thus, new action generation is guided by kernel density estimation. An alternative to UCT is to replace UCB action selection rule [1] by Hierarchical optimistic optimization (HOO) [7,22] to deal with continuous actions. HOO partitions the action space and builds a binary tree to gradually split it into subspaces. Examples of successful continuous MCTS applications have been: control tasks in OpenAI Gym environment [20], robotic planning [18], and action selection in an Olympic Curling simulator [25].

GRAVE outperformed RAVE in the discrete domain. As far as we know, this paper is the first adaptation of GRAVE to the continuous domain. In addition, we add two generic contributions that improve the MCTS performances in the continuous setting. First, the action decomposition strategy introduced proposes to split a multidimensional action into multiple unidimensional actions. This results in a tree policy reinforcing promising action components instead of the wrong ones and leading to better action selection, specifically when the tree search is shallow. Second, the core idea of a constraints-based selective policy suggests the development of a module that automatically extracts constraints and rules from the domain knowledge to reduce the action space before and during the execution of the procedure.

The paper is organized into three remaining sections: Sect. 2 presents related works for discrete and continuous MCTS from which we add some contributions detailed in Sect. 3, and Sect. 4 provides an experimental study on a real-world biological problem.

## 2 Monte Carlo Tree Search

### 2.1 Discrete MCTS

MCTS is a simulation-based tree search in which states of an environment are nodes and actions are edges. The basic version of MCTS uses Monte Carlo

simulations for evaluating the nodes of a search tree in order to direct future simulations towards better-rewarded trajectories. Given an initial computation resource (time, memory or number of iterations) and starting at the root node, MCTS executes four steps iteratively:

- selection: a *tree policy* decides which successor node to visit based on a selection function and each successor node statistics,
- expansion: when the selection step ends on a leaf node, the tree is expanded by adding a new node,
- simulation: from the expanded node, a simulation follows a *rollout policy* until a terminal node is reached,
- backpropagation: the reward value of the simulation is assigned to the expanded node and all of its ancestors.

The standard selection function for MCTS is UCB. It follows the principle of *optimism in the face of uncertainty*, which favors the actions with the highest potential value between exploitation (actions with a high mean reward value) and exploration (actions less selected). During the selection step, the action $a$ is chosen (among the set of legal actions $A(s)$ of state $s$) by applying the UCB formula:

$$argmax_{a \in A(s)}(mean_a + c \times \sqrt{\frac{log(n(s))}{n(s,a)}})$$

where $mean_a$ is the mean reward of $a$, $n(s)$ is the number of times the state $s$ is selected, $n(s,a)$ is the number of times $a$ is chosen after $s$ is selected, and $c$ is the exploration parameter that must be tuned for each problem: a low value favors exploitation while a high value encourages exploration.

The AMAF heuristic consists of updating the statistics of all actions both selected during the selection and simulation steps. Each of these actions is treated as if it was played on a previous selection step. The reward estimate for an action $a$ from a state $s$ is updated when $a$ is chosen in any playout (even if $a$ is not chosen from $s$).

RAVE is a popular UCT enhancement that blends the standard UCT score for each node with the AMAF score. Therefore, each node must hold a separate count of rewards and visits for each type. The UCB formula is replaced by:

$$argmax_{a \in A(s)}((1.0 - \beta_a) \times mean_a + \beta_a \times AMAF_a) \tag{1}$$

where $AMAF_a$ is the AMAF score of the action $a$, and $\beta_a$ is the dynamic weight calculated using $p_a$ the number of rollouts starting with $a$:

$$\beta_a = \frac{pAMAF_a}{pAMAF_a + p_a + bias \times pAMAF_a \times p_a} \tag{2}$$

in which $pAMAF_a$ is the number of rollouts containing $a$.

The GRAVE algorithm uses AMAF statistics of an ancestor state if it has more associated rollouts than a given constant called *ref*, which must be tuned for each problem. The idea behind GRAVE is that a state upper in the tree has better accuracy since it has more associated playouts. GRAVE is a generalization of RAVE since GRAVE with *ref* equals zero is RAVE.

### 2.2 Continuous MCTS

In continuous Markov Decision Processes, standard UCT or RAVE cannot be used since the standard selection step requires the trial of every action at least once, which is impossible in a continuous domain. The progressive widening (PW) heuristic has been proposed to deal with this issue by maintaining a limited number of actions to consider in each state $s$ depending on the number of times $s$ has been visited. Specifically and heuristically, a new child state is sampled from $s$ every time the visitation count of $s$ ($n(s)$) to the power of $pw$ is greater than or equal to its number of children ($n(s)^{pw} \geq |s.children|$). $pw$ is a problem-dependent parameter that controls the number of actions allowed in $s$. In a nutshell, while UCT ensures that the tree grows deeper in the promising regions of the search space by balancing exploration and exploitation, the PW strategy guarantees that it grows wider in those regions.

cRAVE is an extension of RAVE to the case of continuous action and state spaces. It considers a smooth estimate of action and state values using a Gaussian convolution. Formally, it states that the AMAF score of choosing an action $a$ from a state $s$ is weighted by the contribution related to the state-action pairs $(s_i, a_i)$ encountered in every tree-walk $x_s$ starting from $s$:

$$AMAF_{s,a} = \sum_{x_s, a_i \in x_s} e^{-logN_{a,s}\{\frac{d(s,s_i)^2}{\alpha_{state}} + \frac{d(a,a_i)^2}{\alpha_{action}}\}} \times \mathbf{R}(x_s) \tag{3}$$

where $\mathbf{R}(x_s)$ is the cumulative reward obtained after following $x_s$, $N_{a,s}$ denotes the number of state-action pairs involved in every $x_s$ (the sub-tree of $s$), and $\alpha_{action}$ (resp. $\alpha_{state}$) is a problem-dependent parameter tuning the importance of $d(a, a_i)$ (resp. $d(s, s_i)$) representing the distance between the action $a$ (resp. state $s$) and the considered action $a_i$ (resp. state $s_i$) from the sub-tree. The Euclidean distance is commonly chosen, but the choice of such a measure also depends on the problem. $pAMAF_{s,a}$ is the number of tree-walks containing the state $s$ followed by the action $a$ and is also computed using Gaussian convolutions:

$$pAMAF_{s,a} = \sum_{x_s, a_i \in x_s} e^{-logN_{a,s}\{\frac{d(s,s_i)^2}{\alpha_{state}} + \frac{d(a,a_i)^2}{\alpha_{action}}\}} \tag{4}$$

## 3 Contributions

Algorithm 1 encompasses the different contributions presented in this section.

### 3.1 Continuous GRAVE

GRAVE uses AMAF values of a state higher up in the tree than the current state to gather more accurate estimates near the leaves. Its reliability decreases as the number of actions increases: in a continuous action space, the number of times a given action is tried is zero in expectation. An estimation of action and state

**Algorithm 1.** Continuous GRAVE and enhancements

**Input:** $N$ tree-walks, initial state $s_0$, PW parameter $pw$, reference state constant $ref$
**Output:** A search tree

1: Initialize constraints from the CSP module
2: **for** $i = 1$ to $N$ **do**
3: $s = s_0$, $S = \{s\}$
4: $sref = s$
5: **while** $s$ is not a leaf state *and is not simulatable* **do** ▷ Tree-walk step
6: **if** $n(s)^{pw} < |s.children|$ **then** ▷ PW test, section 2.2
7: **if** $n(sref) <= ref$ OR $n(s) > ref$ **then** ▷ GRAVE reference state test
8: $sref = s$
9: **end if**
10: **for all** $a \in s.children$ **do** ▷ Compute $GRAVE(s, a)$
11: $\beta = \frac{sref.pAMAF}{sref.pAMAF + s.p + bias \times sref.pAMAF \times s.p}$ ▷ Eq. 2
12: $grave = (1. - \beta) \times s.mean + \beta \times sref.AMAF$ ▷ Eq. 1
13: **end for**
14: Select $a = argmax\{GRAVE(s, a) \mid a \in s.children\}$
15: **else**
16: Sample a new action $a$ from $A_{CSP}(s)$
17: Add $P(s, a)$ as a child node of $s$ ▷ P(s,a) is the transition function
18: **end if**
19: $s = P(s, a)$, $S = S \cup \{s\}$
20: **end while**
21: **while** $s$ is not a terminal state **do** ▷ Simulation step
22: Sample $a \in A_{CSP}(s)$ based on default policy
23: $s = P(s, a)$, $S = S \cup \{s\}$
24: **end while**
25: $score = evaluate(s)$
26: **for all** $s \in S$ **do** ▷ Backpropagation step
27: Update $s$ with $score$ ▷ Eq. 3 & 4
28: **end for**
29: **end for**

values must be considered, such as Gaussian convolution smoothing. We propose to adapt the GRAVE algorithm to the continuous domain called *continuous GRAVE* (cGRAVE). The only difference is that the AMAF statistics are updated using Gaussian convolution smoothing (line 27 of Algorithm 1 follows Equation (4)). For computing cGRAVE (lines 10 to 13), the closest ancestor state having more rollouts than a given *ref* constant is kept as a reference state (called *sref* and involved in lines 7–9), and its AMAF statistics are calculated (to calculate lines 11–12).

### 3.2 Action Decomposition

Many real-world problems involve continuous action spaces, $a \in \mathbb{R}^d, d \in \mathbb{N}^*$, where the set of possible actions is not finite. In addition, the dimension $d$ of the action space can also be high, making continuous MCTS methods less effective.

To leverage this issue, we propose an *action decomposition* (AD) strategy, which consists of breaking down each action of $d$ components ($a = (x_1, ..., x_d)$) into $d$ unidimensional actions ($a_1 = x_1, ..., a_d = x_d$). Therefore, the choice of each action component is left to the tree policy. In the expansion step, starting from a state node $s$, a first action component is sampled. Then, from this action node, a new action component is expanded. By iterating $d$ times, the final component choice leads to a new state node from which the simulation step takes place.

While using this AD strategy, the search tree contains actions from which no simulation can be done. Line 5 of Algorithm 1 ("*not simulatable*") checks whether or not $s$ is a simulatable state. If $s$ is not simulatable, a new action component must be sampled. Finally, if the AD strategy is not chosen, $s$ is always simulatable since its parent action already comprises $d$ components.

The advantage of such a strategy would be that each action node is considered as a traditional tree node in the selection step, leading to a tree policy that gradually reinforces the best action components at the expense of the wrong ones. Figure 1 illustrates such strategy: instead of considering the actions $a_i = (a_{i,1}, ..., a_{i,d})$, $i \in [\![1, n]\!]$ and $n \in \mathbb{N}^*$, as a whole (and the actions resulted by the combination of different components), they are decomposed such that the resulting tree search is composed of the distinct action components.

The effectiveness of this strategy is dependent on the order in which the action components are expanded. In a general framework, it is often impossible to obtain an optimal order of actions, either because the components are dependent or the order function of the components is unknown. If no information is available, a random order of components can be chosen. Otherwise, a heuristic, or even a dynamic calculation of the order of the components, through iterative deepening search, for instance, may reduce the convergence speed and lead to a case of more favorable complexity.

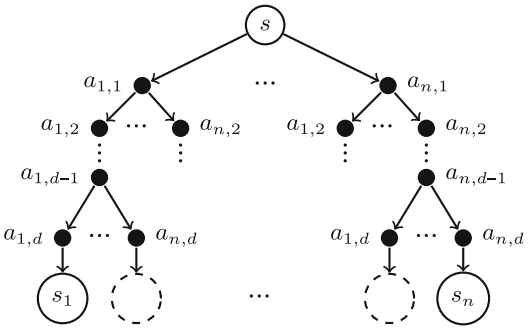



**Fig. 1.** Action decomposition strategy illustrated (in a deterministic case). The multi-dimensional actions are decomposed component by component following a particular ordering and forming a tree structure.

### 3.3 Constraints-Based Selective Policy

The next suggestion we make is a *constraints-based selective policy* (CSP), which takes its roots from selective policies [9], successfully introduced in MCTS variants. The underlying principle is to keep more selectivity in the rollouts. This idea can be applied to any problem by modifying the legal moves set of a node so that moves that are unlikely to be good are pruned during the rollouts. The idea underlying CSP is to automatically extract constraints from the environment definition, some input data, or expert knowledge. The goal is to reduce the action space so that action values that are unlikely to be good or infeasible concerning an extracted constraint are pruned before and during the execution. A module is built to extract constraints that can be used during the expansion step to choose only legal actions and remove ones that will lead to an impossible state (a state in which there is an empty set of legal actions). This module is helpful in the tree to select promising actions, but it can also be during the simulation step, where information sharing (such as kernel regression or AMAF statistics) does not take place: it can be used in playouts to bias the policy.

In real-world applications, the action space $A(s)$ of a state $s$ is bounded, such that each action $a_i$ is defined inside a specified domain $D_i$. Each domain $D_i$ consists of a set of possible values. The CSP allows extracting a set of constraints $C_i$, which reduces its corresponding $D_i$. Such action space is denoted $A_{CSP}(s)$ (lines 16 and 22 in Algorithm 1). These constraints can be implicit or explicit and depend on the application domain. Extracting these constraints helps to reduce the dimension of $D_i$ not only to legal actions but also to actions having a higher probability of being part of a solution. It can be done *a priori* and during the execution of the search. It alleviates the non-locality problem in which some actions may never be reached due to a previously performed action, even if that action belongs to the legal set. The construction of such an extraction module can be viewed as a generalized framework to apply to multiple distinct applications at the price of designing the module. A use case is provided in the next section.

## 4 Application

We empirically validate our method on the real-world problem of identifying continuous parameters of hybrid gene regulatory networks (hGRNs). First, we describe the problem. Then, the experimental design and setup are detailed. Finally, the results are highlighted and analyzed.

### 4.1 Parameters Identification of hGRNs

Gene regulatory networks (GRNs) modeling is a functional framework for studying and understanding the effect of regulations inside a biological system. Usually, a GRN is represented as a directed graph in which vertices abstract one or multiple biological genes ($v_1, v_2$ in Fig. 2a) and edges express either the activation ($\xrightarrow{+n}$) or the inhibition ($\overset{+n}{\dashv}$) of one vertex by another only if the concentration of the source vertex is above its $n^{th}$ threshold. An unlabeled arrow

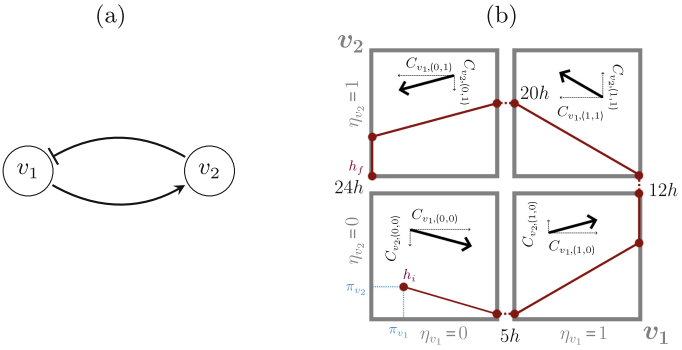



**Fig. 2.** Example of a hGRN depicted as a directed graph (a), and a possible hybrid state graph (b). The hGRN dynamic parameters are depicted as black arrows.

means that $n = 1$. Since each of these regulations may or may not occur, each gene abstraction has a maximum discrete level of concentration corresponding to the maximum number of targets it can regulate. In the context of Fig. 2a, the maximum discrete level of both genes is 1, leading to discrete levels of genes ($\eta_{v_1}$ and $\eta_{v_2}$) in $\{0, 1\}$. This graph is a static representation and is of limited interest because it does not help the modeler to predict any evolution of the system. Although a discrete dynamical framework has been developed, we consider here a hybrid modeling framework that adds chronometric aspects to the discrete framework, because it is fundamental to observe and reason not only about the discrete dynamics of a complex system but also about its chronometric evolution. This is particularly important in biology to optimize medical treatments by taking into account biological rhythms (*chronotherapy*). The hGRN dynamics of Fig. 2a is then built in two steps: (i) First, each grey box, representing a discrete state $\eta = (\eta_{v_1}, \eta_{v_2})$, defines the discrete concentration level of each gene. (ii) the hGRN dynamics are then defined as piecewise linear continuous trajectories (red lines). The trajectory starts from an initial hybrid state $h_i$, which is represented by both a discrete state $\eta$ and a vector determining the precise position $\pi$ inside the discrete state $\eta$. The initial state is defined by $h_i = \left((\eta_{v_1}, \eta_{v_2})^t, (\pi_{v_1}, \pi_{v_2})^t\right) = \left((0,0)^t, (0.25, 0.25)^t\right)$. Then, the evolution inside each discrete state is given by a so-called *celerity vector* (black arrows), which defines the direction and celerity of each gene, e.g., the celerity of $v_1$ in $\eta = (0, 0)$ is denoted $C_{v_1,(0,0)}$. More generally, the celerity of $v$ in $\eta$ is denoted $C_{v,\eta}$. Such models could help biologists make new interpretations of the possible system dynamics. Nevertheless, the bottleneck of the modeling framework is the identification of celerity vectors. We are interested in valid hGRN models of the biological system under study, that is, into hGRN models consistent with knowledge and observations.

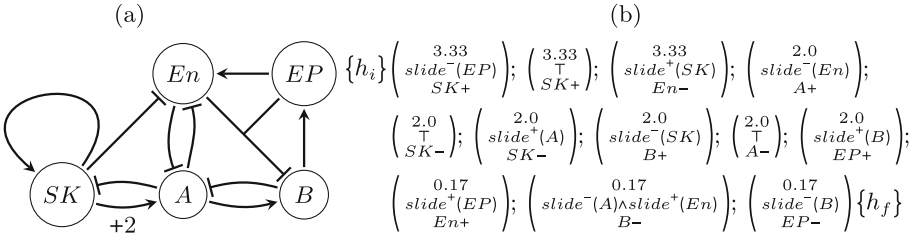



**Fig. 3.** Interaction graph of the 5-genes hGRN (a) and its corresponding biological knowledge (b).

Our approach takes into consideration already-formalized information analyzed by biologists derived from biological data and expertise instead of raw data, which are known to be subject to noisiness and scarcity. The approach abstracts the knowledge extracted from biological experiments under the form of constraints on the global trajectory: it must (i) start from an initial hybrid state $h_i = (\eta_i, \pi_i)$, (ii) verify a triplet of properties in each successive discrete state $(\Delta t, b, e)$ where $\Delta t$ expresses the time spent; $b$ delineates the observed continuous behavior inside the discrete state ($\top$ means the absence of observed behaviors); $e$ specifies the next discrete state transition, and (iii) reach a final hybrid state $h_f = (\eta_f, \pi_f)$. Figure 3b shows the biological knowledge (BK) associated with the interaction graph Fig. 3a of the cell cycle GRN [2] from which we want to extract solutions (valid models) in the next section. Note that the combination of arcs leading from En and EP to B represent a logical conjunction of the two control conditions. Starting from $h_i = \begin{pmatrix}(\eta_{SK}, \eta_A, \eta_B, \eta_{En}, \eta_{EP})^t = (0,0,0,1,0)^t \\ (\pi_{SK}, \pi_A, \pi_B, \pi_{En}, \pi_{EP})^t = (0.5, 0., 0., 1., 1.)^t\end{pmatrix}$, the trajectory must spend 3.33 hours in the discrete state $\eta = (0,0,0,1,0)$. Within this state, the celerity should move towards the next discrete state of $SK$ ($SK+$) to increase the concentration level of gene $SK$. In the meantime, the trajectory must also reach the minimum admissible concentration of $EP$ ($slide^-(EP)$). When $SK$ hits its threshold value, the trajectory jumps into the neighbor state $\eta = (1,0,0,1,0)$. This process continues discrete state after discrete state until the trajectory reaches $h_f$ ($= h_i$, forming a cycle). Any valuation of dynamic variables, i.e., celerities, leading to a trajectory satisfying this BK, is considered a solution to the hGRN identification problem.

### 4.2 Decision-Making Problem Specifications

The dimension of the decision-making problem is not the number of genes but a double exponential product: the number of celerities to identify in each discrete state is equal to $d$ (5, here), the number of possible states is $\prod_{v \in V}(b_v + 1)$ with $b_v$ the maximum discrete level of concentration of gene $v$ (48, here), the total number of celerities is $d \times \prod_{v \in V}(b_v + 1)$ (240, here).

To treat this problem with an MCTS approach, we consider each discrete state as an MCTS state and the choice of celerity vectors as an action. In each timestep, an action (the celerity vector choice) is composed of $d$ continuous variables where $d$ is the number of genes. When considering the use case of Fig. 3, there are five continuous variables, and, because of biological reasons, the action space domain can be bounded to $[-7.; 7.]^5$. As the BK is based on observations of 12 states, there are 12 continuous multidimensional actions to find a solution leading to a shallow-depth MCTS tree. However, due to the equality constraints on the time criterion in BK, the solution space forms a measure zero set, which complicates the learning process because an action component must find an exact floating-point value. Furthermore, some celerities in one discrete state may be identical in one or more other discrete states, leading to a hard-constrained problem. For this reason, a continuous constraint satisfaction problem solver approach failed to extract even one particular solution when considering these five genes GRN [3]. Finally, the reward for a rollout is the length of the trajectory before it ends in a final state: the maximum is 12 if the trajectory successfully passes between all discrete states (see Fig. 3b).

### 4.3 Design of Experiment and Experimental Setting

The goal of the experiments is to assess the efficiency of the different contributions added to the baseline cRAVE. The design of experiments is cumulative. First, cRAVE is compared to itself combined with the constraints-based selective policy (cRAVE$_{CSP}$). In the next step, we add to the previous algorithm the action decomposition strategy (cRAVE$_{CSP-AD}$). Similarly, we replace the cRAVE heuristic in cRAVE$_{CSP}$ with its improvement GRAVE in the continuous domain (cGRAVE$_{CSP}$). Finally, we aggregate the different contributions leading to a final version (cGRAVE$_{CSP-AD}$).

To avoid the disadvantages of the ad-hoc and manual parameter tuning of the algorithms, we decided to use an iterated racing procedure for automatic algorithm configuration. Using the `irace` package [21], we kept the best elite configuration obtained after 1000 iterations to determine the values of the problem-dependent parameters. For the cRAVE heuristics, the parameter space is $bias \in \{1e-15, 1e-14, ..., 1e-2, 0.1\}$, $\alpha_{state} \in [\![1, 100]\!]$, $\alpha_{action} \in [\![1, 100]\!]$, and $pw \in \{0.1, 0.11, ..., 0.89, 0.9\}$. The resulted tuned values are $bias = 0.1$, $\alpha_{state} = 47$, $\alpha_{action} = 87$, and $pw = 0.61$. For the cGRAVE heuristic, $ref$ is found among the values $[\![1, 100]\!]$ and equals 29. The Euclidean distance is chosen for both action and state spaces. Each experiment is run 30 times to obtain statistically significant results. The different policy values obtained are compared for the same computational budget, i.e., 200.000 tree-walks. This number comes from previous experiments assessed in [24].

### 4.4 Experiments

*Domain Knowledge.* To develop the CSP module, we used some knowledge of the hybrid framework. Indeed, some celerities, i.e., action components, are con-

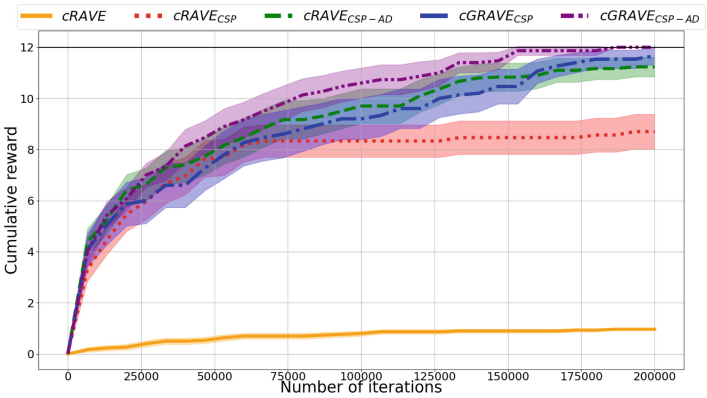



**Fig. 4.** Comparative performances (cumulative reward) of the different variants on the 5 genes hGRN, versus the computational budget (number of iterations). The upper the better: a reward of 12 means that a solution is found.

strained to be the same in the different discrete states that will be encountered by the trajectory. Therefore, *before the execution*, the module propagated some information of BK between the states that share at least one common celerity component. For example, the module automatically extracted that $C_{SK}$ in the third discrete state is constrained by $slide^{+}(SK)$. But it also extracted that the next discrete state shares the same celerity $SK$. Thus, the module helped to determine that (i) the celerity of $SK$ in the third state is positive due to the $slide^{+}$ knowledge (reducing the search space for this action to only positive values) and (ii) that when the value is known, it is kept to the next discrete state. *During the execution*, the module also helped to evict some action values. In the same example of $slide^{+}(SK)$ in the third discrete state, every positive value for $C_{SK}$ is considered a legal move. However, $C_{SK}$ value is impacted by its entry point, i.e., the hybrid state when entering the third discrete state. As a result, depending on the coordinates of the entry point, $C_{SK}$ values are adjusted online. There is no domain knowledge of the order function of the moves in the AD, so we have defined an arbitrary order among the genes (first $SK$, second $A$, then $B$, $En$, and finally $EP$) and kept the same for every decomposition.

*Analysis.* Figure 4 comparatively displays the monotonic evolution of the results obtained by the different tested algorithms. It can be observed that (i) the CSP is largely but non-surprisingly beneficial for the convergence of the different MCTS variants: it helps escape an early blockage, (ii) cRAVE$_{CSP-AD}$ and cGRAVE$_{CSP}$ both help to improve the findings of better cumulative rewards and (iii) when combining the three contributions, cGRAVE$_{CSP-AD}$ ensures to always find a solution of the problem, i.e., an admissible valuation of celerity vectors making it possible to satisfy BK (it hits the maximum threshold near 180,000 simulations).

Cumulative Distribution Function (CDF) curves are built in Fig. 5. Each CDF curve describes the probability of finding a solution at, or below, a given

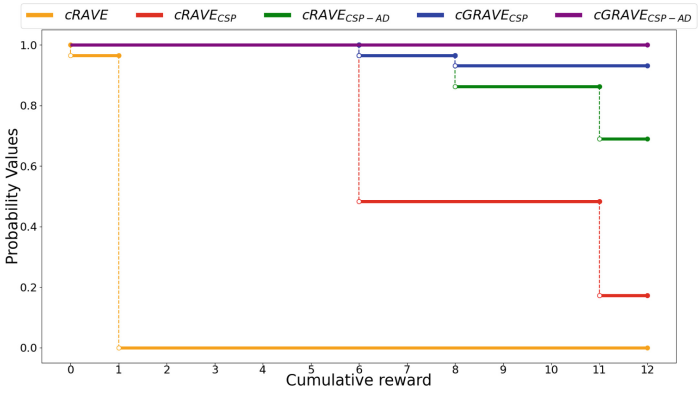



**Fig. 5.** CDF curves showing the best results for the different variants.

**Table 1.** Statistics of cumulative rewards gathered by the different algorithms tested. Bold values denote the best results column by column.

| Alg | mean ± std | max | min | % of solutions |
|---|---|---|---|---|
| cRAVE | 0.97 ± 0.18 | 1 | 0 | 0 |
| cRAVE$_{CSP}$ | 8.7 ± 2.72 | **12** | 6 | 20 |
| cRAVE$_{CSP-AD}$ | 11.2 ± 1.54 | **12** | 6 | 70 |
| cGRAVE$_{CSP}$ | 11.6 ± 1.3 | **12** | 6 | 93.33 |
| cGRAVE$_{CSP-AD}$ | **12.0 ± 0.0** | **12** | **12** | **100** |

cumulative reward score. For instance, with cRAVE$_{CSP}$, there is 100% probability that at the end of a run, a user would obtain a cumulative reward less than or equal to 6, and 50 % probability that the cumulative reward would be less or equal than 11. Table 1 summaries statistics of the best cumulative reward obtained for each run. The mean average and standard deviation of the results are reported, as well as the overall maximum and minimum cumulative reward (the best results column by column are shown in bold). For the maximum, the reader can refer to the x-axis of the rightmost point of each corresponding CDF curve that has more than 0% probability (in the y-axis).

Figure 5 and Table 1 quantitatively illustrate the interest of the different contributions. Without CSP, no solution can be found. And, on top of CSP, the AD strategy and cGRAVE enhance the probability of (i) obtaining a better cumulative reward and (ii) the percentage (%) of problem solutions found. The combination of our proposals (cGRAVE$_{CSP-AD}$) allows us to obtain a solution with a probability of 100%. Overall, the merits of the contributions are empirically demonstrated in this real-world biological problem.

### 4.5 Statistical Analysis

A statistical validation campaign was conducted to evaluate the observed differences in the reported performance values of all algorithm pairs in order to exhibit

**Table 2.** Pairwise Wilcoxon statistical tests (top) with Bonferroni post-hoc analysis (bottom) for $H_0$.

■ Fail to reject H0 □ Reject H0 ($p < 0.05$)

| | | | | |
|---|---|---|---|---|
| cRAVE | 1.17e-6 | 6.94e-7 | 1.45e-7 | 6.79e-8 |
| $\text{cRAVE}_{CSP}$ | | 9.84e-5 | 5.98e-5 | 8.89e-6 |
| $\text{cRAVE}_{CSP-AD}$ | | | 0.176 | 6.46e-3 |
| $\text{cGRAVE}_{CSP}$ | | | | 0.179 |

| | | | | |
|---|---|---|---|---|
| cRAVE | 1.17e-5 | 6.94e-6 | 1.45e-6 | 6.8e-7 |
| $\text{cRAVE}_{CSP}$ | | 9.84e-4 | 5.98e-4 | 8.9e-5 |
| $\text{cRAVE}_{CSP-AD}$ | | | 1.0 | 6.46e-2 |
| $\text{cGRAVE}_{CSP}$ | | | | 1.0 |
| | $\text{cRAVE}_{CSP}$ | $\text{cRAVE}_{CSP-AD}$ | $\text{cGRAVE}_{CSP}$ | $\text{cGRAVE}_{CSP-AD}$ |

the best algorithm variant. We consider the null hypothesis $H_0$ stating that the observed performance scores are equal. First of all, the choice between parametric and non-parametric tests is made according to the independence of the samples (seeds are different), whether or not the data samples are normally distributed (Kolmogorov-Smirnov test), and the homoscedasticity of the variances (Levene's test). As neither normality nor homoscedasticity conditions required for the application of the parametric tests hold (at $\alpha = 5\%$ confidence level), the non-parametric Friedman rank-sum test is employed to assess whether at least two algorithms exhibit significant differences in the observed performance values. The obtained $p$-value equals $7e-21$ showing that the differences among the algorithms are significant. However, we still don't know which pairs of algorithms are different. Therefore, the non-parametric Wilcoxon signed-rank test was performed. In a complementary way, to reduce the issue of Type I errors in multiple comparisons, the Bonferroni correction method was applied. Table 2 shows, on top, the $p$-values obtained with the pairwise Wilcoxon test and, on the bottom, the ones computed with the Bonferroni correction.

If we analyze the conclusions supported by the tests, based on the acceptance or rejection of the above hypotheses, we arrive at the following insights: cRAVE is largely outperformed by the other tested variants. In addition, $\text{cRAVE}_{CSP}$ underperforms compared to $\text{cRAVE}_{CSP-AD}$, $\text{cGRAVE}_{CSP}$, and $\text{cGRAVE}_{CSP-AD}$. The results achieved by $\text{cGRAVE}_{CSP}$ are not statistically different compared to $\text{cRAVE}_{CSP-AD}$ and $\text{cGRAVE}_{CSP-AD}$ (dark boxes in Table 2). However, it can be emphasized that $\text{cRAVE}_{CSP}$ lags behind $\text{cGRAVE}_{CSP}$ and similarly between $\text{cRAVE}_{CSP-AD}$ and $\text{cGRAVE}_{CSP-AD}$.

### 4.6 Visualisation

Figure 6 shows the best solution obtained by $\text{cGRAVE}_{CSP-AD}$ for each run. It illustrates the evolution of gene product concentration versus the time spent.

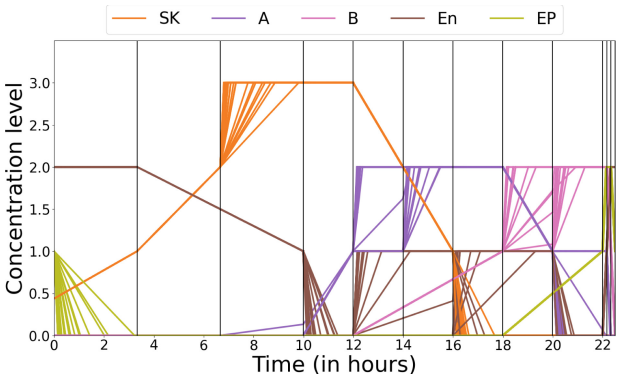



**Fig. 6.** Visualisation of the 30 solutions (one for each run) obtained by cGRAVE$_{CSP-AD}$ on the 5 genes hGRN identification problem. Black vertical lines illustrate the 12 different discrete states.

The graph type is modeled differently than Fig. 2b because of the number of dimensions, but both of them emphasize the same phenomenon: the evolution of concentration (in the y-axis) as a function of the time spent (in x-axis) for the different genes (different curves). This visual confirmation shows that the contributions proposed helped to exhibit solutions, each consistent with BK.

## 5 Conclusion

The contributions proposed in this work concern first a continuous version of the GRAVE heuristic. GRAVE uses the AMAF statistics of an ancestor node when the number of playouts is too low to have meaningful AMAF statistics on the considered node. The AMAF statistics considered are estimated thanks to a smoothing technique (Gaussian convolutions in our study case) as in the cRAVE approach. In addition, we have presented two additional generic improvements: (i) the action decomposition strategy, which allows having a finer-grained action selection step, and (ii) the constraints-based selective policy implying the construction of a module that automatically extracts constraints to reduce the action space by pruning actions before and during the search process. By analyzing the cumulative experiments on the problem of identifying celerity vectors in a hybrid GRN, the results show that cGRAVE combined with the presented contributions largely outperforms cRAVE. It also emerged that cGRAVE, the action decomposition strategy, and the constraints-based selective policy can independently be generic improvements of MCTS in the continuous domain.

The question of hGRN modeling addressed in this paper actually requires a set of solutions to help the biologist develop new hypotheses and design experiments. This multimodal issue has already been addressed in an alternative approach [24]. We plan to develop a new version of cGRAVE$_{CSP-AD}$ that can find diverse plans to the same problem in a single run. We believe that such a modification could be useful in other problems, both in the discrete and continuous settings.

**Acknowledgments.** This work was supported by the French government, through the UCA$^{JEDI}$ Investments in the Future project managed by the National Research Agency (ANR) under reference number ANR-15-IDEX-01. The authors are grateful to the OPAL infrastructure and the Université Côte d'Azur's Center for High-Performance Computing for providing resources and support.

**Disclosure of Interests.** The authors have no competing interests to declare that are relevant to the content of this article.

## References

1. Auer, P., Cesa-Bianchi, N., Fischer, P.: Finite-time analysis of the multiarmed bandit problem. Mach. Learn. 235–256 (2002)
2. Behaegel, J., Comet, J.P., Bernot, G., Cornillon, E., Delaunay, F.: A hybrid model of cell cycle in mammals. In: 6th International Conference on Computational Systems-Biology and Bioinformatics (2015). https://doi.org/10.1142/S0219720016400011
3. Behaegel, J., Comet, J.P., Pelleau, M.: Identification of dynamic parameters for gene networks. In: Proceedings of the 30th IEEE International Conference on ICTAI (2018). https://doi.org/10.1109/ICTAI.2018.00028
4. Bouzy, B., Helmstetter, B.: Monte-Carlo go developments. In: Advances in Computer Games: Many Games, Many Challenges, pp. 159–174 (2004)
5. Browne, C.B., et al.: A survey of Monte Carlo tree search methods. IEEE Trans. Comput. Intell. AI Games 1–43 (2012). https://doi.org/10.1109/TCIAIG.2012.2186810
6. Brügmann, B.: Monte Carlo go. Technical report (1993)
7. Bubeck, S., Stoltz, G., Szepesvári, C., Munos, R.: Online optimization in X-armed bandits. In: NIPS (2008)
8. Cazenave, T.: Generalized rapid action value estimation. In: 24th International Joint Conference on Artificial Intelligence, pp. 754–760 (2015)
9. Cazenave, T.: Nested rollout policy adaptation with selective policies. In: Cazenave, T., Winands, M.H.M., Edelkamp, S., Schiffel, S., Thielscher, M., Togelius, J. (eds.) CGW/GIGA -2016. CCIS, vol. 705, pp. 44–56. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-57969-6_4
10. Chaslot, G.M.J., Winands, M.H., Herik, H.J.V.D., Uiterwijk, J.W., Bouzy, B.: Progressive strategies for Monte-Carlo tree search. New Math. Nat. Comput. **4**, 343–357 (2008)
11. Couëtoux, A., Hoock, J.-B., Sokolovska, N., Teytaud, O., Bonnard, N.: Continuous upper confidence trees. In: Coello, C.A.C. (ed.) LION 2011. LNCS, vol. 6683, pp. 433–445. Springer, Heidelberg (2011). https://doi.org/10.1007/978-3-642-25566-3_32
12. Couëtoux, A., Milone, M., Brendel, M., Doghmen, H., Sebag, M., Teytaud, O.: Continuous rapid action value estimates. In: Asian Conference on Machine Learning, pp. 19–31 (2011)
13. Coulom, R.: Efficient selectivity and backup operators in Monte-Carlo tree search. In: van den Herik, H.J., Ciancarini, P., Donkers, H.H.L.M.J. (eds.) CG 2006. LNCS, vol. 4630, pp. 72–83. Springer, Heidelberg (2007). https://doi.org/10.1007/978-3-540-75538-8_7

14. Coulom, R.: Computing "elo ratings" of move patterns in the game of go. ICGA J. 198–208 (2007)
15. Finnsson, H., Björnsson, Y.: Simulation-based approach to general game playing. In: AAAI, pp. 259–264 (2008)
16. Gelly, S., Silver, D.: Combining online and offline knowledge in UCT. In: Proceedings of the 24th International Conference on Machine Learning (2007)
17. Gelly, S., Silver, D.: Monte-Carlo tree search and rapid action value estimation in computer go. Artif. Intell. **175**, 1856–1875 (2011)
18. Kim, B., Lee, K., Lim, S., Kaelbling, L., Lozano-Pérez, T.: Monte Carlo tree search in continuous spaces using voronoi optimistic optimization with regret bounds. In: Proceedings of the AAAI Conference on Artificial Intelligence, pp. 9916–9924 (2020). https://doi.org/10.1609/aaai.v34i06.6546
19. Kocsis, L., Szepesvári, C.: Bandit based Monte-Carlo planning. In: Fürnkranz, J., Scheffer, T., Spiliopoulou, M. (eds.) ECML 2006. LNCS (LNAI), vol. 4212, pp. 282–293. Springer, Heidelberg (2006). https://doi.org/10.1007/11871842_29
20. Lee, J., Jeon, W., Kim, G.H., Kim, K.E.: Monte-Carlo tree search in continuous action spaces with value gradients. In: Proceedings of the AAAI Conference on Artificial Intelligence, pp. 4561–4568 (2020). https://doi.org/10.1609/aaai.v34i04.5885
21. López-Ibáñez, M., Dubois-Lacoste, J., Pérez Cáceres, L., Stützle, T., Birattari, M.: The irace package: iterated racing for automatic algorithm configuration. Oper. Res. Perspect. **3**, 43–58 (2016). https://doi.org/10.1016/j.orp.2016.09.002
22. Mansley, C., Weinstein, A., Littman, M.: Sample-based planning for continuous action Markov decision processes. In: Proceedings of the International Conference on Automated Planning and Scheduling, pp. 335–338 (2011)
23. Méhat, J., Cazenave, T.: A parallel general game player. KI-künstliche Intelligenz **25**, 43–47 (2011). https://doi.org/10.1007/s13218-010-0083-6
24. Michelucci, R., Callegari, V., Comet, J.P., Pallez, D.: Cellular genetic algorithms for identifying variables in hybrid gene regulatory networks. In: Smith, S., Correia, J., Cintrano, C. (eds.) EvoApplications 2024. LNCS, vol. 14634, pp. 131–145. Springer, Cham (2024). https://doi.org/10.1007/978-3-031-56852-7_9
25. Yee, T., Lisỳ, V., Bowling, M.H., Kambhampati, S.: Monte Carlo tree search in continuous action spaces with execution uncertainty. In: Proceedings of the 25th IJCAI, pp. 690–697 (2016)
26. Świechowski, M., Godlewski, K., Sawicki, B., Mańdziuk, J.: Monte Carlo tree search: a review of recent modifications and applications. Artif. Intell. Rev. (2023). https://doi.org/10.1007/s10462-022-10228-y

# Satellite Resource Scheduling: Compaction Strategies for Genetic Algorithm Schedulers

Darrell Whitley[1(✉)], Ozeas Quevedo de Carvalho[1], Mark Roberts[2], Vivint Shetty[2], and Piyabutra Jampathom[2]

[1] Colorado State University, Fort Collins, USA
darrell.whitley@gmail.com
[2] Naval Research Lab, Washington, DC, USA

**Abstract.** The United States Naval Research Laboratory is currently using permutation-based genetic algorithms for large-scale satellite resource scheduling. This is a real-world, deployed application. The permutations must be mapped to a Gantt chart representing the final schedule. How this mapping is done can have a significant impact on the ability of the search algorithm to discover high-quality solutions. We present new work that uses compaction strategies in combination with genetic algorithms to construct less fragmented schedules. A schedule with "fewer holes" should also translate into better resource utilization. We show that this is indeed the case. This work is impactful because this strategy can be used to improve all genetic algorithm schedulers.

## 1 Introduction

Genetic algorithm schedulers have a successful track record of fielded applications [1,14,20,22,27]. A genetic algorithm scheduler is currently in use by the United States Navy for scheduling multi-resource satellites. The scheduler uses a permutation representation. The genetic algorithm generates a permutation, and a schedule builder maps the permutation to a schedule in the form of a Gantt chart. This separates the optimization problem from the representation problem, enabling a clearer understanding of the impact of optimization strategies and solution representations on the scheduling outcomes.

The permutation representation acts as a priority queue for the schedule builder. Tasks are drawn one at a time from the permutation in order and placed in the schedule. This form of "Resource Scheduling" attempts to place as many tasks as possible without conflict (or with minimal conflict) on some target set of resources. Early in the schedule construction process, almost all tasks are scheduled without conflict. However, if the target resource is oversubscribed, conflicts arise as more and more tasks are placed into the schedule. Our schedulers use two different strategies when there is a conflict.

In the first strategy, when a conflict occurs, we immediately place the task in the schedule in a position that minimizes the overlap between tasks. This creates


M. Affenzeller et al. (Eds.): PPSN 2024, LNCS 15151, pp. 335–350, 2024.
https://doi.org/10.1007/978-3-031-70085-9_21

an opportunity to "repair" a schedule by allocating less than the requested time to the task. For example, if two tasks have a requested duration of 20 min each, but there is a 4-minute overlap, it may make sense to allocate 18 min to each task instead of bumping one of the tasks out of the schedule. This approach may require human intervention to repair a schedule. Minimizing **total overlap time** can produce a schedule that is easier to repair. For brevity, we will refer to this as minimizing **overlaps.**

In the second strategy, if a conflict occurs, the task is not immediately placed in the schedule; instead, the schedule builder proceeds to the next task in the permutation. Here, the priority is to minimize the number of conflicts. All tasks that were ignored due to conflict are later placed in the schedule to also minimize overlaps. In general, this will produce a schedule with fewer total conflicts, but with larger overlaps. One disadvantage is that "large tasks" that require more time are more likely to be in conflict and more likely to result in a larger overlap.

This paper asks how we can build a schedule that is as compact as possible by removing "gaps" in the schedule. This is similar to the problem of computer memory allocation. We would also like to avoid fragmentation, i.e., we wish to prevent the creation of small blocks of time that cannot be scheduled. A less fragmented schedule should also result in fewer conflicts and less total overlapped time when conflicts occur. We introduce different mapping policies designed to avoid schedule fragmentation.

We also demonstrate that conflict minimization and overlap minimization are strongly correlated when resources are not heavily oversubscribed. When there is a small number of conflicts, solutions which minimize overlaps also minimize conflicts. However, when resources are heavily oversubscribed, solutions that minimize conflicts and solutions that minimize overlaps become anti-correlated.

## 2 Satellite Mission Scheduling

The US Naval Research Laboratory (NRL) offers satellite services through the Virtual Mission Operations Center (VMOC), an integrated software suite for satellite mission planning, scheduling, and monitoring. Satellite missions can perform different tasks, like communication, earth observation, navigation, and positioning. The VMOC system schedules these tasks using resources distributed across a constellation of satellites. A "resource" is some computational, sensing, or communication capability installed on one or more satellites. Scheduling is non-preemptive. The nature of the task determines its duration and required resource type. Other task attributes can also affect schedule decision-making.

As satellites orbit the Earth, "lines of sight" are created between the resources and targets. Depending on satellite coverage, a target on Earth may be periodically visible to several resources. Thus, a single task may have multiple scheduling opportunities on multiple different satellite resources at different times. A **visibility window** is a contiguous time interval during which a target is visible to a resource, i.e., when there is a line of sight between them.

While task duration is typically constant, the duration of the visibility window depends on satellite orbital mechanics. The visibility window may exactly

fit the task duration, or it may be longer. Low earth orbit (LEO) satellites have short orbital periods, varying from 86 to 128 min, which enables them to complete 11 to 16 orbits daily. Medium earth orbit (MEO) satellites have longer orbital periods, varying from 668 to 777 min, resulting in fewer orbits per day than LEO satellites. Thus, a target may be visible to LEO resources during multiple short windows a day while visible to MEO resources during fewer but longer windows. There also exists high earth orbit satellites.

A target is not uniformly visible to the resource during the entire visibility window. For example, the slant distance and the off-nadir angle between the resource and the target vary as the satellite moves along its orbit. For imagery tasks, these variables affect the spatial resolution and geometric characteristics of the collected image. For communication tasks, they affect signal quality and delay. Many other factors influence task success, such as resource configuration, swath (the band of the Earth visible to an orbiting satellite), and weather. Since we want to focus specifically on compact schedule construction strategies, these constraints are outside the scope of our current analysis.

It is also necessary to schedule set-up time between consecutive tasks on the same resource to reconfigure it for target acquisition. We assume that each task includes its required set-up time.

### 2.1 Schedule Representation

One way to represent a schedule is a Gantt chart, a set of intervals indicating when tasks start and finish on each resource timeline. However, it is sometimes more efficient to optimize an indirect representation, like a permutation of tasks, rather than directly optimizing a schedule [6,7,24,25]. Schedule representations can be complex, with intricate search spaces, making it challenging to find optimal solutions. When optimizing permutations for schedules, the focus is on the ordering relation among tasks rather than their specific timing.

Since permutations do not encode timing information, a greedy schedule builder maps a permutation to a schedule according to a specified policy. By optimizing a permutation and then mapping it to a schedule, we decouple the optimization problem from the representation problem. Note that a deterministic mapping policy is a surjective-only function. While each permutation uniquely maps to a single schedule, different permutations may map to the same schedule, which forms plateaus in the search space [2].

### 2.2 Problem Formulation and Objective Functions

Given a set of task requests, the scheduling problem is to find, for each task, the visibility window and start time that optimize an objective function. The start time and duration determine the execution window where a task is ultimately scheduled. These concepts are depicted in Fig. 1. The visibility window is the interval $[EST, LFT]$, where $EST$ is the *earliest start time*, and $LFT$ is the *latest finish time*. The execution window is the interval $[t, t + d]$, where $t$ is the

start time, and $d$ is the duration. The task must lie within its visibility window, i.e., in the interval $[EST, LFT]$.

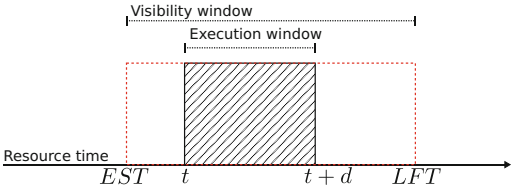



**Fig. 1.** Time flows from left to right. The visibility window is the interval $[EST, LFT]$. The execution window is the interval $[t, t + d]$.

Let $T = \{1, 2, ..., n\}$ be the tasks, and $R = \{1, 2, ..., m\}$ be a set of resources. Each task $i$ is associated with a duration $d_i$ as well as a subset of alternative scheduling options, where each alternative placement is denoted by a 5-tuple $(i, r, d_i, EST, LFT)$ indicating that task $i$ can be scheduled in a window on resource $r$ with duration $d_i$ with some earliest start time $(EST)$ and the latest finish time $(LFT)$. It is possible that a task could be placed on the same resource at different points in time, for example on a low orbit satellite which has multiple orbits per day.

Consider a schedule denoted by $S$. For each task in $S$ there is a schedule **window** for task $i$ which can be denoted by a 4-tuple $w_i = (i, r, a_i, d_i)$ which indicates that task $i$ has been placed on resource $r$ with an assigned start time $a_i$ and duration $d_i$.

A schedule is defined as **complete** if all tasks in set T are placed in the schedule. A complete schedule may have conflicts where two or more tasks are scheduled on a resource at the same time.

Construct a matrix $O$ for each schedule $S$ such that $o_{i,j}$ records the overlap in the window assigned to task $i$ and $j$ on the same resource $r$. If there is no conflict between task $i$ and $j$ in schedule $S$ then $o_{i,j} = 0$.

Since we will use a permutation representation, $\pi = < \pi_1, \pi_2, ..., \pi_n > | \ \pi : T \rightarrow T$, we also can refer to a task by its position in the permutation. Thus, $\pi_q = i$, if task $i$ is placed in the $q^{th}$ position in the permutation representation. Given a deterministic schedule builder, each permutation uniquely maps to one schedule.

Let cost $c_t$ denote the total overlap time in a complete schedule $S$:

$$c_t = \sum_i \sum_j o_{\pi_i, \pi_j} \quad \text{subject to} \quad i < j \tag{1}$$

Obviously, we only need to check for overlaps when task $\pi_j$ is placed in the schedule. In simple problems there can exist complete schedules that have zero overlap. But in many real world problems this is not possible.

We can also create incomplete schedules where we only place a task in the schedule if it can be placed without conflict. A different objective is to maximize the number of tasks that can be scheduled without conflict, which corresponds to minimizing the number of conflicts.

Given the permutation of tasks $\pi = < \pi_1, \pi_2, ..., \pi_n > | \ \pi : T \rightarrow T$, and a binary vector $\mathbf{m} = [m_1, m_2, ..., m_n]$, where $m_i = 0$ if the task $\pi_i$ was allocated without conflict, and $m_i = 1$ otherwise, we can define the number of conflicts $c_n$ as the number of 1 s in vector $\mathbf{m}$:

$$c_n = \|\mathbf{m}\|_1 \tag{2}$$

### 2.3 Related Work

The problem discussed in this work falls under the broader class of satellite range scheduling problems (SRS), known to be NP-complete [1]. Researchers have proposed various solutions to SRS applications over the last three decades. Solutions can be grouped into exact and non-exact methods. Exact methods are primarily Linear Programming [5,13] and Dynamic Programming solutions [15, 17], while non-exact methods comprise heuristic [3,11], metaheuristic [26,27], and machine-learning solutions [4,8].

In this work, we are particularly interested in the choice of schedule representation. Radcliffe et al. [18] indicate that the choice of representation can significantly impact the effectiveness of genetic algorithms in solving complex problems. They strongly suggest a preference for domain-independent representations. Despite this, there is a gap in the literature concerning how the choice of representation affects the performance of genetic algorithm schedulers.

Among metaheuristics, genetic algorithms (GA) have been extensively applied to satellite scheduling [12]. Barbulescu et al. [1] successfully used genetic algorithms with permutation-based representations to schedule the US Air Force Satellite Control Network (AFSCN). The authors proposed two mapping policies, similar to policies in this paper, to convert task permutations into schedules.

More recently, Whitley et al. [23] proposed 1-Pass and 2-Pass strategies, demonstrating that the mapping policy strongly influences the optimization results. Other recent studies use direct or *ad hoc* indirect representations, but concentrate most of the discussion on the optimization methods [14,16,26,27].

### 2.4 Scheduling Policies

The permutation functions as a priority queue. A schedule builder traverses the permutation and decides the visibility window and the start time of each task. However, building a schedule involves additional decisions. Suppose the scheduler cannot schedule a task without creating a conflict. Should the task be bumped out of the schedule, or should it be skipped and scheduled later? If the overlap time is short, should the task be placed in the schedule regardless?

We start by exploring two common scheduling strategies [1,23]. Consider the permutation $< 4, 7, 3, 6, 2, 1, 5 >$, where each number uniquely identifies

a task, and each task has an associated set of visibility windows. A task is *conflict-free* if it can be scheduled without conflict. Otherwise, there exists one or more *conflicting* tasks. Note that whether a task is conflict-free depends on its position in the permutation and how oversubscribed the problem is. For example, any task in the first position is conflict-free because the schedule is still empty. Conversely, the last task in the permutation is more likely to cause a conflict because all previous tasks are already in the schedule.

Every task request includes an ordering of windows where the task might be scheduled. We assume that the first requested window is the most desirable, and our basic policies give priority to scheduling in these windows.

The 1-Pass and 2-Pass scheduling strategies were briefly introduced in the introduction. Using a 1-Pass strategy, when a conflict occurs, we nevertheless place the task in the schedule in a position that minimizes the overlap between tasks. This creates an opportunity to "repair" a schedule by allocating less than the total requested time for a task. This approach is known to be used by some professional human schedulers working in satellite scheduling [2,19].

Using a 2-Pass strategy, if a conflict occurs, the task is initially dropped out of the schedule. After a first complete pass, the subsets of conflict-free and conflicting tasks are determined. But we would also like to place all the tasks in the schedule, even if this causes overlaps. So, a second pass is made over the set of conflicting tasks (in the same original permutation order), and they are placed in the schedule while trying to minimize overlap time.

We next discuss six different scheduling policies, each based on one of these strategies (1-Pass or 2-Pass), with distinct approaches to address task placement.

**Policy First-L1** is a "first fit" policy using a 1-Pass scheduler that places the task in the leftmost position in the scheduling window while using a single pass over the permutation (First-L1: First-fit, Leftmost, 1-Pass). If the task is conflict-free, the scheduler places it as early as possible (i.e., leftmost) in the first requested window or the first available window. Since this is a 1-Pass scheduler, if it is a conflicting task, the scheduler places it on a resource using the window and start time that minimizes overlap time. Let us consider the example permutation. Suppose the scheduler places tasks 4, 7, and 3 without conflict. Assume the scheduler cannot place task 6 without creating conflict with tasks 4, 7, or 3. The scheduler places task 6 using the window and starts time that minimizes overlap time. This policy is repeated until all tasks are scheduled.

Because this policy uses a "leftmost" (as early as possible) placement of tasks, this may or may not result in fragmentation. For example, in Fig. 2, the example "Case 1" is such that the leftmost placement of Task 3 (between Tasks 4 and 7) results in two gaps that fragment the schedule. This does not happen if the Task 3 window intersects with other tasks in the schedule. In Fig. 3 (leftmost), we see a case where Task 3 is placed leftmost, and now Task 3 and Task 4 are adjacent. We refer to this as "sticking to the left".

**Policy First-L2** schedules tasks doing two consecutive passes: one pass for conflict-free tasks and another for conflicting tasks. Otherwise, it is like the First-L1 policy that schedules in the first available window in the leftmost position

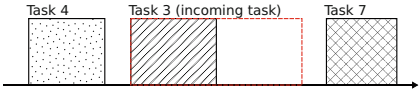



**Fig. 2.** Case 1 - No sticking. The incoming task window does not intersect with previous tasks, so it is impossible to stick it to tasks 4 or 7. Whenever the start time, placing task 3 will create two time gaps.


**Fig. 3.** Case 2 and 3 - Stick to the left (leftmost), and sticking to the right (rightmost). The scheduler sticks the incoming task to the one that intersects with the window on the left (task 4) or on the right (task 7).

(First-L2: a First-fit, Leftmost, 2-Pass scheduler). Let us consider the example permutation again: $< 4, 7, 3, 6, 2, 1, 5 >$ . Suppose the scheduler places tasks 4, 7, 3, and 2 without conflict in the first pass, skipping tasks 6, 1, and 5, the tasks that create a conflict. The scheduler then does a second pass over the unscheduled tasks $< 6, 1, 5 >$ in the same order they appear in the permutation, scheduling them in a location that minimizes total overlap time.

The fundamental difference between these policies is that Policy First-L1 (1-Pass) strictly interprets the permutation as a single queue, while Policy First-L2 (2-Pass) decomposes the permutation in two queues. Otherwise, they behave the same, either scheduling conflict-free tasks as early as possible in the first fit schedule location, or minimizing the overlap time of conflicting tasks.

### 2.5 Schedule Compaction

We present two additional policies that aim to produce less fragmented schedules through the use of some degree of compaction.

**Policy First-LR1** is the same as Policy First-L1, except it makes an extra effort to prevent gaps in the schedule (First-LR1: First-fit, Leftmost and Rightmost, 1-Pass). It is also a 1-Pass scheduler. First, the schedule finds the first available window where the task can be placed. Then the scheduler tries to place the task in the leftmost position. If the task sticks to the left, (Case 2, Fig. 3), then the task is placed in the leftmost position. But assume the leftmost placement of a task yields a gap, as shown in Fig. 2. The scheduler then looks at the rightmost placement of the task. Sometimes, the rightmost placement will yield a placement adjacent to a previously placed job (the task sticks to the right). An example of this is in Fig. 3 (rightmost). If a task does not stick to the right or left, it is placed in the leftmost position by default.

**Policy First-LR2** is the same as Policy First-L2 except it makes an extra effort to prevent gaps in the schedule. It is a 2-Pass scheduler. This policy checks

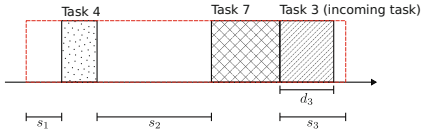



**Fig. 4.** The intersection of Task 3 window (incoming task) with the free time in the schedule results in the sub-windows $s_2 > s_3 > d_3 > s1$, where $d_3$ is the duration of Task 3. Different policies will place Task 3 in different locations.

**Table 1.** Scheduling policies

| Policy | Pass | Compaction | Window fit |
|---|---|---|---|
| First-L1 | 1-Pass | no | first |
| First-L2 | 2-Pass | no | first |
| First-LR1 | 1-Pass | yes | first |
| First-LR2 | 2-Pass | yes | first |
| Best-LR1 | 1-Pass | yes | best |
| Best-LR2 | 2-Pass | yes | best |

both left and right placement in order to determine if one of these placements reduces fragmentation (i.e., if the task sticks to a previously placed task).

There may be more intricate cases. In Fig. 4, for instance, the intersection of the task window with the free time results in one or more sub-windows; in this case, the sub-windows $s_1$, $s_2$, and $s_3$. Given task 3 duration $d_3$, assume that $s_2 > s_3 > d_3 > s_1$, i.e., task 3 fits in the sub-windows $s_2$ and $s_3$, but not in $s_1$. All "first fit" policies would start the task as early as possible in the sub-window $s_2$ (its first fit), assuming no distinction between windows and sub-windows.

### 2.6 Best Fit Policies

The First-LR1 and First-LR2 policies consider only the "first fit", or first available window free from conflict. We introduce two more policies that search for the best-fit scheduling window to further reduce fragmentation.

The Best-LR1 and Best-LR2 policy uses a "best fit" strategy that places a conflict-free task in the smallest window available, while checking both leftmost and rightmost placements. Consider the example in Fig. 4 again. Assume the task could be placed in sub-window $s_2$ or in sub-window $s_3$. The task is placed in sub-window $s_3$ in the leftmost position if this yields a compact result. Otherwise, Best-LR1 uses a 1-Pass strategy and Best-LR1 uses a 2-Pass strategy.

Table 1 summarizes the proposed mapping policies, showing how they relate in terms of number of passes (1-Pass or 2-Pass), presence of compaction strategies, and window fit.

### 2.7 Problem Generator

We developed a problem generator to create representative scheduling problems based on the VMOC scheduler. The problem generator takes three input parameters: the number and types of satellites (LEO, MEO), the number of tasks, and the task density. Task density determines how many scheduling alternatives a task has. The scheduling alternatives appear in order of preference in the dataset, with the first location being the top preference.

Time is measured in minutes, and the schedule horizon covers 1440 min (24 h). Task windows start and end on the same day. Visibility windows are uniformly sampled within predefined ranges, depending on satellite type. LEO windows range from 10 to 15 min, while MEO windows range from 60 to 120 min. The task duration is uniformly sampled within the window duration.

We generated nine instances with increasing numbers of tasks (200 to 1000 tasks) for a fixed set of 5 LEO and 10 MEO satellites. The larger the instance, the more oversubscribed the problem is.

When there are more than 400 tasks and only 15 satellite resources, the resulting problems are heavily oversubscribed. For 600 tasks, almost all solutions have more than 200 conflicts. Our schedules would be less conflicted if some of the MEO satellite requests were scaled back to a range of 40 to 90 min. Another solution to oversubscribed resources is to increase the number of resources.

## 3 Experimental Design

We used an implementation of the Standard Generational Genetic Algorithm [9,10] but without a mutation operator and with tournament selection, with a tournament size $k = 3$. Our prior experience is that mutation is not necessary since permutation crossover operators do not "transmit alleles" and thus are inherently noisy. For each experiment configuration, we ran 30 trials with a limit of 50,000 evaluations each.

Solutions are recombined using Syswerda's Order Crossover [21]. This is the only permutation operator we are aware of specifically designed to be sensitive to "order" and we have repeatly found it to be more effective than other crossover operators for this class of scheduling problems [1,23,25].

Syswerda introduced two permutation crossover operators: an "order" crossover and a "position" crossover operator. However, these operators inherit by both order and position and are identical when correctly parameterized. Assume we are given two permutations as follows:

```
Parent 1: A B C D E F G H I J
Parent 2: D I B G E A J F C H
```

We inherit 5 of the 10 elements by *position* from Parent 2. The offspring after inheriting these positions will be:

```
Offspring: # # B # E A # # C H
```

The remaining elements are inherited from Parent 1 in the *order* in which they appear: `D F G I J`

`Offspring: D F B G E A I J C H`

Note that we could have first selected elements `D I G J F` from Parent 2 and re-ordered them based on their order in Parent 1. But if the number of positions selected by position crossover is the same as the number of positions utilized by order crossover, the results are identical in expectation.

### 3.1 Scheduling Results

We ran a large number of experiments with the six different scheduling policies and with population sizes of 256, 512, and 1024. When we used the "overlap" evaluation function, the best results were *always* produced by a population size of 1024. When we used the "conflicts" evaluation function, the smallest number of conflicts was also produced by a population size of 1024.

Table 2 presents results for problems with $n = 200$, $n = 300$, $n = 400$, $n = 500$ and $n = 600$ tasks for population size 1024. We also ran experiments with up to $n = 1000$ tasks, but these problems were extremely oversubscribed. All of the results in Table 2 are single-objective results.

Because we care about both the number of conflicts and the sum of overlaps, we ran experiments using two different evaluation functions with the genetic algorithm. The first evaluation function minimizes conflicts, and the second evaluation function minimizes the sum of overlap time. Previous satellite schedulers, specifically the Air Force Satellite Control Network (AFSCN) scheduler, used "overlap" as the evaluation function. Since satellites are valuable resources, human schedulers were repairing the schedule, trying to fit in (almost) every task by giving the conflicting tasks (a little) less than the requested time [2,19].

The data includes 15 satellite resources distributed over a mixture of low-orbit and medium-orbit satellites. In general, 1-Pass schedulers yield lower overlap, while 2-Pass schedulers yield lower conflict counts. This pattern holds regardless of whether the evaluation function is "conflicts" or "overlaps". Thus, the schedule builder can also influence the type of solutions that are generated.

Inspection of Table 2 shows there is some trade-off between using only conflicts or only overlaps as the evaluation function. But particularly when scheduling 200 tasks, it is generally the case that when a low overlap time was achieved, the number of conflicts was also low. For 200 tasks, there was little variation in the number of conflicts when the evaluation function optimized conflicts (the results ranged from $6.3 \pm 1.2$ to $7.3 \pm 1.4$). It is possible to minimize both conflicts and overlaps for 200 tasks.

When optimizing overlaps, for the 200-task instance, the conflicts range from 7 to 11. The Best-LR1 and Best-LR2 policies using "overlap" as the evaluation function provide an excellent trade-off: the number of conflicts is 7 or 8, and the overlap time varies from 150 to 156 min. When "conflicts" is used as the evaluation function, the overlap time doubles (to more than 310 min on average

**Table 2.** Scheduling results for different policies, and different numbers of tasks. Results are also given when the fitness function is "conflicts" versus "overlap." The population is 1024. The sample size is 30.

| Tasks | Policy | Fitness=Conflicts | | Fitness=Overlap | |
|---|---|---|---|---|---|
| | | Conflicts | Overlap | Conflicts | Overlap |
| 200 | First-L1 | 7.3 ± 1.4 | 338 ± 68 | 10.2 ± 1.1 | 153 ± 10 |
| 200 | First-LR1 | 7.1 ± 1.7 | 340 ± 82 | 10.6 ± 1.8 | 155 ± 22 |
| 200 | Best-LR1 | **6.3 ± 1.2** | **310 ± 57** | 8.6 ± 2.8 | **150 ± 33** |
| 200 | First-L2 | 6.9 ± 1.3 | 367 ± 72 | 10.2 ± 2.3 | 172 ± 27 |
| 200 | First-LR2 | 6.9 ± 0.8 | 369 ± 49 | 10.0 ± 1.9 | 164 ± 25 |
| 200 | Best-LR2 | 7.3 ± 1.7 | 390 ± 87 | **7.5 ± 2.0** | 156 ± 34 |
| 300 | First-L1 | 45 ± 5.3 | 2371 ± 147 | 62 ± 3.5 | 1697 ± 78 |
| 300 | First-LR1 | 45 ± 5.6 | 2394 ± 146 | 59 ± 3.6 | 1677 ± 66 |
| 300 | Best-LR1 | 41 ± 3.9 | **2304 ± 129** | 56 ± 3.4 | **1661 ± 117** |
| 300 | First-L2 | 40 ± 3.2 | 2420 ± 132 | 50 ± 2.9 | 1967 ± 115 |
| 300 | First-LR2 | 39 ± 2.5 | 2398 ± 111 | 50 ± 2.7 | 1892 ± 63 |
| 300 | Best-LR2 | **37 ± 2.3** | 2366 ± 126 | **47 ± 2.2** | 1827 ± 133 |
| 400 | First-L1 | 88 ± 6.4 | 5264 ± 86 | 118 ± 2.4 | 4385 ± 147 |
| 400 | First-LR1 | 89 ± 7.4 | **5231 ± 92** | 116 ± 2.8 | **4322 ± 103** |
| 400 | Best-LR1 | 85 ± 6.0 | 5239 ± 94 | 110 ± 3.5 | 4388 ± 152 |
| 400 | First-L2 | 79 ± 3.5 | 5383 ± 113 | 99 ± 2.3 | 4741 ± 103 |
| 400 | First-LR2 | 79 ± 4.2 | 5361 ± 101 | 97 ± 3.3 | 4681 ± 100 |
| 400 | Best-LR2 | **77 ± 4.6** | 5368 ± 118 | **92.3 ± 1.7** | 4655 ± 145 |
| 500 | First-L1 | 140 ± 7.0 | **8690 ± 72** | 192 ± 3.6 | **7875 ± 49** |
| 500 | First-LR1 | 140 ± 8.4 | 8707 ± 65 | 188 ± 4.0 | 7895 ± 96 |
| 500 | Best-LR1 | 138 ± 6.7 | 8761 ± 81 | 186 ± 3.5 | 7961 ± 116 |
| 500 | First-L2 | **125 ± 2.7** | 8897 ± 73 | 153 ± 2.4 | 8273 ± 114 |
| 500 | First-LR2 | 127 ± 5.3 | 8911 ± 81 | 151 ± 2.9 | 8293 ± 84 |
| 500 | Best-LR2 | 126 ± 6.2 | 9007 ± 110 | **149 ± 2.9** | 8312 ± 155 |
| 600 | First-L1 | 222 ± 5.8 | 14932 ± 60 | 296 ± 4.9 | **14300 ± 36** |
| 600 | First-LR1 | 224 ± 8.7 | 14922 ± 58 | 297 ± 4.6 | 14317 ± 35 |
| 600 | Best-LR1 | 223 ± 13.5 | **14906 ± 54** | 293 ± 3.6 | 14347 ± 86 |
| 600 | First-L2 | 204 ± 4.0 | 15116 ± 64 | 239 ± 2.8 | 14582 ± 68 |
| 600 | First-LR2 | **202 ± 5.4** | 15147 ± 61 | 236 ± 3.7 | 14571 ± 66 |
| 600 | Best-LR2 | 202 ± 6.6 | 15194 ± 79 | **234 ± 3.4** | 14634 ± 117 |

for all algorithm configurations). Although the system is somewhat oversubscribed with 200 tasks, a similar number of conflicts was reported in real-world AFSCN satellite scheduling data (with approximately 2 to 5 conflicts per 100 tasks scheduled) [1].

The results in Table 2 trend in the same direction when there are 300 tasks. But there is now more of a trade-off. The Best-LR1 and Best-LR2 policies using compaction strategies still produce the best overall results, but the 1-pass results (Best-LR1) do a significantly better job of minimizing overlap time, and the 2-pass results (Best-LR2) do a significantly better job of minimizing conflicts.

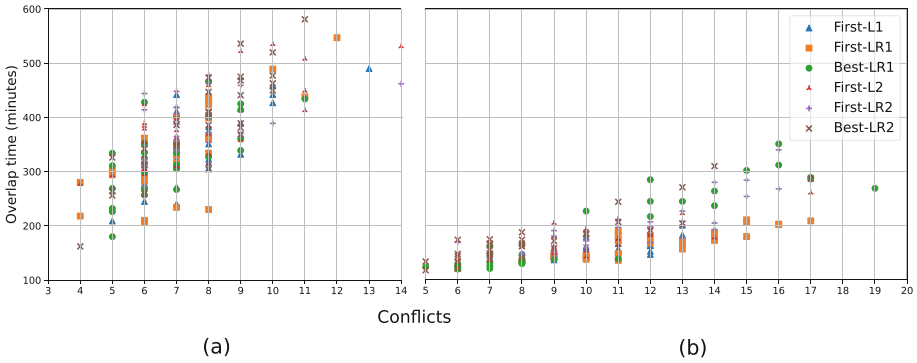



**Fig. 5.** 200 tasks. a) Minimizing Conflicts. b) Minimizing Overlaps. Note the x-axis for plot (a) is from 3 to 14, and the axis for plot (b) is from 5 to 20. In each case, there is a single dominating solution produced by Best-LR2.

The Best-LR2 policy using **overlap** as the evaluation function provides a useful trade-off: the average number of conflicts is 47, and the overlap time is 1827 min. That means 15.7% of all tasks are in conflict. If 47 conflicting tasks have 1827 min of overlap, then each conflicting task has an average overlap of 38 min. The Best-LR2 policy using number of **conflicts** as the evaluation function produced the lowest number of conflicts, with an average of 37. But the average overlap time jumped to 2366 min. Assuming we have a schedule with 37 conflicting tasks with an overlap of 2366 min, then each conflicting task has an average overlap of 64 min. This is an extremely high level of overlap in the schedule.

### 3.2 A Pareto Perspective

Figure 5 shows results for instance size 200 when minimizing conflicts and when minimizing overlap. We present these results in a bi-objective fashion, with **conflicts** on the $x$-axis and **overlap** on the $y$-axis. However, we did not use bi-objective methods for the results in Table 2, and it appears we do not need multi-objective methods as long as the number of conflicts and the overlap time are strongly correlated.

The labels (First-L1, First-L2, First-LR1, First-LR2, Best-LR1, Best-LR2) can be found in Table 1. Clearly, there is a correlation between the number of conflicts and the total overlap time for the 200-task problem in our study. Solutions that have minimal overlap tend to also have minimal numbers of conflicts. However, the impact of the different evaluation functions is also clear. When the evaluation function is **conflicts**, the best solutions reduced the number of conflicts to 4 tasks; when the fitness function is **overlap**, the best solutions reduced the number of conflicts to 5 tasks. In this case, there is not a large difference. But the difference in terms of overlap is much greater. When the evaluation function is **overlap**, there exist solutions where the number of conflicts is 5 and

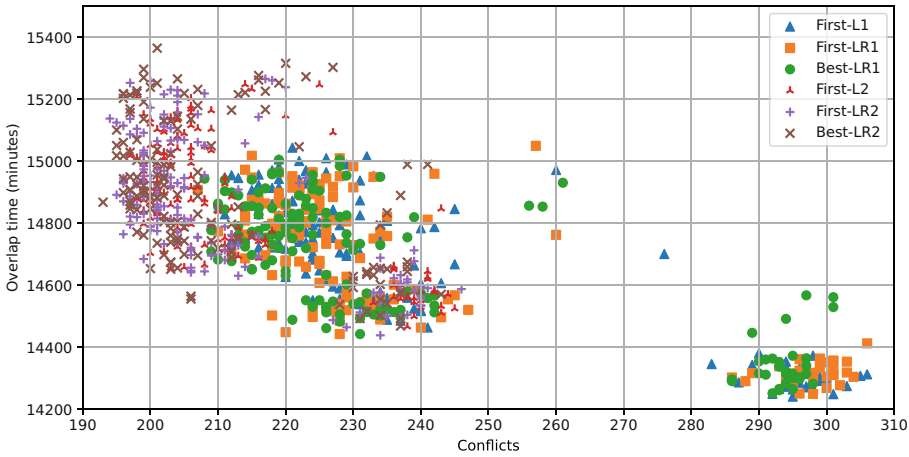



**Fig. 6.** 600 tasks. We merged the best results using both **overlaps** and **conficts** for all fitness configurations to better show the spread of the full data set. We now see the emergence of a Pareto front for this heavily oversubscribed problem. All of the results with more than 280 conflicts occur when minimizing overlap. All of the results with less than 230 conflicts occur when minimizing conflicts. But solutions with low conflicts have high overlaps, and solutions with low overlap have high conflicts.

the overlap is less than 125 min. But when the fitness function is **conflicts**, there are only two solutions below 200 min and none below 150 min.

From a bi-objective point of view, we believe that minimizing overlap yields the best overall results and produces more repairable schedules by taking advantage of small overlaps. But again, this is for less constrained problems, with 200 (and perhaps 300) tasks.

Figure 6 shows bi-objective results for instance size 600 when minimizing conflicts and when minimizing overlap. In this case, the scheduling problem is much more oversubscribed, and the results are now anti-correlated. The problem of selecting a schedule is also much more difficult.

The different algorithms in Table 1 diverge depending on the evaluation function. The 2-Pass algorithms consistently yield fewer conflicts, often reducing the number of conflict below 210. Note that this still represents $210/600 = 35\%$ of the tasks in conflict. Many of the tasks scheduled without conflict must also be the smaller LEO requests.

When the objective is **overlap** ($n = 600$), we can see that it is possible to reduce overlap to less than 14400 min. However, the number of conflicts dramatically increase, in some cases up to 300 of the 600 tasks.

Many of the best results in terms of minimizing conflicts are also produced by the BEST-LR2 placement strategy. The best results in terms of minimizing overlap are also produced by the BEST-LR1 placement strategy, but the FIRST-LR1 strategy is also often competitive. It is extremely interesting to observe that

different schedule building strategies yield results that occupy different parts of the Pareto front.

Given the configuration of our generator with 15 satellite resources, it is reasonable to conclude that 600 task requests are simply too many tasks: a conflict rate above $200/600 = 33.3\%$ is unacceptably high. Indeed, the problems with 600 tasks are much more constrained than the real-world problems we have seen in the literature for any resource scheduling problem. Nevertheless looking at such problems reveals that as satellite resources become more constrained a Pareto front emerges and that multi-objective methods may be required to explore the trade-off of optimizing conflicts while also minimimzing task overlap.

## 4 Conclusions

This paper evaluates genetic algorithm schedulers that utilize a permutation-based representation. The GA scheduler uses a schedule builder to convert a permutation into a schedule in the form of a Gantt chart.

We show that the use of compaction methods can result in schedules that are less fragmented, which in turns results in fewer conflicts and less overlap time. This effect is somewhat independent of the evaluation function. However the results are sensitive to the degree to which resources are oversubscribed. These kinds of trends are both fundamental and important for resource scheduling. Our work exposes biases that have not been carefully documented, despite over 30 years of research in this area.

When scheduling problems are not too oversubscribed, it is possible to minimize both the number of conflicts and the overlaps of the conflicted tasks. Our results for 200 and 300 tasks are closer to real-world scenarios than those with a larger number of tasks. Our data shows that our two objectives, minimizing conflicts and overlap, are correlated for problems that are not too oversubscribed. However, it seems better to use "overlap time" as the evaluation function since minimizing overlap also creates pressure to reduce conflicts.

The use of a 1-Pass or a 2-Pass strategy produces a significant effect on algorithm performance, no matter whether the objective is **conflicts** or **overlap**. This means the design of the scheduler builder can have a significant impact on solution quality, independent of the evaluation function.

## References

1. Barbulescu, L., Watson, J., Whitley, D., Howe, A.: Scheduling space–ground communications for the air force satellite control network. J. Sched. **7**(1), 7–34 (2004). https://doi.org/10.1023/B:JOSH.0000013053.32600.3c
2. Barbulescu, L., Howe, A.E., Whitley, L.D., Roberts, M.: Understanding algorithm performance on an oversubscribed scheduling application. J. Artif. Intell. Res. **27**, 577–615 (2006)
3. Barbulescu, L., Whitley, L.D., Howe, A.E.: Leap before you look: an effective strategy in an oversubscribed scheduling problem. In: Proceedings of the 19th conference on Artificial Intelligence, pp. 143–148 (2004)

4. Chen, H., Luo, Z., Peng, S., Wu, J., Li, J.: HiPGen: an approach for fast generation of multi-satellite observation plans via a hierarchical multi-channel transformer network. Adv. Space Res. **69**(8), 3103–3116 (2022). https://doi.org/10.1016/j.asr.2022.01.037
5. Chen, X., Reinelt, G., Dai, G., Spitz, A.: A mixed integer linear programming model for multi-satellite scheduling. Eur. J. Oper. Res. **275**(2), 694–707 (2019). https://doi.org/10.1016/j.ejor.2018.11.058
6. Davis, L.: Handbook of Genetic Algorithms. Van Nostrand Reinhold, New York (1991)
7. Davis, L.: Job shop scheduling with genetic algorithms. In: Grefenstette, J. (ed.) Int'l. Conf. on GAs and Their Applications, pp. 136–140 (1985)
8. Goh, E., Venkataram, H.S., Hoffmann, M., Johnston, M.D., Wilson, B.: Scheduling the NASA deep space network with deep reinforcement learning. In: 2021 IEEE Aerospace Conference (50100), pp. 1–10. IEEE, Big Sky, MT, USA (Mar 2021). https://doi.org/10.1109/AERO50100.2021.9438519, https://ieeexplore.ieee.org/document/9438519/
9. Goldberg, D.: Genetic Algorithms in Search, Optimization and Machine Learning. Addison-Wesley, Reading, MA (1989)
10. Holland, J.: Adaptation in Natural and Artificial Systems. University of Michigan Press (1975)
11. Kolici, V., Herrero, X., Xhafa, F., Barolli, L.: Local search and genetic algorithms for satellite scheduling problems. In: 2013 Eighth International Conference on Broadband and Wireless Computing, Communication and Applications, pp. 328–335. IEEE, Compiegne, France (Oct 2013). https://doi.org/10.1109/BWCCA.2013.58, http://ieeexplore.ieee.org/document/6690906/
12. Li, S., Yu, Q., Ding, H.: Reviews and prospects in satellite range scheduling problem. Auton. Intell. Syst. **3**(1), 9 (2023). https://doi.org/10.1007/s43684-023-00054-6
13. Linares, L., Vazquez, R., Perea, F., Galán-Vioque, J.: A mixed integer linear programming model for resolution of the antenna-satellite scheduling problem. In: IEEE Transactions on Aerospace and Electronic Systems, pp. 1–13 (2023). https://doi.org/10.1109/TAES.2023.3326422
14. Liu, Q., Li, S., Zhang, P., Liu, F.: Research on satellite communication resource scheduling method based on adaptive genetic algorithm. In: 2023 4th International Symposium on Computer Engineering and Intelligent Communications (ISCEIC), pp. 382–388. IEEE, Nanjing, China (Aug 2023). https://doi.org/10.1109/ISCEIC59030.2023.10271164, https://ieeexplore.ieee.org/document/10271164/
15. Liu, Z., Feng, Z., Ren, Z.: Route-reduction-based dynamic programming for large-scale satellite range scheduling problem. Eng. Optim. **51**(11), 1944–1964 (2019). https://doi.org/10.1080/0305215X.2018.1558445
16. Niu, X., Tang, H., Wu, L.: Satellite scheduling of large areal tasks for rapid response to natural disaster using a multi-objective genetic algorithm. Int. J. Disaster Risk Reduction **28**, 813–825 (2018). https://doi.org/10.1016/j.ijdrr.2018.02.013
17. Peng, G., Song, G., Xing, L., Gunawan, A., Vansteenwegen, P.: An exact algorithm for agile earth observation satellite scheduling with time-dependent profits. Comput. Oper. Res. **120**, 104946 (2020). https://doi.org/10.1016/j.cor.2020.104946
18. Radcliffe, N., Surry, P.: Fitness variance of formae and performance predictions. In: Whitley, D., Vose, M. (eds.) FOGA - 3, pp. 51–72. Morgan Kaufmann (1995)

19. Stottler, R., Richards, R.: Managed intelligent deconfliction and scheduling for satellite communication. In: 2018 IEEE Aerospace Conference, pp. 1–7. IEEE, Big Sky, MT (Mar 2018). https://doi.org/10.1109/AERO.2018.8396654, https://ieeexplore.ieee.org/document/8396654/
20. Syswerda, G.: Schedule optimization using genetic algorithms. In: Davis, L. (ed.) Handbook of Genetic Algorithms, pp. 332–349. Van Nostrand Reinhold, New York, NY (January (1991)
21. Syswerda, G., Palmucci, J.: The application of genetic algorithms to resource scheduling. In: Booker, L., Belew, R. (eds.) Proc. of the 4th Int'l. Conf. on GAs. Morgan Kaufmann (1991)
22. Tormos, P.., Lova, A.., Barber, F.., Ingolotti, L.., Abril, M.., Salido, M.. A..: A genetic algorithm for railway scheduling problems. In: Xhafa, F., Abraham, A. (eds.) Metaheuristics for scheduling in industrial and manufacturing applications, pp. 255–276. Springer Berlin Heidelberg, Berlin, Heidelberg (2008). https://doi.org/10.1007/978-3-540-78985-7_10
23. Whitley, D., Quevedo De Carvalho, O., Roberts, M., Shetty, V., Jampathom, P.: Scheduling multi-resource satellites using genetic algorithms and permutation based representations. In: Proceedings of the Genetic and Evolutionary Computation Conference, pp. 1473–1481. ACM (2023)
24. Whitley, D., Starkweather, T., Fuquay, D.: Scheduling problems and traveling salesmen: the genetic edge recombination operator. In: Schaffer, J.D. (ed.) Proc. of the 3rd Int'l. Conf. on GAs. Morgan Kaufmann (1989)
25. Whitley, L.D., Starkweather, T., Shaner, D.: The traveling salesman and sequence scheduling: quality solutions using genetic edge recombination. In: Davis, L. (ed.) Handbook of Genetic Algorithms, chap. 22, pp. 350–372. Van Nostrand Reinhold, New York (1991)
26. Xu, Y., Liu, X., He, R., Chen, Y.: Multi-satellite scheduling framework and algorithm for very large area observation. Acta Astronaut. **167**, 93–107 (2020). https://doi.org/10.1016/j.actaastro.2019.10.041
27. Zhang, J., Xing, L.: An improved genetic algorithm for the integrated satellite imaging and data transmission scheduling problem. Comput. Oper. Res. **139**, 105626 (2022). https://doi.org/10.1016/j.cor.2021.105626

# Attacker-Defender Strategy Optimization Using Multi-objective Competitive Co-Evolution

Ritam Guha[1(✉)], Ryan Mckendrick[2], Bradley Feest[3], and Kalyanmoy Deb[1]

[1] Michigan State University, East Lansing, MI 48824, USA
{guharita,kdeb}@msu.com
[2] Northrop Grumman, Falls Church, VA 22042, USA
ryan.mckendrick@ngc.com
[3] Northrop Grumman, Redondo Beach, CA 90278, USA
bradley.feest@ngc.com

**Abstract.** Attacker-defender strategy optimization deals with optimizing and deciding on different tactics used by two independent entities working in tandem. Unlike in standard optimization problems, a complete solution of the entire two-agent problem consists of strategies of both agents and evaluation of a solution requires precise information of both strategies. For this reason, a co-evolutionary optimization framework is proposed in this paper to keep two co-evolving populations interacting with each other in tandem to reach their optimal strategies. While co-evolutionary algorithms have been proposed in the past, multi-objective co-evolutionary problems make the optimization task more complex, resulting in a set of Pareto-optimal strategies for each entity. In this paper, we apply a multi-objective competitive co-evolutionary optimization algorithm to a real-world wargame strategy optimization problem. The proposed co-evolutionary algorithm is used to find trade-off sets of competitive wargame strategies for both entities and a novel post-optimization decision-making procedure is also proposed to choose preferred strategies for each entity in tandem, leading to a stable or a cycle of sequential strategies. To the best of our knowledge, this paper marks one of the first-ever applications of multi-objective, competitive, co-evolutionary optimization approaches to a real-world wargame scenario, revealing their impact and importance in practice.

**Keywords:** Attacker-defender system · Competitive co-evolution · Decision-making · Multi-objective games · Multi-agent systems

## 1 Introduction

Attacker-defender strategy optimization is a critical aspect of many security systems. In wargame scenarios, a win is determined by how each side utilizes


M. Affenzeller et al. (Eds.): PPSN 2024, LNCS 15151, pp. 351–366, 2024.
https://doi.org/10.1007/978-3-031-70085-9_22

its resources and develops effective strategies to counter the opponent's movements. While our discussion sheds some light on the study of these strategies, it should be noted that we do not support or glorify the act of war here. We simply illustrate the wargame situation as a possible application scenario leading to a challenging multi-agent co-evolutionary optimization problem. War simulations or games [6,12,20] have a large following in the gaming community and it is a very challenging problem to find optimal strategies for each of the participating entities.

In an attacker-defender system, each entity has its own set of parameters related to different resources, such as logistic information, number of assets, etc., which can be tuned to find a response strategy against the opponent. However, the outcome of one entity's parameter setting can only be evaluated properly by knowing the opponent's parameter settings. Thus, any effort to optimize one entity's strategy cannot be developed in isolation, as it is achieved in most single entity-based optimization problems. Since both entities are trying to optimize their own strategies and since they are responsive to one another, there is a need for co-evolving both entities together. While one entity's strategy gets better with optimization iterations, the other entity is also learning to produce better and better strategies of its own. This kind of two intertwined optimization problems result in the co-evolutionary optimization problem. However, if both entities have multiple conflicting objectives to optimize, the resulting problem becomes more challenging as each entity will now have a Pareto set of alternate solutions to pick a strategy from. In recent years, co-evolutionary algorithms have been gaining interest in various applications of multi-agent systems [14,15,23]. There are two types of co-evolutionary process: cooperative [3,9,10,13] and competitive [11,16,17,24]. When two or more species evolve by cooperating with each other in achieving their own goals, the process is called *cooperative* co-evolution. On the other hand, when multiple species evolve by competing against each other to fulfill their individual goals, it is called *competitive* co-evolution. In the case of wargame strategy optimization, generally, two agents will contradict each other's goals, thereby making the task a competitive co-evolution.

Co-evolutionary algorithms are mainly proposed in the literature for a single objective for each agent [5,21]. However, in a practical multi-agent system, such as in a wargame strategy optimization problem, each agent may have more than one conflicting objective to consider. The objectives of an agent can be completely different from those of the other agent or they can have opposing purpose of minimizing or maximizing the objectives. For example, the defense may minimize the loss of its assets, while the offense would, most likely, try to maximize the loss of the defense's assets. The above discussion amply indicates that solution of the attacker-defender system for multiple conflicting objectives is challenging, requiring efficient optimization algorithms and decision-making procedures.

The rest of the paper is organized as follows: Sect. 2 describes the wargame strategy optimization problem in more detail by outlining the structure of the problem, the objectives, and the complexity of optimization. The proposed multi-objective competitive co-evolutionary framework, including the optimization algorithm and post-optimization decision-making, is described in Sect. 3. The

results obtained by applying the proposed framework to the wargame strategy optimization problem are analyzed and discussed in Sect. 4. Finally, the paper is concluded in Sect. 5. From here onwards, we have used the terms attacker for offense, and defender for defense interchangeably to mean the same thing.

## 2 Wargame Strategy Optimization Problem

Wargame Strategy Optimization Problems (WSOPs) are highly idiosyncratic, and there is not one computing model that can fully represent all types of wargames. Here, we focus on a wargame where there is a clear attacker and a defender. The attacker attempts to inflict as much damage on an Air Base (Air to ground missile hits) while incurring the least cost (monetary expenses and loss of attacker lives) to themselves, but also enforcing the greatest cost on the defender (monetary expenses and loss of defender lives). The defender's goals are in direct competition with the attackers. Both agents' strategies can be broken down into three subsets: research and engineering (RE), force composition (FC), and mission plans (MP). RE and FC decisions have upfront monetary costs and are considered fixed once the mission begins. RE decisions define the capability of assets (e.g., their stealth, speed, sensor ranges, and payload capacities). FC decisions define the number of assets brought to the conflict across roles (e.g., Strike assets and Electronic Warfare assets). MP decisions define how assets coordinate and behave during the conflict (e.g., flight formations, rules of engagement, and routes). Overall, the attacker has 50 variables they can manipulate, and the defender has 24 variables. The attacker variables mainly include striker, sweeper and jammer information, while the defense variables include information about integrated air defense (IAD) systems and interceptors. In this paper, we have attempted to solve the WSOP using a multi-objective competitive co-evolutionary optimization approach.

## 3 Proposed Multi-objective Co-Evolutionary (MoCoEv) Optimization and Decision-Making Methods

In this section, we present the MoCoEv algorithm in detail. Since each entity uses multiple conflicting objectives, MoCoEv is expected to find two distinct sets of Pareto-optimal solutions, each involving all 74 decision variables. Thereafter, a decision-making procedure is needed to choose a sequence of preferred solutions from each Pareto front in a systematic manner.

### 3.1 MoCoEv Optimization Algorithm

In a MoCoEv algorithm, there are two interacting populations, each containing its own decision variables. Like in other evolutionary algorithms, initial populations $Pop_1^{(0)}$ and $Pop_2^{(0)}$ of two problems can contain random solutions $\mathbf{x}$ and $\mathbf{y}$ of sizes $N_1$ and $N_2$, respectively; however, for practical problems with previously known good solutions, initial populations can be seeded with known solutions.

Due to the computational expense in evaluating a strategy ($\mathbf{x}$,$\mathbf{y}$), we use surrogate functions during the optimization process created from a large number of pairs of strategies ($\mathbf{x}$,$\mathbf{y}$) evaluated using hi-fidelity wargame simulation software offline. While the accuracy of the developed surrogate function is a matter of their practical use, we do not address this issue here and focus on the multi-objective co-evolutionary optimization and decision-making aspects of the problem, which is already quite complex and challenging to discuss and comprehend. Here, we want to state our assumption that both entities do not know the decisions made by the other entity but the surrogate models are representative of each entity's assumption about how the other entity might behave. The proposed method can be used to have a better insight into the outcome and dynamics of the wargame based on chosen decision-making aspects, rather than actually using the method in real time.

At iteration $t$, genetic operations can be performed on $Pop_1^{(t-1)}$ as usual, using its own objectives and constraints and a new population $Pop_1^{(t)}$ of size $N_1$ can be created. Here, we follow the NSGA-II's operations for this purpose. A total of $\tau_1$ iterations can continue as above before the next population is updated for consecutive $\tau_2$ iterations. Then again $Pop_1$ can be updated for another $\tau_1$ iterations. This process can continue until a termination condition is satisfied. For simplicity, we ignore constraints in presenting our proposed algorithm. A pseudo-code of the MoCoEv procedure is presented in the supplementary document[1].

Evaluation of a single population member $\mathbf{x}$ of $Pop_1^{(t)}$ requires a specific variable vector $\mathbf{y}$ from the second population $Pop_2$. This is where a number of strategies can be adopted in an MoCoEv algorithm like summing, averaging, considering minimum, or maximum. But, we consider an averaging strategy here as it is the most used one. The $k$-th population member of $Pop_1$, $\mathbf{x}^{(1),k}$, is paired with every member $\mathbf{y}^{(2),l}$ ($l = 1, \ldots, N_2$) of the Population $Pop_2$ one at a time and $N_2$ objective vectors are evaluated. Then, a mean fitness value of $i$-th objective function of the $k$-th $Pop_1$ member is computed as follows:

$$F_i(\mathbf{x}^{(1),k}) = \frac{1}{N_2} \sum_{l=1}^{N_2} f_i(\mathbf{x}^{(1),k}, \mathbf{y}^{(2),l}). \tag{1}$$

Similarly, the fitness of each member of the second population can also be computed by averaging the respective objective values over all $Pop_1$ members.

After the average function values for each objective is computed, they can be used for domination check and other niche-preserving operators of the chosen evolutionary multi-objective (EMO) algorithm. Every member is evaluated for both $F_1$ and $F_2$ using all members of the second population. Then, the mean fitness vector is computed for each member and is used for the domination check. The *crowding distance* values for each member, needed for diversity preservation of non-dominated solutions, are also computed using the aggregate fitness values. Thus, the final trade-off set of solutions of each population ($\mathbf{Z}^{(1)}$ and $\mathbf{Z}^{(2)}$)

[1] Google Drive Supplementary Link.

will correspond to the non-domination principle of the chosen aggregate fitness functions.

The variation operators in the MoCoEv algorithm are recombination and mutation operators which are applied to the respective population, independently. For example, for every generation of $Pop_1$, recombination and mutation operators are applied to $\mathbf{x}$-vectors of $Pop_1$ only, while keeping the current $\mathbf{y}$-vectors of $Pop_2$ unchanged. Every newly created $\mathbf{x}$-vector is then evaluated using the fixed $\mathbf{y}$-vectors of $Pop_2$ and average fitness is used for the survival operator of $Pop_1$. After $\tau_1$ such generations are executed with $Pop_1$, then $\tau_2$ generations are performed by keeping $\mathbf{x}$-vectors fixed.

The above successive cycles of two population updates are continued until a termination condition is met. In this study, we run until a pre-defined number of combined generations ($T$) is elapsed. After the MoCoEv run, a separate non-dominated (ND) set will be found for each agent.

### 3.2 MoCoEv Decision-Making Procedure

Selecting a single Pareto solution for agent from their respective ND set is a challenging decision-making task. We propose the following procedure. Since the wargame is to be played alternatively between the two players, an offline computation of ND strategies becomes a pragmatic approach. In the case of attacker-defender WSOP, the defender (or, attacker) may start by choosing a strategy from its own ND set based on certain initial preference information among its objectives. With the defender's strategy revealed, the attacker can then choose the most preferred strategy from its own ND set, so that maximum effect can be imposed on the defender's chosen strategy. Next, it will be the defender's turn to choose the next appropriate strategy from its own ND set to negotiate the attacker's chosen strategy. These iterative moves can be continued until either an equilibrium state (converged attacker and defender strategies) or an equilibrium cycle (converged cycle of attacker and defender strategies) is arrived.

One of the remaining tasks is to discuss the specific multi-criterion decision-making (MCDM) method where each player can adopt to pick a single appropriate strategy from its own ND set. Let us say that there are $N_1$ attacking strategies and $N_2$ defending strategies after the MoCoEv run.

The first step of the decision-making process is to select one of the agents for initiation. Let us say we choose the defender at first. So, the defender needs to choose one of the $N_2$ final ND strategies. Each of these strategies will have a distribution of objective values as one defending strategy is evaluated against $N_1$ different attacking strategies from the attacker's ND set, leading to $N_1$ objective scores. So, we can calculate the normalized standard deviation of all objective values for each defending strategy. The initial defending strategy is the one having the lowest average normalized standard deviation for the objectives. Intuitively, having a lower normalized standard deviation reflects some type of robustness of a strategy against all the opposing strategies.

After the first strategy is selected from the defender, the attacker needs to select its own strategy. Every attacking strategy is evaluated against the selected defending strategy and the corresponding ND set is identified from them. Finally, for selecting a single attacking strategy from the ND set, we use the highest trade-off-based selection method described in [19]. For any ND set, the trade-off value for an ND point $x_i$ with a neighboring ND point $x_j$ can be calculated as:

$$R(x_i, x_j) = \frac{w_{loss} \times Loss_f(x_i \rightarrow x_j)}{w_{gain} \times Gain_f(x_i \rightarrow x_j)}. \tag{2}$$

Here, $w_{loss}$ and $w_{gain}$ represent the weights provided to loss and gain acquired through moving from one solution to another, respectively. This trade-off value gives an approximation of the amount of loss that should be accepted, compared to the gain for moving from $x_i$ to $x_j$. The final trade-off value for each point gets computed by taking an average of its trade-off with its neighbors. A higher trade-off value for a point indicates that moving away from the point results in a high loss. Thus, the decision-makers usually prefer the highest trade-off point from the ND set. In this way, we keep selecting the highest trade-off point from the ND set of each side from hereon.

After selecting the attacking strategy, the defender selects a strategy using the same procedure as the attacker. This process continues until we can find an equilibrium point where the same defending strategy and attacking strategy keep on getting selected in each iteration or an equilibrium cycle where the same sequence of defending and attacking strategies get selected in a cycle.

## 4 Results and Discussion

After carefully formulating the proposed MoCoEv algorithm and decision-making procedure, we apply the proposed framework to the WSOP. In this section, we first show the final formulation of the problem, followed by some preliminary analysis of the optimization of the problem without any co-evolution, and finally, we evaluate the performance of the proposed co-evolutionary process.

### 4.1 Solution Evaluation and Surrogate Model

In our WSOP case study, we execute each scenario using the 'Command Modern Air and Naval Operations' (CMANO) [2] system. Each scenario takes, on average, about six minutes under a graphics-less accelerated execution. It is still too slow to use this high-fidelity model within an optimization code. Therefore, a surrogate model is learned for each of the five competing attacker/defender objectives.

### 4.2 WSOP Objectives

The objectives and the goals of the WSOP entities are presented in Table 1. The analysis of the dataset used for training the surrogate models reveals that some of the objectives are correlated with each other. In Fig. 1, we can see that *OffenseLifeCost* is highly correlated with *OffenseExpenditures* (corr. coeff.:

**Table 1.** The original objectives and attacker-defender goals for the WSOP.

| Objective | Att. | Def. |
|---|---|---|
| *OffenseHits* | ↑ | ↓ |
| *OffenseLifeCost* | ↓ | ↑ |
| *DefenseLifeCost* | ↑ | ↓ |
| *OffenseExpenditures* | ↓ | ↑ |
| *DefenseExpenditures* | ↑ | ↓ |

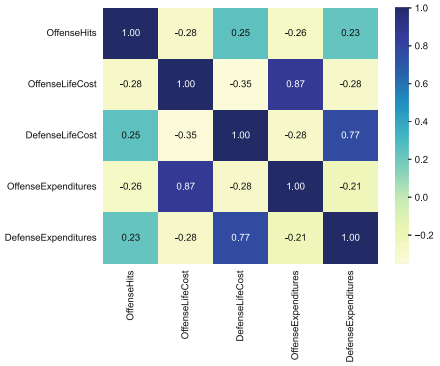



**Fig. 1.** Correlation analysis of WSOP objectives.

0.87) and similarly *DefenseLifeCost* is highly correlated with *DefenseExpenditures* (corr. coeff.: 0.77). These observations make sense because intuitively more *OffenseLifeCost* means more resources of the attacker getting destroyed leading to more *OffenseExpenditures*. The same logic applies to the defending entity, as well. For this reason and to make the problem simpler to solve, we eliminate two objectives – *OffenseLifeCost* and *DefenseLifeCost* – from further consideration. This reduces the number of objectives of the problem to three. We further reduce one more objective by constructing two conflicting objectives, as mentioned in Table 2.

**Table 2.** Reduced objectives for WSOP.

| $i$ | Objective ($f_i$) | Offense Goal | Defense Goal |
|---|---|---|---|
| 1 | *OffenseExpenditures - DefenseExpenditures* | ↓ | ↑ |
| 2 | *OffenseHits* | ↑ | ↓ |

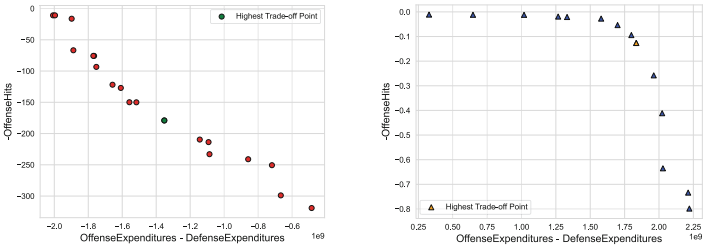
### 4.3 Multi-objective Optimization Without Co-Evolution

To have an understanding of the optimal solutions for each entity separately and without having control from the other entity, we optimize the WSOP objectives for each entity without using co-evolution as a multi-objective optimization problem. For the offense entity, the second objective (Table 2) is to be maximized. We convert this objective to formulate a two-objective minimization problem, as follows:

$$\min \{(\mathit{OffenseExpenditures} - \mathit{DefenseExpenditures}), -\mathit{OffenseHits}\}.$$

Intuitively, they are in conflict, as attempting to cause a large damage to the defense (small value of negative offense hits) will incur a large expenditure for the offense, thereby causing a large first objective value.

We employ the standard NSGA-II procedure [7] without co-evolution and run with 50 population members for 200 generations. The ND front is presented in Fig. 2. From the Figure, we can observe that to increase the number of offense hits from 10 to 300, the offense's expenditure needs to be close to the defense's expenditure. In all cases, an independent optimization of the offense's objectives causes the defense to make larger expenditures than the offense and still not protect the defense's assets to a large number.

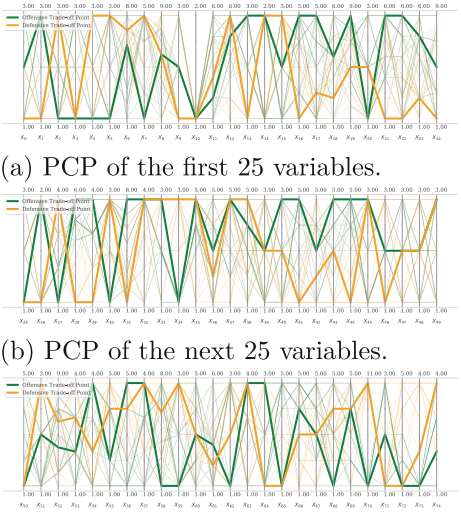



(a) Offense's objectives. (b) Defense's objectives.

**Fig. 2.** Non-dominated fronts for independently optimizing each entity's objectives without co-evolution.

We repeat the NSGA-II application for the defense entity next. Since the first objective is to be maximized (Table 2) for defense, we use the following formulation: min $\{-(\mathit{OffenseExpenditures}$ - $\mathit{DefenseExpenditures}), \mathit{OffenseHits}\}$.

Figure 2b presents the trade-off ND front. Interestingly, the number of offense hits in the ND front is between zero and 0.8 (average over multiple scenarios), much smaller than that obtained by the offense's independent optimization runs. This is because the objective *OffenseHits* is now being minimized, instead of being maximized. A complete control by defense via an independent optimization of its own objectives cause offense to make more expenditure and still not generate too much damage to the defense assets. This demonstrates the power of an optimization run in providing the best possible strategies for the chosen objectives. To demonstrate the difference between the obtained solutions, we choose the highest



(a) PCP of the first 25 variables.

(b) PCP of the next 25 variables.

(c) PCP of the final 25 variables.

**Fig. 3.** PCP for all ND solutions obtained using independent optimizations of offense and defense entities. The highest trade-off solutions are shown in bold.

trade-off objective vectors for each entity from the respective plots: Fig. 2a for offense, and Fig. 2b for defense. In the respective ND front, the specific trade-off solution causes a maximum loss in one objective for a unit gain in the other objective [19], making it a preferred choice among other ND solutions. It is clear that the two trade-off solutions are completely different from each other. This reveals the competitive nature of optimal strategies for the two entities. To understand the difference of ND strategies for two entities, we present the decision variables corresponding to trade-off as well as other ND points using the parallel coordinate plots (PCPs) in Fig. 3. Clearly, the green and orange lines are aligned differently. Focusing on the two highest trade-off solutions (shown in bold), we observe that out of a total of 74 offense and defense variables, 53 variables (almost 70%) have completely different values. This experiment confirms that when we optimize the contradicting objectives, the algorithm searches in different portions of the search space resulting in solutions which are widely different in objective as well as variable space.

### 4.4 Multi-Objective Competitive Co-Evol. (MoCoEv) Optimization

Figures 2a and 2b have clearly shown that when strategies from two entities are intricately involved in defining objectives, individual optimization of one entity alone does not produce satisfactory results for both entities. The solutions are biased towards the entity for which the optimization is performed. This motivates us to consider the WSOP as a co-evolutionary optimization (CoEv) problem. Since each entity has two conflicting objectives on its own, the problem becomes a multi-objective co-evolutionary optimization (MoCoEv) problem. Moreover, since there is conflict in the individual optimal solutions between the two entities, as found by widely different values of the ND front solutions in Fig. 2, the problem becomes more challenging and is known as a multi-objective competitive co-evolutionary optimization (MoCCoEv) problem. In this subsection, we apply the MoCoEv algorithm presented in Sect. 3 to find an ND set of trade-off solutions, each of which takes into account both entities during the optimization process.

**Competitive Trade-Off Solutions:** We use a co-evolutionary version of NSGA-II having a population of size 50 for each co-evolving population and run the proposed MoCoEv algorithm for 200 generations with $\tau_1 = \tau_2 = 5$. The final outcome of the MoCoEv algorithm are two ND sets of solutions (having 10 solutions each), one for each entity. We compute average objective values for each solution for one entity with every member of the other entity and then identify the ND solutions for both entities based on these average values.

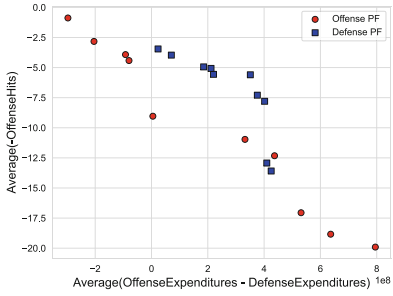



**Fig. 4.** Offense and Defense ND sets by MoCoEv.

The respective ND fronts based on these average objective vectors of each entity are plotted in Fig. 4. Clearly, for the offense population, objectives (*OffenseExpenditures* - *DefenseExpenditures*) and $-$*OffenseHits* are minimized, while for the defense population, the negative of the objectives are minimized.

It is interesting to note that both ND fronts are now closer to each other, meaning that the MoCoEv algorithm is able to emphasize both entities' interests well in arriving at competitive solutions. The number of offense hits is now limited to a maximum of 20, instead of around 300 obtained using the offense's independent optimization. In all defense ND solutions, offense expenditure is more than that of defense. To achieve up to 10 offense hits, the offense needs to make little more expenditure than the defense, while the offense has to outspend the defense to destroy more than 10 assets of defense. For example, to achieve 20 offense hits, the offense has to spend about 8 million units more than the defense, however with about similar expenditure, the offense can achieve 9 offense hits. On the other hand, if the offense is happy with damage of about 2.5 units of assets, the offense entity can spend 2 million units of expenditure less than that of the defense. Moreover, ND solutions for the offense entity have a wider range of objectives than that of the defense entity. This can be due to the existence of more offense variables, thereby providing more ways to find a wider combination of variables. All these observations are interesting providing offense and defense users with a better insight into various alternate solutions before they prepare to launch any action.

**Extracting Common Patterns in Trade-Off Solutions:** The process of extracting common patterns in a set of ND solutions was termed as the task of "innovization" [8]. Patterns can be extracted from the ND solutions manually using certain problem information [8] or using an automated machine learning process [4]. Here, we use a manual process in which all 74 variables of both entities are plotted in the order of their similarity among the obtained ND solutions. For this purpose, we compute the coefficient of variation (CV - $\sigma/\mu$) of each solution and the order of the offense and defense variables in Fig. 6.

A higher CV indicates higher dispersion around the mean. So, a lower CV reflects a better-converged value of variables towards the mean value. It is reported in [1] that a threshold of $\sigma/\mu = 0.3$ is an acceptable limit for assuming a good convergence. With this threshold, we observe that six of the 50 offense variables and six of the 24 defense variables can be considered well converged. The number of values that each variable can take in the original formulation is mentioned in parentheses in Fig. 6.

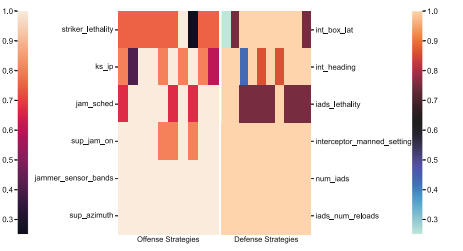



**Fig. 5.** Heatmap of converged variables for offense and defense entities.

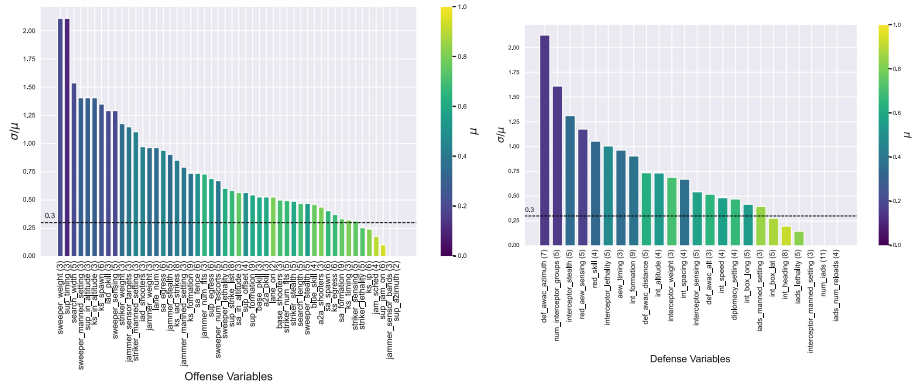



**Fig. 6.** Coefficient of variation for offense and defense variables indicate convergence of certain variables across ND solutions.

To analyze further, we normalize values for the converged variables over the PFs between zero and one and are plotted in a heat map in Fig. 5. Each variable's values in 10 ND solutions are shown in columns for each entity. Almost identical colors for each row (variable) indicate the convergence level visually. For the offense variables, we note that three of the six variables are related to electronic warfare MP, suggesting that there is a set of plans for the offense entity that significantly inhibit the perception and communication of the defender. Other converged offense variables dictate where a strike is safe and effective (KS_IP), as well as the simple fact that carrying more air-to-ground munitions is consistently a good choice (striker_lethality). From the defense's perspective, we observe that more surface-to-air missile launchers (num_iads) that can fire more missiles (num_reloads) for a longer duration (iad lethality) is a cost-effective strategy. This also pairs well with autonomous interceptors (interceptor manned).

**Optimization With and Without Coevolution:** Figure 7 marks the original high-fidelity objective vectors (in yellow circles) used to construct the surrogate models of the objectives. The data were centered around equal expenditures for both entities causing on average around four offense hits. These solutions are marked with the label "Surrogate Fitted Training Points". If we do not perform any optimization run and try to locate the best strategies for the offense entity, we find the respective ND front marked using orange circles on the top-left part of the yellow circles. Similarly, when we locate the best strategies for defense, they are at the bottom-right corner of the yellow circles, marked using purple squares.

We embed the individually optimized ND solutions in the plot for both offense and defense and they are marked using brown circles and black squares, respectively. Notice that both these sets clearly dominate the respective sets obtained from initial high-fidelity data only and without resorting to any optimization. Clearly, these individual optimal solutions are better due to the efforts put in by

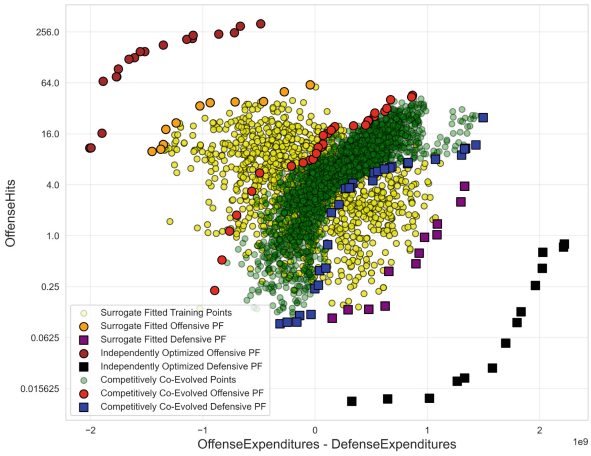



**Fig. 7.** Training data, independently-optimized and MoCoEv strategies. (Color figure online)

the optimization algorithm. However, it was discussed before that these individual solutions are not practical, as the optimizations ignore the ability of the other entity to influence the solutions of its own entity.

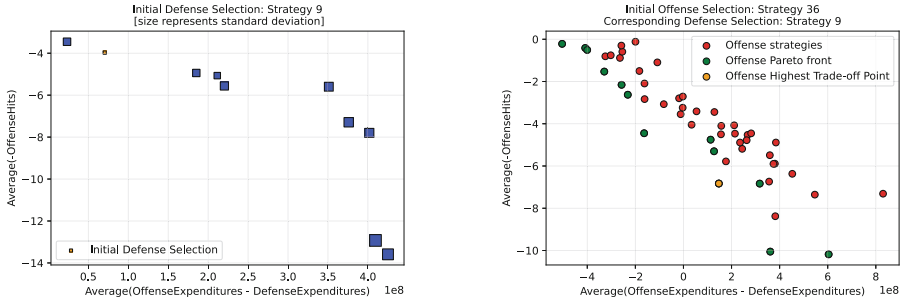
Next, we plot the MoCoEv ND solutions found for both entities in red circles and blue squares. All final strategies found via the MoCoEv optimization are also shown in green-colored open circles. It is clear that the coevolution-based solution sets are closer to each other, thereby respecting each other's abilities to arrive at challenging solutions for each other. Due to the consideration of two conflicting objectives for each entity, both ND sets produce trade-off solutions between two objectives. We argue that the MoCoEv trade-off solutions stay as viable strategies from which each entity can choose a solution.

**Multi-Criterion Decision Making on MoCoEv Solutions:** Finding a set of ND solutions for each entity in a co-evolutionary process is a challenging computing task, but it only completes a part of the whole WSOP task. The next important task is to choose preferred solutions for implementation. Despite a plethora of multi-criterion decision-making (MCDM) studies in the literature [18, 22], MCDM studies for two co-evolved Pareto sets is missing. In this paper, we make an effort to propose a viable MCDM procedure.

Since two entities – defense and offense – are independent, likely they will also make decisions independently. However, the linking of the two entities in defining the objectives suggests that a sequential decision-making task must be made alternating between them. This will lead to a defender-attacker simulation game which will start with declaring a defense's solution. The offense then has the opportunity to choose its solution to counteract the declared defense's solution. After the offense's solution is announced, the defense has the next move to choose

its most appropriate action. This iterative process can continue until it reaches an equilibrium pair of strategies or an equilibrium cycle of strategies. With this iterative scheme of the MCDM procedure in mind, the question remains as to how to choose a preferred solution when the opponent has made its move.

Before we propose an MCDM scheme for this study, we make an important assertion about our decision-making approach. We assume that each entity will confine its decision-making to its own ND set corresponding to the final set of strategies for the other entity. Since the MoCoEv is expected to find the best possible trade-off solution set considering the opponent's best possible moves, each set is expected to have the best counter-moves in them. Hence, it makes sense to use the obtained ND set to choose a solution from.




(a) Selection of initial defense strategy.

(b) Initial offense strategy selection.

**Fig. 8.** Initial strategy selection by offense and defense.

As outlined in Sect. 3.2, we first identify the specific defense strategy corresponding to the smallest standard deviation in its objective vectors arising from the various combinations of solutions in the offense population. Figure 8a marks the size of each defense solution in proportion to the corresponding standard deviation arising from all offense strategies. Strategy 9 (second from top-left point), when combined with each offense ND solution produces the least standard deviation in both defense objective values. Hence, it may be considered the most robust strategy against any strategy that the offense entity may choose next. Thus, *Defense_9* is selected as the initial defense strategy.

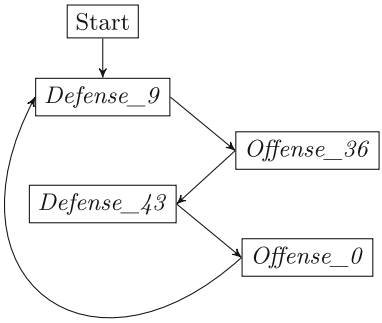



**Fig. 9.** Proposed method results in an equlibrium cycle of strategies.

After selecting the initial defense strategy, we now plot the objective vector of every offense strategy with this defense strategy (*Defense_9*) in Fig. 8b. Clearly,

not all objective vectors are non-dominated to each other. We now identify the non-dominated set from these vectors and choose the highest trade-off point. This strategy *Offense_36*, marked in a yellow circle, is the best response by the offense to the *Defense_9* strategy of the defense.

Due to this lop-sided loss-to-gain ratio, there is no motivation to move to its neighbor for a better outcome. Next, it is the turn of the defense to find the best possible defense strategy from its own MoCoEv-obtained set of strategies. With a similar trade-off analysis of non-dominated solutions obtained from all objective vectors computed using *Offense_36* and each defense's ND solution, we observe that *Defense_43* is the best option. Next, we find *Offense_0* strategy of offense ND set is the best response. One more trade-off analysis of the defense ND set reveals that *Defense_9* is now the best defense strategy. Interestingly, *Defense_9* was one of the chosen strategies considered before and which led to *Offense_0* in a few iterations. Continuing the decision-making process further will lead to a cycle of offense and defense strategies which we have already observed before. Thus, the MCDM process ends up in an equilibrium cycle of strategies between the two entities. This ends the WSOP task. Figure 9 shows the cycle by clearly marking the sequence of defense and offense strategies.

## 5 Conclusions and Future Studies

In this paper, we have proposed a multi-objective, competitive, co-evolutionary optimization algorithm and a decision-making strategy to deal with two agents, whose evaluation functions require both agent's variables. In every sense, the problem has introduced challenges in arriving at practical solutions. The optimization problem is challenging due to the involvement of multiple conflicting objectives and inter-dependencies of both agent's variables, leading to two non-dominated fronts. The decision-making is challenging due to involvement of two independent decision-makers, each deciding from a different non-dominated set of strategies in tandem and in response to opponent's moves. Not only does each agent choose a preferred strategy from its own ND set by trading-off two conflicting goals, the chosen strategy must also be appropriate in response to opponent's recently chosen strategy. Standard evolutionary multi-objective optimization (EMO) and multi-criterion decision-making (MCDM) methods are not as involved as MoCoEv and ensuing decision-making tasks.

We have applied the proposed approaches to a wargame strategy optimization problem (WSOP) to illustrate its working and revealing the complex interactions involved in solving an attacker-defender system. From the five objectives of interest for defense and offense agents, a correlation analysis has allowed us to reduce the problem to have only two objectives for each agent. For the first time, we have proposed a sequential MCDM approach by involving one agent at a time. A trade-off-based MCDM approach has been proposed to find the best ND solution of one entity as a response to all ND solutions of the other strategy. In the specific case study, the proposed MCDM scheme has resulted in an equilibrium cycle of offense-defense strategies as an end result.

This study has clearly shown the advantage of using evolutionary computation in addressing multi-objective competitive multi-agent systems and investigates a number of future studies. Iterated MCDM schemes may involve the depletion of resources into consideration as the game progresses. It may also restrict future moves only to moves allowed by the initial moves by each entity. More than two objectives can be considered to have more flexible trade-off solutions. The MoCoEv and ensuing MCDM approach can be applied to other similar multi-agent systems in achieving a better understanding of the effect of sequential decision-making strategies in achieving safe and secure systems.

**Disclosure of Interes.** I declare no competing interests as defined by Springer Nature, or other interests that might be perceived to influence results and/or discussion reported in this manuscript.

## References


1. Coefficient of variation. https://www.isixsigma.com/dictionary/coefficient-of-variation/. Accessed 27 Sept 2023
2. Command Modern Air and Naval Operations. https://www.matrixgames.com/game/command-modern-air-naval-operations-wargame-of-the-year-edition. Accessed 19 Jun 2024
3. Atashpendar, A., Dorronsoro, B., Danoy, G., Bouvry, P.: A scalable parallel cooperative coevolutionary PSO algorithm for multi-objective optimization. J. Parall. Distrib. Comput. **112**, 111–125 (2018)
4. Bandaru, S., Ng, A.H.C., Deb, K.: Data mining methods for knowledge discovery in multi-objective optimization: Part A - survey. Expert Syst. Appl. **70**, 139–159 (2017)
5. Barbosa, H.J.C.: A genetic algorithm for min-max problems. In: Proceedings of the First International Conference on Evolutionary Computation and Its Application (EvCA'96), pp. 99–109 (1996)
6. De Lima Filho, G.M., Kuroswiski, A.R., Medeiros, F.L.L., Voskuijl, M., Monsuur, H., Passaro, A.: Optimization of unmanned air vehicle tactical formation in war games. IEEE Access **10**, 21727–21741 (2022)
7. Deb, K., Agrawal, S., Pratap, A., Meyarivan, T.: A fast and elitist multi-objective genetic algorithm: NSGA-II. IEEE Trans. Evol. Comput. **6**(2), 182–197 (2002)
8. Deb, K., Srinivasan, A.: Innovization: innovating design principles through optimization. In: Proceedings of the 8th Annual Conference on Genetic and Evolutionary Computation, pp. 1629–1636 (2006)
9. Dorronsoro, B., Danoy, G., Nebro, A.J., Bouvry, P.: Achieving super-linear performance in parallel multi-objective evolutionary algorithms by means of cooperative coevolution. Comput. Oper. Res. **40**(6), 1552–1563 (2013)
10. Garcıa-Pedrajas, N., Hervás-Martınez, C., Munoz-Pérez, J.: Multi-objective cooperative coevolution of artificial neural networks (multi-objective cooperative networks). Neural Netw. **15**(10), 1259–1278 (2002)
11. Goh, C.K., Tan, K.C., Liu, D., Chiam, S.C.: A competitive and cooperative coevolutionary approach to multi-objective particle swarm optimization algorithm design. Eur. J. Oper. Res. **202**(1), 42–54 (2010)
12. Jia, Z.X., Kiang, J.F.: War game between two matched fleets with goal options and tactical optimization. AI **3**(4), 890–930 (2022)

13. Keerativuttitumrong, N., Chaiyaratana, N., Varavithya, V.: Multi-objective co-operative co-evolutionary genetic algorithm. In: Guervós, J.J.M., Adamidis, P., Beyer, H.-G., Schwefel, H.-P., Fernández-Villacañas, J.-L. (eds.) Parallel Problem Solving from Nature — PPSN VII, pp. 288–297. Springer, Berlin, Heidelberg (2002). https://doi.org/10.1007/3-540-45712-7_28
14. Li, Y., Wang, J., Liu, Z.: A simple two-agent system for multi-objective flexible job-shop scheduling. J. Comb. Optim. **43**(1), 42–64 (2022)
15. Luo, J., Cooper, J., Cao, C., Pham, K.: Cooperative adaptive control of a two-agent system. In: 2012 American Control Conference (ACC), pp. 2413–2418. IEEE (2012)
16. McIntyre, A.R., Heywood, M.I.: Multi-objective competitive coevolution for efficient GP classifier problem decomposition. In: 2007 IEEE International Conference on Systems, Man and Cybernetics, pp. 1930–1937. IEEE (2007)
17. Meneghini, I.R., Guimaraes, F.G., Gaspar-Cunha, A.: Competitive coevolutionary algorithm for robust multi-objective optimization: the worst case minimization. In: 2016 IEEE Congress on Evolutionary Computation (CEC), pp. 586–593. IEEE (2016)
18. Miettinen, K.: Nonlinear Multiobjective Optimization. Kluwer, Boston (1999)
19. Mittal, S., Kumar, D., Deb, S.K.: A unified automated innovization framework using threshold-based clustering. In: 2020 IEEE Congress on Evolutionary Computation (CEC), pp. 1–8. IEEE (2020)
20. Ozaki, A., Furuichi, M., Takahashi, K., Matsukawa, H.: Design and implementation of parallel and distributed wargame simulation system and its evaluation. IEICE Trans. Inf. Syst. **84**(10), 1376–1384 (2001)
21. Paredis, J.: Coevolutionary constraint satisfaction. In: Parallel Problem Solving from Nature III (PPSN-III), pp. 46–55 (1994)
22. Steuer, R.E.: Multiple Criteria Optimization: Theory, Computation and Application. Wiley, New York (1986)
23. Wang, K., Gou, C., Duan, Y., Lin, Y., Zheng, X., Wang, F.Y.: Generative adversarial networks: introduction and outlook. IEEE/CAA J. Autom. Sinica **4**(4), 588–598 (2017)
24. Zeng, F., Decraene, J., Low, M.Y.H., Cai, W., Hingston, P.: Studies on pareto-based multi-objective competitive coevolutionary dynamics. In: 2011 IEEE Congress of Evolutionary Computation (CEC), pp. 2383–2390. IEEE (2011)

# On the Multi-objective Optimization of Wind Farm Cable Layouts with Regard to Cost and Robustness

Lee A. Christie(✉), Atakan Sahin, Akinola Ogunsemi, Alexandru-Ciprian Zăvoianu, and John A. W. McCall

National Subsea Centre, Robert Gordon University, Aberdeen, Scotland, UK
{l.a.christie,a.sahin,a.ogunsemi1,c.zavoianu,j.mccall}@rgu.ac.uk

**Abstract.** Offshore wind farms (OWFs) have emerged as a vital component in the transition to renewable energy, especially for countries like the United Kingdom with abundant shallow coastal waters suitable for wind energy exploitation. As net-zero emissions targets propel investments in renewables, OWFs present unique engineering challenges, particularly in the design of cost-effective and efficient infrastructural networks such as layout and electrical system optimization. Diverging from the previous approaches in electrical system optimization for OWFs, this paper introduces network robustness as a pivotal metric in design evaluations, differing from traditional reliability evaluation focused studies. By designing approximate solutions to the capacitated minimum spanning tree (CMST) using an approach grounded in a radial space partitioning strategy, the application of the Non-dominated Sorting Genetic Algorithm II (NSGA-II), and a bespoke domain-specific mutation operator, we present a multi-objective exploration of the cost-robustness trade-off. To demonstrate the effectiveness of our approach and its ability to offer decision makers valuable insight on cable layout designs, we apply it to a real-world case study that considers the Anholt OWF. The obtained results indicate the ability of our approach to discover sets of high-quality solutions, underscoring its potential to enhance the strategic development of robust and economically viable OWF networks.

**Keywords:** topology optimization · network robustness · offshore wind farm · inter-array cabling · optimal trade-offs · planarity constraints

## 1 Introduction and Motivation

The integration of renewable energy sources, particularly wind and solar power, has become a pivotal aspect of energy investment strategies, driven by net-zero emissions policies enacted by several nations. Offshore wind farms (OWFs), in particular, are crucial for meeting the energy demands of countries with extensive shallow coastal waters, like the United Kingdom, where areas such as Dogger


M. Affenzeller et al. (Eds.): PPSN 2024, LNCS 15151, pp. 367–382, 2024.
https://doi.org/10.1007/978-3-031-70085-9_23

Bank offer prime conditions for wind energy harvesting. The rising popularity of OWFs brings with it a set of technical challenges, which are the byproducts of creating an efficient and cost-effective infrastructure. Typically, a wind farm comprises a central substation connected to numerous turbines via a complex cable network. The optimization of wind farm layout design is critical, as it directly influences the facility's power production efficiency and the rate of return on investment.

OWF design can be categorized into two distinct segments: (i) wind farm layout optimization, which determines the placement of wind turbines by addressing the micro-siting problem, and (ii) electrical system optimization, which establishes the cable connection configuration and selects the appropriate cable types. Notably, the costs associated with the electrical infrastructure represent approximately 15% of the total initial outlay for an offshore wind farm, an amount that is on par with the expenditure for the turbines themselves [15].

The wind farm cabling challenge can be systematically broken down into several hierarchical layers. Berzan [1] categorizes this issue into three distinct levels:

**1. The circuit problem** represents the most basic and smallest scale of the issue, focusing on connecting a specified set of turbines to form a single circuit.
**2. The substation problem**, which is of intermediate complexity, involves a designated substation and a group of turbines that need to be connected to it. The objective here is to link the turbines to the substation as cost-effectively as possible, typically resulting in a solution that resembles a spanning tree.
**3. The full farm problem** encompasses a more complex scenario with potentially multiple substations and all turbines. The solution involves creating a forest of spanning trees, with each tree rooted to a substation, to efficiently connect the entire farm.

In the substation and full-farm problems, turbine locations are considered as nodes, and the cables connecting them act as edges in a graph, leading to a capacitated minimum spanning tree (CMST) problem [3] formalization with specific constraints based on cable types. Various heuristic algorithms such as the Esau-Williams algorithm [7], have been proposed to address the CMST problem. To overcome limitations inherent in these heuristics, such as cable capacity constraints, clustering techniques can be employed alongside them for improved effectiveness.

Moreover, metaheuristic algorithms have been developed to refine the solutions offered by traditional CMST heuristics, specifically targeting issues like the propensity to converge on local optima. These algorithms employ probabilistic criteria to more thoroughly explore the design space. Prominent metaheuristic methods include genetic algorithms (GA) and particle swarm optimization (PSO) [14].

In addition to considering the length and cost of cables in the full farm problem, various objectives are taken into account in optimization studies. Objectives include different forms of energy losses and reliability indexes, where the latter

evaluates the amount of energy not supplied due to failures [4,12]. To the best of our knowledge, the evaluation of network robustness has not been previously considered as an objective in the optimization of cable layouts. In this study, we address the cable layout problem by focusing on two main objectives: the cost associated with various types of cables and the robustness of the network. We apply the Non-dominated Sorting Genetic Algorithm II (NSGA-II) [5] in combination with a radial-based space partitioning approach to generate Pareto fronts that illustrate the trade-off between these two key objectives.

The structure of this paper is organized as follows: Sect. 2 introduces the problem, defining the cabling layout, the metrics for evaluation, and the example OWF utilized in this study. Section 3 outlines the proposed methodology. The numerical simulations we conducted are described in Sect. 4, and Sect. 5 presents our conclusions and outlook on future work.

## 2 Problem Definition

The full farm problem necessitates a variety of evaluation criteria to achieve the desired optimal solution. Departing from conventional studies, we recognize network robustness as a critical component in electrical system optimization, enhancing the system's ability to handle failures effectively. In this preliminary study, we focus on two key evaluation criteria—cost and robustness—which must be addressed in a multi-objective optimization context due to their conflicting nature. Additionally, we detail the specifics of the Anholt OWF and its electrical system as this wind farm serves as the case study for our proposed approach.

### 2.1 Cable Cost Evaluation

Evaluating the cost of developed OWF cable layouts across a broad spectrum of cable capacities is one of the key objectives. This requires detailed information on cable costs relative to their respective capacities. The cable costs considered in this analysis encompass both procurement and installation expenses. It is important to note that this cost model does not account for the additional cable length required to connect from the seabed to the turbine's transformer and back.

The primary function of the cable cost model in Eq. 1 is to highlight the relative cost differences between various cabling options, rather than to accurately estimate the total cost of the entire cabling layout. A wind farm can be represented as an undirected graph $G$ with $v$ vertices (representing turbines and substations) and $e$ edges (representing cable segments). Let $G = \{V, E\}$, where $V$ is the set of all vertices and $E$ is the set of all edges. The cost of the network is defined as:

$$C(G) = \sum_{e \in E} \text{cost}(e) \cdot \text{len}(e) \tag{1}$$

where $\text{len}(e)$ is the length of the cable calculated according to the Haversine distance [10] between the geo-location of the two end points, and $\text{cost}(e)$ is

the cost of the cable type assigned by the procedure described in Fig. 2 cross-referenced with the unit pricing grid in Table 1.

### 2.2 Network Robustness Evaluation

In power system modelling, a wide array of methodologies has been proposed to simulate cascading failures [9]. Each model offers unique focuses and advantages, yet comprehensive information about the overall phenomenon is still crucial for accurate simulations. Among these methodologies, modified topological models stand out by incorporating certain electrical properties, such as Kirchhoff's law, line impedance or reactance, line capacity, and flow-based analysis.

From the perspective of cascading failures, robustness refers to the system's ability to maintain normal service for a critical percentage of clients, even when some components fail. This study evaluates the robustness of wind farms using a network-based approach, drawing on the modified topological model for cascading failures developed by Zhang and Chi [17] derived from the admittance model by Grainer and Stevenson [8]. $T$, with $|T| = l$ represents the set of wind turbines in $G$ and $S$ represents the set of the substations[1] in $G$, and $T$, $S \subset V$. We define the robustness of the entire graph G as $R(G)$ measured in Eq. (2):

$$R(G) = \frac{1}{v} \sum_{i \in V} P(i) \tag{2}$$

where $P(i)$ is defined as the percentage of vertices that still serve a substation after the failure of a component $i$. If component $i$ is considered as failed, the component will be removed from the network and the robustness of the network given this failure is computed as $P(i)$ as shown in Eq. (3). This computation is based on the fact that each failed component $i$ creates a total of $k_i$ subgraphs and for each such subgraph $j$, if there is no substation connected, power cannot be supplied by any of the vertices in the subgraph, i.e. the power output associate with the subgraph $j$ is $Q(i, j) = 0$. Alternatively, if subgraph $j$ is still connected to a substation after the failure of component $i$, its associated power $Q(i, j)$ is equal to the proportion of vertices $p(i, j)$ it contains as defined in Eq. (4).

$$P(i) = \sum_{j=1}^{k_i} Q(i, j) \tag{3}$$

$$Q(i, j) = \begin{cases} p(i, j), & \text{if a substation is connected to subgraph } j \\ 0, & \text{otherwise} \end{cases} \tag{4}$$

[1] $S$ is a singleton set for many OWF layouts, including our present use case described in Sect. 2.4.

### 2.3 Multi-objective Formulation

For a fixed set of vertices $V$, the multi-objective problem defined in Eq. (5) considers two objective functions as we seek to determine optimal trade-offs when searching for a wind farm cabling topology $E'$ that simultaneously minimizes the cabling cost $C(G')$ and maximizes robustness $R(G')$ for $G' = \{V, E'\}$:

$$F(G') = \begin{cases} min & C(G') \\ max & R(G') \end{cases} \tag{5}$$

subject to:

$$\forall s \in \{0, 1, \ldots, n-1\} \quad \exists v \in V \text{ such that section}(v) = s \tag{6}$$

$$\forall s \in \{0, 1, \ldots, n-1\} \quad |\{v : \text{section}(v) = s\}| \leq v_{max} \tag{7}$$

$$\forall\, (e_1 \neq e_2) \in E' \quad \neg\text{intersects}(e_1, e_2)\,, \tag{8}$$

where intersects$(e_1, e_2)$ is the condition that edges $e_1$ and $e_2$ are line segments that intersect in the plane, and section$(v)$ is the OWF section number encoded for the vertex $v$ (see Sect. 3.1 for details). The first two constraints are section-related and they ensure that there are no empty sections without turbines Eq. (6) and there are at most $v_{max}$ turbines per section to mitigate excessive curtailment due to maximum cable capacity Eq. (7). Equation (8) concerns graph planarity and aims to prevent overlaps between cables.

### 2.4 The Anholt Offshore Wind Farm

The Anholt OWF was constructed in 2015 in the Kattegat Strait, approximately 15 km off the east coast of Denmark. It comprises a total of $|T|$=111 S (now Siemens Gamesa) SWT-3.6-120 type wind turbines, collectively boasting an installed capacity of 399.6 megawatts (MW). These wind turbines are strategically connected to a substation platform, positioned on the western side of the windfarm, as well as to each other via inter-array cables.

The inter-array subsea cables utilized within the Anholt OWF are standard medium voltage range 33 kilovolt (kV), each tailored to the specific requirements of the number of wind turbines it connects between the designated points and the offshore transformer. They are cross-linked polyethylene (PEX/XLPE) insulated and embedded not less than 1 m into sea bottom. Three variations of inter-array cables are installed: 150cu, 240cu, and 500cu, with cross-section levels adjusted accordingly. Although the precise cable specifications used in the Anholt OWF are not disclosed, we estimate cable capacity in terms of number of total upstream turbines that can be connected. These estimates are listed in Table 1. Despite various sources providing cost estimates for the specified cable cross-sections, it is important to note that these figures were calculated before

the COVID19 pandemic, rendering them somewhat unreliable [6]. For the purposes of simplification in simulation studies, the proportional estimations presented in Table 1 have been selected for use.

Moreover, the connection between onshore facilities and the offshore transformer (33kV to 220kV) platform is facilitated by a 220 kV cable, ensuring efficient transmission of electricity generated by the wind turbines [16]. An illustrative layout of the entire Anholt OWF is provided in Fig. 5a, offering a comprehensive overview of the spatial arrangement and infrastructure of the wind farm.

## 3 Proposed Approach

Our approach to solving the full farm cabling optimization problem begins with the constructing a candidate solution (i.e. start tree) that connects every node of interest (i.e. wind turbine) to a desired starting point, typically the substation. Section 3.1 provides a detailed summary of the proposed topology assignment strategy for constructing the OWF's start tree. The availability of different types of cables transforms the problem into a CMST scenario, where each candidate connection must be assigned specific capacities. This aspect is elaborated upon in Sect. 3.2. Lastly, Sects. 3.3 and 3.4 describe how the efficient optimization of the layout involves the application of a non-linear multi-objective solver, specifically the well-known NSGA-II, with a specially designed, domain-specific, genetic operator.

### 3.1 Cable Topology Assignment Based on Radial Space Partitioning

We define an encoding for our multi-objective optimization problem as shown in Eq. (9), where $n$ is a parameter that controls the domain size of a candidate solution $\mathbf{x}$ by specifying the total number of OWF radial sections into which we wish to cluster all turbines. Conceptually, based on the hierarchical problem taxonomy proposed in [1], $n$ can be used to define the number of substation problems one wishes to decompose the full farm problem into.

$$\begin{aligned} \mathbf{x} &= [x_0, x_1, \ldots, x_{l-1}] \\ x_i &\in \{0, 1, \ldots, n-1\} \end{aligned} \tag{9}$$

Within this encoding, each turbine placeholder $x_i (0 \leq i \leq l-1)$ can be allocated to an arbitrary section, where $l$ is the length of the encoding, equal to the number of turbines.

To construct an initial candidate/start tree, turbines are first sorted radially by their angle relative to the substation and then partitioned into $n$ equally-sized subsets according to angle. If $n$ does not evenly divide the number of turbines, some sections will contain $\left\lfloor \frac{l}{n} + 1 \right\rfloor$ turbines, as illustrated in Fig. 1a.

To generate the cable layout (decode) from the encoding we create $n$ separate minimum spanning trees, using Kruskal's algorithm [11]. The distance metric

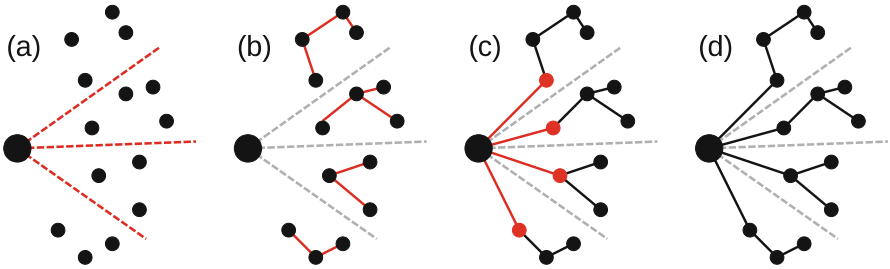



**Fig. 1.** Steps in constructing a full farm spanning tree for cable layout, shown on a simplified wind farm with fourteen turbines. Step (*a*) is only applied for the start tree (i.e. initial solution candidate).

employed is the Haversine formula [10], resulting in a spanning forest consisting of $n$ disjoint trees, as shown in Fig. 1b.

In each section, the turbine closest to the substation is defined as the *subroot*. An additional cable is added to the subsection to connect its associated subroot to the substation. The substation is defined to be the *root*. This creates a single spanning tree of the set of all turbines plus the substation, where the substation is the root and there are $n$ subroots. This layout is represented in Fig. 1c and the resulting overall spanning tree in Fig. 1d.

### 3.2 Cable Type Assignment

The electrical system design of the Anholt OWF incorporates three different types of cables, each distinguished by its cross-sectional size. These cross sections are designed to support the maximum current load that the cable can carry. In Table 1, we present each cable type along with the maximum number of upstream turbines that can be connected using it and the estimated costs assumed for our case study simulations. Here, the maximum turbine load $T_{max}$ represents the capacity that constrains the spanning tree in the CMST problem.

**Table 1.** Selected Cable Specifications Used in Anholt Case Study

| Cross Section ($mm^2$) | Maximum Turbine Load $T_{max}$ | Cost ($M£/km$) |
|---|---|---|
| 150 | 4 | 1 |
| 240 | 7 | 1.5 |
| 500 | 15 | 2.5 |

The cable chosen for each edge is the lowest cost cable with a maximum turbine load greater than or equal to the required load on the edge. This process is illustrated in Fig. 2.

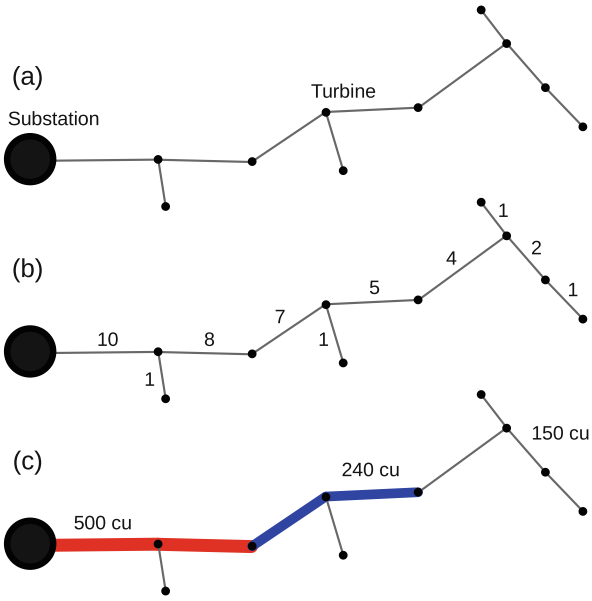



**Fig. 2.** Steps in cable type assignment, shown on one section of the Anholt layout.

### 3.3 Mutation

To ensure that the proposed representation is compatible with a wide range of metaheuristic solvers, we introduce a mutation operator designed to explore a local neighborhood through incremental adjustments. This operator is structured to randomly select with equal probability between two pre-defined movements, ensuring a uniform random distribution in their application:

1. **Move One Vertex:** As illustrated in Fig. 3, a random vertex $i$ (i.e. position $x_i$ in the encoded vector) is chosen and allocated to the previous or the next radial section (this corresponds to the its value either increasing by 1 or decreasing by 1, limited by the bounds 0 and $n-1$).
2. **Swap Two Vertices:** Shown in Fig. 4, this mutation involves swapping the section assignments of two random vertices $i$ and $j$ that reside in adjacent sections (i.e. $x_i \leftrightarrow x_j$ if $|x_i - x_j| = 1$).

The constraint violation checks align with the problem formulation from Sect. 2.3 and are as follows:

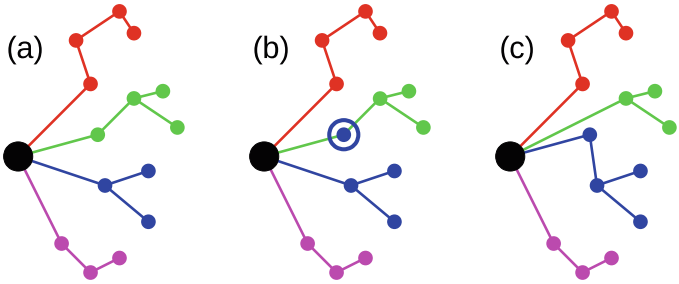



**Fig. 3.** Illustration of "move one vertex" mutation operation.

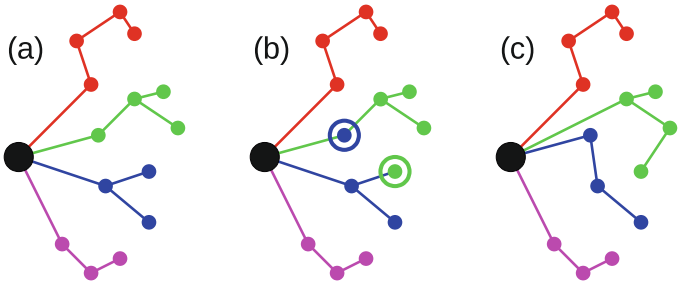



**Fig. 4.** Illustration of "swap two vertices" mutation operation.

1. **Number of Sections:** The total number of non-empty sections must be exactly $n$ as per Eq. 6. This constraint ensures consistency across different optimization scenarios, allowing each configuration (i.e. setting of $n$) to be evaluated and compared independently.
2. **Section Capacity:** The number of vertices in any section must not exceed the maximum turbine load associated with the highest-rated cable type as per Eq. 7. In the case of Anholt, the setting $v_{max} = T_{max}(500cu) = 15$ prevents the scenario where the cable connecting the root to the subroot of a subtree is overloaded.
3. **Graph Planarity:** The resulting graph must be planar as per Eq. 8 to avoid the added costs and technical challenges associated with overlaying cables. This constraint is verified using a standard line-segment intersection test based on the cross-product, given the manageable number of line segments involved.

Section-related constraints (i.e. constraints no. 1 and 2) are handled within the mutation operator, as movements that result in their violation are rejected and alternative movements are explored until a suitable mutant is created. Con-

straint no. 3 (planarity) is treated in a softer manner as its violation is only signaled to the overarching control (meta-)heuristic. These mechanisms and constraints ensure that nearly all metaheuristic search processes can explore viable alterations while strictly complying with operational and technical parameters, thus maintaining the integrity and feasibility of the optimized layouts.

### 3.4 Multi-objective Optimization

To apply the proposed representation for optimizing the dual-objective problem we presently consider, we integrate our bespoke mutation operator in the NGSA-II [5] solver. We use an initial seed population of 1 that only contains the start tree and scale up to an operational population cap of 100 individuals. Each optimization run evaluates a fixed number of $10,001$ candidate solutions (i.e. 100 generations). Crossover operations are disabled as they are too disruptive; our aim in this study is to elicit the performance of the proposed domain-specific mutation operator.

We use the "Feasibility First" constraint handling option from the NSGA-II implementation in *pymoo* [2], which prioritizes feasible solutions during non-dominated sorting throughout the optimization.

### 3.5 Random Walk

To gain basic insights on the benefits of applying NSGA-II and mitigate the possibility that the final multi-objective result quality is solely predicated on the repeated application of the mutation operator, we also implemented a random walk (RW) strategy for this problem. Similar to NSGA-II, the random walk uses the bespoke mutation operator introduced in Sect. 3.3, thus ensuring the satisfaction of section-related constraints. RW is configured not to update the step if a newly generated solution violates the graph planarity constraint – i.e., Eq 8, keeping the search within the feasible space at all times.

Random walks start from the same initial candidate as the NSGA-II runs and explore for a total of 10,001 evaluated solutions (10,000 steps), including infeasible ones, matching the evaluation budget of the NSGA-II runs.

## 4 Results

The real-life Anholt OWF layout serves as the baseline design for our case study. The cabling configuration is illustrated in Fig. 5a. For comparison, in Fig 5b, we illustrate the result obtained when applying the traditional Esau-Williams (EW) heuristic [7] to solve the Anholt CMST problem. Notably, the resulting EW layout does not satisfy the planarity constraint, as several line segments (i.e. cables) that intersect or overlay one another. This indicates that the Anholt scenario is too complex for a basic/direct application of EW and would require more extensive domain expert input/modelling.

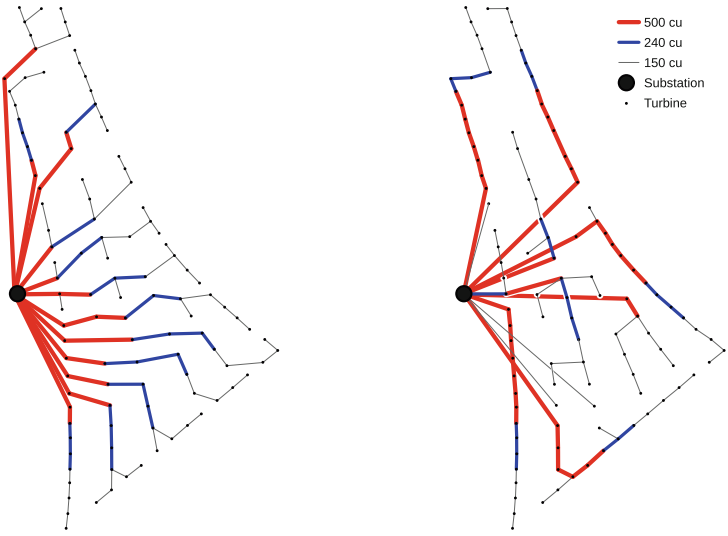



**Fig. 5.** (a) Model of real-life Anholt wind farm layout: cost = 250.8; robustness = 0.9518. (b) Esau-Williams Solution for max capacity 15 turbines: cost = 232.7; robustness = 0.9372.

For both NSGA-II and RW, we conducted 17 numerical experiments by setting the number of sections parameter $n$ from a minimum of 8 to a maximum of 24. The minimum value is dictated by the $T_{max}(500cu) = 15$ scenario setting, as dividing the $|T| = 111$ Anholt turbines into 7 non-empty clusters, would result in at least one cluster with more than 15 turbines. The maximum value of $n$ was set at double the branching factor of the real-life Anholt layout. Given the stochastic nature of our solvers, each of the $2 \times 17 = 34$ numerical experiments was repeated 50 times.

The objective space projections (i.e. Pareto fronts) of the final Pareto non-dominated solution sets (PNs) obtained by our multi-objective approach for each of the 17 NSGA-II numerical experiments are shown in Fig. 6a. As expected, the results show a positive correlation between the number of sections and the overall robustness of the OWF cable layout. Furthermore, for each setting of $n$, the shape of the associated Pareto front (PF) indicates the ability of our approach to identify cost vs. robustness trade-offs, even though robustness improvements become marginal (albeit at a very high level) for $n \geq 15$. Maximizing $n$ would also maximize robustness with the trivial extreme case being each turbine individually attached to the substation for $n = |T| = 111$.

The importance of designing a solving strategy that allows for easy experimentation with several substation branching factors (i.e. settings of $n$) is underlined by the plot in Fig. 6b that shows the combined multi-section PF of all 17 individual optimization runs for both NSGA-II and RW. It is noteworthy that

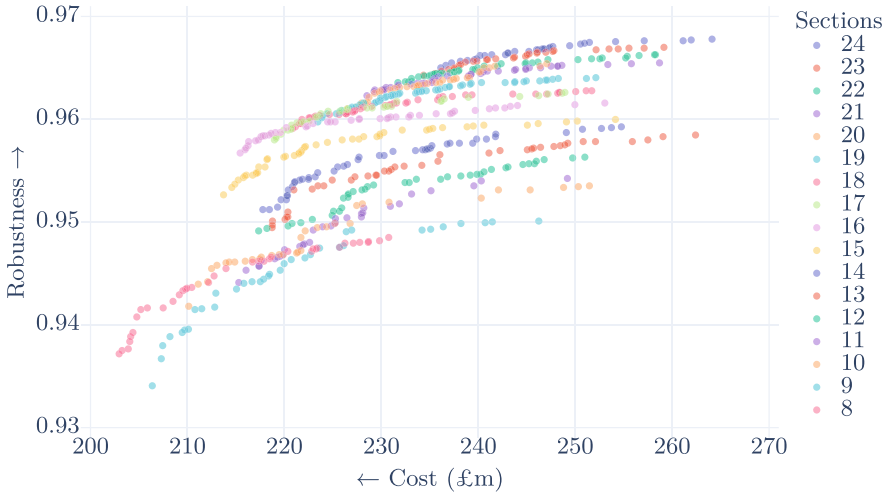
only 12 settings of $n$ contribute to the combined multi-section NSGA-II PF as the optimization results obtained using $n = 9$, 11, 12, 13, and 14, are entirely dominated. Given that $n = 12$ is the branching factor in both the real-life Anholt layout and the EW solution, the plots in Fig. 6 showcase the ability of our approach to quickly provide decision makers with valuable insights on the art of the possible, even with a basic deployment (i.e. no parameter tuning for the solver).

Interestingly, in the combined multi-section PF there is a large discontinuity whereby to obtain non-dominated solutions with a lower cost than those achievable for $n = 16$, there is a need to drop down to $n = 10$ and $n = 8$. This discontinuity is punctuated only by a few solutions from the parameter setting $n = 15$. Figure 7 shows the two optimal layouts for $n = 10$ and $n = 16$ that define this large discontinuity characterized by a relatively small increase in cost but a relatively large jump in robustness. Visually, we see this abrupt regime change correlates to the introduction of a design which eschews $500cu$ cables – used in Fig. 7a – in favor of keeping each section limited to the capacity of the $240cu$ cable as shown in Fig. 7c. One of the few $n = 15$ in-between solutions is highlighted in Fig. 7b and it features a single section linked using a $500cu$ cable.

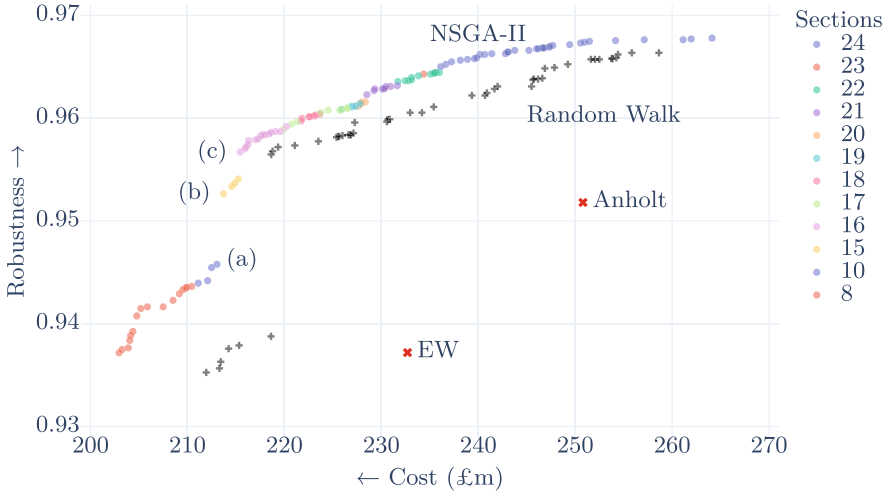
The better overall performance of NSGA-II when compared to the random walk is directly observable in the combined multi-section PFs from Fig. 6b which are extracted from all the numerical experiments we have carried out. Additionally, for each solver, we constructed 50 individual multi-section PFs by aggregating the 17 PFs ($8 \leq n \leq 24$) corresponding to a given independent run number. When analyzing the individual multi-section PFs using the relative hypervolume indicator[2] [18], NSGA-II achieved an average value of 0.699 (with a 0.006 standard deviation) and RW achieved an average value of 0.586 (with a standard deviation of 0.003). This superior average performance of NSGA-II was confirmed by a one-sided Mann-Whitney U test [13] with a 0.01 significance level (p-value $< 0.00001$, Z-score $= 8.61383$).

As previously mentioned, solutions that violate the graph planarity constraint are still presented to NSGA-II, which applies a feasibility-first approach that discards infeasible solutions once feasible solutions are found. Across NSGA-II runs, out of the 10,001 evaluated solutions, an average of 32.7% solutions were feasible. In contrast, RW treats graph planarity as a hard constraint during the search and as a result, only an average of 18.7% of evaluated solutions were feasible. The percentage of feasible solutions does not vary significantly across experiments with different settings for the number of sections parameter (standard deviation of 1.8% for NSGA-II and 0.5% for RW). It is noteworthy that while the proportion of feasible solutions is rather low due to the relatively high chance of mutations causing cable overlays, the comparative results indicate that the search space can be explored quite effectively by a robust solver like NSGA-II when applying strict feasibility-first constraint handling.

[2] The ideal point was set at (200, 0.97) and the anti-optimal/nadir reference point was set at (270, 0.93).




(a) The PFs of the 17 NSGA-II numerical experiments for $8 \leq n \leq 24$. For each section setting, the PF is created from all the non-dominated solutions discovered across the corresponding 50 independent runs.




(b) The combined multi-section PF of all NSGA-II runs (color-coded by section). Black + points denote the corresponding combined multi-section PF of all RW runs

**Fig. 6.** Comparative Pareto fronts of numerical experiments. In subfigure (b), the indicators (a), (b), and (c) refer to the layouts in Fig. 7, whereas "Anholt" refers to the real-life OWF layout and "EW" refers to the Easu-Williams solution; both detailed in Fig. 5.

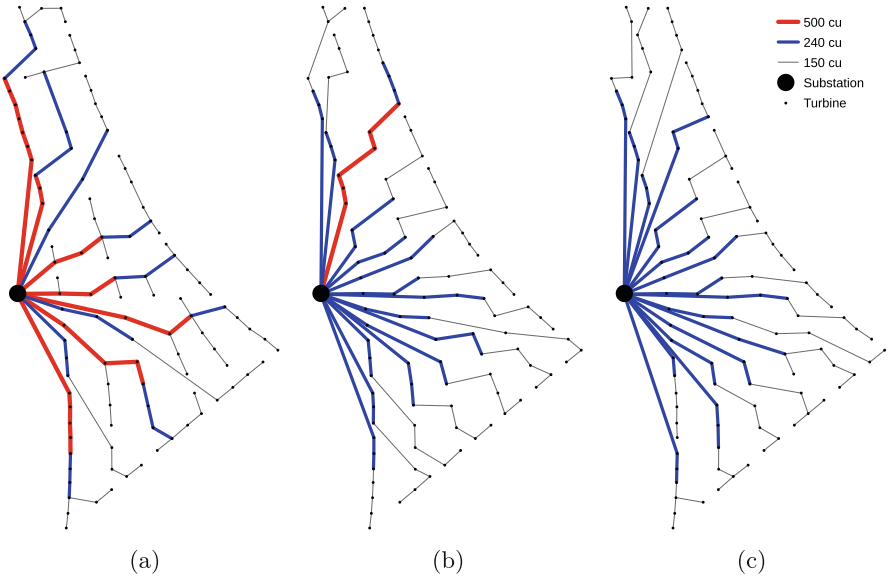



**Fig. 7.** Layouts of the three letter-identified solutions from the combined NSGA-II PF in Fig. 6b. (a) $n = 10$; cost=213.0; robustness=0.9458. (b) $n = 15$; cost=213.7; robustness=0.9526. (c) $n = 16$; cost=215.0; robustness=0.9567.

## 5 Conclusion and Future Work

The inter-array cable layout optimization problem for OWFs has been addressed through the development of a multi-objective formulation that incorporates both robustness and cable cost considerations. Building on previous efforts in the field, our methodology introduces network robustness as a critical metric in the design of OWFs, moving beyond the conventional reliability-focused assessments.

As a preliminary investigation, this newly developed approach was compared with the Esau-Williams heuristic and the original design of the Anholt OWF. This comparison demonstrated the significant advantages of our methodology, highlighting its potential to substantially enhance the strategic development of OWFs that are simultaneously robust and economically viable. Furthermore, the results of our case study highlighted the importance of choosing an appropriate substation branching factor when aiming to optimize a radial OWF topology, a relevant insight to both wind farm designers and equipment manufacturers.

Looking ahead, we plan to further refine our approach by integrated more sophisticated subgraph connection strategies, enabling us to explore more complex scenarios with multiple substations and/or loop-based topologies. Simultaneously, as we develop our understanding of the complexities involved in optimizing offshore wind farm infrastructures (e.g., bathymetric data, ecological impact), we aim to refine our problem definition by including more relevant constraints as well as potentially new objectives. Lastly, for the current experi-

mental setup, one potential enhancement would be to experiment with (combinations of) different constraint handling options provided by *pymoo*: constraint violation as penalty/objective, $\epsilon$-constraint handling, repair.

## References


1. Berzan, C., Veeramachaneni, K., McDermott, J., O'Reilly, U.: Algorithms for cable network design on large-scale wind farms (2011). https://alfagroup.csail.mit.edu/sites/default/files/documents/msrp_techreport.pdf. Accessed 28 Feb 2024
2. Blank, J., Deb, K.: Pymoo: multi-objective optimization in python. IEEE access **8**, 89497–89509 (2020). https://doi.org/10.1109/ACCESS.2020.2990567
3. Chandy, K.M., Lo, T.: The capacitated minimum spanning tree. Networks **3**(2), 173–181 (1973)
4. Dahmani, O., Bourguet, S., Machmoum, M., Guerin, P., Rhein, P., Josse, L.: Optimization and reliability evaluation of an offshore wind farm architecture. IEEE Trans. Sustain. Energy **8**(2), 542–550 (2016). https://doi.org/10.1109/TSTE.2016.2609283
5. Deb, K., Pratap, A., Agarwal, S., Meyarivan, T.: A fast and elitist multiobjective genetic algorithm: NSGA-ii. IEEE Trans. Evol. Comput. **6**(2), 182–197 (2002). https://doi.org/10.1109/4235.996017
6. Dicorato, M., Forte, G., Pisani, M., Trovato, M.: Guidelines for assessment of investment cost for offshore wind generation. Renew. Energy **36**(8), 2043–2051 (2011). https://doi.org/10.1016/j.renene.2011.01.003
7. Esau, L.R., Williams, K.C.: On teleprocessing system design, Part ii: a method for approximating the optimal network. IBM Syst. J. **5**, 142–147 (1966). https://doi.org/10.1147/sj.53.0142
8. Grainger, J.J., W.D., S.: Power system analysis. McGraw-Hill (1999)
9. Guo, H., Zheng, C., Iu, H.H.C., Fernando, T.: A critical review of cascading failure analysis and modeling of power system. Renew. Sustain. Energy Rev. **80**, 9–22 (2017). https://doi.org/10.1016/j.rser.2017.05.206
10. Inman, J.: Navigation and nautical astronomy: for the use of British seamen. F. and J, Rivington (1849)
11. Kruskal, J.B.: On the shortest spanning subtree of a graph and the traveling salesman problem. Proc. Am. Math. Soc. **7**(1), 48–50 (1956). https://doi.org/10.2307/2033241
12. Li, J., et al.: A hybrid cable connection structure for wind farms with reliability consideration. IEEE Access **7**, 144398–144407 (2019). https://doi.org/10.1109/ACCESS.2019.2944888
13. Mann, H.B., Whitney, D.R.: On a test of whether one of two random variables is stochastically larger than the other. Ann. Math. Stat. **18**(1), 50–60 (1947). https://doi.org/10.1214/aoms/1177730491
14. Pérez-Rúa, J.A., Cutululis, N.A.: Electrical cable optimization in offshore wind farms-a review. IEEE Access **7**, 85796–85811 (2019). https://doi.org/10.1109/ACCESS.2019.2925873
15. The Crown Estate, O.C.: Guide to an offshore wind farm (2019). https://www.thecrownestate.co.uk/media/2861/guide-to-offshore-wind-farm-2019.pdf. Accessed 17 Apr 2024
16. Wind, R.: Anholt offshore wind farm project description (2010). https://ens.dk/sites/ens.dk/files/Vindenergi/project_description.pdf. Accessed 28 Feb 2024

17. Zhang, X., Chi, K.T.: Assessment of robustness of power systems from a network perspective. IEEE J. Emerging Sel. Top. Circ. Syst. **5**(3), 456–464 (2015). https://doi.org/10.1109/JETCAS.2015.2462152
18. Zitzler, E., Thiele, L.: Multiobjective optimization using evolutionary algorithms — A comparative case study. In: Eiben, A.E., Bäck, T., Schoenauer, M., Schwefel, H.-P. (eds.) Parallel Problem Solving from Nature — PPSN V, pp. 292–301. Springer, Berlin, Heidelberg (1998). https://doi.org/10.1007/BFb0056872

# EvoVec: Evolutionary Image Vectorization with Adaptive Curve Number and Color Gradients

Egor Bazhenov(✉), Ivan Jarsky, Valeria Efimova, and Sergey Muravyov

ITMO University, Saint-Petersburg, Russia
tujh.bazhenov.kbn00@mail.ru

**Abstract.** Vector and raster graphics are the two main types of 2D images used in computer graphics. Raster graphics are images consisting of pixels (dots); vector images are created using mathematical objects such as lines, curves, and shapes. The main advantage of vector graphics is that they can be scaled without loss of quality, which is useful for advertising, design, frontend development, and other fields of application. At the moment, the issue of vectorization (conversion from raster to vector graphics) has not been fully resolved. There are two main approaches: deterministic algorithms and machine learning-based algorithms. Both of these types are not able to work with a color gradient and have other disadvantages, such as artifacts for deterministic algorithms, and extremely long working time and predefined curve number for machine learning-based algorithms. To solve the problems of existing solutions, we propose an evolutionary algorithm for image vectorization. Its main idea is to iteratively improve vector images using mutations and crossover. The proposed algorithm does not require any necessary parameters other than the original image and can process color gradients. The results of comparison with existing solutions show that our algorithm qualitatively and quickly vectorize images. Particularly, our approach outperforms others in terms of pixel-by-pixel MSE by 15%. The implementation is publicly available (https://github.com/EgorBa/EvoVec-Evolutionary-Image-Vectorization).

**Keywords:** Image vectorization · evolutionary algorithm · machine learning · vector graphics

## 1 Introduction

Raster and vector graphics are the primary forms of visual representation employed in computer graphics. Both types of graphics have their strengths and weaknesses and are used in various areas of design and visuals, depending on the specifics of the task.

Raster graphics or bitmaps consist of pixels (dots), each having a specific color and coordinates on the screen. The color is commonly set by three channels: red, green and blue (RGB). The bitmaps quality depends on their resolution and number of pixels, so this type of graphics is not scalable.


M. Affenzeller et al. (Eds.): PPSN 2024, LNCS 15151, pp. 383–397, 2024.
https://doi.org/10.1007/978-3-031-70085-9_24

Vector graphics are images created using mathematical objects such as lines, curves, and shapes. Each shape is called a path and is defined mathematically using a coordinate vector. Vector images can be scaled without loss of quality [2], as they do not depend on the number of pixels. Vector images are usually saved in SVG [20] format.

The process of converting a vector image to a bitmap image is called rasterization [13], and the one from bitmap to vector image is called vectorization [25]. A rasterization process has already been studied in great detail, because there are a huge number of methods for its implementation [17]. However, the task of vectorization remains relevant, since existing solutions do not yet manage this task well.

There are two main approaches for vectorization: deterministic algorithms and machine learning-based algorithms. Deterministic algorithms include existing solutions from libraries or online services [9]. Their main disadvantage is that the quality of vectorization is not high, because these algorithms do not work with a gradient colors and sometimes create artifacts on a vectorized image. In turn, machine learning-based algorithms (like LIVE [15] or DiffVG [14]) can create structured vector images, so that is their main advantage. Their disadvantages include: long working time, inability to operate with a gradient, and the requirement for knowledge of the curves number in a vectorized image.

In this paper, we propose an another approach to vectorization problem, since existing solutions do not manage the task of vectorization sufficiently. We have achieved quality improvement through the use of an evolutionary algorithm. This allows to vectorize the image without additional parameters and using a gradient.

The result of this paper is an evolutionary algorithm that can vectorize a raster image using a variable curve number. Specifically, the main contributions are as follows:

1. The proposed algorithm does not require input parameters for vectorization.
2. The algorithm operates with a color gradient, which improves the vectorization quality.
3. Vectorized image does not require predefined curve number and has fewer curves than the deterministic algorithm result.
4. It works faster than existing algorithms based on machine learning.

For more information about existing solutions, see the Related Works Sect. 2. The Methods Sect. 3 will describe in detail the algorithm itself and the subtleties of its implementation. In the Comparisons and Experiments 4, the results of comparing the proposed algorithm operation with existing solutions and ablation study are presented. The Conclusion Sect. 5 summarizes the paper.

## 2 Related Work

In this section, we summarize previous approaches and present works that are closely related to our article. The existing surveys [8,23] proposes vectorization methods classification. According to it, the two main groups of methods

are deterministic algorithms and machine learning-based algorithms. Along with these algorithms, we also discuss evolutionary approaches in this section.

### 2.1 Vectorization Algorithms Based on Machine Learning

There are lots of machine learning-based vectorization algorithms, however some of them are universal (designed to vectorize any images) and the others are customized for a predefined type of images. Non-universal algorithms can only be used for highly specialized tasks and therefore they are not of particular interest for the general problem of image vectorization. Among universal algorithms we selected DiffVG [14] and LIVE [15] as the most widely used [11,24] existing approaches.

**DiffVG**. Many machine learning-based vectorization algorithms use a process of a vector image gradual optimization so that its rasterization is similar to a bitmap image. This optimization is possible due to the method of differentiable rasterization of a vector image proposed in the DiffVG [14] paper. In this paper, the authors propose a functional operator that accepts a vector image as input and outputs its raster analog. The main purpose of this operation is to further transfer the resulting bitmap image to the loss function, to call the error back propagation algorithm and to calculate gradients for the parameters of the original vector image. Then, according to the standard practice, an optimization step can be performed to result in vector parameters updating. Thus, the vector image is changed according to the specified loss function, and if, for example, this loss function is MSE, then the vector image is optimized towards more similarity with the custom raster analog.

One of DiffVG advantages is that it does not have a rigid binding to any predefined type of vector images and it is able to process any images. However, DiffVG is not able to dynamically determine which shapes or segments should be added or removed from a vector image. The DiffVG method operates on vector images, preserving their original structure and changing only the values of the shape parameters, for example, color, point coordinates, and stroke width. This is one of the reasons why using DiffVG without additional improvements does not result in the images most closely resembling the original ones.

At the moment, the following innovations are used among the algorithms known: 1) according to a pre-selected scheme, step-by-step addition of the shapes to a vector image is applied (for example, in LIVE method), 2) DiffVG is used in neural networks for the purpose of differentiable transition from vector to raster domain, 3) DiffVG is used for final optimization of a vector image pre-generated by a separate neural network.

To the best of our knowledge, there are still no methods related to DiffVG that could remove unnecessary shapes and segments from a vector image.

**Im2Vec**. One of the first approaches integrating DiffVG into neural networks is the Im2Vec model [21], its main purpose is image vectorization. The model is a VAE (Variational autoencoder) transforming an input bitmap into a hidden space and then generating the parameters of the vector image corresponding to

the input bitmap from the hidden space. The network is trained using DiffVG, due to which the generated vector image is converted to a raster image and a raster loss function is applied, evaluating the similarity of the generated raster image with the original raster image. The main disadvantage of this model is that the model is able to correctly vectorize only images similar to training set. At the same time, according to our experiments, this model is not able to synthesize complex vector images consisting of a large number of shapes and segments.

**LIVE**. The iterative vectorization approach is used in LIVE [15] method. Before running this algorithm, the scheme for adding new shapes must be specified. This scheme determines how many new shapes will be added at each stage of optimization. Instead of applying shapes to the canvas in random places, the algorithm searches for such monochrome areas in the image that are not covered by the desired shape.

The method also introduces two new loss functions. The first penalizes the algorithm for creating self-intersecting shapes, which in practice can lead to the appearance of unnatural patterns. The second loss function seeks to estimate the coincidence of the rasterization of vector shapes with the real area of the original raster image more correctly than by MSE. MSE approach was shown to be problematic due to the following considerations. Vector images are often a composition of various shapes superimposed on each other. The LIVE algorithm suggests adding shapes to a vector image step by step, starting with the outer large shapes, rather than the inner smaller ones. However, using the mean square error (MSE) as a loss function between the current external vector shape and the real raster area containing this external shape and its internal subfigures is incorrect. Instead, it is proposed to use the UDF (User-Defined Function) loss function, which evaluates how precise a vector shape corresponds to the desired boundaries and how correct its color is, rather than the average color of this external shape and its internal ones. The proposed losses allow more effective evaluating and improving the intermediate results of the LIVE algorithm when vectorizing images.

However, the main problem with the LIVE method is its very low image generation speed. In addition, the quality of vectorized images in terms of visual similarity to the original bitmap is still not precise.

### 2.2 Deterministic Algorithms

At the moment, a huge number of deterministic algorithms for vectorization exists [7,9,19,27]. The main shortcomings of such algorithms are vectorization quality, as results are obtained with a partial loss of initial information, and a large number of unnecessary paths used in resulting vector image. These algorithms advantages include high vectorization speed and the absence of necessary parameters tuning to improve vectorization.

In our approach, deterministic algorithms are used to get an initial population. This helps to ensure that the initial vectorization quality of our algorithm will not be worse than the quality of the deterministic ones.

### 2.3 Evolutionary Vectorization

An evolutionary algorithm [1] is a numerical optimization method based on the principles of natural selection and evolution. To obtain more adapted individuals, evolutionary algorithms use mutations [28] and crossovers [22]. A mutation is a change in one particular individual using a special algorithm, that can improve or worsen it. Crossover is an operator which involves combining features from two parent individuals to create offspring. By exchanging features between parents, crossover promotes exploration of the search space and facilitates the creation of new potential solutions. One of the evolutionary algorithm main characteristics is the selection of the initial population [12], because all new individuals will be built on the basis of it. The important parameter of the evolutionary algorithm is the elite percentage [4]—the percentage of individuals selected from the current population for further development. The size of the population [10] has a great influence on the evolutionary algorithm quality, because the chance of getting a more adapted individual with a large population size is higher, therefore it is necessary to select the optimal value for it based on available computational power.

Vectorization of bitmaps using evolutionary algorithms was discussed in the paper [3]. The described approach has a number of significant drawbacks that do not allow it to be used for vectorization. The vectorization process has long working time, and the result is very different from the original images, because the algorithm works with discrete geometric shapes and use masks. This approach makes it possible to add different styles to vectorized images by user, but vectorized image does not have a high vectorization quality.

In our approach, we decided to use evolutionary algorithm, because image vectorization is a task of optimizing the resulting image quality. Our algorithm is more universal and makes it possible to get a vector image similar to the original.

## 3 Method

The main part of the proposed solution is the evolutionary algorithm, which is used to solve optimization problem. Image vectorization aims to optimize vector image to exactly repeat raster image, thus, it is an optimization task. A vectorized image is an individual to which appropriate mutation and crossover operators are applied to obtain a higher quality. For vectorization, we also create a population from various vectorization options and further improve and select only the best ones.

### 3.1 The Initial Population

Several options were considered to create the initial population. After conducting a number of experiments, it was concluded that the result of a deterministic algorithm should be copied to form an initial population of $n$ individuals.

We also studied and tested various Python libraries for initial vectorization (Pixels2Svg[1], Potrace[2], SvgTracer[3]). SvgTracer is the best library we have found for our task, because it is the fastest and its results have the highest quality.

### 3.2 Mutations and Crossovers

To achieve the best quality of vectorization, we use four types of mutations: needle mutation, probabilistic gradient connection of paths based on color differences, probabilistic path deletion, and probabilistic path segment deletion. As crossover, we use a random exchange of paths.

A needle mutation is a probabilistic change in the coordinates of path segments added for preventing path overgrowth and shape correction. This mutation works in such a way that the needle mutation coefficient is added or subtracted to a probabilistically selected coordinate of a path. At the same time, it is limited by the size of the image, and this mutation also has its own subtypes such as: a constant value of the coordinate change and a gradual decrease in the value by which the coordinate changes. For more complex vectorization tasks, the second type of the mutation should be used, and for the rest—the first one. In Fig. 1, example A shows the problem that this mutation is designed to solve.

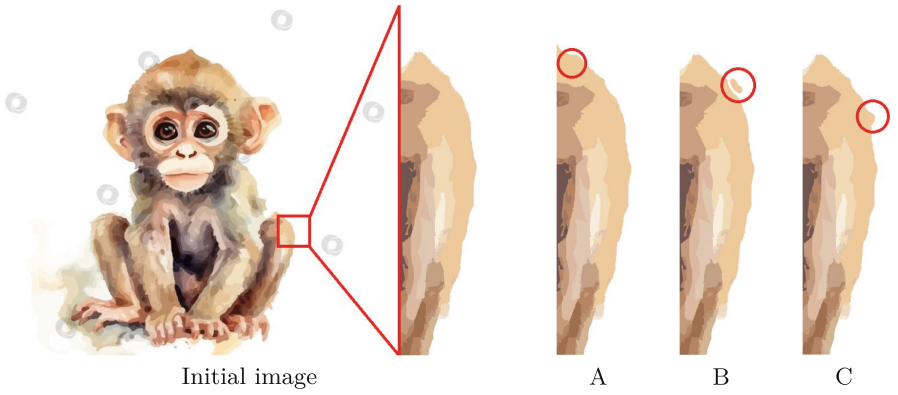



**Fig. 1.** Example of different vectorization problem and their solution with different mutation. A—example of incorrect initial coordinates of a path. This problem is solved by needle mutation. B—artifact of vectorization. Mutation of dropping path removes this artifact. C—extra segment in the path. Mutation of dropping segment smoothes the curve.

While other mutation, the path is deleted with a certain degree of probability. The same approach is used for deleting the segments of the path. These

[1] https://pypi.org/project/pixels2svg/.
[2] https://pypi.org/project/pypotrace/.
[3] https://pypi.org/project/svgtrace/.

mutations help to eliminate artifacts created during the initial vectorization. The mutation for deleting a path has an additional parameter responsible for the size of the deleted area, because large elements of the image should not be deleted. For example, in Fig. 1, Figures B and C illustrate this issue with artifacts.

The main purpose of the fourth mutation is the gradient connection of two paths based on the difference in their colors, set by a threshold, which can be tuned [26]. This mutation is applied with a certain probability, and the connection occurs using a gradient [6] from the color of the first path to the color of the second. Figure 2 shows how the algorithm works.

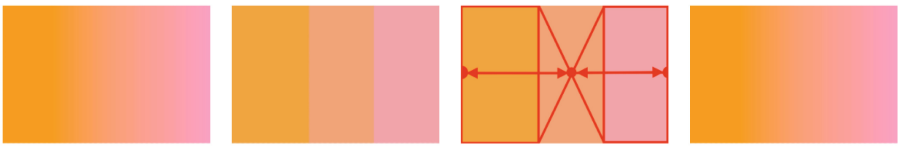


**Fig. 2.** Example of how mutation works for gradient path joining. From left to right: the original image; the image vectorized by a deterministic algorithm; the process of applying the gradient mutation; the vectorized image.

The edge of the shape area is taken as the gradient start points. If there are more than two parts, then the area's edge is taken only from the first part and the last one. For other parts, the center of mass is taken. This mutation creates the images with gradient coloring, which helps render the colors of the original image more accurately.

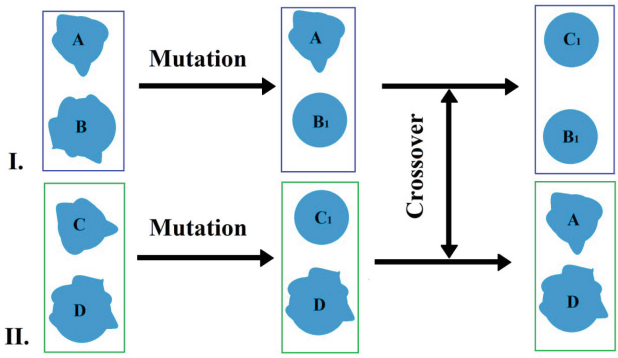



**Fig. 3.** Example of random path exchange crossover.

We use only one crossover for probabilistic path exchange. To achieve better quality, two individuals can exchange random paths with a certain probability. The example is presented in Fig. 3.

Assume that two individuals $I$ and $II$ appeared after the mutation, let each individual has one good path and the other has bad path. Next, a crossover exchange of random paths is applied. For example, it chooses path $A$ from individual $I$ and path $C_1$ from $II$, then swaps them. As a result, two new individuals are created, and individual $I$ is much better due to the crossover.

### 3.3 Selection Function

Three selection function [16] were investigated. The selection function 1 is the pixel-by-pixel difference between the original bitmap image $N$ and the current generated one $M$, formulated as follows:

$$F_1(N, M) = \sum_{i=0}^{w} \sum_{j=0}^{h} |N_{i,j} - M_{i,j}|, \tag{1}$$

where $w$—width of image, $h$—height of image.

The second selection function 2 is the same, but the difference between pixels is calculated exponentially.

$$F_2(N, M) = \sum_{i=0}^{w} \sum_{j=0}^{h} e^{|N_{i,j} - M_{i,j}| * \lambda + c}, \tag{2}$$

where $\lambda = \frac{50}{255}$, $c = 1$ to get a more sensitive difference color map.

In the third selection function 3, the difference between pixels in $F_1$ is squared.

$$F_3(N, M) = \sum_{i=0}^{w} \sum_{j=0}^{h} (|N_{i,j} - M_{i,j}| * \lambda)^2. \tag{3}$$

where $\lambda = \frac{1}{255}$ for normalization pixel colors.

This selection function is close in meaning to MSE [5]. As a part of the testing on different images, it was decided to focus on the third variant of selection function. This allows to account for stronger color differences better, while if the color is close to the original one, then the value function will be small.

In order to calculate the value of the selection function, a rasterizing of a vector image is required. We also studied and tested various Python libraries for rasterization ( Aspore[4], Wand[5], CairoSvg[6]). CairoSvg is the best library for this task due to its speed and quality of rasterization.

[4] https://www.aspose.com/.
[5] https://pypi.org/project/Wand/.
[6] https://pypi.org/project/svgtrace/.

### 3.4 Color Correction

To improve the algorithm operation, the color correction function is added. Often, the original colors of the deterministic algorithm may differ from the original ones. To solve this problem, a special color correction algorithm was created. It selects pixel colors for which the difference between them and the original ones is greater than a certain hyperparameter responsible for the sensitivity of the correction. Next, it selects all paths containing such colors and chooses the color for this path, which prevails in this section of the image. The implementation of the method can be seen in the algorithm's details.

## 4 Comparisons and Experiments

As a part of the work, various experiments were carried out to identify the best parameters of the algorithm, as well as comparison with other existing solutions. Additional functions have been developed to simplify and improve the operation of the algorithm and the experimental setup has been described in detail.

### 4.1 Experimental Setup

The main hyperparameters and thresholds are shown below. The number of individuals in one population is 30. The percentage of the elite is 10%. The number of iterations of the algorithm is 300. These parameters were derived empirically based on a large number of experiments.

All of the experiments were launched on CPU with 16 GB RAM and 10 cores. No GPUs have been used.

### 4.2 Comparison with Machine Learning Algorithms

Firstly, we compare algorithms based on machine learning—LIVE and DiffVG—with our algorithm in terms of execution time, quality, and number of paths. Note that for LIVE $N = 16$, for DiffVG $N = 512$, where $N$ is the predefined number of paths. This constant was chosen in such a way that the working time of the machine learning algorithms was comparable to the working time of our algorithm. Hyperparameters of these algorithms are set to default values. For DiffVG and LIVE $N$ is necessary parameter, but our algorithm does not need it.

Table 1 shows that algorithms based on machine learning have a longer working time than our algorithm. For DiffVG and LIVE, the number of equal pixels in raster and vectorized images is significantly less, which indicates a low quality of vectorization. The number of paths for images vectorized by machine learning algorithms is equal to the initial constant, they have not to be compared by this parameter.

**Table 1.** Comparison of our algorithm with existing vectorization algorithms. The table shows metrics (vectorized image, working time, fitness value and number of paths) of algorithms: LIVE [15], DiffVG [14], Pixel2Svg, SvgTracer, and our algorithm.

| Initial image | LIVE | DiffVG | Pixels2Svg | SvgTracer | Our Algo | Metrics |
|---|---|---|---|---|---|---|
|  |  |  |  |  |  | Result |
| | 51.3 m. | 45.4 m. | 3.9 s. | 7.5 s. | 24.5 m. | Time |
| | 4760 | 4255 | 3278 | 1780 | 1547 | Fitness |
| | 35 | 512 | 1146 | 1790 | 985 | Paths |
|  |  |  |  |  |  | Result |
| | 33.5 m. | 38.4 m. | 17.4 s. | 1.9 s. | 22.1 m. | Time |
| | 4864 | 10720 | 4939 | 830 | 793 | Fitness |
| | 16 | 512 | 1789 | 2187 | 628 | Paths |
|  |  |  |  |  |  | Result |
| | 2.7 m. | 4.2 m. | 1 s. | 1.5 s. | 2.6 m. | Time |
| | 33 | 146 | 89 | 286 | 247 | Fitness |
| | 5 | 512 | 605 | 55 | 7 | Paths |
|  |  |  |  |  |  | Result |
| | 72.4 m. | 15.24 m. | 1.1 m. | 2.1 s. | 5 m. | Time |
| | 16290 | 15090 | 1626 | 1558 | 1550 | Fitness |
| | 16 | 512 | 6311 | 2947 | 1660 | Paths |
|  |  |  |  |  |  | Result |
| | 77.3 m. | 34.48 m. | 10.3 s. | 1.6 s. | 29.2 m. | Time |
| | 7503 | 15344 | 1501 | 1182 | 672 | Fitness |
| | 16 | 512 | 1522 | 798 | 438 | Paths |

### 4.3 Comparison with Deterministic Algorithms

Below we present a comparison of our algorithm with a deterministic algorithm based on various metrics. Pixel2Svg and SvgTracer were chosen as a deterministic algorithms, as they are powerful python libraries for image vectorization.

Table 1 clearly shows the advantage of our approach. The number of equal pixels between the original raster image and the vectorized one is higher due to gradient mutation. The proposed algorithm result contains fewer paths than others, due to the path deletion mutation, which means less weight of the vectorized image and interpretability. Our algorithm loses out in terms of execution time to deterministic algorithms, since at each iteration it is required to calculate the value of the selection function, which is a complex computational operation.

**Table 2.** Relative performance comparison of our algorithm with deterministic algorithms.

| Size | Pixels2Svg | SvgTracer | Comparison criterion is relative to our algorithm |
|---|---|---|---|
| $128 \times 128$ | -1569.8 ± 67.7 % | -500.7 ± 38.9% | Number of paths |
| | -69.6 ± 4.2 % | -16.9 ± 4.1 % | Fitness |
| $256 \times 256$ | -164.8 ± 13.5% | -103.6 ± 10.1% | Number of paths |
| | -27.9 ± 2.5% | -13.8 ± 1.3% | Fitness |
| $512 \times 512$ | -80.4 ± 9.4% | -67.1 ± 8.9% | Number of paths |
| | -18.8 ± 1.7% | -12.1 ± 1% | Fitness |
| $1024 \times 1024$ | -29.3 ± 5.7% | -24.2 ± 4.7% | Number of paths |
| | -9.6 ± 0.9% | -6.8 ± 0.2% | Fitness |

Our evaluation of the algorithm was conducted on image groups comprising 100 images each, with varying resolutions: $128 \times 128$, $256 \times 256$, $512 \times 512$, and $1024 \times 1024$ pixels. As demonstrated in Table 2, our algorithm exhibits a decrease in the computational paths required and an enhancement in the efficacy of the selection function when compared to the performance metrics of existing algorithms.

### 4.4 Ablation Study

In this section, we discuss the importance of the proposed mutations and how they affect the final result [18]. Each mutation is probabilistic, that is why the test results for the same hyperparameters may differ.

The main effect of the needle mutation is changing the path shape by changing their coordinates. This is useful if the vectorized shape does not look very similar to the expected one. Needle mutation was considered of two types. With a constant coefficient of change and uniformly decreasing. The first type proved to be better in the tests, and was later used in the research.

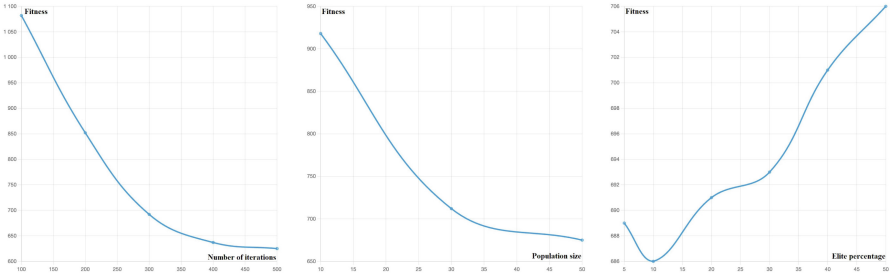
**Table 3.** An impact example of the individual mutations on the ultimate outcome. From left to right: the initial image for vectorization; the image is vectorized by our algorithm; the image is vectorized by our algorithm without needle mutation; the image is vectorized by our algorithm without mutation for deleting paths and segments; the image is vectorized by our algorithm without mutation of gradient fusion.

| Initial image | All mutations | W/o needle | W/o deletion | W/o gradient |
|---|---|---|---|---|
|  |  |  |  |  |
| Fitness | 628 | 736 | 796 | 698 |
| Num ber of paths | 793 | 793 | 2187 | 793 |

The removal of segments and paths is necessary to get rid of artifacts that occur during basic vectorization. It helps simplify the unnecessary complexity and the weight of the image, which is extremely convenient. However, there is an important addition: the deleted area should be less than or equal to the threshold responsible for the maximum square of the deleted area. This was done to avoid the problems associated with the complete removal of image important parts.

The last mutation, which gradiently connects the two or more paths, is also needed to simplify the structure of the images and for a clearer match to the originals due to the smooth transitions characteristic of bitmap images. The threshold between the color difference of paths can be tuned.

Thus, each of the mutations is responsible for a certain part of the improvement of the vectorized image. For example, Table 3 illustrates problems created by the absence of the proposed mutations. Their combination allows to achieve better vectorization results.




(a) Fitness dependence on the iterations number (b) Fitness dependence on the population size (c) Fitness dependence on the elite percentage

**Fig. 4.** Fitness dependencies on various factors.

An important part of the work was to choose the right population size, the number of algorithm iterations, and the elite percentage. Table 4 shows experiment results with various population size and the iterations number. Based on this table, it can be concluded that a larger number of individuals in the population and many algorithm iterations can create a vectorized image with higher quality. It is worth noting that with an increase in these parameters, the quality ceases to improve (see Fig. 4a,4b), so we chose the number of individuals in one population as 30 and the number of the algorithm iterations as 300.

**Table 4.** The dependence of the algorithm final result on the population size and the iterations number.

| | Iterations number of the algorithm | | | | | | |
|---|---|---|---|---|---|---|---|
| | 100 | 200 | 300 | 400 | 500 | | |
| Time | 3.4 m. | 6.7 m. | 10.1 m. | 13.4 m. | 16.9 m. | 10 | Population size |
| Fitness | 1095 | 1004 | 918 | 870 | 834 | | |
| Paths | 784 | 679 | 603 | 566 | 532 | | |
| Time | 10.1 m. | 20.3 m. | 30.3 m. | 40.5 m. | 50.6 m. | 30 | |
| Fitness | 1082 | 852 | 692 | 637 | 625 | | |
| Paths | 781 | 604 | 473 | 426 | 413 | | |
| Time | 16.9 m. | 33.4 m. | 50.7 m. | 1.12 h. | 1.4 h. | 50 | |
| Fitness | 1078 | 860 | 687 | 627 | 616 | | |
| Paths | 780 | 602 | 457 | 411 | 401 | | |

Figure 4c shows the influence of the elite percentage on the final vectorized image. The algorithm working time does not depend on the percentage of the elite, but the vectorized image quality and the paths number depends on it. Thus, the elite percentage equal to 10% is the best.

## 5 Conclusion

We have presented an evolutionary approach designed for the raster images vectorization using a variable number of curves. Unlike existing algorithms our solution is able to work with gradient colors. Additionally, it achieves superior quality results within an acceptable timeframe compared to other existing vectorization algorithms such as LIVE, DiffVG, Pixels2Svg, and SvgTracer. To evaluate the quality of vectorization, we employ a pixel-by-pixel Mean Squared Error (MSE) loss calculation. This allows us to quantitatively measure the similarity between the vectorized image and the original bitmap image.

In order to further enhance the algorithm's performance, we plan to incorporate new types of mutations and crossovers. These additions will address the current limitations, such as long operation time, and contribute to improving the algorithm accuracy. Another improvement involves increasing the algorithm's speed by integrating alternative selection functions. By exploring different selection strategies, we aim to optimize the algorithm's efficiency and reduce processing time.

Furthermore, we plan to directly operate with the contours of vectorized images. This approach will enable us to calculate the quality of vectorized images without the need for excessive rasterization. This advancement will simplify the evaluation process and increase the overall efficiency of the algorithm.

**Acknowledgments.** The research was supported by the ITMO University, project 623097 "Development of libraries containing perspective machine learning methods".

## References

1. Back, T.: Evolutionary Algorithms in Theory and Practice: Evolution Strategies, Evolutionary Programming, Genetic Algorithms. Oxford university press (1996)
2. Bellamy-Royds, A., Cagle, K.: SVG Colors, Patterns & Gradients: Painting Vector Graphics. "O'Reilly Media, Inc." (2015)
3. Bergen, S., Ross, B.J.: Automatic and interactive evolution of vector graphics images with genetic algorithms. Vis. Comput. **28**(1), 35–45 (2012)
4. Bhandari, D., Murthy, C., Pal, S.K.: Genetic algorithm with elitist model and its convergence. Int. J. Pattern Recognit Artif. Intell. **10**(06), 731–747 (1996)
5. Borovikov, V.: Statistica. the art of analyzing data on a computer: For professionals. st. petersburg Piter (2003)
6. Chang, R.I., Su, C.Y.: Color gradient vectorization for SVG compression of comic image. J. Vis. Commun. Image Represent. **33**, 235–246 (2015)
7. Dori, D., Liu, W.: Sparse pixel vectorization: an algorithm and its performance evaluation. IEEE Trans. Pattern Anal. Mach. Intell. **21**(3), 202–215 (1999)

8. Dziuba, M., Jarsky, I., Efimova, V., Filchenkov, A.: Image vectorization: a review. arXiv preprint arXiv:2306.06441 (2023)
9. Frosini, P., Giorgi, D., Melzi, S., Rodolà, E.: Efficient image vectorisation using mesh colours
10. Gotshall, S., Rylander, B.: Optimal population size and the genetic algorithm. Population **100**(400), 900 (2002)
11. Jain, A., Xie, A., Abbeel, P.: Vectorfusion: Text-to-SVG by abstracting pixel-based diffusion models. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 1911–1920 (2023)
12. Kazimipour, B., Li, X., Qin, A.K.: A review of population initialization techniques for evolutionary algorithms. In: 2014 IEEE Congress on Evolutionary Computation (CEC), pp. 2585–2592. IEEE (2014)
13. Laine, S., Karras, T.: High-performance software rasterization on GPUs. In: Proceedings of the ACM SIGGRAPH Symposium on High Performance Graphics, pp. 79–88 (2011)
14. Li, T.M., Lukáč, M., Gharbi, M., Ragan-Kelley, J.: Differentiable vector graphics rasterization for editing and learning. ACM Trans. Graph. (TOG) **39**(6), 1–15 (2020)
15. Ma, X., et al.: Towards layer-wise image vectorization. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 16314–16323 (2022)
16. Majig, M., Fukushima, M.: Adaptive fitness function for evolutionary algorithm and its applications. In: International Conference on Informatics Education and Research for Knowledge-Circulating Society (icks 2008), pp. 119–124. IEEE (2008)
17. Meißner, M., Bartz, D., Günther, R., Straßer, W.: Visibility driven rasterization. In: Computer Graphics Forum, vol. 20, pp. 283–293. Wiley Online Library (2001)
18. Meyes, R., Lu, M., de Puiseau, C.W., Meisen, T.: Ablation studies in artificial neural networks. arXiv preprint arXiv:1901.08644 (2019)
19. Packard, K., Worth, C., Esfahbod, B.: Cairo
20. Quint, A.: Scalable vector graphics. IEEE Multimedia **10**(3), 99–102 (2003)
21. Reddy, P., Gharbi, M., Lukac, M., Mitra, N.J.: Im2Vec: synthesizing vector graphics without vector supervision. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 7342–7351 (2021)
22. Spears, W.M., et al.: Adapting crossover in evolutionary algorithms. In: Evolutionary Programming, pp. 367–384 (1995)
23. Tian, X., Günther, T.: A survey of smooth vector graphics: recent advances in representation, creation, rasterization and image vectorization. IEEE Trans. Visual. Comput. Graphics (2022)
24. Wang, Y., Lian, Z.: DeepVecFont: synthesizing high-quality vector fonts via dual-modality learning. ACM Trans. Graphics (TOG) **40**(6), 1–15 (2021)
25. Wenyin, L., Dori, D.: From raster to vectors: extracting visual information from line drawings. Pattern Anal. Appl. **2**, 10–21 (1999)
26. Young, S.R., Rose, D.C., Karnowski, T.P., Lim, S.H., Patton, R.M.: Optimizing deep learning hyper-parameters through an evolutionary algorithm. In: Proceedings of the Workshop on Machine Learning in High-performance Computing Environments, pp. 1–5 (2015)
27. Zhou, H., Zheng, J., Seah, H.S.: Converting bitmap images into scalable vector graphics. In: Proceedings of the Korean Society of Broadcast Engineers Conference, pp. 435–440. The Korean Institute of Broadcast and Media Engineers (2009)
28. Zhou, Y., Li, X., Gao, L.: A differential evolution algorithm with intersect mutation operator. Appl. Soft Comput. **13**(1), 390–401 (2013)

# Evolution-Based Feature Selection for Predicting Dissolved Oxygen Concentrations in Lakes

Runlong Yu[1], Robert Ladwig[2], Xiang Xu[3], Peijun Zhu[4], Paul C. Hanson[5], Yiqun Xie[6], and Xiaowei Jia[1(✉)]

[1] University of Pittsburgh, Pittsburgh, USA
{ruy59,xiaowei}@pitt.edu
[2] Aarhus University, Aarhus, Denmark
rladwig@ecos.au.dk
[3] University of Science and Technology of China, Hefei, China
demon@mail.ustc.edu.cn
[4] Georgia Institute of Technology, Atlanta, USA
zpj@gatech.edu
[5] University of Wisconsin-Madison, Madison, USA
pchanson@wisc.edu
[6] University of Maryland, College Park, USA
xie@umd.edu

**Abstract.** Accurate prediction of dissolved oxygen (DO) concentrations in lakes requires a comprehensive study of phenological patterns across ecosystems, highlighting the need for precise selection of interactions amongst external factors and internal physical-chemical-biological variables. This paper presents the *Multi-population Cognitive Evolutionary Search (MCES)*, a novel evolutionary algorithm for complex feature interaction selection problems. MCES allows models within every population to evolve adaptively, selecting relevant feature interactions for different lake types and tasks. Evaluated on diverse lakes in the Midwestern USA, MCES not only consistently produces accurate predictions with few observed labels but also, through gene maps of models, reveals sophisticated phenological patterns of different lake types, embodying the innovative concept of "AI from nature, for nature".

**Keywords:** Ecosystem modeling · Adaptive learning · Feature selection

## 1 Introduction

The concentration of dissolved oxygen (DO) in lakes is a key indicator of water quality and the health of freshwater ecosystems. Effective DO monitoring is critical for sustaining aquatic biodiversity and ensuring water security for human consumption [37]. DO concentrations are influenced not only by the exchange


M. Affenzeller et al. (Eds.): PPSN 2024, LNCS 15151, pp. 398–415, 2024.
https://doi.org/10.1007/978-3-031-70085-9_25

of oxygen between air and water, but also by the metabolic processes of primary production and respiration [31]. As articulated by Edward A. Birge one century ago [1]: The fluctuations in a lake's oxygen illustrate its "life cycle" more clearly than many other ecological indicators. This is particularly evident in nutrient-rich eutrophic lakes, where algal blooms can significantly deplete oxygen, creating detrimental "dead zones" for aquatic life.

Accurate prediction of DO concentrations requires a comprehensive study of different phenological patterns across various ecosystems. In particular, DO concentration is closely intertwined with ecosystem phenology, influenced by morphometric and geographic information, mass fluxes, weather conditions, trophic state, and watershed land use. In deeper lakes, for instance, light scarcity and decreased mixing with the oxygen-rich surface can lower oxygen [27,30]. Temperature fluctuations impact oxygen solubility and biochemical activities [33]. Land use changes reshape DO patterns and metabolism phenology [11,38].

Given the importance of predicting DO concentration, scientists across limnology, hydrology, meteorology, and environmental engineering have devised process-based models to simulate the dynamics of freshwater ecosystems. These models, aimed at evaluating the effects of external and internal factors, often combine hydrodynamic and water quality models [10]. Examples include DYRESM [7], GLM [8], MyLake [28]. These models utilize first-order principles (e.g., mass and energy conservation), but also involve many parameterizations or approximations due to incomplete physical knowledge or excessive complexity in modeling certain complex processes, resulting in inherent model bias.

Advanced data-driven methods like deep learning [17], offer an alternative to process-based models for complex scientific problems (e.g., prediction of DO concentration). Their success is contingent upon effective feature selection [2,4,18], however, most extant methods face several major challenges in this problem. Firstly, predicting DO involves sophisticated phenological patterns across various metabolic processes. Directly considering a large number of feature interactions can easily introduce noise, while manual selection risks overlooking critical details. Secondly, most feature selection approaches rely on global models built under expert guidance and lack the flexibility to adapt to various tasks and datasets. As a result, they often fail to select relevant feature interactions for different lake types and prediction tasks. Finally, the sparse nature of DO data further complicates model training, as frequent and comprehensive data collection is hindered by substantial human labor and material costs.

To address these challenges, cognitive evolutionary search (CELS) has been developed [43], utilizing an evolutionary algorithm to adaptively evolve models for selecting feature interactions under task-specific guidance. Unlike conventional methods constrained by model fitness evaluation [35,41,45,46], CELS introduces a novel model fitness diagnosis technique. Despite its advancements, CELS still contends with issues of "reproductive isolation", which may result in the convergence of similar models within a population, thereby limiting the development of specialized models for distinct tasks [19,34].

Leveraging insights from CELS, this paper proposes an advanced evolutionary algorithm, namely *Multi-population Cognitive Evolutionary Search (MCES)*, where cognitive ability refers to the malleability of organisms to orientate to the environment [43]. MCES utilizes multi-population models to effectively serve diverse lake types and predictive tasks, mirroring the adaptive strategies of species in diverse habitats. In MCES, feature interactions are envisaged as genomes and models as organisms within ecosystems, conceptualizing tasks as the natural environments these organisms inhabit. Internally, MCES assesses the capacities of genes, with mutations occurring probabilistically when existing traits are detrimental to survival. The models within MCES undergo crossover and mutation within their respective populations and, though less commonly, engage in inter-population crossover. This genetic diversity and flexibility allow models to dynamically adapt and select appropriate phenological feature interactions, catering to specific environmental conditions of different lake types and tasks.

## 2 Related Work

Existing research suggests that evolution-based feature selection is often limited to filters and wrappers due to model fitness evaluation constraints [35,41]. Specifically, wrapper methods use the performance of the learning algorithm as its evaluation criterion, while filter methods use the intrinsic characteristics of the data. On the other hand, embedded approaches simultaneously select features and learn a classifier, therefore conventional algorithms cannot evaluate the fitness of the model [35,41]. Only genetic programming (GP) and learning classifier systems (LCSs) are able to perform embedded feature selection, but they are not practical [6,20,26,35,41]. For additional research on evolution-based feature selection, please refer to [43].

Research on the population structure of EAs has demonstrated the benefits of segmenting the initial population into multiple sub-populations. These sub-populations exchange information, and regrouping operators are triggered at regular intervals to maintain the population's diversity and balance exploitation with exploration capabilities [39,47]. Various models like the shuffle or update parallel differential evolution (SOUPDE) [36] and the multi-population-based cooperative coevolutionary algorithm (MPCCA) [29] utilize unique mutation strategies and population dynamics to optimize performance across diverse problem-solving scenarios.

## 3 Problem Definition and Preliminaries

**Problem Definition.** Our goal is to predict the DO concentration at a daily scale. We simplify our analysis by dividing the water column into two distinct layers with separate oxygen and metabolic kinetics: the epilimnion (upper surface layer) and the hypolimnion (lower bottom layer). We treat the DO prediction for the epilimnion and hypolimnion as two tasks.

For each lake, we have access to its phenological features $\boldsymbol{x}_t$ on each date $t$. These features, spanning $m$ diverse fields $\boldsymbol{x}_t = \{x_t^1, \cdots, x_t^m\}$, encompass morphometric and geographic details such as lake area, depth, and shape; flux-related data like ecosystem and sedimentation fluxes; weather factors comprising wind speed and temperature; a range of trophic states from dystrophic to eutrophic; and diverse land use proportions extending from forests to wetlands. Besides input features, we also have observed DO concentrations $y_t$ (for both the epilimnion $y_t^{\text{epi}}$ and hypolimnion $y_t^{\text{hyp}}$) on certain days.

We use an embedding layer to convert input phenological features into a series of multi-field feature embeddings $\boldsymbol{f}_t = [\boldsymbol{f}_t^1, \cdots, \boldsymbol{f}_t^m]$, where $\boldsymbol{f}_t^i = \text{embed}(x_t^i)$. Initially, numerical features are bucketed into categories, and each category is then represented as a one-hot vector that is transformed into an embedding vector through a perceptron layer [44]. Our model uses these embeddings to predict DO concentrations $\hat{y}_t$ for both the epilimnion $\hat{y}_t^{\text{epi}}$ and hypolimnion $\hat{y}_t^{\text{hyp}}$.

**Feature Interaction Selection.** The process of feature interaction selection aims to identify the most informative feature interactions that can facilitate the prediction of target DO concentrations $\mathcal{H} : \mathcal{M}(\boldsymbol{f}, \boldsymbol{g}(\boldsymbol{f})) \rightarrow \hat{y}$, where $\boldsymbol{g}$ denotes the set of operations to interact on feature pairs, and $\boldsymbol{g}(\boldsymbol{f})$ denotes the set of interactions. In DO prediction case, the algorithm $\mathcal{H}$ aims to minimize the MSE loss for the outputs of the prediction model $\mathcal{M}$, given as:

$$\mathcal{L}(\mathcal{M}) = \frac{1}{|B|} \sum_{t \in B} \left( y_t - \hat{y}_t \right)^2, \tag{1}$$

where $B$ denotes the set of instance indices in a mini-batch, $\hat{y}_t$ denotes the predictive result given through the learned model.

As the fundamental components in feature interaction, operations are functions where two individual features are converted into an interaction. For the sake of simplicity, we adopt four representative operations as candidate operations to present instantiations of MCES, i.e., $\boldsymbol{g} = \{\oplus, \otimes, \boxplus, \boxtimes\}$. Detailed explanations of these operations and their use in evolution-based feature selection are provided in **Appendix A**.

**Metabolic Process-Based Model.** In this work, we use a metabolic process-based model to simulate DO labels [16]. The process-based model divides the water column into the upper epilimnion and lower hypolimnion during stratified periods. It is focused on analyzing metabolic dynamics in warmer months. Flux features (denoted as $F$) affecting DO concentrations are calibrated with observed data [16]. More details of the process-based model are provided in **Appendix B**.

## 4 Overall Framework

The overall framework is depicted in Fig. 1 and involves two stages of learning: (1) MCES for feature interaction selection using simulated labels, and (2) model refinement using real observed labels.

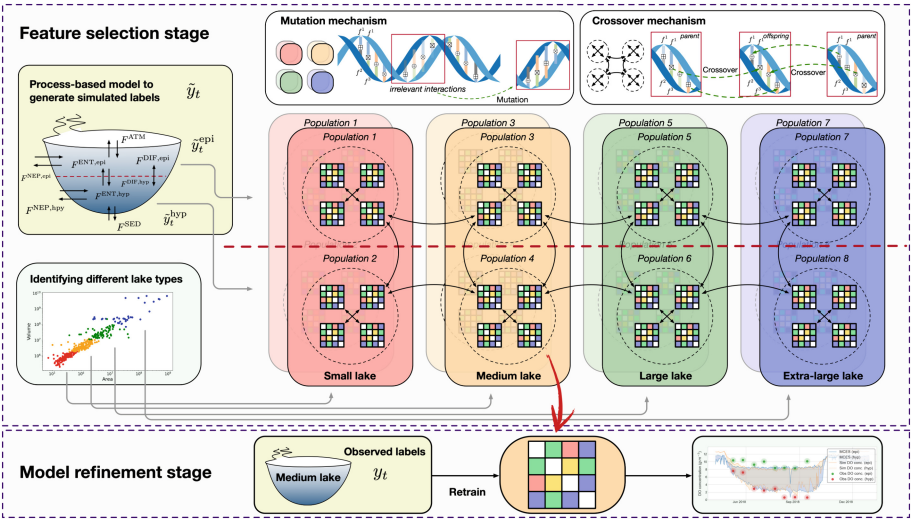



**Fig. 1.** Overall framework.

Specifically, we first utilize metabolic process-based models to generate simulated DO labels given the phenological features $\boldsymbol{x}_t$. Using these simulated labels, we implement our proposed feature interaction selection algorithm MCES. The models in MCES adaptively evolve to select relevant feature interactions within populations for different lake types and tasks. In the subsequent stage, we refine these evolved models using real-world observed DO concentration data. This mirrors natural genetic decoding, where selected feature interactions are further optimized to reflect actual ecological dynamics.

**Identifying Different Lake Types.** As a preprocessing step, lakes are classified based on characteristics critical to oxygen dynamics-primarily surface area and volume. Larger surface areas improve atmospheric oxygen exchange, while greater volumes are associated with higher oxygen consumption in deeper waters. We use a balanced K-means clustering algorithm to uniformly distribute the lakes in dataset $\boldsymbol{L}$ into four categories based on volume and area: small lakes $\boldsymbol{L}_{\mathrm{S}}$, medium lakes $\boldsymbol{L}_{\mathrm{M}}$, large lakes $\boldsymbol{L}_{\mathrm{L}}$, and extra-large lakes $\boldsymbol{L}_{\mathrm{xL}}$.

## 5 Multi-Population Cognitive Evolutionary Search

In this section, we detail the feature selection stage, employing simulated labels $\tilde{y}_t$ to implement MCES. We begin by introducing the establishment of multiple populations, and explain how each individual within these populations is modeled. We then delve into the fundamental mutation and crossover mechanisms. Lastly, we provide an instantiation of the search process.

### 5.1 Multiple Populations

In MCES, each population comprises a set of models designed specifically for a particular combination of lake type and prediction task (e.g., predicting DO in the epilimnion of large lakes). Consequently, we establish eight unique populations: $\boldsymbol{P}_{\mathrm{S}}^{\mathrm{epi}}$, $\boldsymbol{P}_{\mathrm{S}}^{\mathrm{hyp}}$, $\boldsymbol{P}_{\mathrm{M}}^{\mathrm{epi}}$, $\boldsymbol{P}_{\mathrm{M}}^{\mathrm{hyp}}$, $\boldsymbol{P}_{\mathrm{L}}^{\mathrm{epi}}$, $\boldsymbol{P}_{\mathrm{L}}^{\mathrm{hyp}}$, $\boldsymbol{P}_{\mathrm{xL}}^{\mathrm{epi}}$, $\boldsymbol{P}_{\mathrm{xL}}^{\mathrm{hyp}}$. Here the subscript $\{\mathrm{S}, \mathrm{M}, \mathrm{L}, \mathrm{xL}\}$ represents different lake types, and the superscript $\{\mathrm{epi}, \mathrm{hyp}\}$ represents two different prediction tasks (i.e., epilimnion or hypolimnion).

With a population size of $n$ (where $n > 1$), we initialize each population with $n$ models for feature interaction selection: $\boldsymbol{P} = \{\mathcal{M}_1, \cdots, \mathcal{M}_n\}$. We conceptualize each model as a natural organism that evolves to enhance its traits for improved fitness within its environment. These traits, inherited through the organism's genomes, stem from the interplay between features and operations, much like nucleotides and their connections. Following various linkages of nucleotides, we extend the operation set with four types of operations as the search space, i.e., $\boldsymbol{g} = \{\oplus, \otimes, \boxplus, \boxtimes\}$. If $g_k$ is a chosen operation from $\boldsymbol{g}$, an interaction $g_k(\boldsymbol{f}_t^i, \boldsymbol{f}_t^j)$ is defined by applying the operation $g_k$ to a pair of features $(\boldsymbol{f}_t^i, \boldsymbol{f}_t^j)$.

Motivated by the goal of enhancing model fitness through the preservation of beneficial genetic information, we aim to discern and prioritize important features and their interactions via a parameterized method, called internal genetic evaluation. The idea is to introduce a set of relevance parameters to strengthen relevant feature interactions while diminishing or mutating those that contribute less. In this context, we define relevance parameters for features $\boldsymbol{f}_t$ and interactions $\tilde{g}(\boldsymbol{f}_t)$ as $\boldsymbol{\alpha} = \{\alpha_i | 1 \leqslant i \leqslant m\}$ and $\boldsymbol{\beta} = \{\beta_{i,j} | 1 \leqslant i < j \leqslant m\}$, respectively. Here, $\tilde{g}(\boldsymbol{f}_t)$ denotes the interaction of applying any operations from $\boldsymbol{g}$ to a pair of features. The predictive response of our model at time step $t$ is formulated as:

$$\hat{y}_t = \mathcal{M}\big(\boldsymbol{\alpha} \cdot \boldsymbol{f}_t, \boldsymbol{\beta} \cdot \tilde{g}(\boldsymbol{f}_t)\big), \tag{2}$$

where $\mathcal{M}$ can be any individual model in the population. In this work, we use a sequence encoder with Long-Short Term Memory (LSTM) networks [9] to efficiently encode temporal information and the dynamics of feature interactions. The model $\mathcal{M}$ is thus depicted as:

$$\begin{aligned} \boldsymbol{h}_t^{\iota} &= \mathrm{LSTM}\Big([\boldsymbol{\alpha} \cdot \boldsymbol{f}_t, \boldsymbol{\beta} \cdot \tilde{g}(\boldsymbol{f}_t)]; \boldsymbol{h}_{t-1}^{\iota}\Big), \\ \hat{y}_t &= \boldsymbol{W}^{\iota} \cdot \boldsymbol{h}_t^{\iota} + \boldsymbol{b}^{\iota}, \end{aligned} \tag{3}$$

where $\boldsymbol{h}_t^{\iota}$ represents a series of hidden states, and $\boldsymbol{W}^{\iota}$ and $\boldsymbol{b}^{\iota}$ denote the weight and bias parameters, respectively. The loss function for model $\mathcal{M}$, calculated using simulated labels $\tilde{y}_t$, is defined as:

$$\mathcal{L}(\mathcal{M}) = \frac{1}{|B|} \sum_{t \in B} \Big(\tilde{y}_t - \hat{y}_t\Big)^2, \tag{4}$$

where $B$ denotes the set of instance indices within a mini-batch.

We use a regularized dual averaging (RDA) optimizer to learn the relevance parameters $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$, with the aim to distinguish between relevant and irrelevant feature interactions through this process [3,40]. When the absolute value of the cumulative gradient average value in a certain position in $\boldsymbol{\alpha}$ or $\boldsymbol{\beta}$ is less than a threshold, the weight of that position in relevance parameters will be set to 0, resulting in the sparsity of the relevance [22,40]. Meanwhile, the parameters of the embedding layer (for $\boldsymbol{f}_t$) are optimized using the Adam optimizer [14]. Unlike AutoML [23,24], which categorizes $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$ as high-level decisions and treats feature embeddings as lower-level variables for bi-level optimization, our approach simplifies this process. To circumvent the complex and costly bi-level optimization, we update feature embeddings and relevance parameters jointly using single-level optimization with gradients on the training set, as $\nabla_{\boldsymbol{f}}\mathcal{L}(\boldsymbol{f}_{iter-1}, \boldsymbol{\alpha}_{iter-1}, \boldsymbol{\beta}_{iter-1})$ and $\nabla_{\boldsymbol{\alpha},\boldsymbol{\beta}}\mathcal{L}(\boldsymbol{f}_{iter-1}, \boldsymbol{\alpha}_{iter-1}, \boldsymbol{\beta}_{iter-1})$, respectively.

### 5.2 Mutation Mechanism and Crossover Mechanism

With our definitions of population and feature interaction selection models, we further detail the mutation and crossover mechanisms in MCES. The crossover mechanism is bifurcated into intra-population and inter-population crossover.

**Mutation Mechanism.** The mutation serves as a fundamental mechanism of our search process. It primarily aims at mutating the operations associated with irrelevant interactions into alternative operations, and thus generating a new model (the offspring). Specifically, for an interaction $g_k(f_t^i, f_t^j)$, mutation is triggered with a probability $\sigma$ after every $\tau$ steps if the relevance parameter $\beta_{i,j}$ drops below a threshold $\lambda$. In other words, to regenerate a new interaction, the operation $g_k$ of the interaction $g_k(f_t^i, f_t^j)$ mutates into another operation $g_l$, given as:

$$g_k = \begin{cases} g_l \text{ with probability } \sigma, & \text{if } \quad \beta_{i,j} < \lambda, \\ g_k, & \text{otherwise}, \end{cases} \tag{5}$$

where $g_l$ is randomly selected from the operation set as $g_l = \{g \,|\, g \in \boldsymbol{g}, g \neq g_k\}$. The new interaction $g_l(f_i, f_j)$ replaces the irrelevant interaction $g_k(f_i, f_j)$, and its corresponding relevance $\beta_{i,j}$ is reset. Consequently, the parent model $\mathcal{M}$ evolves into its offspring $\mathcal{M}'$, which incorporates these fresh interactions with revised relevance $\boldsymbol{\beta}'$, and maintains features with relevance $\boldsymbol{\alpha}'$ inherited from $\boldsymbol{\alpha}$.

**Intra-Population Crossover Mechanism.** Given a population $\boldsymbol{P} = \{\mathcal{M}_1, \cdots, \mathcal{M}_\nu, \cdots, \mathcal{M}_n\}$, we use $\boldsymbol{\beta}^{\mathcal{M}_\nu}$ to denote the relevance parameters of interactions for each model $\mathcal{M}_\nu$. The obtained $\boldsymbol{\beta}^{\mathcal{M}_\nu}$ can vary across different models in $\boldsymbol{P}$. Therefore, within this population, the models may have a variety of operations for interacting with each feature pair $(f_i, f_j)$, represented as $g_{i,j}^{\boldsymbol{P}} = \{g_{i,j}^{\mathcal{M}_1}, \cdots, g_{i,j}^{\mathcal{M}_\nu}, \cdots, g_{i,j}^{\mathcal{M}_n}\}$. The intra-population crossover mechanism aims to select the most suitable operation (of which interaction has the largest

relevance) within the population to apply on the feature pair for the offspring model $\mathcal{M}'$, given as:

$$g_{i,j}^{\mathcal{M}'} = \arg \max_{g_{i,j}^{\mathcal{M}_\nu} \in g_{i,j}^{\boldsymbol{P}}} \beta_{i,j}^{\mathcal{M}_\nu}. \tag{6}$$

Meanwhile, the relevance parameters of interactions in this offspring model are inherited from their respective parent models (i.e., the selected $\mathcal{M}_\nu$).

**Inter-Population Crossover Mechanism.** For two distinct populations $\boldsymbol{P}_A = \{\mathcal{M}_1^A, \cdots, \mathcal{M}_\nu^A, \cdots, \mathcal{M}_n^A\}$ and $\boldsymbol{P}_B = \{\mathcal{M}_1^B, \cdots, \mathcal{M}_\nu^B, \cdots, \mathcal{M}_n^B\}$, we use $\boldsymbol{\beta}^{\mathcal{M}_\nu^A}$ and $\boldsymbol{\beta}^{\mathcal{M}_\nu^B}$ to denote the relevance of interactions for two populations $\boldsymbol{P}_A$ and $\boldsymbol{P}_B$, respectively. The inter-population crossover mechanism works as follows: For each feature pair $(f_i, f_j)$, we select the most suitable operation from $\boldsymbol{P}_B$ to interact on the feature pair in the offspring model of $\boldsymbol{P}_A$. Conversely, the most suitable operation of the feature pair from $\boldsymbol{P}_A$ is selected for the offspring model of $\boldsymbol{P}_B$, given as:

$$g_{i,j}^{\mathcal{M}'_A} = \arg \max_{g_{i,j}^{\mathcal{M}_\nu^B} \in g_{i,j}^{\boldsymbol{P}_B}} \beta_{i,j}^{\mathcal{M}_\nu^B}, \quad g_{i,j}^{\mathcal{M}'_B} = \arg \max_{g_{i,j}^{\mathcal{M}_\nu^A} \in g_{i,j}^{\boldsymbol{P}_A}} \beta_{i,j}^{\mathcal{M}_\nu^A}. \tag{7}$$

Meanwhile, the relevance parameters of interactions in the offspring models are inherited from their respective parent models.

### 5.3 Instantiation of the Search Process

Utilizing mutation, intra- and inter-population crossover mechanisms, we implement MCES as detailed in Algorithm 1. MCES begins by randomly initializing eight distinct model populations (line 1). It then follows a series of iterative steps (lines 6–28), continuing until convergence. Each iteration involves optimizing offspring models and their relevance parameters within each population.

For every $\tau$ iterations, the algorithm (lines 9–16) selects and replaces the worst model $\mathcal{M}$ in each population $\boldsymbol{P}$ based on the designated loss function (Eq. (4)). The selection process can be expressed as:

$$\mathcal{M} = \arg \max_{\mathcal{M}_\nu \in \boldsymbol{P}} \mathcal{L}(\mathcal{M}_\nu). \tag{8}$$

When the algorithm replaces the worst model $\mathcal{M}$ with the offspring model $\mathcal{M}'$, a new offspring $\mathcal{M}'$ is generated through intra-population crossover and subsequent mutation, enhancing genotypic diversity, thus enabling the search process to effectively avoid local optima and explore global regions.

For every $ep \times \tau$ iterations (lines 18–27), we randomly select a pair of populations, $\boldsymbol{P}_A$ and $\boldsymbol{P}_B$, either based on a shared task (i.e., epilimnion or hypolimnion) across different lake types or on the same lake type but with different tasks, with selection probability balanced by a coin flip. This leads to the generation of new offspring models $\mathcal{M}'_A$, $\mathcal{M}'_B$ through inter-population crossover between $\boldsymbol{P}_A$, $\boldsymbol{P}_B$, promoting the exchange of advantageous genotypic patterns across different lake

**Algorithm 1.** Multi-population Cognitive Evolutionary Search

**Input**: Training dataset of four types of lakes $\boldsymbol{L}_{\mathrm{S}}$, $\boldsymbol{L}_{\mathrm{M}}$, $\boldsymbol{L}_{\mathrm{L}}$, $\boldsymbol{L}_{\mathrm{xL}}$, each lake has features $\boldsymbol{f}_t$, simulated DO labels $\tilde{y}_t^{\mathrm{epi}}$, $\tilde{y}_t^{\mathrm{hyp}}$ over $T$ days; operation set $\boldsymbol{g}$.

1: Initialize eight populations $\boldsymbol{P}_{\mathrm{S}}^{\mathrm{epi}}$, $\boldsymbol{P}_{\mathrm{S}}^{\mathrm{hyp}}$, $\boldsymbol{P}_{\mathrm{M}}^{\mathrm{epi}}$, $\boldsymbol{P}_{\mathrm{M}}^{\mathrm{hyp}}$, $\boldsymbol{P}_{\mathrm{L}}^{\mathrm{epi}}$, $\boldsymbol{P}_{\mathrm{L}}^{\mathrm{hyp}}$, $\boldsymbol{P}_{\mathrm{xL}}^{\mathrm{epi}}$, $\boldsymbol{P}_{\mathrm{xL}}^{\mathrm{hyp}}$, of which any $\mathcal{M}$ has initialized $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$.
2: **for** each $\boldsymbol{P}$ **do**
3: Generate $\mathcal{M}'$ via intra-population crossover in $\boldsymbol{P}$. ▷ Eq. (6)
4: Mutate $\mathcal{M}'$. ▷ Eq. (5)
5: **end for**
6: **while** not converged **do**
7: **for** each $\boldsymbol{P}$ **do**
8: Optimize $\mathcal{M}'$ with $\boldsymbol{\alpha}'$ $\boldsymbol{\beta}'$.
9: **if** $\mathrm{mod}(t, \tau) = 0$ **then**
10: Select the worst $\mathcal{M}$. ▷ Eq. (8)
11: Replace $\mathcal{M}$ in $\boldsymbol{P}$ with $\mathcal{M}'$.
12: **if** $\mathrm{mod}(t, ep \times \tau) \neq 0$ **then**
13: Generate $\mathcal{M}'$ via intra-population crossover.
14: Mutate $\mathcal{M}'$. ▷ Eq. (5)
15: **end if**
16: **end if**
17: **end for**
18: **if** $\mathrm{mod}(t, ep \times \tau) = 0$ **then**
19: Choose $(\boldsymbol{P}_A, \boldsymbol{P}_B)$ either by task or lake type.
20: Generate $(\mathcal{M}'_A, \mathcal{M}'_B)$ via inter-population crossover of $\boldsymbol{P}_A$, $\boldsymbol{P}_B$. ▷ Eq. (7)
21: **for** each $\boldsymbol{P}$ not in $(\boldsymbol{P}_A, \boldsymbol{P}_B)$ **do**
22: Generate $\mathcal{M}'$ via intra-population crossover.
23: **end for**
24: **for** each $\boldsymbol{P}$ **do**
25: Mutate $\mathcal{M}'$. ▷ Eq. (5)
26: **end for**
27: **end if**
28: **end while**
29: **return** the set of best models $\mathcal{M} = \arg\min_{\mathcal{M}_\nu \in \boldsymbol{P}} \mathcal{L}(\mathcal{M}_\nu)$ from each population $\boldsymbol{P}$.

types and tasks. Meanwhile, each remaining population generates its offspring, $\mathcal{M}'$, through intra-population crossover, followed by mutation of all offspring.

Finally, the algorithm concludes by delivering a set of the best models, one from each population (line 29), thereby ensuring a comprehensive exploration and exploitation of the search space across diverse environmental contexts.

## 6 Model Refinement

Inspired by nature's replication and transcription processes, which translate genetic information into protein sequences to equip organisms with diverse functions, we proceed to a model refinement stage. Here our objective is to refine the model to better leverage the features and interactions obtained from MCES. At

this stage, we select the corresponding model by the lake type and task, and then use observed labels for the model refinement. Relevant features and interactions are selected according to their relevance parameters $\boldsymbol{\alpha}$, $\boldsymbol{\beta}$. If $\alpha_i = 0$ or $\beta_{i,j} = 0$, the corresponding features or interactions are fixed to be discarded permanently. Given the scarcity of observed data, we inherit parameters from the preceding LSTM to ensure the model's effective learning, given as:

$$\begin{aligned} \boldsymbol{h}_t^o &= \mathrm{LSTM}\Big([\boldsymbol{\alpha} \cdot \boldsymbol{f}_t, \boldsymbol{\beta} \cdot \tilde{g}(\boldsymbol{f}_t)]; \boldsymbol{h}_{t-1}^o\Big) \\ \hat{y}_t &= \boldsymbol{W}^o \cdot \boldsymbol{h}_t^o + \boldsymbol{b}^o \end{aligned} \tag{9}$$

where $\boldsymbol{h}_t^o$ represents a series of hidden states, with $\boldsymbol{W}^o$ and $\boldsymbol{b}^o$ as the weight and bias parameters. The relevance $\boldsymbol{\alpha}$, $\boldsymbol{\beta}$ are fixed and serve as attention units.

To address the disparity between abundant simulated and scarce observed labels, we've developed a new loss function for LSTM that integrates both types of data through weighted imputation [12,42]. This assigns a greater weight to the loss on observed data and a smaller weight to simulated data, effectively addressing the scarcity of observed labels. The loss function is expressed as:

$$\mathcal{L}(\mathcal{M}) = \frac{1}{|B|} \sum_{t \in B} \mathbb{I}(y_t) \Big(y_t - \hat{y}_t\Big)^2 + \rho \Big(1 - \mathbb{I}(y_t)\Big) \Big(\tilde{y}_t - \hat{y}_t\Big)^2, \tag{10}$$

where $\hat{y}_t$ denotes the predicted DO concentration, $y_t$ is the observed DO concentration, $\tilde{y}_t$ is the simulated DO concentration, $\mathbb{I}(x)$ is an indicator function that equals 1 if $x$ is observed (true) and 0 otherwise (false), and $\rho$ is the tradeoff parameter balancing observed and simulated labels.

## 7 Experimental Evaluation

### 7.1 Dataset

We evaluate the proposed MCES for predicting DO concentration using a dataset that documents over 41 years of ecological observations from 375 lakes in the Midwestern USA, starting in 1979. This dataset includes around 5.58 million daily records, each with 39 fields of phenological features such as morphometric, flux data, weather conditions, trophic states, and land use details. Observed DO data were sourced from the Water Quality Portal (WQP). Lake residence time was taken from the HydroLAKES. Trophic state probabilities (eutrophic, oligotrophic, dystrophic) were from the dataset [25]. Land use proportions of each lake's watershed were taken from the National Land Cover Database (NLCD). An account of these features is available in **Appendix C**. For training MCES, we use data from all 375 lakes. We then selectively conduct testing on lakes that have the most comprehensive DO observations for each type. For large and extra-large lakes, we use data up to 2017 for training, 2018 for validation, and 2019 for testing. For small and medium lakes, where DO observations in 2019 are relatively sparse, we use data up to 2016 for training, 2017 for validation, and 2018 for testing.

### 7.2 Baselines

We compare to a set of baselines in our experiment: *Sim DO Conc.*: This baseline is the metabolic process-based model used in our first stage, leveraging few observed labels to calibrate simulations that can significantly augment the data for other baselines. *LSTM*: As adopted in our model refinement stage, LSTM incorporates simulated labels for weighted imputation and backward gradient adjustments, a necessity for convergence given the scarcity of observed labels. *EA-LSTM & KGSSL* [5,15]: These time series prediction models, which assimilate hydrological behavior and physical processes, respectively, are regarded as cutting-edge within hydrological and ecological domains. LSTM, EA-LSTM, and KGSSL use individual features for input without feature interaction modeling. *AutoInt, AutoGroup, & AutoFeature* [13,21,32]: These methods are at the forefront of feature interaction modeling. They have demonstrated their utility and versatility through extensive commercial deployment, showcasing their capacity to model complex feature combinations.

**Table 1.** Comparative performance of DO concentration ($g/m^3$) prediction in terms of root mean square error (RMSE) across different lake types and tasks. The mean and standard deviation (displayed in grey) of RMSE are calculated from five runs.

| Algo. Name | Small lakes | | Medium lakes | | Large lakes | | Extra-large lakes | |
|---|---|---|---|---|---|---|---|---|
| | Epi. | Hyp. | Epi. | Hyp. | Epi. | Hyp. | Epi. | Hyp. |
| Sim DO conc. | 1.943 ± 0.000 | 2.212 ± 0.000 | 1.940 ± 0.000 | 2.217 ± 0.000 | 2.620 ± 0.000 | 2.937 ± 0.000 | 1.536 ± 0.000 | 2.772 ± 0.000 |
| LSTM | 1.802 ± 0.079 | 1.973 ± 0.064 | 1.744 ± 0.092 | 2.001 ± 0.081 | 2.298 ± 0.088 | 2.630 ± 0.043 | 1.479 ± 0.068 | 2.594 ± 0.056 |
| EA-LSTM | 1.716 ± 0.047 | 1.783 ± 0.098 | 1.676 ± 0.084 | 1.546 ± 0.054 | 2.111 ± 0.045 | 2.629 ± 0.043 | 1.478 ± 0.039 | 2.278 ± 0.062 |
| KGSSL | 1.793 ± 0.044 | 1.467 ± 0.057 | 1.557 ± 0.062 | 1.632 ± 0.094 | 2.064 ± 0.103 | 2.730 ± 0.060 | 1.294 ± 0.047 | 2.425 ± 0.075 |
| AutoInt | 1.510 ± 0.080 | 1.406 ± 0.097 | 1.516 ± 0.088 | 1.626 ± 0.094 | 1.716 ± 0.072 | 1.924 ± 0.078 | 1.112 ± 0.081 | 1.847 ± 0.085 |
| AutoGroup | 1.473 ± 0.078 | 1.509 ± 0.080 | 1.364 ± 0.059 | 1.875 ± 0.072 | 1.384 ± 0.075 | 1.600 ± 0.068 | 0.937 ± 0.085 | 1.953 ± 0.076 |
| AutoFeature | 1.382 ± 0.063 | 1.768 ± 0.089 | 1.422 ± 0.070 | 1.467 ± 0.081 | 1.405 ± 0.084 | 1.465 ± 0.082 | 1.178 ± 0.078 | 1.976 ± 0.093 |
| MCES (-refine) | 1.851 ± 0.204 | 2.044 ± 0.210 | 1.812 ± 0.227 | 2.024 ± 0.223 | 2.374 ± 0.215 | 2.775 ± 0.218 | 1.530 ± 0.208 | 2.648 ± 0.243 |
| MCES (-multi) | 1.315 ± 0.182 | 1.624 ± 0.207 | 1.390 ± 0.203 | 1.473 ± 0.204 | 1.390 ± 0.208 | 1.572 ± 0.194 | 1.151 ± 0.197 | 1.848 ± 0.235 |
| MCES (-inter) | 1.107 ± 0.179 | 1.375 ± 0.193 | 1.161 ± 0.192 | 1.354 ± 0.190 | 1.004 ± 0.198 | 1.307 ± 0.189 | 1.040 ± 0.194 | 1.536 ± 0.223 |
| MCES | 1.076 ± 0.146 | 1.316 ± 0.161 | 1.060 ± 0.137 | 1.288 ± 0.159 | 0.988 ± 0.169 | 1.243 ± 0.156 | 0.918 ± 0.171 | 1.415 ± 0.215 |

### 7.3 Implementation Details

To implement MCES, we perform a grid search setting feature embedding size $|\boldsymbol{f}_t^i| = 15$. We use RDA optimizer [3,40] to discriminate the relevant and irrelevant feature interactions, with the learning rate $\gamma = 10^{-3}$, adjustable hyperparameters $c = 0.5, \mu = 0.8$. We set the population size as $n = 4$. We set the mutation mechanism as the mutation threshold $\lambda = 0.2$, the mutation probability $\sigma = 0.5$, and the mutation step size $\tau = 10$. We set the inter-population crossover step size $ep = 10$, and the tradeoff parameter $\rho = 0.1$. To assess the effectiveness of utilizing multiple populations and refining models, we conduct ablation studies with the following variants: *MCES (-refine)*: This variant drops the model refinement stage, using only the simulated labels from the first stage for training. *MCES (-multi)*: Instead of using multiple populations, this variant trains a single population on all data, and tests it separately across different lake types. This setup also excludes the inter-population crossover mechanism. *MCES (-inter)*: This variant does not include the inter-population crossover mechanism.

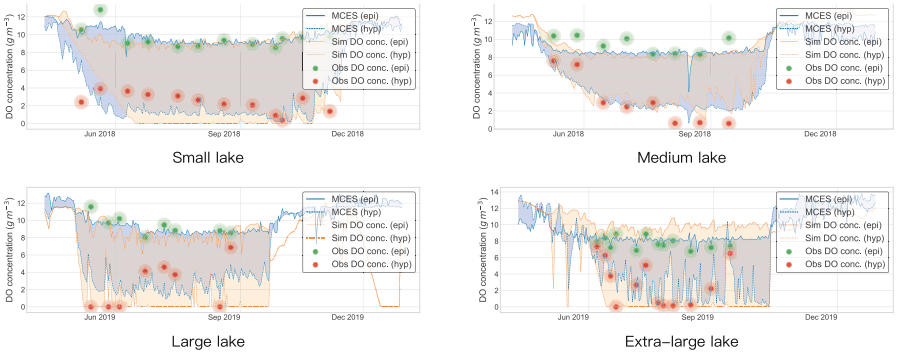



**Fig. 2.** Time-series analysis of DO concentrations: a comparison of predicted (MCES), simulated, and observed values.

### 7.4 Experimental Results

**Performance Comparison.** Table 1 presents a comparative analysis of MCES against various baselines, utilizing root mean square error (RMSE) across diverse lake types and tasks, with both mean and standard deviation calculated over five runs. From the results, we have the following key observations: First, machine learning models universally outperform simulations alone, underscoring the value of integrating observed labels with simulated labels for enhanced prediction accuracy. Second, EA-LSTM and KGSSL surpass LSTM in performance, evidencing the advantage of incorporating hydrological behaviors and physical processes into models, particularly when faced with a scarcity of labels. Third, AutoInt,

AutoGroup, and AutoFeature demonstrate the predictive power of feature interactions, offering significant improvements over models that rely solely on individual feature inputs. Fourth, MCES outperforms all baseline models, attributing its success to the adaptive modeling of interactions through evolutionary operation selection. Unlike other methods that indiscriminately handle all feature interactions, MCES discerns relevant feature interactions for specific lake types and tasks, optimizing their impact while minimizing less pertinent ones. Lastly, ablation study results reveal that MCES performs better than MCES (-refine), highlighting the value of combining real with simulated labels for accuracy. MCES also exceeds MCES (-multi), emphasizing the necessity of distinct populations for different ecological settings. Moreover, MCES outshines MCES (-inter), demonstrating the benefits of inter-population interactions. These interactions promote the sharing of effective feature interactions among populations, thereby improving the algorithm's accuracy and stability.

Figure 2 offers a time-series comparison of predicted (i.e., MCES), simulated, and observed DO concentrations, with a specific emphasis on the summer season of the testing period. The analysis reveals that MCES predictions not only align closely with observed values but also demonstrate sensitivity to subtle features and interactions, enhancing their accuracy. While simulated DO concentrations also generally exhibit a clear trend, there are instances of slight deviation from observed data. Encouragingly, both predicted and simulated values largely mirror observed trends, highlighting the efficacy and significance of our proposed MCES in advancing research in this domain.

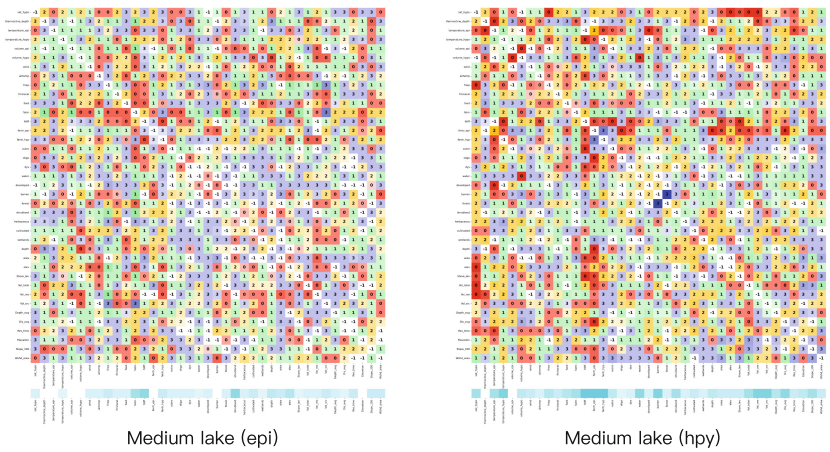



**Fig. 3.** Visualization of gene maps for medium lakes.

**Visualization of Gene Maps.** The model populations accommodate different lake types and tasks, leading to a rich diversity in model traits. These

traits influence their survival and fitness rates, mirroring the selection process for operations and feature interactions. To demonstrate the model's evolutionary process and how feature interactions adapt across different lake types and tasks, we visualize the model's **gene maps**. Adopting an encoding where $\oplus = 0, \otimes = 1, \boxplus = 2, \boxtimes = 3$, we can diagnose the model's fitness as a symmetric matrix. Distinct colors are allocated to each operation, creating a vibrant gene map where each gene symbolizes an interaction; like red "0", green "1", yellow "2", and blue "3". For example, a green "1" within the "`depth` × `area`" block signifies that the element-wise product $\otimes$ is identified as the optimal operation for "`depth`" to interact with "`area`". The intensity of the colors on the gene map is directly correlated with the relevance of the interactions, with darker hues denoting higher relevance and lighter ones suggesting lesser importance. Individual features are also visually encoded as single-hued bars. Interactions deemed irrelevant, with their relevance parameters reduced to 0, are excluded, leaving their corresponding genes depicted in white "−1".

In **Appendix D**, we present gene maps for all lake types and tasks, using end-of-training data to highlight relevant feature interactions for DO concentration prediction. For instance, in Fig. 3, we showcase gene maps for medium lakes.

These maps reveal that, in larger lakes, the DO dynamics are predominantly influenced by sediment oxygen demand and atmospheric exchange, reflecting their extensive water volumes. Conversely, smaller lakes exhibit DO concentrations that are notably impacted by local land use and meteorological factors due to their shallower depths and greater vulnerability to changes in their external watershed environments. Across the board, temperature-related interactions are significant, affecting DO solubility and the lake's biological processes. Additionally, wind speed and atmospheric exchange flux stand out as key drivers of surface gas exchange influencing epilimnion, while the trophic state markers provide indicators of possible oxygen production in the epilimnion and eventual hypolimnetic depletion due to the formation of algal blooms. These findings suggest that diverse ecological factors interplay differently across lake environments, necessitating adaptable prediction models that can cater to these variances.

Additionally, we have included animations in the **supplementary** materials to illustrate the evolution path of MCES. These animations display yearly changes in relevant feature interactions for DO concentrations from 1979. The enduring patterns of feature interactions hint at consistent ecological processes, while deviations in their relevance suggest adaptation to environmental shifts and human activities. Changes in the importance of certain interactions may stem from better land management or climate variations affecting lake stratification. Meanwhile, the emergence of new significant interactions could be a reaction to changes in lake usage or watershed practices. These temporal dynamics underscore the adaptability of MCES, which recalibrates the significance of feature interactions to align with the changing lake environments over time.

In **Appendix E**, we also demonstrate the impact of feature interactions identified by MCES. By adjusting the RDA optimizer's parameters, we consistently

choose a sparser set of feature interactions, accepting a trade-off in accuracy. Simultaneously, a random strategy is applied for comparative purposes, where operations for feature interactions are allocated at random, and some interactions are arbitrarily removed as sparsity intensifies. The gene map showcased at a feature interaction sparsity level around 0.5 offers insight into the model's structure under reduced complexity. This experiment highlights MCES's superior performance even as many feature interactions are discarded, emphasizing its precision in identifying relevant interactions under task guidance. Conversely, the random approach shows a quicker performance drop due to the loss of important interactions. When feature interactions become exceedingly sparse, both methodologies suffer in performance, indicating that a limited set of feature interactions fails to significantly contribute to the model's predictive capabilities. In such cases, individual features primarily drive performance.

## 8 Conclusion

This paper presents a novel evolutionary algorithm, namely *Multi-population Cognitive Evolutionary Search (MCES)*, blending adaptive learning with natural processes for predicting DO concentrations. MCES employs multi-population models customized for varied lake types and tasks, reflecting the diverse survival strategies of species across various habitats. Evaluated on a variety of lakes in the Midwestern USA, MCES not only demonstrates accurate DO concentration predictions with limited observed data but also reveals sophisticated phenological patterns, highlighting its utility for environmental science and freshwater management. MCES introduces an innovative concept of "AI from nature, for nature". We encourage further exploration and development of new algorithms inspired by MCES, aimed at benefiting the environment.

**Acknowledgments.** This research was partially supported by the University of Pittsburgh Center for Research Computing through the resources provided.

**Disclosure of Interests.** The authors have no competing interests to declare that are relevant to the content of this article.

## References

1. Birge, E.A.: Gases dissolved in the waters of Wisconsin lakes. Trans. Am. Fish. Soc. **35**(1), 143–163 (1906)
2. Brookhouse, J., Freitas, A.: Fair feature selection with a lexicographic multi-objective genetic algorithm. In: Rudolph, G., Kononova, A., Aguirre, H., Kerschke, P., Ochoa, G., Tušar, T. (eds.) Parallel Problem Solving from Nature – PPSN XVII: 17th International Conference, PPSN 2022, Dortmund, Germany, September 10–14, 2022, Proceedings, Part II, pp. 151–163. Springer International Publishing, Cham (2022). https://doi.org/10.1007/978-3-031-14721-0_11
3. Chao, S.K., Cheng, G.: A generalization of regularized dual averaging and its dynamics. arXiv preprint arXiv:1909.10072 (2019)

4. Correia, J., Machado, P., Romero, J., Carballal, A.: Feature Selection and Novelty in Computational Aesthetics. In: Machado, P., McDermott, J., Carballal, A. (eds.) Evolutionary and Biologically Inspired Music, Sound, Art and Design, pp. 133–144. Springer Berlin Heidelberg, Berlin, Heidelberg (2013). https://doi.org/10.1007/978-3-642-36955-1_12
5. Ghosh, R., et al.: Robust inverse framework using knowledge-guided self-supervised learning: an application to hydrology. In: Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, pp. 465–474 (2022)
6. Guyon, I., Elisseeff, A.: An introduction to variable and feature selection. J. Mach. Learn. Res. **3**, 1157–1182 (2003)
7. Hamilton, D.P., Schladow, S.G.: Prediction of water quality in lakes and reservoirs. part i-model description. Ecol. model. **96**(1-3), 91–110 (1997)
8. Hipsey, M.R., et al.: A general lake model (GLM 3.0) for linking with high-frequency sensor data from the global lake ecological observatory network (GLEON). Geoscientific Model Dev. **12**(1), 473–523 (2019)
9. Hochreiter, S., Schmidhuber, J.: Long short-term memory. Neural Comput. **9**(8), 1735–1780 (1997)
10. Janssen, A.B., Arhonditsis, G.B., Beusen, A., Bolding, K., Bruce, L., Bruggeman, J., Couture, R.M., Downing, A.S., Alex Elliott, J., Frassl, M.A., et al.: Exploring, exploiting and evolving diversity of aquatic ecosystem models: a community perspective. Aquat. Ecol. **49**, 513–548 (2015)
11. Jenny, J.P., et al.: Urban point sources of nutrients were the leading cause for the historical spread of hypoxia across European lakes. Proc. Nat. Acad. Sci. **113**(45), 12655–12660 (2016)
12. Jia, X., et al.: Physics guided RNNs for modeling dynamical systems: a case study in simulating lake temperature profiles. In: Proceedings of the 2019 SIAM International Conference on Data Mining, pp. 558–566. SIAM (2019)
13. Khawar, F., Hang, X., Tang, R., Liu, B., Li, Z., He, X.: AutoFeature: searching for feature interactions and their architectures for click-through rate prediction. In: ACM International Conference on Information and Knowledge Management (CIKM), pp. 625–634 (2020)
14. Kingma, D.P., Ba, J.: Adam: a method for stochastic optimization. arXiv preprint arXiv:1412.6980 (2014)
15. Kratzert, F., Klotz, D., Shalev, G., Klambauer, G., Hochreiter, S., Nearing, G.: Towards learning universal, regional, and local hydrological behaviors via machine learning applied to large-sample datasets. Hydrol. Earth Syst. Sci. **23**(12), 5089–5110 (2019)
16. Ladwig, R., et al.: Long-term change in metabolism phenology in north temperate lakes. Limnol. Oceanogr. **67**(7), 1502–1521 (2022)
17. LeCun, Y., Bengio, Y., Hinton, G.: Deep learning. Nature **521**(7553), 436–444 (2015)
18. Li, J., et al.: Feature selection: a data perspective. ACM comput. surv. (CSUR) **50**(6), 1–45 (2017)
19. Li, X., Epitropakis, M.G., Deb, K., Engelbrecht, A.: Seeking multiple solutions: an updated survey on niching methods and their applications. IEEE Trans. Evol. Comput. **21**(4), 518–538 (2016)
20. Lin, J.Y., Ke, H.R., Chien, B.C., Yang, W.P.: Classifier design with feature selection and feature extraction using layered genetic programming. Expert Syst. Appl. **34**(2), 1384–1393 (2008)

21. Liu, B., Xue, N., Guo, H., Tang, R., Zafeiriou, S., He, X., Li, Z.: AutoGroup: Automatic feature grouping for modelling explicit high-order feature interactions in CTR prediction. In: Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR), pp. 199–208 (2020)
22. Liu, B., Zhu, C., Li, G., Zhang, W., et al.: AutoFIS: automatic feature interaction selection in factorization models for click-through rate prediction. In: Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining (KDD), pp. 2636–2645 (2020)
23. Liu, H., Simonyan, K., Yang, Y.: Darts: differentiable architecture search. In: International Conference on Learning Representations (2019)
24. Liu, Y., et al.: A survey on evolutionary neural architecture search. IEEE Trans. Neural Netw. Learn. Syst. **34**(2), 550–570 (2021)
25. Meyer, M.F., et al.: National-scale remotely sensed lake trophic state from 1984 through 2020. Sci. Data **11**(1), 77 (2024)
26. Muni, D.P., Pal, N.R., Das, J.: Genetic programming for simultaneous feature selection and classifier design. IEEE Trans. Syst. Man Cybern. Part B (Cybern.) **36**(1), 106–117 (2006)
27. Phillips, J.S.: Time-varying responses of lake metabolism to light and temperature. Limnol. Oceanogr. **65**(3), 652–666 (2020)
28. Saloranta, T.M., Andersen, T.: MyLake-a multi-year lake simulation model code suitable for uncertainty and sensitivity analysis simulations. Ecol. Model. **207**(1), 45–60 (2007)
29. Shang, R., et al.: A multi-population cooperative coevolutionary algorithm for multi-objective capacitated arc routing problem. Inf. Sci. **277**, 609–642 (2014)
30. Solomon, C.T., et al.: Ecosystem respiration: drivers of daily variability and background respiration in lakes around the globe. Limnol. Oceanogr. **58**(3), 849–866 (2013)
31. Sommer, U., et al.: Beyond the plankton ecology group (peg) model: mechanisms driving plankton succession. Annu. Rev. Ecol. Evol. Syst. **43**, 429–448 (2012)
32. Song, W., et al.: Autoint: automatic feature interaction learning via self-attentive neural networks. In: ACM International Conference on Information and Knowledge Management (CIKM), pp. 1161–1170 (2019)
33. Staehr, P.A., et al.: Lake metabolism and the diel oxygen technique: state of the science. Limnol. Oceanogr. Methods **8**(11), 628–644 (2010)
34. Tang, K., Yang, P., Yao, X.: Negatively correlated search. IEEE J. Sel. Areas Commun. **34**(3), 542–550 (2016)
35. Telikani, A., Tahmassebi, A., et al.: Evolutionary machine learning: a survey. ACM Comput. Surv. **54**(8), 1–35 (2021)
36. Weber, M., Neri, F., Tirronen, V.: Shuffle or update parallel differential evolution for large-scale optimization. Soft. Comput. **15**(11), 2089–2107 (2011)
37. Wilson, P.C.: Water quality notes: dissolved oxygen: Sl313/ss525, 1/2010. EDIS **2010**(2) (2010)
38. Woolway, R.I., et al.: Phenological shifts in lake stratification under climate change. Nat. Commun. **12**(1), 2318 (2021)
39. Wu, G., Mallipeddi, R., Suganthan, P.N., Wang, R., Chen, H.: Differential evolution with multi-population based ensemble of mutation strategies. Inf. Sci. **329**, 329–345 (2016)
40. Xiao, L.: Dual averaging method for regularized stochastic learning and online optimization. Adv. Neural Inf. Proc. Syst. **22** (2009)

41. Xue, B., Zhang, M., Browne, W.N., Yao, X.: A survey on evolutionary computation approaches to feature selection. IEEE Trans. Evol. Comput. **20**(4), 606–626 (2015)
42. Ye, Y., et al.: Mane: organizational network embedding with multiplex attentive neural networks. IEEE Trans. Knowl. Data Eng. **35**(4), 4047–4061 (2022)
43. Yu, R., Xu, X., Ye, Y., Liu, Q., Chen, E.: Cognitive evolutionary search to select feature interactions for click-through rate prediction. In: Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, pp. 3151–3161 (2023)
44. Yu, R., Ye, Y., Liu, Q., Wang, Z., Yang, C., Hu, Y., Chen, E.: XCrossNet: feature structure-oriented learning for click-through rate prediction. In: Advances in Knowledge Discovery and Data Mining: 25th Pacific-Asia Conference (PAKDD), pp. 436–447. Springer (2021). https://doi.org/10.1007/978-3-030-75765-6_35
45. Zhang, F., Mei, Y., Zhang, M.: A two-stage genetic programming hyper-heuristic approach with feature selection for dynamic flexible job shop scheduling. In: Proceedings of the Genetic and Evolutionary Computation Conference, pp. 347–355 (2019)
46. Zhao, H., Wu, X., et al.: CoEA: a cooperative-competitive evolutionary algorithm for bidirectional recommendations. IEEE Trans. Evol. Comput. **26**(1), 28–42 (2021)
47. Zhu, L., Ma, Y., Bai, Y.: A self-adaptive multi-population differential evolution algorithm. Nat. Comput. **19**, 211–235 (2020)

# Using Evolutionary Algorithms for the Search of 16-Variable Weight-Wise Perfectly Balanced Boolean Functions with High Non-linearity

Sara Mandujano[1], Adriana Lara[1(✉)], and Juan Carlos Ku Cauich[2]

[1] Instituto Politécnico Nacional, ESFM Edificio 9 Unidad Profesional Adolfo López M., Zacatenco, C.P. 07738 Ciudad de México, Mexico
smandujanov2000@alumno.ipn.mx, alaral@ipn.mx

[2] Computer Science, CINVESTAV-IPN, Av. IPN 2508, San Pedro Zacatenco, Mexico City 07300, Mexico
jcku@cs.cinvestav.mx
http://www.ipn.mx

**Abstract.** New methods to construct adequate Boolean functions for use in cryptography have had to evolve in accordance with the requirements of continually proposed cryptosystems. Among the desired properties in Boolean functions are balancedness, high non-linearity, algebraic immunity, and resilience, all of which contribute to making the cryptosystem more resistant to various attacks. In 2016, the FLIP steam-cipher was proposed, which requires *weight-wise perfectly balanced* Boolean functions. As the field of cryptography evolves, so does the need for new methods of constructing Boolean functions. Our research contributes to this ongoing exploration. Evolutionary algorithms are among the explored approaches. However, much of the work has focused on 8-variable functions. In this investigation, previous work done on 8-variable functions is revisited and applied to the search for 16-variable WPB Boolean functions. The investigation yielded promising results, finding 16-variable WPB Boolean functions with high general non-linearity and weight-wise non-linearities that surpassed previous results. This marks a significant advancement in the exploration of EAs' potential in cryptography.

**Keywords:** Boolean Functions · evolutionary algorithms · cryptography

## 1 Introduction

Among their many applications, Boolean functions play today an important role in cryptography. Within symmetric key cryptography, the two main types of ciphers are block cipher and stream cipher [10,23]. There exist a number of different cryptographic attacks, including linear, differential, and algebraic attacks [18]. In 2016 the stream cipher known as FLIP was presented, which


M. Affenzeller et al. (Eds.): PPSN 2024, LNCS 15151, pp. 416–428, 2024.
https://doi.org/10.1007/978-3-031-70085-9_26

requires a particular type of functions known as Weight-wise Perfectly Balanced (WPB) [4,15,16]. The balance of these WPB functions is global and specific for each weight of the domain elements.

For Boolean functions $f : \mathbb{F}_2^n \to \mathbb{F}_2$, the search space is of size $2^{2^n}$, where $n$ is the number of variables in the function. There exist three main search methods for finding optimum Boolean functions: random search, algebraic methods, and heuristic methods. Additionally, a hybrid search of algebraic and heuristic methods [19] may be used. In this work, heuristic methods are used, specifically evolutionary algorithms (EAs).

The search for ideal Boolean functions for cryptography, in general, has been carried out with single-objective algorithms [2,9,12,14,15] and with multi-objective algorithms [1,18]. Additionally, in previous investigations [18], three representations and evolutionary algorithms are used: genetic algorithms (GAs) with binary encoding, genetic programming (GP) with tree representation, and finally, Cartesian genetic programming (CGP) with graph representation.

Regarding the search and study of the WPB functions, their investigation has thus far only been carried out with single-objective algorithms [7,11,14,15]. However, as far as the authors are aware, so far the use of heuristics has been limited to the construction of 8-variable functions. Thus, this work focuses on the application of EAs for the construction of 16-variable WPB Boolean functions with high non-linearity. Moreover, though the focus is to construct WPB Boolean functions with high non-linearity, throughout the investigation other cryptographic parameters were also evaluated in order to gauge the robustness of the functions generated using this approach.

## 2 Background

Let $\mathbb{F}_2$ be the finite field with two elements. These elements can be represented by $\{0, 1\}$ with addition modulo two, represented by $\oplus$. Similarly, $\mathbb{F}_2^n$ denotes the $n$-dimensional vector space over $\mathbb{F}_2$, $n \in \mathbb{N}$. The following definitions and theorems come from [3,13,15,18].

**Definition 1.** *Every function with domain $\mathbb{F}_2^n$ and co-domain $\mathbb{F}_2$ is called a Boolean function. The set of all Boolean functions is denoted as $\mathcal{B}_n$.*

Thus $\mathcal{B}_n := \{f | f : \mathbb{F}_2^n \to \mathbb{F}_2\}$ and has dimension $2^n$ as a vector space structure; therefore, has cardinality $2^{2^n}$. The set of images of a function $f \in \mathcal{B}_n$ can be represented by a truth table as a vector of length $2^n$ denoted

$$ev(f) := \left(f(v)_{v \in \mathbb{F}_2^n}\right)$$

or through a table as in Table 1. The inputs must always be arranged in the same order, most typically ordered lexicographically.

An example of a Boolean function is the function $f(x) \in \mathcal{B}_2$ defined by

$$f(x_1, x_2) = x_1 \oplus x_2 \oplus 1.$$

The set of images of $f$ in lexicographic order determines the vector

$$[1, 0, 0, 1].$$

**Table 1.** Truth table for the Boolean function $f(x_1, x_2) = x_1 \oplus x_2 \oplus 1$.

| $x_1$ | $x_2$ | $ev(f)$ |
|---|---|---|
| 0 | 0 | 1 |
| 0 | 1 | 0 |
| 1 | 0 | 0 |
| 1 | 1 | 1 |

**Definition 2.** *The support of a vector $x \in \mathbb{F}_2^n$, denoted $supp(x)$, is the set containing the non-zero positions in the vector. The Hamming weight of a vector $x \in \mathbb{F}_2^n$, denoted $w_H(x)$, is the number of non-zero positions in the vector.*

**Definition 3.** *The support of a Boolean function $f \in \mathcal{B}_n$, denoted $supp(f)$, is the set containing the non-zero positions in its truth table. The Hamming weight of a Boolean function $f \in \mathcal{B}_n$, denoted $w_H(f)$, is the number of non-zero positions in its truth table.*

**Definition 4.** *The Hamming distance between two vectors $x, y$, denoted $d_H(x, y)$, indicates the number of positions in which their values are different. The Hamming distance between two functions $f, g$, denoted $d_H(f, g)$, indicates the number of positions in their truth tables where their values differ.*

It can be seen that

$$supp(x) = \{i \mid x_i \neq 0, i = 1, \ldots, n\},$$
$$supp(f) = \{x \in \mathbb{F}_2^n \mid f(x) \neq 0\}.$$

Also,

$$w_H(x) = |supp(x)|,$$
$$d_H(x, y) = w_H(x \oplus y),$$
$$w_H(f) = |supp(f)|,$$

and

$$d_H(f, g) = w_H(f \oplus g).$$

**Definition 5.** *A Boolean function $f \in \mathcal{B}_n$ is globally balanced if its truth table has an equal number of zeros and ones.*

Hence, a Boolean function $f \in \mathcal{B}_n$ is globally balanced if $w_H(f) = 2^{n-1}$. Table 1 is a clear example of this.

**Definition 6.** *A Boolean function $f \in \mathcal{B}_n$ is weight-wise perfectly balanced (WPB) when the truth tables corresponding to vectors of constant Hamming weight $1 \leq k \leq n-1$ are balanced.*

When working with WPB Boolean functions, the truth table corresponding to each vector of constant Hamming weight is referred to as a *slice*. For an n-variable Boolean function, the *slice* with constant Hamming weight $k$ is denoted $E_{n,k}$. There exist different ways to represent Boolean functions, one of which is the Algebraic Normal Form (A.N.F.).

**Definition 7.** *A Boolean function $f \in \mathcal{B}_n$ has a A.N.F. if it can be expressed as*

$$f(x_1, \ldots, x_n) := \bigoplus_{u \in \mathbb{F}_2^n} a_u \left( \prod_{i=1}^{n} x_i^{u_i} \right); \quad u = (u_1, \ldots, u_n), \quad a_u \in \mathbb{F}_2.$$

**Definition 8.** *The basic Boolean functions, over $\mathbb{F}_2^n$, are the affine functions defined by the set*

$$\mathcal{A} := \{f \in \mathcal{B}_n | f(x_1, \ldots, x_n) = a_1 x_1 \oplus \cdots \oplus a_n x_n \oplus a_0 = a \cdot x \oplus a_0\},$$

*$a = (a_1, \ldots, a_n) \in \mathbb{F}_2^n$ and $a_0 \in \mathbb{F}_2$. If $a_0 = 0$, then $f$ is called a linear function.*

**Definition 9.** *The non-linearity, denoted $NL(f)$, of the Boolean function $f$ is the Hamming distance between $f$ and the set of all affine functions.*

Non-linearity can be expressed in Hamming distance notation:

$$NL(f) := \min_{g \in \mathcal{A}} d_H(f, g).$$

Similarly, the weight-wise non-linearity refers to the non-linearity of a particular slice and is denoted $NL_k(f)$. That is, the minimum Hamming distance between the restriction of $f$ to $E_{n,k}$ and the restriction of affine functions in $\mathbb{F}_2^n$ to $E_{n,k}$. These non-linearities are given by:

$$NL_k(f) = \min_{g, deg(g) \leq 1} |supp_k(f + g)| \tag{1}$$

**Definition 10.** *Let $f \in \mathcal{B}_n$, $a \in \mathbb{F}_2^n$. The Walsh-Hadamard transform of $f$, denoted $W_f$, is defined as:*

$$W_f : \mathbb{F}_2^n \to \mathbb{F}_2, \qquad W_f(a) := \sum_{x \in \mathbb{F}_2^n} (-1)^{f(x) + a \cdot x},$$

*where $a \cdot x$ is the usual dot product on $\mathbb{F}_2^n$.*

It is also possible to express non-linearity in terms of the Fourier transform:

**Theorem 11.** *The non-linearity of the Boolean function $f \in \mathcal{B}_n$ is*

$$NL(f) = 2^{n-1} - \frac{1}{2} \max_{a \in \mathbb{F}_2^n} |W_f(a)|.$$

**Theorem 12.** *(Parseval's equality) Let $f \in \mathcal{B}_n$. Then*

$$\sum_{a \in \mathbb{F}_2^n} [W_f(a)]^2 = 2^{2n}.$$

Using Parseval's equality we know that $\max_a |W_f(a)| \geq 2^{\frac{n}{2}}$. Thus, the non-linearity of a Boolean function has an upper bound, $NL(f) \leq 2^{n-1} - 2^{\frac{n}{2}-1}$.

In particular, if $f$ is a globally balanced function with $n$-variables, then it's maximum non-linearity may be obtained using aforementioned bound when $n$ is an even number [20].

**Definition 13.** *A Boolean function is a bent function if it achieves maximum non-linearity.*

If $f$ is a balanced function, then $W_f(0) = 0$. Hence, $f$ can never achieve maximum non-linearity and, conversely, bent functions cannot be balanced. Thus, when looking for balanced Boolean functions, these will inevitably have a non-linearity below the maximum.

**Definition 14.** *The algebraic degree,* deg*, of a Boolean function $f$ is the number of variables in the longest term of the function in ANF.*

**Definition 15.** *The annihilator of a Boolean function $f$ is a Boolean function $g$ over $\mathbb{F}_2^n$ such that $f \cdot g = 0$.*

**Definition 16.** *The algebraic immunity of a Boolean function $f$ over $\mathbb{F}_2^n$, $AI(f)$, is the lowest value $d$ such that $f$ or $f \oplus 1$ admits an annihilating function of degree* deg*.*

## 3 Methodology

This Section presents the details of the evolutionary algorithm (EA) used in this investigation, with the objective of finding WPB Boolean functions with high non-linearity. First, the search space and corresponding encoding of the individuals is presented. Second, the variation operators used in the EA are analysed. Third, the fitness function used to evaluate the individuals is presented. Finally, this Section concludes by detailing the experimental settings employed for the application of the EA used in this investigation.

### 3.1 Search Space

WPB Boolean functions exist within the set of all $n$-variable Boolean functions, $\mathcal{B}_n = \{f : \mathbb{F}_2^n \to \mathbb{F}_2\}$. The naive way to search for the desired WPB Boolean functions would be to explore the entirety of $\mathcal{B}_n$. However, the size of $\mathcal{B}_n$ is super-exponential in $n$, having a cardinality of $2^{2^n}$. Moreover, WPB Boolean functions are only defined for $n = 2^m, m \in \mathbb{N}$ [17].

For $n$-variable Boolean functions with $n \neq 2^m, m \in \mathbb{N}$, Weight-wise Almost Perfectly Balanced (WAPB) functions have been defined [6], however this investigation focuses solely on WPB functions. Since the size of the search-space for WPB Boolean functions grows rapidly, an exhaustive search is only viable for $n \in \{2, 4\}$, neither of which is large enough for use in real-life situations. This investigation seeks to expand on previous research by applying an EA successfully used for the construction of 8-variable WPB Boolean functions [14], now for the construction of 16-variable WPB Boolean functions. Consequently, the cardinality of the search space for this problem is $2^{2^{16}} \approx 2.0035 \times 10^{19728}$.

Based on previous results [14,22] it was decided to use the entirety of $\mathcal{B}_n$ as the search space. This very choice of search space is precisely the reason that makes heuristics a suitable method to tackle this problem.

### 3.2 Encoding and Variation Operators

Based on the results of previous work [14], the encoding used throughout this investigation was the *WPB representation*, as described below.

It is known that a Boolean function may be uniquely represented by its truth table. While this is usually done by showing the images of the function in lexicographic order after the inputs, in the *WPB representation* this is done by dividing the image of the function into $n + 1$ vectors, each corresponding to a slice. Within each slice, the lexicographic order is set as usual. Considering

$$I = [[0], [1, 0, 1, 0], [0, 0, 0, 1, 1, 1], [1, 1, 0, 0], [1]] \tag{2}$$

an example of an individual represented in this manner for a 4-variable Boolean function is shown. This representation allows us to exploit the unique qualities of a WPB Boolean function more efficiently, by ensuring that the desired weight distribution is better preserved during the application of variation operators.

The two variation operators contemplated in the EA are the crossover and the mutation operators. In this work the crossover operator used was the *counter-based crossover (cbc)* operator. Within each slice of an individual every gene is randomly selected from one of the two parents, however two counters keep track of how many genes take a value of one and how many take a value of zero, once either of the counters reaches a value of $l/2$, where $l$ is the length of the slice, the remaining genes in the slice will take the opposite value, thus ensuring balance within each slice. This particular operator was chosen given that it demonstrated the best results in the search for 8-variable WPB Boolean functions with high non-linearity [14,15]. On the other hand, the mutation operator used was the

classic *swap* mutation, but applied to each slice of the WPB representation, as in previous research [14]. That is, with a probability $p_m$, the values of two randomly selected genes will be swapped. If the two genes chosen to be swapped have the same allele, then the function will not be altered.

### 3.3 Fitness Functions

The fitness function for the optimisation of non-linearity in the EA is Function 3, the same as was used for the construction of 8-variable WPB Boolean functions previously [14].

$$fit_{NL}(f) = \delta_{pen} \cdot \left( \min_{2 \leq k \leq 2/n} \{NL(f)\} \right) - pen(f). \tag{3}$$

Three main components can be identified in the fitness function, which when combined evaluate if a function is WPB and, if this is the case, what the non-linearity of said function is.

The non-linearity evaluated in this case is the general non-linearity as described in Section (2). However, in order to make the computations more efficient the calculation is carried out using the function from the Boolean Function module of SAGEMath System [24]. As can be seen, the term describing the non-linearity is also dependent on the $\delta_{pen}$ component, which is a Boolean component, taking a value of one when the penalisation vanishes $pen(f) = 0$, and a value of zero otherwise; which brings us to the penalty component, $pen(f)$, calculated as follows:

$$pen(f) = \sum_{k=1}^{n-1} unb_k(f), \tag{4}$$

where $unb_k(f)$ will be defined later in (5).

The WPB representation is essential for this evaluation, as it must be carried out by slices, given that a function may be globally balanced without being WPB. Slices for $k = 0$ and $k = n$ are not taken into account, this is because they have a cardinality of one, which can never be balanced. In fact, when evaluating the former are taken as having a value of 0 and the latter as having a value of 1.

The unbalancedness of a slice, $unb_k(f)$, is given by

$$unb_k(f) = \left| \frac{\#E_{n,k}}{2} - w_H(f_{(k)}) \right|, \tag{5}$$

where $\#E_{n,k}$ is the cardinality of slice $k$, which in turn is given by

$$\binom{n}{k},$$

and $w_H(f_{(k)})$ is the Hamming weight of the restriction $f_{(k)}$, given by

$$w_H(f_{(k)}) = \frac{\#E_{n,k} - W_{f_{(k)}}(0)}{2}. \tag{6}$$

When calculating the penalty, the restricted Walsh-Hadamard Transform is calculated, this is done in the same way as stated in Sect. 2, with the sole difference that now $a \in E_{n,k}$, rather than $a \in \mathbb{F}_2^n$, as in the usual case.

### 3.4 Experimental Settings

The evolutionary algorithm used for this investigation was programmed by the authors in Python. To improve efficiency certain calculations were carried out using the Boolean Function module of SAGEMath System [24], particularly the evaluations of general non-linearity, algebraic immunity, resiliance and algebraic degree. Additionally, the calculations of the weight-wise non-linearities were calculated with the aid of the code by Gini and Méaux [5]. The programs were executed using a virtual server with a *Manjaro* OS with a RAM of 16GB and 8 cores.

**Table 2.** Outline of the configuration set up used for the EA considered.

| Iterations | Pop. Size | Generations | Tournament size | $p_c$ | $p_m$ |
|---|---|---|---|---|---|
| 30 | 100 | 20,000 | 3 | 1 | 0.7 |

Table 2 shows a summary the parameters used for all the iterations of the experiment carried out as part of this investigation. All experiments were run for 16-variable Boolean functions, using their unique WPB representation, as described in Sect. 3.2. A population of 100 individuals was used in all the iterations. This was a change from the parameters used in previous work [14] on account of the resources required to evaluate 16-variable Boolean functions.

Parent selection was done through 3-tournament elimination, as in previous work [14,15]; three individuals are selected at random from the population, the two with the highest fitness are chosen as parents, and the third is replaced with the offspring after the variation operators are applied. The variation operators were applied based on the results obtained previously [14], as described in Sect. 3.2. That is, applying the counter-based crossover with a probability of $p_c = 1$, and the swap mutation with a probability of $p_m = 0.7$.

Using the described parameters, 30 iterations of the experiment were run, considering 20,000 generations in each case.

Though the EA aims to optimise the general non-linearity of a WPB Boolean function, there are other characteristics that are important for the implementation of WPB functions. Thus, the best individual in each iteration was also evaluated for the following characteristics: (a) degree, (b) algebraic immunity, (c) resiliance and (d) weight-wise non-linearity.

## 4 Results

In this Section the results of the experiment previously described are presented. In addition to the global non-linearities reached by the EA, the complementary

cryptographic parameters of the best individual in each iteration are presented to give a more global understanding of the results obtained. Based on these cryptographic parameters, a comparison is made with other known results [8] where the highest global non-linearity of $NL(f) = 32,598$ has been attained for a WPB Boolean function using algebraic methods. Unfortunately, for the function with $NL(f) = 32,598$, values are not known for any of its other cryptographic parameters, thus function $h_{16}$ [6] is also used as a reference, having the highest known weight-wise non-linearities for a 16-variable WPB Boolean function. Function $h_{16}$ was also constructed using algebraic methods. Table 3 shows the values for each cryptographic parameter of $h_{16}$.

**Table 3.** Cryptographic parameters of function $h_{16}$ [6]. In addition to the non-linearities, $h_{16}$ has $res(f) = 0$, $IA(f) = 8$ y deg $= 14$. $NL_g$ refers to the global non-linearity of the function, while $NL_i, 2 \leq i \leq 14$ specifies the restricted non-linearity of each slice. Slices $0, 1, 15$ and 16 are not considered as they have a value of zero in all cases.

| | $NL_g$ | $NL_2$ | $NL_3$ | $NL_4$ | $NL_5$ | $NL_6$ | $NL_7$ | $NL_8$ | $NL_9$ | $NL_{10}$ | $NL_{11}$ | $NL_{12}$ | $NL_{13}$ | $NL_{14}$ |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| $h_{16}$ | 30704 | 28 | 172 | 688 | 1884 | 3629 | 5103 | 5567 | 5103 | 3629 | 1884 | 688 | 172 | 28 |

To begin with, in all instances the EA managed to find WPB Boolean functions, despite the search space including infeasible, non-balanced individuals. These results are congruent with those obtained previously [14], even though $\mathcal{B}_{16}$ is a significantly larger search space than $\mathcal{B}_8$, thus showing the effectiveness of heuristic methods in approaching this problem.

Then, the fitness values reached from our experiments on each iteration are considered. To do this, the fitness value of the best individual is taken as a reference to asses the performance of the EA results. First, recall that a 16-variable bent function has non-linearity of $NL(f) = 32,640$, and that a 16-variable WPB Boolean function theoretically has a maximum non-linearity of $NL(f) = 32,638$, though in practise the highest non-linearity that has been attained is $NL(f) = 32,598$ [8].

Table 4 shows the results of the experiments. Global non-linearity consistently exceeds 32,200, closely approaching the highest non-linearity reached thus far [8] and far exceeding that of $h_{16}$. Additionally, a distribution of non-linearities has been proposed for 16-variable functions [7], which suggests that using a random search the expectation would be to find functions with non-linearities between 31,886 and 32,300, thus showing the improved results when using an EA as opposed to a random search.

Although the EA only aimed to optimise the non-linearity of WPB Boolean functions, these functions must also exhibit other characteristics. Some of these characteristics are algebraic immunity, resilience and, in this particular case, weight-wise non-linearities, as outlined in Sect. 2. Thus, the best individuals are evaluated for each of these characteristics in order to determine their robustness.

**Table 4.** Non-linearities of WPB Boolean functions generated using 20,000 generations of the EA. As complementary cryptographic parameters all Boolean functions generated had $res(f) = 0$, $IA(f) = 8$ and $\deg = 15$.

| | $NL_g$ | $NL_2$ | $NL_3$ | $NL_4$ | $NL_5$ | $NL_6$ | $NL_7$ | $NL_8$ | $NL_9$ | $NL_{10}$ | $NL_{11}$ | $NL_{12}$ | $NL_{13}$ | $NL_{14}$ |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| $\mu$ | 32311.5 | 37.0 | 229.8 | 818.8 | 2041.9 | 3810.4 | 5487.1 | 6194.7 | 5490.3 | 3807.8 | 2042.2 | 818.3 | 230.0 | 37.5 |
| $\sigma$ | 18.1 | 2.2 | 3.4 | 5.6 | 7.7 | 11.2 | 15.5 | 16.3 | 16.6 | 19.8 | 8.5 | 5.9 | 3.5 | 2.1 |
| min | 32286 | 30 | 218 | 803 | 2024 | 3792 | 5453 | 6153 | 5456 | 3745 | 2018 | 805 | 222 | 32 |
| max | 32356 | 41 | 235 | 828 | 2055 | 3834 | 5508 | 6222 | 5516 | 3835 | 2056 | 829 | 236 | 41 |

In all instances $h_{16}$ is used as a reference, as this is the function with best values as far as the authors are aware.

The weight-wise non-linearities for each slice have a value greater than zero, which in itself is a positive result. Previously, when high global non-linearities were achieved it was often the case that functions would have poor results when weight-wise non-linearities were evaluated [8]. Furthermore, when compared with $h_{16}$ the non-linearity for each slice surpasses those of $h_{16}$ in every instance. This, in combination with the high general non-linearities obtained, is an excellent result, for it shows the high quality of these functions regarding non-linearity both globally and by slice.

When considering resilience, algebraic degree and algebraic immunity, the values show consistency across all iterations. Having ensured that the functions are balanced and have high non-linearity, the third component required to ensure good resistance against correlation attacks is resilience. Unfortunately, in this case all functions have a resilience $res(f) = 0$, which is likely linked to the Siegenthaler bound [21], given that they also have maximum algebraic degree, $\deg = 15$. This result is consistent with previous results, which is one of the factors that has lead to the exploration of weight-wise almost perfectly balanced Boolean functions as an alternative to WPB functions. Algebraic immunity, on the other hand, is required to ensure functions will withstand algebraic attacks. In all cases the Boolean functions obtained have maximum algebraic immunity, $AI = 8$, which is clearly the desired result. This result is on par with that of $h_{16}$.

## 5 Conclusion and Further Work

We endeavoured to expand upon previous work on constructing WPB Boolean functions using evolutionary algorithms. These functions are particularly interesting due to their role in the FLIP [16] stream cipher. However, as the number of variables in the function increases, the complexity of constructing them also increases. Thus, this investigation aimed to extend the application of an EA previously used to construct 8-variable functions [14] to the construction of 16-variable WPB Boolean functions. Additionally, this investigation expanded the scope of the analysis by evaluating other cryptographic parameters in addition to non-linearity, which allowed a better understanding of the quality of the Boolean

functions generated. Giving continuity to the results obtained previously [14], individuals in the EA were represented using the *WPB representation* [15], and taking the entirety of $\mathcal{B}_{16}$ as the search space. Moreover, the same fitness function was used, with the only difference being that the Boolean Functions module [24] was used to conduct evaluations during implementation.

Our results show that this approach is feasible for larger-scale cases, particularly for 16-variable functions. Despite the increased search space size, the EA continued to successfully reach feasible solutions with the desired high non-linearity while searching throughout the entirety of $\mathcal{B}_{16}$. In addition, to the sought-after high non-linearity, our results also yielded good values for other cryptographic parameters, namely algebraic immunity and weight-wise non-linearities, which evidences the robustness of this approach. Notably, the results show that the results obtained using heuristic methods can match and even surpass the results obtained from random search and algebraic methods.

Going forward, we plan to adapt this approach to the search of Weight-wise Almost Perfectly Balanced (WAPB) Boolean functions, immensely increasing the scope of possibility. Particularly, the consideration of WAPB functions allows us to increase the number of variables at a more moderate pace, thus allowing the code to be gradually adjusted to larger cases. In addition, having observed the good performance of evolutionary algorithms for this problem, it would be interesting to explore the use of multi-objective evolutionary algorithms (MOEA) to optimise more than one criterion simultaneously. Having now considered the weight-wise non-linearities in this analysis, the MOEA could aim to optimise each of these non-linearities individually.

**Acknowledgements.** We thank Gini et al., authors of [6–8] for their continued support and willingness to collaborate throughout this investigation. We thank Manzoni et al., authors of [15] for their support in this investigation. The first author acknowledges the CONAHCYT scholarship to pursue graduate studies. The second author acknowledges support from project IPN-SIP 20240568. The authors have no conflicts of interest to declare that are relevant to the content of this article.

## References

1. Aguirre, H., Okazaki, H., Fuwa, Y.: An evolutionary multiobjective approach to design highly non-linear Boolean functions. In: Proceedings of the 9th Annual Conference on Genetic and Evolutionary Computation, pp. 749–756 (2007)
2. Behera, P.K., Gangopadhyay, S.: An improved hybrid genetic algorithm to construct balanced Boolean function with optimal cryptographic properties. Evol. Intel. **15**(1), 639–653 (2022)
3. Carlet, C., Guillot, P.: A new representation of Boolean functions. In: Fossorier, M., Imai, H., Lin, S., Poli, A. (eds.) Applied Algebra, Algebraic Algorithms and Error-Correcting Codes: 13th International Symposium, AAECC-13 Honolulu, Hawaii, USA, November 15–19, 1999 Proceedings, pp. 94–103. Springer, Berlin, Heidelberg (1999). https://doi.org/10.1007/3-540-46796-3_10
4. Carlet, C., Méaux, P., Rotella, Y.: Boolean functions with restricted input and their robustness; application to the flip cipher. Cryptology ePrint Archive (2017)

5. Gini, A., Méaux, P.: Wapb pub. https://github.com/agnesegini/WAPB_pub
6. Gini, A., Méaux, P.: Weightwise almost perfectly balanced functions: secondary constructions for all n and better weightwise nonlinearities. In: Isobe, T., Sarkar, S. (eds.) Progress in Cryptology – INDOCRYPT 2022: 23rd International Conference on Cryptology in India, Kolkata, India, December 11–14, 2022, Proceedings, pp. 492–514. Springer International Publishing, Cham (2022). https://doi.org/10.1007/978-3-031-22912-1_22
7. Gini, A., Méaux, P.: Weightwise perfectly balanced functions and nonlinearity. Cryptology ePrint Archive (2022)
8. Gini, A., Méaux, P.: Weightwise perfectly balanced functions and nonlinearity. In: El Hajji, S., Mesnager, S., Souidi, E.M. (eds.) Codes, Cryptology and Information Security: 4th International Conference, C2SI 2023, Rabat, Morocco, May 29–31, 2023, Proceedings, pp. 338–359. Springer Nature Switzerland, Cham (2023). https://doi.org/10.1007/978-3-031-33017-9_21
9. Jakobovic, D., Picek, S., Martins, M.S., Wagner, M.: Toward more efficient heuristic construction of Boolean functions. Appl. Soft Comput. **107**, 107327 (2021)
10. Katz, J., Lindell, Y.: Introduction to modern cryptography: principles and protocols. Chapman and Hall/CRC (2007)
11. Li, J., Su, S.: Construction of weightwise perfectly balanced boolean functions with high weightwise nonlinearity. Discret. Appl. Math. **279**, 218–227 (2020)
12. López-López, I., Sosa-Gómez, G., Segura, C., Oliva, D., Rojas, O.: Metaheuristics in the optimization of cryptographic Boolean functions. Entropy **22**(9), 1052 (2020)
13. MacWilliams, F.J., Sloane, N.J.A.: The Theory of Error Correcting Codes, vol. 16. Elsevier (1977)
14. Mandujano, S., Ku Cauich, J.C., Lara, A.: Studying special operators for the application of evolutionary algorithms in the seek of optimal Boolean functions for cryptography. In: Advances in Computational Intelligence: 21st Mexican International Conference on Artificial Intelligence, MICAI 2022, Monterrey, Mexico, October 24–29, 2022, Proceedings, Part I, pp. 383–396. Springer (2022). https://doi.org/10.1007/978-3-031-19493-1_30
15. Mariot, L., Picek, S., Jakobovic, D., Djurasevic, M., Leporati, A.: Evolutionary construction of perfectly balanced Boolean functions. arXiv preprint arXiv:2202.08221 (2022)
16. Méaux, P., Journault, A., Standaert, F.-X., Carlet, C.: Towards stream ciphers for efficient FHE with low-noise ciphertexts. In: Fischlin, M., Coron, J.-S. (eds.) Advances in Cryptology – EUROCRYPT 2016: 35th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Vienna, Austria, May 8-12, 2016, Proceedings, Part I, pp. 311–343. Springer, Berlin, Heidelberg (2016). https://doi.org/10.1007/978-3-662-49890-3_13
17. Mesnager, S., Su, S.: On constructions of weightwise perfectly balanced Boolean functions. Cryptogr. Commun. **13**(6), 951–979 (2021)
18. Picek, S., Carlet, C., Guilley, S., Miller, J.F., Jakobovic, D.: Evolutionary algorithms for Boolean functions in diverse domains of cryptography. Evol. Comput. **24**(4), 667–694 (2016)
19. Picek, S., Jakobovic, D., Miller, J.F., Marchiori, E., Batina, L.: Evolutionary methods for the construction of cryptographic Boolean functions. In: Machado, P. (ed.) Genetic Programming: 18th European Conference, EuroGP 2015, Copenhagen, Denmark, April 8-10, 2015, Proceedings, pp. 192–204. Springer International Publishing, Cham (2015). https://doi.org/10.1007/978-3-319-16501-1_16

20. Seberry, J., Zhang, X.-M., Zheng, Y.: Nonlinearly balanced Boolean functions and their propagation characteristics. In: Stinson, D.R. (ed.) Advances in Cryptology — CRYPTO' 93, pp. 49–60. Springer, Berlin, Heidelberg (2001). https://doi.org/10.1007/3-540-48329-2_5
21. Siegenthaler, T.: Correlation-immunity of nonlinear combining functions for cryptographic applications (Corresp.). IEEE Trans. Inf. Theory **30**(5), 776–780 (1984). https://doi.org/10.1109/TIT.1984.1056949
22. Singh, H.K., Alam, K., Ray, T.: Use of infeasible solutions during constrained evolutionary search: a short survey. In: Ray, T., Sarker, R., Li, X. (eds.) Artificial Life and Computational Intelligence, pp. 193–205. Springer International Publishing, Cham (2016). https://doi.org/10.1007/978-3-319-28270-1_17
23. Stinson, D.R.: Combinatorial characterizations of authentication codes. Des. Codes Crypt. **2**(2), 175–187 (1992)
24. The Sage Developers: SageMath, the Sage Mathematics Software System (Version 10.0) (2023). https://www.sagemath.org

# Discovering Rotation Symmetric Self-dual Bent Functions with Evolutionary Algorithms

Claude Carlet[1,2], Marko Đurasevic[3], Domagoj Jakobovic[3], and Stjepan Picek[1,4(✉)]

[1] Department of Mathematics, Université Paris 8, 2 rue de la liberté, 93526 Saint-Denis Cedex, France
picek.stjepan@gmail.com
[2] University of Bergen, Bergen, Norway
[3] Faculty of Electrical Engineering and Computing, University of Zagreb, Unska 3, Zagreb, Croatia
[4] Digital Security Group, Radboud University, 6500 GL, 9010 Nijmegen, The Netherlands

**Abstract.** Bent Boolean functions are interesting mathematical objects with diverse real-world applications. Besides looking at the whole class of bent functions, one could also consider subclasses like rotation symmetric bent functions or (anti)-self-dual bent functions. Such classes are naturally smaller, making it (potentially) easier to enumerate functions inside the class with a computer investigation.

This work considers a novel problem of evolving rotation symmetric (anti)-self-dual bent functions. We consider two solution encodings and a number of evolutionary algorithms. We successfully find rotation symmetric (anti)-self-dual functions for several Boolean function sizes, which are the first known examples of such functions. We hope this work will open a new research direction that will result in finding more such functions for larger dimensions, as well as algebraic constructions that will be valid for infinite Boolean function sizes.

**Keywords:** Boolean functions · rotation symmetric functions · self-dual bent functions

## 1 Introduction

Boolean functions are mathematical objects used in diverse applications, such as coding theory [15,17], combinatorics [28], and cryptography [2]. Boolean functions have also been an active research topic for more than 50 years, resulting in the discovery of many subclasses with specific properties. One such class is bent (maximally nonlinear) Boolean functions. Although bent functions may have limitations for certain applications like cryptography, they are still used in other domains like coding theory [15] and sequences [23]. While bent functions


M. Affenzeller et al. (Eds.): PPSN 2024, LNCS 15151, pp. 429–445, 2024.
https://doi.org/10.1007/978-3-031-70085-9_27

are much rarer than general Boolean functions, many still exist (see Table 2). As such, it makes sense to concentrate on specific subclasses of bent functions. Since subclasses are smaller, the functions belonging there could be easier to enumerate.

The first subclass we consider is rotation symmetric functions. Rotation symmetric functions are functions that are invariant under cyclic shifts of the input coordinates. They are interesting because such functions can have a simple structure and representation. Second, we focus on (anti-)self-dual bent Boolean functions. A bent function is called self-dual if equal to its dual, and anti-self-dual if equal to the complement of its dual [28]. Both subclasses are well-known and have been explored for many years. Still, despite all the work done, it is unknown how many such functions exist already for $n = 8$ inputs. We note that metaheuristics were also often used to search for such functions. For instance, for $n = 9$, metaheuristics searching in the space of rotation symmetric Boolean functions managed to find functions with nonlinearity 241, something that was an open question for several decades whether they even exist [14]. Moreover, we note that both rotation symmetric bent functions and (anti)-self-dual bent functions are successfully constructed with metaheuristics for various Boolean function sizes (see Sect. 3 for details).

In this work, we consider a novel application where we aim to evolve rotation symmetric (anti)-self-dual bent functions. To the best of our knowledge, such a combination was never explored, and before this work, it was not known whether such functions even existed. We conduct our search using evolutionary algorithms and consider two solution representations: bitstring and floating-point. We find rotation symmetric (anti)-self-dual functions for $n = 6, 8$. While we do not find any such function for larger sizes, we observe that bitstring representation works better for smaller $n$ and floating-point representation works better for larger $n$. Since we do not find rotation symmetric (anti)-self-dual bent functions for every tested size, we cannot answer the question of whether they exist in general or if there is something specific for small $n$ only. Still, we consider our work to open a new direction in research of bent Boolean functions, and future work could consider both metaheuristic and algebraic construction perspectives.

## 2 Preliminaries on Boolean Functions

### 2.1 Notation

We denote positive integers with $n$ and $m$. We denote the Galois (finite) field with two elements by $\mathbb{F}_2$ and the Galois field with $2^n$ elements by $\mathbb{F}_{2^n}$. An $(n, m)$-function represents a mapping $F$ from $\mathbb{F}_2^n$ to $\mathbb{F}_2^m$. When $m = 1$, the function is called a Boolean function in $n$ inputs/variables and is denoted by $f$. We endow the vector space $\mathbb{F}_2^n$ with the structure of the field, since for every $n$, there exists a field $\mathbb{F}_{2^n}$ of order $2^n$ that is an $n$-dimensional vector space. The usual inner product of $a$ and $b$ equals $a \cdot b = \bigoplus_{i=1}^{n} a_i b_i$ in $\mathbb{F}_2^n$.

### 2.2 Boolean Function Representations

The Walsh-Hadamard transform $W_f$ is a unique representation of a Boolean function $f$ that measures the correlation between $f(x)$ and the linear functions $a \cdot x$, see, e.g., [2] with the sum calculated in $\mathbb{Z}$.

$$W_f(a) = \sum_{x \in \mathbb{F}_2^n} (-1)^{f(x)+a \cdot x}. \tag{1}$$

The Walsh-Hadamard transform is very useful as many Boolean function properties can be evaluated through it. Since the complexity of calculating the Walsh-Hadamard transform with a naive approach equals $2^{2n}$, it is common to employ a more efficient method called the fast Walsh-Hadamard transform [2], where the complexity is reduced to $n2^n$.

Another unique representation of a Boolean function $f$ on $\mathbb{F}_2^n$ is by its truth table (TT). The truth table of a Boolean function $f$ is the list of pairs of function inputs (in $\mathbb{F}_2^n$) and function values, with the size of the value vector being $2^n$. The value vector is the binary vector composed of all $f(x), x \in \mathbb{F}_2^n$, with a certain order selected on $F_2^n$. Usually, as seen in, e.g., [2], one uses a vector $(f(0), \ldots, f(1))$ that contains the function values of $f$, ordered lexicographically.

*Nonlinearity.* The minimum Hamming distance between a Boolean function $f$ and all affine functions, i.e., the functions with the algebraic degree[1] at most 1 (in the same number of variables as $f$), is called the nonlinearity of $f$. The nonlinearity $nl_f$ of a Boolean function $f$ can be easily calculated from the Walsh-Hadamard coefficients, see, e.g., [2]:

$$nl_f = 2^{n-1} - \frac{1}{2} \max_{a \in \mathbb{F}_2^n} |W_f(a)|. \tag{2}$$

The Parseval relation $\sum_{a \in \mathbb{F}_2^n} W_f(a)^2 = 2^{2n}$ implies that the nonlinearity of any $n$-variable Boolean function is bounded above by the so-called covering radius bound:

$$nl_f \leq 2^{n-1} - 2^{\frac{n}{2}-1}. \tag{3}$$

Eq. (3) can be reached with equality if and only if $n$ is even, and functions reaching the bound are called bent functions.

*Balancedness.* A Boolean function $f$ is called balanced if it takes the value one the same number of times ($2^{n-1}$) as the value zero when the input ranges over $\mathbb{F}_2^n$.

[1] The algebraic degree $deg_f$ of a Boolean function $f$ is defined as the number of variables in the largest product term of the function's algebraic normal form having a non-zero coefficient, see, e.g., [17]. The algebraic normal form is a unique representation where an $n$ variable Boolean function can be considered to be a multivariate polynomial over $\mathbb{F}_2$.

*Bent Boolean Functions.* The functions whose nonlinearity equals the maximal value $2^{n-1} - 2^{n/2-1}$ are called bent. Bent Boolean functions exist for $n$ even only [2]. Bent Boolean functions are a very active research topic with applications in, e.g., coding theory [15] and telecommunications [23]. Bent Boolean functions are rare, and we know the exact numbers of bent Boolean functions for $n \leq 8$ only (see Table 2). Naturally, besides considering the whole class of bent Boolean functions, it is also possible to consider certain subsets that may be easier to analyze or provide additional properties. For more information, we refer interested readers to a survey on bent Boolean functions [5].

### 2.3 Rotation Symmetric Functions

A Boolean function over $\mathbb{F}_2^n$ is called rotation symmetric (RS) if it is invariant under any cyclic shift of input coordinates: $(x_0, x_1, \ldots, x_{n-1}) \rightarrow (x_{n-1}, x_0, x_1, \ldots, x_{n-2})$. The number of rotation symmetric Boolean functions is smaller than the number of Boolean functions, as the output value remains the same for certain input values. Stănică and Maitra used the Burnside lemma to deduce that the number of rotation symmetric Boolean functions equals $2^{g_n}$, where $g_n$ equals [29]:

$$g_n = \frac{1}{n} \sum_{t|n} \phi(t) 2^{\frac{n}{t}}, \tag{4}$$

and $\phi$ is the Euler phi function (counting the number of positive integers less than $n$ that are coprime to $n$). Thus, $g_n$ represents the number of orbits where an orbit is a rotation symmetric partition composed of vectors equivalent under rotational shifts. We provide the number of orbits for the rotation symmetric Boolean functions in Table 1. Already for $n = 10$, exhaustive search is not an option. Bent rotation symmetric functions are maximally nonlinear and invariant under any cyclic shift of input coordinates. Rotation symmetric bent functions are much rarer than general bent functions, and all RS bent functions found belong to classic classes of bent functions (i.e., were already known as bent functions when they have been found) [21].

**Table 1.** The number of orbits for the rotation symmetric Boolean functions. The number of Boolean functions for each dimension then equals $2^{g_n}$.

| variables | 4 | 6 | 8 | 10 | 12 | 14 | 16 |
|---|---|---|---|---|---|---|---|
| $g_n$ | 6 | 14 | 36 | 108 | 352 | 1182 | 4116 |

### 2.4 Self-dual Bent Functions

For a bent function $f$ on $\mathbb{F}_{2^n}$, we define its dual as the Boolean function $\widetilde{f} : \mathbb{F}_{2^n} \rightarrow \mathbb{F}_2$ satisfying:

$$2^{\frac{n}{2}} (-1)^{\widetilde{f}(x)} = W_f(x) \text{ for all } x \in \mathbb{F}_{2^n}. \tag{5}$$

The dual $\widetilde{f}$ of a bent function is also bent. A bent function $f$ is said to be self-dual if $\widetilde{f}(x) \oplus f(x) = 0$ for all $x \in \mathbb{F}_2^n$, and anti-self-dual if $\widetilde{f}(x) \oplus f(x) = 1$. Stated differently, a bent function is called self-dual if it is equal to its dual and anti-self-dual if it is equal to the complement of its dual. Self-dual bent functions are much rarer than general bent functions.

### 2.5 On the Number of Bent Functions

The numbers of Boolean functions (or upper bound values) are given in Table 2. First, we note that bent Boolean functions are rare, and we know the exact number of bent Boolean functions for $n \leq 8$ only. For rotation symmetric functions, the total number of such functions is easy to calculate from Eq. (4). However, the number of rotation symmetric bent functions is not generally known, and experimental results are available for smaller sizes only [29]. Furthermore, for the self-dual bent functions of 8 variables, we have results for quadratic functions (those with the algebraic degree at most two) only. The total number of self-dual bent functions is thus larger. Note there are as many anti-self-dual bent functions as there are self-dual bent functions. We can see that both rotation symmetric bent functions and self-dual bent functions are rare.

**Table 2.** The number of (bent) Boolean functions. Note that no known bound exists on the number of bent rotation symmetric (RS) functions. We denote self-dual bent functions by SD and note that there are as many self-dual bent functions as anti-self-dual functions.

| $n$ | | | | | | | |
|---|---|---|---|---|---|---|---|
| criterion | 4 | 6 | 8 | 10 | 12 | 14 | 16 |
| # general | $2^{16}$ | $2^{64}$ | $2^{256}$ | $2^{1024}$ | $2^{4096}$ | $2^{16384}$ | $2^{65536}$ |
| # bent | 896 | 5425430528 | $2^{106.3}$ | $2^{638}$ | $2^{2510}$ | $2^{9908}$ | $2^{39203}$ |
| # RS | $2^6$ | $2^{14}$ | $2^{36}$ | $2^{108}$ | $2^{352}$ | $2^{1182}$ | $2^{4116}$ |
| # RS bent | 8 | 48 | 15104 | – | – | – | – |
| # SD | 20 | 42896 | 104960 | – | – | – | – |

More information about Boolean functions can be found in, e.g., [2,17].

## 3 Related Work

Many works consider the design of Boolean functions with metaheuristics. The first such works can be traced to more than 25 years ago [22]. Since balanced Boolean functions have direct applications in cryptography, they were considered before bent Boolean functions. Indeed, we can trace the first application of evolutionary algorithms to evolve bent Boolean functions to 2003 [8]. Yang et al. used evolutionary algorithms and the trace representation of Boolean functions to evolve bent

Boolean functions [32]. Hrbacek and Dvorak used Cartesian Genetic Programming to evolve bent Boolean functions up to 16 variables [10]. The authors investigated various configurations of algorithms to speed up the evolution process. Since the size of the search space and the cost of evaluating individuals are common difficulties when evolving Boolean functions, this approach allowed better results than related works. Picek and Jakobovic used genetic programming to evolve algebraic constructions that are then used to construct bent Boolean functions [25]. The authors provided results of up to 24 variables. Since the constructions in the evolution process are evaluated for small sizes only, this approach avoids the computational bottleneck connected with large Boolean functions. Following a similar approach, Mariot et al. used evolutionary strategies to evolve a secondary construction based on cellular automata for quadratic bent functions [20]. Husa and Dobai used linear genetic programming to evolve bent Boolean functions and provided results up to 24 inputs [11].

Stănică et al. used simulated annealing to evolve rotation symmetric Boolean functions [30]. To our knowledge, this is the first work that considers metaheuristics and rotation symmetric Boolean functions. Kavut and Yucel used a steepest-descent-like iterative algorithm to construct imbalanced Boolean functions in 9 variables where the authors considered the generalized rotation symmetric Boolean functions [14]. Liu and Youssef used simulated annealing to construct balanced rotation symmetric Boolean functions [16]. Wang et al. employed genetic algorithms to construct rotation symmetric Boolean functions [31]. Carlet et al. investigated several evolutionary algorithms to evolve balanced and bent rotation symmetric Boolean functions [3]. They concluded that using the bitstring or floating-point representation works better than the symbolic one, which opposes common results when evolving general bent functions.

To our knowledge, there is one work that considers metaheuristics and self-dual functions. Carlet et al. used several evolutionary algorithms and showed it is possible to evolve (anti)-self-dual bent functions for Boolean functions up to 16 variables [4]. They concluded that symbolic encoding works better than the bitstring one and that the problem does not seem more difficult than evolving general bent functions (despite self-dual functions being much rarer).

Besides these subsets of bent functions, related works show that the community investigated several more options. For instance, Picek et al. considered evolving quaternary bent Boolean functions, which are a generalization of bent (binary) Boolean functions [27]. Mariot et al. used evolutionary algorithms to evolve hyper-bent Boolean functions [18]. Hyper-bent Boolean functions are a subclass of bent Boolean functions that achieve maximum distance from all bijective monomial functions. Mariot and Leporati used a genetic algorithm to evolve semi-bent Boolean functions by spectral inversion where the solutions are encoded with the Walsh-Hadamard spectrum [19]. For a recent overview of metaheuristic approaches for the design of Boolean functions, we refer readers to [7].

# 4 Methodology

## 4.1 Solution Encodings

**Bitstring Encoding.** The most common option for encoding a Boolean function is the bitstring (BS) encoding [7], which commonly represents the truth table of a Boolean function. For a Boolean function with $n$ inputs, the truth table is coded as a bit string with a length of $2^n$. However, for rotation symmetric Boolean functions, the number of truth table entries to be coded is lower and is specified in Table 1. For each evaluation, the bitstring genotype is first decoded into the full Boolean truth table (which is trivial since we know the orbits), after which we transform it into the Walsh-Hadamard spectrum and, finally, calculate the nonlinearity property.

**Floating-point Encoding.** The second approach to representing a Boolean function is the floating-point genotype, defined as a vector with continuous variables. This requires defining the translation of a vector of floating-point numbers into the corresponding genotype, which is then translated into a complete truth table (binary values). The idea behind this translation is that each continuous variable (a real number) of the floating-point genotype represents a subsequence of bits in the genotype. All real values in the floating-point vector are restricted to the interval $[0, 1]$. If the genotype size is $g_n$, the number of bits represented by a single continuous variable of the floating-point vector can vary:

$$decode = \frac{g_n}{dimension}, \tag{6}$$

where the parameter *dimension* denotes the floating-point vector size (number of real values). This parameter can be modified if the genotype size is divisible by its value. The first step of the translation is to convert each floating-point number to an integer value. As each real value must represent *decode* bits, the size of the interval decoding to the same integer value is given as $interval = \frac{1}{decode}$. To obtain a distinct integer value for a given real number, every element $d_i$ of the floating-point vector is divided by the calculated interval size, generating a sequence of integer values as $nt\_value_i = \lfloor \frac{d_i}{interval} \rfloor$. The final translation step involves decoding the integer values into a binary string that can be used for evaluation. From there, we calculate the Walsh-Hadamard spectrum and nonlinearity. For further details on using floating-point representation for evolving rotation symmetric Boolean functions, see [3].

## 4.2 Fitness Functions

To evolve bent Boolean functions, one only needs to check that the maximal absolute value in the Walsh-Hadamard transform equals $2^{\frac{n}{2}}$ (see Eq. (2)). For self-dual functions, each Walsh-Hadamard coefficient must not only be equal to this absolute value: considering Eq. (5), its sign must agree with the corresponding output value in the function's truth table. For instance, if $f(a) = 0$ for $a \in \mathbb{F}_2^n$, the corresponding coefficient $W_f(a)$ in the Walsh-Hadamard transform

must assume the value of $+2^{\frac{n}{2}}$, and $-2^{\frac{n}{2}}$ otherwise; for anti-self-dual functions, the previous values are inverted.

Therefore, our first fitness function (denoted by $fit_1$) counts the number of entries in the Walsh-Hadamard transform whose absolute value is equal to $2^{\frac{n}{2}}$ and whose sign matches the corresponding output value in the truth table. Formally, given $f : \mathbb{F}_2^n \to \mathbb{F}_2$, its fitness score under $fit_1$ is defined as:

$$fit_1(f) = |\{a \in \mathbb{F}_2^n : \ W_f(a) = 2^{\frac{n}{2}} \cdot (-1)^{f(a)}\}| \ . \tag{7}$$

Since the number of entries in the Walsh-Hadamard transform is equal to the truth table size ($2^n$), the range of this fitness function is $[0, \ldots, 2^n]$, where $2^n$ denotes the optimal value that corresponds to a self-dual bent function.

The second fitness function we employ takes a closer look into the deviation of each Walsh-Hadamard entry from the desired value. Apart from the number of correct values, as evaluated by $fit_1$, we sum the absolute differences (from either $2^{\frac{n}{2}}$ or $-2^{\frac{n}{2}}$) of every incorrect coefficient, and divide the sum with the product of the maximal possible difference ($2^{\frac{n}{2}}$) by the total number of entries ($2^n$). Consequently, the deviation part is normalized in $[0, 1]$, and its difference from 1 is simply added to the number of correct entries computed through $fit_1$. Hence, the fitness score of $f : \mathbb{F}_2^n \to \mathbb{F}_2$ under $fit_2$ is formally defined as:

$$fit_2(f) = fit_1(f) + \left[1 - \frac{\sum_{a \in \mathbb{F}_2^n} \left|2^{\frac{n}{2}} \cdot (-1)^{f(a)} - W_f(a)\right|}{2^n \cdot 2^{\frac{n}{2}}}\right] \ . \tag{8}$$

The integer part of $fit_2$ always equals the value obtained with $fit_1$. In particular, when the normalized sum of the deviations is 0 (that is, we reached an optimal solution), the difference from 1 is not added to $fit_1$. Thus, the optimal fitness value for $fit_2$ is the same as $fit_1$, i.e., $2^n$.

In the experiments, both of these fitness functions are used for both self-dual and anti-self-dual bent functions. Note that since our search is constrained to the rotation symmetric functions space only, the fitness functions do not need to add any constraints regarding it. Finally, we note that one can efficiently calculate the Walsh-Hadamard spectrum by using the fast Walsh-Hadamard transform (butterfly algorithm of complexity $n2^n$).

### 4.3 Algorithms and Parameters

**Bitstring Encoding.** For bitstring encoding (denoted by TT), we employ a steady-state selection with a 3-tournament elimination operator (denoted by SST). In each iteration of the algorithm, three individuals are chosen at random from the population for the tournament, and the worst one in terms of fitness value is eliminated. The two remaining individuals in the tournament are used with the crossover operator to generate a new child individual, which then undergoes mutation with individual mutation probability $p_{mut} = 0.5$. The mutated child replaces the eliminated individual in the population.

We use the simple bit mutation and the shuffle mutation. The simple bit mutation inverts a randomly selected bit, and the shuffle mutation shuffles

the bits within a randomly selected substring. We used one-point and uniform crossover operators. The one-point crossover combines a new solution from the first part of one parent and the second part of the other parent with a randomly selected breakpoint. The uniform crossover randomly selects one bit from both parents at each position in the child bitstring that is copied. Each time the evolutionary algorithm invokes a crossover or mutation operation, one of the previously described operators is randomly selected.

**Floating-point Encoding.** When FP encoding is used, one can vary the number of bits a single FP value will represent (*decode*, Eq., (6)). Based on related work [3], all FP-based algorithms use the same setting with $decode = 3$. The floating-point representation can be used with any continuous optimization algorithm, which increases its versatility. In our experiments, we investigated the following algorithms: Artificial Bee Colony (ABC) [13], Clonal Selection Algorithm (CLONALG) [1], CMA-ES [9], Differential Evolution (DE) [24], Optimization Immune Algorithm (OPTIA) [6], and a GA-based algorithm with steady-state selection (GA-SST), which is also used with TT and whose behavior is described above. With this encoding, the GA uses arithmetic, average, BLX-$\alpha$, heuristic, local, flat, one-point, and SBX crossover [26], as well as simple mutation (generates a random number from the given interval). Additionally, we employed an experimental algorithm based on the GA-SST selection scheme with the Hooke-Jeeves pattern search as a local search operator, denoted by Genetic Hooke-Jeeves (GHJ). Due to lack of space, we do not provide parameter values for all the algorithms but note we used the ECF software framework[2] with default parameter values [12]. All algorithms use the same stopping criterion of $10^6$ evaluations.

## 5 Experimental Results

First, we run exhaustive search for $n = 4$, which was trivial as there are only 20 self-dual bent functions (and 20 anti-self-dual functions). We then checked each of those functions if they are also rotation symmetric. We found 2 self-dual and 2 anti-self-dual functions when $n = 4$. Table 3 summarizes all the results obtained through the experimental analysis. The table represents the maximum nonlinearity values obtained for each experiment in 30 executed runs. The results are separated into two main categories, depending on whether self-dual (SD) or anti-self-dual (ASD) functions evolved and whether the fitness function $fit_1$ or $fit_2$ was used. Each group provides the results for the TT representation with SST and the FP representation with different optimization algorithms.

The results demonstrate that the optimal result was obtained in every experiment for the smallest ($n = 6$) problem size. However, for Boolean functions in 8 inputs, we see that in many cases, the optimal values are not obtained, especially for the anti-self-dual case, where the optimal value was not obtained even once. For larger problem sizes, no optimal solution was obtained. Until size 12, we see that using the TT representation resulted in better results than the FP

[2] http://solve.fer.hr/ECF/.

**Table 3.** The summary of the best-obtained experimental results. We denote self-dual functions with SD and anti-self-dual functions with ASD. The best results are in bold style.

| | | Rep. | Alg. | 6 | 8 | 10 | 12 | 14 | 16 |
|---|---|---|---|---|---|---|---|---|---|
| Self-dual (SD) | $fit_1$ | TT | SST | **64** | **256** | 385 | **718** | 910 | 1504 |
| | | FP | ABC | **64** | 224 | 288 | 569 | 875 | 1536 |
| | | FP | CLONALG | **64** | **256** | 355 | 690 | 896 | 1584 |
| | | FP | DE | **64** | 224 | 290 | 498 | 812 | 1536 |
| | | FP | GHJ | **64** | **256** | 308 | 544 | 896 | 1532 |
| | | FP | OptIA | **64** | 224 | 330 | 630 | 847 | 1536 |
| | | FP | SST | **64** | **256** | 376 | 643 | 973 | 1768 |
| | $fit_2$ | TT | SST | **64** | **256** | **420** | 708 | 938 | 1696 |
| | | FP | ABC | **64** | 216 | 330 | 606 | 966 | 1920 |
| | | FP | CLONALG | **64** | **256** | 340 | 660 | 868 | 1568 |
| | | FP | DE | **64** | 208 | 307 | 480 | 770 | 1432 |
| | | FP | GHJ | **64** | **256** | 360 | 680 | 973 | 2016 |
| | | FP | OptIA | **64** | 224 | 278 | 652 | 910 | 1576 |
| | | FP | SST | **64** | **256** | 338 | 678 | 983 | 1808 |
| Anti-self-dual (ASD) | $fit_1$ | TT | SST | **64** | 224 | 360 | 708 | 938 | 1616 |
| | | FP | ABC | **64** | 176 | 335 | 528 | 952 | 1632 |
| | | FP | CLONALG | **64** | 224 | 325 | 648 | 847 | 1608 |
| | | FP | DE | **64** | 160 | 270 | 486 | 910 | 1456 |
| | | FP | GHJ | **64** | 224 | 275 | 558 | 882 | 1560 |
| | | FP | OptIA | **64** | 182 | 305 | 630 | 854 | 1648 |
| | | FP | SST | **64** | 224 | 384 | 660 | 917 | 1744 |
| | $fit_2$ | TT | SST | **64** | 224 | 362 | 694 | 868 | 1744 |
| | | FP | ABC | **64** | 192 | 310 | 600 | 924 | 2032 |
| | | FP | CLONALG | **64** | 224 | 332 | 630 | 931 | 1568 |
| | | FP | DE | **64** | 160 | 286 | 459 | 812 | 1480 |
| | | FP | GHJ | **64** | 224 | 305 | 660 | 952 | **2064** |
| | | FP | OptIA | **64** | 184 | 324 | 606 | 847 | 1544 |
| | | FP | SST | **64** | 224 | 309 | 606 | **1085** | 1664 |

representation. However, for sizes 14 and 16, we see a reverse trend since the FP representation resulted in better results. Regarding the FP representation, no algorithm consistently achieved the best results; however, in many cases, the GHJ and SST algorithms performed better than others. Finally, regarding the fitness function, there is, again, not a consistent trend among all the experiments, but it is safe to say that $fit_2$ generally leads to better results.

To better outline the differences between the methods, Figs. 1 and 2 represent the results obtained by the 30 algorithm executions in the form of box plots. For the FP representation, only the SST and GHJ algorithms were selected, as they generally performed better than the others. For problem size 6, the algorithms obtained the optimal value in almost all executions. However, for size 8, the algorithms have obtained this optimal value only in a few runs. The figures for the remaining sizes just confirm what was said previously about the results, that the TT representation performs better for problem sizes lower than 14 and that fitness function $fit_2$ generally leads to better results than $fit_1$.

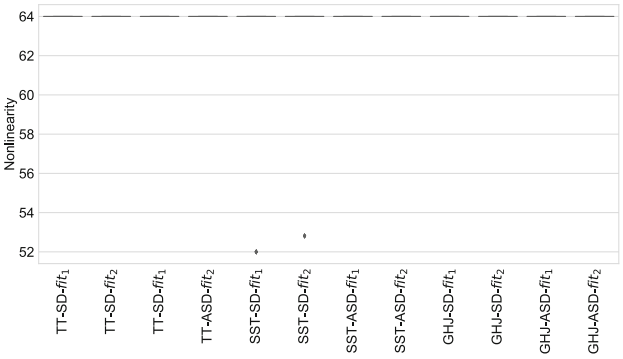
To further investigate whether the differences between the results are statistically significant, we have used the Kruskal-Wallis test with the Bonferroni correction method and Dunn's post hoc test. The critical value was set to 0.05. For brevity, for the FP, only the results obtained by GHJ were used (since this method achieved among the best results). For $n = 6$, the tests demonstrate that there is neither a difference between the representations (TT against FP) nor a difference between the two fitness functions, both for self-dual and anti-self-dual functions. For problem sizes 8, 10, and 12, the statistical tests reveal that the results obtained by the FP representation are significantly worse than those obtained by the TT representation. On the other hand, sizes 14 and 16 demonstrate a reverse trend. In that case, the results obtained using the FP representation are significantly better than those obtained by the TT representation. Furthermore, for size 16 and the FP representation, the $fit_2$ fitness function resulted in significantly better results.

Table 4 represents the execution times of the tested representations and fitness functions. For brevity, for the FP representation, only the execution times obtained by GHJ are presented in the table. All methods were coded in the C++ programming language using the ECF framework[3]. The experiments were performed on a Windows 10 PC with an AMD Ryzen Threadripper 3990X 64-core processor and 128 GB RAM. The table shows that for smaller problem sizes, all variants have the same execution time. However, as the problem size increases, we observe that the execution time of the FP representation increases more than the execution time of the TT representation. Regarding the difference between the two fitness functions, the results are not consistent. In some cases, the algorithm terminates faster when using the deviation function, and in some cases not. We postulate this happens as the deviation factor should allow more fine-grained evolution, but when solutions get stuck in local optima, the value may be too small to influence the evolution process (since nonlinearity can be even value only). Potentially, a local search that allows slightly worse solutions could be a mechanism to overcome this problem.

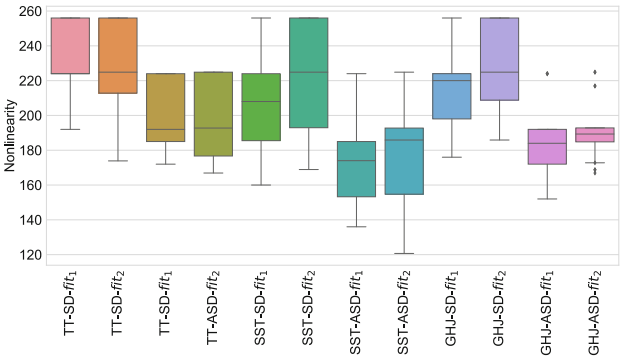
### 5.1 Limitations

The main limitation of our work is that we found rotation symmetric (anti)-self-dual bent functions in three dimensions only (4, 6, and 8 inputs). As such, while

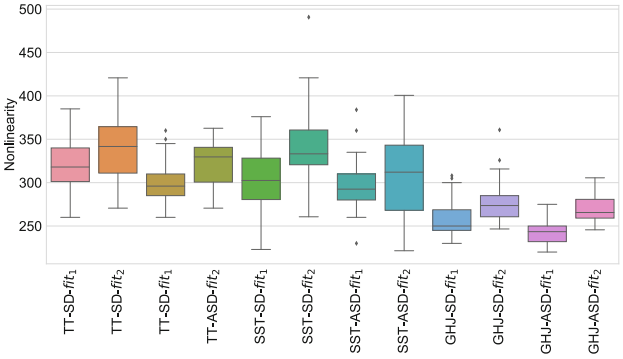
[3] http://ecf.zemris.fer.hr/.




(a) Size 6

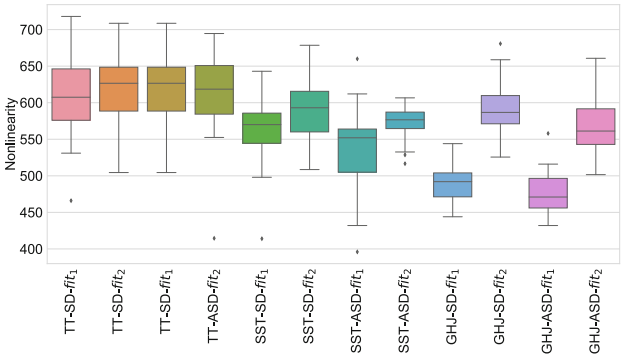



(b) Size 8

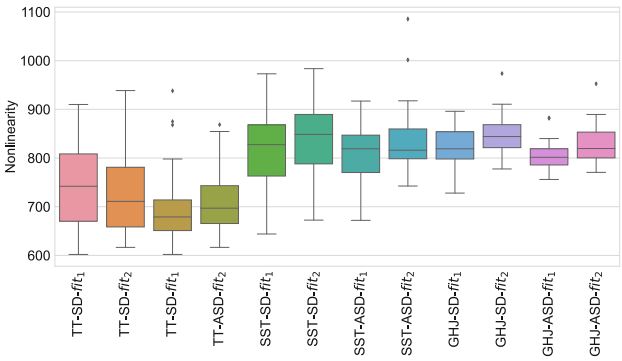



(c) Size 10

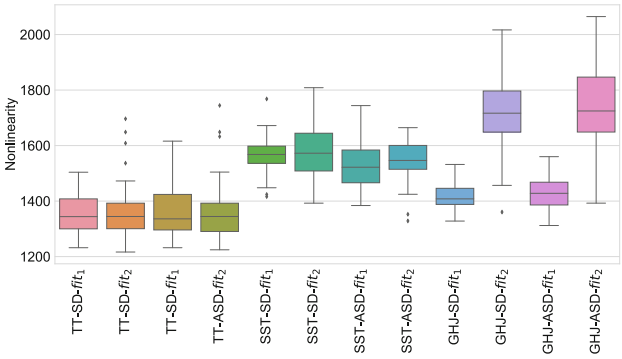
**Fig. 1.** Box plot results for problem sizes 6, 8, and 10




(a) Size 12




(b) Size 14




(c) Size 16

**Fig. 2.** Box plot results for problem sizes 12, 14, and 16

**Table 4.** Execution times (in seconds) of selected experiments.

| | | Rep. | 6 | 8 | 10 | 12 | 14 | 16 |
|---|---|---|---|---|---|---|---|---|
| SD | $fit_1$ | TT | 2 | 4 | 13 | 53 | 90 | 135 |
| | | FP | 2 | 4 | 18 | 53 | 240 | 937 |
| | $fit_2$ | TT | 2 | 4 | 13 | 52 | 65 | 227 |
| | | FP | 1 | 3 | 10 | 46 | 294 | 972 |
| ASD | $fit_1$ | TT | 2 | 5 | 13 | 51 | 32 | 274 |
| | | FP | 2 | 5 | 18 | 53 | 234 | 956 |
| | $fit_2$ | TT | 2 | 5 | 13 | 56 | 36 | 153 |
| | | FP | 1 | 4 | 12 | 48 | 250 | 1004 |

it is a good indication that such functions may exist for every $n$, more results are needed to confirm this. Moreover, from Table 2, we can see that for those sizes, it is possible to conduct an exhaustive search of rotation symmetric functions, and then, each such function can be checked if it is also bent and (anti)-self-dual. We note that it would not make sense first to try to find all (anti)-self-dual functions and then check which ones are rotation symmetric since then, we need to start from the search space of $2^{256}$. Another limitation (or, better said, trade-off) lies in the differences in the evolutionary algorithm performance considering different encodings. For rotation symmetric functions, bitstring and floating-point encodings work significantly better than the symbolic one [3], while the situation for the (anti)-self-dual function is the opposite [4]. As such, it would not necessarily make sense to consider the simultaneous evolution of rotation symmetric bent functions and (anti)-self-dual bent functions.

### 5.2 Future Work

Since this is the first work to consider the evolution of rotation symmetric (anti)-self-dual functions, and before it, it was not even known that such functions existed, there are multiple potential future research directions. First, our work considered the evolution of (anti)-self-dual functions within the space of rotation symmetric functions only. While the approach worked well for smaller sizes, it is unclear how difficult it is for the evolutionary process to converge from rotation symmetric functions to (anti)-self-dual bent ones. Thus, one option could be to first evolve (anti)-self-dual bent functions and then run an exhaustive search on those functions and check which ones are rotation symmetric. We believe this approach is more difficult than the one we followed, but it could result in different functions being obtained. Next, our evolutionary search considered several encodings and algorithms, but more thorough parameter tuning is possible, which could also result in successful results for larger $n$.

From the mathematical perspective, our results do not answer whether rotation symmetric (anti)-self-dual functions exist for every even $n$. If so, can we find an algebraic construction for it? The results we obtained could be used as the

first step in the process of finding an algebraic construction. More precisely, one could take the truth tables of all obtained functions, translate them into their algebraic forms, and try to find some pattern that would allow a construction to be built.

## 6 Conclusions

In this work, we provide a novel problem and ask whether it is possible to construct rotation symmetric (anti)-self-dual bent functions. Our answer is affirmative, as we find such functions for several (consecutive) sizes. We also observe that bitstring representation works well for smaller sizes, while floating-point representation is better for larger sizes. Considering this is a new problem, there are various possible future work directions, and we hope our research will inspire the community to try to find more such functions and better understand their properties.

## References

1. Brownlee, J., et al.: Clonal selection algorithms. Complex Intelligent Systems Laboratory, Swinburne University of Technology, Australia (2007). https://researchbank.swinburne.edu.au/file/c0a1d68c-8126-4a20-a530-eb6cb2cdd636/1/PDF technical Report 070209A
2. Carlet, C.: Boolean Functions for Cryptography and Coding Theory. Cambridge University Press, Cambridge (2021). https://doi.org/10.1017/9781108606806
3. Carlet, C., Durasevic, M., Gasperov, B., Jakobovic, D., Mariot, L., Picek, S.: A new angle: on evolving rotation symmetric boolean functions. In: Smith, S.L., Correia, J., Cintrano, C. (eds.) Applications of Evolutionary Computation - 27th European Conference, EvoApplications 2024, Held as Part of EvoStar 2024, Aberystwyth, UK, April 3-5 (2024), Proceedings, Part I. Lecture Notes in Computer Science, vol. 14634, pp. 287–302. Springer (2024). https://doi.org/10.1007/978-3-031-56852-7_19
4. Carlet, C., Durasevic, M., Jakobovic, D., Mariot, L., Picek, S.: Look into the mirror: Evolving self-dual bent boolean functions. In: Giacobini, M., Xue, B., Manzoni, L. (eds.) Genetic Programming, pp. 161–175. Springer Nature Switzerland, Cham (2024)
5. Carlet, C., Mesnager, S.: Four decades of research on bent functions. Des. Codes Cryptography **78**(1), 5–50 (2015). https://doi.org/10.1007/s10623-015-0145-8, http://dx.doi.org/10.1007/s10623-015-0145-8
6. Cutello, V., Nicosia, G., Pavone, M.: Real coded clonal selection algorithm for unconstrained global optimization using a hybrid inversely proportional hypermutation operator. In: Proceedings of the 2006 ACM Symposium on Applied computing, pp. 950–954 (2006)
7. Djurasevic, M., Jakobovic, D., Mariot, L., Picek, S.: A survey of metaheuristic algorithms for the design of cryptographic boolean functions. Cryptography Commun. **15**(6), 1171–1197 (Jul 2023). https://doi.org/10.1007/s12095-023-00662-2
8. Fuller, J., Dawson, E., Millan, W.: Evolutionary generation of bent functions for cryptography. In: Proceedings of the IEEE Congress on Evolutionary Computation, CEC 2003, Canberra, Australia, December 8-12, pp. 1655–1661. IEEE (2003)

9. Hansen, N., Müller, S.D., Koumoutsakos, P.: Reducing the time complexity of the derandomized evolution strategy with covariance matrix adaptation (CMA-ES). Evol. Comput. **11**(1), 1–18 (2003)
10. Hrbacek, R., Dvorak, V.: Bent function synthesis by means of cartesian genetic programming. In: Bartz-Beielstein, T., Branke, J., Filipič, B., Smith, J. (eds.) Parallel Problem Solving from Nature - PPSN XIII, pp. 414–423. Springer International Publishing, Cham (2014)
11. Husa, J., Dobai, R.: Designing bent Boolean functions with parallelized linear genetic programming. In: Proceedings of the Genetic and Evolutionary Computation Conference Companion, pp. 1825–1832. GECCO '17, Association for Computing Machinery, New York, NY, USA (2017). https://doi.org/10.1145/3067695.3084220
12. Jakobovic, D., Đurasević, M., Picek, S., Gašperov, B.: ECF: A C++ framework for evolutionary computation. SoftwareX **27**, 101640 (2024). https://doi.org/10.1016/j.softx.2024.101640, https://www.sciencedirect.com/science/article/pii/S2352711024000116
13. Karaboga, D., Gorkemli, B., Ozturk, C., Karaboga, N.: A comprehensive survey: artificial bee colony (ABC) algorithm and applications. Artif. Intell. Rev. **42**, 21–57 (2014)
14. Kavut, S., Yücel, M.D.: 9-variable Boolean functions with nonlinearity 242 in the generalized rotation symmetric class. Inf. Comput. **208**(4), 341–350 (2010). https://doi.org/10.1016/j.ic.2009.12.002, https://www.sciencedirect.com/science/article/pii/S0890540109002454
15. Kerdock, A.: A class of low-rate nonlinear binary codes. Inf. Control **20**(2), 182–187 (1972)
16. Liu, W.M., Youssef, A.: On the existence of $(10, 2, 7, 488)$ resilient functions. IEEE Trans. Inf. Theory **55**(1), 411–412 (2009). https://doi.org/10.1109/TIT.2008.2008140
17. MacWilliams, F.J., Sloane, N.J.A.: The Theory of Error-Correcting Codes. Elsevier, Amsterdam, North Holland (1977). ISBN: 978-0-444-85193-2
18. Mariot, L., Jakobovic, D., Leporati, A., Picek, S.: Hyper-bent Boolean functions and evolutionary algorithms. In: Sekanina, L., Hu, T., Lourenço, N., Richter, H., García-Sánchez, P. (eds.) Genetic Programming, pp. 262–277. Springer International Publishing, Cham (2019)
19. Mariot, L., Leporati, A.: A genetic algorithm for evolving plateaued cryptographic Boolean functions. In: Dediu, A.-H., Magdalena, L., Martín-Vide, C. (eds.) TPNC 2015. LNCS, vol. 9477, pp. 33–45. Springer, Cham (2015). https://doi.org/10.1007/978-3-319-26841-5_3
20. Mariot, L., Saletta, M., Leporati, A., Manzoni, L.: Heuristic search of (semi-)bent functions based on cellular automata. Nat. Comput. **21**(3), 377–391 (2022)
21. Mesnager, S.: Bent Functions. Springer International Publishing, Cham (2016). https://doi.org/10.1007/978-3-319-32595-8
22. Millan, W., Clark, A., Dawson, E.: An effective genetic algorithm for finding highly nonlinear Boolean functions. In: Han, Y., Okamoto, T., Qing, S. (eds.) Information and Communications Security, pp. 149–158. Springer, Berlin Heidelberg, Berlin, Heidelberg (1997)
23. Olsen, J., Scholtz, R., Welch, L.: Bent-function sequences. IEEE Trans. Inf. Theory **28**(6), 858–864 (1982)
24. Pant, M., Zaheer, H., Garcia-Hernandez, L., Abraham, A., et al.: Differential evolution: a review of more than two decades of research. Eng. Appl. Artif. Intell. **90**, 103479 (2020)

25. Picek, S., Jakobovic, D.: Evolving algebraic constructions for designing bent Boolean functions. In: Proceedings of the Genetic and Evolutionary Computation Conference 2016, pp. 781–788. GECCO '16, Association for Computing Machinery, New York, NY, USA (2016). https://doi.org/10.1145/2908812.2908915
26. Picek, S., Jakobovic, D., Golub, M.: On the recombination operator in the real-coded genetic algorithms. In: 2013 IEEE Congress on Evolutionary Computation, pp. 3103–3110 (2013). https://doi.org/10.1109/CEC.2013.6557948
27. Picek, S., Knezevic, K., Mariot, L., Jakobovic, D., Leporati, A.: Evolving bent quaternary functions. In: 2018 IEEE Congress on Evolutionary Computation, CEC 2018, Rio de Janeiro, Brazil, July 8-13, 2018, pp. 1–8. IEEE (2018)
28. Rothaus, O.: "On bent" functions. J. Comb. Theory, Ser. A **20**(3), 300–305 (1976)
29. Stănică, P., Maitra, S.: Rotation symmetric Boolean functions-count and cryptographic properties. Discrete Appl. Math. **156**(10), 1567–1580 (2008). https://doi.org/10.1016/j.dam.2007.04.029, https://www.sciencedirect.com/science/article/pii/S0166218X07001734
30. Stănică, P., Maitra, S., Clark, J.A.: Results on rotation symmetric bent and correlation immune Boolean functions. In: Roy, B.K., Meier, W. (eds.) Fast Software Encryption, 11th International Workshop, FSE 2004, Delhi, India, February 5–7, 2004, Revised Papers. Lecture Notes in Computer Science, vol. 3017, pp. 161–177. Springer, Berlin, Heidelberg (2004)
31. Wang, Y., Gao, G., Yuan, Q.: Searching for cryptographically significant rotation symmetric boolean functions by designing heuristic algorithms. Secur. Commun. Netw. **2022**, 1–6 (2022). https://doi.org/10.1155/2022/8188533
32. Yang, M., Meng, Q., Zhang, H.: Evolutionary design of trace form bent functions. Cryptology ePrint Archive, Paper 2005/322 (2005). https://eprint.iacr.org/2005/322, https://eprint.iacr.org/2005/322

# Author Index




M. Affenzeller et al. (Eds.): PPSN 2024, LNCS 15151, pp. 447–448, 2024.
https://doi.org/10.1007/978-3-031-70085-9