Towards a Small Diverse Pareto-optimal Solutions Set Generator for Multiobjective Optimization Problems

Courtney Powell
Hokkaido University
Sapporo 060-0811
Japan
kotoni@ist.hokudai.ac.jp

Katsunori Miura
Kitami Institute of Technology
Kitami 090-8507
Japan
k-miura@mail.kitami-it.ac.jp

Masaharu Munetomo
Hokkaido University
Sapporo 060-0811
Japan
munetomo@iic.hokudai.ac.jp

ABSTRACT

Multiobjective evolutionary algorithms (MOEAs) try to produce enough and sufficiently diverse Pareto-optimal tradeoff solutions to cover the entire Pareto surface. However, in practical scenarios, presenting numerous solutions to stakeholders may result in confusion and indecision. This paper proposes a method for generating a small (user-specified) number of well-distributed Pareto-optimal feasible solutions for multiobjective problems. The proposed method can be applied to a set of aggregate solutions produced by (1) one MOEA over multiple runs, (2) several different MOEAs, or (3) a universal set of feasible solutions produced by one or more constraint solvers.

CCS CONCEPTS

• **Theory of computation → Multiobjective algorithms**;

KEYWORDS

Cloud resource allocation, constrained optimization, multiobjective optimization, NSGA-III, workflow optimization.

ACM Reference format:

C. Powell, K. Miura, and M. Munetomo. 2018 Towards a Small Diverse Pareto-optimal Solutions Set Generator for Multiobjective Optimization Problems. In *Proceedings of ACM GECCO conference, Kyoto, Japan, July 2018 (GECCO'18)*, 2 pages.
DOI: 10.1145/nnnnnnn.nnnnnnn

1 INTRODUCTION

In multiobjective optimization, the goal is to find optimized solutions for problems involving two or more objectives. When the objectives are in conflict, the result is an optimal set of tradeoff solutions that ideally comprises solutions for which no further improvement in any objective is possible without degradation in the other objectives. Multiobjective evolutionary algorithms (MOEAs) are used extensively to solve these complex problems as they can simultaneously optimize the conflicting objectives and find and maintain multiple solutions in one single simulation run [1]. However, depending on the problem and number of objectives, this can result in an MOEA producing hundreds of solutions as it is desired that a sufficient number of Pareto-optimal tradeoff solutions be found to cover the entire Pareto surface. Assuming that the number of solutions produced by an MOEA is equal to the number of individuals in its population, then the number of solutions cannot be reduced below a certain number as small population sizes result in interbreeding and poor convergence [2-4]. In fact, recommended population sizes can range between 90 and 300 [5].

In practical applications, this may result in confusion and indecision for stakeholders as each solution will in fact differ only marginally from its neighboring solutions in objective space. Further compounding this situation is the fact that one solution in objective space can correspond to multiple solutions in decision space. To overcome this dilemma, this paper proposes a method for generating a small (user-specified) number of well-distributed Pareto-optimal feasible solutions for multiobjective problems.

2 PROPOSED SMALL DIVERSE PARETO-OPTIMAL SOLUTIONS SET GENERATOR

Figure 1 illustrates the procedure employed by the proposed generator. It starts with an initial population chosen from the universal set of solutions, with the number of individuals determined by the user. Nondominated sorting is then carried out, and each individual is associated with a reference point from the set of supplied reference points. This is followed by niching, in which the lowest-ranked individuals are discarded and the remaining individuals are combined with a new set of solutions retrieved from the universal set. This process is repeated until there are no remaining solutions in the universal set. The nondominated sorting, reference points association, and niching processes are based on the processes in NSGA-III [5].

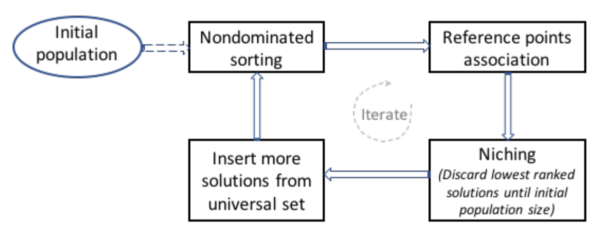

Figure 1: Procedure employed by the proposed generator.

3 CASE STUDY: GENOME ANALYSIS WORKFLOW CLOUD RESOURCE OPTIMIZATION

Selecting and configuring the most efficient combination of infrastructure components from the wide variety of components offered by cloud service providers (CSPs) is extremely challenging because of the need to satisfy various constraints associated with system performance, running cost, resource availability, resource reliability, makespan, task precedence, etc. The intercloud brokerage method proposed by Miura et al. [6] presents users with a universal set of feasible infrastructure deployment configurations from which to choose. However, while the method guarantees that the configurations generated satisfy user constraints and requirements, it generates numerous feasible configuration patterns and does not specify which configurations are optimal. The proposed generator can be applied in this scenario to find a user-specified number of diverse Pareto-optimal configurations for a set of user objectives as outlined below.

3.1 Sample Workflow and Objectives

The sample genome analysis workflow utilized in this case study is shown in Fig. 2. The goal is to optimize the assignment of the workflow tools (FastQC, TopHat2, and Cufflinks) to virtual machine (VM) instances. Table 1 shows the various parameters and user objectives considered.

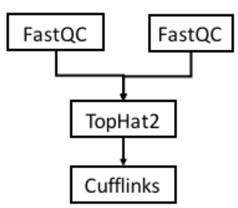

Figure 2: Sample genome analysis workflow.

Table 1: Parameters and objectives used in the case study

Parameter	Value
Feasible resource configurations (solutions) from constraint solver	1120
Desired no. of optimized solutions	6
User objectives	Availability *(maximize)*, Cost, Makespan *(minimize)*
AWS regions	Tokyo, Sydney
AWS instance families	m4, c4

For the parameters and values presented in Table 1, the proposed generator produced the six solutions presented in Table 2. The results show that of the 1120 feasible solutions, only two are Pareto-optimal. However, these two solutions each correspond to two resource configurations, where each configuration comprises four sub-lists that, respectively, correspond to the four tools in the workflow. Each sub-list has the form *[toolname, VM-type, location]*. Solution 1 corresponds to the following resource configurations (boldface highlights the differences):

1. [[FastQC, m4.large, Sidney], [FastQC, m4.large, Sidney], [TopHat2, m4.large, Sidney], **[Cufflinks, m4.large, Sidney]]**
2. [[FastQC, m4.large, Sidney], [FastQC, m4.large, Sidney], [TopHat2, m4.large, Sidney], **[Cufflinks, m4.large, Tokyo]**

Solution 2 corresponds to the following resource configurations:

1. [[FastQC, m4.large, Sidney], **[FastQC, m4.large, Sidney]**, [TopHat2, m4.large, Sidney], [Cufflinks, m4.4xlarge, Sidney]]
2. [[FastQC, m4.large, Sidney], **[FastQC, m4.large, Tokyo]**, [TopHat2, m4.large, Sidney], [Cufflinks, m4.4xlarge, Sidney]]

Table 2: Generated solutions

Solution No.	Objective values **[Cost, Makespan, Avail.]**	Rank
1	**[3.53, 20.09, 0.9984]**	**1**
2	**[6.23, 20.09, 0.9985]**	**1**
3	[3.54, 20.09, 0.9984]	2
4	[6.24, 20.09, 0.9985]	2
5	[3.55, 20.09, 0.9984]	3
6	[6.33, 20.09, 0.9985]	3

4 CONCLUSION

This paper proposed a small diverse Pareto-optimal solutions set generator for multiobjective optimization problems that is applicable to a set of aggregate solutions produced by one MOEA over multiple runs, several different MOEAs, or a universal set of feasible solutions produced by one or more constraint solvers. Obtaining a small set of Pareto-optimal solutions in this manner enables stakeholders to easily choose a Pareto-optimal configuration based on their desired preference. For example, it is clear that Solution 1 gives a Pareto-optimal configuration with a lower cost (3.53) than Solution 2 (6.23), whereas Solution 2 gives one with a higher availability (0.9985) than Solution 1 (0.9984).

ACKNOWLEDGMENT

This work is supported by CREST, Japan Science and Technology Agency (Grant No. JPMJCR1501).

REFERENCES

[1] K. Deb. 2001. *Multi-objective optimization using evolutionary algorithms*. John Wiley & Sons.
[2] V.K. Koumousis and C.P. Katsaras. 2006. A saw-tooth genetic algorithm combining the effects of variable population size and reinitialization to enhance performance. IEEE Trans Evol Comput, 10, 1 (2006) 19-28.
[3] J.T. Alander. 1992. On optimal population size of genetic algorithms. In *Proceedings of CompEuro'92*, 65-69.
[4] A. Piszcz and T. Soule. 2006. Genetic programming: Optimal population sizes for varying complexity problems. In *Proceedings of the Genetic and Evolutionary Computation Conference,* 953–954.
[5] K. Deb and H. Jain. 2014. 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.
[6] K. Miura, T. Ohta, C. Powell, and M. Munetomo. 2016. Intercloud brokerages based on PLS method for deploying infrastructures for big data analytics. In *Proceedings of the 2016 IEEE International Conference on Big Data (Big Data)*, 2097-2102.