# Knowledge-Driven Reference-Point Based Multi-Objective Optimization: First Results

Henrik Smedberg
School of Engineering Science, University of Skövde
541 28 Skövde, Sweden
henrik.smedberg@his.com

## ABSTRACT

Multi-objective optimization problems in the real world often involve a decision maker who has certain preferences for the objective functions. When such preferences can be expressed as a reference point, the goal of optimization changes from generating a complete set of Pareto-optimal solutions to generating a small set of non-dominated solutions close to the reference point. Reference-point based optimization algorithms are used for this purpose. The preferences of the decision maker in the objective space can be interpreted as knowledge in the decision space. Extracting this knowledge iteratively from the solutions generated during optimization, and feeding it back into the optimization algorithm can in principle improve convergence towards the reference point. Since the knowledge is extracted during runtime, this approach is termed as *online knowledge-driven optimization*. In this paper a recent knowledge discovery technique called flexible pattern mining is used to extract explicit rules that are used to generate new solutions in R-NSGA-II. The performance of the proposed FPM-R-NSGA-II is demonstrated on 3, 5 and 10 objective DTLZ problems. In addition to converging to a set of preferred solutions, FPM-R-NSGA-II also converges to a set of explicit rules which describe the decision maker's preferences in the decision space.

## CCS CONCEPTS

• **Theory of computation → Evolutionary algorithms**; • **Applied computing → Multi-criterion optimization and decision-making**;

## KEYWORDS

multi-objective optimization, decision making, knowledge discovery, reference-point

**ACM Reference Format:**
Henrik Smedberg. 2019. Knowledge-Driven Reference-Point Based, Multi-Objective Optimization: First Results. In *Genetic and Evolutionary Computation Conference Companion (GECCO '19 Companion), July 13–17, 2019, Prague, Czech Republic.* ACM, New York, NY, USA, 4 pages. https://doi.org/10.1145/3319619.3326911


*GECCO '19 Companion, July 13–17, 2019, Prague, Czech Republic*

ACM ISBN 978-1-4503-6748-6/19/07…$15.00
https://doi.org/10.1145/3319619.3326911

## 1 INTRODUCTION

Many real-world optimization problems involve multiple conflicting objectives and therefore lead to multiple Pareto-optimal solutions. Mathematically, a multi-objective optimization problem (MOOP) is defined as:

$$Minimize: \mathbf{F}(\mathbf{x}) = \{f_1(\mathbf{x}), f_2(\mathbf{x}), ..., f_M(\mathbf{x})\}$$
$$Subject\ to: \mathbf{x} \in \mathbf{S}$$

where $\mathbf{F}(\mathbf{x})$ is a set of $M$ objective functions, $\mathbf{x}$ is a vector of $n$ variables representing a solution and $\mathbf{S}$ is the feasible region of the search space. The search space may be constrained, but the feasible region $\mathbf{S}$ must be non-empty. Given that the Pareto-optimal solutions are available, the task of a decision maker (DM) is to choose one (or a few) solution(s) for practical implementation. Due to their population-based nature, Multi-Objective Evolutionary Algorithms (MOEAs) are often used for solving MOOPs. In many cases, however, the DM has some *a priori* preferences which can be supplied to the optimization algorithm. One of the most common ways of expressing preferences is to provide a reference-point. The goal of reference-point based algorithms is to generate only a small set of solutions close to the reference point provided by the DM, thus saving the computational expense of finding all Pareto-optimal solutions.

In this preliminary study, an exploration of how such reference-point based algorithms can be improved through *knowledge-driven optimization* is conducted.

### 1.1 Knowledge Discovery in MOO

There is a growing interest in discovering knowledge through the analysis of solutions generated from MOO. With the motivation to supply the DM with an understanding of the relationships between the decision space and the objective space. Such an analysis may reveal both the impact certain variables have in the objective space, and also what causes a solution to be Pareto-optimal. An early example this kind of analysis can be found in [5] where the authors propose *innovization* (innovation through optimization) as a manual approach to finding relationships in the variables from the Pareto-optimal set of solutions. In [1], a thorough review of different ways to extract knowledge from MOO solutions is offered.

Knowledge from MOO solutions can be represented in many different ways and there are many data mining methods and machine learning techniques available for knowledge discovery. Different knowledge representations can broadly be categorized into two types, implicit and explicit. Implicit representations lack formal notation and the knowledge cannot easily be transferred or processed automatically. Explicit representations on the other hand have a standard form that can easily be interpreted, such as the **`if-then`**

statements produced by association rule mining, or different methods of descriptive statistics. Given the standard notation of explicit knowledge, it can be stored in a knowledge base and retrieved for future application, or used programmatically in an algorithm.

## 2 KNOWLEDGE-DRIVEN OPTIMIZATION

Knowledge-Driven Optimization (KDO) is the application of knowledge discovery techniques on MOO solutions with the aim of using the discovered knowledge to enhance the current or future optimization runs [1]. The former would be achieved through *online* KDO by applying knowledge discovery techniques *during* a currently executing optimization run, and the latter through *offline* KDO by applying knowledge discovery techniques on the solutions obtained from a completed optimization run.

### 2.1 Offline KDO

In Multi-Criteria Decision Making (MCDM), *a-posteriori* methods allow the DM to analyze a complete set of solutions that represent the Pareto-optimal front. Solutions are usually analyzed using scatter plots, and parallel coordinate plots. However, the knowledge obtained through such graphical methods is implicit and cannot be used effectively in future optimization tasks of a similar nature. Instead, if explicit knowledge can be extracted from the solutions, it can be stored in a knowledge base for future use. On one hand, the extracted knowledge provides the DM a greater insight into the solutions and reveals the relationship between the decision space and the objective space. On the other hand, the extracted knowledge can also be used for implementing offline KDO algorithms. An offline KDO algorithm would utilize an expert system to search for appropriate rules in the knowledge base and use them to either redefine a future optimization scenario or alter the search behaviour of the MOEA. The expert system would employ several criteria to select the rules from the knowledge base. These criteria may include, details about MOOP formulation such as objectives, constraints and variables and their bounds, the type of MOEA used and its parameter settings, the region of preference, etc.

### 2.2 Online KDO

Online KDO aims to use knowledge discovery techniques *online*, i.e during an optimization run to extract knowledge concerning the good or preferred solutions, and apply it to direct the search towards better or more preferred solutions.

The idea of online KDO is not entirely new. The Learnable Evolution Model (LEM) proposed in [10] uses rules to guide solutions towards better regions of the search space in the context of single objective optimization. LEM alternates between a machine learning mode and a regular evolutionary mode. Another technique that has received some attention in the literature is the Estimation of Distribution Algorithm (EDA) [8]. It explores the search space by sampling new solutions from probabilistic models built from promising solutions.

Online KDO can be achieved with implicit or explicit knowledge. However, using an explicit representation has the additional advantage of evolving human-interpretable knowledge along with faster convergence.

### 2.3 Handling Different Variable Types in KDO

In order to apply KDO, an appropriate form for representing explicit knowledge should be chosen. This choice is governed by the types of variables involved. MOOPs generally involve two types of variables: (i) continuous variables, which can take any real value between their bounds, and (ii) discrete variables, which can take a countable number of values. Discrete variables are further categorized as: (a) integers, (b) practically discrete, (c) ordinal categorical, and (d) nominal categorical. Continuous and integer variables do not need any introduction. Practically discrete variables can only take a value from a predefined set of real numbers. Categorical variables are those for which the numerical value has no significance, but only a programmatic convenience. Ordinal categorical variables can take values which have an implicit ordering among them. For example, a variable representing a temperature setting of {`Low`, `Medium`, `High`} is ordinal. Categorical variables with no implicit ordering are called nominal variables. An example is a variable representing machine types {`Machine A`, `Machine B`, `Machine C`}. Association rules is a form of explicit knowledge that can be used for all the above variable types. Hence, they are used in the KDO algorithm described in the next section.

## 3 PROPOSED KDO ALGORITHM

A recent method for extracting rules from a MOO datasets was introduced in [2]. The method, Flexible Pattern Mining (FPM) is based on the popular a priori algorithm which is used for sequential pattern mining (SPM). The main difference between them is that while SPM only finds rules where variables take certain values, eg. $\{x_i = c\}$, FPM extends the a priori algorithm to also extract rules where a variable can take a range of values, eg. $\{x_i < c_1\}$ or $\{x_i > c_2\}$. By supplying a set of *selected* solutions and a set of *unselected* solutions, FPM is able to generate rules that distinguish the former from the latter, ensuring that a minimum percentage of solutions in the selected set are covered. This percentage is referred to as the *significance*, *sig* of the extracted rule. Originally, FPM was proposed to only generate up to three significant rules per variable, i.e. $x_i < c_1$, $x_i > c_2$ and $x_i = c_3$. In this paper, FPM has been modified to generate a set of all significant rules pertaining to a variable. Given the set of significant rules, distributions over the variables in the selected set of solutions can be generated and later sampled.

FPM can be used for both online and offline KDO. The general framework for applying FPM in online KDO can be defined as:

(1) choose a selected and unselected set of solutions from the current generation,
(2) perform FPM to extract rules based on the selected and unselected sets of solutions.
(3) generate a new generation of solutions using the extracted rules.

The knowledge discovery technique can be modular and interchangeable, and resides outside the actual optimization algorithm,

In this paper, online KDO is used to modify the popular R-NSGA-II [6] algorithm through rule-based mutation described in the next section. R-NSGA-II is a preference based algorithm that extends the well known NSGA-II [4] algorithm to incorporate the preferences of the DM through the specification of reference points. R-NSGA-II

modifies the selection operator of regular NSGA-II to converge on the Pareto-optimal front, near the supplied reference points.

### 3.1 Rule-Based Mutation

The proposed approach, FPM-R-NSGA-II, modifies the mutation operator of R-NSGA-II by generating a sample based on a distribution generated from the FPM rules that describe all optimization variables.

The mutated value is obtained using an empirical probability distribution constructed from the rules extracted by FPM. Consider the rules:

(1) $x_1 < 0.1, sig = 0.1$
(2) $x_1 < 0.3, sig = 0.6$
(3) $x_1 < 0.6, sig = 0.7$
(4) $x_1 < 0.9, sig = 0.9$
(5) $x_1 < 1.0, sig = 1.0$

The first rule indicates that 10% of the selected set of solutions had a value of $x_1$ less than 0.1, and the second rule indicates that 60% of the selected set had a value of $x_1$ less than 0.3, from which we can derive that 50% of the selected set had a value between 0.1 and 0.3. The empirical distribution constructed from the given rules is shown in Figure 1. This distribution is sampled to find the mutated value of $x_1$.

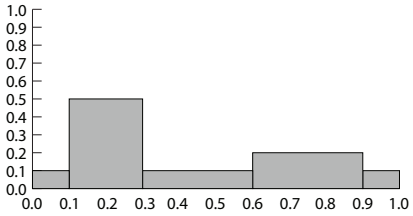



**Figure 1: A probability distribution generated from the five rules mentioned above.**

FPM generates rules of both `less than` < and `greater than` > type. However, to generate the above distribution, all rules must have the same sign. Therefore, any rules of > type are converted to < by modifying the significance as $1 - sig$.

### 3.2 Experimental Setup

In this study, the mean convergence rates of the two algorithms, R-NSGA-II and FPM-R-NSGA-II are analyzed and compared through the use of convergence plots. While many performance metrics have been proposed for comparing MOEAs, only a few exist for comparing reference-point based algorithms. In this paper, the recently proposed Expanding Hypercube-metric (EH-metric) [3] is used to compare the algorithms.

The EH-metric is calculated by expanding a hypercube with the reference point at its centre. The hypercube expands until all non-dominated solutions generated by an algorithm are enclosed. The EH-metric is defined as the area under the curve generated by plotting the size of a hypercube against the number of unique solutions it envelops as it expands. The greater this area, the better performing the algorithm is. The size of the hypercube represents convergence to the reference point, while the number of unique solutions represents diversity. The main advantage of EH-metric is that it combines the two main aspects of trade-off solutions, i.e. convergence and diversity, into a single metric. Thus, it is similar in principle to the hypervolume metric. Unlike many other reference-point based performance metrics, as for instance R-metric [9], EH-metric has no user-specified parameters and does not rely on hypervolume or inverted generational distance to calculate the performance of a dataset.

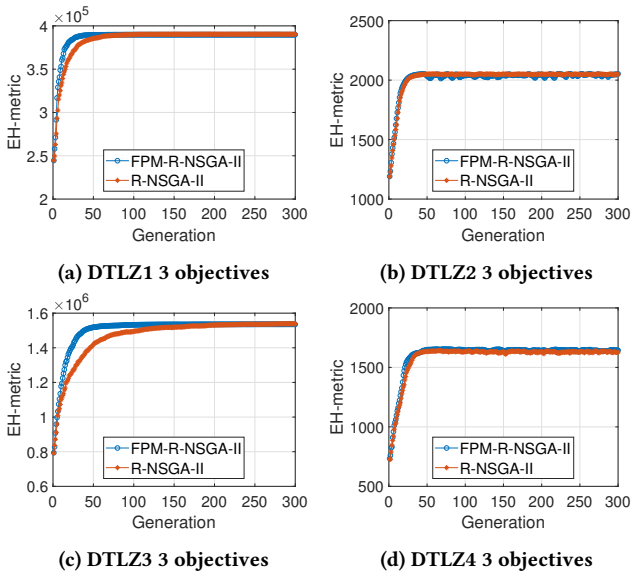
The test problems considered in the study are DTLZ1-4 from the DTLZ suite of scalable MOOPs [7]. Problems with 3, 5 and 10 objectives are considered. Each experiment was replicated 11 times for all test cases.

R-NSGA-II used *simulated binary crossover* (SBX) and *polynomial mutation* with the parameters for crossover probability $p_c$ set to 0.9 and crossover distribution index $\eta_c$ set to 10, mutation probability $p_m$ set to $1/n$ where $n$ is the number of variables and the mutation distribution index $\eta_m$ set to 20. R-NSGA-II also has a parameter $\epsilon$ to control the spread of solutions around the reference point, $\epsilon$ was set to 0.002. The population size $N$ was set to 100 individuals and to terminate the runs an evaluation budget of $100 \times N \times M$ was used, where $M$ is the number of objectives in the problem.

FPM-R-NSGA-II uses the same values for parameters shared with R-NSGA-II. The minimum significance for FPM was set to 0.5, and the selected set is defined to be the $k$-nearest neighbours of the reference point chosen from the non-dominated solutions of the current generation. All remaining non-dominated solutions form the the unselected set. The value of $k$ is set to half the size of the current non-dominated set. All parameters remain unchanged for all experiments.

## 4 RESULTS AND DISCUSSION

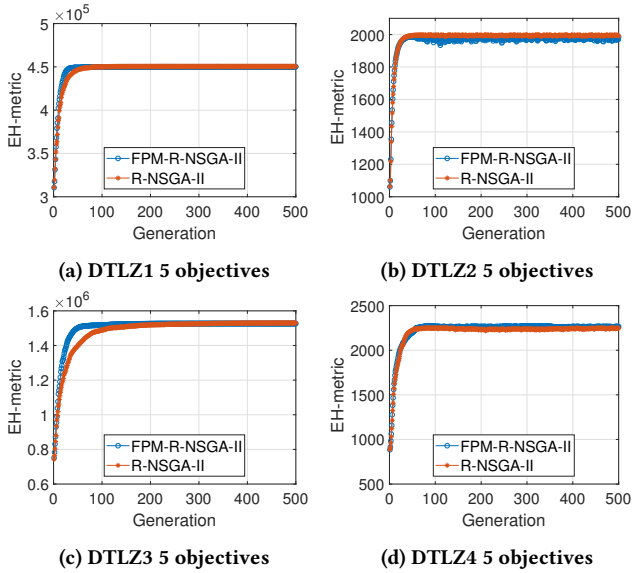
The convergence plots for the 3, 5 and 10 objective cases are shown in Figures 2, 3 and 4 respectively.




**(a) DTLZ1 3 objectives** **(b) DTLZ2 3 objectives**

**(c) DTLZ3 3 objectives** **(d) DTLZ4 3 objectives**

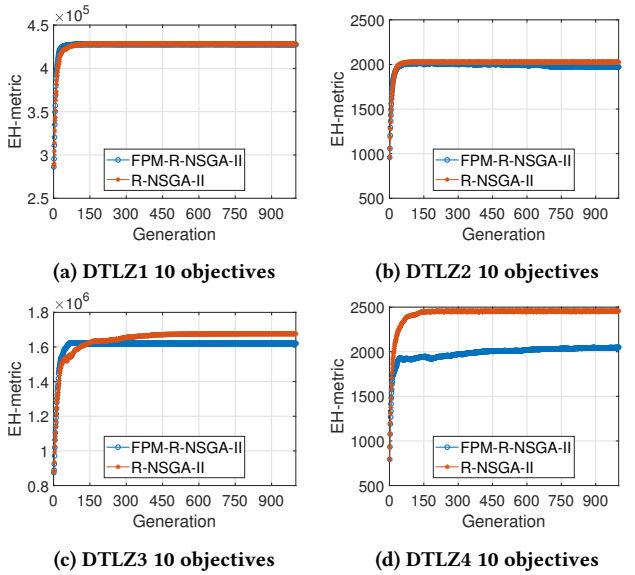
**Figure 2: Convergence plots for DTLZ1-4 for 3 objectives.**




(a) DTLZ1 5 objectives (b) DTLZ2 5 objectives

(c) DTLZ3 5 objectives (d) DTLZ4 5 objectives

**Figure 3: Convergence plots for DTLZ1-4 for 5 objectives.**




(a) DTLZ1 10 objectives (b) DTLZ2 10 objectives

(c) DTLZ3 10 objectives (d) DTLZ4 10 objectives

**Figure 4: Convergence plots for DTLZ1-4 for 10 objectives.**

In the 3-objective cases FPM-NSGA-II is able to converge faster than R-NSGA-II in the for DTLZ1 and 3. For DTLZ3, FPM-R-NSGA-II converges much faster then R-NSGA-II, and for DTLZ1 the convergence for FPM-R-NSGA-II is faster, but the difference is much smaller. For DTLZ2 and 4 both algorithms behave similarly, with R-NSGA-II being more consistent for DTLZ2 while the variance for FPM-R-NSGA-II is higher. For the 5-objective cases, the results look similar to the 3-objective cases. On DTLZ1 and 3, FPM-R-NSGA-II converges faster than R-NSGA-II. However the difference between the algorithms is smaller, for DTLZ1 the algorithms have similar convergence rates. The variance for FPM-R-NSGA-II among the runs in the case of DTLZ2 is higher than before. For 10-objective cases, FPM-R-NSGA-II behaves similarly to R-NSGA-II for DTLZ1 and 2, while performing much worse in the cases of DTLZ3 and 4. FPM-R-NSGA-II even fails to converge close to the reference point in these cases.

The results suggest that FPM-R-NSGA-II can perform better than R-NSGA-II in some cases, especially on lower objective problems. While performing similar in others, and performing the worst in higher objective problems, even worse than the original R-NSGA-II.

## 5 CONCLUSIONS AND FUTURE WORK

Although online KDO has received some focus in the literature, the most popular method of online KDO is with the use of EDAs. In this paper, an online KDO approach that uses FPM to build a probability distribution which is sampled to replace the mutation operator from R-NSGA-II is introduced. The experimental results suggest that the new approach is able to converge faster than R-NSGA-II on some of the considered test problems, and behaves similarly on some others, while also performing poorly in some cases. However the study did not take into account how much computation time could have be saved by terminating the runs once convergence had been achieved. The relatively higher computational cost of FPM-R-NSGA-II is balanced by the fact that in addition to convergence to the reference point, the algorithm also generates rules that conform to preferred solutions. The three parameters that affect the proposed approach are, the minimum required significance and the value of $k$ for choosing $k$-nearest neighbours as the selected set from non-dominated solutions. The impact of these parameters on the performance of the algorithm will be considered in future studies. Understanding why FPM-R-NSGA-II behaves much worse than R-NSGA-II on some of the higher objective cases is also important for the future development of FPM based online KDO algorithms.

## REFERENCES

[1] Sunith Bandaru, Amos H.C. Ng, and Kalyanmoy Deb. 2017. Data mining methods for knowledge discovery in multi-objective optimization: Part A - Survey. *Expert Systems with Applications* 70 (2017), 139–159. https://doi.org/10.1016/j.eswa.2016.10.015

[2] Sunith Bandaru, Amos HC Ng, and Kalyanmoy Deb. 2017. Data mining methods for knowledge discovery in multi-objective optimization: Part B-New developments and applications. *Expert Systems with Applications* 70 (2017), 119–138.

[3] Sunith Bandaru and Henrik Smedberg. 2019. A parameterless performance metric for reference-point based multi-objective evolutionary algorithms. In *The Genetic and Evolutionary Computation Conference*. ACM. https://doi.org/10.1145/3321707.3321757

[4] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan. 2002. A fast and elitist multiobjective genetic algorithm: NSGA-II. *IEEE Transactions on Evolutionary Computation* 6, 2 (April 2002), 182–197. https://doi.org/10.1109/4235.996017

[5] Kalyanmoy Deb and Aravind Srinivasan. 2006. Innovization: Innovating design principles through optimization. In *Proceedings of the 8th annual conference on Genetic and evolutionary computation*. ACM, 1629–1636.

[6] Kalyanmoy Deb and J Sundar. 2006. Reference point based multi-objective optimization using evolutionary algorithms. In *Proceedings of the 8th annual conference on Genetic and evolutionary computation*. ACM, 635–642.

[7] Kalyanmoy Deb, Lothar Thiele, Marco Laumanns, and Eckart Zitzler. 2005. Scalable test problems for evolutionary multiobjective optimization. In *Evolutionary multiobjective optimization*. Springer, 105–145.

[8] Mark Hauschild and Martin Pelikan. 2011. An introduction and survey of estimation of distribution algorithms. *Swarm and Evolutionary Computation* 1, 3 (2011), 111–128.

[9] Ke Li, Kalyanmoy Deb, and Xin Yao. 2018. R-metric: Evaluating the performance of preference-based evolutionary multiobjective optimization using reference points. *IEEE Transactions on Evolutionary Computation* 22, 6 (2018), 821–835.

[10] Ryszard S Michalski. 2000. Learnable evolution model: Evolutionary processes guided by machine learning. *Machine learning* 38, 1-2 (2000), 9–40.