Set-SMAA for Finding Preferable Multi-Objective Solutions

Rotem Dror[1], Amir Kantor[2], Michael Masin[2], and Segev Shlomov[1]
[1]Technion – Israel Institute of Technology, [2]IBM Research – Haifa, Israel
{rtmdrr,segevs}@campus.technion.ac.il,{amirka,michaelm}@il.ibm.com

ABSTRACT

Multi-objective optimization problems involve more than one objective to be optimized simultaneously. Typically, however, there is *no* single solution that simultaneously optimizes them all. This can be overcome by calculating a *set* of compromise solutions, called the *Pareto frontier*. We hereby introduce a new approach, called **Set-SMAA**. It extends *stochastic multi-criteria acceptability analysis (SMAA)* to find a small set of *preferable solutions* from the frontier. This applies to diverse use cases, from decision-making and product recommendation, to *evolutionary multi-objective optimization algorithms* and their fitness measures.

CCS CONCEPTS

•Applied computing → Multi-criterion optimization and decision-making;

KEYWORDS

multi-objective optimization, decision-making, stochastic multi-criteria acceptability analysis, recommender systems, fitness evaluation

ACM Reference format:
Rotem Dror[1], Amir Kantor[2], Michael Masin[2], and Segev Shlomov[1]. 2017. Set-SMAA for Finding Preferable Multi-Objective Solutions. In *Proceedings of GECCO '17 Companion, Berlin, Germany, July 15–19, 2017,* 2 pages.
DOI: http://dx.doi.org/10.1145/3067695.3076005

1 INTRODUCTION

A **multi-objective (MO) decision problem** is given by a finite sequence $P_1, \ldots, P_n$ of vectors in $\mathbb{R}^m$. That is, there are $n \geq 0$ **solutions** (a.k.a., *options*, *alternatives*), and $m \geq 1$ **objectives** (a.k.a., *criteria*), and each solution is given a real-valued *score* in each of the objectives. W.l.o.g., we assume that in all objectives greater values are better. The ultimate problem in this context is finding the *best solution*. However, generally there is *no* solution that is best in all objectives. In order to pick the best solution, one needs additional information on the *preferences* of the *decision maker (DM)*.

A DM's preferences can be represented by a **utility function**. That is, a function $f : \mathbb{R}^m \to \mathbb{R}$ that is monotonically nondecreasing. For each solution P_k, $f(P_k)$ is a number that represents the *overall* value, or, *utility*, for the DM. Here also, greater values are better. Monotonicity reflects the fact that an improvement in some (or all) of the objectives can only improve the utility. Given a utility function f, it is easy to find the best solution(s). These are, of course, those that maximize f.

In many cases, prior knowledge about the DM is limited, let alone a precise representation of her preferences. At the basis of our approach lies the realization that even without precise information, if we take a **distribution** of utility functions for the problem at hand, we can systematically find a (usually, small) set S of solutions that are preferable—in the sense that *it is most likely that one of the solutions in S is best for the DM.*

The foundations of our approach lies in a probabilistic model similar to that appearing in *domain criterion* methods [4]. We use Monte Carlo simulations to make estimations, similarly to other *stochastic multi-criteria acceptability analysis (SMAA)* methods [5]. Our methods, collectively referred to as **Set-SMAA**, are essentially an extension of SMAA for choosing *sets* of solutions. Some of these are implemented in a decision-support tool by IBM [3].

2 SET-SMAA

Consider a probability space defined on a set $\mathcal{U}$ of utility functions, and assume a pre-given pseudorandom generator `G()` of utility functions in $\mathcal{U}$, in accordance to the distribution at hand.

Construction of coverage database. Fix $\varepsilon \geq 0$. We generate a large number N of utility functions using `G()`. For each generated function f_j, we find the set of solutions that **ε-cover** f_j. Meaning, those solutions that maximize f_j up to an additive constant ε. For each such solution k, we add j to a set U_k containing all the indexes of the utility functions that are ε-covered by solution k. The family of sets $\{U_k\}_{k=1}^n$ is called the **coverage database (CDB)**.

Ranking solution sets. Using the CDB, for each set of solutions S, we define a **coverage measure** by $v(S) = \left|\bigcup_{k \in S} U_k\right| / N$, indicating the fraction of utility functions that at least one solution in S covers. This can be used to *rank* any number of solutions and determine their total significance for the DM.

Finding preferable solutions. The essence of calculating a set of preferable solutions involves limiting the size of S. There are several ways of doing so, depending on the application at hand.

Max Cover: Maximizing the coverage measure while the number of solutions is constrained to be below a fixed threshold. This requires solving a *combinatorial optimization problem.* **Min Size:** A dual approach of minimizing the number of solutions while the coverage measure is constrained to be above a fixed threshold. While these approaches represent two extreme cases, one may devise other variants of the combinatorial problem.

Top-K: A naïve (as U_k are generally *not* disjoint) but fast heuristic: collecting (only) the top K solutions according to their coverage measure (i.e., according to the size of U_k).

Greedy Methods: A family of heuristics achieving most of the precision of the combinatorial approaches while still being relatively efficient. Using the CDB, we iteratively collect the solution that adds most to the set of utility functions that are already covered.

GECCO '17 Companion, Berlin, Germany
 ACM ISBN 978-1-4503-4939-0/17/07.
DOI: http://dx.doi.org/10.1145/3067695.3076005

Various termination conditions can be imposed on the algorithm: limiting the number of iterations of the algorithm, the total cover reaches a certain threshold, or when the marginal contribution of the next solution to the total cover is below a certain threshold.

The above formulations apply to any distribution of utility functions. This provides a powerful framework that supports a wide range of use cases in which there is some domain-specific or DM-specific knowledge about preferences of DMs (e.g., customer segmentation). Nevertheless, we believe that these methods are just as beneficial when such information is unavailable.

In case there is no such information, we pick a simple class $\mathcal{U}$ of linear utility functions of the form $f = \sum_{i=1}^{m} w_i x_i$, or variants such as $f = \sum_{i=1}^{m} w_i \, x_i^{1/\alpha}$, for a fixed $\alpha \geq 1$. Here, each utility function f is defined by a weight vector $w = (w_i)_{i=1}^{m}$ on the *standard simplex* in the m-dimensional space. As there is no indication of any preferred region on the simplex, we take the *uniform distribution* on the simplex. W.l.o.g., we assume that the values $(x_i)_{i=1}^{m}$ of the objectives are in the range of $[0, 1]$. Note, one may generate functions using a *uniform grid* on the simplex instead of taking random samples.

3 EXPERIMENTS AND RESULTS

We empirically tested three research questions: (1) the coverage of **Set-SMAA** methods in practical datasets; (2) the runtime efficiency of our algorithms as a function of their parameters; and (3) a direct quality evaluation by Set-SMAA users.

In order to address the first two questions (see Test 1 and 2) we use two original datasets. **House** dataset, which is taken from the U.S. Department of Housing and Urban Development [2], includes the characterization of dozens of houses according to 3 criteria: year built, square footage, and price. Its frontier contains 348 solutions. The university dataset **Uni**, which is taken from CWUR 2015 World University Rankings [1], uses 9 objectives to rank the world's top-1000 universities. We eliminated the first 198 universities to obtain a large frontier containing 802 solutions. Two additional datasets, **House_double** (6 objectives) and **Uni_double** (18 objectives), were created by doubling the criteria in the first two datasets: for each objective i (with value x_i), we added an objective (say, i') valued $\sqrt{x_i}$. Another dataset, **Uni_half** (5 objectives), was created by eliminating part of the objectives of **Uni**.

Test 1: Coverage. For each of the datasets, we applied different Set-SMAA methods for finding preferable solutions (**Max Cover**, **Greedy**, **Top-K**), for different values of the parameter ε (0, 0.01, 0.02, and 0.05); ε designates some level of "impreciseness", or insensitivity to exact utility scores. The results were obtained using $N = 10^4$ linear utility functions generated with uniform distribution on the simplex. In each case, we measured coverage of the utility space per number of returned solutions (which is another independent variable). The reported coverage is measured using a uniform grid of $2 \cdot 10^4$ to $3 \cdot 10^4$ points on the simplex, since it provides a common ground for comparison without dependence on sampling.

In datasets with up to 6 objectives, 5 solutions cover over 80% and 9 solutions cover over 95% of the utility space even with $\varepsilon = 0$. For more objectives, the numbers are from 14 solutions (**Uni** with 80% coverage) to 30 solutions (**Uni_double** with 95% coverage). Reducing sensitivity by as little as $\varepsilon = 0.01$, enables the same coverage with up to 30% less solutions, while $\varepsilon = 0.05$ enables to reduce the number of solutions by up to 60%. In our experiments we obtained that the number of returned solutions generally increases linearly with the number of objectives, in order to reach the same coverage. However, we also know that it largely depends on the structure of the frontier.

The **Greedy** method statistically dominates **Top-K** heuristic throughout the experiments. The difference in coverage is significant only for a relatively small range of chosen solutions (3 to 7 solutions), which is important, however, for decision-making applications. There, the difference can approach 15% when $\varepsilon \neq 0$. The difference in coverage between the optimal **Max Cover** and **Greedy** can seldom reach a few percent for this range of solutions.

Test 2: Efficiency. We evaluated execution time and coverage as a function of the number of generated utility functions (from 500 to 10^4 samples). Apart from **Max Cover**, where combinatorial optimization is significant and time-consuming, execution time is roughly linear with the sample size. **Greedy** with 10^4 samples takes a few seconds to complete, while taking 500 samples is about 20 times faster (100msec on Intel Core i7-4710MQ with 16GB RAM). **Top-K** is faster by about 30%. The most time-consuming part is the construction of the CDB. Coverage, in contrast, is generally robust with sample size. Even a small number of sampled functions bring a high level of coverage. This suggests using Set-SMAA, and especially the **Greedy** or **Top-K** methods, as part of *evolutionary MO optimization algorithms*.

Test 3: User studies. A few user studies conducted at IBM provide encouraging evidence for the capabilities of our methods in locating the stronger solutions. First, users were generally satisfied with the algorithms' performance: when tested on several real-life datasets, the solution sets selected by Set-SMAA **Greedy** method received an average satisfaction score slightly above 4, on a scale of 1 (worst) to 5 (best). Users also pointed out that these were decent *baseline* solutions, with which other solutions on the frontier may be compared, and which facilitate the exploration of the alternatives. The studies also suggest that taking $\alpha = 2$ is generally preferable to $\alpha = 1$, i.e., users tend to favor more balanced solutions.

4 CONCLUSION

We presented a set of new methods, collectively called **Set-SMAA**, for reducing the size of the Pareto frontier by extracting preferable solutions that retain most of its utility and competence. This approach does *not* require prior knowledge of DMs' preferences, but if any statistical (i.e., aggregative) or concrete information exists, it could be easily incorporated in our algorithms. With a few datasets, we demonstrated that a small number of high-quality solutions were generally enough to capture most of the utility space.

REFERENCES

[1] 2015. CWUR 2015 World University Rankings. http://cwur.org/2015.php. (2015).
[2] 2016. U.S. Department of Housing and Urban Development. https://www.hudhomestore.com/Home/Index.aspx. (2016).
[3] 2017. IBM Watson Tradeoff Analytics Service. https://www.ibm.com/watson/developercloud/tradeoff-analytics.html. (2017).
[4] JR Charnetski and RM Soland. 1978. Multiple-attribute decision making with partial information: The comparative hypervolume criterion. *Naval Research Logistics (NRL)* 25, 2 (1978), 279–288.
[5] T Tervonen and JR Figueira. 2008. A survey on stochastic multicriteria acceptability analysis methods. *J. of Multi-Criteria Decision Analysis* 15 (2008), 1–14.