THE MULTI-OBJECTIVE DISTINCT CANDIDATES OPTIMIZATION APPROACH

Rasmus K. Ursem

Grundfos Management A/S, Poul Due Jensens Vej 7, DK-8850 Bjerringbro, Denmark

ursem@cs.au.dk

Peter Dueholm Justesen

Aarhus University

Department of Computer Science, Aabogade 34, DK-8200 Aarhus N, Denmark

juste@cs.au.dk

Abstract Traditional multi-objective optimization algorithms typically return several hundred non-dominated candidate solutions. From practical point-of-view, a small set of 5-10 distinct candidates is often preferred because post-processing of many solutions may be too costly, too time-consuming, too difficult to compare design differences, or similar solutions turn out to be statistically equal in prototyping and manufacturing. Interestingly, these limitations apply to most real-world problems.

In this paper, we introduce Multi-objective Distinct Candidates Optimization (MODCO) as an approach to find a user-defined number of clearly different solutions wrt. performance and design. In MODCO, we distinguish between generalized and domain-specific preferences, where generalized preferences address the aforementioned limitations and the domain-specific preferences cover the DM's whishes if available.

Keywords: Multi-objective optimization, distinct candidates, diversity management

1. Introduction

Successful application of multi-objective optimization to a real-world problem typically consists of two steps. First, the optimization step where the problem is set up, the chosen algorithm is executed, and all non-dominated solutions are gathered. Second, the decision making step where the single solution to implement is chosen among the non-dominated solutions found in step 1, see figure 1 and Deb [1, pp. 5].

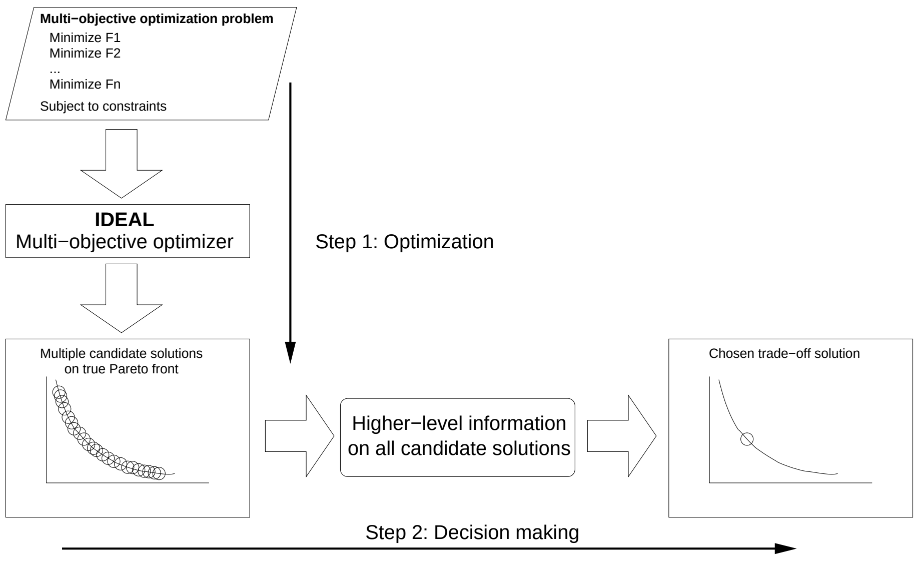
Multi−objective optimization problem
Minimize F1
Minimize F2
...
Minimize Fn
Subject to constraints
IDEAL
Multi−objective optimizer
Step 1: Optimization
Multiple candidate solutions on true Pareto front
Higher−level information on all candidate solutions
Chosen trade−off solution
Step 2: Decision making

Figure 1. Process for application of multi-objective optimization.

In this process, the decision maker (DM) has to apply his preferences among the objectives to select the final solution. Veldhuizen and Lamont categorize the point in the process where the DM apply his preferences into three categories; 1) *a priori* – before the optimization is initiated, 2) *progressive* – during the optimization, and 3) *a posteriori* – after the optimization is finished [2]. Algorithms in category 1 typically transform the multi-objective problem into a single objective by specifying an utility function combining the multiple objectives. The weighted sum approach is the most widely known algorithm in this category. The progressive algorithms in category 2 usually incorporate the DM's preferences in form of decision support systems, see [3] for a survey. Finally, category 3 algorithms exclude the DM's preferences from the search. Instead, they typically produce a large set of Pareto-optimal solutions for the DM to choose from in step 2.

The drawback of approach 1 and 2 is that the DM has to make a choice regarding the importance of the involved objectives *prior* to the actual optimization, which may be difficult before the DM has seen any solutions. In addition, such choices are highly domain-specific and problem dependent, and algorithms are thus hard to generalize for a broader range of applications. In contrast, the traditional MO algorithms in category 3 are generally applicable. However, these algorithms produce hundreds or thousands of solutions and the notion is to leave it to the DM to gather the "higher-level information" in step 2 on this set and choose the actual solution to implement. The often large set returned by *a posteriori* algorithms pose a serious problem because it may be *impossible* to gather "higher-level information" on such a large set. In short, time, money, and other reasons may prevent the application of the higher-level information gathering methods (further simulation, prototype construction, testing, etc.) on a set of more than 5-10 solutions. Consequently, we consider the current algorithms as either too domain-specific (category 1 or 2) or too general (category 3), because a huge set of candidate solutions is returned. Naturally, pruning the set using the DM's preferences is the obvious remedy for this drawback. However, this approach pose another problem because it may be difficult for the DM to state his preferences as explicit decision making rules. In our view, selection from a huge set or pruning the set tend to make the DM focus on the performance (objective space) and neglect the design differences (search space). In contrast, only investigating a few solutions promotes a better balance between performance and design. Furthermore, the low number of solutions allows the DM to apply preferences, use decision rules, and evaluate objectives not stated explicitly.

The MODCO approach address these challenges by incorporating *generalized preferences* into the algorithm with the goal of finding a small set of 5-10 distinct candidates to make step 2 manageable *without* stating explicit preference relations. In MODCO, the concept *generalized preferences* covers a *priori* considerations that are relevant to most if not all real-world applications. This analysis answers the following questions (further elaborated in section 2 and 3):

1 **Number of candidates:** $K_{NC} \in [1 : \infty] \subseteq \mathbb{N}$
How many candidates is it practically and economically feasible to inspect, analyze, and compare in post-processing?

2 **Performance distinctiveness:** $K_{PD} \in [0.0 : 1.0] \subset \mathbb{R}$
How different should the candidates be performance-wise?

3 **Design distinctiveness:** $K_{DD} \in [0.0 : 1.0] \subset \mathbb{R}$
How different should the candidates be design-wise?

4 **Simulator accuracy:** $K_{SA} \in [0.0 : 1.0]^M \subset \mathbb{R}^M$
What is the accuracy of the involved simulators?

In MODCO, the parameters K_{NC}, K_{PD}, K_{DD}, and K_{SA} constitute the generalized preferences, and they may be implemented as the secondary selection criterion in an algorithm. Hence, MODCO algorithms aim at reducing the DM's task in step 2 by dividing the higher-level information into two groups, *generalized preferences* as an a priori analysis to step

1 and the *domain-specific information gathering* as a precursor to the decision making in step 2. In this, the domain-specific information gathering includes further investigations such as visual inspection, detailed simulation, and prototype testing on the distinct candidates followed by evaluation of the DM's implicit or explicit preferences regarding the objectives. Thus, a MODCO algorithm combines category 1 and 3 by integrating the generalized preferences a *priori* and leaving the domain-specific part to a *manageable* second step a *posteriori*.

The paper is structured as follows. Section 2 provides the motivation for the MODCO approach by summarizing 5 years of observations from real-world industrial MO problems. Section 3 lists the features of the ideal MODCO algorithm and the goals of MODCO. After having introduced the MODCO ideas, we provide a survey of related research in section 4. Finally, section 5 concludes the paper.

2. Motivation for MODCO algorithms

The application of multiobjective optimization in an industrial context raise a number of interesting challenges, dilemmas, and unforseen obstacles. The following observations are gathered from more than 6 years of work at Grundfos R&T solving more than 30 multiobjective optimization tasks in very diverse engineering disciplines including fluid mechanics, motor design, motor control, structural mechanics, electronics, robust design optimization, value-chain optimization, production optimization, etc. Discussions with the DMs involved in these projects have lead to a number of considerations that form the basis for the MODCO approach. In short, the arguments for incorporaing generalized preferences into the optimization algorithm fall in three categories; 1) post-processing many Pareto-optimal solutions, 2) physical realization of a solution, and 3) decision making among large sets of solutions. These categories are discussed and elaborated in the following sections.

2.1 Post-processing many Pareto-optimal solutions

In an industrial application context, the multiobjective optimization is often only a small step in a large design process. Hence, the solutions found by an MO algorithm need to be further investigated to verify each solution against results from other parts of the design process. Consequently, post-processing a large number of solutions is typically infeasible for the following reasons: i) It is too expensive, ii) it is too time-consuming, and iii) the optimization problem may only cover a part of the full design.

Regarding economy, it is often rather expensive to construct a physical prototype of a simulated design. For instance, a 3D prototype print and performance test of a single pump housing may cost up to 5000 Euro. Thus, investigating 100 candidates is out of the question.

Concerning time, a large set of candidate solutions is also problematic if it takes a lot of time to post-process a single solution. As mentioned above, the optimization is often only a small step in a larger design process and further extended simulation may be needed to verify the design in greater detail. For example, conducting a full-range CFD simulation on a design may take several days or weeks, which makes post-processing of large sets impossible. Furthermore, such an investigation often only constitutes a part of the simulations carried out on a design. Other types of analyses include stress analysis, cost calculation, robust design investigations, etc.

In addition to economy and time, the optimization may only be addressing a part of the total design. For example, the optimization problem may represent a sub-circuit of a pump controller circuit. Thus, additional unmodeled features may have to be handled in post-processing. The motivation for only looking at a partial problem are numerous. Some reasons include: i) No sufficiently accurate model exists, e.g., electro-magnetic noise in relation to circuits, ii) impossible to build a formal model of the consequences of a design, iii) misguiding of the search by problematic pairing of a high-accurate model for some objectives with a low-accurate model for other objectives, and iv) a desire to run the development project in parallel sub-teams, e.g., design pump hydraulics and electromotor at the same time.

2.2 Physical realization of a single solution

In an industrial design process, the simulation of a solution is usually followed by a prototype testing and then finally creation of machining tools for mass production. These three steps: i) simulation, ii) prototype testing, and iii) mass production each pose some challenges that make multiobjective optimization in its traditional form less favorable.

Regarding simulation, finding a large Pareto-optimal set is not always meaningful. Two nearby solutions may differ by e.g. 0.1% in performance, but the simulator may have an expected inaccuracy in the range of 1%-5% when the simulated solution is compared with a physical test of the design. Hence, it does not make sense to report a lot of performance-wise similar solutions since prototype testing of them may prove them to be statistically equal.

In prototype testing, one problem is to make a 100% accurate representation of the simulated design. All prototyping methods have tolerances. Consequently, there will be small differences between the simulated design and a corresponding prototype. In addition to this, no testing equipment is 100% accurate. Thus, repeating test will not give the exact same result.

Finally, preparing the design for mass production also renders some challenges. In simulation and prototyping, the “design changes” were caused by inaccuracies in simulator, prototype construction, and measurement. However, realization of the design may require small changes in the design, e.g., to make the part castable with a bi-directional casting process. Furthermore, mass produced parts also are not completely identical since all production processes introduce small variations and it is very costly to approach a 0% tolerance. Furthermore, the process itself may introduce some random variation such as size of welding seams which is not modeled in the simulation. Hence, the produced parts differ from the simulated even if no deliberate modifications have been made to adapt the design to production.

2.3 Decision making among large sets of solutions

Challenges related to decision making often come as either i) an abstract or general statement from a development project such as “we would like to evaluate the pros and cons of 3-5 clearly different designs”, ii) local selection among similar Pareto-optimal solutions, or iii) as the DM’s difficulties in interpreting the results and stating explicit preference rules.

General statements from development projects are often seen in concept studies where it is important to investigate and compare significantly different designs instead of fine-tuning the performance of a design. Hence, the DM is more interested in a small set of clearly different designs rather than a large set covering a smooth transition.

Concerning the local selection among Pareto-optimal solutions, DMs tend to favor solutions residing in the so-called knee regions of the Pareto front if no domain-specific preferences can be determined a priori. Hence, solutions not in knee regions can be omitted from the set of returned solutions as the DM is typically not willing to accept a large decrease in one objective to gain a small increase in another.

Regarding the DM’s abilities, it is often the case that such persons have a non-optimization background and in some cases even a non-technical background. Thus, the DM may not have the technical skills to make the right decision among a set of 500-1000 solutions representing a smooth transition from one design type to another. Presenting a low number of 5-10 clearly different solutions is more feasible because it allows the DM to inspect each solution in great detail and make a decision without explictly stating preference relations.

3. Features of the ideal MODCO algorithm

The observations presented in the previous section can be condensed into a number of desirable features for the ideal MODCO algorithm:

1 **Return Pareto-optimal solutions.**
This is identical to goal 1 of traditional MO algorithms.

2 **User-defined maximal number of returned solutions.**
Resources such as money and time sets a firm limit on the number of solutions manageable in decision making. Any MODCO algorithm should have an upper limit on number of returned solutions.

3 **User-defined distinctiveness of the returned solutions.**
The application often state if the goal is to perform an exploratory search, to return a set of fine-tuned similar alternative solutions, or something in between. Furthermore, the DM or domain expert typically knows if the goal is to find clearly different designs (diversity in search space), performance-wise different solutions (diversity in objective space), or a combination of these two.

4 **User-defined accuracy of simulators.**
The accuracy of the simulators plays a key role in the later decision making step. The MODCO algorithm should incorporate any known simulator accuracies to ensure that the differences in simulated performance also shows in later post-processing. More precisely, if A dominates B in optimization, then this should hold for subsequent steps (prototyping, test, etc.) in the design process.

5 **Return solutions in knee regions or according to user-defined preferences.**
DMs are rarely (if ever) interested in solutions not located in knee regions of the Pareto front unless domain-specific preferences can be defined. Hence, the ideal MODCO algorithm should return solutions located in knee regions or alternatively solutions that comply with the user-defined preferences.

Feature 1 is obviously required by any multi-objective algorithm. Features 2-4 constitute general preferences that can be analyzed and defined *a priori* to the optimization. Feature 5 is motivated by observations on the decision making process in more than 30 multi-objective optimization tasks carried out at Grundfos R&T. Please note that feature 3 *does not* express the DM's preferences regarding the objectives importance, but only *how* diverse the returned solutions should be.

Regarding features 2-4, we define the user-variables for setting these requirements on the returned set. Table 1 lists the four variables supporting the above mentioned features. Naturally, it is up to the MODCO algorithm to transform the given value to an actual *approach* for implementing the feature. The parameters K_{DD} and K_{PD} for defining

Table 1. The list of user-variables supporting feature 2-4.

Feat.	Description	Variable
2	Max no. of distinct candidates	$K_{NC} \in [1 : \infty) \subseteq \mathbb{N}$
3	Design distinctiveness	$K_{DD} \in [0.0 : 1.0] \subset \mathbb{R}$
3	Performance distinctiveness	$K_{PD} \in [0.0 : 1.0] \subset \mathbb{R}$
4	Simulator accuracy	$K_{SA} \in [0.0 : 1.0]^M \subset \mathbb{R}^M$

distinctiveness should be interpreted as shown in figure 2. As seen, a value of 0.0 indicates a user-preference for a low distinctiveness, which corresponds to a preference for very similar solutions. A value of 1.0 corresponds to high distinctiveness, i.e., very different solutions. An intermediate value of, e.g., 0.5 represents a desire for a medium level of distinctiveness. In contrast to methods with explicit formulation of preferences, this approach allows incorporation of rather vauge statements from the DM or domain expert. For example, a domain expert may say *"For this problem, I know that many somewhat different solutions have roughly the same performance."* In MODCO, such a statement can be transformed into $K_{PD} = 0.0$ meaning "roughly same performance" and $K_{DD} = 1.0$ or perhaps $K_{DD} = 0.5$ representing a desire for highly or somewhat different solutions.

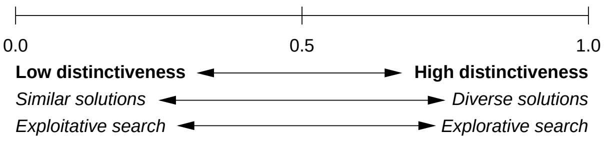

Figure 2. Distinctiveness range for parameters K_{DD} and K_{PD}.

Regarding simulator accuracy, the vector K_{SA} contains a value for each objective where 1.0 corresponds to a 100% accurate simulator and 0.0 means the returned result is no better than a random value.

3.1 Three goals of MODCO algorithms

The previous sections containing motivation and MODCO features form a basis for reformulating the three goals of traditional MO algorithms, closeness, distribution and spread, to goals for MODCO algorithms. In short, we suggest the following three goals:

1 **Closeness** – the distinct candidates should be on the true Pareto front (or as close as possible).

2 **Global distinctiveness** – the candidate set should have a user-defined max size K_{NC} and contain clearly distinct solutions found in accordance with K_{DD}, K_{PD}, and K_{SA}.

3 **Local multi-objective optimality** – each candidate should be of primary interest to the DM, e.g., in a knee region if no explicit preferences are given.

Goal 1 of MODCO algorithms is to find solutions on the true Pareto front. This goal represent feature 1 and is identical to goal 1 in MO.

Goal 2 represents the DM's desire for a set of clearly distinct solutions with a user-defined maximal number of solutions in the set. The *a priori* analysis of feature 2-4 can be summarized in one goal stating how many and how different the returned solutions should be.

Goal 3 expresses the observation mentioned in section 2.3 regarding the DM's preference for solutions located in knee regions of the Pareto front or alternatively in accordance with the DM's explicit preferences. Thus, goal 3 represents feature 5 of the ideal MODCO algorithm.

4. Survey of MODCO-related algorithms

The MODCO approach was first studied in our earlier paper [9]. The paper contains a brief introduction to the approach, the Cluster-Forming Differential Evolution (CFDE), and a detailed study on performance and effects of changing the MODCO parameters K_{PD} and K_{DD}. A real-world study on mechanical and electrical engineering problems have been submitted to BIOMA 2010 [10].

The five features of the ideal MODCO algorithm have received some attention by other researchers.

Feature 1, closeness to true Pareto front, is undoubtedly the most investigated since it is one of the goals of traditional MO research, for surveys see [11, 12, 13].

Feature 2 about returning a limited set of distinct solutions has been investigated in the approach known as "modeling to generate alternatives (MGA)" suggested by Brill [4] and later applied in several studies

[5, 6, 7, 8]. In short, the main purpose in MGA is to present a set of maximally different solutions to allow the DM to evaluate non-modeled objectives and implicit preferences and decision rules. The idea in MGA is to first find a starting solution, then define a maximal allowed drop in fitness(es), and search one-by-one for other solutions that are maximally different in search space from previously found solutions while fulfilling the constraint of having the fitness(es) above the defined relaxation target. Although applicable for generating alternatives, the approach excludes the prime paradigm of MO – to find trade-offs wrt. the objectives. In the MGA approach, a solution just has to lie in the hypercube spanned by the relaxation thresholds. Thus, finding the maximally different solutions to accommodate the *unmodeled* objectives may result in suboptimal solutions wrt. the *modeled* objectives. Feature 2 has also been investigated in studies incorporating the DM's preferences into the algorithm, for a survey see [3]. To the authors knowledge, no paper suggests to integrate *general* preferences into the multi-objective algorithm with the goal of returning a user-defined maximal number of candidate solutions. However, several authors suggest to modify a traditional MO algorithm with various kinds of subpopulation schemes to increase the search performance or parallelize the MOEA, for example [14, 15]. Regarding on-the-fly clustering of solutions, Koch and Zell suggest the MOCS algorithm [16]. MOCS cluster the solutions in objective space with the aim of diversifying the search.

Feature 3 regarding setting a user-defined design and performance distinctiveness is to the authors knowledge also not investigated. Generally, traditional MO research focus at obtaining an even population distribution, but not distinctiveness among a small set of candidates.

Feature 4 on simulator accuracy has, to some extent, been investigated in a related form known as robust design optimization. It should be noted that this research primarily deals with the manufacturing step of the design process mentioned in section 2.2, i.e., limiting the problems originating from the tolerances in mass production. Several papers have been published on robust design and some methods incorporate the link back to the design variables, e.g., Li's work on robust optimization [17].

Feature 5 about finding solutions in knee regions has been investigated by a couple of authors in the MO research community. The most interesting study is presented by Branke et al. [18] who define two metrics (angle and a utility function) for detecting knee regions and favor solutions in knee regions. Another study is published by Rachmawati and Srinivasan [19] who suggest to use a weighted sum similar to the utility function of Branke et al. [18]. Regarding incorporation of user-preferences, numerous studies have been performed, see [3].

5. Conclusions

In this paper, we have introduced the MODCO approach, which is motivated by a number of observations from real-world problems from the manufacturing industry. The goal in MODCO is to return a small set of clearly distinct candidate solutions to allow a feasible in-depth post-processing of the candidates. The main idea in MODCO is to refine the multiobjective search by incorporating general preferences from the DM and the domain expert's knowledge of the problem at hand. Through a few parameters, rather vague statements from the DM or domain expert is trainsformed into desired characteristics of the returned set of solutions. In addition to this, the MODCO approach allows incorporation of any known simulator inaccuracies to further refine the search.

The MODCO approach may be seen as an extension or modification of the traditional MO concept. Thus, a reformulation of the three goals closeness, distribution, and spread of traditional MO is also in place. The three MODCO goals are closeness, global distinctiveness, and local multiobjective optimality. The closeness goal states a desire to find solutions on the Pareto front, which is identical to the first goal of traditional MO. The global distinctiveness goal states that the returned set of candidate solutions should comply with the user's general preferences on number of solutions, performance distinctiveness, design distinctiveness, and known simulator accuracy. Finally, the local multiobjective optimality goal express the DM's preferences for finding solutions in knee regions or according to domain-specific preferences.

References

[1] K. Deb, *Multi-Objective Optimization using Evolutionary Algorithms.* Chichester, UK: John Wiley & Sons, 2001, iSBN 0-471-87339-X.

[2] D. A. V. Veldhuizen and G. B. Lamont, "Multiobjective Evolutionary Algorithms: Analyzing the State-of-the-Art," *Evolutionary Computation*, vol. 8, no. 2, pp. 125–147, 2000.

[3] C. A. C. Coello, "Handling Preferences in Evolutionary Multiobjective Optimization: A Survey," in *2000 Congress on Evolutionary Computation*, vol. 1., 2000, pp. 30–37.

[4] E. D. Brill Jr., "The use of optimization models in public-sector planning," *Management Science*, vol. 25, no. 5, pp. 413–422, 1979.

[5] E. D. Brill Jr., J. M. Flach, L. D. Hopkins, and S. Ranjithan, "Mga: A decision support system for complex, incompletely defined problems," *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 20, no. 4, pp. 745–757, 1990.

[6] E. D. Brill Jr., S.-Y. Chang, and L. D. Hopkins, "Modeling to generate alternatives: The hsj approach and an illustration using a problem in land use planning," *Management Science*, vol. 28, no. 3, pp. 221–235, 1982.

[7] D. H. Loughlin, S. R. Ranjithan, E. D. Brill Jr., and J. W. Baugh Jr., "Genetic Algorithm Approaches for Addressing Unmodeled Objectives in Optimization Problems," *Engineering Optimization*, vol. 33, pp. 549–569, 2001.

[8] E. Zechman and S. R. Ranjithan, "An evolutionary algorithm to generate alternatives (eaga) for engineering optimization problems," *Engineering Optimization*, vol. 36, no. 5, pp. 539–553, 2004.

[9] P. D. Justesen and R. K. Ursem, "Multiobjective distinct candidates optimization (modco): A cluster-forming differential evolution algorithm," in *Proceedings of the fifth international conference on Evolutionary Multi-Criterion Optimization (EMO-2009)*, vol. 1, 2009, pp. 525–539.

[10] P. D. Justesen and R. K. Ursem, "Preference-based Multi-objective distinct candidates optimization" submitted to *BIOMA 2010*, temporarily available at `www.daimi.au.dk/~ursem/publications/Justesen_BIOMA2010.pdf`.

[11] C. A. Coello Coello, "A Comprehensive Survey of Evolutionary-Based Multiobjective Optimization Techniques," *Knowledge and Information Systems. An International Journal*, vol. 1, no. 3, pp. 269–308, 1999.

[12] C. A. Coello Coello, "An Updated Survey of GA-Based Multiobjective Optimization Techniques," *ACM Computing Surveys*, vol. 32, no. 2, pp. 109–143, 2000.

[13] M. Reyes-Sierra and C. A. Coello Coello, "Multi-Objective Particle Swarm Optimizers: A Survey of the State-of-the-Art," *International Journal of Computational Intelligence Research*, vol. 2, no. 3, pp. 287–308, 2006.

[14] J. Branke, H. Schmeck, K. Deb, and R. Maheshwar, "Parallelizing multi-objective evolutionary algorithms: Cone separation," in *Proceedings of the 2004 IEEE Congress on Evolutionary Computation*, pp. 1952–1957.

[15] A. Molyneaux, G. Leyland, and D.Favrat, "A New, Clustering Evolutionary Multi-Objective Optimisation Technique," in *Proceedings of the Third International Symposium on Adaptive Systems—Evolutionary Computation and Probabilistic Graphical Models*, 2001, pp. 41–47.

[16] T. E. Koch and A. Zell, "MOCS: Multi-objective clustering selection evolutionary algorithm," in *GECCO 2002: Proceedings of the Genetic and Evolutionary Computation Conference*, 2002, pp. 423–430.

[17] M. Li, "Robust optimization and sensitivity analysis with multi-objective genetic algorithms: Single- and multi-disciplinary applications," Ph.D. dissertation, University of Maryland, 2007.

[18] J. Branke, K. Deb, H. Dierolf, and M. Osswald, "Finding knees in multi-objective optimization," in *Parallel Problem Solving from Nature - PPSN VIII*, 2004, pp. 720–729.

[19] L. Rachmawati and D. Srinivasan, "A multi-objective evolutionary algorithm with weighted-sum niching for convergence on knee regions," in *GECCO 2006: Proceedings of the 8th annual conference on Genetic and evolutionary computation*, 2006, pp. 749–750.