Interactive Multiobjective Evolutionary Algorithms

Andrzej Jaszkiewicz\(^1\) and Jürgen Branke\(^2\)

\(^1\) Poznan University of Technology, Institute of Computing Science
jaszkiewicz@cs.put.poznan.pl
\(^2\) Institute AIFB, University of Karlsruhe, 76128 Karlsruhe, Germany
branke@aifb.uni-karlsruhe.de

Abstract. This chapter describes various approaches to the use of evolutionary algorithms and other metaheuristics in interactive multiobjective optimization. We distinguish the traditional approach to interactive analysis with the use of single objective metaheuristics, the semi-a posteriori approach with interactive selection from a set of solutions generated by a multiobjective metaheuristic, and specialized interactive multiobjective metaheuristics in which the DM’s preferences are interactively expressed during the run of the method. We analyze properties of each of the approaches and give examples from the literature.

7.1 Introduction

As already discussed in Chapters 1 and 6, in order to find the best compromise solution of a multiobjective optimization (MOO) problem, or a good approximation of it, MOO methods need to elicit some information about the DM’s preferences. Thus, MOO methods may be classified with respect to the time of collecting the preference information as methods with either a priori, a posteriori, or progressive (interactive) articulation of preferences (Hwang et al., 1980; Słowiński, 1984). While the previous Chapter 6 discussed the use of (partial) a priori preference information in evolutionary MOO, here we will focus on interactive approaches. Note, however, that the two issues are closely related, as methods working with (partial) a priori information can be turned into interactive methods simply by allowing the DM to adjust preferences and re-start or continue the optimization interactively.

In general, interactive methods have the following main advantages:

- The preference information requested from the DM is usually much simpler than the preference information required by a priori methods.
They have moderate computational requirements in comparison to a posteriori methods.

As the DM controls the search process, he/she gets more involved in the process, learns about potential alternatives, and is more confident about the final choice.

Evolutionary algorithms and other metaheuristics may be used for interactive MOO in several ways. Traditional interactive methods usually assume that an underlying single objective, exact solver is available. This solver is used to solve a series of substitute single objective problems, whose solutions are guaranteed to be (weakly) Pareto optimal solutions. However, for many hard MOO problems, e.g., nonlinear or NP-hard combinatorial problems, no such efficient solvers are available. For such problems, one may use a single objective metaheuristic in place of an exact solver. This straightforward adaptation of the classical approach we will call the traditional approach to interactive analysis with the use of single objective metaheuristics.

In recent years, many multiobjective metaheuristics have been developed (Coello et al., 2002; Deb, 2001). Most multiobjective metaheuristics aim at generating simultaneously, in a single run, a set of solutions being a good approximation of the whole Pareto optimal set. This set of solutions may then be presented to the DM to allow him/her choose the best compromise solution a posteriori. However, if the set of generated solutions and/or the number of objectives is large, the DM may be unable to perceive the whole set and to select the best solution without further support characteristic to interactive methods. Thus, the approach in which the DM interactively analyzes a large set of solutions generated by a multiobjective metaheuristic will be called the semi-a posteriori approach.

Furthermore, the DM may also interact with a metaheuristic during its run. Such interaction with evolutionary algorithms has been proposed for problems with (partially) subjective functions, and is known under the name of interactive evolution (Takagi, 2001). Finally, if at least the underlying objectives are known, we have interactive multiobjective metaheuristics, where the DM interacts with the method during its run, allowing it to focus on the desired region of the Pareto optimal set.

This chapter is organized as follows. In the next section, the traditional approach to interactive analysis with the use of single objective metaheuristics is analyzed. The semi-a posteriori approach is discussed in Section 7.3. Section 7.4 makes a short excursion to efficiency measures. Then, Section 7.5 contains the discussion of the interactive multiobjective metaheuristics. Future trends and research directions are discussed in the final section.
7.2 Traditional Approach to Interactive Analysis with the Use of Single Objective Metaheuristics

As was mentioned above, the traditional interactive methods rely on the use of an exact single objective solver (e.g., Miettinen, 1999, and also Chapter 2). In general, the single objective solver is used to solve a substitute single objective problem whose optimal solution is guaranteed to be (weakly) Pareto optimal. For example, in the simplest version of the reference point method (c.f., Chapter 1), in each iteration, the DM specifies a reference point in the objective space. The reference point defines an achievement scalarizing function that is optimized as substitute single objective problem with the use of an exact single objective solver. The single solution generated in this way is presented to the DM, and he/she decides whether this solution is satisfactory. If the DM is not satisfied, he/she adjusts the reference point and the process is repeated.

Typical examples of achievement scalarizing functions are the \textit{weighted linear scalarizing function} defined as:

\[
s_j(x, \Lambda) = \sum_{j=1}^{J} \lambda_j f_j(x),
\]

and the \textit{weighted Tchebycheff scalarizing function} defined as:

\[
s_{\infty}(x, z^0, \Lambda) = \max_j \{\lambda_j (f_j(x) - z^0_j)\}
\]

where \(z^0\) is a \textit{reference point}, and \(\Lambda = [\lambda_1, \ldots, \lambda_J]\) is a weight vector such that \(\lambda_j \geq 0 \ \forall j\).

A well-established theory is associated with the generation of Pareto-optima through solving substitute single objective problems. For example, it may be proved that each weighted linear scalarizing function has at least one global optimum (minimum) belonging to the set of Pareto-optimal solutions (Steuer, 1986). A Pareto-optimal solution that is a global optimum of a weighted linear scalarizing function is called \textit{supported Pareto-optimal solution} (Ulungu and Teghem, 1994). Furthermore, each weighted Tchebycheff scalarizing function has at least one global optimum (minimum) belonging to the set of Pareto-optimal solutions. For each Pareto-optimal solution \(x\) there exists a weighted Tchebycheff scalarizing function \(s\) such that \(x\) is a global optimum (minimum) of \(s\) (Steuer, 1986, ch. 14.8). If, for a given problem, no efficient exact solver exists, a straightforward approach could be to use a single objective metaheuristic instead. One should be aware, however, that the mentioned above theoretical results are valid in the case of exact solvers only. If a metaheuristic solver is used several potential problems appear:

- Metaheuristics do not guarantee finding a global optimum. Thus, when applied to solve a substitute single objective problem, they may produce a dominated solution.
• Solutions generated in different iterations of the interactive method may dominate each other, which may be very misleading to the DM.
• No control mechanism allowing the DM to guide the search for the best compromise may be guaranteed. For example, the DM improving the value of one of the objectives in the reference point, expects that this objective will also improve in the newly generated Pareto optimal solution. Since metaheuristics do not give any guarantee on the quality of generated solutions, the change on this objective may even be opposite.
• Computational efficiency may be insufficient. In interactive methods, generation of new solutions needs to be done on-line. Thus, while a running time of several minutes may be fully acceptable for solving most single objective problems, it may be unacceptable when used within an interactive method.

Note that the potential non-optimality of metaheuristics may sometimes also be beneficial. In particular, a metaheuristic applied to the optimization of a linear scalarizing function may produce a non-supported Pareto-optimal solution as an approximate solution of the substitute problem. However, such results are due to randomness and cannot be relied upon.

Several examples of this traditional approach may be found in the literature. Alves and Clímaco (2000) use either simulated annealing or tabu search in an interactive method for 0-1 programming. Kato and Sakawa (1998); Sakawa and Shibano (1998); Sakawa and Yauchi (1999) have proposed a series of interactive fuzzy satisficing methods using various kinds of genetic algorithms adapted to various kinds of problems. Miettinen and Mäkelä use two different versions of genetic algorithms with constraint handling (among other solvers) in NIMBUS (Miettinen, 1999; Miettinen and Mäkelä, 2006; Ojalehto et al., 2007). Gabrys et al. (2006) use a single objective EA within several classical interactive procedures, i.e., STEP, reference point approach, Tchebycheff method, and GDF.

### 7.3 Semi-a-Posteriori Approach – Interactive Selection from a Set of Solutions Generated by a Multiobjective Metaheuristic

Classical a posteriori approaches to MOO assume that the set of (approximately) Pareto optimal solutions is presented to the DM, who is then able to select the best compromise solution. In recent years, many multiobjective metaheuristics have been developed, see, e.g., Coello et al. (2002); Deb (2001) for overviews. The typical goal of these methods is to generate in a single run a set of solutions being a good approximation of the whole Pareto set. Because metaheuristics such as evolutionary algorithms or particle swarm optimization concurrently work with sets (populations) of solutions, they can
simultaneously generate in one run a set of solutions which approximate the Pareto front. Thus, they are naturally suited as a posteriori MOO methods.

The set of potentially Pareto optimal solutions generated by a multiobjective metaheuristic may, however, be very large (Jaszkiewicz, 2002b), and even for relatively small problems may contain thousands of solutions. Obviously, most DMs will not be able to analyze such sets of solutions, in particular in the case of many objectives. Thus, they may require some further support which is typical for interactive methods.

In fact, numerous methods for interactive analysis of large finite sets of alternatives have been already proposed. This class of methods includes: Zionts method (Zionts, 1981), Korhonen, Wallenius and Zionts method (Korhonen et al., 1984), Köksalan, Karwan and Zionts method (Köksalan et al., 1988), Korhonen method (Korhonen, 1988), Malakooti method (Malakooti, 1989), Taner and Köksalan method (Taner and Köksalan, 1991), AIM (Lotfi et al., 1992), Light-Beam-Search-Discrete (Jaszkiewicz and Słowiński, 1997), Interactive Trichotomy Segmentation (Jaszkiewicz and Ferhat, 1999), and Interquad (Sun and Steuer, 1996). These methods are usually based on some traditional interactive methods (with on-line generation of solutions through optimization of the substitute problems). Thus, from the point of view of the DM the interaction may look almost the same, and he/she may be unaware whether the traditional or semi-a posteriori interaction is used. For example, using again the simplest version of the reference point method, in each iteration, the DM specifies a reference point in the objective space. The reference point defines an achievement scalarizing function. Then, the set of potentially Pareto optimal solutions is searched for the best solution with respect to this scalarizing function, which is presented to the DM. If the DM is not satisfied he/she adjusts the reference point and the process is repeated.

The advantages of this semi-a posteriori approach are:

- A large number of methods for both computational and interactive phases exist.
- All heavy calculations are done prior to interaction, thus interaction is very fast and saves the DM time.
- All solutions presented to the DM are mutually non-dominated. Note, however, that it does not necessarily mean that they are Pareto-optimal, since Pareto-optimality cannot be guaranteed by metaheuristics.
- Some control mechanisms allowing the DM to guide the search for the best compromise may be guaranteed.
- Additional various forms of statistical analysis of the pre-generated set of solutions (e.g. calculation of correlation coefficients between objectives) and graphical visualization may be used.

Generation of a high quality approximation of the whole Pareto set may, however, become computationally too demanding, especially in the case of realistic size combinatorial problems and a high number of objectives. Several studies (Jaszkiewicz, 2002b, 2003, 2004b; Purshouse, 2003) addressed the issue of
computational efficiency of multiobjective metaheuristics compared to iteratively applied single objective metaheuristics to generate an approximation to the Pareto front. Below the idea of the efficiency index of a multiobjective metaheuristic with respect to a corresponding single objective metaheuristic introduced in Jaszkiewicz (2002b, 2003, 2004b) is presented.

Despite a large number of both multiobjective metaheuristics and interactive methods for analysis of large finite sets of alternatives, few examples of the semi-a posteriori approach may be found in the literature. Hapke et al. (1998) use an interactive Light Beam Search-discrete method over a set of solutions of project scheduling problems generated by Pareto simulated annealing. Jaszkiewicz and Ferhat (1999) use the interactive trichotomy segmentation method for analysis of a set of solutions of a personnel scheduling problem generated by the same metaheuristic. Tanaka et al. (1995) use an interactive procedure based on iterative evaluation of samples of points selected from a larger set generated by an MOEA. They use a radial basis function network to model the DM’s utility function and suggest new solutions for evaluation.

7.4 Excursion: Efficiency Index

The general idea of the efficiency index presented here is to compare running times of single and multiobjective metaheuristics needed to generate solution sets of comparable quality. Many techniques have been proposed for evaluation of sets of solutions in the objective space, see Chapter 14. Obviously, no single measure can cover all aspects of the quality. In this section, we will focus on the measures based on scalarizing functions, mainly because they can be directly applied to evaluate results of both single and multiobjective metaheuristics.

Assume that a single objective metaheuristic is applied to the optimization of an achievement scalarizing function. The quality of the approximate solution generated by the single objective metaheuristic is naturally evaluated with the achievement scalarizing function. Furthermore, one may run a single objective metaheuristic a number of times using a representative sample of achievement scalarizing functions defined, e.g., by a representative sample of weight vectors. Then, the average value of the scalarizing functions over the respective generated solutions measures the average quality of solutions generated.

Scalarizing functions are also often used for evaluation of the results of multiobjective metaheuristics (e.g., Czyzak and Jaszkiewicz, 1996; Viana and de Sousa, 2000). In Hansen and Jaszkiewicz (1998) and Jaszkiewicz (2002a) we have proposed a scalarizing functions-based quality measure fully compatible with the above outlined approach for evaluation of single objective metaheuristics. The quality measure evaluates sets of approximately Pareto optimal solutions with average value of scalarizing functions defined by a representative sample of weight vectors.
In order to compare the quality and the computational efficiency of a multiobjective metaheuristic and a single objective metaheuristic, a representative sample $\Lambda_R$ of normalized weight vectors is used. Each weight vector $\lambda \in \Lambda_R$ defines a scalarizing function, e.g., a weighted Tchebycheff scalarizing function $s_\infty(z, z^0, \lambda)$. As the reference point $z^0$ one may use an approximation of the ideal point. All scalarizing functions defined by vectors from set $\Lambda_R$ constitute the set $S_a$ of functions. $|\Lambda_R|$ is a sampling parameter, too low values of this parameter may increase the statistical error, but, in general, the results will not depend on it.

In order to evaluate the quality of solutions generated by a single objective metaheuristic the method solves a series of optimization problems for each achievement scalarizing function from set $S_a$. For each function $s_\infty(z, z^0, \lambda) \in S_a$, the single objective metaheuristic returns a solution corresponding to point $z^\lambda$ in objective space. Thus, the average quality of solutions generated by the single objective metaheuristic is:

$$Q_s = \frac{\sum_{\lambda \in \Lambda_R} s_\infty(z^\lambda, z^0, \lambda)}{|\Lambda_R|}$$

Note that due to the nature of metaheuristics, the best solution with respect to a particular achievement scalarizing function $s_\infty(z, z^0, \lambda') \in S_a$ may actually be found when optimizing another function $s_\infty(z, z^0, \lambda'')$ from set $S_a$. If desired, this can be taken into account by storing all solutions generated so far in all single objective runs, and using the respective best of the stored solutions instead of $z^\lambda$ for each $\lambda \in \Lambda_R$.

In order to evaluate the quality of solutions generated by a multiobjective metaheuristic, a set $A$ of solutions generated by the method in a single run is evaluated. For each function $s_\infty(z, z^0, \lambda) \in S_a$ the best point $z^\lambda$ on this function is selected from set $A$, i.e., $s_\infty(z^\lambda, z^0, \lambda) \leq s_\infty(z, z^0, \lambda)$ $\forall z \in A$. Thus, the average quality of solutions generated by the multiobjective metaheuristic is:

$$Q_m = \frac{\sum_{\lambda \in \Lambda_R} s_\infty(z^\lambda, z^0, \lambda)}{|\Lambda_R|}$$

The quality of solutions generated by both the single objective metaheuristic and the multiobjective metaheuristic may be assumed approximately the same if $Q_s = Q_m$.

The two quality measures $Q_s$ and $Q_m$ are used in order to compare computational efforts (running times) needed by a single objective metaheuristic and a multiobjective metaheuristic to generate solutions of the same quality. First, the single objective metaheuristic is run for each function $s_\infty(z, z^0, \lambda) \in S_a$. Let the average time of a single run of the single objective metaheuristic be denoted by $T_s$ ($T_s$ is rather independent of $|\Lambda_R|$ since it is an average time of a single objective run). Then, the multiobjective metaheuristic is run. During the run of the multiobjective metaheuristic the quality $Q_m$ is observed using
the same set $S_a$ of functions. $Q_m$ is applied to a set $A$ of potentially Pareto optimal solutions generated up to a given time by the multiobjective meta-heuristic. This evaluation is repeated whenever set $A$ changes. Note that the set $S_a$ is used only to calculate $Q_m$, it has no influence on the work of the multiobjective metaheuristic, which is guided by its own mechanisms (e.g., Pareto ranking). The multiobjective metaheuristic is stopped as soon as $Q_m \leq Q_s$ (or if some maximum allowed running time is reached). Let the running time of the multiobjective metaheuristic be denoted by $T_m$ If condition $Q_m \leq Q_s$ was fulfilled one may calculate the efficiency index of the multiobjective metaheuristic with respect to the single objective metaheuristic:

$$EI = \frac{T_m}{T_s}$$

In general, one would expect $EI > 1$, and the lower $EI$, the more efficient the multiple objective metaheuristic with respect to the single objective method.

The efficiency index may be used to decide what MOO method might be most efficient. Assume that a DM using an iterative single-objective approach has to do $L$ iterations before choosing the best compromise solution\(^1\). If $EI < L$, then the off-line generation of approximately Pareto-optimal solutions with the use of a multiobjective metaheuristic would require less computational effort than on-line generation with the use of a single objective metaheuristic. In other words, $EI$ compares efficiency of the traditional interactive approach and the semi-a posteriori approach.

The efficiency index has been used in several studies (Jaszkiewicz, 2002b, 2003, 2004b), similar techniques were also used by Purshouse (2003). Although the quantitative results depend on the test problem (e.g., TSP, knapsack, set covering) and method, the general observation is that the relative computational efficiency of multiobjective metaheuristics reduces below reasonable levels with increasing number of objectives and instance size. For example, the computational experiment on multiobjective knapsack problems (Jaszkiewicz, 2004b) indicated that "the computational efficiency of the multiobjective metaheuristics cannot be justified in the case of 5-objective instances and can hardly be justified in the case of 4-objective instances". Thus, the semi-a posteriori would become very inefficient compared to an interactive approach in such cases.

7.5 Interactive Multiobjective Metaheuristics

Evolutionary algorithms and other metaheuristics are black box algorithms, i.e., it suffices to provide them with a quality measure for the solutions generated. There are almost no restrictions regarding this quality measure; a

\(^1\) Of course, in practice, this number is not known in advance and depends on the problem, the method and the DM, but often may be reasonably estimated on the basis of past experience
closed-loop description of the solution quality, or additional information such as gradients, is not necessary. This makes metaheuristics applicable to an enormous range of applications. In most applications, a solution’s quality can be evaluated automatically. However, there are also applications where a solution’s quality cannot be computed automatically, but depends on user preferences. In this case, interactive evolutionary computation (Takagi, 2001) may be used. An interactive evolutionary algorithm relies on the user to evaluate and rank solutions during the run. These evaluations are then used to guide the further search towards the most promising regions of the solution space.

One popular example is the generation of a picture of a criminal suspect, with the only source of information being someone’s memory (Caldwell and Johnston, 1991). The problem may be modeled as an optimization task where solutions are faces (built of some building blocks) and the objective function is a similarity to the suspect’s face. Of course, the similarity may be evaluated only subjectively by the human witness. Another example may be optimization of aesthetics of a design (Kamalian et al., 2004, 2006). An example in the realm of MCDM can be found in Hsu and Chen (1999).

Human fatigue is a crucial factor in such algorithms, as the number of solutions usually looked at by metaheuristics may become very large. Thus, various approaches based on approximate modeling (e.g., with a function learned from evaluation examples) of the DM’s preferences have been proposed in this field (Takagi, 2001). The evolutionary algorithm tries to predict a DM’s answers using this model, and asks the DM to evaluate only some of the new solutions.

There are apparent similarities of this field to interactive MOO. In both cases we are looking for solutions being the best from the point of view of subjective preferences. Thus, a very straightforward approach could be to apply an interactive evolutionary algorithm to a MOO problem asking the DM to evaluate presented solutions. However, in MOO, we assume to at least know the criteria that form the basis for the evaluation of a solution, and that these can be computed. Only how these objectives are combined to the overall utility of a solution is subjective. A direct application of interactive evolutionary algorithms to MOO would not take into account the fact that many solutions could be compared with the use of the dominance relations without consulting the DM. In other words, in an MOO problem, user evaluation is only necessary to compare mutually non-dominated solutions.

Interactive multiobjective metaheuristics are methods that are specifically adapted to interactive MOO, and use the dominance relation and the knowledge about the objectives to reduce the number of questions asked to the DM. Note that they may present to the DM intermediate solutions during the run, while in the case of the traditional or the semi-a priori approach only final solution(s) are presented to the DM.

As opposed to the traditional approach to interactive analysis, the interactive multiobjective metaheuristics do not make a complete run of a single objective method in each iteration. They rather modify the internal workings
of a single or multiobjective metaheuristic, allowing interaction with the DM during the run.

Several methods belonging to this class may be found in the literature. Tanino et al. (1993) proposed probably the first method of this kind. The method is based on a relatively simple version of a Pareto-ranking based multi-objective evolutionary algorithm. The evaluation of solutions from the current population is based on both dominance relation and on preferences expressed iteratively by the DM. The DM has several options: He/she may directly point out satisfactory/unsatisfactory solutions, or specify aspiration/reservation levels that are used to identify satisfactory/unsatisfactory solutions.

Kamalian et al. (2004) suggest to use an a posteriori evolutionary MOO followed by an interactive evolutionary algorithm. First a Pareto ranking-based evolutionary algorithm is run to generate a rough approximation of the Pareto front. Then, the DM selects a sample of the most promising solutions that are subsequently used as starting population of a standard interactive evolutionary algorithm.

Kita et al. (1999) interleave generations of a Pareto ranking-based evolutionary algorithm with ranking of the solutions by a DM, while Kamalian et al. (2006) allow the user to modify the Pareto ranking computed automatically by changing the rank of some of the solutions.

Several authors allow the DM to set and adjust aspiration levels or reference points during the run, and thereby guide the MOEA towards the (from the DM’s perspective) most promising solutions. For example, Fonseca and Fleming (1998) allow the user to specify aspiration levels in form of a reference point, and use this to modify the MOEA’s ranking scheme in order to focus the search. This approach is discussed in more detail also in Chapter 6. The approach proposed by Geiger (2007) is based on Pareto Iterated Local Search. It first approximates the Pareto front by calculating some upper and lower bounds, to give the DM a rough idea of what can be expected. Based on this information, the DM may restrict the search to the most interesting parts of the objective space. Ulungu et al. (1998) proposed an interactive version of multiple objective simulated annealing. In addition to allowing to set aspiration levels, solutions may be explicitly removed from the archive, and weights may be specified to further focus the search. Thiele et al. (2007) also use DM’s preferences interactively expressed in the form of reference points. They use an indicator-based evolutionary algorithm, and use the achievement scalarizing function to modify the indicator and force the algorithm to focus on the more interesting part of the Pareto front.

Deb and Chaudhuri (2007) proposed an interactive decision support system called I-MODE that implements an interactive procedure built over a number of existing EMO and classical decision making methods. The main idea of the interactive procedure is to allow the DM to interactively focus on interesting region(s) of the Pareto front. The DM has options to use several tools for generation of potentially Pareto optimal solutions concentrated in the desired regions. For example, he/she may use weighted sum approach,
utility function based approach, Tchebycheff function approach or trade-off information. Note that the preference information may be used to define a number of interesting regions. For example, the DM may define a number of reference (aspiration) points defining different regions. The preference information is then used by an EMO to generate new solutions in (hopefully) interesting regions.

The interactive evolutionary algorithm proposed by Phelps and Köksalan (2003) allows the user to provide preference information about pairs of solutions during the run. Based on this information, the authors compute a “most compatible” weighted sum of objectives (i.e., a linear achievement scalarizing function) by means of linear programming, and use this as single substitute objective for some generations of the evolutionary algorithm. Note that the weight vector defines a single search direction and may change only when the user provides new comparisons of solutions. However, since only partial preference information is available, there is no guarantee that the weight vector obtained by solving the linear programming model defines the DM’s utility function, even if the utility function has the form of a weighted sum. Thus, the use of a single weight vector may bias the algorithm towards some solutions not necessarily being the best for the DM. This bias may become even more significant when the DM’s preferences cannot be modeled with a linear function.

Instead of using linear programming to derive a weighting of the objectives “most compatible” with the pairwise comparisons as in Phelps and Köksalan (2003), Barbosa and Barreto (2001) use two evolutionary algorithms, one to find the solutions, and one to determine the “most compatible” ranking. These EAs are run in turn: first, both populations (solutions and weights) are initialized, then the DM is asked to rank the solutions. After that, the population of weights is run for some generations to produce a weighting which is most compatible with the user ranking. Then, this weighting is used to evolve the solutions for some generations, and the process repeats. Todd and Sen (1999) also try to learn the user’s utility function, but instead of only considering linear weightings of objectives, they use the preference information provided by the DM to train an artificial neural network, which is then used to evaluate solutions in the evolutionary algorithm.

The method of Jaszkiewicz (2007) is based on the Pareto memetic algorithm (PMA)(Jaszkiewicz, 2004a). The original PMA samples the set of scalarizing functions drawing a random weight vector for each single iteration and uses this during crossover and local search. In the proposed interactive version, preference information from pairwise comparisons of solutions is used to reduce the set of possible weight vectors. Note that different from the approaches above, it is not attempted to identify one most likely utility function, but simultaneously allows for a range of utility functions compatible with the preference information specified by the user.
7.6 Summary

In this chapter, we have described three principal ways for interactively using evolutionary algorithms or similar metaheuristics in MOO.

The traditional approach to interactive MOO with the use of single objective metaheuristics is a straightforward adaptation of the classical interactive methods. It suffers, however, from a number of weaknesses when metaheuristics are used in place of exact solvers, since many important theoretical properties are not valid in the case of heuristic solvers.

The semi-a posteriori approach allows combining multiobjective metaheuristics with methods for interactive analysis of large finite sets of alternatives. An important advantage of this approach is that various methods from both classes are available. The semi-a posteriori approach allows overcoming a number of weaknesses of the traditional approach. It may, however, become computationally inefficient for large problems with larger number of objectives.

Interactive multiobjective metaheuristics is a very promising class of methods specifically adapted to interactive solving of hard MOO problems. According to some studies (Phelps and Köksalan, 2003; Jaszkiewicz, 2007) they may be computationally efficient even for a large number of objectives and require relatively low effort from the DM. Such specifically designed methods may combine the main advantages of metaheuristics and interactive MOO avoiding weaknesses of the other approaches. Note that the way the DM interacts with such methods may be significantly different from the traditional approaches. For example, the DM may be asked to compare solutions being known to be located far from the Pareto optimal set.

Both interactive methods and the use of metaheuristics are among the most active research areas within MOO. Combination of these approaches may results in very effective methods for hard, multidimensional MOO problems. Despite of a number of proposals known from the literature, this field has not yet received appropriate attention from MOO researchers community.

References


