Chapter 13
Multiobjective Memetic Algorithms

Andrzej Jaszkiewicz, Hisao Ishibuchi, and Qingfu Zhang

13.1 Introduction

Multiple conflicting points of view, which are often taken into account in real life applications, naturally result in a multiple objective optimization problem (MOP) [848]. In order to find the best compromise solution of a MOP, or a good approximation of it, Multiobjective Optimization (MOO) methods need some preference information from a decision maker. According to when and how the preference information is used in the solution procedure, MOO methods can be classified as either methods with a priori, a posteriori, or progressive (interactive) articulation of preferences [400].

In recent years, the demand for new applications and the increasing computing power have led to growing interest in computationally hard multiobjective problems, e.g. nonlinear or combinatorial optimization problems. These problems arise in many areas such as scheduling, timetabling, production facilities design, vehicle routing, telecommunication routing, investment planning and location. Problems of this kind are difficult to solve even in the single objective case. Encouraged by the success of metaheuristics in single-objective optimization (see e.g. [694]), much effort has been made in developing MOO metaheuristics.

Andrzej Jaszkiewicz
Poznan University of Technology, Institute of Computing Science, Piotrowo 2,
60-965 Poznan, Poland
e-mail: jaszkiewicz@cs.put.poznan.pl

Hisao Ishibuchi
Department of Computer Science and Intelligent Systems, Osaka Prefecture University,
1-1 Gakuen-cho, Nakaku, Sakai, Osaka 599-8531, Japan
e-mail: hisaoi@cs.osakafu-u.ac.jp

Qingfu Zhang
The School of Computer Science & Electronic Engineering University of Essex,
Colchester, CO4 3SQ, UK
e-mail: qzhang@essex.ac.uk

F. Neri et al. (Eds.): Handbook of Memetic Algorithms, SCI 379, pp. 201–217.
springerlink.com

Traditional MOO methods usually assume that an underlying single-objective exact solver is available. This solver is used to solve a series of substitute single-objective optimization problems sequentially. The objective functions of these substitute optimization problems could be an aggregation function of the individual objectives of the MOP in question. Their optimal solution can be Pareto optimal solutions of the MOP under some conditions. However, for many hard MOPs, no efficient exact solvers are available. One can use a single-objective metaheuristic instead of an exact solver. A single run of the metaheuristic can generate a single approximate Pareto-optimal solution and therefore many runs are required to generate multiple solutions. This approach is simple but not very efficient.

Many dedicated multiobjective metaheuristics have recently been developed [131, 196, 418]. These methods aim at generating in a single run a set of solutions for approximating the whole Pareto optimal front. The set of solutions could be then presented to the decision maker (DM) to allow it to choose the best compromise solution in a posteriori or interactive manner [422].

A multiobjective metaheuristic is often a modified version of a single objective heuristic method. It is natural to expect that the best results may be achieved by adapting the most efficient single objective methods to the multiobjective problems. Memetic algorithms proved to be one of the most efficient metaheuristic paradigms for single objective optimization [618]. For this reason, many attempts have been made to extend memetic algorithms to multiobjective optimization.

The purpose of this review is to present and discuss basic concepts in multiobjective memetic algorithms and to characterize some state-of-the-art algorithms. In the next section, we introduce some basic definitions in MOO. In the third section, we discuss the main ideas in multiobjective memetic algorithms. The fourth section presents several typical multiobjective memetic algorithms. Some specific implementation issues are discussed in the fifth section. In the last section we discuss some further research topics in this area.

13.2 Basic Definition and Concepts

In this section, we introduce some basic concepts and aggregation functions in multiobjective optimization.

13.2.1 Basic Concepts

A *multiobjective optimization problem* (MOP) can be stated as follows:

$$\begin{aligned} &\text{maximize } F(x) = (f_1(x), \ldots, f_m(x)) \\ &\text{subject to} \qquad\quad x \in \Omega \end{aligned} \tag{13.1}$$

where Ω is the *decision space*, $F : \Omega \to R^m$ consists of m real-valued objective functions. R^m is called the *objective space*. The *attainable objective set* is defined as the set $\{F(x)|x \in \Omega\}$.

Ω can be a subset of a base set S and often be described by several constraints $C_1,\ldots,C_k$. i.e.

$$\Omega = \{x \in S | x \text{ satisifies all the constraints } C_1,\ldots,C_k\}. \tag{13.2}$$

In this case, Ω is called the feasible solution space and any solution in Ω is a feasible (candidate) solution. A solution in S is infeasible if it is not in Ω, in other words, it violates at least one constraint.

If $x \in R^n$, all the objectives are continuous and Ω is described by

$$\Omega = \{x \in R^n | h_j(x) \leqslant 0, j = 1,\ldots,k\}, \tag{13.3}$$

where h_j are continuous functions, we call (1) a *continuous MOP*. If Ω is a finite or countably infinite set, then (1) is a *combinatorial MOP*.

Domination is widely used to compare different solutions in multiobjective optimization.

Definition 13.1. Let $u, v \in R^m$, u is said to *dominate* v if and only if $u_i \geqslant v_i$ for every $i \in \{1,\ldots,m\}$ and $u_j > v_j$ for at least one index $j \in \{1,\ldots,m\}$[1].

Domination defines a strict partial ordering in the objective space- not any two vectors are comparable based on domination. For example, $(1,0)$ and $(0,1)$ do not dominate each other.

Definition 13.2. Let $x, y \in \Omega$, x is said to *dominate* y if and only if $F(x)$ dominates $F(y)$.

Obviously, a rational decision maker prefers x to y if x dominates y.

Definition 13.3. $x^* \in \Omega$ is a *Pareto optimal solution* and $F(x^*)$ is a *Pareto optimal vector* to (13.1) if no other solution in Ω can dominate x^*. The set of all the Pareto optimal solutions is called the Pareto optimal set (PS) and the set of all the Pareto optimal vectors is the Pareto front (PF).

We should point out that the above definition refers to the global optimality. x is called locally Pareto optimal if it cannot be dominated by any solutions in a neighborhood of x.

In many real-life applications of multiobjective optimization, an approximation to the PF is required by a decision maker for selecting the final preferred solution. Most MOPs may have many or even infinite Pareto optimal vectors. It is very time-consuming, if not impossible, to obtain the complete PF. On the other hand, the decision maker may not be interested to have an unduly huge number of Pareto optimal vectors to deal with due to overflow of information. Therefore, many multiobjective optimization algorithms are to find a manageable number of approximate Pareto optimal solutions to approximate the Pareto set or Pareto front. Researchers and practitioners are often more interested in approximating the Pareto front than

[1] This definition of domination is for maximization. All the inequalities should be reversed if the goal is to minimize the objectives in (13.1).

the Pareto set because the objective space is of lower dimension and it is easy to visualize an approximate Pareto front. However, recent work has shown that approximation of the Pareto set is also very important.

Definition 13.4. Given a set of solutions P, $x \in P$ is called a nondominated solution in P if no solution in P can dominate x.

Many multiobjective metaheurstics are based on Pareto dominance. These methods often select nondominated solutions from a set of solutions.

Definition 13.5. $z^{id} = (z_1, \ldots, z_m)$ is called the ideal objective vector if z_i is the maximal function value of $f_i(x)$ over Ω.

Definition 13.6. $z^{nadir} = (z_1, \ldots, z_m)$ is call the nadir objective vector if

$$z_i = inf\{y_i | (y_1, \ldots, y_m) \in PF\} \tag{13.4}$$

Ideal objective vectors and nadir vectors in the objective space are the upper and lower bounds of the PF, which are of interest since they are useful for determining the range of the Pareto front and for normalizing the objectives so that all the objectives in the same range. A typical normalization is:

$$f_i(x) \leftarrow \frac{f_i(x) - z_i^{nad}}{z_i - z_i^{nad}} \tag{13.5}$$

It might be not practical to obtain the exact ideal and nadir vectors, one can substitute them by approximate ones.

13.2.2 Aggregation Functions

In traditional optimization, a widely-used strategy for dealing with a MOP is to aggregate all the individual objective functions and then optimize the aggregation function. In the following, we introduce three popular aggregation approaches.

13.2.3 Weighted Sum Approach

This approach considers a convex combination of the different objectives. Let $\lambda = (\lambda_1, \ldots, \lambda_m)^T$ be a weight vector, i. e., $\lambda_i \geqslant 0$ for all $i = 1, \ldots, m$ and $\sum_{i=1}^{m} \lambda_i = 1$. Then the aggregated function is

$$g^{ws}(x|\lambda) = \sum_{i=1}^{m} \lambda_i f_i(x). \tag{13.6}$$

where we use $g^{ws}(x|\lambda)$ to emphasize that λ is a coefficient vector in this objective function while x is the variables to be optimized. A maximal solution of a weighted sum function is Pareto optimal to (13.1) if all the weights are positive. Moreover,

for any Pareto optimal solution x^* to a convex MOP, there exists a weight vector such that x^* is the maximal solution to (13.6). However, for a non-convex MOP, there may exist a Pareto optimal solution which is not a maximal solution to any weighted sum function.

13.2.4 Tchebycheff Approach

In this approach, the aggregation function to be minimized is in the form

$$g^{te}(x|\lambda, z^*) = \max_{1 \leqslant i \leqslant m} \{\lambda_i(-z_i^* - f_i(x))\} \tag{13.7}$$

where z^* is the ideal point or a point dominated by the ideal point. Each Tchebycheff aggregation function has at least one global minimum which is Pareto optimal to (13.1). Under some mild conditions for each Pareto optimal point x^*, there exists a weight vector λ such that x^* is an optimal solution of (13.7). Therefore, one is able to obtain different Pareto optimal solutions by altering the weight vector.

One weakness with this approach is that its aggregated function could be flat in some regions. To overcome it, the following aggregated function can be used:

$$g^{te}(x|\lambda, z^*) + \rho g^{ws}(x|\lambda) \tag{13.8}$$

Aggregation methods are still a very active research topic in traditional optimization. The readers interested in more detail about aggregation methods may wish to consult [233, 241, 599].

13.3 Adaptation of Memetic Algorithms for Multiobjective Optimization – Basic Concepts

Memetic algorithms have to evaluate or compare a set of solutions at each generation for determining their contribution to the next generation. In the single objective case, several different evaluation functions or mechanisms have been used and studied in solution evaluation, the objective function itself, however, is the most natural and widely used evaluation function [618]. In multiobjective optimization, no such a natural choice for the evaluation exists. The evaluation mechanism is one of the major issues in the design of multiobjective memetic algorithms. A good evaluation mechanism should guide the solutions generated to

- approach the Pareto front,
- and at the same time disperse over all (or some desired) regions of the Pareto front.

Two main classes of evaluation mechanisms have been proposed for multiobjective memetic algorithms, i.e., mechanisms based on the dominance relation and mechanisms based on aggregation functions. Of course, these two mechanisms could be hybridized together. Below we discuss these two evaluation mechanisms.

13.3.1 Dominance-Based Evaluation Mechanisms

As pointed out in Section 2, the dominance relation defines a partial order in the set of all feasible solutions. All the Pareto-optimal solutions are the best with respect to this order. Therefore, the use of dominance relation in evaluation mechanisms creates a selection pressure towards the Pareto front. Dominance relation alone leaves, however, many pairs of solutions incomparable. For this reason, dominance relation on its own may not be able to define a single best solution in a neighborhood or in a tournament. Thus, multiobjective memetic algorithms need additional evaluation mechanisms with dominance relation to distinguish different solutions.

Probably, the most popular dominance based evaluation mechanism is Pareto ranking, which was originally suggested by Goldberg [325] and has been widely used in multiobjective evolutionary algorithms (see e.g.[131, 196]). In this mechanism, the dominance relation is used to rank all the solutions in the current population. Different algorithms may use slightly different versions of Pareto ranking. Srinivas and Deb [844] used the most direct implementation of the Goldberg's idea in their Nondominated Sorting Genetic Algorithm (NSGA) [199]. It assigns rank 1 to all solutions nondominated in the current population. Then, the nondominated solutions are temporarily removed from the population and the next rank is assigned to the solutions nondominated in the remaining part of the population. The process is continued until all solutions in the population are ranked.

Dominance relation may also be used to guide local search-based memetic algorithms. For example, Knowles and Corne [472, 473] proposed a greedy local search method mainly based on dominance relation. Their idea is to accept a new neighborhood solution if it dominates the current solution. In population-based Pareto local search [21, 50, 705], the neighborhood of each solution of the current population is explored, and if no solution of the population weakly dominates a generated neighbor, the neighbor is added to the population.

An obvious advantage of dominance relation is its independence on any monotonic transformation of objective functions. Furthermore, particular dominance-based evaluation mechanisms are usually very simple and have no or few parameters. For example, Pareto local search is a fully parameter-free method.

Dominance-based evaluation mechanisms may have, however, some significant disadvantages. Although dominance relation assures the pressure towards the Pareto front, it does not necessarily assure the dispersion of the solutions over all regions of the Pareto front. Thus, the basic method often needs to be extended by introducing some additional dispersion mechanisms. For example, several researchers [276, 472, 844] suggested the use of fitness sharing to improve Pareto ranking. The idea is to penalize a solution if it is too close, either in the objective or in the decision space, to some other solutions in the current population. Note that many fitness sharing techniques use some kind of distance measures in the objective space. Hence, the techniques are not invariant of scaling and more general of monotonic transformation of objective functions.

Another disadvantage of dominance-based evaluation mechanisms is that the selection pressure decreases with the growing number of objectives. The larger the

number of objectives is, the lower the chance that one of two solutions dominates the other. In particular, a population of a multiobjective memetic algorithm may easily contain mainly or only mutually non-dominated solutions which are incomparable based on the dominance relation.

Increasing the number of objectives may also deteriorate the efficiency of algorithms based purely on the dominance relation. For example, in the case of Pareto local search, the number of neighborhood solutions to be accepted may become very large, and the size of the population to be maintained may grow enormously.

Furthermore, efficient single-objective local search algorithms usually use a number of advanced, problem-specific speed-up techniques based on the properties of the objective function. Such techniques often cannot be directly adapted to dominance-based mechanisms. There is still very little work that applies speed-up techniques in local search algorithms based on dominance relations [542].

13.3.2 Aggregation Function-Based Evaluation Mechanisms

Evaluation of new solutions with the use of Aggregation functions is another typical evaluation mechanism. Aggregation functions have well-established theoretical properties as tools for generating Pareto-optimal solutions in traditional MOO (see section 2). Thanks to these properties of Aggregation functions, their use also induces a pressure towards the Pareto front. Of course, a single Aggregation function would guide a metaheuristic towards a single Pareto solution. This drawback could be, however, overcome by the use of multiple Aggregation functions defined by various weight vectors. For example, Serafini [805] used the mechanism of random walk to modify the weights randomly in each iteration. Ulungu et al. [897] and Zhang and Li [957] used a predefined set of well dispersed weight vectors. Czyzak and Jaszkiewicz [696] and Hansen [358] modified the weights deterministically in each iteration in order to obtain a form of repulsion between a population of solutions, Hajela and Lin [350] allowed the weights to evolve during the search. Ishibuchi and Murata [409] and Jaszkiewicz [419] generated weight vectors randomly in each iteration.

An important advantage of Aggregation functions-based evaluation mechanisms is the fact that by the use of various weight vectors they naturally assure dispersion of the search over all regions of the Pareto front. Thus, no additional dispersion mechanisms like the fitness sharing are needed. Another advantage of such mechanisms is that many speed-up techniques may easily be used in local search based on Aggregation functions.

A disadvantage of Aggregation functions-based evaluation mechanisms is their dependence on monotonic transformations of objective functions. A simple change of units in one objective may significantly deteriorate the algorithm performance. It is thus very important to assure that all the objectives take their values in comparable ranges. It may be achieved with the use of normalized objective values (see section 2). Some methods, e.g. Jaszkiewicz's MOGLS [420], perform automatic scaling of objectives.

According to the properties mentioned in section 2, weighted Tchebycheff and composite Aggregation functions have the advantage over linear Aggregation functions of being able to generate all Pareto optimal solutions. Note, however, that the properties concern only optimal solutions of the Aggregation functions. A suboptimal solution of a linear Aggregation function found by a metaheuristic may appear to be a nonsupported Pareto optimal solution. Some experiments indicated that the use of linear Aggregation functions may yield better results for some particular problems (see e.g. [359, 419]). Nevertheless, weighted Tchebycheff or composite Aggregation functions should still be considered as the first choice for Aggregation functions-based evaluation mechanisms.

13.3.3 Problem Landscapes in Multiobjective Optimization

Intuitively, a problem (fitness) landscape is a graph where solutions play the role of vertices and edges indicate the neighborhood relation or a distance measure in the decision space between solutions [581, 618, 762]. In the single-objective case, it is labeled on vertices with real values of the fitness function. In the multiobjective case, it is labeled with vectors of real values.

A simple conclusion of the No free lunch theorem [940] is that no optimization algorithm may work for all possible landscapes. The properties of landscapes may be analyzed e.g. by the distance between local optima [76], fitness-distance correlation [581] or scatter plots of fitness versus distance. Several authors observed that single-objective memetic algorithms perform very well for problems with the 'big valley' property. This property means that there is a correlation between quality of solutions and their distance, i.e. good solutions tend to be located close according to some distance measure in the decision space.

Landscape analysis of multiobjective problems has not achieved significant attention yet. Very few such studies have been performed [276, 357]. However, it is natural to expect that (approximately) Pareto-optimal solutions do not need to be close in the decision space. For example, Pareto-optimal solutions corresponding to optima of particular objectives may be very distant in the decision space if the objectives are independent or conflicting. On the other hand, some solutions close in the objective space may also be close in the decision space [357].

This observation puts some new light on the typical statement that "population-based methods are ideal candidates for solving multiobjective problems" (see e.g. [196], Preface). In fact, single-objective population-based methods are rather designed to converge towards the vicinity of good solutions, and some natural convergence mechanism, e.g. genetic drift [328], may be beneficial in the single-objective case. The same convergence may, however, cause a multiobjective population-based method converge to a sub-region of the Pareto front only. Thus, dispersion mechanisms that are 'side' elements of single-objective algorithms may become crucial in the multiobjective case. Indeed, some studies indicate that various population-based methods have problems with assuring proper dispersion of solutions even if the convergence to the Pareto front is very good [420].

Furthermore, the 'big valley' property and the convergence of the population well explain the effectiveness of recombination operators. The recombination constructs a new solution by combining properties of the parents, and so, creating a solution being close to the parents and other good solutions. This offspring solution may be then efficiently improved by local search. In fact, some very successful recombination operators such as respectful operators [581] are directly designed to preserve properties common to both parents.

In multiobjective cases, the population may contain some very distant solutions with few or no common properties. Recombination of such solutions does not need to produce good offspring and may deteriorate efficiency and effectiveness of the whole algorithm. Thus multiobjective memetic algorithms may require some specialized mechanisms for selection of promising parents for recombination [412].

13.3.4 Archive of Potentially Pareto-optimal Solutions

In the single-objective case, the outcome of the algorithm should be the best solution found, even if in some cases it is not contained in the final population. In the multiobjective case, an analogue of the single best solution is the set of potentially Pareto-optimal solutions, i.e. solutions that are not dominated by any other solutions generated by the algorithm. Some initial population-based multiobjective memetic algorithms did not take this fact into account and assumed that their outcome is the final population. This means that many potentially Pareto-optimal solutions could have been lost. Thus, it is natural to maintain an additional archive of potentially Pareto-optimal solutions. Please note, however, that the size of this archive may become enormously large and its maintenance may become the main factor influencing the efficiency of the algorithm. Thus some techniques for reduction of the archive size have been proposed.

13.3.5 Evaluation of Multiobjective Memetic Algorithms

With the increasing number of multiobjective memetic algorithms and other metaheuristics, the issue of their evaluation and comparison becomes of crucial importance. Although full evaluation of single objective metaheuristics is already a complicated task that involves many aspects like quality of results and computational efficiency, some difficulties are specific to the multiple objective case. In the single objective case, when two algorithms generate two solutions, their comparison is straightforward. Either one of the solutions is better or they are equally good on the single objective function. In the multiobjective case we are dealing with evaluation and comparison of sets of solutions from the point of view of multiple criteria. In some cases, two sets of potentially Pareto-optimal solutions may be compared based on the dominance relation only with the use of so-called set outperformance relations [421, 970]. For example if all solutions in one set are covered (are dominated or equal) by solutions from another set the latter should be considered better. These relations leave, however, many pairs of sets incomparable. Thus, a number of

quality indicators have been proposed. The quality indicators, usually, assign a single real value to each set. It is natural to expect that the proper quality indicators should properly rank sets comparable with set outperformance relations. Several quality indicators like hypervolume or R-indicator have this property. For detailed analysis of this issue see e.g. [421, 969, 970].

13.4 Examples of Multiobjective Memetic Algorithms

13.4.1 MOGLS of Ishibuchi and Murata

Ishibuchi and Murata [409] proposed multiobjective genetic local search (MOGLS), which is the first well-known multiobjective memetic algorithm. Their MOGLS uses a weighted sum fitness function for parent selection and local search. Pareto dominance is used only for maintaining an archive population. The archive population is updated at every generation so that it includes all non-dominated solutions among examined ones during the current execution of MOGLS. At each generation, the weight vector is randomly updated when a pair of parents is selected from the current population by roulette wheel selection based on the weighted sum fitness function. An offspring is generated from the selected pair of parents. The current weight vector is used for local search from the generated offspring. When the next pair of parents is selected, the weight vector is randomly updated. In this manner, the next population is generated by iterating random weight update, parent selection, crossover, mutation and local search. Some non-dominated solutions in the archive population are randomly selected and added to the current population as elite individuals. MOGLS of Ishibuchi and Murata [409] has some good properties such as the use of archived non-dominated solutions as parents and the use of aggregation functions with multiple weight vectors. Its performance, however, is not so high because its implementation is too naive. Its performance can be easily improved by a number of simple tricks such as the increase in the selection pressure for parent selection, the choice of a good starting solution for local search with the current weight vector, and the specification of a good balance between genetic search and local search [409, 410].

13.4.2 M-PAES

Memetic Pareto Archived Evolution Strategy (M-PAES) method proposed by Knowles and Corne [472, 473] is a memetic algorithm based fully on the dominance relation. The method is composed of two sequential phases - local search phase and recombination phase. In the local search phase the local search is independently applied to each starting solution. The local search is based on the dominance relation. The new neighborhood solution is rejected if it is dominated by the current solution, and accepted if it dominates the current solution. If the two solutions are mutually nondominated, the two solutions are compared with a local archive of potentially Pareto-optimal solutions and the one from a less crowded region is accepted. This

acceptance rule is an additional dispersion mechanism. In recombination phase, randomly selected solutions from the current population created in local search phase and solutions from global archive of potentially Pareto-optimal solutions are recombined. The acceptance criterion of the offspring again takes into account both dominance relation and location in the more or less crowded region of the global archive. Although the method uses some dispersion mechanism in both phases, the experiment in [420] indicated that the method may be strongly affected by genetic drift.

13.4.3 NSGA-II with LS

Deb and Goel [198] proposed an algorithm in which local search is used to improve results of a standard multiobjective evolutionary algorithm. The method combines dominance-based and aggregation functions-based guiding mechanisms. The method starts by using Nondominated Sorted Genetic Algorithm-II NSGA-II that uses recombination and some dominance-based elitist dispersion mechanism. The algorithm is fully based on the dominance relation. Pairs of solutions are selected from the current population by binary tournament selection based on the dominance relation and a crowding measure. In the second phase each solution generated by NSGA-II is a starting point for local search. The local search is based on weighted linear aggregation functions. The weight vector is set automatically depending on the location of the solution in comparison to other solutions. Intuitively, solutions located close to the best value on a given objective will have a large weight value corresponding to this objective. In other words, each solution is pushed in the direction in which it is already good. The authors report that the hybrid approach improves performance of NSGA-II on a number of engineering design problems.

Cheng et al. [125] also proposed a multiobjective memetic algorithm based on the NSGA II method [199]. In each generation of NSGA-II they apply local search to just one potentially Pareto-optimal solution. To choose this solution a weighted linear aggregation function is drawn at random. Then 2-tournament based on the current aggregation function is used to select the single solution to which local search is applied. The method is applied to the multiobjective job shop scheduling problem on which the method performs better than a benchmark non-memetic evolutionary algorithm.

Garret and Dasgupta [304] hybridized NSGA II with a variant of tabu search for the multiobjective quadratic assignment problems. They studied the influence of the length of tabu search runs and noticed that with increasing number of objectives it becomes more beneficial to perform more short runs.

13.4.4 MOGLS of Jaszkiewicz

Jaszkiewicz has proposed a multiobjective genetic local search (MOGLS) method based on the aggregation functions guiding mechanism [419]. Alike the MOGLS of Ishibuchi and Murata a random weight vector of the aggregation function is drawn in

each iteration. In a single iteration, two solutions are recombined and local search, or, more generally, a specialized heuristic, is applied to the offspring. The local search optimizes the current aggregation function. The random selection of weight vectors assures dispersion over all regions of the Pareto front. In each iteration, the search is pushed in a different direction but always towards the Pareto front. The method uses a relatively large population of solutions and some effort is made to define its size automatically. The solutions for recombination are also drawn based on the current aggregation function. In the original version parents are drawn at random from among some (e.g. 20) solutions being the best on this function. Thus only solutions being very good from the point of view of the current aggregation function could be recombined. This technique gives a high chance of constructing a new solution that performs well on the same function. In the improved version called Pareto memetic algorithm [421], this selection was based on the tournament selection with many solutions taking part in the single tournament. This mechanism improves efficiency of the selection. The method has been applied to the multiobjective traveling salesperson problem (TSP) [419], multiobjective knapsack problem [420], and multiobjective set covering problem [421]. In [423] it has been combined with a very efficient Lin-Kernighan local search for single objective TSP. Since a weighted linear aggregation function was used in this case, it was possible to directly apply the Lin-Kernighan method for the standard single objective TSP.

13.4.5 RM-MEDA

RM-MEDA (Regularity Model-Based Multiobjective Estimation of Distribution Algorithm) [957] for continuous MOPs is a an example of utilizing problem-specific knowledge in designing multiobjective heuristics. Under certain smoothness assumptions, the *PS* of a continuous MOP defines a piecewise continuous $(m-1)$-dimensional manifold in the decision space. where m is the number of the objectives. Therefore, the *PS* of a continuous bi-objective optimization problem is a piecewise continuous curve in R^n while the *PS* of a continuous MOP with three objectives is a piecewise continuous surface. The idea behind RM-MEDA is to force its population to converge to a $(m-1)$ piecewise continuous $(m-1)$-dimensional manifold. At each generation, RM-MEDA firstly extracts statistical information from some selected good solutions and then estimates the distribution of Pareto optimal points by using a probability model whose centroid is a $(m-1)$-dimensional manifold. New solutions are generated by sampling from the model thus built. The major computational overhead in RM-MEDA lies in model building. It is very costly to build a very accurate model. The local principal component analysis, a low-cost statistical algorithm, has been used for modeling. The experimental results have demonstrated that RM-MEDA works well, particularly when the PS is not a linear manifold. Recently, RM-MEDA has been generalized to the case when the dimensionality of the PS is unknown [961].

13.4.6 MOEA/D

MOEA/D (multiobjective evolutionary algorithm based on decomposition) [955] is a simple and generic multiobjective metaheuristic. It uses a aggregation method to decompose the MOP into N single objective optimization subproblems and solves these subproblems simultaneously (where N is a control parameter set by users). In MOEA/D, N procedures are employed and different procedures are for solving different subproblems. A neighborhood relationship among all the subproblems (procedures) is defined based on the distances of their weight vectors. Neighboring subproblems should have similar fitness landscapes and optimal solutions. Therefore, neighboring procedures can speed up their searches by exchanging information. In a simple version of MOEA/D [955] , each individual procedure keeps one solution in its memory, which could be the best solution found so far for its subproblems; it generates a new solution by performing genetic operators on several solutions from its neighboring procedures, and updates its memory if the new solution is better than old one for its subproblem. A procedure also passes its new generated solution on to some (or all) of its neighboring procedures, who will update their current solutions if the received solution is better. A major advantage of MOEA/D is that single objective local search can be used in each procedure in a natural way since its task is for optimizing a single objective subproblem. Several improvements on MOEA/D have been made recently. Li and Zhang suggested using two different neighborhood structures for balancing exploitation and exploration [516]. Zhang et al [961] proposed a scheme for dynamically allocating computational effort to different procedures in MOEA/D in order to reduce the overall cost and improve the algorithm performance, this implementation of MOEA/D is efficient and effective and has won the CEC'09 MOEA competition. Nebro and Durillo developed a thread-based parallel version of MOEA/D [652], which can be executed on multicore computers. Palmers et al. proposed an implementation of MOEA/D in which each procedure recorded more than one solutions [699]. Ishibuchi et al. proposed using different aggregation functions at different search stages [413].

13.4.7 MGK Population Heuristic

Gandibleux at al. [301] proposed a hybrid population heuristic for combinatorial problems. They used a Pareto ranking-based evolutionary algorithm as the population heuristic. The algorithm starts by seeding the initial population with some very good solutions. The solutions may be supported by Pareto-optimal solutions found by either an exact or approximate method. Furthermore, local search is applied during the run of the method. The method has been applied to a permutation scheduling problem and to the knapsack problem.

13.4.8 Memetic Approach by Chen and Chen

Chen and Chen [124] combine a dominance-based local search with a Pareto ranking-based multiobjective evolutionary algorithm. They use a special kind of local search with several species exploiting different regions in the objective space. The method has been applied to the problem of flexible process sequencing.

13.4.9 SPEA2 with LS

Schuetze et al. [797] combined SPEA2 [968] algorithm with local search for continuous multiobjective problems. They developed Hill-Climber with Sidestep procedure based on the dominance relation than can move either towards or along the Pareto front. They reported significant improvements of the performance in comparison to the standard SPEA2 algorithm.

SPEA2 was also hybridized with a gradient-based local search for continuous problems by Harada eta al. [365]. They compared two approaches, GA with local search and GA then local search. They reported better performance of the latter for continuous problems.

13.4.10 Interactive Memetic Algorithm by Dias et al.

Dias et al. [212] proposed an interactive multiobjective memetic algorithm. The algorithm optimizes an aggregation function based on the preferences of the decision maker. The algorithm works like a standard single objective memetic algorithm, however, the whole set of potentially Pareto-optimal solutions may be presented to the decision maker. The algorithm uses hot start technique. When the decision maker changes his/her preferences the current population is optimized further with a new aggregation function. The algorithm has been applied to the dynamic location problem.

13.4.11 SMS-EMOA with Local Search

Koch et al. [475] hybridized SMS-EMOA [64] with a gradient-based local search for continuous problems. SMS-EMOA in an indicator-based evolutionary algorithm that optimizes the hypervolume of the dominated space. The authors used a multiobjective Newton method.

13.5 Implementation of Multiobjective Memetic Algorithms

When we implement a multiobjective memetic algorithm for a particular MOO problem, we have a large number of options in its design. This means that we have a number of implementation issues to be taken into account. Typical issues are as follows: choice of a population-based multiobjective global search algorithm, choice

of a local search algorithm, timing of local search, selection of starting points for local search, and allocation of the available computation time to global search and local search. Each of these issues is briefly explained in the following:

1. The choice of a population-based multiobjective global search algorithm: This choice includes the related settings in the chosen global search algorithm (e.g., coding of solutions, genetic operators, parameter specifications, etc.). In early proposals of multiobjective memetic algorithms, evolutionary algorithms were mainly used for global search. This is because other population-based multiobjective algorithms were not popular in 1990s. Recently various multiobjective algorithms have been proposed based on different population-based global search mechanisms such as particle swarm optimization [134], ant colony optimization [214], and differential evolution [30, 769]. New types of multiobjective evolutionary algorithms have been also proposed based on estimation of distribution algorithms [957], indicator-based algorithms [64, 244, 967], and multiple aggregation functions [967]. As a result, we have a wide variety of options with respect to the global search part of multiobjective memetic algorithms.

2. The choice of a local search algorithm such as hill-climbing, simulated annealing and tabu search: This choice includes the related settings in the chosen local search algorithm (e.g., a generation mechanism of a neighboring solution, an acceptance criterion of neighbors, a termination condition of local search, etc.). The specification of an acceptance criterion is usually the same as the choice of a local search guiding mechanism. Problem-specific heuristics can be incorporated into local search, which usually improves the search ability of multiobjective memetic algorithms [413]. In the case of combinatorial optimization, generation mechanisms of neighbouring solutions in local search are usually similar to mutation operators in evolutionary algorithms. That is, new solutions are generated in a similar manner in local search and mutation whereas they have different acceptance criteria. On the other hand, different mechanisms are often used to generate new solutions in local search and mutation when multiobjective memetic algorithms are designed for continuous optimization. This is because the gradient information of objective functions is often used in local search to find better solutions whereas mutation usually modifies a part of the current solution randomly.

3. Timing of local search: Local search can be combined with a population-based multiobjective global search algorithm in various manners with respect to the timing of local search. Usually local search is invoked at every generation of a population-based multiobjective global search algorithm. In this case, local search starts from an offspring in global search. That is, global search can be viewed as providing local search with good starting points. Then the improved solutions by local search are used as parents in global search. That is, local search can be viewed as providing global search with good parents. In this manner, a population of solutions is improved by alternately using global search and local search. Local search is not necessarily to be used at every generation. For example, it can be used at every 10 generations or every 100 generations. One extreme case is the use of local search only before global search. In this case, local search can be viewed as generating a good initial population for global search. The basic idea behind this implementation is ”better

solutions may be obtained by recombining good locally-optimal solutions". Another extreme case is the use of local search only after global search. In this case, global search can be viewed as generating good starting points for local search. In other words, local search can be viewed as being used for the final improvement of global search results. The basic idea behind this implementation is "the local search ability of population-based algorithms is not high" and "better solutions may exist in the vicinity of good solutions".

4. Choice of staring points for local search: When local search is applied, starting points should be chosen from the current (or offspring) population. One naive implementation is to apply local search to all solutions in the current population. Another implementation is to apply local search to each solution probabilistically. Of course, other mechanisms can be implemented to choose starting points for local search such as the choice of only a small number of very good solutions with respect to some local search guiding mechanisms and the application of local search only to non-dominated solutions.

5. Allocation of available computation time to global search and local search: In single-objective memetic algorithms, many more solutions are usually examined by local search than population-based global search. This is not a bad strategy because the goal of single-objective optimization is usually to find a single optimal solution. In the case of multiobjective optimization, however, it is not a good strategy to spend too much computation time on local search for a specific direction even if some Pareto optimal solutions can be found by local search. This is because the goal of multiobjective optimization is not to find some Pareto optimal solutions but to approximate the entire Pareto front. We need to search for Pareto optimal solutions in various directions. Thus we should not spend too much computation time on local search for a specific direction. As a result, it is very important to allocate available computation time to global search and local search [411]. Moreover the computation time for local search should be appropriately reallocated to various local search directions.

As we have already explained in a previous subsection, landscape analysis is very useful in the design of efficient multiobjective memetic algorithms. For example, if a multiobjective problem has many local optima where local search is trapped, it may be a better idea to shallowly examine only a few neighbors of many starting points rather than to deeply examine many neighbors of a few starting points. However, the landscape of real-world multiobjective problems is often unknown. In that case, it is important to understand characteristics of each component of multiobjective memetic algorithms. For example, if a local search algorithm with high search ability towards the Pareto front is available, it may be a good idea to use a population-based global search algorithm with high diversity maintenance ability. On the other hand, if we use a population-based global search algorithm with high convergence property towards a part of the Pareto front, the diversity improvement by local search is important. The point is to fully utilize the synergy effect of using both global search and local search.

13.6 Conclusions

Multiobjective memetic algorithms constitute a very promising class of multiobjective metaheuristics. In many experiments they proved their efficiency for both combinatorial and continuous MOO problems.

Despite the need for new efficient memetic methods and the need for further applications, a number of other important directions for further research could suggested:

- The use of landscapes analysis in the design of recombination operators or the whole methods. Despite of some promising preliminary results discussed above, we are far from full understanding of the influence of landscapes of MOO problems on the performance of multiobjective memetic algorithms.
- The use of Pareto local search in multiobjective memetic algorithms. PLS has recently proved [542] to be a powerful technique for some multiobjective combinatorial optimization problems, being able to compete with memetic algorithms based on standard local search. PLS, however, becomes prohibitively inefficient with increasing number of objectives. Thus, hybridization with some global search techniques seems to be a promising approach.
- The use of advanced local search techniques, e.g. candidate moves, in MOO. Such techniques may have crucial influence on the performance of the local search component, and thus on the performance of the whole multiobjective memetic algorithm.
- Hybridization of other population-based algorithms, e.g. ant colony optimization, particle swarm optimization, differential evolution, with local search in MOO. Such algorithms may provide alternative global search components often competitive to evolutionary algorithms.
- Handling of many objectives in multiobjective memetic algorithms. Since, in general, the size of the Pareto front and the time needed to approximate it grows fast with the increasing number of objectives, interactive approaches seem to be a promising direction. In this case, some partial preference information may be used to focus the search on the desired regions of the Pareto front.