A Comparison of Diverse Approaches to Memetic Multiobjective Combinatorial Optimization

Joshua D. Knowles and David W. Corne
Department of Computer Science, University of Reading, UK
J.D.Knowles@reading.ac.uk, D.W.Corne@reading.ac.uk
FAX: +44(0) 118 975 1994, TEL: +44 (0) 118 931 8983
http://www.rdg.ac.uk/~ssr97jdk

Abstract

Memetic algorithms (MAs) are, at present, amongst the most successful approximate methods for combinatorial optimization. Recently, their range of application in this domain has been extended, with the introduction of several MAs for problems possessing multiple objectives. In this paper, we consider two of the newest of these MAs, the random directions multiple objective genetic local searcher (RD-MOGLS) of Jaszkiewicz, and the memetic Pareto archived evolution strategy (M-PAES), recently introduced by us. The two algorithms work in different ways: M-PAES employs a form of Pareto ranking in its selection mechanism, as used in several multiobjective evolutionary algorithms (MOEAs); whereas RD-MOGLS uses randomly weighted utility functions to judge solution quality, drawing from multiobjective tabu search and simulated annealing approaches. These two different approaches to memetic multiobjective optimization are briefly described, and their possible strengths and weaknesses identified. Finally, the two algorithms are applied to the multiobjective 0/1 knapsack problem. Their performance is compared on nine instances of the problem, using statistical methods developed previously.

1 Introduction

The general (unconstrained) multiobjective combinatorial optimization (MOCO) problem can be expressed as:

$$\begin{array}{ll} \text{“maximize”} & \mathbf{f}(\mathbf{x}) = (f_1(\mathbf{x}), \ldots, f_k(\mathbf{x})) \\ \text{subject to} & \mathbf{x} \in X \end{array} \tag{1}$$

where $\mathbf{x}$ is a discrete solution vector, and X is a finite set of feasible solutions. The objective function $\mathbf{f}(\mathbf{x})$ maps X into $\Re^k$, where $k \geq 2$ is the number of objectives. The term maximize appears in quotation marks because, in general, there does not exist a single solution that is maximal on all objectives. Therefore, one may seek to find a set of solutions $X^* \subseteq X$, the Pareto optimal set, with the property that :

$$\begin{array}{c} \forall\, \mathbf{x}^* \in X^* \; \nexists \mathbf{x} \in X \text{ such that } \mathbf{x} \succ \mathbf{x}^* \\ \text{where } \mathbf{x} \succ \mathbf{x}^* \text{ iff } \forall i \in \{1, \ldots, k\}\ f(x_i) \geq f(x_i^*) \\ \wedge\ \exists\, i \in \{1, \ldots, k\} : f(x_i) > f(x_i^*) \end{array} \tag{2}$$

where $x \succ x^*$ is read as x *dominates* x^*, and solutions in the Pareto optimal set are also known as efficient or admissible solutions.

Many single-objective combinatorial optimization problems that can be solved in polynomial time, become $\mathcal{NP}$-hard when formulated as corresponding MOCO problems [15], e.g. the assignment problem, minimum spanning tree etc. This makes the need for good approximation methods for MOCO especially important. Thus, memetic algorithms (MAs), which are amongst the most successful metaheuristics in combinatorial optimization, would seem a natural choice for this domain.

In recent years, there has already been growing interest in good approximate methods for MOCO. The approaches taken so far can be roughly split into two families: The local search methods, including tabu search and simulated annealing, e.g. [3, 5, 14], and the population-based multiobjective evolutionary algorithms (MOEAs) [2]. In either case, a key issue is how solution quality is judged in order to direct the search. Generally, one of two methods is employed: Pareto ranking of solutions, or achievement scalarizing (utility) functions. The local search methods proposed to date have used, almost exclusively, scalarizing functions, whereas MOEAs have mainly used Pareto ranking. In order to design a MA for MOCO, one must also choose which of these paradigms would be most appropriate. In the next section we examine each of them in turn.

2 Methods

Pareto ranking uses the notion of dominance, as used in Equation 2, to estimate solution quality. That is, solutions are compared one against the other, and each is judged based on whether it dominates, or is dominated by, other solutions. This method is well suited to assigning mating opportunities (fitness) to the members of a whole population of solutions, because all directions of the search are implicitly and simultaneously accounted for, in such a process. This efficiency at dealing with populations of solutions has led to its widespread use in MOEAs. However, a problem arises when Pareto ranking is used in local search where only two solutions, the current and candidate (mutant) solution, are compared at each step. Frequently, pairs of solutions $(\mathbf{x}, \mathbf{x}' \in X)$ will be *nondominated* with respect to each other:

$$\mathbf{x} \sim \mathbf{x}' \quad \text{iff} \quad \mathbf{x} \not\succ \mathbf{x}' \wedge \mathbf{x}' \not\succ \mathbf{x} \tag{3}$$

That is, neither is better than the other on all objectives. Thus, it is not possible to judge which of the current and mutant solutions is better, and much of the selection pressure is lost. However, if a *comparison set* of the best (non-dominated) solutions found during the search is maintained, and used to aid in judging solution quality in these undecidable cases, then a very high selection pressure can result. This is the basis of the local search method, (1+1)-PAES, introduced by Knowles and Corne [10, 11]. The PAES technique has the advantage that one particular search direction is not favoured in the local search, so solutions spread across the whole Pareto front can be found from a single search. Pseudocode for a very simple, PAES-like local searcher is given in Figure 1. In fact, in (1+1)-PAES, efficiency is ensured by storing only a limited number of nondominated solutions in P, and a technique for encouraging diversity is also employed.

In contrast to Pareto ranking, achievement scalarizing functions work by aggregating the vector of objective scores into a scalar measure of quality. For a detailed account see [16]. In many cases, a simple linear blend of the objectives is achieved by taking the inner product of the vector of objective values, and a vector of weights $\boldsymbol{\lambda}$ having the following properties:

Initialize: $P = \emptyset$
  Generate $\mathbf{x} \in X$ randomly
  $P \longleftarrow P \cup \mathbf{x}$
Main Loop: $\mathbf{x}' \in X \longleftarrow$ mutate($\mathbf{x}$)
  if ($\mathbf{x}' \succ \mathbf{x}$)
    $P \longleftarrow P \cup \mathbf{x}'$, $\mathbf{x} \longleftarrow \mathbf{x}'$
  else if ($\mathbf{x} \sim \mathbf{x}'$){
    $i = 0$, *replace* $=$ TRUE
    while ($i < |P|$){
      if ($\mathbf{x}' \succ \mathbf{x_i} \in P$)
        $P \longleftarrow P \cup \mathbf{x}', P \longleftarrow P \setminus \mathbf{x_i}$
      else if ($\mathbf{x_i} \succ \mathbf{x}'$)
        replace $=$ FALSE, $i = |P|$
      $i++$ }
    if(*replace* $==$ TRUE)
      $\mathbf{x} \longleftarrow \mathbf{x}'$}

Figure 1: PAES-like Pareto ranking local search. P is the set of nondominated solutions.

$$\boldsymbol{\lambda} = [\lambda_1, \ldots \lambda_k], \ \sum_{i=0}^{k} \lambda_i = 1, \ \forall i \ \lambda_i \geq 0 \tag{4}$$

In order to approximate the whole Pareto front, a dispersed set of weight vectors must be used to direct the search. A maximally dispersed set can be constructed using a technique described in [1]. The advantages of the scalarizing technique include computational parsimony, and the fact that solution quality is judged independently from other solutions. The latter makes it a good candidate for use in a parallel memetic algorithm. However, it has disadvantages: When a population of solutions is available and mating opportunities are to be allocated, it is difficult to use scalarizing functions. Often, the method of assigning fitness based on a randomly selected weight vector is used, but this does not seem to be a good way of making maximum use of the current solutions available.

Which of the two methods of assessing solution quality, described above, is more effective, is still an open question. In [7], Jaszkiewicz argues that MOEAs that use Pareto ranking are not ideal for MOCO because they must use niching methods to ensure solution diversity, and at their heart they lack an objective function to drive the search. The argument is not supported by reference to any empirical or theoretical evidence, showing that MOEAs using Pareto ranking actually perform poorly compared to any other method, however. In the same paper, Jaszkiewicz states that the use of Pareto ranking is not suited to local search, either, reasoning that selection pressure cannot be maintained. These aspersions of Pareto ranking may yet be vindicated, but the use of linear scalarizing functions can also be criticized. Theoretically, they cannot be used to find non-supported efficient solutions, i.e. solutions lying in non-convex portions of the true Pareto front. Of course, this problem can be overcome with the use of non-linear Tchebycheff utility functions [16], but these do not perform as well as linear functions in some cases, e.g. see [7]. And empirically, where linear weighting methods have been used in MOEAs, they have not been successful when compared to other methods [17]. Nonetheless, until multiobjective tabu search and simulated annealing methods, employing linear aggregation, are compared directly with the best MOEAs, one should not be too dogmatic about which approach is more efficient.

Up to now, the MAs proposed for MOCO [6, 7] have used achievement scalarizing functions because the local search phases in them are based on the vast majority of local search methods which use this technique. However, we have introduced a new MA for MOCO, called the memetic Pareto archived evolution strategy (M-PAES), which is based on (1+1)-PAES. In a recent paper [12] we have demonstrated that it is competitive with one of the best MOEAs, the strength Pareto evolutionary algorithm (SPEA) of Zitzler and Thiele [18]. In [12], detailed pseudocode of the M-PAES algorithm is also given.

In this paper, M-PAES is compared with a memetic algorithm that employs achievement scalarizing functions; the RD-MOGLS algorithm of Jaszkiewicz, which is fully described in [7]. The comparison is a first step towards understanding the different performance characteristics that may be achieved from these two different approaches to directing multiobjective searches in MAs.

3 Experiments

We use for testing, a well-known MOCO problem, the multiobjective 0/1 knapsack problem, defined thus[1]: Given a set of n items and a set of k knapsacks, with

$$\begin{aligned} p_{i,j} &= \text{profit of item } j \text{ according to knapsack } i, \\ w_{i,j} &= \text{weight of item } j \text{ according to knapsack } i, \\ c_i &= \text{capacity of knapsack } i, \end{aligned}$$

find a vector $\mathbf{x} = (x_1, x_2, \ldots, x_n) \in \{0,1\}^n$, such that the capacity constraints

$$e_i(\mathbf{x}) = \sum_{j=1}^{n} w_{i,j} \cdot x_j \leq c_i \quad (1 \leq i \leq k) \tag{5}$$

[1]The definition is taken directly from [17]

are satisfied and for which $\mathbf{f}(\mathbf{x}) = (f_1(\mathbf{x}), \ldots, f_k(\mathbf{x}))$ is maximum, where

$$f_i(\mathbf{x}) = \sum_{j=1}^{n} p_{i,j} \cdot x_j \tag{6}$$

and $x_j = 1$ if and only if item j is selected.

The instances of the problem that we use here, are taken from a recent paper [18] by Zitzler and Thiele (ZT), where they are described fully. There are nine problems altogether, of differing combinations of size (number of items), and number of objectives (knapsacks). At the time of writing, the problems are available from an Internet web-site[2].

Comparison is made between three algorithms: M-PAES, RD-MOGLS, and (1+1)-PAES. (The latter is used as a baseline.) Setting the parameters of the three algorithms to provide a fair comparison, is not easy. As in [12], our approach is to use a 'good' set of parameters, found empirically, for each algorithm. It is not possible, or even desirable, to use exactly the same parameter settings for each algorithm, as they differ considerably. However, we do hold a number of core parameters/conditions constant across the three algorithms. These are: The neighbourhood operator, which is a random bit-flip mutation applied with a probability of $4/n$, where n is the length of the chromosome; the constraint handling, which is carried out in the same way as described in [18]; the stopping criterion, which is the maximum number of evaluations, max_evals, set for each problem size as described in [12]; and the recombination operator used by both RD-MOGLS and M-PAES, which is two-parent uniform crossover. For M-PAES and (1+1)-PAES, the other parameters are set exactly as described in [12].

To help understand the other parameters that must be set in RD-MOGLS, the algorithm is described in outline: An initial population of S solutions is generated, one by one, by selecting a utility function at random, and starting from a randomly generated solution vector, optimizing locally. The initial population are placed in a set, CS, of the current solutions. A set of potentially efficient solutions, PE, (initially empty) is also maintained, and this is updated with the initial population of solutions, as well. During the main loop of the algorithm, a utility function is selected at random. The best N solutions in the current population, CS, on the utility function, are then copied to a temporary population, TP. The following steps are then repeated N times: Two solutions in TP are selected at random and recombined to generate an offspring;

[2]http:/www.tik.ee.ethz.ch/∼zitzler

then the offspring is optimized locally on the utility function; the offspring is added to CS, and it is also added to TP, replacing the worst member; finally, the set PE is updated with the offspring.

initial population size S:	100
temporary population size N:	20
efficient solution set size $\|PE\|$:	100
local search fails *l_fails*:	5

Figure 2: Parameter settings for RD-MOGLS.

The parameters chosen for RD-MOGLS are given in Figure 2. The initial and temporary population sizes were derived empirically from a few test runs of RD-MOGLS. The size of PE was set equal to the equivalent nondominated *archive*, used in M-PAES and (1+1)-PAES. Finally, there must also be a stopping criterion for each of the local search phases. This detail is omitted in [7], so we choose to end each local search phase when *l_fails* consecutive moves do not improve the solution. This is similar to the method used in M-PAES. Once again, *l_fails* was set empirically.

4 Results

4.1 Performance Metrics

As in previous research, we measure the performance of the algorithms tested, using a statistical comparative assessment technique adapted from [4]. We refer the reader to [8, 11] for a complete description of our implementation of the technique and a discussion of its advantages and disadvantages.

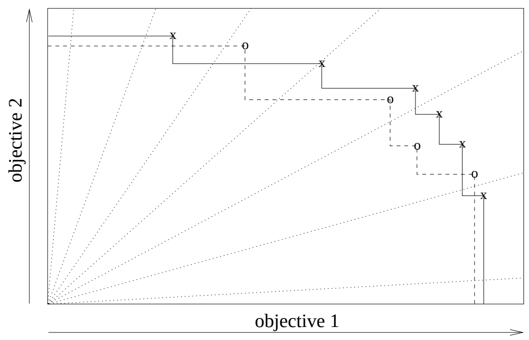

Figure 3: A collection of two sets of nondominated vectors. Each set of vectors defines an *attainment surface*, dividing the objective space. The attainment surfaces can be sampled using a number of angled sampling lines, as shown.

Here, the reader need only understand the following points. 1. Any set of objective vectors can be viewed as defining a surface (an *attainment surface*) in objective space, dividing the space into a dominated region and a nondominated region (Figure 3). 2. A collection of runs of an algorithm will thus generate a collection of such surfaces. 3. The collection of attainment surfaces can be sampled at various points using lines, angled in the direction of increasing value in each objective, that intersect the surfaces (Figure 3). 4. The intersection points along each of sampling lines gives a univariate distribution which can be analysed using standard non-parametric statistical tests. 5. When the collections of attainment surfaces come from different algorithms, statistical inferences as to which algorithm's distribution of surfaces is 'better', along each sample line, can be made. Using these points, we are able to input *collections* of individual runs from each of a pair of algorithms, and from these, present results in two ways: First, as a pair of numbers [a,b] indicating, respectively, the percentage of the space where algorithm A outperforms algorithm B, and the percentage of the space where algorithm B outperforms algorithm A. The percentages returned are the result of a set of Mann-Whitney U tests [13] performed on each sample line on the collections of data, at a given confidence level (we choose 95%). Second, for two-objective problems, we can plot surfaces representing the median, best, or worst surface that the algorithm returns. Here, we plot only the median surface of the three algorithms tested, on the two-objective problems. In all cases approximately 500 lines were used to sample the attainment surfaces.

4.2 Analysis

Each algorithm was run 30 times on each problem. The set of nondominated solutions found in each run was used to give a statistical indication of performance, using the techniques referred to above. The median surfaces plotted for the two-objective problems are shown in Figure 4. A number of observations can be made from these plots. In all three plots the median surface generated by RD-MOGLS extends beyond the surface generated by M-PAES in at least one of the objectives. However, the M-PAES algorithm generates a median surface that dominates the surface of RD-MOGLS over a larger portion of the tradeoff front. With increasing problem size (items), the M-PAES algorithm's performance improves relative to RD-MOGLS. Both M-PAES and the baseline algorithm (1+1)-PAES give smoother, more convex median surfaces than RD-MOGLS.

The statistical results for all the problems are summarised in Table 5. Comparison is made between M-PAES and RD-MOGLS only. Thus, the statistic [65.4,

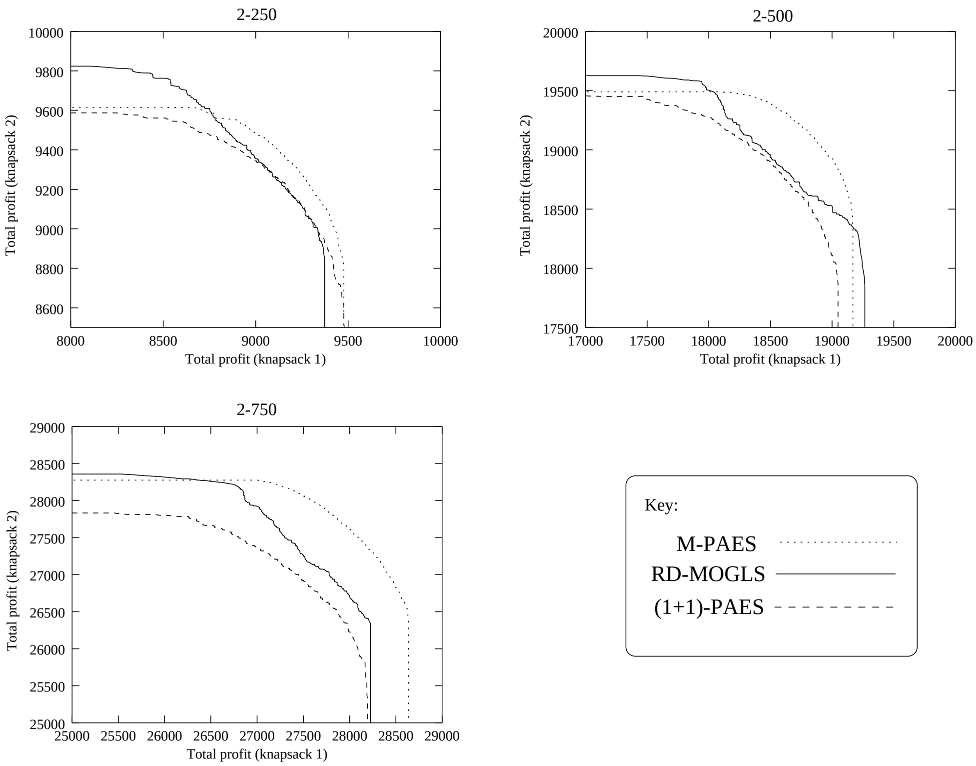

Figure 4: Median surfaces calculated from 30 runs of the three algorithms, M-PAES, RD-MOGLS, and (1+1)-PAES, on the 2-objective knapsack problems.

Knapsacks	Items		
	250	500	750
2	[65.4, 28.2]	[49.4, 0]	[61.5, 0]
3	[72.7, 25.1]	[89.8, 0]	[100, 0]
4	[0, 0]	[0, 0]	[0, 0]

Figure 5: The results of testing M-PAES against RD-MOGLS using our statistical techniques [11] and 30 independent runs of each algorithm. The knapsack problems have 2, 3 or 4 objectives and 250, 500 or 750 items, as indicated.

28.2] in the upper left entry in the table means that M-PAES gives a better distribution of surfaces than RD-MOGLS over 65.4% of the tradeoff front, and vice-versa, RD-MOGLS gives a better distribution than M-PAES on 28.2%, on the 250 item, 2 knapsack problem. Again, several observations from these results can be made: The first row of the table verifies that M-PAES performs well on the two-objective problems, as suggested by the plots in Figure 4. Its relative performance increases as the number of items increases. This is true, not only on the 2-objective problem but also the 3-objective problems. However, as the number of objectives increases, the performance of RD-MOGLS increases relative to M-PAES, so that there is no statistical difference between the non-dominated surfaces generated by M-PAES and RD-MOGLS on the 4-objective problems.

5 Summary and conclusion

Multiobjective combinatorial optimization (MOCO) problems often demand the application of approximate methods for their solution. So, memetic algorithms, which have proved to be very effective on a number of single-objective combinatorial problems, may offer an ideal approach to MOCO. However, dealing with multiple objectives is not straight forward: One must decide how to guide selection, given a vector of objective values. In this paper two different approaches to solving this problem, within a memetic algorithm framework, were outlined. One, M-PAES, uses the method of Pareto ranking of solutions, while the other, RD-MOGLS, employs random utility functions to linearize the objective vector. The two approaches were applied to a well-known MOCO problem, the multiobjective 0/1 knapsack problem. Both algorithms work well, generating results that are better than the baseline local search algorithm, (1+1)-PAES, produced. However, on the small number of experiments carried out, M-PAES, was found to be superior overall, using a sophisticated statistical measure of the solution sets discovered. Nonetheless, RD-MOGLS, seemed capable of generating solutions over a wider range in each of the objectives, than M-PAES, and on the four-objective problems there was no discernible difference in performance between the two MAs. Clearly, further investigation of both approaches is needed; this study merely begins a process of development and comparison of MAs for MOCO. Further work will be aimed at continuing this investigation on a number of other MOCO problems. In addition to the quality of solutions generated, the computational cost and ease of parallelization will also be considered.

Acknowledgments

The first author would like to thank BT Labs Plc. for financial support during his PhD.

References

[1] P. C. Borges and M. Pilegaard Hansen. A basis for future successes in multiobjective combinatorial optmization. Technical Report IMM-REP-1998-8, Institute of Mathematical Modelling, Technical University of Denmark, 2800 Lyngby, Denmark, 1998.

[2] C. A. Coello Coello. A Comprehensive Survey of Evolutionary-Based Multiobjective Optimization Techniques. *Knowledge and Information Systems. An International Journal*, 1(3):269–308, August 1999.

[3] P. Czyżak and A. Jaszkiewicz. Pareto simulated annealing - a metaheuristic technique for multiple-objective combinatorial optimization. *Journal of Multi-Criteria Decision Analysis*, 7(1):34–47, January 1998.

[4] C. M. Fonseca and P. J. Fleming. Nonlinear System Identification with Multiobjective Genetic Algorithms. In *Proceedings of the 13th World Congress of the International Federation of Automatic Control*, pages 187–192, San Francisco, California, 1996. Pergamon Press.

[5] X. Gandibleux, N. Mezdaoui, and A. Fréville. A tabu Search Procedure to Solve Multiobjective Combinatorial Optimization Problems. In R. Caballero and R. Steuer, editors, *Proceedings volume of MOPGP'96*, pages 291–300, Berlin, 1996. Springer-Verlag.

[6] H. Ishibuchi and T. Murata. Multi-Objective Genetic Local Search Algorithm. In T. Fukuda and T. Furuhashi, editors, *Proceedings of the 1996 International Conference on Evolutionary Computation*, pages 119–124, Nagoya, Japan, 1996. IEEE.

[7] A. Jaszkiewicz. Genetic local search for multiple objective combinatorial optimization. Technical Report RA-014/98, Institute of Computing Science, Poznan University of Technology, December 1998.

[8] J. D. Knowles and D. W. Corne. Assessing the Performance of the Pareto Archived Evolution Strategy. In A. S. Wu, editor, *Proceedings of the 1999 Genetic and Evolutionary Computation Conference. Workshop Program*, pages 123–124, San Francisco, CA, July 1999. Morgan Kaufmann.

[9] J. D. Knowles and D. W. Corne. Local Search, Multiobjective Optimization and the Pareto Archived Evolution Strategy. In B. McKay, editor, *Proceedings of The Third Australia-Japan Joint Workshop on Intelligent and Evolutionary Systems*, pages 209–216, Ashikaga, Japan, November 1999. Ashikaga Institute of Technology.

[10] J. D. Knowles and D. W. Corne. The Pareto Archived Evolution Strategy: A New Baseline Algorithm for Multiobjective Optimisation. In *1999 Congress on Evolutionary Computation*, pages 98–105, Piscataway, NJ, July 1999. IEEE Service Center.

[11] J. D. Knowles and D. W. Corne. Approximating the Nondominated Front Using the Pareto Archived Evolution Strategy. *Evolutionary Computation*, 8(2):149–172, 2000.

[12] J. D. Knowles and D. W. Corne. M-PAES: A Memetic Algorithm for Multiobjective Optimization. In *Proceedings of the 2000 Congress on Evolutionary Computation (CEC 2000)*, Piscataway, NJ, 2000. IEEE. (To appear).

[13] W. Mendenhall and R. J. Beaver. *Introduction to Probability and Statistics - 9th edition.* Duxbury Press, International Thomson Publishing, Pacific Grove, CA, 1994.

[14] M. Pilegaard Hansen. Tabu Search in Multiobjective Optimisation : MOTS. In T. Stewart and R. van den Honert, editors, *Proceedings of 13th International Conference on Multiple Criteria Decision Making*, Berlin, 1996. Springer-Verlag.

[15] E. L. Ulungu and J. Teghem. Multi-objective Combinatorial Optimization Problems: A Survey. *Journal of Multi-Criteria Decision Analysis*, 3:83–104, 1994.

[16] A. P. Wierzbicki. The use of reference objectives in multiobjective optimization. In G. Fandel and T. Gal, editors, *MCDM Theory and Application, Proceedings Hagen/Konigswinter 1979*, Lecture Notes in Economics and Mathematical Systems 177, pages 468–486, Berlin, 1980. Springer-Verlag.

[17] E. Zitzler. *Evolutionary Algorithms for Multiobjective Optimization: Methods and Applications.* PhD thesis, Swiss Federal Institute of Technology (ETH), Zurich, Switzerland, November 1999.

[18] E. Zitzler and L. Thiele. Multiobjective Evolutionary Algorithms: A Comparative Case Study and the Strength Pareto Approach. *IEEE Transactions on Evolutionary Computation*, 3(4):257–271, November 1999.