DEPARTMENT OF OPERATIONS RESEARCH
UNIVERSITY OF AARHUS

Working Paper no. 2001/1

Solving Biobjective Combinatorial Max-Ordering Problems by Ranking Methods and a Two-Phases Approach

Matthias Ehrgott
Anders J.V. Skriver

ISSN 1600-8987

Solving Biobjective Combinatorial Max-Ordering Problems by Ranking Methods and a Two-Phases Approach

MATTHIAS EHRGOTT*
Department of Engineering Science
University of Auckland
Private Bag 92019
Auckland
New Zealand

ANDERS J.V. SKRIVER
Department of Operations Research
University of Aarhus, Building 530
Ny Munkegade
DK - 8000 Århus C
Denmark

March 2, 2001

Abstract

In this paper we propose a new method to solve biobjective combinatorial optimization problems of the max-ordering type. The method is based on the two-phases method and ranking algorithms to efficiently construct K best solutions for the underlying (single objective) combinatorial problem. We show that the method overcomes some of the difficulties of procedures proposed earlier. We illustrate this by an example and discuss the difficulties in extending it to more than two objectives.

Keywords: MCDM, biobjective optimization, max-ordering problems, ranking methods, combinatorial optimization.

1 Introduction

Max-ordering (MO) problems are multicriteria optimization problems in which the goal is to minimize the worst of several objective functions. They can be formulated as follows.

$$\min_{x \in S} \max_{i=1,\dots,Q} f_i(x), \tag{1}$$

where $f_i(x)$ denotes the objective functions of the problem. The problem is denoted max-ordering instead of min-max in order not to confuse terminology with single objective problems, i.e. $\min_{x \in S} \max_{e \in x} w_e$, which finds solutions where the largest weight is minimal, e.g. the path where the largest edge-weight is minimal. Max-ordering problems arise in various applications, see Rana and Vickson [23] or Warburton [29], and as subproblems in interactive methods for the solution of multicriteria optimization problems such as the

*Corresponding author. Email: m.ehrgott@auckland.ac.nz

GUESS method (Buchanan [3]), STEM (Benayoun et al. [2]), and the interactive weighted Tchebycheff method (Steuer and Choo [25]).

In this paper we consider max-ordering problems in a combinatorial context, i.e. we assume that S is a finite set, e.g. the set of paths between two nodes of a network, or the set of spanning trees of a graph.

There is a number of previous research papers on this topic (Ehrgott [5], Hamacher and Ruhe [15], Murthy and Her [21], Ehrgott et al. [9]). See also Ehrgott and Gandibleux [8] for more references. Various authors observed that, even in the bicriteria case, max-ordering problems are usually $\mathcal{NP}$-complete. The methods proposed for their solution include branch and bound (Rana and Vickson [23]), labeling algorithms (for shortest path problems, Murthy and Her [21]) and ranking methods (Ehrgott [5], Hamacher and Ruhe [15]), that is the application of algorithms to find K best solutions of (single objective) combinatorial problems.

We also propose methods involving ranking algorithms actually overcoming the main problem of the method proposed in Hamacher and Ruhe [15], at least for the case of two objectives, see the discussion after Algorithm 1. Our method also overcomes a weakness of the method proposed in Murthy and Her [21], see Section 4. We combine the ranking method with the two-phases method originally developed for the determination of all Pareto optimal solutions of bicriteria combinatorial optimization problems, Ulungu and Teghem [27], and so far, successfully applied to a number of such problems. We mention Ehrgott [6], Lee and Pulat [18] for network flow, Ulungu and Teghem [26] and Visée et al. [28] for knapsack, Ulungu and Teghem [27] for assignment, and Ramos et al. [22] for spanning tree problems.

2 Basic Results

In this section we introduce some notation for multicriteria (combinatorial) optimization and we prove some basic results which will justify the correctness of our method.

Consider a multicriteria optimization problem

$$\min_{x \in S} \{f_1(x), \ldots, f_Q(x)\}.$$

We use the notation $f(x) = (f_1(x), \ldots, f_Q(x))$ for the vector of objective functions. A feasible solution x^* is called Pareto optimal, if there is no $x \in S$ such that $f(x) \leq f(x^*)$ and $f(x) \neq f(x^*)$, where $\leq$ is understood component-wise. The set of Pareto optimal solutions of S is denoted $Par(S)$. If x^* is Pareto optimal, $f(x^*)$ is called efficient.

In multiobjective combinatorial optimization, Pareto optimal solutions can be classified into supported and unsupported Pareto optimal solutions. The former are those x^* for which there exists a weighting vector $\lambda = (\lambda_1, \ldots, \lambda_Q)$ such that

$$f(x^*) = \min_{x \in S} \sum_{i=1}^{Q} \lambda_i f_i(x).$$

The existence of unsupported Pareto optimal solutions is a characteristic property of multiobjective combinatorial optimization problems.
We shall also use the notation $g(x) = \max_{i=1,\ldots,Q} f_i(x)$ for the max-ordering objective value of a feasible solution $x \in S$. With these definitions we are ready to prove some basic results. The first one is wellknown, see e.g. Hamacher and Ruhe [15]. We state the proof for completeness.

Lemma 1 *There is at least one optimal solution of the max-ordering problem* $\min_{x \in S} g(x)$ *which is Pareto optimal.*

Proof : Suppose x^* is an optimal solution of the max-ordering problem, but is not Pareto optimal. Since S is finite, there must then exist a feasible solution $x \in S$ dominating x^*, i.e. such that $f_i(x) \leq f_i(x^*)$ for $i = 1, \ldots, Q$ with one strict inequality. Because $g(x) \leq g(x^*)$, it follows that x also solves the max-ordering problem optimally. ■

The next Lemma is specifically stated for two objectives. It formalizes the argument that the maximum of two functions is minimal, if the objective values are as equal as possible. Its proof is immediate from the definition of the max-ordering problem and Lemma 1.

Lemma 2 *Let* $Par(S) = \{x_1, \ldots, x_p\}$ *be the set of Pareto optimal solutions of a bicriteria combinatorial optimization problem. Assume that* $f_1(x_i) \leq f_1(x_{i+1})$ *and* $f_2(x_i) \geq f_2(x_{i+1})$ *for* $1 = 1, \ldots, p-1$ *and define* $K := \min\{i : f_2(x_i) < f_1(x_i)\}$. *Then the following hold.*

1. *If* $K = 1$, x_1 *solves the max-ordering problem.*

2. *If* $K = \infty$, x_p *solves the max-ordering problem.*

3. *Otherwise* x_K *or* x_{K-1} *(or both) solve the max-ordering problem.*

A special case occurs if there is a Pareto optimal solution with both objectives equal.

Lemma 3 *If there is a Pareto optimal solution such that* $f_1(x) = f_2(x)$ *then* x *also minimizes* $g(x)$.

These three lemmas state that we can restrict our search for a solution for a minimizer of $g(x)$ to Pareto optimal solutions, with their two objectives as equal as possible. In other words, Pareto optimal max-ordering solutions will be located close to the halving line $f_1 = f_2$ in criterion space. Lemma 2 suggests to rank Pareto optimal solutions according to increasing values of f_1 (or f_2). This strategy would, however, imply the generation of supported and unsupported Pareto optimal solutions. And with the desired max-ordering solutions expected to be centrally located in the Pareto set, we would expect to enumerate half of all Pareto optimal solutions, involving excessive computational effort. Taking the difficulty of generating unsupported solutions into account (see Ehrgott [7]), we propose a different approach.
Our algorithm makes use of the information of Lemmas 1 to 3 in a more intelligent way and proceeds in two phases.

3 The Algorithm

First, we look for the two supported Pareto optimal solutions for which $f_1(x_i) \leq f_2(x_i)$ and $f_1(x_j) > f_2(x_j)$, $j > i$, according to the order of Lemma 2. We shall call them x_1 and x_2 in the algorithm. To do so, we start with solutions x_1 and x_2 minimizing objectives f_1 and f_2, respectively. We then proceed to solutions where the difference of objective values is smaller. When this is no longer possible, we will either have one supported Pareto optimal solution with $f_1(x) = f_2(x)$, or we end up with two neighboring supported Pareto optimal solutions, say x_1 and x_2 such that $f_1(x_1) < f_2(x_1)$ and $f_1(x_2) > f_2(x_2)$. According to Lemma 3, the first case solves $\min_{x \in S} g(x)$, and any other Pareto optimal solution must have one objective value smaller and one bigger than x. Of course, it may happen that one of the objectives dominates the other completely, i.e. $\min_{x \in S} f_1(x) \geq \max_{x \in Par(S)} f_2(x)$ (cases 1 or 2 in Lemma 2). In this case the problem is trivial, and we can easily detect it when computing x_1 and x_2 for the first time.
Should we terminate Phase 1 with two solutions, we will have to investigate unsupported solutions in the right-angled triangle defined by the hyperplane through the point $f(x^*)$ with normal λ and $(g(x^*), g(x^*))$, where x^* is the current best solution, see Figure 2. For this we use the ranking algorithm. In fact, $f(x_1)$ and $f(x_2)$ uniquely define weights λ_1, λ_2 such that both $x_1$and x_2 are optimal solutions of

$$\min_{x \in S} \lambda_1 f_1(x) + \lambda_2 f_2(x).$$

We can now apply a ranking algorithm to find second, third, ... best solutions for this problem, in order to find unsupported solutions in the identified triangle. A similar pro-

cedure was proposed for the identification of all unsupported Pareto optimal solutions in Coutinho-Rodrigues et al. [4].
The algorithm will stop if we encounter a solution x with $f_1(x) = f_2(x)$, as this must be the optimal solution we are looking for, or $\lambda_1 f_1(x) + \lambda_2 f_2(x) \geq g(x^*)$, because any further solutions will no longer be in the triangle and therefore no longer a candidate for a MO optimal solution. In the latter case, the currently best solution is the optimal solution of the max-ordering problem.

Algorithm - Phase 1

1. *Solve* $\min_{x \in S} f_1(x)$, *let* x_1 *be the optimal solution and let* $f_1^1 := f_1(x_1), f_2^1 = f_2(x_1)$.
2. *If* $f_1^1 \geq f_2^1$ *STOP,* $x^* = x_1$ *is an optimal solution.*
3. *Solve* $\min_{x \in S} f_2(x)$, *let* x_2 *be the optimal solution and let* $f_1^2 := f_1(x_2), f_2^2 = f_2(x_2)$.
4. *If* $f_2^2 \geq f_1^2$ *STOP,* $x^* = x_2$ *is an optimal solution.*
5. *If* $f(x_1) = f(x_2)$ *STOP,* $x^* = x_1$ *(or* x_2*) is an optimal solution.*
6. *Let* $x^* := argmin\{g(x_1), g(x_2)\}$ *be the currently best solution.*
7. *Let* $\lambda_1 := f_2^1 - f_2^2, \lambda_2 := f_1^2 - f_1^1$.
8. *Solve* $\min_{x \in S} \lambda_1 f_1(x) + \lambda_2 f_2(x)$, *let* x_3 *be the optimal solution and let* $f_1^3 := f_1(x_3)$, $f_2^3 = f_2(x_3)$.
9. *If* $f_1^3 = f_2^3$ *STOP,* $x^* = x_3$ *is an optimal solution.*
10. *If* $x_3 = x_2$ *or* $x_3 = x_1$ *call Phase 2(*λ_1, λ_2*).*
11. *If* $f_1^3 < f_2^3$ *then* $x_1 = x_3, f_1^1 = f_1^3, f_2^1 = f_2^3$.
12. *If* $f_1^3 > f_2^3$ *then* $x_2 = x_3, f_1^2 = f_1^3, f_2^2 = f_2^3$.
13. *Go to 6.*

The idea of the first phase is illustrated in Figure 1. With solutions x_1 and x_2 we compute the normal to the line connecting (f_1^1, f_2^1) and (f_1^2, f_2^2). This normal serves as a weighting vector for combining the two objectives, and its negative is the direction in which we search for a new supported Pareto optimal solution which is eventually found at x_3 with objective values (f_1^3, f_2^3).

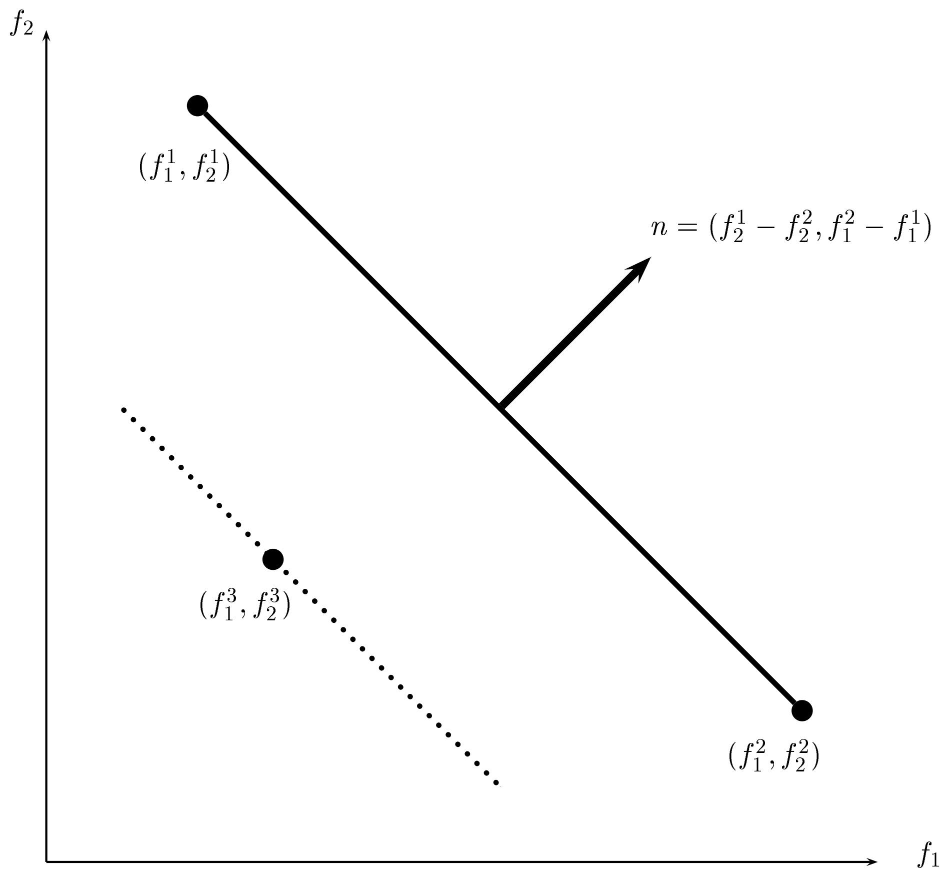

Figure 1: Illustration of Search Direction in Phase 1

We remark that the values λ_1, λ_2, identified at the end of Phase 1, are the best choice of λ in the method proposed by Hamacher and Ruhe [15] and will overcome the problem that for an unfortunate choice of λ, that method turns out to be complete enumeration of all feasible solutions.

Algorithm - Phase 2

1. *$K := 3$.*
2. *Use a K-best algorithm to find the K-best solution of $\min_{x \in S} \lambda_1 f_1(x) + \lambda_2 f_2(x)$. Denote this solution x^K.*
3. *If $\lambda_1 f_1(x^K) + \lambda_2 f_2(x^K) \geq g(x^*)$ STOP, x^* is an optimal solution.*
4. *If $f_1(x^K) = f_2(x^K)$ STOP, $x^* = x^K$ is an optimal solution.*
5. *If $f_1(x^K) > f_1^2$ then $K := K + 1$, go to 2.*

6. *If* $f_2(x^K) > f_2^1$ *then* $K := K + 1$, *go to 2.*

7. $K := K + 1$, *If* $g(x^K) < g(x^*)$ *then* $x^* := x^K$, *go to 2.*

We illustrate the algorithm on an example. In Figure 2 we show the objective values of 6 feasible points indexed in the order of their generation.

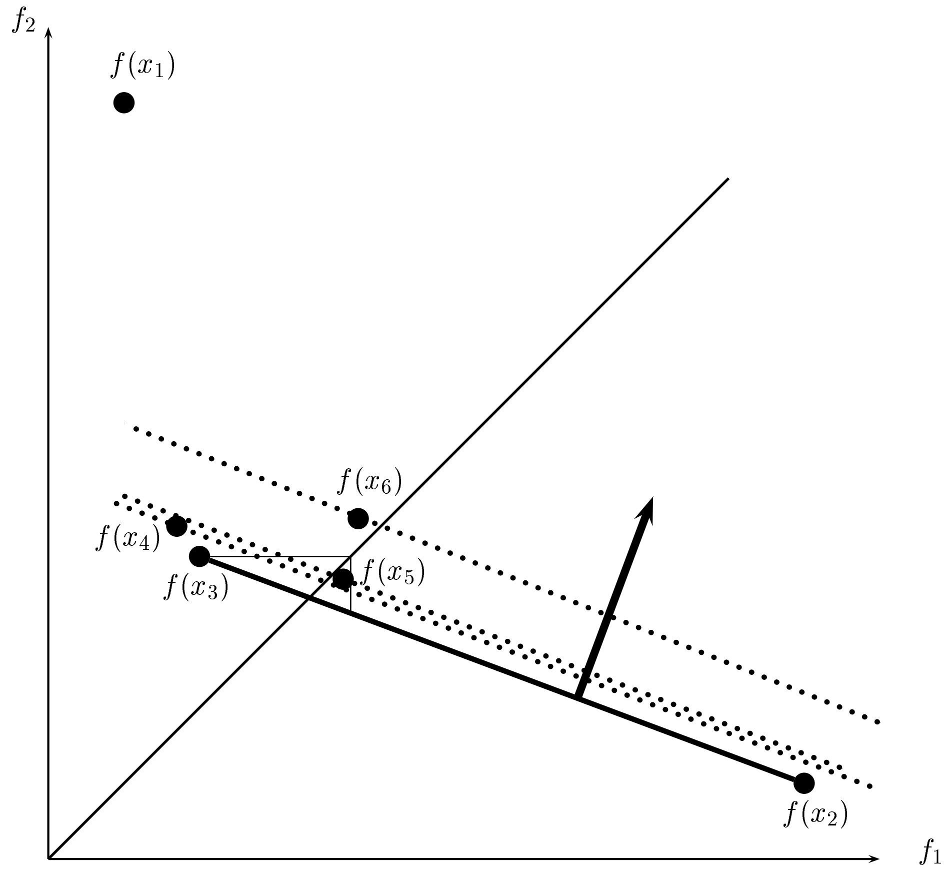

Figure 2: Illustrative Example

In Phase 1, x_1 and x_2 will be generated first. Weights λ_1 and λ_2 are computed corresponding to the normal to a line connecting $f(x_1)$ and $f(x_2)$ and $x^* = x_2$. Solution of the weighted sum problem in Step 8 results in x_3. Since $f_1(x_3) < f_2(x_3)$, f_1^1 and f_1^2 are replaced by the objective values of x_3. The current best x^* is updated to x_3. The second weighted sum problem uses updated λ's corresponding to the normal to the line connecting $f(x_2)$ and $f(x_3)$. Assume x_3 is returned as optimal solution. Thus no new supported Pareto optimal solution is found, and we continue with Phase 2 to investigate the earlier defined triangle. Note that the supported solution x_4 is not generated in Phase 1.

We know that x_3 and x_2 are first and second best solutions of the weighted sum problem, therefore we are searching for the third best by searching in direction λ. This turns out to be x_4, which is discarded as not being in the triangle ($f_2(x_4) > f_2(x_3) = g(x^*)$). So we set $K = 4$, identify x_5 as the next solution, which passes all tests. In our example x_5 replaces x_3 as the current best solution and K is set to 5. The next solution is x_6, the combined objective value of which is larger than that of the third corner point of the triangle. We will therefore find no further points in the triangle and stop with the optimal solution $x^* = x_5$.

Remark 1 *In Phase 2 the following situation might occur: The solution of the weighted sum problem is another supported Pareto optimal solution which is, as x_1 and x_2, optimal for the weighted sum problem. Its objective function vector lies on the line between $f(x_1)$ and $f(x_2)$. In this case, this point creates two new and smaller triangles. We can restrict search to the one which is intersected by the halving line $f_1 = f_2$.*

4 Lagrange Relaxation of Max-Ordering Problems

In this section we describe why Lagrange relaxation of max-ordering problems with linear objective functions does not work. This approach has earlier been suggested as a pruning method for a label correcting approach in Murthy and Her [21].
Consider the usual reformulation of (1)

$$\begin{array}{lrcll} \min & z & & & \\ \text{s.t.} & z & \geq & f_i(x) & \forall\, i = 1, \ldots, Q \\ & x & \in & S & \\ & z & \in & \mathbb{R}. & \end{array} \tag{2}$$

A Lagrange relaxation of the first set of constraints in (2) is an appealing thing to do, as it simplifies the constraints to the original ones. This leads to the following problem, where λ is the vector of Lagrange multipliers:

$$\begin{array}{lrcl} \min & z & + & \sum_{i=1}^{Q} \lambda_i (f_i(x) - z) \\ \text{s.t.} & x & \in & S \\ & \lambda & \geq & 0. \end{array}$$

Rearranging the objective function leads to

$$\min_{x \in S} \left(1 - \sum_i \lambda_i\right) z + \sum_{i=1}^{Q} \lambda_i f_i(x),$$

where $\sum_i \lambda_i = 1$ to avoid an unbounded problem (since $z \in \mathbb{R}$). We thus end up with the following simple problem

$$\begin{array}{lrcl} \min & \sum_{i=1}^{Q} \lambda_i f_i(x) & & \\ \text{s.t.} & x & \in & S \\ & \lambda & \geq & 0 \\ & \sum_{i=1}^{Q} \lambda_i & = & 1. \end{array} \tag{3}$$

The multipliers are determined in the Lagrangian dual of (2), which has the objective function

$$\max_{\lambda} \min_{x \in S} \sum_{i=1}^{Q} \lambda_i f_i(x), \tag{4}$$

where the multipliers still have to fulfill the convexity constraints. (4) is easily solved by minimizing $f_i(x)$ for all i, and then setting $\lambda_i = 1$ for the largest $f_i(x)$.
We conclude that this approach will in fact return the worst possible Pareto optimal solution to our original problem (1) in the bicriteria case. With more than two objectives, worse solutions may exist.

5 K-best Algorithms

As we propose the use of ranking algorithms, our method is obviously restricted to such combinatorial optimization problems for which efficient methods for finding K-best solutions are available. We briefly review some of these here.
The largest amount of research on ranking solutions is available for the shortest path problem. Algorithms developed by Azevedo et al. [1], Martins et al. [19] or Eppstein [11] are very efficient. The best complexity known is $O(m+n\log n+K)$ by Eppstein's method. However, numerical experiments reported by Martins et al. [20] show their algorithm to be very competitive. Its complexity is $O(m + Kn\log n)$.
The second problem for which several methods are known, is the minimum spanning tree problem. We mention papers by Gabow [12] and Katoh et al. [16]. The best known complexity is $O(Km + \min(n^2, m\log\log n))$.
In the seventies and eighties some general schemes for ranking solutions of combinatorial optimization problems have been developed by Lawler [17] and Hamacher and Queyranne [14]. The application of the latter led to algorithms for matroids (Hamacher and Queyranne [14]), with the special case of uniform matroids discussed in Ehrgott [5]. The complexity of the latter is $O(K(n+m) + \min\{n\log n, nm\})$. Finally, an algorithm to rank (integer) network flows was presented in Hamacher [13]. Its complexity is $O(Knm^2)$.
We note that only algorithms allowing the construction of solutions with the same objective function values are applicable in our method. This is evident from the fact that at the

beginning of Phase 2, we have x_1 and x_2 as optimal, i.e. first and second best solutions of the weighted sums problem.

6 Discussion

The algorithm we propose solves the max-ordering problem for two criteria. It works efficiently, as it restricts search (in general) to a small subset of feasible solutions, where max-ordering solutions can be found. As it starts its search from supported Pareto optimal solutions which are much easier to generate than unsupported ones, it will in general enumerate only few solutions. It thereby resolves the difficulties of the ranking method proposed by Hamacher and Ruhe [15] in which the construction of an appropriate λ was an open question.

In addition, for large scale problems, when even the intelligent search applied in our algorithm might result in the enumeration of many feasible solutions (after all the problem is $\mathcal{NP}$-complete), the algorithm can be stopped at any time with the current best as an approximate solution. By computing $g(x^*)-g_{LB}$, where g_{LB} is a lower bound on g, we even have a bound on the distance from the real optimal solution. g_{LB} can easily be calculated and updated in Phase 1 in a straightforward manner. Initially, $g_{LB} = \max\{f_1^1, f_2^2\}$ with updates occurring whenever x_1 or x_2 is updated.

A natural question is the extension of the algorithm to more than two objectives. With such an endeavor we encounter two major difficulties. The first one is that problems with at least three objectives cannot be reduced to subproblems with two objectives only. Thus, in the multicriteria case all criteria have to be considered simultaneously.

Example 1 *Consider a combinatorial problem with three objectives and the following set of efficient vectors (objective vectors of Pareto optimal solutions)*

$$\left\{ \begin{pmatrix} 7 \\ 5 \\ 3 \end{pmatrix}, \begin{pmatrix} 6 \\ 4 \\ 8 \end{pmatrix}, \begin{pmatrix} 9 \\ 4 \\ 2 \end{pmatrix}, \begin{pmatrix} 6 \\ 8 \\ 2 \end{pmatrix} \right\}$$

The unique max-ordering solution is the first one, with $g(x) = 7$. However, looking at only two of the objectives at a time, we obtain the following. For f_1, f_2 only, the minimal value of $g(x)$ is attained at the second solution, for f_2, f_3 it is the third, and for f_1, f_3 it is the fourth. Thus none of the bicriteria subproblems yields the true optimal solution.

The second major difficulty is in the generalization of Phase 1. This problem has been observed by many researchers applying the method for the generation of all Pareto optimal solutions. In contrast to the bicriteria case, there may exist supported efficient points, which lie above (rather than below) a previously constructed hyperplane. For a discussion see Solanki et al. [24]. This kind of problem is very similar to the problem encountered in computing Nadir points for problems with at least three objectives see Ehrgott and Tenfelde [10] for a recent discussion. Further work is required to generalize our method in order to develop at least a heuristic to find a good λ in Phase 1 that will enable an efficient application of the ranking algorithms in Phase 2.

References

[1] J.A. Azevedo, M.E.O. Santos Costa, J.J.E.R. Silvestre Madeira, and E.Q.V. Martins. An algorithm for the ranking of shortest paths. *European Journal of Operational Research*, 69:97–106, 1993.

[2] R. Benayoun, J. de Montgolfier, J. Tergny, and O. Laritchev. Linear programming with multiple objective functions: Step method (stem). *Mathematical Programming*, 1(3):366–375, 1971.

[3] J.T. Buchanan. A naive approach for solving MCDM problems. *Journal of the Operational Research Society*, 48(2):202–206, 1997.

[4] J.M. Coutinho-Rodrigues, J.C.N. Climaco, and J.R. Current. An intercative bi-objective shortest path approach: Searching for unsupported nondominated solutions. *Computers and Operations Research*, 26(8):789–798, 1999.

[5] M. Ehrgott. On matroids with multiple objectives. *Optimization*, 38(1):73–84, 1996.

[6] M. Ehrgott. Integer solutions of multicriteria network flow problems. *Investigacao Operacional*, 19:229–243, 1999.

[7] M. Ehrgott. Approximation algorithms for combinatorial multicriteria optimization problems. *International Transcations in Operational Research*, 7:5–31, 2000.

[8] M. Ehrgott and X. Gandibleux. A survey and annotated bibliography of multiobjective combinatorial optimization. *Operations Research Spektrum*, 22:425–460, 2000.

[9] M. Ehrgott, S. Nickel, and H.W. Hamacher. Geometric methods to solve max-ordering location problems. *Discrete Applied Mathematics*, 93:3–20, 1999.

[10] M. Ehrgott and D. Tenfelde. Nadir values: Computation and use in compromise programming. Technical report, University of Kaiserslautern, Department of Mathematics, 2000. Report in Wirtschaftsmathematik Nr. 60/2000, submitted to European Journal of Operational Research.

[11] D. Eppstein. Finding the k shortest paths. *SIAM Journal on Computing*, 28(2):652–673, 1998.

[12] H.N. Gabow. Two algorithms for generating weighted spanning trees in order. *SIAM Journal of Computing*, 6(1):139–150, 1977.

[13] H.W. Hamacher. A note on K best network flows. *Annals of Operations Research*, 57:65–72, 1995. Special Volume "Industrial Systems".

[14] H.W. Hamacher and M. Queyranne. K best solutions to combinatorial optimization problems. *Annals of Operations Research*, 4:123–143, 1985.

[15] H.W. Hamacher and G. Ruhe. On spanning tree problems with multiple objectives. *Annals of Operations Research*, 52:209–230, 1994.

[16] N. Katoh, T. Ibaraki, and H. Mine. An algorithm for finding k minimum spanning trees. *SIAM Journal of Computing*, 10(2):247–255, 1981.

[17] E.L. Lawler. A procedure for computing the k best solutions to discrete optimization problems and its application to the shortest path problem. *Management Science*, 18:401–405, 1972.

[18] H. Lee and P.S. Pulat. Bicriteria network flow problems: Integer case. *European Journal of Operational Research*, 66:148–157, 1993.

[19] E.Q.V. Martins, M.M.B. Pascoal, and J.L.E. Dos Santos. Deviation algorithms for ranking shortest paths. *International Journal of Foundations of Computer Science*, 10(3):247–261, 1999.

[20] E.Q.V. Martins, M.M.B. Pascoal, and J.L.E. Dos Santos. A new improvement for a k shortest paths algorithm. Technical report, Universidade de Coimbra, 2000.

[21] I. Murthy and S.S. Her. Solving min-max shortest-path problems on a network. *Naval Research Logistics*, 39:669–683, 1992.

[22] R.M. Ramos, S. Alonso, J. Sicilia, and C. González. The problem of the optimal biobjective spanning tree. *European Journal of Operational Research*, 111:617–628, 1998.

[23] K. Rana and R.G. Vickson. A model and solution algorithm for optimal routing of a time-chartered containership. *Transportation Science*, 22:83–96, 1988.

[24] R.S. Solanki, P.A. Appino, and J.L. Cohon. Approximating the noninferior set in multiobjective linear programming problems. *European Journal of Operational Research*, 68:356–373, 1993.

[25] R.E. Steuer and E.U. Choo. An interactive weighted Tchebycheff procedure for multiple objective programming. *Mathematical Programming*, 26:326–344, 1983.

[26] E.L. Ulungu and J. Teghem. Application of the two phases method to solve the bi-objective knapsack problem. Technical report, Faculté Polytechnique de Mons, Belgium, 1994.

[27] E.L. Ulungu and J. Teghem. The two-phases method: An efficient procedure to solve bi-objective combinatorial optimization problems. *Foundations of Computing and Decision Sciences*, 20(2):149–165, 1994.

[28] M. Visée, J. Teghem, M. Pirlot, and E.L. Ulungu. Two-phases method and branch and bound procedures to solve the bi-obective knapsack problem. *Journal of Global Optimization*, 12:139–155, 1998.

[29] A. Warburton. Aproximation of Pareto optima in multiple-objective shortest-path problems. *Operations Research*, 35(1):70 –79, 1987.