Yokohama Publishers
ISSN 1349-8169 ONLINE JOURNAL

Pacific Journal of Optimization
Vol. 2, No. 3, pp. 521–544, September, 2006

A LEVEL SET METHOD FOR MULTIOBJECTIVE COMBINATORIAL OPTIMIZATION: APPLICATION TO THE QUADRATIC ASSIGNMENT PROBLEM

Matthias Ehrgott*, Dagmar Tenfelde-Podehl and Thomas Stephan

Abstract: Multiobjective combinatorial optimization problems have received increasing attention in recent years. Nevertheless, many algorithms are still restricted to the bicriteria case. In this paper we propose a new algorithm for computing all Pareto optimal solutions. Our algorithm is based on the notion of level sets and level curves and contains as a subproblem the determination of K best solutions for a single objective combinatorial optimization problem. We apply the method to the Multiobjective Quadratic Assignment Problem ($MOQAP$). We present two algorithms for ranking QAP solutions and finally give computational results comparing the methods.

Key words: *multiobjective programming, combinatorial optimization, level sets, K-best solution, quadratic assignment problem.*

Mathematics Subject Classification: *90C29, 90C27*

1 Multiobjective Programming and Level Sets

In many real world decision problems several conflicting objectives have to be taken into account. With increasing awareness of this, multicriteria problem formulations have become more and more popular. In order to solve the resulting mathematical models, methods of multicriteria optimization have been developed and incorporated into Decision Support Systems. This branch of mathematical programming has been flourishing over the last two decades and is still gaining popularity, see e.g. [10, 16, 18, 26, 51] for recent monographs and surveys.

A multiobjective mathematical program is written as

$$\min_{X \in \mathcal{X}} (g_1(X), g_2(X), \ldots, g_Q(X)) \tag{1}$$

where g_q, $q = 1, \ldots, Q$ are Q possibly conflicting objective functions and the feasible set $\mathcal{X}$ is a finite subset of $\mathbb{R}^n$. We denote by $g : \mathbb{R}^n \longrightarrow \mathbb{R}^Q$, $g(X) = (g_1(X), g_2(X), \ldots, g_Q(X))$ the vector valued objective function of the multicriteria optimization problem (1). We will use the concept of *Pareto optimality* to define the minimization in (1) in this paper.

*The research of M. Ehrgott has been partially supported by University of Auckland grant 3602178/9275 and grant Ka 477/27-1 of the Deutsche Forschungsgemeinschaft.

Definition 1. *A solution $X^* \in \mathcal{X}$ is called* Pareto optimal *if and only if there is no $X \in \mathcal{X}$ such that $g(X) < g(X^*)$, i.e. $g_q(X) \leq g_q(X^*)$, $q = 1, \ldots, Q$ and $g(X) \neq g(X^*)$. If X^* is Pareto optimal then $g(X^*)$ is called* efficient. *If $X, Y \in \mathcal{X}$ and $g(X) < g(Y)$ we say that X* dominates *Y and $g(X)$* dominates *$g(Y)$. The set of all Pareto optimal solutions is denoted by $\mathcal{X}_{Par}$, the* Pareto set. *The set of all efficient points is denoted by $\mathcal{Y}_{eff}$, the* efficient set.

Independent of the properties of the objective function g or the constraint set $\mathcal{X}$, Pareto optimal solutions can be characterized geometrically. In order to state this characterization we introduce the notion of *level sets* and *level curves*.

Definition 2. *Let $b_q \in \mathbb{R}$.*

1. *The set $L^q_{\leq}(b_q) := \{X \in \mathcal{X} : g_q(X) \leq b_q\}$ is called the* level set *of g_q with respect to the level b_q.*
2. *The set $L^q_{=}(b_q) := \{X \in \mathcal{X} : g_q(X) = b_q\}$ is called the* level curve *of g_q with respect to the level b_q.*

The following characterization of Pareto optimal solutions by level sets and level curves was given by Ehrgott et al. [19].

Lemma 1. *Let $X^* \in \mathcal{X}$. Then X^* is Pareto optimal if and only if*

$$\bigcap_{q=1}^{Q} L^q_{\leq}\left(g_q(X^*)\right) = \bigcap_{q=1}^{Q} L^q_{=}\left(g_q(X^*)\right).$$

Because we will use the result of Lemma 1 throughout the paper the following notation will be convenient. For $b \in \mathbb{R}^Q$ let

$$\mathcal{X}(b) := \{X \in \mathcal{X} : g_q(X) \leq b_q,\ q = 1, \ldots, Q\} = \bigcap_{q=1}^{Q} L^q_{\leq}(b_q).$$

Correspondingly, $\mathcal{X}(b)_{Par}$ will denote the Pareto set of $\mathcal{X}(b)$. Because of Lemma 1, level sets are useful tools to answer certain questions, that are both relevant to decision makers in real world applications and interesting from a theoretical point of view.

Problem 1: Given a feasible solution X, does there exist a feasible solution which dominates X? Literally, this means checking the condition of Lemma 1.

Problem 2: Given a vector $b \in \mathbb{R}^Q$ of upper bounds, determine $\mathcal{X}(b)_{Par}$. Note that many interactive methods include the possibility for the decision maker to specify upper bounds as reservation levels, see e.g. Miettinen [51, Section 5.6] and references therein.

Problem 3: Compute $\mathcal{X}_{Par}$. This has to be done when a final decision is made in an a-posteriori fashion and a most preferred solution is chosen from among the Pareto set.

All three problems can be solved using the characterization given in Lemma 1. We will now show that the essential problem in this list is Problem 2.

Lemma 2. 1. *Problem 1 is a special case of Problem 2.*

2. *Problems 2 and 3 are equivalent.*

Proof. 1. This is obvious by choosing $b_q = g_q(X)$.

2. Problem 2 is a special case of Problem 3 with $\mathcal{X} = \mathcal{X}(b)$. For the converse, choose b_q big enough so that $\mathcal{X}_{Par} = \mathcal{X}_{Par}(b)$, e.g. $b_q = \sup_{X\in\mathcal{X}} g_q(X)$. □

For a more detailed answer to the relationship of Problems 2 and 3, note that the range of values that efficient points can attain is given by a lower and upper bound on the efficient set $\mathcal{Y}_{eff}$ defined by the ideal and nadir points of the multiobjective programming problem (1). The *ideal point* $y^I = (y_1^I, \ldots, y_Q^I)$ is given by

$$y_q^I := \inf_{X\in\mathcal{X}} g_q(X)$$

and the *nadir point* $y^N = (y_1^N, \ldots, y_Q^N)$ is defined by

$$y_q^N := \sup_{X\in\mathcal{X}_{Par}} g_q(X).$$

For the combinatorial problems we consider later in this paper, the set of efficient points is finite, i.e. compact, so that we will be able to substitute inf by min and sup by max. Although the ideal point can be found through the solution of Q single objective problems $\min_{X\in\mathcal{X}} g_q(X)$, computing the nadir point y^N is in general a hard problem, when $Q > 2$ objectives are present (see Ehrgott and Tenfelde-Podehl [20] for a recent discussion of this topic).

A popular heuristic to get an estimation of the nadir point uses the pay-off table. With a minimizer X^q of each objective function g_q compute the pay-off table, a $Q \times Q$ matrix

$$P = \left(g_i(X^j)\right)_{i=1,\ldots,Q;j=1,\ldots,Q}$$

and let

$$\tilde{y}_q^N := \max_{i=1,\ldots,Q} g_q(X^i).$$

$\tilde{y}_q^N$ is called the *estimated nadir point*. It should be mentioned that arbitrarily large over- or underestimation of the nadir point is possible if there are more than two objective functions and a minimizer of one of the objectives is not unique (see Korhonen et al. [39] for an example).

With the nadir point, we can choose $b_q = y_q^N$ as upper bounds to see that Problem 3 is a special case of Problem 2. We will comment on the effects of using $b_q = \tilde{y}_q^N$ rather than $b_q = y_q^N$ in Section 4 when we present numerical results.

In the following section we develop an algorithm to solve Problem 2 above for multiobjective combinatorial optimization (MOCO) problems.

2 An Algorithm for Multiobjective Combinatorial Optimization Based on Level Sets

In this section we develop a method for the determination of Pareto optimal solutions in a multicriteria combinatorial optimization (MOCO) problem based on the characterization given in Lemma 1. The procedure uses an algorithm which solves the problem of finding a K best solution in a combinatorial optimization problem. We will see that the proposed

procedure is not restricted to problems with only two criteria but can be used with any number $Q \geq 2$ of objectives. It can therefore be seen as a generalization of a ranking algorithm for bicriteria shortest path problems by Climaco and Martins [46].

A MOCO problem is a multiobjective program (1) where the feasible set $\mathcal{X}$ is finite. The feasible set is given by a set of linear constraints with integer (in particular, binary) variables that define some combinatorial structure as trees, paths, cycles, etc. of a graph. The objective functions are generally linear functions, often arising as the sum of weights of the elements of the combinatorial structure described by the constraints. The set of efficient solutions of a MOCO problem is partitioned into supported and nonsupported ones. Supported efficient solutions are those which are optimal for a weighted sum problem $\min_{X\in\mathcal{X}} \lambda^T g(X)$, where $\lambda \in \mathbb{R}^Q, \lambda > 0$. For a survey on the state of the art in multiobjective combinatorial optimization see Ehrgott and Gandibleux [17].

The goal is to find all Pareto optimal solutions of a MOCO problem respecting given reservation levels $b_q,\ q = 1,\ldots,Q$. In other words we want to compute $\mathcal{X}(b)_{Par}$.

Instead of an explicit computation of the intersection of level sets and checking the condition of Lemma 1, we will generate one level set ($L^1_{\leq}(b_1)$, without loss of generality) in order of increasing values of the corresponding objective function, and then check for each element of this level set if it is also contained in the other level sets and if it dominates or is dominated by a solution found before.

Let us assume that we have found $\tilde{L} = \{X^1_1,\ldots,X^1_K\} \subseteq L^1_{\leq}(b_1)$ and that $g_1(X^1_1) \leq \cdots \leq g_1(X^1_K)$ and no $X \in \mathcal{X} \setminus \tilde{L}$ with $g_1(X) < g_1(X^1_K)$ exists. Furthermore assume that $\mathcal{X}_{pot} = \{X_{i_1},\ldots,X_{i_k}\} = \tilde{L}_{Par} \subseteq \tilde{L}$ is the subset of (potentially) Pareto optimal solutions.

Let X be a $K+1$-best solution for $\min_{X\in\mathcal{X}} g_1(X)$ and assume $g_q(X) \leq b_q,\ q = 2,\ldots,Q$ (otherwise X cannot satisfy the criterion of Lemma 1). Then

$$g_1(X_{i_1}) \leq \cdots \leq g_1(X_{i_k}) \leq g_1(X).$$

and $\{X_{i_1},\ldots,X_{i_k},X\} \subset L^1_{\leq}(g_1(X))$. Let i_{max} be the maximal index such that $g_1(X_{i_{max}}) < g_1(X)$. We consider two situations.

If there exists $j \in \{i_1,\ldots,i_{max}\}$ such that $g_q(X_j) \leq g_q(X)$ for all $q = 2,\ldots,Q$ then $X_j \notin \bigcap_{q=1}^Q L^q_{=}(g_q(X))$ but $X_j \in \bigcap_{q=1}^Q L^q_{\leq}(g_q(X))$ and X is not Pareto optimal due to Lemma 1.

Otherwise by considering some X_j in the set $\{X_{i_{max}+1},\ldots,X_{i_k}\}$ we can restrict our attention to the restricted objective function vector $g^2 = (g_2,\ldots,g_Q)$. Here four cases can occur.

- If $g^2(X_j) < g^2(X)$ then X is not Pareto optimal.
- If $g^2(X) < g^2(X_j)$ then X_j is not Pareto optimal.
- If $g^2(X_j) = g^2(X)$ then X is a potentially Pareto optimal solution (since X_j is) and is included in $\mathcal{X}_{pot}$
- Otherwise $g^2(X_j)$ and $g^2(X)$ are not comparable and X_j and X do not dominate each other. If this situation occurs for all X_j then X is added to $\mathcal{X}_{pot}$.

Because a level set $L^q_{\leq}(b_q)$ is either empty if $b_q < \min_{X\in\mathcal{X}} g_q(X)$ or can be written as $\{X_1,\ldots,X_K\}$ with $g_q(X_j) \leq g_q(X_{j+1}),\ j = 1,\ldots,K-1$ we will now turn our attention to the computation of K best solutions for combinatorial optimization problems.

2.1 K-Best Solutions of Combinatorial Optimization Problems

In 1985 Hamacher and Queyranne [30] published a binary search tree (BST) algorithm for finding a K best solution in a combinatorial optimization problem. Assuming that a method for computing a best and second best solution for a combinatorial optimization problem is available the BST Algorithm is based upon the following idea.

First, determine a best solution X_1 and a second best solution X_2 with respect to the whole feasible set $\mathcal{X}$. Then partition $\mathcal{X}$ into two disjoint subsets $\mathcal{X}_1$ and $\mathcal{X}_2$ in such a way that X_1 is a best solution with respect to $\mathcal{X}_1$ and X_2 is a best solution with respect to $\mathcal{X}_2$. In both subsets $\mathcal{X}_1$ and $\mathcal{X}_2$ find a second best solution X_1^2 and X_2^2, respectively. The comparison of X_1^2 and X_2^2 yields the third best solution X_3 with respect to $\mathcal{X}$.

Supposing that $X_3 = X_2^2$, rename $\mathcal{X}_2$ as $\tilde{\mathcal{X}}_2$ and partition $\tilde{\mathcal{X}}_2$ into two disjoint subsets $\mathcal{X}_2$ and $\mathcal{X}_3$, again in such a way that X_2 is a best solution with respect to $\mathcal{X}_2$ and X_3 is a best solution with respect to $\mathcal{X}_3$. A comparison of second best solutions X_1^2 (with respect to $\mathcal{X}_1$), X_2^2 (with respect to $\mathcal{X}_2$) and X_3^2 (with respect to $\mathcal{X}_3$) yields the fourth best solution X_4 with respect to $\mathcal{X}$.

Continuing this procedure up to the k^{th} iteration the feasible set $\mathcal{X}$ is partitioned into k disjoint subsets $\mathcal{X} = \mathcal{X}_1 \dot{\cup} \mathcal{X}_2 \dot{\cup} \ldots \dot{\cup} \mathcal{X}_k$, and a best and second best solution with respect to each of these subsets is known. A comparison of all second best solutions X_1^2, X_2^2, ..., X_k^2 yields the $(k+1)$ best solution X_{k+1} with respect to $\mathcal{X}$.

Note that in the BST Algorithm only once (in the first step) a best solution has to be computed. In all subsequent iterations second best solutions are required.

Figure 1 gives an example with four iterations in the BST Algorithm, where the fourth best solution X_4 is the best of X_1^2, X_2^2 and X_3^2.

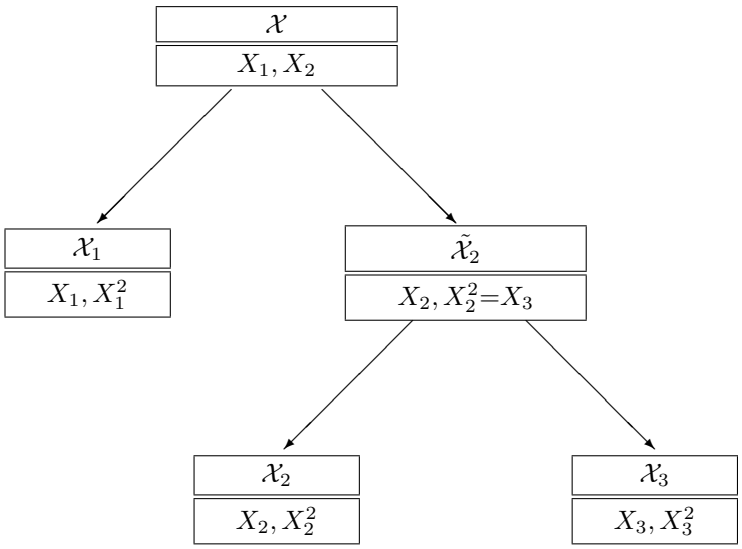

Figure 1: Four Iterations in the BST Algorithm.

An alternative general procedure has been given by Lawler [43] and Murty [52]. This procedure is not a binary one as opposed to the BST Algorithm. Van der Poort [65, 66] compared the number of restricted (by fixing variables to 0 and 1) problems that have to be solved by Lawler's and Hamacher and Queyranne's algorithm for finding the k best

solution of a TSP. He found Lawler's algorithm to be superior in this regard. However, Hamacher and Queyranne [30] showed that in the case that the feasible set $\mathcal{X}$ is a clutter (i.e. $X_1 \in \mathcal{X}, X_2 \in \mathcal{X} \Rightarrow X_1 \not\subseteq X_2$) it holds that the complexity of the Lawler-Murty algorithm is always an upper bound for the complexity of the BST algorithm. Therefore we concentrate on the BST algorithm.

As mentioned before, the use of the algorithm requires a method for computing a best and second best solution of the combinatorial optimization problem under consideration. For many problems, special algorithms are available which exploit the particular structure of the problem. 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. [2], Martins et al. [47] or Eppstein [22] 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. [48] show their algorithm to be very competitive. Its complexity is $O(m + Kn \log n)$.

A problem class closely related to the class of shortest path problems is the one of finding shortest simple paths. Ranking algorithms for this kind of problems were proposed by Carraresi and Sodini [6], Katoh et al. [35], Martins et al. [49] and Yen [68]. It is interesting to note that Hamacher and Queyranne's algorithm can be seen as generalization of Carraresi and Sodini's method.

The third problem for which several methods are known, is the minimum spanning tree problem. We mention papers by Gabow [25] and Katoh et al. [34]. The best known complexity is $O(Km + \min(n^2, m \log \log n))$.

The application of the BST algorithm led to algorithms for matroids (Hamacher and Queyranne [30]), with the special case of uniform matroids discussed in Ehrgott [14]. The complexity of the latter is $O(K(n+m) + \min\{n \log n, nm\})$. Ranking methods for matroid intersections are proposed by Camerini and Hamacher [5]. Chegireddy and Hamacher [8] present an $O(Kn^3)$ algorithm to find K-best perfect matchings, Brucker and Hamacher [3] discuss K-best solutions for polynomially solvable scheduling problems, and finally, an algorithm to rank (integer) network flows was presented in Hamacher [29]. Its complexity is $O(Knm^2)$.

Using the previous tools we are now able to describe the general algorithm for finding Pareto optimal solutions of a MOCO problem.

Pareto optimal solutions with reservation levels

Input: Instance of a MOCO problem with Q criteria, reservation levels $b_1, \ldots, b_Q$
Output: The set $\mathcal{X}(b)_{Par}$ of all Pareto optimal solutions respecting reservation levels b

Step 1: Find an optimal solution X_1 of $\min_{X \in \mathcal{X}} g_1(X)$
Step 2: **if** $g_1(X_1) > b_1$ **then** Stop, $\mathcal{X}(b)_{Par} = \emptyset$
$k := 1$
$\mathcal{X}(b)_{Par} := \{X_k\}$
Step 3: $k := k + 1$
Apply a ranking algorithm to compute the k-best solution X_k for g_1
/* See Section 3.3 and Section 3.4 */
if $g_1(X_k) > b_1$ or no further solution exists **then** Stop, Output $\mathcal{X}(b)_{Par}$
Step 4: **if** $X_k \in L^q_{\leq}(b_q)$ for all $q = 2, \ldots, Q$ **then** goto Step 5
else goto Step 3
Step 5: **for** $1 \leq i \leq k-1$
if X_k dominates X_i, **then** $\mathcal{X}(b)_{Par} = \mathcal{X}(b)_{Par} \setminus \{X_i\}$

else if X_i dominates X_k **then** break and goto Step 3
else if $g(X_k) = g(X_i)$ **then** $\mathcal{X}(b)_{Par} = \mathcal{X}(b)_{Par} \cup \{X_k\}$ break and goto Step 3
Step 6: $\mathcal{X}(b)_{Par} = \mathcal{X}(b)_{Par} \cup \{X_k\}$
goto Step 3

Note that we deal with combinatorial optimization problems, hence the number of solutions is finite and thus the algorithm given above stops after a finite number of steps.

A question that remains to be answered is the choice of the objective for which the level set is constructed. Obviously, one that is small seems to be an intuitively good choice. Therefore, the q which yields the smallest value $b_q - y_q^I$ is recommended. This choice was confirmed in our numerical tests for the multiobjective quadratic assignment problem in Section 4.

3 Solving the Multiobjective Quadratic Assignment Problem

Our goal is now to demonstrate the effectiveness of our algorithm by applying it to a specific problem, the quadratic assignment problem (QAP) with multiple objectives, see below for more details about the QAP. The reason for choosing this notoriously difficult problem is twofold. Most of the algorithms for multiobjective combinatorial optimization problems are only suitable for problems with two criteria and those, which are able to cope with more than two objectives are highly problem dependent, often corresponding to "easy" problems. In the literature, up to now no algorithm is known that can handle the multiobjective QAP with more than two criteria and we want to design an algorithm that is able to deal with such hard problems. The second reason is due to the lack of ranking algorithms specialized for the QAP. Whereas the literature is full of ranking algorithms for "easy" problems (see the literature overview in the last section), for hard problems as the QAP there are hardly any specialized algorithms. Hence we aimed at developing both a multiobjective algorithm and a ranking algorithm for hard combinatorial problems.

3.1 Quadratic Assignment Problems

The first appearance of the quadratic assignment problem (QAP) was in 1957 in an article by Koopmans and Beckmann [38] as a mathematical model of assigning a set of economic activities to a set of locations. Thus, the QAP occurred at first in the context of a facility location problem, still one of its major applications. Examples for facility location problems are the design of a hospital layout by Elshafei [21] and a campus planning model by Dickey and Hopkins [11].

Another example for the usage of the QAP is the so called wiring problem in electronics by Steinberg [62]: n modules have to be placed on n places on a board, where the modules are pairwise connected by a number of wires and the places on the board are given. Let f_{kl} be the number of wires connecting two modules k and l, and d_{ij} be the distance between two places i and j on the board. Then the length of wires needed for connecting the modules k and l which are assigned to the places i and j is given by $f_{kl}d_{ij}$. Now the problem is to find an assignment of modules to places that minimizes the total length of the wires needed.

A general formulation of the QAP is given by Lawler [42]: Let $B = b_{kilj}$, where $i, j, k, l =$

$1, \ldots, n$, be a 4-dimensional array of reals. Then the QAP is given by

$$\min_{\pi \in S_n} \sum_{i=1}^{n} \sum_{j=1}^{n} b_{\pi(i)i\pi(j)j} \tag{2}$$

where S_n is the set of all permutations of $\{1, \ldots, n\}$.

In the case of a facilities layout problem n facilities are to be assigned to n locations. A flow matrix $F = (f_{kl})$ and a distance matrix $D = (d_{ij})$ are given, where f_{kl} is the flow of materials moving from facility k to facility l in a pre-specified period of time and where d_{ij} represents the distance from location i to location j. Then $f_{\pi(i)\pi(j)}d_{ij}$ is the distance travelled between facilities $\pi(i)$ and $\pi(j)$, when simultaneously assigning facility $\pi(i)$ to location i and facility $\pi(j)$ to location j. The objective is to find a permutation π such that the total distance $g(\pi) = \sum_{i=1}^{n} \sum_{j=1}^{n} f_{\pi(i)\pi(j)} d_{ij}$ is minimized.

In this case B is divided into two matrices F and D, where $b_{kilj} = f_{kl}d_{ij}$ for $1 \leq i, j, k, l \leq n$. Using the correspondence between permutations and permutation matrices we get another QAP formulation given by Koopmans and Beckmann [38].

Consider the set $\{1, \ldots, n\}$ and two $n \times n$ matrices $F = (f_{kl}), D = (d_{ij})$, $i, j, k, l = 1, \ldots, n$. Then the (QAP) can be written as

$$\begin{aligned} \min \sum_{i=1}^{n} \sum_{j=1}^{n} \sum_{k=1}^{n} \sum_{l=1}^{n} f_{kl} d_{ij} x_{ik} x_{jl} & \\ \text{subject to } \sum_{i=1}^{n} x_{ij} &= 1 \quad j = 1, \ldots, n \\ \sum_{j=1}^{n} x_{ij} &= 1 \quad i = 1, \ldots, n \\ x_{ij} &\in \{0, 1\} \quad i, j = 1, \ldots, n. \end{aligned} \tag{3}$$

The QAP is $\mathcal{NP}$-complete in the strong sense (see the book by Çela [7] for references) and notorious for its difficulty. Some of the largest instances which are solved optimally to date are from Nugent et al. [53] with $n = 30$ and from Krarup and Pruzan [40] with $n = 32$, see Anstreicher et al. [1]. For more details about reference problems see the QAPLIB [57].

The feasible set of the QAP is denoted by $\mathcal{X}$, where $\mathcal{X}$ is the set of permutation matrices. Throughout this paper we will focus on the Koopmans-Beckmann formulation (3). This formulation can be linearized using the following well known result of Kaufman and Broeckx [36] published in 1978, which is in fact an application of the more general approach proposed by Glover [27].

In the literature there are many different linearizations, see for example [4, 24, 36, 54, 56], but the one by Kaufman and Broeckx is probably the smallest one in terms of the number of variables and constraints.

The integer formulation (3) of the QAP is equivalent to the following mixed integer linear program with n^2 Boolean variables, n^2 real variables and $n^2 + 2n$ constraints.

$$
\begin{aligned}
\min & \sum_{i=1}^{n}\sum_{k=1}^{n} y_{ik} \\
\text{subject to } & \sum_{i=1}^{n} x_{ik} = 1 \quad k = 1,\ldots,n \\
& \sum_{k=1}^{n} x_{ik} = 1 \quad i = 1,\ldots,n \\
c_{ik}x_{ik} + & \sum_{j=1}^{n}\sum_{l=1}^{n} f_{ij}d_{kl}x_{jl} - y_{ik} \leq c_{ik} \quad i,k = 1,\ldots,n \\
& x_{ik} \in \{0,1\} \quad i,k = 1,\ldots,n \\
& y_{ik} \geq 0 \quad i,k = 1,\ldots,n
\end{aligned}
\tag{4}
$$

In this formulation $c_{ik} = \sum_{j=1}^{n}\sum_{l=1}^{n} f_{ij}d_{kl}$ and the additional variables take the values $y_{ik} = x_{ik}\sum_{j=1}^{n}\sum_{l=1}^{n} f_{ij}d_{kl}x_{jl}$.

The best existing exact algorithms are branch and bound algorithms the performance of which depends strongly on the quality of the lower bounds. These lower bounds can be computed by solving a relaxed linearization.

3.2 Multicriteria Models of the QAP

The QAP with multiple objectives (MOQAP) has found little attention so far. We only found a few references [32, 45]. These are closely related to the facility layout problems. A number of papers propose approaches to facility layout based on the quadratic assignment problem [12, 23, 44, 58, 64].

Probably the first time the multicriteria facilities layout problem appeared in literature was in 1979: Rosenblatt [58] considered a bicriteria problem where one criterion was to minimize the material handling costs and the second one was to maximize the total adjacency score. To this end Rosenblatt proposed to reformulate the adjacency-maximization problem as a quadratic assignment problem. This reformulation allowed him to linearly combine the two objective functions and to determine the complete set of supported efficient solutions by varying the parameters and solve single objective QAPs (in fact he disregarded the set of nonsupported efficient solutions). A very similar approach was followed by Dutta and Sahu [12]. They also combined the qualitative and quantitative measure linearly. In Fortenberry and Cox [23], the objective function is different. Instead of combining the two objective functions linearly Fortenberry and Cox propose to take the parameters representing the desirability of being adjacent as weights resulting in a "multiplicity model".

Urban [64] points out that this multiplicity model has two main disadvantages. First of all if there is no flow between two facilities i and k then this pair of facilities does not contribute to the objective function at all, regardless of the adjacency desirability value. The second disadvantage is concerned with the consequences of taking a negative value (-1) for representing undesirable adjacencies as proposed by Fortenberry and Cox [23]. But taking a negative value results in the odd effect that pairs of facilities are more penalized if they have a large work flow between them than if they have a small amount of flow. To overcome these deficiencies Urban [64] proposes to consider a QAP having as objective function a combination of an additive and a multiplicity model.

Malakooti and D'Souza [45] also used the quadratic assignment problem with additively aggregated objective functions as the basis for their investigations. In contrast to the other approaches presented so far they put emphasis on the question of how to determine the weights for the linear combination.

Jacobs [32] pursues a different approach. He describes an interactive layout system for unequally sized rooms and forbidden areas (solid space, circulation space). The subprocedure he uses to generate feasible layouts is mainly a trial and error approach. For measuring the quality of the resulting layout he mentions four objectives, namely distances, structure (as simple as possible), available space (either to minimize or to maximize) and adjacency preferences. He combines the objectives linearly to obtain again a single objective function.

To our knowledge Malakooti [44] stresses the need for determining the nonsupported efficient solutions for the first time in the area of multicriteria facilities layout problems explicitly.

There are different research articles which are based on the models and objective functions described so far, e.g. Harmonosky and Tothero [31] (additive aggregation of different qualitative and quantitative criteria not given in detail as in Urban [64], but not restricted to the bicriteria case and including normalization of parameters; construction heuristic with pairwise exchange), Shang [61] (objective function as in Urban [64], enhanced with two factors a and b corresponding to f_{ik} and r_{ik} respectively, where r_{ik} is determined by applying the Analytical Hierarchy Process (AHP), see Saaty [59]; Simulated Annealing) and Suresh and Sahu [63] (model and objective function of Rosenblatt [58]; Simulated Annealing).

In [41] Krause and Nissen consider a restricted QAP, where the model is extended by "positive zoning constraints" forcing certain facilities to be adjacent. Apart from several approaches where these zoning constraints are added to the objective function they also deal with "real" bicriteria problems where the second objective is to minimize the number of violated zoning constraints.

In [9] and [60] Chen and Sha propose a quite different way to tackle multicriteria facilities layout problems. They use the fact that for the quadratic assignment problem it is possible to find a closed form expression for the mean and the variance of the cost distribution of all feasible layouts. These expressions were already presented by Graves and Whinston in 1970 [28] and rely on the given data only. The determination does not involve any optimization procedure. See also Wallace et al. [67] and Khare et al. [37] for extensions. Using two slightly different approaches to normalize the objectives in order to include both quantitative and qualitative measure into one objective Chen and Sha define two new measures to evaluate the quality of the layout. Given a tolerance probability α of the solution being dominated by another solution the authors propose a two-exchange heuristic in which only layouts are returned for which the probability of being not dominated is larger than $1 - \alpha$. They also determine an expression for the probability of one solution being better than the others. This probability is returned with the corresponding solution so that the decision maker has an additional possibility to rate the quality of the solution.

So generally, there is a lack of methods that consider more than two objectives and that are able to generate all (supported and unsupported) Pareto optimal solutions of MOQAPs. Our method addresses both issues.

3.3 The BST Algorithm Adapted for the QAP

Because the general algorithm presented in Section 2.1 works for any MOCO problem, the only adaption that has to be made for a specific problem is to design an algorithm to compute the K best solution to the problem. We first adapt the BST Algorithm to the QAP and

develop an alternative algorithm in the next section.

Finding a best solution X_1 can be done using well known solution methods for the QAP (for example [13, 33, 55, 50]). For computing a second best solution X_2 let us assume that a best solution X_1 is already known. Now we exclude X_1 from the feasible set $\mathcal{X}$ and minimize over the set $\mathcal{X} \setminus \{X_1\}$, which means that we have to solve the problem

$$\min_{\mathcal{X}\setminus\{X_1\}} \sum_{i,j,k,l=1}^{n} f_{ij} d_{kl} x_{ik} x_{jl}.$$

Then a best solution in $\mathcal{X} \setminus \{X_1\}$ is a second best solution in $\mathcal{X}$. By the special structure of the permutation matrices we can exclude X_1 from the feasible set $\mathcal{X}$ by adding the constraint

$$\sum_{(i,j)\,:\,x_{ij}=1 \text{ in } X_1} x_{ij} \leq n-2 \tag{5}$$

to the Koopmans-Beckmann formulation (3) of the QAP. Constraint (5) is a generalized upper bound constraint.

Lemma 3. *The constraint (5) holds for all $X \in \mathcal{X} \setminus \{X_1\}$.*

Proof. Let $X \in \mathcal{X}$ and $X \neq X_1$. Since X and X_1 are permutation matrices and $X \neq X_1$ there exist at least two index pairs $(i,j),(k,l) \in \{1,\ldots,n\} \times \{1,\ldots,n\}$ such that $x_{ij} = x_{kl} = 1$ in X_1 but $x_{ij} = x_{kl} = 0$ in X. □

Thus finding a second best solution requires the solution of (3) augmented by a generalized upper bound constraint:

$$\begin{aligned} \min \sum_{i=1}^{n}\sum_{j=1}^{n}\sum_{k=1}^{n}\sum_{l=1}^{n} f_{ij} d_{kl} x_{ik} x_{jl} & \\ \text{subject to } \sum_{i=1}^{n} x_{ij} &= 1 \quad j = 1,\ldots,n \\ \sum_{j=1}^{n} x_{ij} &= 1 \quad i = 1,\ldots,n \\ \sum_{(i,j)\,:\,x_{ij}=1 \text{ in } X_1} x_{ij} &\leq n-2 \\ x_{ij} &\in \{0,1\} \quad i,j = 1,\ldots,n \end{aligned} \tag{6}$$

The remaining question is how to partition the feasible set $\mathcal{X}$. We assume that X_1 is a best and X_2 a second best solution in $\mathcal{X}$ and X_2 is computed by (6). Then $X_1 \neq X_2$ and therefore there exists an index pair $(i,j) \in \{1,\ldots,n\} \times \{1,\ldots,n\}$ such that $x_{ij} = 1$ in X_1 but $x_{ij} = 0$ in X_2. Let $\mathcal{X}_1 := \{X \in \mathcal{X} : x_{ij} = 1\}$ and $\mathcal{X}_2 := \{X \in \mathcal{X} : x_{ij} = 0\}$. Then $\mathcal{X} = \mathcal{X}_1 \cup \mathcal{X}_2$ and $\mathcal{X}_1 \cap \mathcal{X}_2 = \emptyset$. Furthermore X_1 is a best solution in $\mathcal{X}_1$ and X_2 is a best solution in $\mathcal{X}_2$.

In general, let $I, O \subset \{1,\ldots,n\} \times \{1,\ldots,n\}$, $I \cap O = \emptyset$ and $\mathcal{X}_{I,O} \subset \mathcal{X}$ be defined as

$$\mathcal{X}_{I,O} := \{X \in \mathcal{X} : x_{ij} = 1 \ \forall (i,j) \in I \text{ and } x_{kl} = 0 \ \forall (k,l) \in O\}.$$

$\mathcal{X}_{I,O}$ is called restricted feasible set and $\min_{X\in\mathcal{X}_{I,O}} \sum_{i,j,k,l=1}^{n} f_{ij} d_{kl} x_{ik} x_{jl}$ is called restricted QAP. Then $\mathcal{X}_1$ and $\mathcal{X}_2$ can be written as $\mathcal{X}_{I_1,O_1}$ and $\mathcal{X}_{I_2,O_2}$, respectively, where $I_1 = O_2 =$

$\{(i,j)\}$ and $O_1 = I_2 = \emptyset$. The sets I and O contain now the information about fixed variables in the restricted feasible set. In order to find K best solutions, we may have to further partition a restricted feasible set.

If we add the constraints

$$\begin{aligned} x_{ij} &= 1 \quad \text{for all } (i,j) \in I \\ x_{kl} &= 0 \quad \text{for all } (k,l) \in O \end{aligned}$$

to (6) the partition of a restricted feasible set is analogous to the partition of $\mathcal{X}$.

3.4 The Multiple Search Tree Algorithm

In this section we develop an alternative algorithm for computing the K best solutions of a QAP. As seen in the last section using the BST Algorithm means partitioning the current feasible set $\mathcal{X}_{I_q,O_q}$ into two disjoint subsets by fixing a variable x_{ij} to one (resulting in $\mathcal{X}^1_{I_q,O_q}$) and to zero (yielding $\mathcal{X}^2_{I_q,O_q}$) respectively. Because feasible solutions X are permutation matrices fixing x_{ij} to one automatically fixes all other variables in row i and column j to zero. Thus, if the current problem is of size l, the restricted problem on $\mathcal{X}^1_{I_q,O_q}$ is of size $l-1$. But fixing a variable x_{ij} to zero does not reduce problem size. Therefore to find the next best solution with respect to $\mathcal{X}_{I_q,O_q}$ we have to solve one problem of size $(l-1)$ and one of size l.

In order to reduce problem size for all restricted QAPs the idea is to fix variables to one only. To explain the idea in more detail let us consider the feasible set $\mathcal{X}$ with known best and second best solution X_1 and X_2. Furthermore let i be a row where a variable x_{ij} occurs with $x_{ij} = 1$ in X_1 and $x_{ij} = 0$ in X_2 (alternatively this can be done with respect to a column). We partition $\mathcal{X}$ into n disjoint subsets by fixing each variable in row i to one. This is possible since $\{X \in \mathcal{X} : x_{ij} = 0\} = \dot\cup_{k\neq j}\{X \in \mathcal{X} : x_{ik} = 1\})$. Then X_1 is a best solution with respect to one of these subsets and the same holds for X_2. So $\mathcal{X}$ is partitioned into $\mathcal{X} = \mathcal{X}_1 \dot\cup \mathcal{X}_2 \dot\cup \ldots \dot\cup \mathcal{X}_n$, where X_1 is a best solution with respect to $\mathcal{X}_i$ for one $i \in \{1,\ldots,n\}$ and X_2 is a best solution with respect to $\mathcal{X}_j$ for one $j \in \{1,\ldots,n\}, j \neq i$.

Thus finding candidates for the third best solution requires the computation of a second best solution with respect to $\mathcal{X}_i$ and $\mathcal{X}_j$ and the computation of best solutions with respect to $\mathcal{X}_k, k \in \{1,\ldots,n\}, k \neq i,j$. This means that two times a second best solution and $(n-2)$ times a best solution has to be computed. Since each of these problems has size $(n-1)$ finding a third best solution with respect to $\mathcal{X}$ requires the solution of n problems of size $(n-1)$ as compared to solving two problems of size $(n-1)$ and n, respectively. We chose this approach because the computation times needed for solving QAPs dramatically increase with n. Therefore it seems to be a good idea to solve more smaller problems rather than fewer problems of full size. See Section 4 for empirical confirmation of this intuition.

Similar to the last section let $I \subset \{1,\ldots,n\} \times \{1,\ldots,n\}$ and define $\mathcal{X}_I \subset \mathcal{X}$ as

$$\mathcal{X}_I := \{X \in \mathcal{X} : x_{ij} = 1 \ \forall (i,j) \in I\}.$$

Then $\mathcal{X}_I$ is called restricted feasible set and $\min_{X\in\mathcal{X}_I} g(X)$ is called a restricted QAP. Adding the constraints

$$\begin{aligned} x_{ij} &= 1 \quad (i,j) \in I \\ \sum_{\substack{(i,j):\\ x_{ij}=1 \ in \ X_{rest}}} x_{ij} &\leq n-2, \end{aligned}$$

where X_{rest} is a best solution with respect to $\mathcal{X}_I$ to the QAP leads to the following minimization problem. Solving it gives a second best solution in a restricted QAP, where a best solution X_{rest} is known.

$$\begin{aligned}
\min \sum_{i=1}^{n}\sum_{j=1}^{n}\sum_{k=1}^{n}\sum_{l=1}^{n} f_{ij}d_{kl}x_{ik}x_{jl} & \\
\text{subject to } \sum_{i=1}^{n} x_{ij} &= 1 \quad j = 1,\ldots,n \\
\sum_{j=1}^{n} x_{ij} &= 1 \quad i = 1,\ldots,n \\
x_{ij} &= 1 \quad (i,j) \in I \\
\sum_{\substack{(i,j):\\ x_{ij}=1 \, in \, X_{rest}}} x_{ij} &\leq n-2 \\
x_{ij} &\in \{0,1\} \quad i,j = 1,\ldots,n.
\end{aligned}$$

Now consider a restricted feasible set $\mathcal{X}_{I_q}$, $|\mathcal{X}_{I_q}| \geq 2$, with X_q and X_q^2 as known local best and second best solution. Choose two index pairs $(i_q, j_1), (i_q, j_2) \in \{1,\ldots,n\} \times \{1,\ldots,n\}$, $(i_q, j_1) \neq (i_q, j_2)$, in such a way that $x_{i_q j_1} = 1$ in X_q and $x_{i_q j_2} = 1$ in X_q^2.

Let $J = \{j : (i,j) \notin I_q \ \forall \ i\}$ (i.e. $|J| = n - |I_q|$) be the set of all column indices that do not occur in I_q. Therefore J contains all column indices such that in the corresponding column a variable is not fixed to one so far, i.e. the information which of the variables $x_{i_q j}$ in row i_q can be fixed to one.

Let $I_j^{new} := I_q \cup \{(i_q, j)\}$ for $j \in J$. Then $\mathcal{X}_{I_q} = \bigcup_{j \in J} \mathcal{X}_{I_j^{new}}$ and $\mathcal{X}_{I_j^{new}} \cap \mathcal{X}_{I_k^{new}} = \emptyset$ for all $j, k \in J, j \neq k$. Furthermore X_q is a best solution in $\mathcal{X}_{I_{j_1}^{new}}$ and X_q^2 is a best solution in $\mathcal{X}_{I_{j_2}^{new}}$.

Therefore the restricted feasible set $\mathcal{X}_{I_q}$ is partitioned into $n - |I_q|$ disjoint subsets $\mathcal{X}_{I_j^{new}}$ in such a way that in two of them a best solution is known. This leads to the MST Algorithm.

Before stating it we remark that for formulating the algorithm in an exact way we need a slightly more sophisticated notation: The sets of column indices will be indexed by the iteration index k. Moreover when partitioning a set $\mathcal{X}_{I_q}$ those subsets in which an optimal solution is not yet known get a subscript j depending on the column for which a variable is fixed and a superscript k which is again the iteration index. See Step 7 in the algorithm for details and the subsequent example for illustration.

Multiple Search Tree (MST) Algorithm

Input: Instance of a QAP of size n with flow matrix F and distance matrix D
Integer K, $2 \leq K \leq n!$
Output: A K-best solution X_K for the QAP

Step 1: $I_1 := \emptyset$
$BestSol := SecBestSol := \emptyset$
$k := 2$
Step 2: Compute X_1 and X_1^2 in $\mathcal{X}_{I_1} = \mathcal{X}$
$SecBestSol := SecBestSol \cup \{X_1^2\}$
Step 3: $X_k := argmin\{\, g(X) : X \in BestSol \cup SecBestSol \,\}$
If $k = K$ **then STOP**

Step 4: **If** $X_k \in BestSol$ **then** $BestSol := BestSol \setminus \{X_k\}$, goto Step 5
else $SecBestSol := SecBestSol \setminus \{X_k\}$, goto Step 6

Step 5: (X_k is a best solution in $\mathcal{X}_{I_j^l}$, $1 \leq l \leq k-1$, $j \in J_l$)
$I_k := I_j^l$
Compute X_k^2 in $\mathcal{X}_{I_k}$
$SecBestSol := SecBestSol \cup \{X_k^2\}$
$k := k+1$, goto Step 3

Step 6: (X_k is a second best solution in $\mathcal{X}_{I_q}$, $1 \leq q \leq k-1$)
If $|\mathcal{X}_{I_q}| = 2$ **then** $k := k+1$, goto Step 3

Step 7: Choose $(i_q, j_1) \neq (i_q, j_2)$ such that $x_{i_q j_1} = 1$ in X_q (best solution in $\mathcal{X}_{I_q}$) and $x_{i_q j_2} = 1$ in X_k
$J_k := \{j : (i,j) \notin I_q \ \forall\ i\}$
$I_k := I_q \cup \{(i_q, j_2)\}$
$I_j^k := I_q \cup \{(i_q, j)\}$ for all $j \in J_k \setminus \{j_1, j_2\}$
$I_q := I_q \cup \{(i_q, j_1)\}$

Step 8: Compute a second best solution X_q^2 and X_k^2 in $\mathcal{X}_{I_q}$ and $\mathcal{X}_{I_k}$, respectively
Compute a best solution Y_j^k in each set $\mathcal{X}_{I_j^k}$ $\forall j \in J_k \setminus \{j_1, j_2\}$
$BestSol := BestSol \cup \{Y_j^k\}$ for all $j \in J_k \setminus \{j_1, j_2\}$
$SecBestSol := SecBestSol \cup \{X_q^2, X_k^2\}$
$k := k+1$, goto Step 3

Note that this (MST) algorithm is, as the procedure of Murty [52] and Lawler [43], not binary. But the main difference between these two algorithms is the fact that we only fix variables to one while in the Lawler/Murty procedure variables are fixed both to one and to zero. As already mentioned the main advantage of fixing a variable to one is that in case of assignment constraints this means fixing $2n-1$ variables simultaneously, while fixing a variable to zero fixes only this one variable.

Figure 2 shows an example for $n = 4$, where the feasible set $\mathcal{X}$ is partitioned into 4 disjoint subsets $\mathcal{X}_{I_1}, \mathcal{X}_{I_2}, \mathcal{X}_{I_3{}^2}$ and $\mathcal{X}_{I_4{}^2}$. Local second best solutions X_1^2 and X_2^2 in $\mathcal{X}_{I_1}$ and $\mathcal{X}_{I_2}$, respectively, and local best solutions Y_3^2 and Y_4^2 in $\mathcal{X}_{I_3^2}$ and $\mathcal{X}_{I_4^2}$, respectively, have to be computed. These solutions are the candidates for the third best solution X_3 and it turns out that $Y_4^2 = X_3$. Then a local second best solution X_3^2 in $\mathcal{X}_{I_3}$ has to be computed and the candidates for the fourth best solution are $X_1^2, X_2^2, Y_3^2, X_3^2$ and it turns out that $X_4 = X_2^2$. Now $\mathcal{X}_{I_2}$ is partitioned into 3 disjoint subsets $\mathcal{X}_{I_1^4}, \mathcal{X}_{I_4}$ and $\mathcal{X}_{I_2}$. A local best solution Y_1^4 in $\mathcal{X}_{I_1^4}$ and local second best solutions X_4^2 and X_2^2 in $\mathcal{X}_{I_4}$ and $\mathcal{X}_{I_2}$, respectively, have to be computed. Thus the candidates for the fifth best solution X_5 are $X_1^2, Y_1^4, X_4^2, X_2^2, Y_3^2$ and X_3^2.

Before we come to some computational results we specify the generic algorithm we gave in Section 2 for the MOQAP. Since the formulation is not very different from the generic one we only note the small change that is necessary:

Replace the second line of Step 3 ("Apply a ranking algorithm ...") by

Apply the BST algorithm to compute the k-best solution X_k for g_1

or by

Apply the MST algorithm to compute the k-best solution X_k for g_1

depending on which of the two ranking algorithms we want to use.

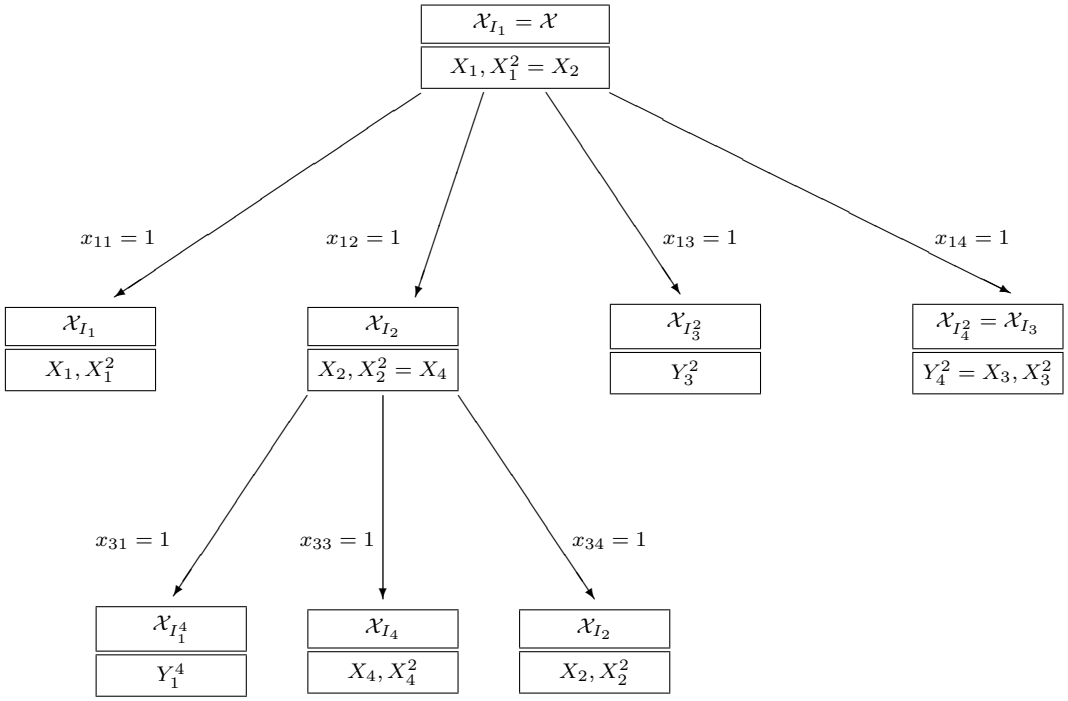

Figure 2: Example of the MST Algorithm for $n = 4$.

4 Computational Results

In this section we give computational results obtained by using the MST and BST Algorithm for the determination of Pareto optimal solutions of MOQAP.

A method for finding a K best solution in the single criterion problem is required (see Sections 3.3 and 3.4). Since the QAP with single objective is already very hard to solve our aim is not to show the efficiency but the effectiveness of our algorithm. Therefore we restricted ourselves to small instances and solved the single objective QAP by just using the linearization of Kaufman and Broeckx (see Section 3) instead of going deeper into different algorithmic approaches to the QAP. This linearization was implemented using AMPL and CPLEX 7.0. The main algorithm of Section 2.1 was implemented in C++. If a state of the art QAP code is available, this can be substituted for our generic AMPL/CPLEX calls.

All examples in this section are generated uniformly distributed (integers in the interval $[0, 49]$). The main diagonals of the distance matrices D_q are set to zero ($d_{jj}^q = 0$ for all $j = 1, \ldots, n$ and $q = 1, \ldots, Q$), since the distance from a location to itself is in general assumed to be zero. To have the model as general as possible, we did not assume that the triangle inequality is satisfied. For example if the distances are measured in time, then the triangle inequality does not hold necessarily (e.g. congested routes on the direct link, free routes on the detour).

In all examples the estimated nadir point $\tilde{y}^N$ was chosen as reservation level vector.

Table 1 shows results achieved by problems of size $n = 4, \ldots, 7$ and number of objective functions $Q = 2, \ldots, 6$. The values are average values over five problems.

	n	4		5		6		7	
Q		BST	MST	BST	MST	BST	MST	BST	MST
2	#QAPs solved	12.8	15.2	79.8	84.8	346.4	375.4	1798.5	2019.6
	Total QAP time	0.84	0.92	5.50	5.19	30.57	26.15	213.84	174.35
3	#QAPs solved	17.8	19.4	100.6	102.8	521.2	550.6	2226.3	2371.6
	Total QAP time	1.16	1.18	6.68	6.05	41.59	36.57	242.03	198.63
4	#QAPs solved	21.4	22.2	101.6	104.2	629.6	647.6	3195.4	3461.6
	Total QAP time	1.35	1.38	6.83	6.32	47.74	42.17	325.97	272.1
5	#QAPs solved			102.6	103.8	541	570.6		
	Total QAP time			7.38	6.79	43.38	37.31		
6	#QAPs solved					532.2	563.5		
	Total QAP time					43.27	38.13		

Table 1: Computational results.

The value in the first row specifies the average number of QAPs that have to be solved by using the BST and MST Algorithm, the second row value specifies the average total time (CPU time in seconds) that is needed for solving the QAPs.

Number of facilities n	$Q = 2$		$Q = 3$		$Q = 4$	
	$b = \tilde{y}^N$	$b = \infty$	$b = \tilde{y}^N$	$b = \infty$	$b = \tilde{y}^N$	$b = \infty$
4	3.4	3.4	4.8	7.1	7.2	10.8
	$[1, 7]$	$[1, 7]$	$[2, 9]$	$[4, 9]$	$[4, 12]$	$[8, 14]$
5	6	6	8.2	11.8	16.6	27.2
	$[3, 9]$	$[3, 9]$	$[4, 15]$	$[6, 22]$	$[7, 35]$	$[15, 44]$
6	7.2	7.2	22.6	27.9	44.6	68.8
	$[3, 16]$	$[3, 16]$	$[15, 34]$	$[16, 46]$	$[22, 64]$	$[39, 100]$
7	10.2	10.2	36.3	44.2	94.0	130.1
	$[6, 15]$	$[6, 15]$	$[24, 57]$	$[30, 60]$	$[50, 114]$	$[99, 179]$

Table 2: Number of Pareto optimal solutions.

Table 2 shows the number of Pareto optimal solutions using the estimated nadir point as reservation levels and the true number of Pareto optimal solutions. These are the same for BST and MST, of course. Again the entries of the coefficient matrices of the objective functions are integers, uniformly distributed in the interval $[0, 49]$. The values given in the first row are average values over ten instances, in the second row we give the minimum and the maximum number of Pareto solutions we found for the ten instances. This shows how the number increases with problem size and number of objectives. It also clearly illustrates that the use of the estimated nadir point prevents many Pareto optimal solutions from being found.

In each of these examples using the MST method had the effect that more QAPs than using the BST method had to be solved. But except for $n = 4$ the time needed using the

MST Algorithm was shorter than using the BST Algorithm which confirms our intuition that reducing the problem size outweighs an increased number of problems to be solved.

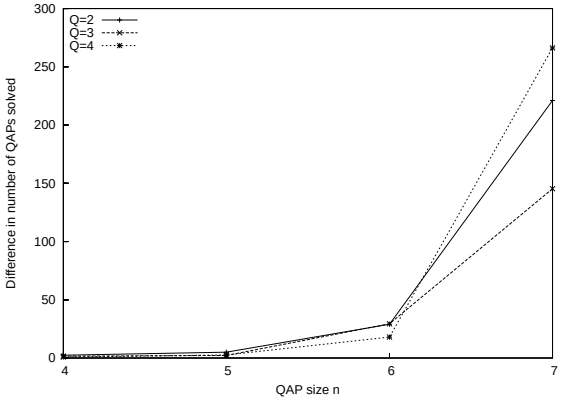

Figure 3: Difference in the number of solved QAPs using the MST and BST method depending on the problem size n.

Figure 3 shows the difference in the number of QAPs solved using the MST and BST method in the case of two, three and four objective functions, depending on the problem size n. $\triangle|QAP|$ represents the difference in the number of solved QAPs. The pictures show that with increasing problem size n the difference in the number of solved QAPs using the MST and BST Algorithm is also increasing.

Figure 4 shows the time needed to solve the QAPs using the MST method in percentage of the time needed using the BST method depending on the problem size n. This illustrates that the advantage of the MST over the BST method increases with problem size.

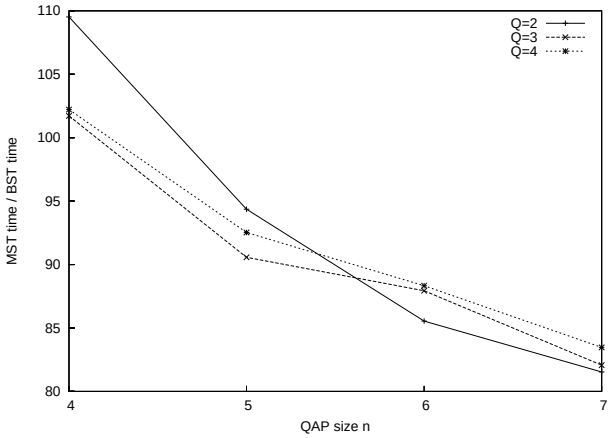

Figure 4: MST time in percentage of BST time depending on problem size n.

Table 3 shows the results for 10 examples of size $n = 7$ and $Q = 4$ objectives. The table contains the size of $L^q_{\leq}(\tilde{y}^N_q)$, the size of $\mathcal{X}(\tilde{y}^N)$ and the number of Pareto optimal solutions,

$|\mathcal{X}(\tilde{y}^N)_{Par}|$. Additionally we give the number of AMPL calls, i.e. the number of QAPs solved during the procedure, and the time used for all AMPL computations.

Example	$\lvert L^q_{\leq}(\tilde{y}^N_q)\rvert$	$\lvert\mathcal{X}(\tilde{y}^N)\rvert$	$\lvert\mathcal{X}(\tilde{y}^N)_{Par}\rvert$	#AMPL calls		AMPL time	
				BST	MST	BST	MST
1	3020	1934	101	3996	4193	369.33	310.63
2	2468	1191	101	3479	3700	348.85	288.03
3	3560	1564	107	4464	4544	374.63	315.74
4	1797	1411	96	2847	3225	298.27	257.06
5	4892	4162	112	5034	5036	425.53	351.98
6	3504	1011	93	4463	4611	395.14	321.42
7	4461	3403	85	4899	4897	402.02	326.15
8	4385	3552	80	4932	4962	415.14	336.57
9	2692	2014	128	4031	4203	377.33	305.71
10	3669	1382	97	4760	4834	407.31	332.06

Table 3: Computational results for 10 examples with $n = 7$ and $Q = 4$.

Figure 5 illustrates the results of Table 3. It shows the number of solved QAPs against the size of $L^q_{\leq}(\tilde{y}^N_q)$. In each example the number of solved QAPs using the MST method is higher than using the BST method. With increasing size of $L^q_{\leq}(\tilde{y}^N_q)$ the difference between the MST and BST Algorithm is getting smaller.

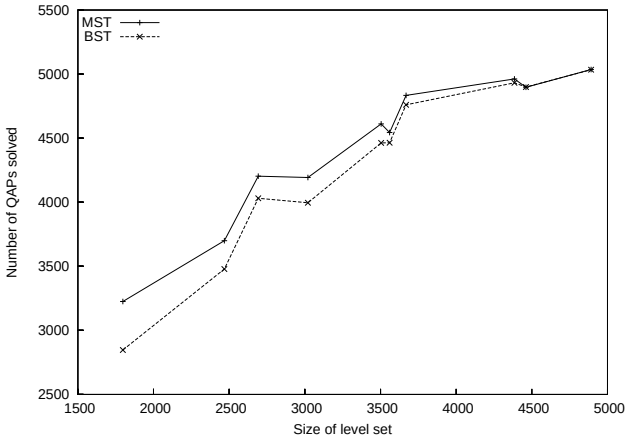

Figure 5: Number of solved QAPs versus the size of $L^1_{\leq}(\tilde{y}^N_q)$ for $n = 7$ and $Q = 4$.

Figure 6 illustrates Table 3 by showing the total QAP time needed for the MST and BST algorithms depending on $|L^q_{\leq}(\tilde{y}^N_q)|$. For each example the computation time of the BST method is higher than for the MST method.

Figure 6 shows (unsurprisingly) that the computation time depends on the size of the level set $L^q_{\leq}(\tilde{y}^N_q)$. Therefore the goal should be to compute the smallest level set $L^q_{\leq}(\tilde{y}^N_q)$. In our experiments it turned out that computing the level set with $\tilde{y}^N_{min} - y^I_{min} = \min\{\tilde{y}^N_q - y^I_q : q = 1, \ldots, Q\}$ often achieves this goal.

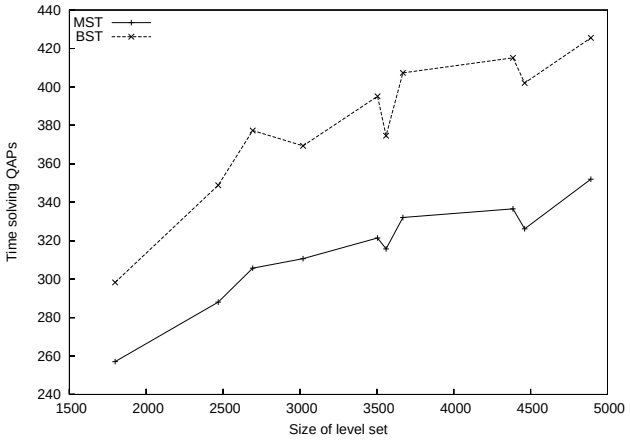

Figure 6: Total QAP time versus the size of $L^q_{\leq}(\tilde{y}^N_q)$ for $n = 7$ and $Q = 4$.

All results obtained for the examples indicate the advantage of using the MST instead of the BST Algorithm. But further computations with dimensions $n \geq 8$ should be performed with a competitive QAP solver.

5 Conclusions and Final Remarks

We have developed a general algorithm for the computation of Pareto optimal solutions in a MOCO problem. The main advantage of our algorithm is that it is not restricted to two objectives. It works with any number of criteria, by considering one of the objectives explicitly by the K best procedure and evaluate the others in the process. It can therefore be seen as a generalization of a ranking algorithm for bicriteria shortest path problems by Climaco and Martins [46]. The main drawback appears to be that in the worst case all feasible solutions have to be enumerated. But remember that this will be the case for any algorithm that finds the complete Pareto set, because every feasible solution could be Pareto optimal, see Ehrgott [15] for some further references on this.

Our algorithm can also be converted to an interactive procedure, where the decision maker can change reservation levels in the course of the process, to guide the search towards a most preferred solution. In such a procedure aspiration levels can also be considered. The algorithm would then discard solutions below the aspiration level value.

We have adapted this algorithm to the MOQAP and proposed an alternative method for determining a level set in the QAP. There are a couple of open questions concerning the algorithm and the implementation. Having shown that the method proposed in this paper works it would be interesting to use more sophisticated algorithms to solve the single objective QAPs occurring as subproblems, in particular to solve problems with $n \geq 8$. Special cases of the QAP with symmetric matrices or polynomially solvable cases could be investigated. In future research we will apply the general algorithm to other combinatorial optimization problems.

References

[1] K. Anstreicher, N. Brixius, J.-P. Goux, and J. Linderoth, Solving large quadratic assignment problems on computational grids, *Mathematical Programming Series B* 91 (2002) 563–588.

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

[3] P.J. Brucker and H.W. Hamacher, K-optimal solution sets for some polynomially solvable scheduling problems, *European Journal of Operational Research* 41 (1989) 194–202.

[4] R.E. Burkard. Locations with spatial interactions: The quadratic assignment problem, in *Discrete Location Theory*, P.B. Mirchandani and R.L. Francis (eds.), Wiley, New York, 1991, pp. 387–437.

[5] P.M. Camerini and H.W. Hamacher, Intersection of two matroids: (Condensed) border graphs and ranking, *SIAM Journal of Algebaric and Discrete Methods* 2 (1989) 16–27.

[6] P. Carraresi and C. Sodini, A binary enumeration tree to find k shortest paths, *Methods of Operations Research* 45 (1983) 177–188.

[7] E. Çela, *The Quadratic Assignment Problem*, Kluwer Academic Publishers, Dordrecht, Boston, London, 1998.

[8] C.R. Chegireddy and H.W. Hamacher, Algorithms for finding k-best perfect matchings, *Discrete Applied Mathematics* 18 (1987) 155–165.

[9] C.-W. Chen and D. Y. Sha, A design approach to the multi-objective facility layout problem, *International Journal of Production Research* 37 (1999) 1175–1196.

[10] K. Deb, *Multi-Objective Optimization using Evolutionary Algorithms*, John Wiley & Sons, Chichester, 2001.

[11] J.W. Dickey and J.W. Hopkins. Campus building arrangement using TOPAZ, *Transportation Research* 6 (1972) 59–68.

[12] K.N. Dutta and S. Sahu, A multigoal heuristic for facilities design problems: MUGHAL, *International Journal of Production Research* 20 (1982) 147–154.

[13] C.S. Edwards, A branch and bound algorithm for the Koopmans-Beckmann quadratic assignment problem, *Mathematical Programming Study*, 13 (1980) 35–52.

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

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

[16] M. Ehrgott, *Multicriteria Optimization. 2nd edition*, Springer-Verlag, Berlin, 2005.

[17] M. Ehrgott and X. Gandibleux, A survey and annotated bibliography of multiobjective combinatorial optimization, *OR Spektrum* 22 (2000) 425–460.

[18] M. Ehrgott and X. Gandibleux (eds.), *Multiple Criteria Optimization – State of the Art Annotated Bibliographic Surveys*, Kluwer Academic Publishers, Boston, 2002.

[19] M. Ehrgott, H.W. Hamacher, K. Klamroth, S. Nickel, A. Schöbel, and M.M. Wiecek, A note on the equivalence of balance points and Pareto solutions in multiple-objective programming, *Journal of Optimization Theory and Applications* 92 (1997) 209–212.

[20] M. Ehrgott and D. Tenfelde-Podehl, Computation of ideal and nadir values and implications for their use in MCDM methods, *European Journal of Operational Research* 151 (2003) 119–131.

[21] A.N. Elshafei, Hospital layout as a quadratic assignment problem, *Operations Research Quarterly* 28 (1977) 167–179.

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

[23] J.C. Fortenberry and J.F. Cox, Multicriteria approach to the facilities layout problem, *International Journal of Production Research* 23 (1985) 773–782.

[24] A.M. Frieze and J. Yadegar, On the quadratic assignment problem, *Discrete Applied Mathematics* 5 (1983) 89–98.

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

[26] T. Gal, T. Stewart, and T. Hanne (eds.), *Multicriteria Decision Making – Advances in MCDM Models, Algorithms, Theory, and Applications*, Kluwer Academic Publishers, Norwell, 2002.

[27] F. Glover, Improved linear integer programming formulations of nonlinear integer problems, *Management Science* 22 (1975/76) 455–460.

[28] G. W. Graves and A. B. Whinston, An algorithm for the quadratic assignment problem, *Management Science* 17 (1970) 453–471.

[29] H.W. Hamacher, K best network flows, *Annals of Operations Research* 57 (1995) 65–72.

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

[31] C.M. Harmonosky and G.K. Tothero, A multi-factor plant layout methodology, *International Journal of Production Research* 30 (1992) 1773–1789.

[32] F.R. Jacobs, A layout planning system with multiple criteria and a variable domain representation, *Management Science* 33 (1987) 1020–1034.

[33] B.K. Kaku and G.L. Thompson, An exact algorithm for the general quadratic assignment problem, *European Journal of Operational Research* 23 (1986) 382–390.

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

[35] N. Katoh, T. Ibaraki, and H. Mine, An efficient algorithm for k shortest simple paths, *Networks* 12 (1982) 411–427.

[36] L. Kaufman and F. Broeckx, An algorithm for the quadratic assignment problem using Benders decomposition, *European Journal of Operational Research* 2 (1978) 204–211.

[37] V.K. Khare, M.K. Khare, and M.L. Neema, Estimation of distribution parameters associated with facilites design problem involving forward and backtracking of materials, *Computers and Industrial Engineering* 14 (1988) 63–75.

[38] T.C. Koopmans and M.J. Beckmann, Assignment problems and the location of economic activities, *Econometrica* 25 (1957) 53–76.

[39] P. Korhonen, S. Salo, and R.E. Steuer, A heuristic for estimating Nadir criterion values in multiple objective linear programming, *Operations Research* 45 (1997) 751–757.

[40] J. Krarup and P.M. Pruzan, Computer-aided layout design, *Mathematical Programming Study* 9 (1978) 75–94.

[41] M. Krause and V. Nissen, On using penalty functions and multicriteria optimisation techniques in facility layout, in *Evolutionary Algorithms in Management Applications*, J. Biethahn (ed.), Springer-Verlag, Berlin, 1995, pp. 153–166.

[42] E.L. Lawler, The quadratic assignment problem, *Management Science* 9 (1963) 586–599.

[43] E.L. Lawler, A procedure for computing the k best solutions and its application to the shortest path problem, *Management Science* 18 (1972) 401–405.

[44] B. Malakooti, Multiple objective facility layout: A heuristic to generate efficient alternatives, *International Journal of Production Research* 27 (1989) 1225–1238.

[45] B. Malakooti and G.I. D'Souza Multiple objective programmig for the quadratic assignment problem, *International Journal of Production Research* 25 (1987) 285–300.

[46] E.Q.V. Martins and J.C.N. Clímaco, On the determination of the nondominated paths in a multiobjective network problem, *Methods of Operations Research* 40 (1981) 255–258.

[47] 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 (1999) 247–261.

[48] E.Q.V. Martins, M.M.B. Pascoal, and J.L.E. Dos Santos, A new improvement for a k shortest paths algorithm, *Investigação Operacional* 21 (2001) 47–60.

[49] E.Q.V. Martins, M.M.B. Pascoal, and J.L.E. Dos Santos, Deviation algorithms for ranking shortest paths, *The International Journal of Foundations of Computer Science* 69 (1999) 97–106.

[50] T. Mautor and C. Roucairol, A new exact algorithm for the solution of quadratic assignment problems, *Discrete Applied Mathematics* 55 (1982) 281–293.

[51] K. Miettinen, *Nonlinear Multiobjective Optimization*, Kluwer Academic Publishers, Dordrecht, 1999.

[52] K.G. Murty, An algorithm for ranking all the assignments in increasing order of cost, *Operations Research* 16 (1968) 682–687.

[53] C.E. Nugent, T.E. Vollman, and J. Ruml, An experimental comparison of techniques for the assignment of facilities to locations, *Operations Research* 16 (1968) 150–173.

[54] M.W. Padberg and M.P. Rijal, *Location, Scheduling, Design and Integer Programming*, Kluwer Academic Publishers, Boston, 1996.

[55] P. Pardalos and J. Crouse, A parallel algorithm for the quadratic assignment problem, In *Proceedings of the Supercomputing Conference 1989*, ACM Press, 1989, pp. 351–360.

[56] P. Pardalos, F. Rendl, and H. Wolkowicz, The quadratic assignment problem: A survey and recent developments, in *Quadratic Assignment and Related Problems*, P. Pardalos and H. Wolkowicz (eds.), American Mathematical Society, Providence, 1994, pp. 1–42.

[57] QAPLIB, http://www.seas.upenn.edu/qaplib/

[58] M.J. Rosenblatt, The facilities layout problem: A multi-goal aproach, *International Journal of Production Research* 17 (1979) 323–332.

[59] T.L. Saaty, *The Analytic Hierachy Process*, McGraw-Hill, New York, 1980.

[60] D.Y. Sha and C.-W. Chen, A new approach to the multiple objective facility layout problem, *Integrated Manufacturing Systems* 12 (2001) 59–66.

[61] J.S. Shang, Multicriteria facility layout problem: An integrated approach, *European Journal of Operational Research* 66 (1993) 291–304.

[62] L. Steinberg, The backboard wiring problem: A placement algorithm, *SIAM Review* 3 (1961) 37–50.

[63] G. Suresh and S. Sahu, Multiobjective facility layout using simulated annealing, *International Journal of Production Economics* 32 (1993) 239–254.

[64] T.L. Urban, A multiple criteria model for the facilities layout problem, *International Journal of Production Research* 25 (1987) 1805–1812.

[65] E.S. van der Poort, *Aspects of Sensitivity Analysis for the Traveling Salesman Problem*, PhD thesis, Rijksuniversiteit Groningen, The Netherlands, 1997.

[66] E.S. van der Poort, M. Libura, G. Sierksma, and J.A.A. van der Veen, Solving the k-best traveling salesman problem, *Computers & Operations Research* 26 (1999) 409–425.

[67] H. Wallace, G.G. Hitchings, and D.R. Towill, Parameter estimation for distributions associated with the facilites design problem, *International Journal of Production Research* 14 (1976) 263–274.

[68] J.Y. Yen, Finding the k shortest loopless paths in a network, *Management Science* 17 (1971) 712–716.

Manuscript received 20 January 2006
revised 24 July 2006
accepted for publication 24 July 2006

Matthias Ehrgott
Department of Engineering Science, The University of Auckland, Private Bag 92019, Auckland, New Zealand
E-mail address: m.ehrgott@auckland.ac.nz

Dagmar Tenfelde-Podehl
Volkswagen AG, GOG Prozess- und Organisationsentwicklung, Brieffach 1807, 38436 Wolfsburg, Germany
E-mail address: dagmar.tenfelde-podehl@volkswagen.de

Thomas Stephan
Fachbereich Mathematik, Technische Universität Kaiserslautern, Postfach 3049, 67653 Kaiserslautern, Germany