Allocation Search Methods for a Generalized Class of Location-Allocation Problems

Martin Bischoff*, Kerstin Dächert

Institute of Applied Mathematics, University of Erlangen-Nuremberg, Germany

March 18, 2007

We consider a generalized class of location-allocation problems, in which N new facilities are to be located in the plane with respect to M objects. Each object is associated with a convex cost function, specifying the expenses for serving the object from any location in the plane.

For the resulting multi-dimensional mixed-integer optimization problem, we compare various traditional and new search methods. In particular, we apply multi-start, (variable) neighborhood search, tabu search, simulated annealing, an evolutionary algorithm and an ant colony optimization algorithm. They all have in common that they use the well-known alternate location and allocation algorithm [Cooper, 1964] as core local search function.

We intend to impart a generalized view on these randomized search methods and also examine the efficiency of the different search strategies in solving the multi-connection location-allocation problem, a relatively new instance of the generalized class of location-allocation problems.

Computational results show that the most crucial feature of the heuristics is the ability to combine a diversified search over the whole solution space with an intensified search near the best-known solution.

Keywords: meta-heuristics; location; mixed-integer optimization; location-allocation; randomized search

---

1 Problem Formulation

In this paper we consider a generalized class of location-allocation problems, denoted as *multi facility location problem with generalized objects* (MFLPO) [see Bischoff and Klamroth, 2007b], that can be specified as follows: A set of N new facilities $\mathbf{x}_1, \ldots, \mathbf{x}_N$ must be located in the $\mathbb{R}^2$ plane to minimize the sum of M non-negative convex cost functions, $c_m : \mathbb{R}^2 \to \mathbb{R}$, $m = 1, \ldots, M$. Each cost function may model the expenses for serving one

*Corresponding author, e-mail address: bischoff@am.uni-erlangen.de

object from one of the new facilities. Note, that an object is not necessarily represented by a point in the plane like an *existing facility*, but may model abstract destinations, or as for example in the multi connection location problem discussed later in this section, flows between points in the plane that have to be routed through the new facilities.

The allocation of objects to new facilities is established by the binary variables y_{mn}, $m = 1, \ldots, M$, $n = 1, \ldots, N$, where

$$y_{mn} = \begin{cases} 1 & \text{if object } m \text{ is assigned to new facility } n, \\ 0 & \text{otherwise.} \end{cases}$$

Since every object must be assigned to exactly one new facility, we require that $\sum_{n=1}^{N} y_{mn} = 1$, $m = 1, \ldots, M$. Therefore, the set of all feasible assignments $Y \in \{0,1\}^{M \times N}$ which satisfy this restriction is given by

$$\mathcal{Y} = \Big\{ Y \in \{0,1\}^{M \times N} : \; Y = (y_{mn})_{\substack{m=1,\ldots,M, \\ n=1,\ldots,N}}, \; \sum_{n=1}^{N} y_{mn} = 1, \; m = 1, \ldots, M \Big\},$$

and the number of feasible assignments equals $|\mathcal{Y}| = M^N$. Since we do not consider additional constraints like new facilities with limited capacities, every object is simply assigned to that new facility which minimizes its transportation costs.

The *multi facility location problem with generalized objects* (MFLPO) can be formulated as

$$\begin{aligned} \min \quad & \sum_{m=1}^{M} \sum_{n=1}^{N} y_{mn} c_m(\mathbf{x}_n) \\ \text{s.t.} \quad & \sum_{n=1}^{N} y_{mn} = 1, \quad m = 1, \ldots, M \\ & y_{mn} \in \{0,1\}, \quad m = 1, \ldots, M, \; n = 1, \ldots, N \\ & \mathbf{x}_n \in \mathbb{R}^2, \qquad n = 1, \ldots, N. \end{aligned} \tag{MFLPO}$$

Bischoff and Klamroth [2007b] considered this generalized problem class and proposed two complementary Branch & Bound methods, one branching on the continuous location variables, the other on the discrete assignment variables.

The well-known *(uncapacitated) multi facility location-allocation problem* (MFLP), also denoted as *(generalized) multi Weber problem* is one instance of (MFLPO). It is obtained by specifying the cost functions

$$c_m(\mathbf{x}) = w_m d(\mathbf{a}_m, \mathbf{x}), \quad \forall \mathbf{x} \in \mathbb{R}^2, \; m = 1, \ldots, M,$$

where $d : \mathbb{R}^2 \times \mathbb{R}^2 \to \mathbb{R}$ is a distance function in $\mathbb{R}^2$ and $\mathbf{a}_m \in \mathbb{R}^2$, $m = 1, \ldots, M$ are the so called *existing facilities* with *weights* (demand units) $w_m \in \mathbb{R}_+$. This facility location problem was first formally stated by Cooper [1963], who also showed that it is neither convex nor concave, but may have a huge number of local minima.

Until nowadays the MFLP has received a reasonable amount of attention. The literature which is most relevant for the scope and purpose of this paper is summarized in the

following. For further reading, we refer to Love et al. [1988], Drezner [1995], Francis et al. [1992] and Drezner and Hamacher [2002]. Due to the NP-hardness of this problem [see Megiddo and Supowit, 1984], many solution methods have been proposed to determine at least local optimal or near optimal solutions of large problem instances. The first heuristics have been proposed by Cooper [1964]. Among other methods, the author formulates the well-known alternate location and allocation algorithm. Love and Juel [1982] propose five neighborhood search heuristics. For the corresponding neighborhood structure, tabu search [Brimberg and Mladenović, 1996a] and a variable neighborhood search method [Brimberg and Mladenović, 1996b] has been developed. Using an approximation result Chen [1983] eliminates the assignment variables in the objective function and obtains a continuous approximated problem that can be solved with a quasi-Newton method. Bongartz et al. [1994] present a projection method for the (MFLP). Houck et al. [1996] develop a genetic algorithm for (MFLP) and compare it with several heuristics to obtain a good initial solution for a given (MFLP). A comparison and improvements of a number of heuristics for (MFLP) are provided in Brimberg et al. [2000]. Although only applicable for smaller instances, several exact solution methods have been developed for the (MFLP). Kuenne and Soland [1972] present a Branch & Bound algorithm for (MFLP) under Euclidean distance functions that is based on a partial enumeration of the assignment variables. The solution method by Love and Morris [1975] for (MFLP) under rectangular distance functions applies set reduction results and a p-median solution method for the remaining subproblem. Since under Euclidean distance functions, in an optimal solution of (MFLP) the convex hulls of the subsets of existing facilities that are allocated to the same new facility are disjoint Rosing [1992] proposes an enumeration of all combinations of partitions into N disjoined convex hulls in order to obtain the optimal solution of (MFLP).

In this paper we mainly focus on another instance of (MFLPO), the *multi-connection location-allocation problem* (MCLP). This location problem models the transfer of products between processing facilities. On its transport, every product must pass one *connection location*, of which a given number may be freely located in the plane. In order to minimize the total transportation costs, we are interested in finding optimal connection locations.

More formally, let L existing facilities $\mathbf{a}_1, \ldots, \mathbf{a}_L \in \mathbb{R}^2$ and M *flows* $\mathbf{f}_m = (i_m, j_m)$, $i_m, j_m \in \{1, \ldots, L\}$, $m = 1, \ldots, M$ be given, such that each flow $\mathbf{f}_m$, $m \in \{1, \ldots, M\}$ connects two existing facilities $\mathbf{a}_{i_m}, \mathbf{a}_{j_m}$ with an intensity of $w_m \in \mathbb{R}_+$. Since all flows must be routed through one of the connection locations $\mathbf{x}_n \in \mathbb{R}^2$, $n \in \{1, \ldots, N\}$, the transportation cost caused by flow $\mathbf{f}_m = (i_m, j_m)$ is given by

$$c_m(\mathbf{x}) = w_m\big(d(\mathbf{a}_{i_m}, \mathbf{x}) + d(\mathbf{x}, \mathbf{a}_{j_m})\big), \quad \forall \mathbf{x} \in \mathbb{R}^2,$$

where $d : \mathbb{R}^2 \times \mathbb{R}^2 \to \mathbb{R}$ is assumed to be a metric in $\mathbb{R}^2$.

Altogether, the multi connection location problem can be formulated as

$$\begin{aligned} \min \quad & \sum_{m=1}^{M} \sum_{n=1}^{N} y_{mn} w_m \big(d(\mathbf{a}_{i_m}, \mathbf{x}_n) + d(\mathbf{x}_n, \mathbf{a}_{j_m})\big) \\ \text{s.t.} \quad & \sum_{n=1}^{N} y_{mn} = 1, \quad m = 1, \ldots, M \\ & y_{mn} \in \{0, 1\}, \quad m = 1, \ldots, M, \ n = 1, \ldots, N \\ & \mathbf{x}_n \in \mathbb{R}^2, \qquad n = 1, \ldots, N. \end{aligned} \tag{MCLP}$$

This location problem has previously been considered by Huang et al. [2005] for both the uncapacitated and the capacitated case, where it is denoted as (un-)capacitated N-connection location problem. Huang et al. [2005] show basic properties of (MCLP) and present a location-allocation heuristic which is based on the alternate location and allocation algorithm developed by Cooper [1963]. Bischoff and Klamroth [2007b] apply two different Branch & Bound methods to the (MCLP).

The rest of this paper is organized as follows. In the following section, we give an outline of the subproblem decomposition and the resulting alternate location and allocation algorithm for the (MFLPO) in general, and the (MCLP) in particular. In all heuristics which are described in Section 3, this procedure is applied as local search procedure. Each search method, that is the multi-start method, K-neighborhood search, variable neighborhood search, tabu search, simulated annealing, threshold accepting, the evolutionary algorithm and the ant colony optimization algorithm, is discussed in a separate subsection. Computational results obtained with a huge set of randomly generated (MCLP) example problems to compare the efficiency of the search methods are presented in Section 4.

2 Subproblem Decomposition

Suppose, $\overline{\mathbf{x}}_1, \ldots, \overline{\mathbf{x}}_N \in \mathbb{R}^2$ is a given set of locations for (MFLPO). The remaining subproblem of optimally allocating the objects is

$$\sum_{m=1}^{M} \min\{c_m(\overline{\mathbf{x}}_n),\ n = 1, \ldots, N\}.$$

That is, the optimal objective value with respect to the given locations is simply obtained by allocating that new facility to an object which minimizes its costs.

On the other hand, by fixing the assignment variables $\overline{Y} = (\overline{y}_{mn})_{\substack{m=1,\ldots,M \\ n=1,\ldots,N}} \in \mathcal{Y}$, for the objective function holds:

$$\sum_{m=1}^{M}\sum_{n=1}^{N} \overline{y}_{mn} c_m(\mathbf{x}_n) = \sum_{n=1}^{N}\sum_{m\in\mathcal{M}_n} c_m(\mathbf{x}_n),$$

where $\mathcal{M}_n := \{m \in \{1, \ldots, M\} :\ \overline{y}_{mn} = 1\}$, $n = 1, \ldots, N$ is the set of objects to which the new facility $\mathbf{x}_n$ is allocated. This subproblem is composed of the sum of N independent single-facility problems, in which each new facility $\mathbf{x}_n$ must be located to minimize $\sum_{m\in\mathcal{M}_n} c_m(\mathbf{x}_n)$, $n = 1, \ldots, N$. Since we restricted the cost functions c_m, $m = 1, \ldots, M$ to be convex, we also obtain convex objective functions and the global optimal location can be found, e.g., by applying a quasi-Newton method.

It is well-known, that for (MFLP) the resulting single facility subproblems are generalized Weber problems which are typically solved with the Weiszfeld algorithm [Weiszfeld, 1937] for l_p-distances $p \in (1, \infty)$. In case of the squared Euclidean distance $d(\mathbf{x}, \mathbf{y}) = (y_1 - x_1)^2 + (y_2 - x_2)^2$, the optimal location of the Weber problem equals the center of gravity of objects with weights w_m, positioned at the existing facility locations $\mathbf{a}_m$, $m = 1, \ldots, M$. It is given by

$$\mathbf{x} = \frac{\sum_{m=1}^{M} w_m \mathbf{a}_m}{\sum_{m=1}^{M} w_m}.$$

Also (MCLP) simplifies to N independent generalized Weber problems. By setting

$$w_{ij} = \begin{cases} w_m & \text{if there exists } m \in \{1, \dots, M\} \text{ such that } \mathbf{f}_m = (i, j), \\ 0 & \text{otherwise,} \end{cases}$$

for all $i, j = 1, \dots, L$, and since d is symmetric, for the (MCLP) objective function holds

$$\sum_{i,j=1}^{L} \sum_{n=1}^{N} \overline{y}_{ijn} w_{ij} \big(d(\mathbf{a}_i, \mathbf{x}_n) + d(\mathbf{x}_n, \mathbf{a}_j)\big) = \sum_{n=1}^{N} \sum_{i,j=1}^{L} \big(\overline{y}_{ijn} w_{ij} + \overline{y}_{jin} w_{ij}\big) d(\mathbf{a}_i, \mathbf{x}_n).$$

We thus obtain the sum of N independent Weber objective functions,

$$\sum_{n=1}^{N} \sum_{i=1}^{L} \overline{w}_i d(\mathbf{a}_i, \mathbf{x}_n), \quad \text{where } \overline{w}_i = \sum_{j=1}^{L} \big(\overline{y}_{ijn} w_{ij} + \overline{y}_{jin} w_{ij}\big), \ i = 1, \dots, L.$$

Obviously, a solution of (MFLPO) allocating less than N new facilities results in a function value which is never better than the best function value of a solution, in which all N new facilities are allocated to some objects. Additionally, since only the partition induced by the assignment variables but not the order of the new facilities affects the function value, the indices of the new facilities can be arbitrarily exchanged without changing the quality of the solution. Based on these results, Cooper [1963] showed for the (MFLP) that for M objects and N locations, for which the total number of feasible assignments is $|\mathcal{Y}| = M^N$, it is sufficient to consider a dominating set with a number of

$$S(M, N) = \frac{1}{N!} \sum_{m=0}^{M-1} (-1)^m \binom{N}{i} (N - i)^M$$

assignments only. $S(M, N)$ is the Stirling number of the second kind, it equals the number of ways of partitioning a set of M elements into N nonempty sets.

One possibility to determine the optimal solution of (MFLPO) is enumerating all allocations and solving the corresponding single-facility subproblems. However, since the Stirling number of the second kind increases exponentially with the size of the problem, this method is applicable for very small problems only.

There are mainly two possibilities of restricting the heuristics to search in the dominating subset. Whenever a solution is selected which is not in the dominating set, it could either be modified or replaced. However, both options bias the search process in the sense that some regions of the solution space are reached easier than others. Therefore, we prefer the heuristics to search the whole set of feasible assignments $\mathcal{Y}$ with a total number of M^N elements. Although solutions may eventually be considered where less than N new facilities are assigned, the probabilities are not tampered. Consequently, those search strategies which do not concentrate the search on promising regions on their own are not advantaged, and the search process of the others is not biased.

Based on the decomposition into N single facility location subproblems and the allocation subproblem, respectively, Cooper [1963] developed the alternate location and allocation algorithm for the (MFLP). Note that the alternate location-allocation algorithm ($\mathcal{LA}$) as outlined here can in fact be applied on any instance of the generalized class (MFLPO).

Alternate Location-Allocation Algorithm ($\mathcal{LA}$)

Input: Y: initial assignment

Do:

Location Step:

With respect to Y, solve the N single-facility location problems.

Let X denote the optimal locations.

Allocation Step:

With respect to X, solve the assignment problem.

Let Y denote the optimal assignment.

until a stopping criterion is satisfied.

Output: (X, Y) with objective value F

Since $\mathcal{LA}$ is applied as a subroutine in all of the following heuristics, we denote for brevity by $(X, Y, F) = \mathcal{LA}(\overline{Y})$, that $\mathcal{LA}$ is executed with initial assignment $\overline{Y}$ and returns the local minimal solution (X, Y) with objective value F. Further, $(X, Y, F) = \mathcal{LA}(\overline{\mathcal{Y}})$, where $\overline{\mathcal{Y}} \subseteq \mathcal{Y}$ is a set of assignments, denotes that $\mathcal{LA}$ is applied on all assignments $\overline{Y} \in \overline{\mathcal{Y}}$, and (X, Y) with objective value F is the best local minimum found in all of the runs.

Obviously, it is also possible to initialize this procedure with location variables $\overline{X}$ and exchanging the location and allocation steps, instead of supplying assignment variables $\overline{Y}$ in the beginning. The algorithm has converged to a local minimum and terminates, if either the locations do not change in a location step or the assignments remain the same in an allocation step, respectively.

3 Heuristics

The search methods described in this section are sorted in an ascending order concerning the level of complexity. In order to emphasize the specific search strategy, every heuristic is outlined in a unified scheme, before certain details of the implementation are explained. The reader is encouraged to compare the schematic descriptions among each other to find similarities (e.g., the application of ($\mathcal{LA}$) as local search procedure) and differences in the way, in which the set of assignments is examined.

Remember that ($\mathcal{LA}$) is initialized with an assignment only, and returns a whole local optimal solution of the problem. All single facility subproblems are exclusively solved within the $\mathcal{LA}$ procedure, not in the heuristics. Thus, all of the following search methods can be considered as meta-heuristics, learning exclusively of the local optimal subset of the solution space and providing only assignments that - according to their search strategy - seem promising to result in a better local optimal solution.

Multi-Start Method

The *random restart method*, also known as the *multi-start* version of ($\mathcal{LA}$) [see Cooper, 1964], is the simplest method performing the search concept described above. In order to

possibly obtain better local optimal solutions, ($\mathcal{LA}$) is restarted multiple times with random initial assignments. It can formally be described as follows:

Multi-Start Method

Input: I: number of assignments to evaluate

$(X^\star, Y^\star, F^\star) = \mathcal{LA}(\{Y_i \in \mathcal{Y},\ \text{randomly chosen},\ i = 1, \ldots, I\})$.

Output: $(X^\star, Y^\star, F^\star)$

Neighborhood Search

Assuming that close to local optimal solutions even better local optimal solutions can be found, it seems promising to examine the solutions in a specified *neighborhood* of the best known local optimal solution. In this paper, we define the *K-neighborhood* of an assignment as the set of all assignments which can be obtained by allocating exactly K existing facilities to different new facilities. The set of assignments in the K-neighborhood of an assignment $\overline{Y}$ is denoted by $\mathcal{Y}_K(\overline{Y})$.

Neighborhood search strategies were first realized for (MFLP) in the heuristics H1 to H5 published by Love and Juel [1982]. Evaluating all assignments in the 2-neighborhood was already classified as too time-consuming due to the exponentially increasing neighborhood size. Thus, heuristics H4 and H5 search only a subspace of the 2-neighborhood which can be obtained by allocating exactly two existing facilities to the same new facility, which must be different to those that they were allocated to in the original solution. Therefore, both heuristics are also denoted as two-opt switching methods in Houck et al. [1996].

In the schematic description below, a general K-neighborhood search algorithm is provided. In this version, all assignments in the K-neighborhood are evaluated. If at least one improving solution has been determined, the neighborhood search is restarted with respect to the best of them. If no superior solution was found, the neighborhood size is successively increased up to a given upper bound $K_{\max}$. Alternatively it may also be possible to first evaluate the assignments in all neighborhoods $K = 1, \ldots, K_{\max}$, before moving to the best of all improving assignments.

The number of neighboring assignments increases exponentially with the size of the neighborhood, since the number of neighbors in the K-neighborhood is

$$\binom{M}{K}(N-1)^K.$$

Additionally, most of the $\mathcal{LA}$ subroutines starting with an assignment in a close neighborhood quickly return to the original local optimal solution. Altogether, the main drawback of the K-neighborhood search heuristics is that, on the one hand they easily get stuck in local optimal solutions if K is small, while on the other hand it is numerically impractical to choose a large K, due to the exponentially increasing neighborhood size.

Variable Neighborhood Search

K-Neighborhood Search

Input: $K_{\max}$: maximum neighborhood size
Y: initial assignment

$(X^\star, Y^\star, F^\star) = \mathcal{LA}(Y)$, set $K = 1$.
While $K \leq K_{\max}$ **do:**
$(X, Y, F) = \mathcal{LA}(\mathcal{Y}_K(Y^\star))$
If $F < F^\star$,
$(X^\star, Y^\star, F^\star) = (X, Y, F)$, set $K = 1$.
Otherwise, set $K = K + 1$.

Output: $(X^\star, Y^\star, F^\star)$

To intensify the search near local optimal solutions while also examining diversified solutions in distant neighborhoods in reasonable time, *variable neighborhood search* was suggested by Mladenović and Hansen [1997] as a natural variant of neighborhood search. Here, instead of an exhaustive search in the given neighborhood, a predefined number of randomly chosen neighbors is evaluated only. If no better solution can be found in the specified subset of the current neighborhood, the neighborhood size is increased in order to enable a diversified search. Consequently, this method quickly scans a large region of the solution space and may even accidentally avoid local traps since not all neighboring solutions are evaluated.

Variable neighborhood search has previously been proposed for facility location problems, see Hansen and Mladenovič [1997] and Brimberg and Mladenović [1996a,b] for applications on the p-median problem and the MFLP, respectively.

In this paper we consider the version as described in Brimberg and Mladenović [1996b] for (MFLP). It is interesting to note that Brimberg et al. [2000] reported this method to be the superior search strategy for (MFLP) (in comparison to various other heuristics), which gave consistently the best results in moderate computation time.

Variable Neighborhood Search

Input: I: number of assignments to evaluate per iteration
$K_{\max}$: maximum neighborhood size
Y: initial assignment

$(X^\star, Y^\star, F^\star) = \mathcal{LA}(Y)$, set $K = 1$.
While $K \leq K_{\max}$ **do:**
$(X, Y, F) = \mathcal{LA}(\{Y_i \in \mathcal{Y}_K(Y^\star),\ \text{randomly chosen},\ i = 1, \ldots, I\})$.
If $F < F^\star$,
$(X^\star, Y^\star, F^\star) = (X, Y, F)$, set $K = 1$.
Otherwise, set $K = K + 1$.

Output: $(X^\star, Y^\star, F^\star)$

This algorithm can easily be extended by specifying additional parameters. For example, a minimum neighborhood size $K_{\min} \in \mathbb{Z}_+$ could be specified, instead of starting in the 1-neighborhood only. Also the increase of the neighborhood size may be accelerated by defining a step size $\Delta_K \in \mathbb{Z}_+$, instead of evaluating solutions in every neighborhood $K = 1, \ldots, K_{\max}$. Additionally, the number of considered neighbors could be varied with the size of the neighborhood instead of constantly evaluating I neighbors only. For further extensions of this search method we refer to Brimberg and Mladenović [1996b].

Tabu Search

Both neighborhood search heuristics are greedy in the sense that they only accept a new solution if it is better than the currently best known solution of the problem. They are unable to climb out of local traps. To overcome this difficulty, *tabu search* can be applied, which moves to the best solution in the specified neighborhood structure even if it is inferior than the current solution. In order to attain new assignments, previous moves must be saved in a tabu list of predefined length using the FIFO method. Feasible are only those moves which are not in the current tabu list. Thereby, the method is able to move to inferior solutions, expecting to reach a better solution in the end. Typically, the search is stopped if a maximum number of moves without improvement has been performed.

The original concept of tabu search as an extension for other heuristics in order to enhance the search strategy has originally been proposed by Glover [1986]. For further information on tabu search, we refer to Glover and Kochenberger [2003].

Brimberg and Mladenović [1996a] consider a tabu search variant of the 1-neighborhood search for (MFLP) which can be regarded as an enhanced tabu search version of the H3 heuristic by Love and Juel [1982], with the exception, that no $\mathcal{LA}$ local search is applied. Instead, tabu search is supposed to realize the local search itself. That method, however, gave poor results in comparison to other search strategies for (MFLP) [see Brimberg et al., 2000]. For this reason and since we restrict ourselves to methods applying $\mathcal{LA}$, searching the assignments, and learning from the local optimal solutions of the problem only, we considered a different version which has previously been considered by Brimberg and Mladenović [1996b] for (MFLP).

Only the assignment $\overline{Y}$ is stored in the tabu list. We do not explicitly forbid a whole subset of the search space $\mathcal{Y}$ as it is typically the case in the tabu search method. However, since $\overline{Y}$ is a local optimal assignment and all assignments are locally optimized by $\mathcal{LA}$ before verifying its feasibility, we implicitly forbid the whole subset of assignments $Y \in \mathcal{Y}$ which yield $\overline{Y}$ when applying the local search.

This tabu search strategy can be regarded as an adaption of the variable neighborhood search within the tabu search framework. The inner loop in fact corresponds to the variable neighborhood algorithm, with the difference that it may also move to inferior neighboring assignments if no better solution has been found. Once a maximum number of non-improving steps has been reached, the search process terminates.

Simulated Annealing and Threshold Accepting Method

Simulated annealing is a different search concept which is also capable of escaping from local minima. Its first application as a solution method for combinatorial optimization problems goes back to Kirkpatrick et al. [1983] and Cerny [1985]. Its ease of implementation, its

Tabu Search

Input: I: number of assignments to evaluate per iteration
$J_{\max}$: maximum number of non-improving steps
$K_{\max}$: maximum neighborhood size
Y: initial assignment

$(\overline{X}, \overline{Y}, \overline{F}) = \mathcal{LA}(Y)$, set $K = 1$, $J = 0$.
While $J \leq J_{\max}$ **do:**
  While $K \leq K_{\max}$ **do:**
    $(X, Y, F) = \mathcal{LA}(\{Y_i \in \mathcal{Y}_K(\overline{Y}),$ randomly chosen, $i = 1, \ldots, I\})$.
    If Y is not tabu and $F < \overline{F}$,
      set $(\overline{X}, \overline{Y}, \overline{F}) = (X, Y, F)$, $K = 1$, $J = 0$, insert $\overline{Y}$ in the tabu list,
    otherwise,
      set $K = K + 1$.
  Let $(\widetilde{X}, \widetilde{Y}, \widetilde{F})$ denote the best non-tabu solution
  found in the K-neighborhoods of $\overline{Y}$, $K = 1, \ldots, K_{\max}$.
  Set $(\overline{X}, \overline{Y}, \overline{F}) = (\widetilde{X}, \widetilde{Y}, \widetilde{F})$, set $K = 1$, $J = J + 1$, insert $\overline{Y}$ in the tabu list.

Output: $(X^\star, Y^\star, F^\star)$, best local optimal solution determined

ability to escape local optima and convergence properties have made it a popular technique over the last decades. Simulated annealing has also been applied on location problems [see e.g. Ernst and Krishnamoorthy, 1999]. For an overview of theoretical development and application domains of simulated annealing, we refer to the textbook van Laarhoven and Aarts [1987].

At each iteration of the simulated annealing algorithm, an *acceptance condition* is verified in order to decide whether to move to a new solution or to continue the search from the current one. Improving solutions are always accepted, while inferior solutions are accepted with a certain probability, in the hope of moving out of a local trap. Typically, this probability is non-increasing with the number of iterations.

Simulated annealing has an analogy to the physical annealing of solids, which in general reaches a lower state of energy if cooled sufficiently slow. The Metropolis acceptance criterion [Metropolis et al., 1953], which models the change of states in thermodynamic systems, is used to decide whether moving to inferior solutions is feasible. Therefore, the parameter which controls the probability of accepting non-improving solutions, given by the Boltzmann factor $\exp(-\Delta E/T)$, is denoted as *temperature* T. In terms of thermodynamic systems, ΔE denotes the change in energy, where in the optimization method it is the difference between the current objective value $\overline{F}$ and the objective value of the new solution F.

In order to implement a cooling schedule which does not depend on absolute objective values, we implemented a slightly different acceptance condition than the one that is typically used in simulated annealing. Instead of the absolute difference between the objective values, we selected the relative difference $\Delta F = (\overline{F} - F)/\overline{F}$. A new solution is thus accepted

with the probability

$$P = \begin{cases} 1 & \text{if } F < \overline{F}, \\ \exp\left(-\frac{\Delta F}{T}\right) & \text{otherwise.} \end{cases}$$

where T is the current temperature. Starting with an initial temperature $T_{\max}$, it is successively reduced by a constant factor $\lambda \in (0,1)$ until a lower bound $T_{\min}$ is reached and the algorithm terminates.

Simulated Annealing / Threshold Accepting Method

Input: $T_{\max}$: initial temperature
$T_{\min}$: final temperature
$K_{\max}$: maximum neighborhood size
Y: initial assignment
acceptance condition
cooling schedule

$(\overline{X}, \overline{Y}, \overline{F}) = \mathcal{LA}(Y)$, set $T = T_{\max}$.
While $T \geq T_{\min}$ **do:**
  $(X, Y, F) = \mathcal{LA}(Y \in \mathcal{Y}_K(\overline{Y}),\ K \in \{1, \ldots, K_{\max}\},$ randomly chosen).
  If the acceptance condition is satisfied,
    set $(\overline{X}, \overline{Y}, \overline{F}) = (X, Y, F)$.
  Update T according to the cooling schedule.

Output: $(X^\star, Y^\star, F^\star)$, best local optimal solution determined

---

The *threshold accepting method* [Moscato and Fontanari, 1990] is a search strategy which is strongly related to simulated annealing. In fact, both methods only differ in the acceptance condition. In the threshold accepting method, the current solution with objective value $\overline{F}$ is replaced by the new solution with objective value F, if $\overline{F} - F \leq T$, where T also is a deterministic value which is typically non-increasing with the number of iterations. As in the simulated annealing method, we replaced $\overline{F} - F$ by the relative difference ΔF. In terms of probabilities, our threshold acceptance condition can therefore be specified as follows. A new solution is accepted with the probability

$$P = \begin{cases} 1 & \text{if } T \geq \Delta F, \\ 0 & \text{otherwise,} \end{cases}$$

where T is the threshold in the current iteration. In comparison to simulated annealing, only few applications of the threshold accepting method are reported [Glover and Kochenberger, 2003]. We applied both simulated annealing and the threshold accepting method on the (MCLP), see Section 4 for parameter settings and the results.

Evolutionary Algorithm

In contrast to the previous heuristics, we now consider *population-based* search strategies. Here, a whole set of solutions is evaluated and observed simultaneously in order to gather more information about the solution space.

The denotation *evolutionary algorithm* is an umbrella term for all heuristics which simulate a neo-Darwinian evolutionary process. Besides the well-known *genetic algorithm* [Holland, 1975] there exist other evolutionary algorithms having slightly different features and key aspects. Among others, there is the *memetic algorithm* [Moscato, 1989], a hybrid method which basically is obtained by combining a genetic algorithm with a local search procedure. Since specific concepts of several subclasses are aggregated in the heuristic explained below, we decided to use the general naming.

All evolutionary algorithms have in common that they examine a whole set of solutions in parallel in each iteration. According to their *fitness*, which typically depends on their objective values, solutions are selected for *recombination*, assuming that a composition of good elements eventually yields even better solutions. In order to successively improve the quality of the whole set, inferior solutions are rejected and replaced by new, typically better ones. New solutions can additionally be *mutated*, that is, with a certain probability, every element may be changed randomly in order to preserve a reasonable level of diversity and to escape from sub-optimal regions. Characteristic of the genetic algorithm is that there is a clear distinction between the *genotype* and the *phenotype* of a solution. The genotype is typically a representation of the solutions in a different alphabet, most frequently binary values, while the phenotype denotes their original representation. The evolutionary operators are applied on the genotype, assuming that correlations among the solutions can be exploited which could not have been identified in the original representation.

Many evolutionary strategies have been applied on location problems. For example, Jaramillo et al. [2002] evaluates the performance of genetic algorithms for the (un-)capacitated fixed charge problem, the maximum covering problem and the medianoid and centroid problem. Genetic algorithms have also been developed for location problems in the continuous space. Recently, a genetic algorithm is applied as a subroutine in a solution method for Weber problems in the presence of barriers [Bischoff and Klamroth, 2007a]. For useful references on evolutionary algorithms in general, and genetic algorithms in particular, we refer to Reeves [1997].

Similar to our evolutionary algorithm, the genetic algorithm proposed by Houck et al. [1996] for the (MFLP) performs $\mathcal{LA}$ in order to improve the quality of every solution. The main difference to our evolutionary algorithm is that, besides the generalized problem class (MFLPO), Houck et al. [1996] use the coordinates of the new facilities in the continuous space as a representation of an individual, while we suggest to consider the assignment variables instead. Note, that both sets of variables are completed to solutions by the $\mathcal{LA}$ procedure, only the order of the location and allocation step must be chosen accordingly. However, the two different representation schemes entail the application of other recombination and mutation operators which results in completely different search strategies.

We decided to use integer numbers to represent the assignment variables. For example, the vector $(1, 2, 1, 3, 2)$ corresponds to the assignment variables

$$y_{mn} = \begin{cases} 1 & \text{if } (m,n) \in \{(1,1),(2,2),(3,1),(4,3),(5,2)\}, \\ 0 & \text{otherwise,} \end{cases}$$

for all $m = 1, \ldots, 5$, $n = 1, \ldots, 3$.

Concerning the selection operator, we applied *tournament selection*, which is implemented as follows. A subset of $\bar{I} \leq I$ assignments is randomly chosen from the current set of assign-

Evolutionary Algorithm

Input: I: number of assignments considered simultaneously
J: number of assignments to evaluate per iteration
selection, recombination, mutation operators
reinsertion operator
stopping criterion

$(X_i, Y_i, F_i) = \mathcal{LA}(Y_i \in \mathcal{Y},$ randomly chosen$)$, $i = 1, \ldots, I$.
Do:
Apply selection, recombination and mutation operators
to determine a set of new assignments $\overline{Y}_1, \ldots, \overline{Y}_J$.
$(\overline{X}_j, \overline{Y}_j, \overline{F}_j) = \mathcal{LA}(\overline{Y}_j)$, $j = 1, \ldots, J$.
Apply reinsertion operator to replace some assignments Y_i
by new assignments $\overline{Y}_j$, $i \in \{1, \ldots, I\}$, $j \in \{1, \ldots, J\}$
until the stopping criterion is satisfied.

Output: $(X^\star, Y^\star, F^\star)$, best local optimal solution determined

ments, out of which the best assignment is then selected for recombination. Tournament selection is a *rank-based* operator, since the actual objective values are neglected. In order to obtain J (not necessarily different) assignments for recombination J tournaments are performed per iteration. Note that it is not mandatory to choose $J \leq I$.

The selected assignments are pairwise recombined using *uniform crossover*. Here, for each pair, a binary crossover mask $\mathbf{m}_{\text{cross}} \in \{0,1\}^M$ is stochastically generated, using a Bernoulli distribution with a parameter $p_{\text{cross}} \in (0,1)$. All elements $m \in \{1, \ldots, M\}$ for which $\mathbf{m}_{\text{cross}}$ equals one are exchanged. Other crossover operators, as for example single- and multi-point crossover exhibit a strong *positional bias*, meaning that they imply correlations among the position of a gene and its value. However, since in this type of problem no information can be drawn from the number of an object for its allocation, we decided to use the uniform crossover. The same holds for the mutation operator. Also mutation is implemented using a stochastically generated mask on the basis of a Bernoulli distribution with a parameter $p_{\text{mutate}} \in (0,1)$, the so-called mutation mask $\mathbf{m}_{\text{mutate}} \in \{0,1\}^M$. Every element $m \in \{1, \ldots, M\}$, for which $\mathbf{m}_{\text{mutate}}$ equals one, is set to a random value $n \in \{1, \ldots, N\}$, using the uniform distribution. Additionally p_{mutate} could be varied during the search process in order to augment the exploration in the beginning, and to intensify the search near good solutions in the end of the algorithm.

After applying $\mathcal{LA}$ on the new assignments, the $\overline{J}$ best of them replace the $\overline{J}$ worst of the previous set of assignments, $\overline{J} \leq J$. By requiring $\overline{J} < \min\{I, J\}$, we ensure an elitist strategy in the sense that the best-known assignment always remains in the current set.

The algorithm terminates if the best known solution could not be improved within a given number of Δ_T iterations. Additionally, a *homogeneity termination condition* is verified. That is, it is examined whether the number of assignments in the current set which have the same objective value as the best-known solution exceed a given threshold Δ_I.

Ant Colony Optimization Algorithm

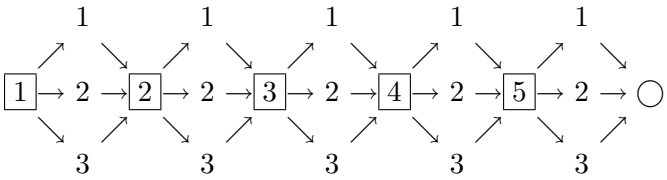
While the evolutionary algorithm described above has a pool of solutions with elements out of which new solutions are constructed, ant colony optimization algorithms select solutions according to explicitly specified probabilities.

Inspired by real ants which use pheromone trails as a medium for communication to determine shortest paths, Dorigo [1992] proposed the first ant-based algorithm, named *ant system* for the *traveling salesman problem* (TSP). Out of this nature-inspired search strategy, a metaheuristic approach for combinatorial optimization problems emerged in the last decade, the so-called *ant colony optimization* [Dorigo and Caro, 1999].

The search concept can be described as follows. In the *exploration phase*, no dominating pheromone trails exist and ants basically perform random search to find a solution. Its counterpart is called *exploitation phase*, where the search is focused to the promising regions of the search space, after good solutions have been determined and the corresponding pheromone trails have been intensified.

This heuristic approach has been applied on various combinatorial optimization problems, for an overview we refer to Dorigo and Stützle [2004]. Also for discrete location problems in graphs, ant-based algorithms have been successfully applied [e.g. Venables and Moscardini, 2006]. A combination of an ant colony optimization heuristic with a local search procedure is also suggested by Dorigo and Stützle [2004], which in fact gave excellent results in practice [e.g. Stützle and Hoos, 2000]. The ant colony optimization algorithm presented in this paper bears most similarities to the ant-based heuristic for the data clustering problem developed by Kao and Cheng [2006]. To point out the strong interrelation between these two problems, note that the cluster centers may be regarded as new facilities. While clustering is performed by the ant-based algorithm, the quality of the clusters is evaluated in a separate procedure.

In order to apply ant colony optimization, the whole solution space, or, in our case, the whole feasible set for the assignment variables, must be represented in a network. We selected a representation that contains M *decision nodes*, each corresponding to one object $m \in \{1, \ldots, M\}$, having N outgoing arcs specifying to which new facility $n \in \{1, \ldots, N\}$ this object is allocated. For an example, consider the network for $M = 5$ objects and $N = 3$ new facilities:



All of these ant-based algorithms have in common that they probabilistically construct solutions by successively moving through the network. Nodes are selected according to *transition probabilities* that are computed by *pheromone trail* values and, optionally, additional *heuristic information*. In detail, the transition probability of selecting node n at decision node m, and thus assigning new facility n to object m, is given by

$$p_{mn} = \frac{\tau_{mn}^{\alpha} \cdot \eta_{mn}^{\beta}}{\sum_{n=1}^{N} \tau_{mn}^{\alpha} \cdot \eta_{mn}^{\beta}}, \quad m = 1, \ldots, M,\ n = 1, \ldots, N,$$

where τ_{mn} is the pheromone trail value and η_{mn} is the value containing the heuristic information and $\alpha, \beta > 0$ are two parameters wich determine the relative influence of each of the

two factors.

Heuristic information was implemented in the original ant system algorithm for the TSP [Dorigo, 1992], in order to provide additional information about the problem structure. There, η_{ij} is set to the distance between city i and city j in order to amplify the selection of a closer city in each step. In (MFLPO) we can supply similar information about the geometry of the problem. An estimate of the benefit of allocating a new facility to an object is provided by setting

$$\eta_{mn} = \left(c_m(\mathbf{x}_n^\star)\right)^{-1}, \quad m = 1, \ldots, M,\ n = 1, \ldots, N,$$

where $\mathbf{x}_1^\star, \ldots, \mathbf{x}_N^\star$ are the local optimal locations corresponding to the best-known solution determined by the $\mathcal{LA}$ local search procedure. We select these best-known locations in order to intensify the search in the region corresponding to the best-known assignments. Therefore, η_{mn}, $m = 1, \ldots, M$, $n = 1, \ldots, N$ is updated each time an improving solution is found.

In the ant colony optimization heuristic of Kao and Cheng [2006] for the clustering problem, heuristic information is implemented in a similar way. In particular, η is set to the reciprocal of the Euclidean distance between the data point to the cluster center. In contrast to our method, Kao and Cheng [2006] compute η in every single construction step of the assignments with respect to the centers of the clusters defined so far, which increases the probability to obtain compressed clusters with a lower objective value.

The idea of supplying additional information of the locations when selecting the allocations is a completely new concept in comparison to all previously described search strategies. In order to examine the consequences of this extension, we evaluated two versions of our search strategy, one in which we selected $\alpha = \beta = 1$, such that both factors have equal influence, and one without any heuristic information, i.e. $\alpha = 1$, $\beta = 0$.

Ant Colony Algorithm

Input: J: number of assignments to evaluate per iteration
decision rule
update rule
stopping criterion

Set all assignments equiprobable.
Do:
By means of a decision rule based on the assignment probabilities,
determine a new set of J feasible assignments $\overline{Y}_1, \ldots, \overline{Y}_J$.
$(\overline{X}_j, \overline{Y}_j, \overline{F}_j) = \mathcal{LA}(\overline{Y}_j)$, $j = 1, \ldots, J$.
According to an update rule and wrt. $(\overline{X}_j, \overline{Y}_j, \overline{F}_j)$, $j = 1, \ldots, J$,
update the assignment probabilities
until the stopping criterion is satisfied.

Output: $(X^\star, Y^\star, F^\star)$, best local optimal solution determined

---

Our ant-based algorithm can be further classified as an *ant colony system*, which is an improved version of the original ant system [Dorigo and Gambardella, 1997]. The specific features of this method are explained in the following.

The decision rule which is used in the original ant system [Dorigo, 1992] is based exclusively on the transition probabilities and is known as *random proportional rule*. In the ant colony system, however, the *pseudo-random proportional rule* is applied. Here, in order to augment the utilization of the gathered information, the node with the highest transition probability is selected with probability $q_0 \in [0,1]$. Only with probability $(1-q_0)$ the *random proportional rule* is applied. Obviously, by setting $q_0 = 0$ the pseudo-random proportional rule results in the random proportional rule.

Further, in this ant-based algorithm the pheromone trail values are updated according to a strong elitist strategy. After constructing and evaluating a number of J assignments, only the values of the currently best-known assignment are increased. Formally, let the assignment with function value $F^\star$ be given by

$$y^\star_{mn} = \begin{cases} 1 & \text{if } n = n_m \in \{1,\dots,N\}, \\ 0 & \text{otherwise,} \end{cases} \qquad m = 1,\dots,M.$$

In ant colony systems the pheromone trail values are changed as follows:

$$\tau_{mn_m} := (1-\rho)\cdot\tau_{mn_m} + \rho\cdot(F^\star)^{-1}, \quad m = 1,\dots,M.$$

The parameter $\rho \in [0,1]$ models the *intensification* of the trail by the new pheromone with respect to the existing amount. We decided to intensify the pheromone trail values of the local optimal assignments obtained after applying the $\mathcal{LA}$ procedure. As an alternative, the values of the original assignments could be intensified which were selected in the network and used to initialize $\mathcal{LA}$.

In order to augment diversification and to avoid selecting the same assignment multiple times, the pheromone trail values of every constructed assignment is artificially reduced by setting

$$\tau_{mn_m} := (1-\xi)\cdot\tau_{mn_m} + \xi\cdot\tau^0_{mn_m}, \quad m = 1,\dots,M,$$

where $\xi \in (0,1)$ is a given parameter value and τ^0_{mn} are the initial pheromone trail values, $n = 1,\dots,N$, $m = 1,\dots,M$. According to this formula, the pheromone trail values always remain positive. Consequently, every assignment can be constructed with a positive probability. Note, that this is an analogy to the evolutionary algorithm, where the mutation operator ensures that theoretically every assignment can be reached.

The search process is stopped if the best known solution could not be improved within a given number of Δ_T iterations.

4 Computational Results

All search strategies have been implemented in Matlab (Release 14) and have been evaluated on a Sun Fire V20z machine with two AMD Opteron 2.4GHz CPUs and 8GB memory. We selected (MCLP) as a special case of (MFLPO) for computational testing and chose the squared Euclidean metric as underlying distance function. Here, the minima of the resulting single facility Weber problems, i.e. the centers of gravity, can be computed with little effort, in contrast to other distance functions as, for example the Euclidean metric, where typically the Weiszfeld algorithm is applied to solve the single facility Weber problems. Therefore,

larger test problems can be evaluated more quickly and we obtain a better comparison of the search strategies, since the variations in the computation time for solving the location subproblems play a minor role as it would be the case, if an iterative procedure had been applied.

The set of test problems was generated as follows. Each coordinate of the existing facilities $\mathbf{a}_l$, $l = 1, \ldots, L$ was randomly selected from $\{0, \ldots, 1000\}$. Between every pair of existing facilities $\mathbf{a}_i$, $\mathbf{a}_j$, $i, j \in \{1, \ldots, L\}$, $i < j$, a flow was defined with an intensity randomly chosen from $\{5, \ldots, 25\}$. The number of connection locations was set to $\lceil \frac{1}{2} L \rceil$. The problem sizes range from five existing facilities with ten flows to 50 existing facilities with 1225 flows, respectively. See Table 1 for an overview.

Table 1: Test Problem Sizes

Type	M	L	N
1	10	5	3
2	15	6	3
3	21	7	4
4	28	8	4
5	36	9	5
6	45	10	5
7	55	11	6
8	66	12	6
9	78	13	7

Type	M	L	N
10	91	14	7
11	105	15	8
12	120	16	8
13	136	17	9
14	153	18	9
15	171	19	10
16	190	20	10
17	231	22	11
18	276	24	12

Type	M	L	N
19	325	26	13
20	378	28	14
21	435	30	15
22	595	35	18
23	780	40	20
24	990	45	23
25	1225	50	25

The performance of every method with respect to the problem sizes are presented in Tables 2, 3, 4 and 5. Since five test problems of each size have been generated and each of them has been solved ten times with every heuristic, the values in Tables 2, 3 and 5 are averaged over a total of 50 runs.

In the tables, the methods are abbreviated as follows:

MS	:	Multi-Start Method
KNS	:	K-Neighborhood Search
VNS	:	Variable Neighborhood Search
TS	:	Tabu Search
SA	:	Simulated Annealing
TA	:	Threshold Accepting
EA	:	Evolutionary Algorithm
ACO	:	Ant Colony System without Heuristic Information
ACO+	:	Ant Colony System with Heuristic Information

We tried to select the parameter values in such a way that the computation times and also the number of $\mathcal{LA}$-calls of all search methods are within the same order of magnitude in order to better compare their efficiencies. However, due to the different search concepts and termination conditions, this was only manageable up to a certain degree. For example, in the K-neighborhood search method the number of $\mathcal{LA}$-calls is predefined and increases exponentially with the size of the problem, while the evolutionary algorithm and the ant

colony algorithm stops if no improving solution has been found after a given number of Δ_T iterations. The detailed parameter settings for all heuristics are discussed in the following.

Since the genetic algorithm starts with a whole set of assignments while the ant colony algorithm and the random restart heuristic require none, the most equitable option is to provide randomly generated initial assignments. In detail, to initialize the search, an assignment $Y \in \mathcal{Y}$ is given by

$$Y = (y_{mn})_{\substack{m=1,\dots,M,\\ n=1,\dots,N}},\ y_{mn} = \begin{cases} 1 & \text{if } n = \overline{n}_m, \\ 0 & \text{otherwise,} \end{cases}\ m = 1,\dots,M,\ n = 1,\dots,N,$$

where $\overline{n}_m \in \{1,\dots,N\}$, $m = 1,\dots,M$ are randomly selected.

In the multi-start method, a total number of $I = 2M$ assignments is evaluated, each assignment is randomly chosen as described above.

Due to the huge problem sizes, only the 1-neighborhood search could be evaluated. Note that $K = 1$ already yields a total number of 29400 neighbors per assignment for the test problems of type 25. For the 2-neighborhood search, the number of neighbors per assignment increases to $431.8566 \cdot 10^6$. Even heuristic H5 of Love and Juel [1982], searching a subset of the 2-neighborhood only, is not practicable since a lower bound on the number of assignments in this subset is $\frac{1}{2}(M-1)M \cdot (N-2)$, which is $17.243100 \cdot 10^6$ for problems of size 25.

For variable neighborhood search, we selected parameters similar to the setting proposed by Brimberg et al. [2000]. The authors reported variable neighborhood search to be the superior search strategy for (MFLP) in comparison to the projection method by Bongartz et al. [1994], the tabu search method by Brimberg and Mladenović [1996b], p-median plus Weber [Hansen et al., 1998] and the genetic algorithm by Houck et al. [1996]. We set $I = 1$, such that one solution is evaluated in each neighborhood only and $K_{\max} = M$ in order to enable a diversified search.

Also in the tabu search method, only one solution is evaluated in each neighborhood, i.e. $I = 1$. We reduced the maximum neighborhood size to $K_{\max} = \lfloor \frac{1}{4}M \rfloor$, and therefore selected $J_{\max} = 4$ such that four consecutive non-improving steps to inferior local optimal solutions are allowed. The three preceding local optimal solutions are stored in the tabu list such that dropping back to recently considered local optimal assignments is impossible.

In the simulated annealing method, we selected a maximum neighborhood size of $K_{\max} = \lceil \frac{1}{3}M \rceil$. The initial temperature is set to $T_{\max} = 1$, and the final temperature to $T_{\min} = 10^{-8}$. Starting with $T_{\max}$, the temperature is reduced by the factor

$$\lambda = \exp\left(\frac{\log(T_{\min}) - \log(T_{\max})}{2 \cdot M - 1}\right)$$

in each iteration, until $T_{\min}$ is reached and the termination condition is satisfied. Based on this cooling schedule, a total number of $2M$ assignments are evaluated. The same parameter settings were used for the threshold accepting method.

The number of simultaneously considered assignments in the evolutionary algorithm is set to $I = K$. The same number of assignments is selected for recombination, i.e., $J = K$. Recall that the tournament selection operator is applied, and therefore good assignments may be selected multiple times. The number of competitors per tournament is set to $\overline{I} = \lceil e^{-1} \cdot K \rceil$. Further, the number of old assignments replaced by new ones is set to $\overline{J} = \min(\lceil \frac{9}{10}K \rceil, K-1)$, in order to guarantee that the best-known assignment never gets

lost. Concerning the generation of the recombination and mutation masks, we selected $p_{\text{cross}} = \frac{1}{3}$ and $p_{\text{mutate}} = \frac{2}{M}$. The maximum number of iterations without improvement is set to $\Delta_T = 100$ iterations. Finally, by specifying $\Delta_I = \lceil \frac{19}{20} K \rceil$, the algorithm stops if 95% of the currently considered assignments have the same objective value as the best-known solution.

In each iteration of the ant colony system, $J = \frac{1}{2}K$ assignments are evaluated. The parameter for controlling the pseudo-random proportional rule is set to $q_0 = \frac{1}{10}$. We selected the evaporation rate $\rho = \frac{9}{10}$. Further, ξ, the factor for weakening already considered assignments, is set to $\frac{1}{10}$. In the end of the first iteration, the pheromone values are initialized by setting $\tau_{mn}^0 = MF^\star$, $m = 1, \ldots, M$, $n = 1, \ldots, N$, where $F^\star$ is the function value of the currently best-known local optimal solution. Concerning the exponents α and β, we considered the two different settings: $\alpha = 1$, $\beta = 0$, ignoring the heuristic information (ACO), and $\alpha = 1 = \beta = 1$ using heuristic information (ACO+). Both versions terminate after $\Delta_T = 100$ iterations without improvement.

Table 2: Average Computation Time in Seconds.

	MS	KNS	VNS	TS	SA	TA	EA	ACS	ACS+
1	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.08	0.07
2	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.09	0.08
3	0.03	0.04	0.02	0.02	0.02	0.02	0.03	0.13	0.11
4	0.05	0.05	0.04	0.03	0.03	0.03	0.04	0.17	0.14
5	0.09	0.11	0.06	0.05	0.05	0.05	0.08	0.30	0.25
6	0.14	0.13	0.09	0.08	0.08	0.08	0.11	0.35	0.29
7	0.25	0.28	0.16	0.12	0.14	0.14	0.27	0.55	0.39
8	0.32	0.36	0.22	0.17	0.18	0.18	0.28	0.61	0.45
9	0.50	0.66	0.38	0.31	0.30	0.30	0.49	1.32	0.88
10	0.69	0.80	0.44	0.39	0.36	0.36	0.55	1.10	0.88
11	0.96	1.36	0.68	0.54	0.51	0.50	0.94	1.45	1.12
12	1.33	1.49	0.77	0.61	0.64	0.64	0.70	1.56	1.11
13	1.78	2.52	1.23	0.94	0.92	0.91	1.51	2.60	1.96
14	2.39	2.80	1.58	1.23	1.16	1.16	2.00	3.15	2.02
15	3.07	4.93	2.06	1.46	1.54	1.52	2.61	3.51	2.56
16	3.77	4.85	2.55	2.18	1.84	1.85	2.99	4.70	3.11
17	6.14	8.29	3.84	2.99	2.87	2.87	4.00	6.91	4.41
18	9.52	12.41	5.96	4.74	4.27	4.27	5.55	9.06	5.26
19	14.77	18.47	9.34	7.77	6.24	6.28	7.35	14.87	9.20
20	20.39	25.62	13.99	9.61	8.76	8.72	9.32	18.78	11.05
21	29.09	39.88	18.23	13.60	12.46	12.31	11.70	22.47	14.80
22	64.31	81.92	41.03	33.06	29.22	29.29	27.06	58.42	32.15
23	129.05	152.29	80.66	61.20	49.29	49.16	35.40	84.78	56.06
24	239.10	333.28	143.45	134.75	91.73	92.32	62.55	171.60	101.35
25	408.72	548.10	262.79	226.75	153.28	153.88	88.20	230.73	148.68
avr	37.46	49.63	23.58	20.10	14.64	14.67	10.55	25.57	15.94

Table 3: Average Number of $\mathcal{LA}$-Calls.

	MS	KNS	VNS	TS	SA	TA	EA	ACS	ACS+
1	20	25	15	17	21	21	18	207	206
2	30	40	22	21	31	31	14	211	212
3	42	96	32	40	43	43	65	214	214
4	56	115	47	49	57	57	68	248	223
5	72	198	57	68	73	73	117	321	320
6	90	208	66	84	91	91	146	325	322
7	110	356	89	96	111	111	277	354	325
8	132	418	114	120	133	133	268	387	345
9	156	640	144	162	157	157	379	564	491
10	182	732	153	191	183	183	396	471	490
11	210	1064	191	221	211	211	576	507	491
12	240	1093	182	228	241	241	393	502	463
13	272	1582	254	282	273	273	739	637	647
14	306	1615	284	314	307	307	883	686	600
15	342	2564	317	328	343	343	1030	667	641
16	380	2321	357	451	381	381	1077	831	758
17	462	3316	408	486	463	463	1190	950	839
18	552	4263	526	616	553	553	1355	981	796
19	650	5392	639	837	651	651	1569	1312	1159
20	756	6514	800	861	757	757	1637	1430	1163
21	870	8961	845	1015	871	871	1793	1363	1262
22	1190	13173	1132	1472	1191	1191	2903	1996	1797
23	1560	19076	1672	2062	1561	1561	2744	2257	2224
24	1980	32166	1996	3143	1981	1981	3713	3008	2775
25	2450	42861	2823	3869	2451	2451	3931	2912	3039
avr	524	5952	527	681	525	525	1091	934	872

In Table 2 the computation times of all methods with respect to the different problem sizes are presented. Each value is averaged over the five test problems and the ten runs. We can see that the 1-neighborhood search has the peak values for all larger problem sizes and also the multi-start method has a relatively high computation time. The evolutionary algorithm terminates relative quickly and reaches the lowest average computation time of 10.55 seconds. All other methods took about 150 to 250 seconds for the largest problem instances on the average and about 15 to 25 seconds on the total average. The relatively high computation times of the ant colony optimization method (ACO and ACO+) for the smaller problem sizes can be explained by its termination condition. Although the best-known solution is found in relatively view iterations, an additional number of 100 iterations without improvement, and the corresponding $100K$ $\mathcal{LA}$-calls (cf. Table 3) must be performed before the termination condition is satisfied. The evolutionary algorithm has little computation times for the smaller scale problems, since besides this stopping criterion it additionally terminates at a certain degree of homogeneity, which is quickly reached for

Table 4: Total Number of Times the Best-Known Solution was Found.

	MS	KNS	VNS	TS	SA	TA	EA	ACS	ACS+
1	44	47	43	42	46	45	41	50	50
2	48	28	47	37	45	47	37	50	50
3	47	39	45	38	46	43	39	50	42
4	36	24	33	19	28	30	14	41	34
5	30	43	46	45	48	49	38	50	48
6	36	24	44	44	47	45	31	50	46
7	32	25	40	32	34	34	39	44	37
8	16	14	33	28	22	23	24	38	30
9	7	16	29	22	21	22	16	33	31
10	16	10	36	32	38	40	16	45	34
11	8	6	11	15	16	8	8	17	21
12	9	10	34	34	43	45	18	45	37
13	8	6	36	21	25	26	22	35	20
14	9	17	33	34	32	31	27	35	33
15	2	1	29	19	26	24	11	26	19
16	1	0	11	16	6	6	8	11	15
17	2	13	29	28	29	28	19	27	27
18	3	6	33	29	24	24	13	19	16
19	0	1	5	7	2	5	2	7	12
20	0	1	11	7	6	6	5	8	7
21	0	2	16	21	14	19	7	12	10
22	0	3	3	6	6	0	6	0	1
23	0	3	4	8	2	4	2	2	5
24	0	0	2	3	1	0	1	0	0
25	0	0	3	6	1	1	1	0	1
$\sum$	354	339	656	593	608	605	445	695	626

smaller scale problems.

Note that the average number of $\mathcal{LA}$-calls displayed in Table 3 is not always proportional to the computation times. This is due to the fact that $\mathcal{LA}$ quickly terminates if it is initialized with an assignment near a local minimum, as it is the case in the 1-neighborhood search, for example, while it typically takes several iteration steps when starting with an arbitrary assignment, like, e.g., in the multi-start method. A high number of $\mathcal{LA}$-calls together with a relatively low computation time consequently is an indication that the $\mathcal{LA}$-procedure drops back to the original local optimal solution without gathering new information about the search space in the corresponding $\mathcal{LA}$-calls. Note that the population-based heuristics execute the local search relatively often. Especially the evolutionary algorithm executes $\mathcal{LA}$ most frequently (besides the 1-neighborhood search), although it has the lowest average computation time.

Since we evaluated five different test problems of each size ten times, every method may

Table 5: Average Relative Difference in Percent of All Objective Values to the Best-Known Objective Value.

	MS	KNS	VNS	TS	SA	TA	EA	ACS	ACS+
1	0.035	0.044	0.108	0.508	0.154	0.061	0.233	0.000	0.000
2	0.005	0.622	0.018	0.652	0.104	0.052	0.301	0.000	0.000
3	0.016	0.192	0.020	0.077	0.017	0.074	0.098	0.000	0.039
4	0.032	0.121	0.040	0.200	0.063	0.056	0.182	0.012	0.044
5	0.044	0.221	0.007	0.161	0.011	0.001	0.039	0.000	0.018
6	0.004	0.651	0.003	0.099	0.019	0.054	0.102	0.000	0.071
7	0.045	0.313	0.011	0.107	0.036	0.035	0.053	0.002	0.036
8	0.016	0.342	0.010	0.114	0.095	0.080	0.073	0.035	0.074
9	0.057	0.433	0.040	0.233	0.155	0.177	0.211	0.108	0.156
10	0.049	0.792	0.032	0.209	0.068	0.052	0.114	0.032	0.080
11	0.041	0.340	0.057	0.091	0.054	0.063	0.122	0.045	0.083
12	0.023	0.457	0.001	0.018	0.000	0.001	0.077	0.000	0.061
13	0.108	0.433	0.031	0.165	0.138	0.133	0.164	0.063	0.182
14	0.036	0.345	0.006	0.113	0.027	0.043	0.076	0.003	0.105
15	0.045	0.395	0.024	0.159	0.036	0.038	0.117	0.071	0.098
16	0.111	0.433	0.013	0.090	0.031	0.019	0.126	0.039	0.134
17	0.026	0.221	0.008	0.075	0.013	0.017	0.057	0.014	0.041
18	0.050	0.280	0.009	0.042	0.019	0.019	0.070	0.041	0.065
19	0.040	0.239	0.027	0.088	0.080	0.070	0.070	0.074	0.075
20	0.093	0.294	0.019	0.107	0.054	0.051	0.067	0.083	0.108
21	0.055	0.258	0.008	0.049	0.032	0.028	0.065	0.048	0.082
22	0.048	0.205	0.017	0.053	0.029	0.029	0.044	0.019	0.057
23	0.076	0.207	0.024	0.037	0.040	0.037	0.047	0.030	0.071
24	0.096	0.174	0.023	0.046	0.039	0.031	0.043	0.031	0.066
25	0.068	0.161	0.015	0.037	0.027	0.024	0.035	0.026	0.077
avr	0.049	0.327	0.023	0.141	0.054	0.050	0.104	0.031	0.073

altogether obtain up to 50 times the best-known solution per problem size. Examining the values Table 4, it is not possible to definitely specify a superior search strategy, that outperforms all others. Especially the ability of population-based heuristics (EA, ACO, ACO+) to recombine good solutions has less effect on the solution quality than we expected. We thus can say that attending a whole set of good solutions simultaneously not necessarily leads to better solutions for this kind of (MFLPO). In combination with a local search method, a diversified search starting from one best-known solution may in fact be preferred. This effect can clearly be recognized by the results of the evolutionary algorithm. Mostly based on the recombination of good solutions, this method examines many assignments (cf. Table 3), but most likely quickly drops back to already examined local minimal solutions (cf. Table 2). It consequently does not reach the best-known solution as often as other methods that are based on a rather diversified search, as, for example, the variable neighborhood

search.

On a closer look, we can see that only ACO and ACO+ reached the best-known solution 50 times, although for smaller scale problems, up to size six only. We further note that ACO reached the best-known solutions most of the times, closely followed by the variable neighborhood search, which, together with the tabu search method outperformed ACO for larger problem instances.

The interesting effect that ACO reaches the best-known solutions more often than the ant colony optimization method with heuristic information can be explained as follows. Heuristic information guides the search in a greedy way to a good solution, thus, quicker computation times can be reached (cf. Tables 2 and 3). However, the aspired aim is not necessarily the global optimum, wherefore in these cases it is harder to find a better solution than without exterior influence.

In all tables the values of simulated annealing and threshold accepting are extremely similar. With the given problem data, both acceptance conditions, whether stochastically-based or deterministic, yield the same results.

In Table 5, the average relative difference to the best known objective values are specified. These values, given in percent, are computed as follows. Let F_j^i be the objective values obtained by a search method for the test problems $j = 1, \ldots, 5$ of a fixed problem size in the runs $i = 1, \ldots, 10$. Let $F_j^\star$, $j = 1, \ldots, 5$ be the best-known solutions obtained for these test problems by any of the heuristics. The corresponding value in Table 5 is given by

$$100 \cdot \frac{1}{5} \sum_{j=1}^{5} \frac{1}{10} \sum_{i=1}^{10} \frac{F_j^i - F_j^\star}{F_j^\star}.$$

Most search methods have the peak values around the problem sizes of ten. Huang et al. [2005] also noticed a decrease of the average relative difference for larger problem sizes in the evaluation of the $\mathcal{LA}$ local search for (MCLP) We can explain this effect as follows. As most heuristics determine the best-known solutions of the smaller-scale problems most of the times, the corresponding relative differences are relative low. These values increase with the size of the problem, since finding the best-known solution becomes more difficult. At a certain size however, the objective values of (MCLP) are so huge, that the relative difference of all determined local optimal solutions becomes lower again.

According to the values in Table 5, we see that all search methods obtained solutions with function values very close to the best-known values. Even the worst value, obtained by the 1-neighborhood search with test problems of size ten has, in the average, less than one percent relative distance to the best-known solutions. At a closer look we can see that variable neighborhood search, closely followed by ACO, reached the best results on the average.

5 Conclusions

We have outlined various randomized search methods for a generalized class of location-allocation problems that we denote as *multi-facility location-allocation problem with generalized objects* (MFLPO). Since every instance of this class can be decomposed into an assignment problem and a set of single-facility location problems, a well-known iterative

search method, known as *alternate location-allocation algorithm* can be applied. All considered heuristics use this local search procedure as a subroutine to complete and to improve examined solutions. In fact, they only pass arbitrary assignments to the local search subprocedure, which in turn delivers a local optimal solution of the problem. We described the different search strategies in an ascending order concerning the level of complexity. Particular similarities and distinctive differences are outlined in order to provide a unified view on these randomized search methods.

A large set of randomly generated instances of the multi-connection location-allocation problem (MCLP) under squared Euclidean distances is used for evaluation. The results show that population-based methods do not perform significantly better than other methods, as, for example variable neighborhood search. We therefore conclude that a well-balanced mixture of an intensified search in the interesting regions of the search space on the one hand, and a diversified search over the whole solution space on the other hand is more important than the plain ability of combining good solutions.

Furthermore, we evaluated two versions of an ant colony optimization algorithm, which differ solely in attending or neglecting *heuristic information*, a feature of ant-based algorithms to further intensify the search in promising regions. The ant-based algorithm which used the heuristic information needed less evaluations and computation time to converge, but also obtained inferior results than the version neglecting it. Thus can draw the conclusion that even though externally influencing the search by supplying heuristic information may be useful to guide the search to locally improved solutions, it is an unwanted feature, if additionally a local search procedure is applied.

Although we only evaluated (MCLP) in this paper, the allocation search methods are described for (MFLPO) in general, and therefore can be applied on all its instances. Also a transfer to related problems, as for example, clustering problems, is viable.

References

M. Bischoff and K. Klamroth. An efficient solution method for Weber problems with barriers based on genetic algorithms. *European Journal of Operational Research*, 177:22–41, 2007a.

M. Bischoff and K. Klamroth. Two branch & bound methods for a generalized class of location-allocation problems. *Operations Research*, 2007b. Submitted for publication.

I. Bongartz, P. H. Calamai, and A. R. Conn. A projection method for l_p norm location-allocation problems. *Mathematical Programming*, 66:283–312, 1994.

J. Brimberg and N. Mladenović. Solving the continuous location-allocation problem with tabu search. *Studies in Locational Analysis*, 8:23–32, 1996a.

J. Brimberg and N. Mladenović. A variable neighborhood algorithm for solving the continuous location-allocation problem. *Studies in Locational Analysis*, 10:1–12, 1996b.

J. Brimberg, P. Hansen, N. Mladenovic, and E.D. Taillard. Improvements and comparison of heursitics for solving the uncapacitated multisource Weber problem. *Operations Research*, 38:444–460, 2000.

V. Cerny. Thermodynamical approach to the traveling salesman problem: An efficient simulation algorithm. *Journal of Optimization Theory and Applications*, 45:41–51, 1985.

R. Chen. Solution of the minisum and minimax location-allocation problems with euclidean distances. *Naval Research Logistics Quarterly*, 30:449–459, 1983.

L. Cooper. Location-allocation problems. *Operations Research*, 11:331 – 343, 1963.

L. Cooper. Heuristic methods for location-allocation problems. *SIAM Review*, 6:37–53, 1964.

M. Dorigo. *Optimization, Learning and Natural Algorithms*. PhD thesis, PhD thesis, Dipartimento di Elettronica, Politecnico di Milano, 1992.

M. Dorigo and G. Di Caro. The ant colony optimization meta-heuristic. In D. Corne, M. Dorigo, and F. Glover, editors, *New Ideas in Optimization*, pages 11–32. McGraw-Hill Education, 1999.

M. Dorigo and L. M. Gambardella. Ant Colony System: A cooperative learning approach to the traveling salesman problem. *IEEE Transactions on Evolutionary Computation*, 1: 53–66, 1997.

M. Dorigo and T. Stützle. *Ant Colony Optimization*. MIT Press, Cambridge, MA, 2004.

Z. Drezner, editor. *Facilities Location: A Survey of Applications and Methods*. Springer-Verlag, New York, 1995.

Z. Drezner and H.W. Hamacher, editors. *Facility Location: Applications and Theory*. Springer-Verlag, New York, 2002.

A. T. Ernst and M. Krishnamoorthy. Efficient algorithms for the uncapacitated single allocation p-hub median problem. *Location Science*, 4:139–154, 1999.

R.L. Francis, F. Leon, L.F. McGinnis, and J.A. White. *Facility Layout and Location: An Analytical Approach*. Prentice-Hall, New York, 2nd edition, 1992.

F. Glover. Future paths for integer programming and links to artificial intelligence. *Computers and Operations Research*, 13:533–549, 1986.

F. Glover and G. Kochenberger, editors. *Handbook of Metaheuristics*. Kluwer Academic Publishers, Norvell, MA, 2003.

P. Hansen and N. Mladenovìc. Variable neighborhood search for the p-median problem. *Location Science*, 5:207–226, 1997.

P. Hansen, N. Mladenovìc, and É. Taillard. Heuristic solution of the multisource Weber problem as a p-median problem. *Operations Research Letters*, 22:55–62, 1998.

J. H. Holland. *Adaption in Natural and Artificial Systems*. University of Michigan Press, Michigan, 1975. re-issued by MIT Press (1992).

C. R. Houck, J. A. Joines, and M. G. Kay. Comparison of genetic algorithms, random restart and two-opt switching for solving large loacation-allocation problems. *Computers and Operations Research*, 23:587–596, 1996.

S. Huang, R. Batta, K. Klamroth, and R. Nagi. K-connection location problem in a plane. *Annals of Operations Research*, 136:193–209, 2005.

J. H. Jaramillo, J. Bhandury, and R. Batta. On the use of genetic algorithms to solve location problems. *Computers and Operations Research*, 29:761–779, 2002.

Y. Kao and K. Cheng. An ACO-based clustering algorithm. In M. Dorigo, L.M. Gambardella, M. Birattari, A. Martinoli, R. Poli, and T. Stützle, editors, *Proceedings of the 5th International Workshop ANTS 2006 (pp. 340–347)*, pages 340–347. Springer Verlag, 2006.

S. Kirkpatrick, C. D. Gelatt, and M. P. Vecchi. Optimization by simulated annealing. *Science*, 220:671–680, 1983.

R.E. Kuenne and R.M. Soland. Exact and approximate solutions to the multisource Weber problem. *Mathematical Programming*, 3:193–209, 1972.

R. F. Love and H. Juel. Properties and solution methods for large location-allocation problems. *Journal of the Operational Research Society*, 33:443–452, 1982.

R.F. Love and J.G. Morris. A computation procedure for the exact solution of location-allocation problems with rectangular distances. *Naval Research Logistics Quaterly*, 22: 441–453, 1975.

R.F. Love, J.G. Morris, and G.O. Wesolowsky. *Facilities Location: Models & Methods*. North Holland, New York, 1988.

N. Megiddo and K. J. Supowit. On the complexity of some common geometric location problems. *SIAM Journal on Computing*, 13:182–196, 1984.

N. Metropolis, A. Rosenbluth, M. Rosenbluth, A. Teller, and E. Teller. Equation of state calculations by fast computing machines. *Journal of Chemical Physics*, 21:1087–1092, 1953.

N. Mladenović and P. Hansen. Variable neighborhood search. *Computers and Operations Research*, 24:1097–1100, 1997.

P. Moscato. On evolution, search, optimization, genetic algorithms and martial arts: towards genetic algorithms. Technical report, Technical Report Caltech Concurrent Computation Program, Report 826, California Institute of Technology, Pasadena, California, 1989.

P. Moscato and J. F. Fontanari. Convergence and finite time behaviour of simulated annealing. *Advances in Applied Probability*, 18:747–771, 1990.

C. R. Reeves. Genetic algorithms for the Operations Researcher. *INFORMS Journal on Computing*, 9:231–250, 1997.

K.E. Rosing. An optimal method for solving the (generalized) multi-Weber problem. *European Journal of Operational Research*, 58:414–426, 1992.

T. Stützle and H. Hoos. MAX-MIN Ant System. *Future Generation Computer Systems*, 16:889–914, 2000.

P. J. M van Laarhoven and E. H. L. Aarts. *Simulated Annealing: Theory and Applications*. Kluwer Academic Publishers, Dordrecht, Boston, Norwell, Massachusetts, 1987.

H. Venables and A. Moscardini. An adaptive search heuristic for the capacitated fixed charge location problem. In M. Dorigo, L.M. Gambardella, M. Birattari, A. Martinoli, R. Poli, and T. Stützle, editors, *Proceedings of the 5th International Workshop ANTS 2006 (pp. 340–347)*, pages 348–355. Springer Verlag, 2006.

E. V. Weiszfeld. Sur le point pour lequel la somme des distances de n points donnés est minimum. *Tohoku Mathematical Journal*, 43:335–386, 1937.