*Working Paper Series*

# HEARIN CENTER FOR ENTERPRISE SCIENCE

HCES-02-04

## A Simple Filter-and-Fan Approach to the Facility Location Problem

by

Peter Greistorfer
César Rego
Bahram Alidaee

***The University of Mississippi***

Director, Keith Womer
School of Business Administration
The University of Mississippi
Post Office Box 1848
University, MS 38677-1848
(662) 915-5820
http://hces.bus.olemiss.edu

# A Simple Filter-and-Fan Approach to the Facility Location Problem

Peter Greistorfer[a], César Rego[b] and Bahram Alidaee[b]

*a* Institut für Industrie und Fertigungswirtschaft, Karl-Franzens-Universitä Graz, Universitätsstraße 15/G2, 8010 Graz, Austria. peter.greistorfer@kfunigraz.ac.at

*b* Hearin Center for Enterprise Science, School of Business Administration, University of Mississippi, University, MS 38677, USA. {crego, balidaee}@bus.olemiss.edu

Latest Revision: March 5, 2004.

**Abstract** — The design of effective neighborhood search procedures is a primary issue for the performance of local search and advanced metaheuristic algorithms. Several recent studies have focused on the development of variable depth neighborhoods that generate sequences of interrelated elementary moves to create more complex compound moves. These methods are chiefly conceived to produce an adaptive search as the number of elementary moves in a compound move may vary from one iteration to another depending on the state of the search. The objective is to achieve this goal with modest computational effort. Although ejection chain methods are currently the most advanced methods in this domain they usually require more complex implementations. The filter-and-fan (F&F) method appears as an alternative to ejection chain methods allowing for the creation of compound moves based on an effective tree search design. This paper reports the first implementation of the F&F approach to the uncapacitated facility location problem (UFLP). We examine a simple version of the F&F method in this preliminary study, and demonstrate its effectiveness on a set of 45 standard benchmark problems from the literature. The attractive performance of the method motivates an investigation of more advanced variants of F&F, and applications to additional problem settings.

**Keywords:** Metaheuristics, compound neighborhoods, variable depth methods, facility location, combinatorial optimization.

## 1. Introduction

Throughout a span of more than two decades a variety of methods have been developed and implemented for the uncapacitated facility (or warehouse) location problem (UFLP). We report the first application of the Filter and Fan (F&F) method to this problem, and demonstrate that it competes very effectively with previous methods on the two primary testbeds of benchmark problems from the literature. A noteworthy feature of our approach is its simplicity, which results from focusing on a subset of basic component strategies available to the method and using an implicit form of tabu search memory embedded in straightforward legitimacy conditions. The quality of the outcomes motivates further studies that examine more advanced versions of this approach incorporating more explicit and comprehensive types of tabu search memory, together with associated strategies derived from the F&F design.

The key ideas underlying the F&F method may be briefly described as follows.

Several recent studies in local search have been concerned with generating compound moves to explore the neighborhood space more effectively. By convention a compound move is one that can be decomposed into a sequence of more elementary component moves (or submoves). We define the *depth* or *level* of a compound move as the number of submoves in that move. The method is called *static* if the number of levels is fixed in advance (i.e. before the neighborhood search starts) and *dynamic* (or variable-depth) otherwise. Usually advanced neighborhood search methods are dynamic.

In graph-based optimization problems, there are basically two types of compound move constructions: (1) connected moves that preserve a feasible graph structure at each level and (2) disconnected moves that do not necessarily preserve such feasibility. A typical example for a type-(1) neighborhood is the so-called $k$-opt move where for $k>2$ the method generates a sequence of 2-opt moves (as its component move). Perhaps the most famous type-(2) neighborhood is the Lin-Kernighan procedure (Lin and Kernighan, 1973) for the TSP, which performs sequences of disconnected 2-opt moves for a variable number of levels. More general variable-depth neighborhoods are those based on ejection chain methods (Glover 1992, 1996). There are several types of ejection chains, some structured to induce successive changes in problem variables and others structured to induce changes in particular types of model components (such as nodes and edges of a graph). For implementations and other developments of ejection chain methods see Rego (1998a, 1998b, 2001) as well as Cavique, Rego, and Themido (1999).

The F&F approach is primarily conceived to generate type-(1) variable depth neighborhoods. The method was initially proposed in Glover (1997) as an improvement method for the scatter search, and further extended in Rego and Glover (2002). In the latter, the method is proposed as an alternative to ejection chain methods, which characteristically generate moves that can not be obtained by neighborhoods that preserve the feasibility at each step. The method underlies the use of specialized candidate lists that offer a useful basis for creating compound moves within exponential neighborhoods while using an economic degree of effort. Specifically, the F&F method is a combination of the *filtration* and the *sequential fan* candidate list strategies used in tabu search (Glover and Laguna 1997).

We propose a local search algorithm for the UFLP that uses a simple version of the F&F method to create paths of neighborhoods of varying characteristics within different stages and threads of the search. This paper is structured as follows. The problem is stated in Section 2. Section 3 gives a detailed description of the proposed algorithm. Section 4 discusses the computational results and conclusions follow in Section 5.

## 2. The uncapacitated facility location problem

The problem considered is the uncapacitated facility location problem (UFLP), also known as the warehouse location problem. For a survey see Krarup and Pruzan (1983), Cornuéjols, Nemhauser and Wolsey (1990), Gao and Robinson (1994) or, more recently, the comprehensive study of Körkel (1999). The underlying model of the UFLP is as follows. Consider a set $S = \{1,\ldots,s\}$ of warehouse or facility locations and a set $D = \{1,\ldots,d\}$ of customers to be served. With each customer $j \in D$ is associated a demand $b_j$ and $c_{ij}$ is the transportation cost of totally serving a customer $j$ by facility $i \in S$. Also, there is a fixed cost $F_i$ if facility $i$ is built (or opened). The objective is to find a set $W \subseteq S$ of opened facilities that minimizes the total cost.

After normalizing customer demands to $b_j = 1$, a linear programming formulation for the UFLP, using decision variables $x_{ij}$ and $y_i$, is:

$$
\begin{aligned}
Min \quad & \sum_{i=1}^{s} \sum_{j=1}^{d} c_{ij} x_{ij} + \sum_{i=1}^{s} F_i y_i & \\
s.t. \quad & x_{ij} \le y_i & i \in S, j \in D \\
& \sum_{i=1}^{s} x_{ij} = 1 & j \in D \\
& x_{ij} \ge 0 & i \in S, j \in D \\
& y_i \in \{0,1\} & i \in S
\end{aligned}
$$

In the formulation variables $x_{ij}$ indicate whether or not customer $j$ is served by facility $i$ in the solution, and similarly $y_i$ specifies whether facility $i$ is opened or closed.

A special characteristic of the UFLP is that a solution can be fully defined by the set of open facilities in that solution as the set of customers to be served by each facility is implicitly given by the minimum cost assignment of each customer to one of the open facilities. Therefore, especially in local search, it is natural to represent a UFLP solution by a vector $Y=(y_1, \ldots, y_s)$ where $y_i=1$ if the facility $i$ is open and 0 otherwise.

The UFLP belongs to the class of NP-hard problems (Papadimitriou and Yannakakis 1991). Several exact solution methods based on duality and Lagrangean relaxations have been proposed (Cornuéjols, Fisher and Nemhauser 1977, Erlenkotter 1978, Guignard 1988, Beasley 1993); however exact solution methods naturally are limited in terms of problem sizes and structure. It has been shown that real instances of the UFLP typically comprise a very large number of strong local optima and therefore are significantly more difficult to solve than randomly generated instances of similar size. Therefore, effective heuristic methods are required to provide optimal or near-optimal solutions for instances of larger, more realistic dimensions. These procedures include local search oriented methods based on standard add/drop or exchange neighborhoods (e.g. see Kuehn and Hamburger 1963). However, the use of advanced metaheuristics for this problem has been fairly neglected; interesting exceptions can be found in Kratica, Filipović and Tošić (1998), Sun (2003) and Michel and Van Hentenryck (2003).

## 3. The Filter and Fan Algorithm

The F&F model can be illustrated by a neighborhood tree where branches represent submoves and nodes identify solutions produced by these moves. An exception is made for the root node, which represents the starting solution to which compound moves are to be applied. The maximum number of levels considered in one sequence of moves defines the depth of the tree. The construction of the F&F tree is based on the use of a candidate list strategy and on weeding out promising moves by applying evaluations in successive levels of the tree search, where the moves that pass the evaluation criteria at one level are subjected to additional evaluation criteria at the next. The neighborhood tree is explored level by level in a breadth search strategy. For each level $k$, the method generates $\eta_1.\eta_2$ moves by the *fan candidate list strategy*, then a subset $M(k)$ of $\eta_2$ moves is selected by the *filter candidate list strategy* to generate the solutions for the next level.

A typical F&F method starts as a standard descent local search method by performing moves as long as they improve the best current solution. Once a local optimum is found (in the descent phase) the best $M(0)$ moves (among the $M$ moves evaluated to establish local optimality) are used to create the first level of the F&F neighborhood tree. The next levels are created as follows. Letting $\eta_1$ be the number of $M(k)$ moves for level $k$, the method proceeds by selecting a subset $A_i(k)$ of $\eta_2$ moves from $M(0)$ associated with each solution $Y_i(k)$ ($i=1,\ldots,\eta_1$) to generate $\eta=\eta_1.\eta_2$ trial solutions for the level $k+1$ (as a result of applying $\eta_2$ moves to each solution at level $k$). For example, consider $\eta_1=2\eta_2$. The method stops branching as soon as an improved solution is found, then switches back to the descent phase starting with this new solution.

We define $A(\text{k})=\{A_1(k), A_2(k), \ldots, A_{\eta_1}(k)\}$ ($|A_i(k)|=\eta_2$) as the set of $\eta$ moves evaluated at the level $k$ from which the set $M(k)=\{m_{1k}, m_{2k}, \ldots, m_{\eta_1 k}\}$ is selected, $M(k)\subset A(k)$, $k>0$. The process is repeated by creating a set $Y(k+1)$ of solutions obtained by applying $M(k)$ moves to the associated solutions in $Y(k)$ and keeping these solutions as starting points for the next level of the F&F tree.

The process of selecting $\eta_2$ moves has to obey a set a legitimacy conditions specific to the type of move utilized. For the UFLP, we consider moves defined by the classic flip-neighborhood that flips the status of one facility from 0 (close) to 1 (open) or vice versa. Clearly a feasible solution can be generated if and only if there exists at least one open facility. As infeasibility is not permitted in the search process of our algorithm, we establish a legitimacy condition that keeps the method from closing the only open facility in the current solution without opening a new one. Additionally, in order to avoid a number of duplicated solutions (that are most likely to arise at level $k=2$), a special legitimacy restriction is imposed to a move $m_{t1}$ by restricting the flip of two variables $y_i$ and $y_j$ so that $i>j$, where $y_i \in M(0)$ and $y_j \in A_t(1)$. Although the latter legitimacy conditions also apply to further levels of the tree, it is expected that the chance for duplications drop significantly as the number of levels gets larger. Consequently, it is easier to screen for duplications at the point where a move becomes a candidate to be included in $M(k+1)$, or equivalently when the associated solution becomes a candidate to include in $Y(k+1)$. An appropriate hash function is used to conveniently identify such duplications.

More elaborate designs of the F&F method allow different types of moves for combination at each level of the tree, so that compound moves can be obtained by different neighborhoods applied under appropriate legitimacy conditions. By continuing the tree search after a local optimum is found, local optimality is overcome in “intermediate” levels of the F&F tree. Then the best trial solution encountered throughout the tree is chosen to re-initiate the descent phase. We consider a simplified version where only flip moves are used for both the descent phase and the neighborhood search tree.

Our goal in this paper is to create a simple but effective local search algorithm for the UFLP in which the neighborhood search avoids most of the sophisticated ideas prescribed in advanced metaheuristic strategies. More advanced versions may result by introducing an explicit tabu search memory structure to augment the implicit memory embodied in the legitimacy conditions. Similarly, gains may be expected by replacing the descent phase with a tabu search phase, which, for example, can activate the F&F strategy based on the use of a critical event memory. Fundamentally, the F&F strategy can be used either to intensify the search in regions of high quality (elite) solutions or to diversify the search by propelling the method to a different region of the solution space.

Enhanced constructions of the underlying look-ahead process may also be provided by the use of ejection chain processes (performed from nodes at the current level) as a foundation to determine promising component moves to dynamically update the candidate list. Moreover, high evaluation trial solutions found throughout the ejection chain can be recorded for further consideration. Related possibilities for enhancement that make recourse to associated elements of tabu search are discussed later.

The proposed F&F algorithm undertakes two fundamental steps. The first step is a classical local search procedure that starts with all facilities open, then improves that solution by flipping the facility that locally minimizes the objective function value and the process is repeated until no improvement is possible by closing a new facility. Let $M$ be the set of all moves evaluated in the last iteration of this descent method, then we keep the $\eta_0$ "best" moves in $M$ to create the initial candidate list $M(0)$ for the F&F tree used in the next step. As clarified later the term "best" does not necessarily refer to the net change in the objective function value created by the move but may include a bias factor introduced by memory considerations used to guide the search at different layers.

The basic skeleton of the proposed F&F algorithm is as in Figure 1. Denote $Y^*$ the best solution found so far and $L$ an upper limit for the number of levels of the F&F tree.

Step 0. *Generate a candidate list of component moves*

(a) Let $Y$ be a binary vector representing the initial solution with all variables set to 1. Perform a local search using flip-moves that switch the status of one variable from 1 to 0 or vice-versa. Let $M$ be the set of all moves evaluated in the last iteration of this procedure.
(b) Create a candidate list $M(0)$ with the $\eta_0$ highest evaluation moves in $M$.
(c) Consider $Y=Y^*$ the new starting solution. Apply the best $\eta_1$ moves in $M(0)$ to $Y^*$ to create the first level of the F&F tree with solutions $Y_i(1)$ ($i=1,\dots,\eta_1$). Set $k=1$.

Step 1. *Generate the Filter and Fan tree*

(a) Identify the best $\eta_2$ legitimate moves derived from $M(0)$ for each solution $Y_i(k)$ ($i=1,\dots,\eta_1$) to create sets $A_i(k)$ ($j=1,\dots,\eta_1$).
(b) Evaluate each move in $A_i(k)$, applied to the associated solution $Y_i(k)$, and compute the value of the corresponding trial solution.
(c) If the best trial solution found is better than $Y^*$, perform the associated move from $Y_i(k)$ to $Y^*$ and go to Step 0.
(d) Otherwise, select the $A(k)$ moves that led to the best $\eta_1$ trial solutions to become the members of $M(k)$.
(e) Apply the $M(k)$ moves to the corresponding solutions $Y_i(k)$ to create $Y(k+1)$.
(f) If $k=L$ stop. Otherwise set $k=k+1$ and repeat Step 1.

Figure 1. The Basic Filter-and-Fan Procedure for the UFLP

We now undertake to provide further explanation on the foundation of our approach.

Classical neighborhoods for the UFLP are the *switch*-neighborhood that switches the status of one facility from open to close or vice versa by flipping a single variable at a time and the *swap*-neighborhood that simultaneously closes one facility and opens another.

The filter and fan approach used in this study extends and generalizes the use of these two types of neighborhoods by using compound moves derived from the switch neighborhood. The method proceeds by performing moves that flip the value of one variable at each node of the F&F tree structure. A swap move implicitly results whenever in two successive nodes of a given branch of the tree, one variable flips from 0 to 1 and another variable flips from 1 to 0. Since the outcome of a move chosen at one node of the tree is transmitted to subsequent nodes of the same branch, the method is *adaptive* in the sense that the type of neighborhood and the move itself is chosen according to the current state of the search. Also, the method is *dynamic* since the number of single flips used to compose a compound move depends on the level of the tree where the best trial move was found, which usually varies from one iteration to another, again depending on the state of the search.

## 4. Computational results

The performance of the proposed algorithm was evaluated on a set of 45 benchmark instances taken from two sources. These include 15 instances from OR-Library (Beasley 1993) with sizes ranging from 16x50 to 100x1000. The remaining instances, obtained with the generator provided by Kratica et al. (2001), correspond to 6 classes of problems (MO, MP, ..., MT) each containing 5 instances with sizes ranging from 100x100 to 2000x 2000. The algorithm was coded in Pascal (FreePascal 1.0.10) compiler and runs were carried out on a 1GHz Pentium III processor under a Windows 2000 Pro SR3 platform using the DOS-extender go32v2.

All results refer to the straightforward strict version of the F&F method, i.e. a version that terminates at a given maximum level $L$ if no improved solution was found on this last level. We consider the following fixed set of parameters: we selected $|M(k)| = 9 \ \ (k \geq 0)$ and $L$=6, while making no restrictions to the move sets. This means that for all sets $A(k)$ we consider a total of $\eta=9|S|$ solutions. Additionally, this variant starts with $M=M(0)=S$.

Our implementation takes advantage of a very effective gain update technique following Michel and Hentenryck (2003) which always gives a fast and direct access to the complete set of possible neighborhood solutions by calculating gains instead of evaluating the whole set of solutions after performing the corresponding trial moves.

Table 1 depicts the experimental results that are relevant for the analysis of the algorithm. The first three columns give the problem name, the size in terms of the number of facilities ($|S|$) and customers ($|D|$), and the best known solution. All best known solutions are optimal, except for the larger M* instances MS* and MT* (Kratica et al. 2001).

The next four columns report results obtained in Step 1 and Step 2 of the F&F procedure, respectively. For each step we provide the relative percentage deviation (RPD) from the optimum or best known solution and the number of iterations (Iter) needed to find the corresponding solution. The last two columns report the running time for the algorithm to find its best solution (BEST) and to terminate (END).

The computational results can be summarized as follows. On Beasley's testbed, 13 out of 15 instances were solved to optimality and solutions for the remaining 2 instances fall in the range of 0.6% and 1.24% deviation from the optimum objective function value. As for the 30 Kratica et al. generated problems the algorithm found all the best known solutions among which 20 are optimal.

We recall that the algorithm stops when the current best solution is not improved in one iteration of the F&F procedure. Consequently, it is interesting to notice that over the entire set of test problems the algorithm finds the best known solutions for 30 instances even before invoking the tree search. However, the tree search phase was crucial to improve the first local optimum and find 13 more best known solutions. Also, we can see that for the only two instances that the algorithm fails to find an optimal solution, the tree search still improves the local optimum found in the descent phase for one instance. As far as the computation time is concerned, the algorithm is extremely fast. It finds solutions that are 0.12% on average above the optimum for Beasley's instances in 0.23 seconds. We should point out that the M* instances are characterized to model real applications and are typically difficult to solve as they exhibit a large number of local optima and transportation costs and fixed costs are inversely correlated. In spite of this inherent difficulty our algorithm finds all best know solutions for M* instances in 4.05 seconds on average. Overall the algorithm finds 43 out of 45 best known solutions in about 2.04 seconds on average with an average deviation of 0.04% above the best known solutions.

Considering that the core of the proposed algorithm is the generation of compound moves to enhance the classical flip-based neighborhoods, the results clearly show that the filter-and-fan approach provides a very effective framework to explore the solution space in facility location problems and suggests its use in other classes of combinatorial optimization problems.

Although the performance of our algorithm is well established by the foregoing results, we should not conclude this section before making a brief comparative analysis with other very efficient algorithms in the literature.

In spite of the significant number of heuristic algorithms that have been proposed for the UFLP, at the time of this writing there are only two that can compete with our simple proposed F&F. These are the genetic local search algorithm in Kratica et al. (2001) and the tabu search algorithm by Michel and Hentenryck (2003). As shown in the latter reference, the tabu search algorithm appears more robust and is usually faster than the genetic algorithm.

Another tabu search algorithm has been recently proposed in Sun (2003). The algorithm uses frequency-based long term memory as a diversification mechanism necessary to find all the optimal solutions for the smaller instances of Or-Library. Unfortunately, no results are reported on larger M* instances. Instead the author generates his own instances and provides percentage deviations relative to lower bounds obtained by an unspecified Lagrangean algorithm. Neither solution values for this TS algorithm nor lower bounds are provided and instances are not available.

Based on the information available, we conjecture that Michel and Hentenryck's tabu search implementation (MH-TS) is perhaps the fastest and most effective algorithm available at this point. However, because the algorithm depends on a random parameter used to select a move from the current available candidates, the authors report solutions obtained over 100 runs, which makes it difficult to establish a direct comparison with our F&F algorithm that runs only once with a fixed setting of parameters. Perhaps, a meaningful observation is that comparing the average solution quality produced by this tabu search implementation over the same benchmark set used in our tests, the F&F algorithm finds 7 better solutions compared to 2 better solutions out of the 9 solutions that differ in these two algorithms. Table 2 illustrates these results.

Another interesting observation concerns the computational times. The Michel and Hentenryck's tabu search algorithm runs on a 2GHz Pentium IV while the F&F algorithm runs on a 1GHz Pentium III processor. Appropriate ratios to scale the relative speed of these processors can be derived using averaged CPU floating-point benchmarks (SPECfp_base2000) reported at www.spec.org. We conjecture that a factor of 2.44 should be used to scale the running times of the MH-TS algorithm to the ones produced by the F&F algorithm. This suggests that the MH-TS algorithm takes on average about the same time as F&F to find its best solution for smaller instances of Beasley (0.25 compared to 0.23 taken by F&F). However it turns out that for larger instances (M*) the F&F is more than 1.7 times faster than the MH-TS (7.07 compared to 4.05 seconds) and this ratio seems to persist as the problem size increases, although for some instances (e.g. MS4) the F&F algorithm is more than 2.5 times faster than MH-TS (10.42 compared to 4.12 seconds).

| Problem | | | | Filter and Fan | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| | | | | Step 1 | | Step 2 | | Time | |
| Name | \| $S$\| | \| $D$\| | Best Known | RPD | Iter | RPD | Iter | Best | End |
| cap71 | 16 | 50 | 932615.75 | 0.00 | 6 | 0.00 | 1 | 0.00 | 0.05 |
| cap72 | 16 | 50 | 977799.40 | 0.00 | 8 | 0.00 | 1 | 0.00 | 0.00 |
| cap73 | 16 | 50 | 1010641.45 | 0.00 | 12 | 0.00 | 1 | 0.00 | 0.06 |
| cap74 | 16 | 50 | 1034976.97 | 0.26 | 13 | 0.00 | 2 | 0.00 | 0.06 |
| cap101 | 25 | 50 | 796648.437 | 0.00 | 11 | 0.00 | 1 | 0.00 | 0.05 |
| cap102 | 25 | 50 | 854704.20 | 0.00 | 17 | 0.00 | 1 | 0.00 | 0.06 |
| cap103 | 25 | 50 | 893782.11 | 0.00 | 18 | 0.00 | 1 | 0.00 | 0.06 |
| cap104 | 25 | 50 | 928941.75 | 0.61 | 22 | 0.00 | 2 | 0.00 | 0.05 |
| cap131 | 50 | 50 | 793439.56 | 0.00 | 36 | 0.00 | 1 | 0.00 | 0.00 |
| cap132 | 50 | 50 | 851495.32 | 0.00 | 42 | 0.00 | 1 | 0.00 | 0.00 |
| cap133 | 50 | 50 | 893076.71 | 0.08 | 43 | 0.00 | 2 | 0.00 | 0.00 |
| cap134 | 50 | 50 | 928941.75 | 0.61 | 47 | 0.00 | 2 | 0.00 | 0.00 |
| capa | 100 | 1000 | 17156454.4 | 6.95 | 97 | 0.00 | 4 | 0.93 | 2.74 |
| capb | 100 | 1000 | 12979071.5 | 1.04 | 94 | 0.60 | 4 | 2.25 | 3.57 |
| capc | 100 | 1000 | 11505594.3 | 1.24 | 92 | 1.24 | 1 | 0.22 | 1.32 |
| Average | | | | 0.72 | | 0.12 | | 0.23 | 0.53 |
| MO1 | 100 | 100 | 1156.91 | 0.53 | 96 | 0.00 | 3 | 0.05 | 0.11 |
| MO2 | 100 | 100 | 1227.67 | 0.00 | 96 | 0.00 | 1 | 0.11 | 0.16 |
| MO3 | 100 | 100 | 1286.37 | 0.94 | 97 | 0.00 | 4 | 0.11 | 0.22 |
| MO4 | 100 | 100 | 1177.88 | 0.16 | 97 | 0.00 | 2 | 0.11 | 0.16 |
| MO5 | 100 | 100 | 1147.60 | 0.00 | 97 | 0.00 | 1 | 0.11 | 0.17 |
| MP1 | 200 | 200 | 2460.10 | 0.00 | 196 | 0.00 | 1 | 0.28 | 0.99 |
| MP2 | 200 | 200 | 2419.32 | 0.00 | 197 | 0.00 | 1 | 0.22 | 1.04 |
| MP3 | 200 | 200 | 2498.15 | 0.00 | 197 | 0.00 | 1 | 0.22 | 1.09 |
| MP4 | 200 | 200 | 2633.56 | 0.45 | 197 | 0.00 | 2 | 0.22 | 1.15 |
| MP5 | 200 | 200 | 2290.16 | 0.00 | 196 | 0.00 | 1 | 0.22 | 0.93 |
| MQ1 | 300 | 300 | 3591.27 | 0.00 | 296 | 0.00 | 1 | 0.49 | 2.20 |
| MQ2 | 300 | 300 | 3543.66 | 0.04 | 297 | 0.00 | 2 | 0.61 | 2.37 |
| MQ3 | 300 | 300 | 3476.81 | 0.85 | 297 | 0.00 | 3 | 0.61 | 2.53 |
| MQ4 | 300 | 300 | 3742.47 | 0.00 | 296 | 0.00 | 1 | 0.50 | 2.25 |
| MQ5 | 300 | 300 | 3751.33 | 0.00 | 297 | 0.00 | 1 | 0.55 | 2.31 |
| MR1 | 500 | 500 | 2349.86 | 0.00 | 496 | 0.00 | 1 | 1.15 | 5.32 |
| MR2 | 500 | 500 | 2344.76 | 0.00 | 496 | 0.00 | 1 | 1.15 | 5.33 |
| MR3 | 500 | 500 | 2183.24 | 0.00 | 495 | 0.00 | 1 | 1.10 | 4.95 |
| MR4 | 500 | 500 | 2433.11 | 0.00 | 496 | 0.00 | 1 | 1.15 | 5.65 |
| MR5 | 500 | 500 | 2344.35 | 0.00 | 496 | 0.00 | 1 | 1.20 | 5.54 |
| MS1 | 1000 | 1000 | 4378.63 | 0.00 | 995 | 0.00 | 1 | 4.12 | 19.33 |
| MS2 | 1000 | 1000 | 4658.35 | 0.05 | 996 | 0.00 | 2 | 4.56 | 20.54 |
| MS3 | 1000 | 1000 | 4659.16 | 0.67 | 996 | 0.00 | 3 | 4.94 | 21.15 |
| MS4 | 1000 | 1000 | 4536.00 | 0.00 | 996 | 0.00 | 1 | 4.12 | 20.00 |
| MS5 | 1000 | 1000 | 4888.91 | 0.00 | 996 | 0.00 | 1 | 4.12 | 21.91 |
| MT1 | 2000 | 2000 | 9176.51 | 0.00 | 1996 | 0.00 | 1 | 17.91 | 82.72 |
| MT2 | 2000 | 2000 | 9618.85 | 0.00 | 1996 | 0.00 | 1 | 17.91 | 89.86 |
| MT3 | 2000 | 2000 | 8781.11 | 0.00 | 1995 | 0.00 | 1 | 17.68 | 80.30 |
| MT4 | 2000 | 2000 | 9225.49 | 0.00 | 1996 | 0.00 | 1 | 18.02 | 85.36 |
| MT5 | 2000 | 2000 | 9540.67 | 0.00 | 1996 | 0.00 | 1 | 18.01 | 88.15 |
| Average | | | | 0.12 | | 0.00 | | 4.05 | 19.13 |
| Total | | | | 0.32 | | 0.04 | | 2.78 | 12.93 |

Table 1. Experimental Results for the Filter-and-Fan Algorithm

| Problem | Solution | |
|---|---|---|
| Name | MH-TS | F&F |
| cap101 | 796820.49 | **796648.44** |
| cap103 | 893795.67 | **893782.11** |
| cap131 | 793577.21 | **793439.56** |
| cap132 | 851495.33 | **851495.32** |
| cap133 | 893104.93 | **893076.71** |
| capb | **13022893.30** | 13057343.82 |
| capc | **11514330.70** | 11647844.39 |
| M01 | 1156.95 | **1156.91** |
| M03 | 1286.60 | **1286.37** |

Table 2 Comparative results between TS and F&F

## 5. Conclusion

Local search algorithms for the facility location problem typically use flip-based neighborhoods that change the status (open or close) of one facility at each iteration of the method. In this paper we successfully attempted to enhance the performance of a flip-neighborhood by generating sequences of flip moves within a simple Filter-and-Fan approach. The results obtained on different sets of benchmark problems demonstrate that the method is highly effective for providing optimal and near-optimal solutions for the UFLP in a very short computation time.

The results obtained on a set of classical benchmark problems clearly show that the proposed F&F approach is competitive with or even outperforms current state-of-the-art algorithms in solving these problems. However, a variety of ways exist to enhance our simple version of F&F. Besides the observations in Section 3 and the strategic components described in Glover (1997) and Rego and Glover (2002) to enhance the basic F&F procedure, there are a few options that seem quite natural to employ that stem from the following observation.

To some extent the F&F method can be interpreted as performing multiple threads of tabu search from the root node of the F&F tree using a limited short-term memory component derived from the legitimacy restrictions. From this perspective a straightforward enhancement results by creating a more general algorithm managed by two basic types of short-term memory components: (1) a *branch-memory* that is local to each branch of the F&F tree; (2) a *tree-memory* that is global to the F&F tree. A limited form of branch-memory is implicitly defined in the legitimacy restrictions of the tree search process, but the inclusion of more explicit forms of memory allows different levels of flexibility by using either one of the two indicated types of memory or both memories combined. Branch-memory serves to forbid reverse-flips while tree-memory is conceived to produce a higher level of diversification of the search among the different branches of the tree. The latter ensures that each variable cannot switch its value more than once during the entire tree search.

Higher levels of intensification and diversification can be achieved by incorporating more advanced memory structures as prescribed in tabu search. We conjecture that an effective integration of memories organized at different layers provides a useful mean for the creation of the $M(k)$ candidate lists as well as a vehicle to drive the search in an iterative process that performs the F&F procedure for a number of iterations until a given stopping criterion is met as in general tabu search implementations.

## References


Beasley, J.E. (1993). "Lagrangean heuristics for location problems," *European Journal of Operational Research*, 65:383–399.

Cavique, L., C. Rego and I. Themido (1999). "Subgraph ejection chains and tabu search for the crew scheduling problem," *Journal of the Operational Research Society*, 50:608–616.

Cornuéjols, G., M.L. Fisher and G.L. Nemhauser (1977). "Location of bank accounts to optimize float: An analitytic study of exact and approximate algorithms," *Management Science*, 23:789–810.

Cornuéjols, G., G.H. Nemhauser and L. Wolsey (1990). "The uncapacitated facility location problem," In P. Mirchandani and R. Francis, editors, Discrete Location Theory, 119–171. John Wiley and Sons, Inc., New York.

Erlenkotter, D. (1978). "A dual-based procedure for uncapacitated facility location," *Operations Research*, 26:992–1009.

Gao, L.L. and E.P. Robinson (1994). "Uncapacitated facility focation: General solution procedures and computational experience," *European Journal of Operational Research*, 76:410–427.

Glover, F. (1992). "New ejection chain and alternating path methods for traveling salesman problems," *Computer Science and Operations Research*, 449–509.

Glover, F. (1996). "Ejection chains, reference structures and alternating path methods for traveling salesman problems," *Discrete Applied Mathematics*, 65:223–253.

Glover, F. (1998). "A template for scatter search and path relinking," In J.-K. Hao, E. Lutton, E. Ronald, M. Schoenauer, and D. Snyers, editors, Artificial Evolution. *Lecture Notes in Computer Science*, 1363, 3–51. Springer, Heidelberg.

Glover, F. and M. Laguna (1997). "Tabu Search," *Kluwer Academic Publishers*, Boston.

Guignard, M. (1988). "A lagrangean dual ascent algorithm for simple plant location problems," *European Journal of Operational Research*, 35:193–200, 1988.

Körkel, M. (1999). "Effiziente Verfahren zur Lösung unkapazitierter Standort-Probleme," *Verlag für Wissenschaft und Forschung*, Berlin.

Krarup, J. and P.M. Pruzan (1983). "The simple plant location problem: Survey and synthesis," *European Journal of Operational Research*, 12:36–81.

Kratica, J., V. Filipović and D. Tošić (1998). "Solving the uncapacitated warehouse location problem by SGA with add-heuristic," Technical report, Faculty of Mechanical Engineering, Department of Mathematics, Belgrade.

Kratica, J., D. Tošić , V. Filipović and I. Ljubic (2001). "Solving the simple plant location problem by genetic algorithm," RAIRO Operations Research, 35:127–142.

Kuehn, A.A. and M.J. Hamburger (1963). "A heuristic program for locating warehouses," Management Science, 9:643–666.

Lin, S. and B. Kernighan (1973). "An effective heuristic algorithm for the traveling salesman problem," *Operations Research*, 21:498–516.

Michel, L. and P. Van Hentenryck (2003). "A simple tabu search for warehouse location," *European Journal of Operational Research*, to appear.

Papadimitriou, C. and M. Yannakakis (1991). "Optimization, approximation and complexity classes," *Journal of Computer and System Sciences*, 43:425–440.

Rego, C. (1998a). "Relaxed tours and path ejections for the traveling salesman problem," *European Journal of Operational Research*, 106:522–538.

Rego, C. (1998b). "A subpath ejection method for the vehicle routing problem," *Management Science*, 44:1447–1459.

Rego, C. (2001). "Node ejection chains for the vehicle routing problem: Sequential and parallel algorithms," *Parallel Computing*, 27:201–222.

Rego, C. and F. Glover (2002). "Local search and metaheuristics for the traveling salesman problem," In G. Gutin and A. Punnen, editors, The Traveling Salesman Problem and its Variations, 309–368. *Kluwer Academic Publishers*.

Sun, M. (2003). "A tabu search heuristic for the uncapacitated facility location problem," In C. Rego and B. Alidaee, editors. Adaptive Memory and Evolution: Tabu Search and Scatter Search, *Kluwer Academic Publishers*, to appear.