Annals of Operations Research 136, 285–302, 2005
© 2005 Springer Science + Business Media, Inc. Manufactured in The Netherlands.

Looking Ahead with the Pilot Method

STEFAN VOß stefan.voss@uni-hamburg.de
ANDREAS FINK fink@econ.uni-hamburg.de
Universität Hamburg, Institut für Wirtschaftsinformatik, Von-Melle-Park 5, D-20146 Hamburg, Germany

CEES DUIN c.w.duin@uva.nl
Universiteit van Amsterdam, Faculteit der Economische Wetenschappen en Econometrie, Roeterstraat 11, NL-1018 WB Amsterdam, The Netherlands

Abstract. The pilot method as a meta-heuristic is a tempered greedy method aimed at obtaining better solutions while avoiding the greedy trap by looking ahead for each possible choice. Repeatedly a master solution is modified; each time in a minimal fashion to account for *best* choices, where choices are judged by means of a separate heuristic result, the *pilot* solution.

The pilot method may be seen as a meta-heuristic enhancing the quality of (any) heuristic in a system for *heuristic repetition*. Experiments show that the pilot method as well as similar methods can behave quite competitively in comparison with well-known and accepted meta-heuristics. In this paper we review some less known results. As a higher time complexity is usually associated with repetition, we investigate a simple short-cut policy to reduce the running times, while retaining an enhanced solution quality. Furthermore, we report successful experiments that incorporate a distinguishing feature of the pilot method, which is the extension of neighborhoods into "local" search, creating tabu search hybrids.

Keywords: meta-heuristics, pilot method, rollout method, combinatorial optimization

1. Introduction

Combinatorial optimization problems are challenging in terms of finding efficient algorithms. Often these problems turn out to be $\mathcal{NP}$-hard (Garey and Johnson, 1979) and one resorts to heuristics.

Greedy construction algorithms are frequently the first choice for hard combinatorial optimization problems. Such algorithms may accept myopic choices when bringing a partial solution towards feasibility. Once feasibility is obtained, a second phase may be employed which performs certain changes, for the most part maintaining feasibility and hopefully leading to better objective function values. In this respect local search mechanisms come into play, however, again behaving greedily in a steepest descent or steepest ascent manner (whatever problem, minimization or maximization is considered). Recently, local search concepts have been married with superordinate mechanisms leading to meta-heuristics.

Meta-heuristics are iterative master processes guiding and modifying the operations of subordinate heuristics to efficiently produce high-quality solutions. Usually, meta-heuristics iteratively operate on a complete (or incomplete) single solution or a collection of solutions, while the subordinate heuristics may be high (or low) level procedures such as a simple local search procedure or even a construction method. Even in meta-heuristics greedy choices are performed and usually measured by means of an immediate gain possibly leading to suboptimal choices. Therefore, we need a so-called *heuristic measure* to evaluate this gain as well as a mechanism to guide the choices. For a definition of meta-heuristics see Voß et al. (1999), p. ix, the concept of heuristic measure is given by, e.g., Rayward-Smith and Clare (1986) and Voß (1990).

Usually, the heuristic measure is performed rather shortsighted. Therefore, especially in the artificial intelligence community, researchers have thought about look ahead features (see, e.g., Pearl, 1984). In that sense a heuristic measure may incorporate intelligent mechanisms to evaluate certain decisions or moves not based on an immediate gain but on the outcome after, say, a few iterations. Related approaches have applied this concept as meta-heuristics for combinatorial optimization problems (Bertsekas, Tsitsiklis, and Wu, 1997; Duin and Voß, 1999). In this paper we review and further investigate the *pilot method* of Duin and Voß (1994, 1999).

The pilot method is a tempered greedy algorithm based on look ahead results, pilots, obtained by heuristic repetition for each possible choice. (Note that the pilot method as our "Preferred Iterative LOok ahead Technique" is motivated by the idea to "Perform Improved Look ahead with Objective-value Tests".) Once a heuristic approach is known, one may produce with it enhanced solutions, by adopting the heuristic as the so-called pilot heuristic into the pilot method. A couple of independently developed approaches from the literature may be re-interpreted – by means of the concept of heuristic measure – as applying the key ideas of the pilot method.

Of course, as a repetitive system of extensive heuristic repetition, the pilot method entails considerable running times. However, one may apply policies to reduce these times. In this paper we examine the *evaluation depth*, i.e., a parameter controlling the number of times that the extensive heuristic repetition restarts.

As pointed out already in Duin and Voß (1999), the pilot procedure need not be a constructive heuristic (for finding feasible solutions). It may also be a procedure for improving solutions like steepest descent. In this paper we integrate this idea into a local search method, designing a hybrid algorithm in combination with tabu search.

In the following we revisit the concept of heuristic measure and describe the pilot method in more detail (Section 2). In Duin and Voß (1999) we primarily focused on solving the Steiner tree problem in graphs, a problem with a wide variety of applications in network design and location theory. This paper's tests of the pilot method are given on different combinatorial optimization problems. We include a survey of some results from the literature as well as new results (see Sections 3–6). Some conclusions are given in Section 7.

2. The pilot method

Using a known algorithm such as a greedy construction heuristic as a building block or application process, the *pilot method* (Duin and Voß, 1994, 1999) is a meta-heuristic with the primary idea of performing repetition using the application process as a *look ahead* mechanism. In each iteration (of the pilot method) one tentatively computes for every possible choice (or move) a so-called "pilot" solution, recording the best results in order to extend at the end of the iteration a so-called "master" solution with the corresponding move. One may apply this strategy by successively performing, e.g., a cheapest insertion heuristic for all possible local choices (i.e., starting anew from each incomplete solution that can result from the inclusion of any not yet included element at some position in the current incomplete solution).

In the following we explain the pilot method in more detail following Duin and Voß (1999). Consider a combinatorial optimization problem defined on a finite set of elements E weighted by a cost function $c : E \to \mathrm{R}$. The problem is to select at minimum cost a subset (or solution) $S^* \subset E$, satisfying some feasibility properties. As an example consider the traveling salesman problem (TSP), i.e., the problem of finding a least cost tour in a given graph – a minimum cost subset of the edges forming a cycle through all nodes of the graph; see, e.g., Lawler et al. (1985). We assume the availability of a known heuristic H for the problem, such as the nearest neighbor or the cheapest insertion heuristic for the TSP.

Imperatives for almost any heuristic are: be greedy, perform trial and error, fix variables, look ahead. The pilot method combines all these elements. By looking ahead with H as a so-called pilot heuristic, one cautiously builds up a partial solution M, the master solution. Separately for each element $e \notin M$, the pilot method is to extend tentatively a copy of M to a (fully grown) solution in such a way that e is included. Let $p(e)$ denote the objective function value of the solution obtained by pilot H for $e \in E - M$, and let e_0 be a most promising such element, i.e., $p(e_0) \leq p(e)$ for all $e \in E - M$. Element e_0 is included in M by changing it in a minimal fashion. On the basis of the changed master solution M, new pilot calculations are started for each $e \notin M$, providing a new solution element e'_0, and so on. This process could proceed, e.g., until further pilot calculations do not lead to improvements.

To give an example, let us consider as a pilot heuristic the nearest neighbor algorithm for solving the TSP: Starting with a vertex $i_1 = 1$, one chooses greedily the edge that leads to a nearest node i_2 of i_1. In the next iteration, from i_2 one adds to the partial solution $\{(1, i_2)\}$, the edge to a vertex $i_3 \in V - \{i_1, i_2\}$ that is nearest to i_2 and so on until (i_{n-1}, i_n) enters the solution. Then (i_n, i_1) concludes the tour.

Heuristics often use functions or measures to guide their choices. The nearest neighbor algorithm applies the distance function to a not yet included node. It is a normal constructive heuristic (construction method), where the next choice is the greedy choice as evaluated by the heuristic measure. (Note that by a construction method we mean a method that starts from some initial state (feasible or infeasible) and iteratively performs steps towards feasibility, generating a solution, e.g., by adding suitable elements of E.) Similarly, as in the greedy heuristic H that it exploits, the pilot method evaluates which next node to include, but by a new function, derived from H. Again, a node e is evaluated by means of a heuristic measure $p(e)$, but this time $p(e)$ provides an objective test, by looking fully ahead (performing a look ahead with objective-value tests).

As the pilot method is generic with respect to the type of problem and defines an iterative master process that guides and modifies the operations of some subordinate heuristic, it may be regarded as a meta-heuristic. Moreover, we may also distinguish between a *guiding process* and an *application process*. The guiding process decides upon possible (local) moves and forwards its decision to the application process which then executes the chosen move. In addition, it provides information for the guiding process (depending on the requirements of the respective meta-heuristic).

Alternatively, in Duin and Voß (1999) we characterized the pilot method for combinatorial optimization by drawing a parallel with branch and bound. Imagine a branch and bound algorithm for a minimization problem, that fully branches a (sub)problem to all possible values of a solution variable to be fixed. Algorithmically, the pilot method applies the same rules – visiting unexplored subproblems in order of best bounds and fathoming a subproblem if the bound is worse than an incumbent solution – with only one essential difference: the heuristic pilot method bounds each (sub)problem with an upper bounding procedure, whereas the exact branch and bound algorithm would calculate a lower bound. As a better upper bound is of lower objective value, the algorithm persistently chooses as the next subproblem to be branched the subproblem of the new incumbent solution. One returns the incumbent as final solution when it does not move further down, to a child problem with a better upper bound. (At this point all open subproblems are "fathomed" with a bound equal or worse than the incumbent.)

Imagine a branch and bound algorithm for the TSP that successively fixes edge variables $x_1, x_2, \ldots, x_n$ of the tour; for edge x_1 starting with vertex 1, we must fix the other incident vertex, then in a child problem the other vertex of edge x_2 adjacent to x_1 is to be fixed and so on. With the nearest neighbor heuristic as an upper bounding procedure, we first produce the ordinary heuristic result on the main problem, say of objective value $p(1)$. Then the nearest neighbor heuristic is separately performed on $n - 1$ subproblems, each time with a different first edge $x_1 = (1, i)$ being fixed; thus producing pilot results $p(i)$ for $i = 2, 3, \ldots, n$. The subproblem with the best bound, say $x_1 = (1, i_0)$ has pilot result $p(i_0) = \min_{i \in V - \{1\}} p(i)$, is chosen for further branching, i.e., edge $(1, i_0)$ is permanently accepted as the first edge of the partial solution, and so on.

The quickest stopping rule would be to stop as soon as none of the child nodes has a better pilot result than the parent node. We favor a more comprehensive but slower stopping rule that continues the branching when the best pilot result matches the former incumbent result. The reason is that branching in a later part of the algorithm can still lead to a better solution. This may also be true for other non-chosen subproblems having a bound equal to the incumbent (or almost equal); so why not branch them also (or allow for hill climbing, i.e., intermediate deterioration with respect to the objective function value)? The answer is simply that we cannot do this in the context of a heuristic of polynomial time complexity. On the other hand, the strength of the method is that it already applies much trial and error and hill climbing. When in a subproblem the next edge is fixed we do perform hill climbing if this edge is unattractive. The pilot heuristic is there to check whether a mature solution with this edge can still be most attractive; if so, it will be chosen. Thus the pilot method is a tempered greedy method.

Finally, let us note that if infeasible solutions must be taken into account, this can be dealt with by penalizing infeasibility in the objective function. While in the extreme case $p(e)$ can be set to infinity for infeasible solutions, more subtly differentiated approaches may be appropriate depending on the application (see Section 5).

2.1. Related research

The pilot method was first invented in the early 1990s for the Steiner tree problem in graphs (see Duin and Voß, 1994 for a survey). Later, similar ideas were developed under different names. The most famous one is the *rollout method* by Bertsekas, Tsitsiklis, and Wu (1997). While we have motivated the pilot method as a "branch and bound using opposite bounds", the look ahead mechanism of the rollout method is based on dynamic programming exploiting the same concept of repetition as the pilot method. Applications are given in Bertsekas, Tsitsiklis, and Wu (1997), Bertsekas and Castanon (1999), and Meloni, Pacciarelli, and Pranzo (2004). The latter paper shows how elementary rollout methods (i.e. pilot methods) perform very well in the difficult area of scheduling problems. Two other successful applications are presented by Sourd (2001) and Sourd and Chrétienne (1999), who apply a meta-heuristic named "Branch and Greed"; this method is identical to the pilot method.

Additional concepts that are related to the pilot method are the manifold search of Gavish (1991), the fan approach of Glover, Taillard, and de Werra (1993), iterative greedy heuristics of Amberg et al. (1999) as well as some other ideas e.g., (Balas and Vazacopoulos, 1998; Belvaux et al., 1998).

As an example, the basic construction of the *fan approach* (Glover, Taillard, and de Werra, 1993; Glover, 1998) underlies the use of a candidate list strategy within a local search algorithm based on discovering promising moves by applying evaluations in successive levels of a neighborhood tree, 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.

Guided local search (Balas and Vazacopoulos, 1998; Voudouris and Tsang, 2003) is a penalty-based meta-heuristic that is used in connection with local search algorithms with the aim of improving them by penalizing certain features. That is, whenever the local search algorithm settles in a local optimum, the objective function of the problem under consideration is augmented to favor certain features or to avoid them. For instance, for the TSP the inclusion of certain edges could be avoided by modifying their length.

In the *limited discrepancy search* (see, e.g., Harvey and Ginsberg, 1995, Caseau, Laburthe, and Silverstein, 1999) a greedy heuristic is transformed into a search algorithm by branching in a limited number of cases when the choice criterion of the heuristic observes some borderline case or where the choice is not compelling, respectively.

2.2. *Speeding up a pilot method*

In each iteration of a pilot method a considerable computational effort is necessary, leading to a rather high time complexity. Suppose that one run of the subordinate, the pilot heuristic, has time complexity $O(f(t))$, where t represents the size of the problem instance. Since one chooses in the pilot method the elements of the final master solution M one by one from E, each time when computing a pilot solution for all $e \in E - M$, one can expect for the pilot method the worst case running time not to exceed $O(|M||E| f(t))$.

To diminish the time requirements one can resort to parallel processing, obtaining different pilot solutions simultaneously (diminishing time factor $|E|$). However, there are also other prospects to speed up a pilot method. For instance, with a different implementation than the straightforward one it may be possible to reduce the time complexity of the subordinate heuristic. One can also limit the number of iterations by modifying the master solution each time to a greater extent (diminishing $|M|$). A filtering approach diminishes time factor $|E|$. It executes the pilot evaluations $p(e)$ only for e within a limited set of candidates. For example, the candidate list could be obtained by computing with short-cuts approximate pilot values. Finally, as we investigate in this paper, it can be reasonable to restrict the pilot process to a given *evaluation depth*, say $D << |E|$. That is, the pilot method is performed up to an incomplete solution (e.g., partial assignment) with k elements and then completed by continuing with a conventional greedy heuristic. We will see, in shortened pilot methods for different problems, how much smaller than $|E|$ the value of D can be, without seriously influencing the solution quality. The other options of speeding up were tested successfully in Duin and Voß (1999) with respect to the Steiner problem in graphs.

2.3. *Improvement methods*

The basic idea of (greedy) local search is to successively perform at each step a move (a most promising one, according to some measure) within a given neighborhood structure. As such steepest descent methods may lead to local optima of non-satisfying solution quality, the application of modern heuristic search methods like simulated annealing and various tabu search approaches is more appropriate.

The core of the pilot method is heuristic repetition; a consequent characteristic – in terms of local search – is neighborhood extension. The look ahead mechanism of the pilot method relates to an increased neighborhood depth. By exploiting the evaluation of neighbors at larger depths the pilot method can guide the neighbor selection at depth one. We may strive to improve likewise the quality of the neighborhood, both for greedy local search and for modern heuristic search methods, incorporating evaluations by means of a pilot strategy to guide the search into promising regions of the search space. To evaluate the outcome of a specific local search step (or choice) the pilot process may perform a number of iterations. Note that in this case the termination criterion for the pilot process may control the total running time.

We will report in Section 6 on a hybrid strategy combining the pilot method with a tabu search approach.

3. Pilot method for the continuous flow shop scheduling problem

It may be interesting to investigate the pilot method with respect to the TSP or other types of problems that are represented by means of permutations. Here we consider a special case of the time dependent traveling salesman problem (TDTSP); see, e.g., Gouveia and Voß (1995). Note that the TDTSP itself is a special case of the quadratic assignment problem (QAP). As a specific example for an application of the TDTSP we consider the continuous flow shop scheduling problem (see Fink and Voß, 2003 and the references therein).

Flow shop scheduling problems focus on processing a given set of n jobs, where each job has to be processed in an identical order on a given number of m machines. Each machine can process only one job at a time. Continuous flow shop scheduling problems have the additional restriction that the processing of each job has to be continuous, i.e., once the processing of a job begins, there must not be any waiting times between the processing of any consecutive tasks of this job. This may be due to technological restrictions or lack of intermediate storage capacity between the processing stages (machines) if completed tasks are not allowed to remain on a machine. Such problems occur, e.g., in chemical or steel production processes. See, for instance, Dudek, Panwalkar, and Smith (1992), who emphasize the practical importance of this problem type. Continuous processing of a job generally determines an inevitable delay d_{ik}, $1 \leq i \leq n$, $1 \leq k \leq n$, $i \neq k$, on the first machine between the start of job i and job k. The delay may be computed as

$$d_{ik} = \max_{1 \leq j \leq m} \left\{ \sum_{h=1}^{j} t_{ih} - \sum_{h=2}^{j} t_{k,h-1} \right\}$$

where t_{ij}, $1 \leq i \leq n$, $1 \leq j \leq m$ denotes the processing time of job i on machine j. The objective is to construct a permutation $\pi = (\pi_1, \ldots, \pi_n)$ of the jobs that minimizes some given objective function. Here π_i denotes the job that is positioned at the i-th position of a schedule. The objective considered here is to minimize the total processing time (flow time)

$$F(\pi) = \sum_{i=2}^{n} (n + 1 - i) d_{\pi(i-1),\pi(i)} + \sum_{i=1}^{n} \sum_{j=1}^{m} t_{ij} \, .$$

3.1. Application of the pilot method

We explore the pilot method with a cheapest insertion heuristic (called CHINS) as the underlying application process. CHINS is applied for all possible local steps. That is, in each iteration we perform the cheapest insertion heuristic for all incomplete solutions (partial sequences) resulting from adding some not yet included job at some position to the current incomplete solution. As this leads to a time complexity of $O(n^6)$ it may be reasonable to restrict the pilot process to a given evaluation depth (see the experimental results presented below). That is, the pilot method is performed until an incomplete solution with a given number of jobs is reached (and, hopefully, the most significant construction decisions have been done); this solution is completed by continuing with a conventional (myopic) cheapest insertion heuristic. Note that an evaluation depth of 1 corresponds to repeating CHINS for all possible initial jobs.

3.2. Computational results

In Fink and Voß (2003) we apply the heuristics for benchmark data sets of Taillard (1993) which have been generated for unrestricted flow shop scheduling. The data sets are available from the OR library.[1] We use problem instances with 20 (ta001–ta030), 50 (ta031–ta060), 100 (ta061–ta090), and 200 (ta091–ta110) jobs. These instances partly differ in the number of machines, which is mostly irrelevant for our purposes. So we present results for problem instances of the same number of jobs. To assess our heuristic results we used a mixed integer programming formulation based on a formulation of the QAP in Picard and Queyranne (1978) to compute optimal results for the problem instances with 20 jobs and to compute lower bounds provided by the linear programming (LP) relaxation for problem instances with 50 and 100 jobs. For $n = 200$ we compare to the best results obtained during all experiments. Generally, the results are stated as average percentage deviations (dev.). Computation times are given as average CPU time (t) in seconds on a Pentium II with 266 MHz.

Table 1 shows the results of the application of simple construction methods and the pilot method. The deviations for the identity permutations (in accordance with the numbering of the jobs), given as reference, show that there is a large potential for optimization.

Table 1
Average deviations due to the application of different construction methods and the pilot method.

	$n = 20$		$n = 50$		$n = 100$		$n = 200$	
	dev. (%)	t	dev. (%)	t	dev. (%)	t	dev. (%)	t
Identity	43.98	–	59.61	–	69.26	–	74.15	–
NN	25.81	0.0	29.88	0.0	30.21	0.1	21.13	0.3
Pilot-1-NN	19.15	0.0	26.49	0.2	27.90	1.3	19.62	11.2
CHINS	3.21	0.0	4.52	0.1	6.18	0.2	2.79	1.6
Pilot-1-CHINS	0.93	0.0	2.99	1.3	4.32	20.1	1.62	315.0
Pilot-10-CHINS	0.27	0.8	1.59	37.5	3.18	879.0	0.65	7612.4
Pilot-20-CHINS	0.25	1.2	1.26	107.6	2.81	3068.4	–	–
Pilot-CHINS	0.25	1.2	1.23	189.4	2.26	8217.2	–	–

The results of the nearest neighbor heuristic (NN, i.e. appending at each step a not yet included job with a minimal inevitable delay to the last job of the yet incomplete sequence), even when iteratively performed for all jobs used as initial job (Pilot-1-NN), are much worse than those of CHINS. Repeating CHINS for all jobs treated as initial job (Pilot-1-CHINS) leads to clearly better results at the expense of an increased running time, which is especially relevant for the larger problem instances. The last three rows show the results of the pilot method with an evaluation depth of 10 (Pilot-10-CHINS), 20 (Pilot-20-CHINS), or an unbounded evaluation depth (Pilot-CHINS), respectively. While these results are of high quality the self-imposed running time limit of 10,000 seconds hindered the completion of the unbounded pilot method and Pilot-20-CHINS for the problem instances with 200 jobs. It should be noted that the majority of the small problem instances with 20 jobs are not solved to optimality by the considered construction methods (e.g., the unbounded pilot method solved 10 out of 30 instances to optimality).

Figures 1 and 2 show the effect of the evaluation depth on the effectiveness of the pilot method for ten instances each with $n = 50$ and $n = 100$, respectively. These results confirm the expectation that the most important decisions of the construction process are made at the first steps. So it seems reasonable to use a rather small evaluation depth when running time is important. To supplement these results it should be noted that we have obtained similar results regarding the evaluation depth for another permutation-type problem, i.e., the pattern sequencing problem. Here we are given a set of n patterns that have to be located in a sequence (e.g., patterns may represent groupings of some locations) with the objective of minimizing the average location spread (see, e.g., Fink and Voß, 1999 for a detailed description of the pattern sequencing problem as well as a comparison of different meta-heuristics such as simulated annealing and tabu search).

Further results from Fink and Voß (2003) indicate that the unbounded pilot method may provide solutions of equal quality when compared to state-of-the-art simulated annealing or tabu search implementations if additional computation time is allowed.

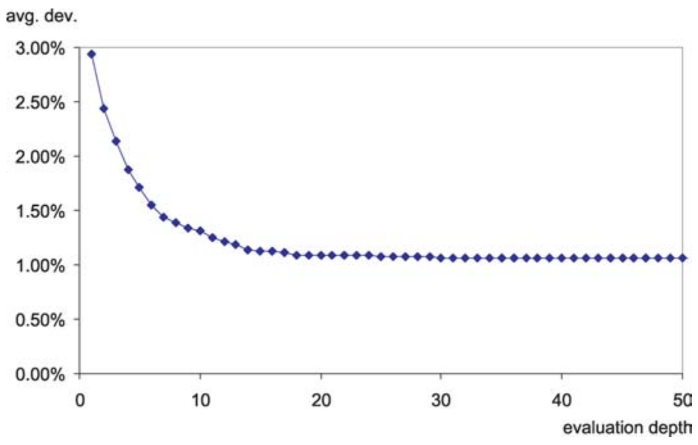

Figure 1. Results at different evaluation depths (ta041-050 with $n = 50$).

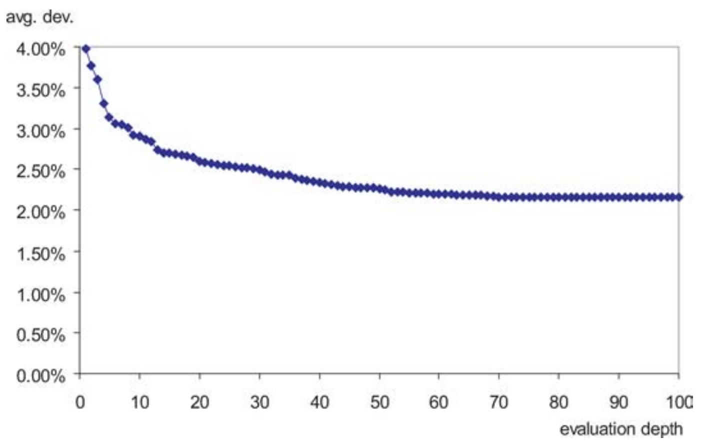

Figure 2. Results at different evaluation depths (ta071-080 with $n = 100$).

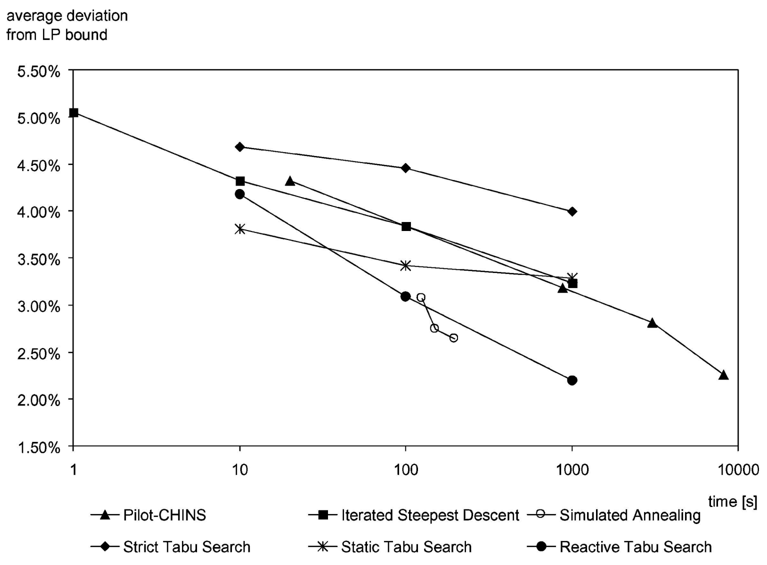

Figure 3. Solution quality vs. running time for problem instances with $n = 100$ jobs.

Figure 3 shows this for instances with 100 jobs. This motivated the research question whether an integration of these methods could lead to improved results. The first research results are provided for the ring load balancing problem in Section 6.

4. Pilot method for the ring network design problem

The *ring network design problem* (*RNDP*), as it has been considered by Gendreau, Labbé, and Laporte (1995), is a special case of the *general ring network design problem*

(*GRNDP*) according to Fink, Schneidereit, and Voß (2000). Given a graph G with vertex set $V = \{1, \dots, n\}$ and edge set E with non-negative edge weights c_{ij} for all $(i, j) \in E$, the RNDP is defined as follows. A solution is a ring $R = (i_1, \dots, i_k)$ with corresponding edges $((i_1, i_2), \dots, (i_{k-1}, i_k), (i_k, i_1))$. Whenever a ring contains a direct link between i and j, a cost c_{ij} is incurred. That is, only fixed construction costs are taken into account, and the sum of construction costs of R is given by $c(R) = \sum_{j=1}^{k-1} c_{i_j, i_{j+1}} + c_{i_k, i_1}$. A budget b limits the construction costs by $c(R) \leq b$. For every pair of nodes $i, j, i < j$, a revenue r_{ij} is obtained if and only if i and j both belong to the ring. The objective is to construct a feasible ring with maximum profit as defined by the obtained revenues minus the incurred construction costs.

4.1. Application of the pilot method

The application of the pilot method to the RNDP is similar to the application to the continuous flow shop scheduling problem. Starting from an empty solution (ring), we successively construct a larger ring by considering all possibilities for including some not yet included node at some position into the current ring until the budget constraint prevents any such extension. In each iteration of the pilot method we evaluate all such options for extension by means of the conventional (myopic) cheapest insertion procedure (i.e., by the respective objective function value eventually obtained). As discussed before, we consider restricting the pilot process to some evaluation depth. That is, the pilot method is performed until an incomplete solution with a given number of nodes is reached; this solution is completed by continuing with a conventional cheapest insertion heuristic.

4.2. Computational results

In order to assess the pilot method, in Fink, Schneidereit, and Voß (2000) we have generated random instances of the RNDP according to a procedure described in Gendreau, Labbé, and Laporte (1995). For each instance, first coordinates of n points P_i were randomly generated in the unit square according to a continuous uniform distribution. Then, each of the values c_{ij} was set to the Euclidean distance between P_i and P_j. The budget value was defined as $b = 0.75 \cdot \sqrt{n/2}$. For all pairs (i, j) with $i < j$ the revenue $r_{ij} = r_{ji}$ was randomly determined according to a continuous uniform distribution in $(0, 18.4 \cdot b/n^2)$. The pilot method was applied for randomly generated data sets with 40, 80, and 120 nodes (ten problem instances each).

The pilot method obtained very good results, although it required unreasonable computation times (Fink, Schneidereit, and Voß, 2000). Therefore, we again examined the effect of limiting the evaluation depth. The development of the pilot process is shown in Figure 4. The lines show the best objective function value obtained during the application of the pilot method for the ten problem instances with 80 nodes. These results are similar to those of, e.g., the continuous flow shop problem (note that the RNDP is a maximization problem). As can be seen, an evaluation depth of approximately 10 seems

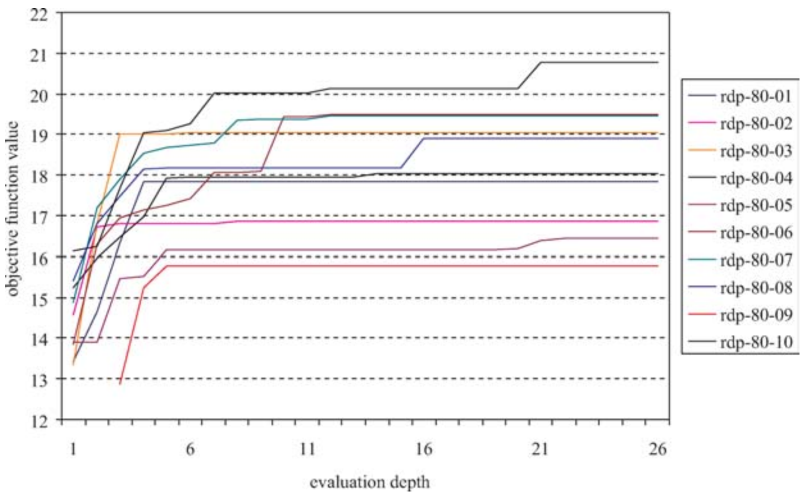

Figure 4. Development of best objective function values for the pilot method for instances with 80 nodes.

reasonable to speed up the computation, where in this case an evaluation depth of 22 would have led to the same results as the unbounded pilot method.

5. Pilot method for the minimum weight vertex cover problem

The minimum weight vertex cover problem as a basic problem underlies specific real-world situations from areas such as location theory or telecommunications network and circuit design. Given an undirected graph $G = (V, E)$ and a positive weight w_v for each vertex $v \in V$ the objective is to determine a subset $V' \subseteq V$ satisfying $\cup_{v \in V'} e(v) = E$, where $e(v)$ denotes the set of edges incident to vertex v, such that the sum of the incurred costs $\sum_{v \in V'} w_v$ is minimized. That is, for each edge in E, at least one of its two incident vertices must belong to the selected vertex set. The minimum weight vertex cover problem is one of the classic $\mathcal{NP}$-hard problems as proven by Karp (1972). Up-to-date references on the minimum weight vertex cover problem are provided in Shyu, Yin, and Lin (2004).

5.1. Application of the pilot method

Following the lines of the preceding applications we use a pilot strategy for enhancing a construction procedure. Yet we apply the pilot strategy in connection with an improvement method which operates on a solution space that includes incomplete (infeasible) solutions. We take into account infeasibilities by adding appropriate penalty values to the objective function. For each edge $e = (i, j)$ that is not covered by the considered vertex set (i.e., $i \notin V'$ and $j \notin V'$), we add a penalty value

$$\min\{w_i(1 + 1/\gamma_i), w_j(1 + 1/\gamma_j)\}$$

to the objective function, where γ_v denotes the degree of a vertex $v \in V$. This penalizes uncovered edges by taking into account the weights as well as the degrees of incident vertices. We start with an empty solution (i.e., the set V' of selected vertices is empty) and successively choose the best local move (regarding the inclusion or exclusion of one vertex) by evaluating such neighbors with a steepest descent until a local optimum, and with that a feasible solution, is obtained.

5.2. Computational results

We assess the pilot method in comparison to the recent research of Shyu, Yin, and Lin (2004), where best overall results have been obtained by means of an advanced ant-colony optimization (ACO) algorithm. We use a set of large problem instances from Shyu, Yin, and Lin (2004) with 300 vertices and the number of edges ranging from 300 to 5000. For each scenario there are ten problem instances. We report average objective function values (avg. res.) and average deviations (avg. dev.) from the best results of Shyu, Yin, and Lin (2004) as reference (see Table 2). Due to large computation times we restrict ourselves to small evaluation depths from 1 up to 20. (Computation times with respect to using a Pentium IV with 1.8 GHz are about 20 seconds per problem instance for an evaluation depth of 1. The CPU times scale linearly for the considered range of evaluation depth values.) Table 2 shows the results for an evaluation depth of 1, 10, and 20 (Pilot-1, Pilot-10, and Pilot-20, respectively) in comparison to the ACO results of Shyu, Yin, and Lin (2004). Figure 5 further shows the effect of an increasing evaluation depth from 1 up to 10 (also for the set of 70 problem instances with 300 nodes). The outcome resembles the computational results for the other problems considered in this paper as high-quality results can be obtained even for small evaluation depths.

6. Pilot method for the ring load balancing problem

In this section we investigate the possibility of developing hybrid methods combining the pilot method with local search and tabu search. In particular, we develop hybrids

Table 2
Numerical results for the minimum weight vertex cover problem.

	ACO	Pilot-1		Pilot-10		Pilot-20	
$(\|V\|;\|E\|)$	Avg. res.	Avg. res.	Avg. dev. (%)	Avg. res.	Avg. dev. (%)	Avg. res.	Avg. dev. (%)
(300;300)	7342.7	7382.0	0.54	7298.9	−0.60	7298.9	−0.60
(300;500)	9517.4	9623.6	1.12	9450.0	−0.71	9449.6	−0.71
(300;750)	11166.9	11219.0	0.47	11068.7	−0.88	11068.5	−0.88
(300;1000)	12241.7	12249.7	0.07	12146.4	−0.78	12136.3	−0.86
(300;2000)	14894.9	14912.4	0.12	14836.4	−0.39	14817.2	−0.52
(300;3000)	16054.1	16024.9	−0.18	15960.3	−0.58	15952.4	−0.63
(300;5000)	17545.4	17473.6	−0.41	17405.2	−0.80	17405.2	−0.80
Average			0.24		−0.68		−0.72

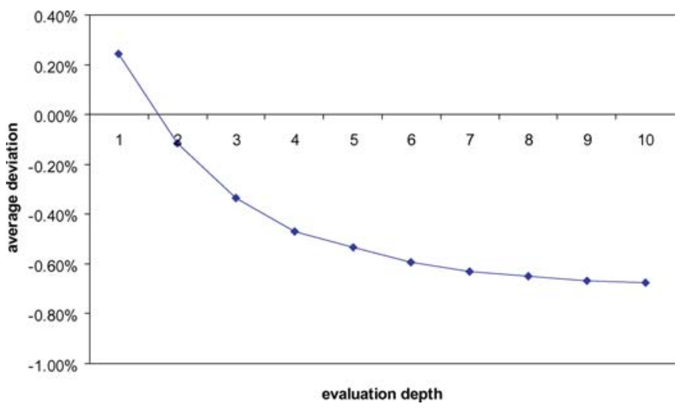

Figure 5. Average deviations of the pilot method applied to minimum weight vertex cover problem instances with 300 nodes.

that incorporate pilot strategies into steepest descent/mildest ascent and tabu search. The problem under consideration is the ring load balancing problem which comes from the so-called SONET technology. A SONET ring consists of a set of connected facilities deployed in ring configuration with parallel protection channels. In the event of cable failures the traffic, such as telephone calls moving in one direction, e.g. clockwise, is reversed counterclockwise such that calls are not interrupted.

Here we consider the problem of balancing the loads for bidirectional communication rings for the case when demand splitting is not allowed, which occurs as a part of SONET design (Cosares and Saniee, 1994; Myung, Kim, and Tcha, 1997). It should be noted that the ring load balancing problem is not a pure design problem on a strategic level as we assume a given ring whose loads have to be determined. Furthermore, from an algorithmic point of view we need to develop fast and effective methods as the load balancing has to be performed several times with fast response requirements.

Given is a ring of nodes with a set of communication demands between node pairs. Assuming that the communication demands occur simultaneously, the task is to decide for each demand whether to route it clockwise or counterclockwise, minimizing the maximum bandwidth requirement on any of the links between adjacent nodes. That is, given a set of n nodes and a set of demands between pairs of nodes, find a direction for each of the demands so that the maximum of the loads on the links in the network is as small as possible. Figure 6 shows a problem instance with five nodes where communication demands d_{ij} for the communication between nodes i and j are shown as dotted lines with respective values. (For ease of exposition we assume $i < j$ for the visual representation of the demands.)

6.1. Application of the pilot method

We explore the incorporation of the pilot method into improvement procedures with the aim of enhancing the quality of the evaluation of the neighborhood; see Section 2.3.

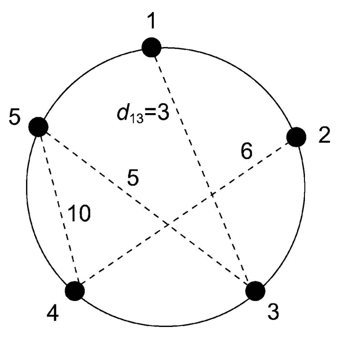

Figure 6. A sample problem instance of the ring load balancing problem.

We employ the straightforward neighborhood defined by switching the routing direction for one demand (node pair). The quality of such a local search step is assessed by the quality of the solution eventually obtained after performing a number of iterations of a pilot process. In particular, we enhance a simple steepest descent/mildest ascent (SDMA) approach as well as reactive tabu search (RTS) by evaluating neighbors with an SDMA pilot for ten iterations. This results in two hybrid approaches SDMA/SDMA-10 and RTS/SDMA-10 where in each case the former acronym describes the guiding process and the latter acronym describes the pilot process.

6.2. Computational results

We report the numerical outcomes of our approaches for the hardest problem instances of the ring load balancing problem from Cosares and Saniee (1994), Fink and Voß (2001), and Myung, Kim, and Tcha (1997). These instances may be characterized by means of the demands which are decentralized. In Table 3 we denote the number of nodes by n which ranges from 5 to 30. For the demands of the considered scenario the number p of origin-destination pairs ranges between 10 and 435. For each combination of n and the different demand scenarios we have ten problem instances. The demands themselves are

Table 3
Numerical results for the ring load balancing problem.

		Pilot method					
$(n;p)$	Opt	SDMA/SDMA-10		RTS/SDMA-10		Reactive tabu search	
(5;10)	186.0	186.0	0.00%	186.0	0.00%	186.0	0.00%
(10;45)	728.3	728.5	0.03%	728.3	0.00%	728.3	0.00%
(15;105)	1599.9	1600.3	0.03%	1599.9	0.00%	1602.2	0.14%
(20;190)	2720.2	2726.4	0.23%	2720.3	0.00%	2736.2	0.59%
(25;300)	4218.4	4219.3	0.02%	4219.3	0.02%	4238.7	0.48%
(30;435)	5943.3	5943.5	0.00%	5943.3	0.00%	5987.7	0.75%
Average deviation:			0.05%		0.004%		0.33%

randomly selected within the interval [5, 100]. All computations have been performed on a Pentium II with 266 MHz using a time limit of 100 CPU seconds as termination criterion to have a fair comparison.

Among those heuristics previously tested on these instances the RTS implementation of Fink and Voß (2001) seems to be the most efficient method. Here we investigate the two pilot method hybrids as described in the previous section in comparison with the original version of the RTS according to Fink and Voß (2001). The results in Table 3 (giving average objective function values and the percentage deviation from optimality) reveal that it is advantageous to modify the RTS with the look ahead mechanism of the pilot method as the results may be considerably improved. Even the steepest descent mildest ascent approach, which is usually clearly outperformed by RTS, is able to provide better results compared to the RTS when hybridized with the pilot method.

7. Conclusions

The pilot method may be regarded as an effective meta-heuristic to overcome the greedy trap. As the basic mechanism of the pilot method requires a considerable computation time it is reasonable to think about all means of speeding up the objective value tests. The focus of this paper was to restrict the evaluation depth. Numerical results for various problems (both summarizing results from the literature as well as new results for problems where the pilot method had previously not yet been applied) show that even small values of the evaluation depth allow for considerable enhancements of the solution quality.

Thus a shorter evaluation depth is a simple and yet effective means for speeding up the pilot method. However, it seems hard, if not impossible, to provide a general answer to the question: what evaluation depth is best. It depends on several factors: the nature of the problem, the importance of a (further) improvement, the CPU time to spare, et cetera. Most important in this respect seems to be computational experience.

Section 6 of this paper contained an experiment showing that the strategy of the pilot method can also enhance existing local search methods. Future research may study such mixed strategies to improve the quality of modern meta-heuristics. To decrease the time requirements, more advanced policies for speeding up the pilot method, combining a shorter evaluation depth with other options (see Section 2.2), can be investigated. Further, much more comprehensive numerical testing is required.

Acknowledgemnts

We greatly appreciate the comments of three anonymous referees and one of the editors which contributed to an improved presentation of the paper. This paper was presented at ISOLDE, Fredericton; an earlier version was presented at the First International Workshop on Heuristics, Beijing.

Note

1. http://people.brunel.ac.uk/~mastjjb/jeb/info.html

References

Amberg, A., L. Gouveia, P. Martins, and S. Voß. (1999). "Iterative Heuristic Metastrategies." In *MIC'99, Third Metaheuristics International Conference*, Angra dos Reis, Rio de Janeiro, Brazil, July 19–23, 1999, pp. 13–16.

Balas, E. and A. Vazacopoulos. (1998). "Guided Local Search with Shifting Bottleneck for Job Shop Scheduling." *Management Science* 44, 262–275.

Belvaux, G., N. Boissin, A. Sutter, and L.A. Wolsey. (1998). "Optimal Placement of Add/Drop Multiplexers: Static and Dynamic Models." *European Journal of Operational Research* 108, 26–35.

Bertsekas, D.P. and D.A. Castanon. (1999). "Rollout Algorithms for Stochastic Scheduling Problems." *Journal of Heuristics* 5, 89–108.

Bertsekas, D.P., J.N. Tsitsiklis, and C. Wu. (1997). "Rollout Algorithms for Combinatorial Optimization." *Journal of Heuristics* 3, 245–262.

Caseau, Y., F. Laburthe, and G. Silverstein. (1999). "A Meta-Heuristic Factory for Vehicle Routing Problems." In J. Jaffar (ed.), *Principles and Practice of Constraint Programming – CP '99*, *Lecture Notes in Computer Science*, 1713, Berlin: Springer, pp. 144–158.

Cosares, S. and I. Saniee. (1994). "An Optimization Problem Related to Balancing Loads on SONET Rings." *Telecommunication Systems* 3, 165–181.

Dudek, R.A., S.S. Panwalkar, and M.L. Smith. (1992). "The Lessons of Flowshop Scheduling Research." *Operations Research* 40, 7–13.

Duin, C.W. and S. Voß. (1994). "Steiner Tree Heuristics – A Survey." In H. Dyckhoff, U. Derigs, M. Salomon, and H.C. Tijms. (Eds.), *Operations Research Proceedings 1993*, Berlin: Springer, pp. 485–496.

Duin, C.W. and S. Voß. (1999). "The Pilot Method: A Strategy for Heuristic Repetition with Application to the Steiner Problem in Graphs." *Networks* 34, 181–191.

Fink, A., G. Schneidereit, and S. Voß. (2000). "Solving General Ring Network Design Problems by Meta-Heuristics." In M. Laguna and J.L. González Velarde (eds.), *Computing Tools for Modeling, Optimization and Simulation*, Boston: Kluwer, pp. 91–113.

Fink, A. and S. Voß. (1999). "Applications of Modern Heuristic Search Methods to Pattern Sequencing Problems." *Computers & Operations Research* 26, 17–34.

Fink, A. and S. Voß. (2001). "Efficient Meta-Heuristics Approaches for Ring Load Balancing." In *Proceedings of the 9th International Conference on Telecommunication Systems*, 243–250. Southern Methodist University, Dallas.

Fink, A. and S. Voß. (2003). "Solving the Continuous Flow-Shop Scheduling Problem by Metaheuristics." *European Journal of Operational Research* 151, 400–414.

Garey, M.R. and D.S. Johnson. (1979). *Computers and Intractability – A Guide to the Theory of NP-completeness*." New York: Freeman.

Gavish, B. (1991). "Manifold Search Techniques Applied to Quadratic Assignement Problems (QAP)." Working paper, Owen Graduate School of Management, Vanderbilt University, Nashville.

Gendreau, M., M. Labbé, and G. Laporte. (1995). "Efficient Heuristics for the Design of Ring Networks." *Telecommunication Systems* 4, 177–188.

Glover, F. (1998). "A Template for Scatter Search and Path Relinking." In J.-K. Hao, E. Lutton, E. Ronald, M. Schoenauer, and D. Snyers (eds.), *Artificial Evolution*, *Lecture Notes in Computer Science*, 1363, Berlin: Springer, pp. 13–54.

Glover, F., E. Taillard, and D. de Werra. (1993). "A User's Guide to Tabu Search." *Annals of Operations Research* 41, 3–28.

Gouveia, L. and S. Voß. (1995). “A Classification of Formulations for the (Time-Dependent) Traveling Salesman Problem.” *European Journal of Operational Research* 83, 69–82.

Harvey, W. and M. Ginsberg. (1995). “Limited Discrepancy Search.” In *Proceedings of the 14th International Joint Conference on Artificial Intelligence (IJCAI-95)*, Morgan Kaufmann, San Mateo, pp. 607–615.

Karp, R. (1972). “Reducibility Among Combinatorial Problems.” In R.E. Miller and J.W. Thatcher (eds.), *Complexity of Computer Computations*, New York: Plenum Press, pp. 85–103.

Lawler, E.L., J.K. Lenstra, A.H.G. Rinnooy Kan, and D.B. Shmoys. (1985). *The Traveling Salesman Problem*. Chichester: Wiley.

Meloni, C., D. Pacciarelli, and M. Pranzo. (2004). “A Rollout Metaheuristic for Job Shop Scheduling Problems.” *Annals of Operations Research* 131, 215–235.

Myung, Y.-S., H.-G. Kim, and D.-W. Tcha. (1997). “Optimal Load Balancing on SONET Bidirectional Rings.” *Operations Research* 45, 148–152.

Pearl, J. (1984). Heuristics: Intelligent Search Strategies for Computer Problem Solving. Reading, Addison-Wesley.

Picard, J.-C. and M. Queyranne. (1978). “The Time-Dependent Traveling Salesman Problem and its Application to the Tardiness Problem in One-Machine Scheduling.” *Operations Research* 26, 86–110.

Rayward-Smith, V.J. and A. Clare. (1986). “On Finding Steiner Vertices.” *Networks* 16, 283–294.

Shyu, S.J., P.-Y. Yin, and B.M.T. Lin. (2004). “An Ant Colony Optimization Algorithm for the Minimum Weight Vertex Cover Problem.” *Annals of Operations Research* 131, 283–304.

Sourd, F. (2001). “Scheduling Tasks on Unrelated Machines: Large Neighborhood Improvement Procedures.” *Journal of Heuristics* 7, 519–531.

Sourd, F. and P. Chrétienne. (1999). “Fiber-to-object Assignment Heuristics.” *European Journal of Operational Research* 117, 1–14.

Taillard, E. (1993). “Benchmarks for Basic Scheduling Instances.” *European Journal of Operational Research* 64, 278–285.

Voß, S. (1990). *Steiner-Probleme in Graphen*. Frankfurt am Main: Hain.

Voß, S., S. Martello, I.H. Osman, and C. Roucairol (eds.) (1999). *Meta-Heuristics: Advances and Trends in Local Search Paradigms for Optimization*. Boston: Kluwer.

Voudouris, C. and E.P.K. Tsang. (2003). “Guided Local Search.” In F. Glover and G.A. Kochenberger (eds.), *Handbook of Metaheuristics*. Boston: Kluwer, pp. 185–218.