Effective Local and Guided Variable Neighbourhood Search Methods for the Asymmetric Travelling Salesman Problem

Edmund K. Burke, Peter I. Cowling, and Ralf Keuthen

Automated Scheduling, Optimization, and Planning Group (ASAP),
School of Computer Science & IT,
University of Nottingham,
Jubilee Campus,
Nottingham, NG8 1BB, UK
{ekb, pic, rxk}@cs.nott.ac.uk
http://www.asap.cs.nott.ac.uk

Abstract. In this paper we present effective new local and variable neighbourhood search heuristics for the asymmetric Travelling Salesman Problem. Our local search approach, HyperOpt, is inspired by a heuristic developed for a sequencing problem arising in the manufacture of printed circuit boards. In our approach we embed an exact algorithm into a local search heuristic in order to exhaustively search promising regions of the solution space. We propose a hybrid of HyperOpt and 3-opt which allows us to benefit from the advantages of both approaches and gain better tours overall. Using this hybrid within the Variable Neighbourhood Search (VNS) metaheuristic framework, as suggested by Hansen and Mladenović, allows us to overcome local optima and create tours of very high quality. We introduce the notion of a "guided shake" within VNS and show that this yields a heuristic which is more effective than the random shakes proposed by Hansen and Mladenović. The heuristics presented form a continuum from very fast ones which produce reasonable results to much slower ones which produce excellent results. All of the heuristics have proven capable of handling the sort of constraints which arise for real life problems, such as those in electronics assembly.

1 Introduction

The Travelling Salesman Problem (TSP) is one of the best studied combinatorial optimization problems of our time. The TSP consists of a set of n cities $\{1, 2, \ldots, n\}$, associated with a cost matrix (c_{ij}), with $i, j \in \{1, 2, \ldots, n\}$, defining the travelling costs between cities i and j. The aim of the TSP is to determine a tour of minimum cost visiting each city exactly once and returning to the starting city. In the case that the travelling cost from city i to city j is not necessarily equal to the travelling cost from j to i we speak of an *asymmetric* TSP (ATSP).

Symmetric TSPs (STSPs), especially Euclidean problems where cities lie in a two dimensional plane and use the Euclidean distance metric, have been

very well researched in the literature [21][10] and fast and powerful heuristic approaches have been proposed [1], [15] [21]. The ATSP has been less well studied and the heuristic approaches proposed so far do not match their symmetric counterparts in effectiveness. Many heuristics that are successful for symmetric instances cannot be applied efficiently to asymmetric problems.

The heuristic approaches for the ATSP can be loosely divided into two groups. The first group contains the constructive approaches which create a feasible tour from scratch. These include the well known *Nearest Neighbour*, *Nearest Insertion* and *Multiple Fragment* (often referred to as *Greedy*) algorithms [1][14][21] as well as some approaches based on the *Assignment Problem* (AP)[16]. The Assignment Problem is equivalent to finding a minimum weight matching of the bipartite graph formed by repeating each vertex on both sides of the bipartition and orienting all directed edges in the original city graph in the same direction. This represents a generalization of the ATSP with either a tour or a minimum cycle cover as solution. The *Repeated Assignment Algorithm* is one of these algorithms [6]. It is based on the solutions of the assignment problem where an AP algorithm is repeatedly applied until a valid tour is found. *Patching Algorithms*, are also based on minimum cycle covers where the cycles are repeatedly *patched* (combined) to create a tour. Various patching strategies, like the Karp-Steele Patching heuristic [12], Greedy Karp-Steele Patching or Contract-or-Patch [7] have been suggested in the literature. A truncated branch-and-bound subtour elimination procedure has been introduced in [23].

The second group includes the local search heuristics 3-opt[14], the Kanellakis and Papadimitriou algorithm [11], which is based on the successful Lin-Kernighan heuristic for the STSP [15], and their variants. Further approaches to extend local search either by restarting (iterative or chained approaches) or evolutionary algorithms [17] have been suggested in the literature as well. It has been shown in [18] that an ATSP of n cities can be transformed to a symmetric problem of $3n$ cities. However, since many industrial applications of the TSP are asymmetric further research is necessary and good heuristic approaches are desirable, especially those that can handle the additional constraints that arise in industrial ATSP applications. These applications range from problems in flow shop scheduling [11][19], steel hot strip mill scheduling [5], to sequencing problems for numerically controlled punch press [13] or drilling machines [22] where the asymmetry of the problem occurs due to asymmetry in machine set ups.

The heuristics we describe were inspired by such an industrial application arising in the manufacture of printed circuit boards [2][3]. This application concerns the movement of a placement head for a numerically controlled component placement machine. The placement head moves between a feeder magazine, where the electronic components are stored, and placement locations on the circuit board. When considering placement locations as cities and the movement times between two placement locations as the distance, this sequencing problem can be modelled as a TSP. However, since different component types are usually assigned to different feeder slots in the magazine, the head movement time be-

tween two placement locations is dependent on the movement direction, making the problem asymmetric. The machine we were considering in [2][3] is equipped with multiple placement heads. This allows the machine to pick up as many components as heads are available from the feeder magazine, move back to the circuit board and place them without having to return to the feeder magazine. This feature inspired our heuristic approach where an exact algorithm is used to ensure optimality of these small pick-and-place subsequences. The optimal subsequences are then re-embedded into the original placement sequence. The resulting heuristic represents a novel approach where an exact algorithm is used within the framework of a local search heuristic.

This paper is structured as follows. In the next section we introduce our new heuristic approaches. First we are going to describe the HyperOpt approach for the asymmetric Travelling Salesman Problem. We then propose an approach which is a hybrid of HyperOpt and 3-opt, to benefit from the advantages of both methods. Then we introduce Variable Neighbourhood Search (VNS), as described by Hansen and Mladenović in [8][9], and show how the hybrid method proposed earlier can be used efficiently for local search in a VNS framework. We further introduce guided shakes to accelerate convergence for our VNS approach gaining tours of near optimal quality for the test instances considered. The third section presents the computational results for the ATSP. In the last section we present the conclusions we have drawn from our experiments and give a brief insight into planned future research.

2 Heuristic Methods for the ATSP

In this section we propose new heuristic approaches for the ATSP based on local search. The local search routine we introduce is based on splitting the original problem into small subproblems which are then solved to optimality using an exact algorithm. We discuss advantages and disadvantages of this approach in comparison to local search 3-opt for the ATSP, and introduce a hybrid between 3-opt and our new approach. In order to enable local search to create solutions of very high quality we demonstrate how this hybrid approach can be embedded efficiently into the Variable Neighbourhood Search metaheuristic framework proposed by Hansen and Mladenović in [8][9]. We further investigate how the "shaking" process of the Variable Neighbourhood Search method can be guided using information gained in a phase of pre-processing to improve the metaheuristics performance and find solutions of near optimal quality.

2.1 Local Search for the ATSP

HyperOpt belongs to the class of local improvement algorithms which, starting from an initial solution, repeatedly perform tour modifications for as long as improvements can be found. A variant of the algorithm described here has already been applied successfully to very large instances of the Euclidean Travelling Salesman Problem. This variant compares well to established local search approaches such as 2- and 3-opt [1][10].

The basic idea of HyperOpt local search for the ATSP is as follows. Starting from an initial tour, first two hyperedges are defined by the cities p_0 and q_0 and their h successors in the tour, as illustrated in Fig. 1(a). Next all inner edges of the hyperedges are removed from the tour leaving only the starting and finishing cities of each hyperedge attached to the tour, Fig. 1(b). The unconnected cities are then reconnected to the rest of the tour optimally using a dynamic programming algorithm. This feature of our local search routine, which we refer to as a HyperOpt move, was introduced to ensure optimality of the small pick-and-place sequences arising for multi-headed placement machines. Since the number of placement heads available is limited in practice (it typically lies between 2 and 4) enforcing optimality on these small pick-and-place sequences is computationally still tractable. Two possible outcomes of an asymmetric HyperOpt move with $h = 4$ are shown in Fig. 1(c) and (d). The orientation of the fixed tour segments is not changed, since evaluating the impact of such a change has time complexity $O(n)$, and is very expensive in comparison to the evaluation of a HyperOpt move which requires $(2h-2)^2 \cdot 2^{(2h-2)}$ iterations, and is thus independent of n. Computational results of the HyperOpt approach with parameter h taking values of 3 and 4 will be discussed later in Section 3.

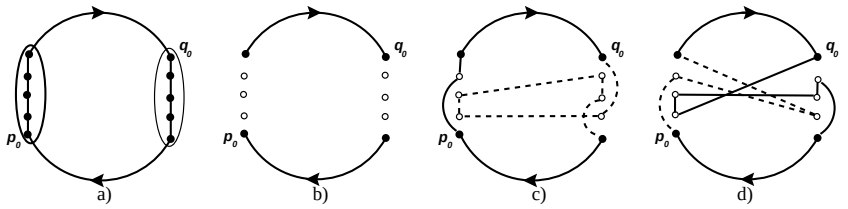

Fig. 1. HyperOpt move

An obvious drawback of the HyperOpt approach for the ATSP is its inability to reorder tour segments involving more than $h-1$ successive cities. However, HyperOpt can perform complex reorderings of the unconnected cities which would be difficult to find using other approaches. In order to overcome this problem we suggest a hybrid of HyperOpt with the simplest tour segment reordering algorithm for the TSP, 3-opt [14]. Having deleted 3 edges from an ATSP tour, there is only one way to reconnect which does not reverse any tour segments, illustrated in Fig. 2 below.

In our hybrid approach we first choose a hyperedge with starting city p_0 and evaluate the possible HyperOpt moves involving this hyperedge. If no improvement is found the tour is searched for an improving 3-opt move involving the edge connecting p_0 and its successor. In section 3 we discuss the computational results achieved by this hybrid approach for 3- and 4-HyperOpt.

To keep computational time low we made use of implementation techniques which proved to work well for local search approaches for the TSP [1][10]. Instead of a *steepest descent* approach, where the best move for a given tour is performed,

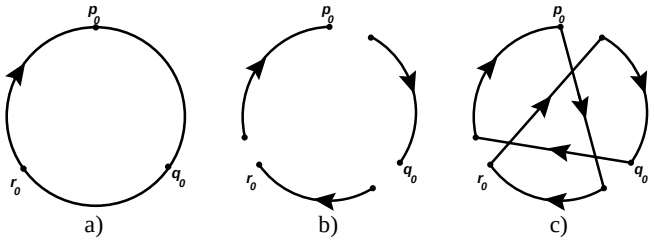

Fig. 2. 3-opt move for ATSPs

we successively select one hyperedge and evaluate for it all possible HyperOpt moves. When one or more improving moves are found, the best move is chosen and performed. The same low cost approach is taken when the tour is searched for a 3-opt move. To further speed up the heuristic approaches we made use of a stack to store hyperedges which are to be searched for improving moves [1][4]. Starting with a stack consisting of all available hyperedges in the tour, only hyperedges which have been affected by a HyperOpt or 3-opt move are stored on this stack and will be considered in later iterations. The aim of this approach, similar to the *no-look-bits* of [1], is to avoid the re-evaluation of moves which have been unsuccessful in previous iterations.

2.2 Variable Neighbourhood Search

In this section we propose an efficient variant of the Variable Neighbourhood Search heuristic proposed by Hansen and Mladenović [8][9]. This extends the work in [4] for symmetric TSPs and in [3] for electronic assembly. We introduce the notion of *guided shakes* for Variable Neighbourhood Search methods as a method to restart local search in order to overcome local optima, improving on the performance of random shaking strategies suggested in the original work by Hansen and Mladenović [8][9].

The aim of Variable Neighbourhood Search approaches is to avoid poor local optima by systematically changing neighbourhood in order to explore an increasingly larger region of the solution space. Hansen and Mladenović suggest various VNS strategies for a wide range of combinatorial optimization problems [8][9]. The VNS strategy we consider here is given in Fig. 3. The major difference between our approach and the VNS approach proposed by Hansen and Mladenović in [8][9] lies in the change of neighbourhoods used to restart the local search. Once a heuristic has been chosen for local search and an initial tour has been determined, the pivotal component of VNS is the *shaking* process. In contrast to the random shakes proposed by Hansen and Mladenović for the STSP we suggest a different strategy which guides the shakes based on information gained about the tour prior to the first shake.

Immediately after the first local search has been performed a list of suspect hyperedges is determined which we will use in our shakes. The quality of a hyperedge $H = (p_0, p_1, \ldots, p_h)$ is estimated by

$$\sum_{i=0}^{h-1} c_{p_i,p_{i+1}} - \min_{q \in \{1,2,\ldots,n\}} (c_{p_i,q}),$$

where a high value indicates a hyperedge of poor quality. We store the $\frac{n}{3}$ hyperedges having the highest value in the list of low quality hyperedges.

For shaking we have chosen a variant of the HyperOpt neighbourhood. Instead of two hyperedges, three hyperedges are removed from the tour and the tour is restored in a random fashion. Reversal of a tour segment is avoided as otherwise too much information from the tour may be destroyed. An illustration of this shaking strategy is shown in Fig. 4.

```
Choose Neighbourhood Structures {N_1, ..., N_m};
Select Local Search procedure;
Create an initial tour T;
Set: T* = T, k = 1;
Do{
   Apply Local Search procedure to T;
   IF (length(T) < length(T*))
      Set: T* = T, k = 1;
   ELSE{ Set T = T*;
         IF (k < m)
           Set k = k + 1;
   }
   Shake Tour T using neighbourhood N_k;
} Until stopping criterion is met;
OUTPUT best found tour: T*;
```

Fig. 3. Variable Neighbourhood Search

To guide the shaking process and destroy segments of the tour of low quality, one of the hyperedges we shake is chosen at random from the list of low quality hyperedges. The remaining two hyperedges for shaking are then selected randomly out of the set of 20 nearest neighbours of the starting city of the low quality hyperedge.

To prevent the local search from falling back to a local optimum visited earlier, the length of the hyperedges used for shaking starts at $k = (h + 1)$, where h denotes the neighbourhood size used for the local search HyperOpt. When local search does not lead to an improvement in tour quality, the length of the shaking hyperedges k is increased by one and the tour is abandoned. In order not to destroy too much of the current tour and considering the size of the test problems we bounded the maximal length of shaking hyperedges by 5 for instances of less than 50 cities and by 6 otherwise.

For local search we have chosen the HyperOpt/3-opt hybrid with HyperOpt parameter $h = 3$ which provided the best compromise between high quality tours

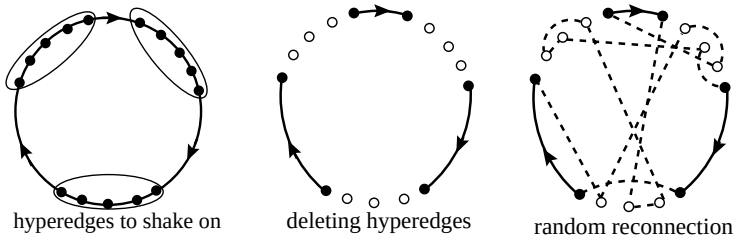

Fig. 4. Schema of a hyper-shake

and low computing time. The implementation of this approach is as described earlier except that we are also using nearest neighbour lists in order to keep computing times low. This means that for HyperOpt as well as 3-opt, we are not considering all possible moves but only a subset of nearest neighbours. This subset consists of the 20 nearest neighbours to and from each city. As a stopping criterion for VNS we limit the number of shakes performed by VNS to the number of cities of the problem.

3 Computational Results

In this section we apply our local and Variable Neighbourhood Search methods to all 27 ATSP instances in Reinelts Travelling Salesman Problem library `TSPlib` [20]. The solution quality of the heuristic approaches is expressed in percent excess over optimum. Seven of the instances `ftv90`, `ftv100` up to `ftv160` represent submatrices of problem `ftv170`. The largest instances provided by `TSPlib`, `rbg323` up to `rbg458`, represent stacker crane problems [14] . However, the performance of the heuristics for the 27 instances should provide us with a good indication of their ability to find good solutions for other ATSPs. To construct initial solutions from which local search is started, we have chosen the *Nearest neighbour* heuristic [21]. This lead on the average to solutions of better quality for all the heuristic approaches considered here than the *Multiple Fragment* (*Greedy*) algorithm [1].

All the heuristic calculations we refer to below have been carried out on a Pentium II 400 MHz PC with 128 Mb memory running a Linux 2.2.13 kernel. The algorithms were coded in C++ using the GNU `g++` compiler with `-O2` compiler option. The computational results given in Table 1 show the average excess over optimum and average CPU-time over 10 runs for each problem, using different Nearest Neighbour starting tours and different seeds for the randomized VNS heuristics.

The first column of Table 1 lists the names of the TSPlib instances considered, where the number indicates the problem size except for problem class `ftv` for which this number is equal to $\#cities - 1$. Because of space limitations we used the following abbreviations. `HO` is used to abbreviate HyperOpt where the number preceding `HO` indicates the parameter h used. The 3-opt/HyperOpt hybrid

approaches are christened HY where the preceding number again indicates parameter h used in HyperOpt. As above VNS abbreviates Variable Neighbourhood Search using the random shaking strategy proposed by Hansen and Mladenović [8][9], while GVNS indicates the use of our VNS strategy which uses guided shakes. Both VNS strategies considered here use HY3 for local search.

Out of the five local search approaches 3opt, 3HO, 4HO, HY3 and HY4 we can see that the hybrid approaches HY yield significantly better results for nearly all problems, significantly outperforming its constituent heuristics 3-opt and HyperOpt (HO). The fastest hybrid approach HY3 outperforms 3-opt on the average by about 0.7% while the much slower hybrid HY4 can improve on this by another percent gaining an average excess rate over all instances of about 5%.

Table 1. Computational Results

	Excess in % over optimum							CPU-times in sec.						
TSP	3opt	3HO	4HO	HY3	HY4	VNS	GVNS	3opt	3HO	4HO	HY3	HY4	VNS	GVNS
br17	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.01	0.01	0.05	0.01	0.16	1.31	1.25
ft53	7.13	12.31	7.16	5.57	4.37	0.33	0.12	0.09	0.14	1.05	0.28	1.16	10.6	10.6
ft70	3.06	3.74	2.51	2.32	1.81	0.23	0.13	0.22	0.23	1.62	0.50	2.00	20.3	20.8
ftv33	5.81	9.24	5.33	4.76	4.90	0.12	0.29	0.02	0.05	0.29	0.07	0.33	3.21	3.23
ftv35	2.89	4.96	3.68	3.99	3.04	0.33	0.09	0.02	0.05	0.33	0.08	0.36	3.20	3.36
ftv38	3.20	5.73	4.95	5.04	4.36	0.66	0.37	0.03	0.06	0.47	0.08	0.50	3.57	3.45
ftv44	6.29	6.83	5.40	4.86	3.67	1.02	0.60	0.05	0.09	0.51	0.14	0.58	4.96	4.92
ftv47	5.02	10.48	6.70	4.87	3.23	0.20	0.10	0.06	0.11	0.72	0.16	0.91	5.52	5.58
ftv55	7.08	10.14	7.13	6.35	6.16	0.24	0.08	0.10	0.13	0.95	0.23	0.94	7.05	6.58
ftv64	7.10	10.99	7.48	5.09	4.18	0.54	0.44	0.16	0.18	1.14	0.32	1.33	10.5	10.6
ftv70	8.15	11.33	8.28	8.17	6.47	0.81	0.82	0.19	0.23	1.50	0.48	1.75	12.5	12.7
ftv90	14.02	21.33	16.85	15.34	10.84	0.30	0.32	0.37	0.34	2.39	0.66	2.93	19.8	18.9
ftv100	10.91	10.50	11.42	8.78	9.09	0.46	0.41	0.50	0.49	2.96	1.02	3.45	23.9	22.3
ftv110	12.72	16.76	13.19	10.09	9.67	1.36	1.48	0.65	0.58	3.12	1.18	3.75	28.0	28.4
ftv120	11.78	16.52	11.02	10.93	8.01	1.13	0.63	0.84	0.53	3.96	1.44	4.77	32.5	31.6
ftv130	10.19	19.95	13.00	12.11	8.91	1.66	1.09	1.13	0.85	5.03	1.74	6.42	36.1	34.1
ftv140	12.20	14.80	12.97	11.31	9.84	1.45	0.98	1.41	0.91	6.26	2.31	7.71	40.7	40.1
ftv150	9.52	15.44	13.60	10.75	9.49	2.03	1.23	1.73	1.01	7.10	2.86	9.05	45.5	45.9
ftv160	14.65	17.77	12.50	11.71	9.68	3.02	2.41	2.25	1.38	8.48	3.88	10.1	50.7	49.9
ftv170	11.64	19.73	17.01	11.85	9.45	2.07	1.83	2.71	1.48	8.82	4.50	11.8	55.5	54.4
kro100	7.07	8.61	7.72	5.16	3.71	0.81	0.78	0.61	0.56	3.37	1.15	4.32	24.6	24.0
pr43	0.29	0.34	0.47	0.19	0.19	0.01	0.01	0.04	0.08	0.50	0.12	0.56	4.22	4.25
rbg323	0.33	1.75	0.82	0.42	0.39	0.09	0.08	23.0	5.87	32.3	27.0	49.6	280.1	282.2
rbg358	1.00	2.14	1.76	0.82	0.98	0.04	0.08	32.3	5.54	37.7	36.0	62.0	306.5	300.1
rbg403	0.08	0.20	0.17	0.07	0.09	0.00	0.00	47.3	7.34	50.4	49.0	82.7	288.6	284.6
rbg443	0.09	0.11	0.04	0.04	0.05	0.00	0.00	63.1	11.1	53.5	63.5	90.6	347.0	338.6
ry48p	4.27	3.01	2.99	2.65	2.77	0.28	0.28	0.06	0.10	0.69	0.16	0.78	5.29	4.97
av.	6.78	9.43	7.19	6.04	5.01	0.71	0.54	–	–	–	–	–	–	–

We can see that both VNS approaches, the random shaking strategy (VNS) suggested by Hansen and Mladenović [8][9] and our guided VNS approach, are

capable of improving significantly on the results gained by pure local search, gaining average excess rates of well below 1%. For nearly all instances our guided VNS (`GVNS`) produced tours of same or better quality than the original, random VNS strategy (`VNS`). The guided restart mechanism of `GVNS` enabled the heuristic to create tours with an average excess rate of about 0.5% which represents a major improvement over the excess rates gained by all of the other heuristic approaches. In particular for the "hard" instances (`ftv170` and its subproblems) which proved difficult for the local search guided VNS was able to produce tours of good quality.

4 Conclusions

In this paper we presented heuristic approaches which embed an exact algorithm within the framework of a local search heuristic for the asymmetric Travelling Salesman Problem. This approach, HyperOpt, was inspired by a problem arising in electronics manufacture and is a development of heuristics that have previously been applied successfully to electronics manufacture as well as symmetric TSPs [3][4]. We further proposed a hybrid of HyperOpt with the well known 3-opt local search technique in order to benefit from the advantages of both approaches, yielding a fast, effective approach which is significantly greater than the sum of its parts.

The computational results are encouraging. The HyperOpt/3-opt hybrid approaches proved able to find tours of better or same quality than 3-opt for nearly all of the 27 test problems considered in this paper. We further used the fastest HyperOpt/3-opt hybrid as a local search heuristic in a Variable Neighbourhood Search framework, as suggested by Hansen and Mladenović [8][9], and show how we may use a "guided shakes" strategy to improve on the random shakes in [8][9]. This metaheuristic approach yields very good results with an average excess rate over optimum of about 0.5% for the test problems considered here. We think that this approach represents a powerful method to determine near optimal solutions for asymmetric TSPs. All our approaches have shown themselves robust enough to handle a wide range of constraints such as occur in industrial problems.

References

1. Bentley, J.L.: Fast algorithms for geometric traveling salesman problems. ORSA Journal on Computing, **4** (1992) 387–411
2. Burke, E.K., Cowling, P.I., Keuthen, R.: New models and heuristics for component placement in printed circuit board assembly. In: Proceedings of the 1999 International Conference on Information, Intelligence and Systems (ICIIS99), Bethesda, MD, USA. IEEE Computer Society Press, (1999) 133–140
3. Burke, E.K., Cowling, P.I., Keuthen, R.: Effective heuristic and metaheuristic approaches to optimize component placement in printed circuit board assembly. In: Proceedings of the Congress on Evolutionary Computation CEC2000, San Diego, CA, USA. IEEE Computer Society Press, (2000) 301–308

4. Burke, E.K., Cowling, P.I., Keuthen, R.: Embedded local search and variable neighborhood search heuristics applied to the travelling salesman problem. University of Nottingham, Technical Report (2000)
5. Cowling, P.I.: Optimization in steel hot rolling. In: Optimization in Industry. John Wiley & Sons, Chichester, England, (1995) 55–66
6. A. Frieze A., Galbiati, G., Maffioli, F: On the worst-case performance of some algorithms for the asymmetric traveling salesman problem. Networks **12** (1982) 23–39
7. Glover, F., Gutin, G., Yeo, A., Zverovich, A.: Construction heuristics for the asymmetric TSP. European Journal of Operational Research **129** (2000) 555-568
8. Hansen, P., Mladenović, N.: An introduction to variable neighborhood search. In: S. Voss, S. Martello, I.H. Osman and C. Roucairol (eds.): Meta-Heuristics: Advances and Trends in Local Search Paradigms for Optimization, pages 433-458, Kluwer Academic Publishers, Boston, MA (1999)
9. Hansen, P., Mladenović, N.: Variable neighborhood search: Principles and applications. In: Invited papers at Euro XVI. Brussels, Belgium, (1998)
10. Johnson, D.S., McGeoch, L.A.: The travelling salesman problem: A case study. In: Aarts, E.H.L. and Lenstra, J.K. (eds.): Local Search in Combinatorial Optimization. John Wiley & Sons, New York (1997) 215–310
11. Kanellakis, P.C., Papadimitriou, C.H.: Local search for the asymmetric traveling salesman problem. Oper. Res. **28** (1980) 1086–1099
12. Karp, R.M.: A patching algorithm for the nonsymmetric traveling salesman problem. SIAM Journal on Computing **8** (1979) 561–573
13. Kolohan, F., Liang, M.: A tabu search approach to optimization of drilling operations. Comp. in Eng. **31** (1996) 371–374
14. Lawler, E.J., Lenstra, J.K., Rinnoy Kan, A.H.G., Shmoys, D.B.: The travelling salesman problem: A guided tour of combinatorial optimization. Wiley, New York (1985)
15. Lin, S., Kernighan, B.W.: An effective heuristic algorithm for the travelling salesman problem. Operat. Res. **21** (1973) 498–516
16. Martello, S., Toth, P.: Linear assignment problems. Annals of Discrete Mathematics **31** (1987) 259–282
17. Merz, P., Freisleben, B.: Genetic local search for the TSP: New results. In: Proceedings of the 1997 IEEE International Conference on Evolutionary Computation (1997) 159–164
18. Papadimitriou, C.H., Steiglitz, K.: The complexity of local search for the traveling salesman problem. SIAM Journal on Computing **6** (1977) 76–83
19. Pekney, J.F., Miller, D.L.: Exact solution of the no-wait flowshop scheduling problem with a comparison to heuristic methods. Computers & Chemical Engineering **15** (1991) 741–748
20. Reinelt, G.: TSPLIB - A travelling salesman problem library. ORSA-Journal of the Computing **3** (1991) 376–384
21. Reinelt, G.: The travelling salesman: Computational solutions for TSP applications. Lecture Notes in Computer Science, Springer-Verlag, Berlin Heidelberg New York (1994)
22. Walas, R.A., Askin, R.G.: An algorithm for NC turret punch press tool location and hit sequencing. IIE Transactions **16** (1984) 280–287
23. Zhang, W.: Depth-first branch-and-bound versus local search: A case study. In: Proceedings of the 17th National Conference on Artificial Intelligence (AAAI 2000), Austin, TX, USA (2000) 260–266