A Heuristic for the Asymmetric Traveling Salesman Problem (extended abstract)

Janez Brest* Janez Žerovnik† ‡

*Faculty of Electrical Engineering and Computer Science, University of Maribor
Smetanova 17, 2000 Maribor, Slovenia
janez.brest@uni-mb.si

†Faculty of Mechanical Engineering, University of Maribor
Smetanova 17, 2000 Maribor, Slovenia
janez.zerovnik@uni-mb.si

‡Institute of Mathematics, Physics and Mechanics
Jadranska 19, 1111, Ljubljana, Slovenia
janez.zerovnik@imfm.uni-lj.si

1 Introduction

The traveling–salesman problem (TSP) is one of the most studied problems in combinatorial optimization [16, 14]. The TSP is simply stated, has practical applications, and is a representative of a large class of important scientific and engineering problems. The TSP can be viewed as a graph–theory problem if the cities are identified with the vertices of a graph, and the links between the cities are associated with arcs. A weight corresponding to the inter–city distance is assigned to each arc. The TSP is equivalent to finding a minimal weighted Hamiltonian circuit in the complete graph K_n. However, in its usual physical interpretation, where the vertices of a graph are cities and edges represent roads interconnecting them, the graph is most likely not complete. To remedy this situation, graph is usually completed by adding arcs with the cost of the shortest path in the original graph.

TSP is an example of a NP-hard problem [10]. It is therefore reasonable to design heuristic algorithms which find near–optimal solutions. According to [1], several hundreds of papers were published on TSP and probably every approach for attacking NP-hard optimization problems has also been tested or has even been formulated for TSP.

An instance of the TSP is given by distance matrix $D = (d_{ij})$ of dimension $n \times n$; where d_{ij} represents the weight of the arc from city i to city j in $N = \{1, ..., n\}$. If $d_{ij} = d_{ji}$ for every pair i and j in N then the TSP is *symmetric*, otherwise it is *asymmetric* (ATSP). Our heuristics can be used for general TSP, but we restict our focus on ATSP. There are currently three classes of modern heuristics for the asymmetric traveling salesman problem proposed by

J. Cirasella et al. [7] classical tour construction heuristics such a Nearest Neighbor and Greedy algorithm, local search algorithms based on re-arranging segments of the tour, as exemplified by the Kanellakis-Papadimitriou algorithm, and algorithms based on patching together cycles in a minimum cycle cover, the best of which are variants on an algorithm proposed by Zhang. Our heuristics is based on the well-known arbitrary insertion procedure [17]. This algorithm was not payed too much attention, maybe because of the known worst case performance [17, 9]. In the worst case, the arbitrary insertion on ATSP can give solutions with costs as much as n times the optimal [9], while for the symmetric case the solution is always better than two times the optimum [17]. However, because of our relatively good experience with insertion–and–optimization approach on PTSP [18], a probabilistic generalization of TSP [11], we started our experiment, in which we have repeatedly run the arbitrary insertion based procedure followed by a local optimization phase. Interestingly, we got surprisingly good results within short computation times. While the arbitrary insertion procedure is well known although not much used, the local search based on the arbitrary insertion neighborhoods seems to be even less popular as we are not aware of any work using such neighborhood structure.

Recently, very good results were obtained with an exact algorithm for ATSP on a class of random instances [15]. Studies of the asymmetric traveling salesman polytope give hope to solve large instances of ATSP in general (see, for example [6] or [8]). The exact algorithms tend to be very time consuming, because their time complexity is superpolynomial. An alternative, perhaps more practical approach, is to design approximation algorithms which give solutions of reasonable quality in a short time. There is a simple $O(\log n)$ approximation algorithm for ATSP with triangle inequality [9]. Interestingly, the $1 \cdot \log_2 n$ approximation of [9] has been slightly improved to $0.999 \cdot \log_2 n$ [2] and to $0.842 \cdot \log_2 n$ [12] only after 20 years indicating hardness of the problem. The existence of a constant factor algorithm is open [13]. On the other hand, the well–known 3/2 approximation algorithm of Christofides was recently generalized to obtain approximation algorithms for TSP with β–triangle inequalities [3].

In this note we first give the results of tests of our algorithm on all ATSP instances of the TSPLIB library [16], which were available at the time of the experiment. The results we obtain are consistent with our experience with insertion–and–optimization approach on PTSP [18], a probabilistic generalization of TSP [11]. While the quality of results is comparable with our earlier tests [5], the running times are now much shorter. Bearing in mind high inherent paralelism of the heuristics, we claim that the approach we use may indeed be useful. Finally, we give upper bounds for the approximation ratio of a variant of our heuristics which depends on the "asymmetry" of the instance. Recently, aproximability of TSP with triangle inequality was extended to aproximability of TSP with β–triagle inequalitiy [3, 4]. It may be possible to use the ideas briefly explained here to extend the approximabily results from the symmetric TSP to asymmetric case where the asymmetry factor would play a role similar to the that of β.

2 The Heuristic

The main idea of our heuristic is based on the *arbitrary insertion algorithm* [17], a relaxation of the cheapest insertion algorithm. The solutions are further improved by a local optimization phase.

Algorithm RAI (Randomized Arbitrary Insertion):

1. *Start with a tour consisting of a given vertex and self–loop.*
2. *Randomly choose a vertex not on the tour.*
3. *Insert this vertex between neighboring vertices on the tour in the cheapest possible way. If the tour is still incomplete, go to step 2.*
4. *Keep this tour solution, say S.*
5. *Repeat n^2-times steps 6 through 10.*
6. *Randomly choose i and j ($i, j \in N = \{1, \ldots, n\}$, $1 \leq i \leq j \leq n$).*
7. *From the circuit with all vertices remove a path beginning with vertex i through vertex j, and connect vertex $i-1$ with vertex $j+1$.*
8. *Randomly choose a vertex from the removed path.*
9. *Insert this vertex between two neighboring vertices on the tour in the cheapest possible way. If the tour is still incomplete go to step 8.*
10. *Compare current solution with the solution S. Keep the better one.*

First four steps generate an initial circuit. In the main loop – steps 6 through 10 an optimization is performed. Some vertices are removed from the circuit and later they are randomly reinserted into the circuit once more in the cheapest possible way.

There is an interesting question on how many times the optimization should be performed (step 5). We repeated it n^2-times. We have no theoretical arguments for this choice; it turned out to give good results in reasonably short time. In each iteration, the number of deleted and reinserted vertices is at most n and for each insertion of a vertex at most n different insertion positions are compared. The worst case time complexity of our algorithm is thus $O(n^4)$.

3 Computational Results

Since we needed optimal solution value for evaluation of the results, we tested the algorithm on all ATSP instances from TSPLIB library [16]. The results obtained in our experiments are shown in Table 1. The second and the third column indicate the name of ATSP instance and the number of cities, respectively. The optimal tours are known for all of problem instances and their costs are shown in the fourth column. For each of the problem instances, 100 independent runs were performed.

Recall that in each run n^2 tours are generated. In the next column the average time (in seconds) of 100 independent runs is shown (SunFire v40z with FC3 Linux). In the last two columns the shortest and the average solution length for the 100 runs are shown, respectively. For all but four instances (problems: *ft70*, *kroa124p*, *rbg323*, *rbg358*) the optimal solutions have been found. If we look at the averages of the 100 independent runs we can see that our heuristic has solved the problems within 3% from optimum on average. If we look at the best solutions, we can see that our heuristic has solved majority of problems within 0.5% from optimum. there was only one exception: the best solution obtained for the problem instance *rbg323* was 0.67% from optimum.

In the last three columns, the number of runs in which our algorithm found the solution equal to the optimal solution, are presented for $n^2, 2n^2, 10n^2$ repetitions of step 5 (algorithm RAI), respectively.

Table 1: The results for the asymmetric TSP instances.

	Graph	n	Opt.	$t[s]$	min.	%	avg.	%	(n^2)	$(2n^2)$	$(10n^2)$
1	br17	17	39	0.001	39	0.00	39.00	0.00	100	100	100
2	ftv33	34	1286	0.004	1286	0.00	1288.16	0.17	96	99	100
3	ftv35	36	1473	0.005	1473	0.00	1481.06	0.55	26	29	30
4	ftv38	39	1530	0.008	1530	0.00	1541.27	0.74	18	15	17
5	p43	43	5620	0.009	5620	0.00	5620.71	0.01	21	32	49
6	ftv44	45	1613	0.011	1613	0.00	1637.51	1.52	22	24	37
7	ftv47	48	1776	0.014	1776	0.00	1780.18	0.30	17	34	90
8	ry48p	48	14422	0.014	14422	0.00	14517.20	0.66	6	5	16
9	ft53	53	6905	0.020	6905	0.00	6941.17	0.52	44	50	79
10	ftv55	56	1608	0.025	1608	0.00	1618.92	0.68	30	46	91
11	ftv64	65	1839	0.044	1839	0.00	1852.51	0.75	16	23	50
12	ft70	70	38673	0.058	38850	0.47	39170.30	1.30	0	0	0
13	ftv70	71	1950	0.061	1950	0.00	1965.79	0.81	7	32	81
14	ftv90	91	1579	0.159	1579	0.00	1583.31	0.27	8	15	22
15	kro124p	100	36230	0.238	36241	0.03	37237.20	2.78	0	0	0
16	ftv100	101	1788	0.237	1788	0.00	1791.95	0.22	15	9	23
17	ftv110	111	1958	0.343	1958	0.00	1963.59	0.29	9	12	21
18	ftv120	121	2166	0.481	2166	0.00	2179.07	0.60	6	10	16
19	ftv130	131	2307	0.656	2307	0.00	2321.84	0.64	5	6	10
20	ftv140	141	2420	0.885	2420	0.00	2434.40	0.60	6	5	11
21	ftv150	151	2611	1.161	2611	0.00	2645.88	1.34	10	13	19
22	ftv160	161	2683	1.521	2683	0.00	2717.77	1.30	18	19	38
23	ftv170	171	2755	1.957	2755	0.00	2801.16	1.68	4	2	8
24	rbg323	323	1326	29.28	1335	0.68	1352.01	1.96	0	0	0
25	rbg358	358	1163	45.29	1164	0.09	1174.84	0.93	0	0	0
26	rbg403	403	2465	76.36	2465	0.00	2465.74	0.03	80	87	92
27	rbg443	443	2720	115.9	2720	0.00	2720.64	0.02	70	76	88

Remark. In [5] we have also compared our heuristic with Farthest Insertion (far), and Farthest Insertion followed by OR-opt [14] (p. 220). OR-opt local optimization proceeds as follows. For each connected string of s cities (s equals 3 first, then 2, then 1), we test to see if the string can be relocated between two other cities at reduced cost. If it can, we make the appropriate changes. After considering all strings of three cities, all strings of two cities and then all strings of one city are considered. When no further exchanges improve the solution, the algorithm terminates. Farthest insertion was repeated n times, each time another vertex was used as initial tour. The running times for farthest insertion are much shorter, therefore we only give the shortest solution length. The tours constructed by farthest insertion were then improved by OR–opt. The solutions after local optimization are given in column far+OR. The execution times were measured and then the algorithm RAI was let running for the same amount of time (column $t[s]$). The solutions obtained by RAI are given in column RAI. Note that the solutions obtained by RAI algorithm were usually better with only three exceptions,

the instances *ry48p*, *rbg323*, and *rbg358*.

We conclude that the fast and simple heuristics RAI performs remarkably well and is competitive both in terms of solution quality and execution times with the best heuristics proposed in the literature.

4 Approximation bounds

We say, for every $\beta \geq 1/2$, that an input instance of the general TSP satisfies the β–triangle inequality if

$$d_{ij} \leq \beta(d_{ik} + d_{kj})$$

for all vertices i, j, k. Note that this definition is valid for general TSP including ATSP. By Δ_β–TSP we denote the TSP whose input instances satisfy the β–triangle inequality. Recently, the well–known 3/2 approximation algorithm of Christofides for symmetric TSP was generalized to obtain approximation algorithms for symmetric TSP with β–triangle inequalities. For $1/2 \leq \beta \leq 1$, there is a $(1 + \frac{2\beta-1}{3\beta^2-2\beta+1})$ approximation algorithm [3]. For $1 \leq \beta$, there is a $\frac{3}{2}\beta^2$ approximation algorithm [4]. For more details and further references on related work, see [4].

Let us define the asymmetry factor

$$\alpha = \max_{i,j \in V(G)} \left\{ \frac{d_{ij}}{d_{ji}}, \frac{d_{ji}}{d_{ij}} \right\}.$$

If weights (distances) equal to 0 appear in the TSP instance, then we take $0/0 = 1$ and $d_{ij}/0 = \infty$. Hence ATSP with asymmetry factor ∞ are possible. $\alpha = 1$ ATSP is just a TSP.

Theorem 1. *Let A be an approximation algorithm for Δ_β–TSP with approximation ratio r. Then there is an approximation algorithm for Δ_β–ATSP with approximation ratio $\frac{1+\alpha}{2}r$ where α is the asymmetry factor.*

The proof will be given in the full paper.

Let us consider the following variant of the algorithm RAI in which we replace the initial tour construction (Steps 1-5) and leave the iterative improvement part (Steps 6-10) unchanged.

Algorithm ARAI (Approximation Algorithm based on Randomized Arbitrary Insertion):

1. *Generate a symmetric instance by taking $\tilde{d}_{ij} = \min\{d_{ij}, d_{ji}\}$.*
2. *Run an approximation algorithm for Δ_β–TSP.*
3. *Choose the cheapest of the two oriented tours.*
4. *Keep this tour solution, say S.*
5. *Repeat n^2-times steps 6 through 10.*
6. *Steps 6-10 are as in algorithm RAI.*

Theorem 2. *ARAI is an approximation algorithm on instances with finite asymmetry factor.*

The idea of the proof is as follows: Given an instance with finite asymmetry factor, compute a suitable β and use the approximation algorithm for Δ_β–TSP with approximation ratio, say r. Then by Theorem 1, already initial tour produced by ARAI is a $\frac{1+\alpha}{2}r$ approximation.

References

[1] M. Dell'Amico, F. Maffioli and S. Martello (1997): *Annotated Bibliography in Combinatorial Optimization*, John Wiley & Sons, New York.

[2] M. Bläser (2003): "A new approximation algorithm for the asymmetric TSP with triangle inequality". Presented at *Proceedings of the fourteenth annual ACM-SIAM symposium on discrete algorithms*, 2003, 638-645.

[3] H. Böckenhauer, J. Hromkovič, R.Klasing, S. Seibert and W.Unger (2000): "Approximation algorithms for the TSP with sharpened triangle inequality", In: *Information Processing Letters* **75**, 133–138.

[4] H. Böckenhauer, J. Hromkovič, R.Klasing, S. Seibert and W.Unger (2002): "Towards a notion of stability of approximation for hard optimization tasks and the traveling salesman problem". In: *Theoretical Computer Science* **285**, 3–24.

[5] J. Brest and J. Žerovnik (1999): "An approximation algorithm for the asymmetric traveling salesman problem". In: *Ric. oper.* **28**, 59–67.

[6] S. Chopra and G. Rinaldi (1996): "The Graphical Asymmetric Traveling Salesman Polyhedron: Symmetric Inequalities". In: *SIAM J. Discrete Math.* **9**, 602–624.

[7] J. Cirasella, D. S. Johnson, L. A. McGeoch and W. Zhang (2001): "The asymmetric traveling salesman problem: Algorithms, instance generators, and tests". In: *Lecture Notes in Computer Science* **2153**, 32–59.

[8] M. Fischetti and P. Toth (1997): "A Polyhedral Approach to the ATSP". In: *Management Science* **43**, 1520–1536.

[9] A. M. Frieze, G. Galbiati and F. Maffioli (1982): "On the Worst-Case Performance of some Algorithms for the Asymmetric Traveling Salesman Problem". In: *Networks* **12**, 23–39.

[10] M. R. Garey and D. S. Johnson (1979): *Computers and Intractability*. W.H. Freeman and Co., San Francisco.

[11] P. Jaillet (1988): "A Priori Solution of a Traveling Salesman Problem in which a Random Subset of the Customers are Visited". In: *Operation Research* **36**, 929–936.

[12] H. Kaplan, M. Lewenstein, N. Shafrir and M. Sviridenko (2003): "Approximation Algorithms for Asymmetric TSP by Decomposing Directed Regular Multigraphs". Presented at *44th Annual IEEE Symposium on Foundations of Computer Science (FOCS'03)*, p.56.

[13] Samir Khuller (1998): "Problems". In: *Journal of Algorithms* **28** (1), 192–195.

[14] E. L. Lawler, J. K. Lenstra, A. H. G. Rinnooy Kan and D. B. Shmoys (1985): *The Traveling Salesman Problem*. John Wiley & Sons, New York.

[15] D. L. Miller and J. F. Pekny (1991): "Exact Solution of Large Asymmetric Traveling Salesman Problems". In: *Science* **251**, 754–761.

[16] G. Reinelt (1991): "TSPLIB - a Traveling Salesman Problem Library". In: *European Journal of Operations Research* **52**, p. 125.

[17] D. J. Rosenkrantz, R. R. Stearns and P. M. Lewis (1977): "An Analysis of Several Heuristics for the Traveling Salesman Problem", In: *SIAM J. Comput.* **6**, 563–581.

[18] J. Žerovnik (1995): "A Heuristics for the Probabilistic Traveling Salesman Problem". Presented at *Proceedings of the International Symposium on Operational research 1995 (SOR'95)*, (V.Rupnik, M.Bogataj, eds.), Slovenian Society Informatika, Ljubljana 1995, 165–172.