International Journal of Mechanical, Industrial and Aerospace Sciences
ISSN: 2517-9950
Vol:1, No:1, 2007

A New Heuristic Algorithm for the Classical Symmetric Traveling Salesman Problem

S. B. Liu, K. M. Ng, and H. L. Ong

***Abstract*—**This paper presents a new heuristic algorithm for the classical symmetric traveling salesman problem (TSP). The idea of the algorithm is to cut a TSP tour into overlapped blocks and then each block is improved separately. It is conjectured that the chance of improving a good solution by moving a node to a position far away from its original one is small. By doing intensive search in each block, it is possible to further improve a TSP tour that cannot be improved by other local search methods. To test the performance of the proposed algorithm, computational experiments are carried out based on benchmark problem instances. The computational results show that algorithm proposed in this paper is efficient for solving the TSPs.

***Keywords*—**Local search, overlapped neighborhood, traveling salesman problem.

I. Introduction

THE document is a template for *Word (doc)* versions. traveling salesman problem is one of the classical challenging combinatorial optimization problems. The objective of the TSP is to minimize the total distance traveled by visiting all the nodes once and only once and then returning to the depot node. The classical formulation of the TSP is stated as follows. Let a network $G = (N, A, C)$ be defined with N denoting the set of nodes on the network, A denoting the set of arcs and $C = [c_{ij}]$ denoting the matrix of costs.

A common application of the TSP is the movement of people, equipment and vehicles around tours of duty to minimize the total traveling cost. For example, in a school bus routing problem, it is required to schedule a school bus to pick up waiting students from the pre-specified locations. Post routing is another application of the TSP. The postman problem is modeled as traversing a given set of streets in a city, rather than visiting a set of specified locations. Moreover, the TSP plays an important role in general post problem, where the houses or streets are far away from each other. Besides the above mentioned applications, some other seemly unrelated problems are solved by formulating them as the TSP. The genome sequencing problem occurs in the field of bio-engineering. The aim of this problem is to find the genome sequence based on the markers that serve as landmarks for the genome maps. The TSP plays an important role in genome sequencing by providing a tool for building sequences from experimental data on the proximity of individual pairs of markers. By considering markers as cities, a genome sequence can be viewed as a TSP path traveling through each marker once and only once. The drilling problem is another application of the TSP with the objective of minimizing the total travel time of the drill. In the electronic industry, printed circuit boards are widely found in common electronic devices. The printed circuit board normally has a very large number of holes used for mounting components or integrated chips. These holes are typically drilled by automated drilling machines that move between specified locations to drill a hole one after another. Therefore, the locations in the drilling problem correspond to the cities in the TSP. The applications of the TSP are not limited to the examples described above. A detailed review of the applications of the TSP can be found in [1-2].

It has been proved that TSP is NP-hard in [3] which implies that a polynomial bounded exact algorithm for TSP is unlikely to exist. In this paper, we presented a heuristic algorithm for solving the TSP and its performance is illustrated based on the benchmark TSP instances. The algorithm proposed in this paper is based on the work in [4], in which an existing solution is divided into overlapped blocks and then each block is explored separately. By doing local search using the Generalized Crossing (GC) method, which is developed in [5] for the vehicle routing problem (VRP), each block is explored intensively in order to improve the existing solution. This paper is organized as follows. In Section II, a literature review is given on the TSP exact and approximate algorithms. The details of the proposed algorithm are presented in Section III. The computational results are provided in Section IV and the concluding remarks and possible future work are given in Section V.

II. Literature Review

The TSP has been studied intensively during the last 50 years and many exact and heuristic algorithms have been

S. B. Liu is a PhD. candidate in the Department of Industrial & Systems Engineering, National University of Singapore, Kent Ridge Crescent, Singapore 119260 (corresponding author phone: (65) 6516-6514; fax: (65) 6777-1434 e-mail: g0403853@nus.edu.sg).

K. M. Ng is an assistant professor with the Department of Industrial & Systems Engineering, National University of Singapore, Kent Ridge Crescent, Singapore 119260 (e-mail: isenkm@nus.edu.sg).

H. L. Ong is an associate professor with the Department of Industrial & Systems Engineering, National University of Singapore, Kent Ridge Crescent, Singapore 119260 (e-mail: iseonghl@ nus.edu.sg).

developed. These algorithms include construction algorithms, iterative improvement algorithms, branch-and-bound and branch-and-cut exact algorithms, and many metaheuristic algorithms, such as simulated annealing (SA), tabu search (TS), ant colony (AC) and genetic algorithm (GA).

Some of the well known tour construction procedures are the nearest neighbor procedure by Rosenkrantz et al. [6], the Clarke and Wright savings' algorithm [7], the insertion procedures [6], the partitioning approach by Karp [8] and the minimal spanning tree approach by Christofides [9] etc.

The branch exchange is perhaps the best known iterative improvement algorithm for the TSP. The 2-opt and 3-opt heuristics were described in Lin [10]. Lin and Kernighan [11] made a great improvement in quality of tours that can be obtained by heuristic methods. Even today, their algorithm remains the key ingredient in the most successful approaches for finding high-quality tours and is widely used to generate initial solutions for other algorithms. Or [12] developed a simplified edge exchange procedure requiring only $O(n^2)$ operations at each step, but producing tour nearly as good as the average performance of 3-opt algorithm.

One of the earliest exact algorithms is due to Dantzig et al. [13], in which linear programming (LP) relaxation is used to solve the integer formulation by adding suitably chosen linear inequality to the list of constraints continuously. Branch and bound (B&B) algorithms are widely used to solve the TSPs. Several authors have proposed B&B algorithm based on assignment problem (AP) relaxation of the original TSP formulation. These authors include Eastman [14], Held and Karp [15], Smith et al. [16], Carpaneto and Toth [17], Balas and Crhistofides [18]. Some branch and cut (B&C) based exact algorithms were developed by Crowder and Padberg [19], Padberg and Hong [20], Grötschel and Holland [21].

Besides the above mentioned exact and heuristic algorithms, metaheuristic algorithms have been applied successfully to the TSP by a number of researchers. SA algorithms for the TSP were developed by Bonomi and Lutton [22], Golden and Skiscim [23] and Nahar et al. [24], Lo and Hus [25] etc. Tabu search metaheuristic algorithms for the TSP have been proposed by Knox [26], and Fiechter [27] etc. The AC is a relative new metaheuristic algorithm which is applied successfully to solve the TSP. Some work based on AC technology was reported by Dorigo et al. [28], Gomez and Banan [29], Bullnheimer et al. [30] and Tsai et al. [31] etc. Genetic algorithms for the TSP were reported by Grefenstette et al. [32], Whitley et al. [33], and Nguyen et al. [34] etc.

Comprehensive review of the techniques developed for the TSP can be found in [1-2, 35-37].

III. Proposed Heuristic Algorithm

It is conjectured that the chance of improving a good solution by moving a node to a position far away from its original position is small. Therefore, it is reasonable to define a small size neighborhood based on the vicinity of nodes. The heuristic algorithm proposed in this paper divides an existing TSP tour into blocks which are overlapped with the blocks next to them and each block is then explored separately. Here we use the GC method as the local search method within each block. The overlaps between any two blocks make it possible to further improve a block if the positions of nodes in its next block are changed.

The proposed heuristic algorithm is implemented with a backtrack strategy. Whenever a block reaches local optimum and is improved by the GC method, it will backtrack to the recently searched block for possible improvement. If a block cannot be improved by the GC method, the search procedure will move to the next block. This procedure is repeated until no improvement is possible. The computational requirements of the above strategy can be reduced as follows: whenever the positions of nodes in the overlap of two blocks are changed, the GC procedure will backtrack to the previous block which is overlapped with the current block; otherwise, it will move to the next block.

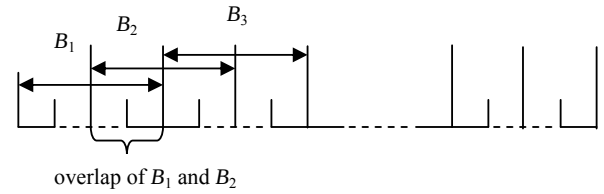

Fig. 1 Illustration of overlapped blocks

The overlapped blocks are illustrated in Fig. 1 above. As shown in this figure, a TSP tour is divided into overlapped blocks B_1, B_2, B_3 and so on. For example, after the local optimal solution in B_2 is obtained and the partial sequence in B_2 is improved, the GC local search will backtrack to B_1 and continue to do local search. If the partial solution in B_2 cannot be further improved, the GC local search will continue to search in B_3. This backtrack search procedure is repeated until no improvement is possible.

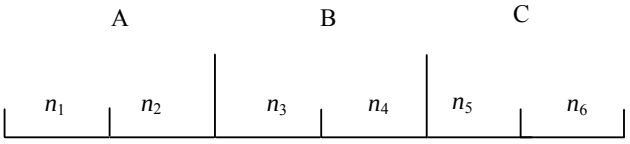

Fig. 2 Initial node sequence in a block

To present the details of the proposed method, we use S_B to denote the size of block, and S_O to denote the size of overlap. The size of block or the size of overlap is the number of nodes in the block or in the overlap respectively.

A block with $S_B = 6$ is illustrated in Fig. 2. This block is cut into three pieces, represented by A, B and C respectively. Five new partial sequences can be generated by sequencing the three pieces. As presented in the GC local search method developed in [5], any of the three pieces A, B and C can also be reversed to form new sequence. For each sequence in Fig. 3, 7 new sequences can be produced by reversing nodes in one

piece, two pieces or three pieces as illustrated in Fig. 4. Therefore, a total of 48 TSP tours (including the original tour) can be generated from a sequence. The implementation of the GC method is a first-improvement strategy, which means the first neighbor of the current sequence that is found to be better than the current sequence is made the current sequence.

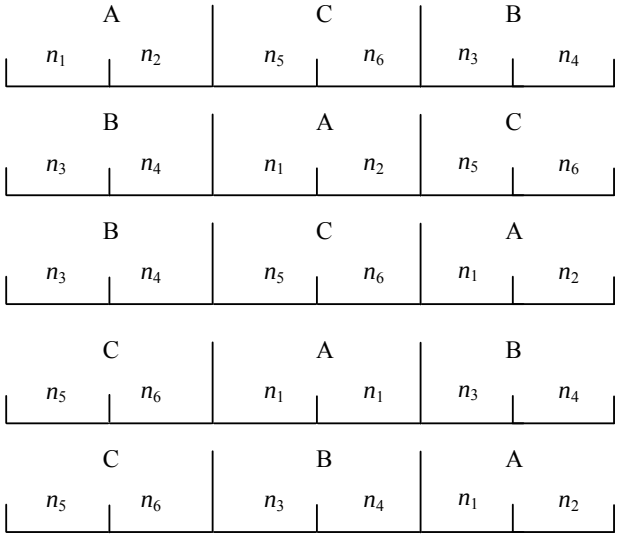

Fig. 3 Sequences generated by re-sequencing three pieces

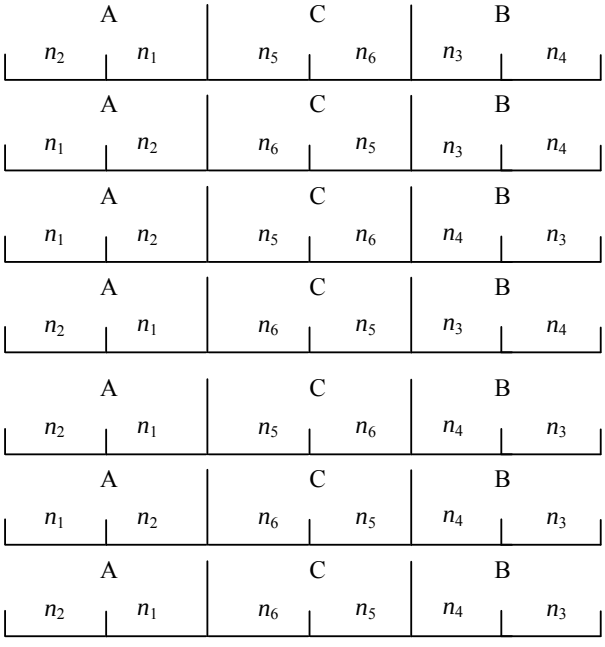

Fig. 4 New sequences generated by reversing nodes in each piece

The proposed heuristic algorithm starts to search from small size of block with $S_B = 3$. If GC does not make any improvement for a given number of trials with different S_B, the algorithm will terminate. For a given S_B, if S_O := Coefficient × S_B, the number of blocks divided from a TSP tour is about

$$\frac{n}{S_B - S_O} = \frac{n}{S_B \times (1 - Coefficient)},$$

where n is the size of the TSP and Coefficient is the ratio of S_O/S_B with $0 < \text{Coefficient} \leq 1$. As the GC local search method has a computational complexity of $O(S_B^2)$, the computational complexity for a given S_B is

$$O(\frac{n}{S_B \times (1 - Coefficient)} S_B^2) = O(n \times S_B).$$

Therefore, the computational complexity of proposed algorithm is $O(n^3)$ if the size of block ranges from 3 to n. However, in practice the proposed algorithm is much faster than $O(n^3)$ as it can terminate when the maximal number of trials without improvement is reached.

IV. Computational Experiments

To test the performance of the proposed algorithm, computational experiments are carried out based on 20 benchmark problems obtained from the TSP library (accessible via the web at http://www.iwr.uni-heidelberg.de/groups/comopt/software/TSPLIB95/). The same set of problems was also used in [38] and [39]. It is noted that in this study, we follow the rule used in TSPLIB to compute the travel distance. That is, all the distances are rounded to the nearest integer values. Therefore, it is possible that the results may not be consistent with the results reported by some researchers who compute the traveling distance in different ways. The proposed algorithm has been coded in C++ and all the experiments are implemented on a Pentium IV 2.6 GHz PC with 512MB RAM.

In this study, the performance of the proposed algorithm is compared with the co-adaptive neural network (CAN) approach proposed in [39]. Similar to the computational experiments conducted in [39], the proposed algorithm is implemented for 10 runs for each problem and the computational results are summarized in Table I. The solutions presented in bold characters show the smallest average deviation and the smallest deviation for each problem respectively. The following information is presented in Table I:

- the problem name
- the size of the problem
- the optimal length of tour Z_{opt}
- the percentage deviation from Z_{opt} of the average solution value over the 10 replications, represented by %PDM
- the percentage deviation from Z_{opt} of the best solution over the 10 replications, represented by %PDB
- the average computation time in seconds

TABLE I
COMPUTATIONAL RESULTS OF THE PROPOSED ALGORITHM

Problem	size	optimal	CAN			proposed algorithm		
			%PDM	%PDB	time[1]	%PDM	%PDB	time[2]
Eil51	51	426	2.89	0.94	0.27	**1.76**	**0.70**	0.87
Eil76	76	538	4.35	**2.04**	0.52	**2.96**	2.23	2.30
Eil101	101	629	3.78	**1.11**	1.22	**3.64**	2.54	4.28
berlin52	52	7542	7.01	**0.00**	0.32	**3.94**	**0.00**	0.95
bier127	127	118282	3.00	**0.69**	1.88	**2.50**	0.77	6.57
Ch130	130	6110	**2.82**	1.13	1.97	2.88	**0.70**	4.95
Ch150	150	6528	3.23	1.78	2.62	**3.20**	**1.53**	9.58
Rd100	100	7910	3.64	1.19	1.15	**2.83**	**0.00**	3.95
Lin105	105	14379	**1.08**	**0.00**	1.27	2.17	0.70	3.72
Lin318	318	42029	4.31	2.65	9.97	**4.14**	**2.43**	33.41
kroa100	100	21282	1.31	0.57	1.14	**0.86**	**0.00**	5.07
krob100	100	22141	**2.20**	1.53	1.15	2.94	**0.61**	3.68
kroc100	100	20749	1.70	0.80	1.11	**1.53**	**0.10**	4.18
krod100	100	21294	1.87	0.80	1.16	**1.73**	**0.07**	4.25
kroe100	100	22068	2.56	1.52	1.15	**2.32**	**0.00**	4.18
kroa150	150	26524	**3.06**	1.55	2.77	3.56	**0.82**	6.72
krob150	150	26130	**2.60**	1.06	2.63	**2.60**	**1.05**	7.12
kroa200	200	29368	3.27	0.92	4.74	**2.72**	**0.84**	14.75
krob200	200	29437	**2.31**	**0.88**	4.74	3.70	2.04	10.61
fl1400	1400	20127	4.26	**2.12**	82.85	**3.39**	2.32	1549.93
Average			3.06	1.16	6.23	2.77	0.97	84.05

1 – the average computation time for each replication based on Silicon Graphics O2 workstation
2 – the average computation time for each replication based on PC with 2.6GHz CPU and 512MB RAM

The computational results presented in Table I show that the proposed algorithm can get an average deviation of 2.77%, which is much smaller than the average deviation 3.06% obtained by CAN. Moreover, the proposed algorithm obtains 14 best solutions while CAN only obtains 7 best solutions among the 20 TSPs. When average performance is concerned, the proposed algorithm obtains the smallest average deviation for 15 TSPs while CAN only obtains 6 best average deviations.

As our computational experiments are conducted on different platforms from CAN, it is difficult to compare the computation time taken by CAN and the proposed algorithm. However, the average computation time taken by the proposed algorithm shown in Table I is reasonable even for large size problems up to 1400 cities. In general, our algorithm is superior to CAN regarding both the average deviation and the smallest deviation from the optimal solutions.

V. CONCLUSIONS AND REMARKS

This paper presents an overlapped neighborhood based local search algorithm to solve TSPs. By dividing a solution into small blocks with each block being explored by the GC local search method, the proposed heuristic algorithm is able to obtain better solutions for TSPs when compared with a co-adaptive neural network method proposed in the literature. We believe that the performance of our algorithm can be further improved by hybridizing with metaheuristic algorithms, such as tabu search and ant colony optimization, and so this is an area of further research. In addition, the idea of applying the overlapped neighborhood based local search method to the TSP as described in this paper can be extended to other similar or related combinational optimization problems as well, such as the vehicle routing problems and machine scheduling problems.

REFERENCES

[1] E. L. Lawler, J.K. Lenstra, A.H.G. Rinnooy Kan, and D.B. Shmoys, *The traveling salesman problem*. Ed. Chichester: John Wiley & Sons, 1985.
[2] D.L. Applegate, R.E. Bixby, V. Chvátal, and W.J. Cook, *The traveling salesman problem: A computational study*. Princeton: Princeton University Press, 2006.
[3] M.R. Garey, and D.S. Johnson, *Computers and intractability: A guide to the theory of NP-completeness*. San Francisco: W.H. Freeman, 1979.
[4] S.B. Liu, and K.M. Ng, and H.L. Ong, "An overlapped neighborhood search method for general sequencing problems," Working Paper, Department of Industrial & Systems Engineering, National University of Singapore, 2007.
[5] L. Zeng, H.L., Ong, and K.M. Ng, "A generalized crossing local search method for solving vehicle routing problems," *The Journal of the Operational Research Society*, vol. 58, pp. 528-532, 2007.
[6] G. Clarke, and J. Wright, "Scheduling of vehicles from a central depot to a number of delivery points". *Operations Research*, vol. 12, pp. 568-581, 1964.

[7] D. Rosenkrantz, R.E. Sterns, and P.M. Lewis, "An analysis of several heuristics for the traveling salesman problem," *SIAM Journal on Computing*, vol. 6, pp. 563-581, 1977.
[8] R. M. Karp, "Probabilistic analysis of partitioning algorithms for the traveling salesman problem in the plane," *Mathematics of Operations Research*, vol. 2, pp.209-224, 1977.
[9] N. Christofides, "Worst-case analysis of a new heuristic for the traveling salesman problem," Report 388, Graduate School of Industrial Administration, Carnegie Mellon University, February 1976.
[10] S. Lin, "Computer solutions of the traveling salesman problem," *Bell System Technical Journal*, vol.44, pp. 2245-2269, 1965.
[11] S. Lin, and B.W. Kernighan, "An effective heuristic algorithm for the traveling-salesman problem," *Operations Research*, vol. 21, pp. 498-516, 1973,
[12] I. Or, "Traveling salesman-type combinatorial problems and their relation to the logistics of regional blood banking," Ph.D. Thesis, Northwestern University, Evanston, IL, 1976.
[13] G.B. Dantzig, D.R. Fulkerson, S.M. and S.M. Johnson, "Solution of a large-scale traveling-salesman problem," *Operations Research*, vol. 2, pp. 393-410, 1954.
[14] W.L. Eastman, "Linear programming with pattern constraints," Ph.D. Thesis, Harvard University, Cambridge, MA, 1958.
[15] M. Held, and R.M. Karp, "The traveling salesman problem and minimum spanning trees: Part II," *Mathematical Programming*, vol. 1, pp. 6-25. 1971.
[16] T.H.C. Smith, V. Srinivasan, and G.L. Thompson, "Computational performance of three subtour elimination algorithms for solving asymmetric traveling salesman problems," *Annals of Discrete Mathematics*, vol. 1, pp. 495-506, 1977.
[17] G. Carpaneto, and P. Toth, "Some new branching and bounding criteria for the asymmetric travelling salesman problem," *Management Science*, vol. 26, pp. 736-743, 1980.
[18] E. Balas, and N. Christofides, "A restricted lagrangean approach to the traveling salesman problem," *Mathematical Programming,* vol. 21, pp. 19-46, 1981.
[19] H. Crowder, and M.W. Padberg, "Solving large-scale symmetric travelling salesman problems to optimality," *Management Science*, vol. 26, pp. 495-509, 1980.
[20] M.W. Padberg, and S. Hong, "On the symmetric traveling salesman problem: A computational study," *Mathematical Programming Study*, vol. 12, pp. 78-107, 1980.
[21] M. Grötschel, and O. Holland, "Solution of large-scale symmetric travelling salesman problems," *Mathematical Programming*, vol. 51, pp. 141-202, 1991.
[22] E. Bonomi, and J.-L. Lutton, "The *N*-city travelling salesman problem: Statistical mechanics and the Metropolis algorithm," *SIAM Review*, vol. 26, pp. 551-568, 1984.
[23] B.L. Golden, and C.C. Skiscim, "Using simulated annealing to solve routing and location problems," *Naval Research Logistics Quarterly*, vol. 33, pp. 261-280, 1986.
[24] S. Nahar, S.Sahni, and E. Shragowitz, "Simulated annealing and combinatorial optimization," *International Journal of Computer Aided VLSI Design*, vol. 1, pp. 1-23, 1989.
[25] C.C. Lo, and C.C. Hus, "An Annealing framework with learning memory," *IEEE Transactions on Systems, Man and Cybernetics*, *Part A*, vol. 28, pp. 1-13, 1998.
[26] J. Knox, "An application of TABU search to the symmetric traveling salesman problem," Ph.D. Thesis, Center for Applied Artificial Intelligence (CAAI), Graduate School of Business, University of Colorado, 1988.
[27] Fiechter, C.-N. "A parallel tabu search algorithm for large scale traveling salesman problems," Working Paper 90/1, Département de Mathématiques, École Polytechnique. Fédérale de Lausanne, 1990.
[28] M. Dorigo, V. Maniezzo, and A. Colorni, "The ant system: optimization by a colony of cooperating agents," *IEEE Transactions on Systems, Man and Cycbernetics*, *Part B*, vol. 26, pp. 29-42, 1996.
[29] O. Gomez , and B. Banan, "Reasons of ACO's success in TSP," *Ant Colony Optimization And Swarm Intelligence, Proceedings Lecture Notes In Computer Science*, vol. 3172, pp. 226-237, 2004.
[30] B. Bullnheimer, R.F. Hartl, and C. Strauss, "An improved ant system algorithm for the vehicle routing problem," *Annals of Operation Research*, vol. 89, pp. 319-328, 1999.
[31] C.F. Tsai, C.W. Tsai, and C.C. Tseng, "A new and efficient ant-based heuristic method for solving the traveling salesman problem," *Expert Systems*, vol. 20, pp.179-186, 2003.
[32] J.J. Grefenstette, R. Gopal, B. Rosmaita, and D. VanGucht, "Genetic algorithms for the traveling salesman problem," in *Proceedings of the first International Conference on Genetic Algorithms*, pp. 160–168, 1985.
[33] D. Whitley, T. Starkweather, and D. Fuquay, "Scheduling problems and traveling salesman: The genetic edge recombination operator," in *Proceedings of the third international conference on Genetic algorithm*, pp. 133–140, 1989.
[34] H.D. Nguyen, I. Yoshihara, K. Yamamori, and M. Yasunaga, "Implementation of an effective hybrid GA for large-scale traveling salesman problems," *IEEE Transactions on Systems, Man, and Cybernetics—Part B*: *Cybernetics*, vol. 37, pp. 92-99, 2007.
[35] L.D. Bodin, and B.L. Golden, A.A. Assad, M. Ball, "Routing and scheduling of vehicles and crews, the state of art," *Computers and Operations Research*, vol. 10, pp. 63-212, 1983.
[36] G. Laporte, "The traveling salesman problem: An overview of exact and approximate algorithms," *European Journal of Operational Research*, vol. 59, pp. 231-247, 1992.
[37] D.S. Johnson, and L.A. Mcgeoch, "The traveling salesman problem: A case study," In: *Local Search in Combinatorial Optimization*. E. Aarts, J.K. Lenstra, Ed. New York: John Wiley & Sons, 1997, pp. 215-310.
[38] S. Somhom, A. Modares, and T. Enkawa, "A self-organising model for the traveling salesman problem," *The Journal of the Operational Research Society*, vol. 48, pp. 919-928, 1997.
[39] E.M. Cochrane, and J.E. Beasley, "The co-adaptive neural network approach to the Euclidean traveling salesman problem," *Neural Networks*, vol. 16, pp. 1499-1525, 2003.