Simulated Annealing with Probabilistic Neighborhood for Traveling Salesman Problems

Yang Li, Aimin Zhou, Guixu Zhang
Computer Science & Technology Department
East China Normal University
500 Dongchuan Road, Shanghai, China, 200241
Email: yangle.sun@gmail.com, {amzhou,gxzhang}@cs.ecnu.edu.cn

***Abstract*—The *traveling salesman problem (TSP)* is a classical combinatorial optimization problem. Due to its NP-hard nature, modern heuristic methods are usually applied to find an approximation of the best tour of a TSP. It has been proved that most of the edges of a best tour are linked by neighbor cities. By using this heuristic information, a probabilistic neighborhood model is defined in this paper to guide the tour constructing in a heuristic method. This probabilistic neighborhood model is then applied in the framework of a simulated annealing (SA) algorithm. The proposed algorithm is compared to a basic SA algorithm on a set of test instances. The preliminary results show that an SA with probabilistic neighborhood can outperform a generic SA.**

I. Introduction

Many real world applications involve finding an optimal permutation of some elements. For example in the case of business application, a salesman must find a shortest tour to visit a given set of cities once and only once and return back to the original city. These kind of problems, which are called *traveling salesman problems (TSPs)*, can be mathematically formulated as [1], [2]

$$\min f(\pi) = \sum_{i=1}^{n-1} d_{\pi_i,\pi_{i+1}} + d_{\pi,1} \tag{1}$$

where $\pi = (\pi_1, \pi_2 \cdots \pi_n)$ is a permutation of n integers $\{1, 2, \cdots, n\}$; $d_{i,j} > 0$ denotes the distance between the ith city and the jth city.

The major difficulty with a TSP is that as the number of cities increases, the number of possible permutations increases exponentially. It has been proved that TSPs are NP-hard problems and thus there does not exist an algorithm which can solve them in polynomial time [3]. Thus, many researchers resort to use a heuristic method to find an approximation of the best tour of a TSP. Although the importance of TSP for both scientific research and real world applications was recognized long time ago, the problem became popular in 1950s and a lot of works on heuristic methods have been done ever since then. The most widely utilized algorithms can be classified into the following categories.

- *Local search methods:* The greedy algorithm is a natural choice for constructing a TSP tour [4]. To construct an optimal tour, it starts from a city and then finds the next city from the neighborhood of the current city until a whole tour is constructed. The methods based on minimum spanning tree (MST) can first find a MST and then construct a tour from the MST thus built [5]. Except these constructive methods, there are also some methods which repeatedly improve a given tour. Among them, 2-opt, 3-opt heuristics [6] and the Lin-Kernighan (LK) method [7] are widely used and LK method might be one of the most efficient methods for TSPs. The basic idea of these methods is to iteratively delete some edges and replace them with different edges. Some other methods, such as guided local search [8], have also been applied to TSPs.
  Advantages of these local search methods are (1) they are simple to understand and implement; and (2) they are efficient, *i.e.*, they usually find a tour very quickly. However as their name suggests, local search methods can not find good approximations.
- *Global search methods:* Global optimization methods have also been applied to tackling TSPs. By considering each city as a neuron and edges as neuron connections, a TSP can be converted to a constrained network optimization problem and be tackled by neural networks [9] or elastic networks [10] methods. Evolutionary algorithms (EAs) can deal with permutation problems directly by designing proper problem encoding strategies and reproduction operators [11], [12], [13]. Some new EA diagrams, such as ant colony optimization [14], particle swarm optimization [15], *etc.*, have been extended to deal with TSPs. Simulated annealing (SA) is another global search method which has been applied to TSPs [16]. Unlike neural network and EA methods, SA emphasizes more on how to update a solution when the new one is worse than the parent one.
  Comparing to local search methods, global search methods utilize more randomness in the search process. Although there is not guarantee, global search methods still have the probability to find the global optimum tour for a TSP. These global search methods are usually criticized by their slow convergence and bad performance on large scale problems.

The above two groups of methods have their own advantages and disadvantages. It is also natural to combine them to improve their performances on TSPs [17], [18]. Recently,

978-1-4244-9953-3/11/$26.00 ©2011 IEEE

multiobjective TSPs have also attracted much attention [19], [20].

Heuristic information about the problems to solve is always helpful for designing efficient problem solvers. Both local search methods and global search methods mentioned above utilize TSP heuristic information more or less. In this paper, we propose a probabilistic neighborhood model to use TSP heuristic information and to guide TSP tour reproduction. We apply this model to an SA based algorithm and call the new algorithm *probabilistic neighborhood model biased simulated annealing (pnm SA)*.

The rest of the paper is organized as follows. Section II describes the basic idea of pnm SA and the details to implement it. The basic SA is introduced as well. Section III presents the experimental results of pnm SA and its competitor, a basic SA. Finally, Section IV concludes this paper and outlines some topics for further research along this line.

II. Probabilistic Neighborhood Model and Its Implementation in SA

It has been proved that most of the edges of the best tour of a TSP are connections between neighbor cities. Thus when applying reproduction operators to a TSP tour, 'good' edges, which are linked by cities close to each other, should be kept; while 'bad' edges, which are linked by cities far away from each other, should be destroyed. The key issues to use this heuristic information are (1) measuring the closeness between cities, and (2) thus guiding operations on TSP tours. We address the two issues by building a probabilistic neighborhood model. Since the model is highly related to the reproduction operators, we introduce a basic SA algorithm first, and then modify its reproduction operators.

A. Basic SA

In [21], an SA, which is called basic SA hereafter, for TSP was introduced. The work in this paper is based on the basic SA.

1) Algorithm framework: In each generation, basic SA maintains a TSP tour $\pi = (\pi_1\pi_2\cdots\pi_n)$. The framework of basic SA is shown in Algorithm 1. We would like to explain the above algorithm as follows.

- *Tour structure:* The tour π forms a cycle. Thus the next city of city n, city $(n+1)$ is actually city 1. *i.e.* $1 = (n+1)$ mod n.
- *Initialization:* In *Line 1*, the initial tour is randomly generated as follows: set $\pi_i = i$ for all $i = 1, 2, \cdots, n$; then for each $i = 1, 2, \ldots, n-1$, randomly select $\pi_j \in \{\pi_i, \pi_{i+1}, \cdots, \pi_n\}$ and exchange π_i and π_j.
- *Termination condition:* There are two termination conditions: a given maximum number of generations, $maxt$, is reached or the tour has not been changed in $maxut$ generations in *Line 4*.
- *Selection procedure:* An SA rule based selection is performed in *Lines 7 -13*. There are three situations: (1) the offspring is better and it is accepted; (2) the offspring is worse but it is accepted by a probability where $rand()$ returns a random number in $[0, 1]$; and (3) the offspring is worse and it is discarded.

Algorithm 1: Procedure of basic SA

```
1  Generate an initial tour π and calculate its length f(π);
2  Set the initial temperature T := T_0;
3  Set iteration index t := 1;
4  while not terminate do
5  |  Generate a new tour π' by a reproduction operator
   |  and calculate its length f(π');
6  |  Set Δf := f(π') − f(π');
7  |  if Δf < 0 then
8  |  |  Replace π by π';
9  |  else if exp{−Δf/T} < rand() then
10 |  |  Replace π by π';
11 |  else
12 |  |  Discard π';
13 |  end
14 |  if t mod T_u = 0 then
15 |  |  Update temperature T := αT;
16 |  end
17 |  Update iteration count t := t + 1.
18 end
```

Algorithm 2: Reproduction procedure of basic SA

```
1  Randomly select a sub-tour from π and denotes it as:
       (π_{i_1} ··· π_{i_k})
   where π_{i_1} is connected with π_{i_1−1}, π_{i_k} is connected with
   π_{i_k+1}, and 1 < k < n − 1;
2  if rand() < 0.5 then
3  |  Revrse it and get a new tour π' as:
   |      (π_1 ··· π_{i_1−1}π_{i_k} ··· π_{i_1}π_{i_k+1} ··· , π_n);
4  else
5  |  Randomly select an edge π_jπ_{j+1} from the rest of the
   |  edges to destroy;
6  |  Insert the sub-tour between π_j and π_{j+1} to get a new
   |  tour π' as:
   |      (π_1 ··· π_{i_1−1}π_{i_k+1} ··· π_jπ_{i_1} ··· π_{i_k}π_{j+1} ··· π_n).
7  end
```

2) Reproduction procedure in basic SA: The basic SA utilizes two reproduction operators: one is to reverse a sub-tour; the other is to insert a sub-tour into another location. In each generation, one operator is randomly selected to generate a new tour. The details of the procedure are introduced in Algorithm 2.

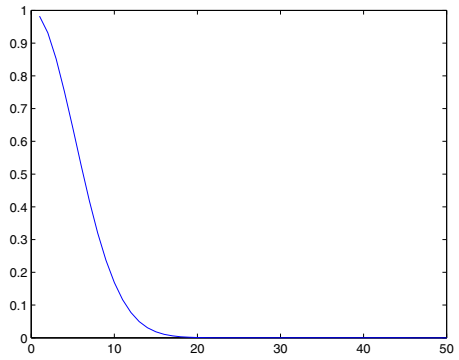


Fig. 1. P_{i,i_j} versus j in the case of $n = 50$ and $\beta = 0.15$: the horizontal coordinate denotes j, and the vertical coordinate denotes P_{i,i_j}.

B. Probabilistic Neighborhood Model and Its Implementation

It is clear from Algorithm 2, both the sub-tours to maintain and the edges to destroy are randomly selected. Thus the 'good' edges could be destroyed and the 'bad' edges could be kept. The probabilistic neighborhood model is proposed to solve this problem.

Before executing the algorithm, the distances between each pair of the cities are calculated. For city i, all other cities $j = 1, 2, \cdots, n$ and $j \neq i$ are sorted by an ascending order according to their distances to city i, *i.e.* $d_{i,j}$. Let the sorted distances be

$$d_{i,i_1} \leq d_{i,i_2} \leq \cdots \leq d_{i,i_{n-1}}$$

where $i_j, j = 1, 2, \cdots, n-1$ is the i_jth closest city to city i. Define a probability as

$$P_{i,i_j} = \exp\left\{-\frac{j^2}{(\beta n)^2}\right\}$$

where $j = 1, 2, \cdots, n-1$, and β is an algorithm parameter.

P_{i,i_j} actually measures a 'connection strength' or another kind of 'closeness' between city i and city i_j. It should be noted that we do not use the distance d_{i,i_j} but use the order j to define P_{i,i_j}. The reason is that the distances between a city and other cities may be quite different. Fig. 1 illustrates the relationship between P_{i,i_j} and j in the case of $n = 50$ and $\beta = 0.15$. It is clear that with $\beta = 0.15$, about top 10% closest neighbor cities connected with city i have over than 70% probability to be kept in the selection procedure.

Based on the probability neighborhood model, the procedures of sub-tour selection and edge selection are shown in Algorithms 3 and 4 respectively.

Algorithm 3: Procedure of sub-tour selection

```
repeat
    Randomly select a start city π_i;
    Set k := 0, s := 0;
    while s − k < n − 2 and rand() < P_{π_{i−k−1}, π_{i−k}} do
        Set k := k − 1;
    end
    while s − k < n − 2 and rand() < P_{π_{i+s}, π_{i+s+1}} do
        Set s := s + 1;
    end
until s − k ≥ 1;
return sub-tour (π_k π_{k+1} ··· π_s).
```

Algorithm 4: Procedure of edge selection

```
Randomly select a start city π_i;
while π_i π_{i+1} is in the selected tour or
  rand() < P_{π_i, π_{i+1}} do
    Set i := i + 1;
end
return edge π_i π_{i+1}.
```

III. Experiments

In this paper, 12 widely used test instances from TSPLIB [22] are used in experimental study: att48, st70, eil76, rat99, ch130, bier127, kroA150, kroB200, ts225, lin318, rd400, and gr431. The numbers in the problem names denote the number of the cities in the problems.

Both pnm SA and basic SA are assessed by the 12 test problems and the parameters in the experiments are as follows.

- The initial temperature is $T_0 = 1.0$.
- The maximum number of generations is $maxt = 10,000n$.
- The maximum number of unchanged generations is $maxut = 100n$.
- The parameters to update SA temperature are $T_u = 100n$ and $\alpha = 0.95$.
- The parameter in pnm SA is $\beta = 0.15$.
- The experimental results are based on 100 independent execution of the two algorithms for each instance.

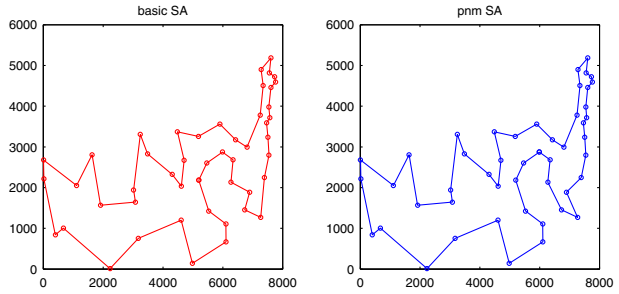
A. Experimental Results

The statistical results, including the min, max, mean, std. values over 100 independent runs, are shown in Table I. To compare the performances of the two algorithms directly, the best tours among 100 runs are also plotted in Figs. 2 and 3 for some test problems.

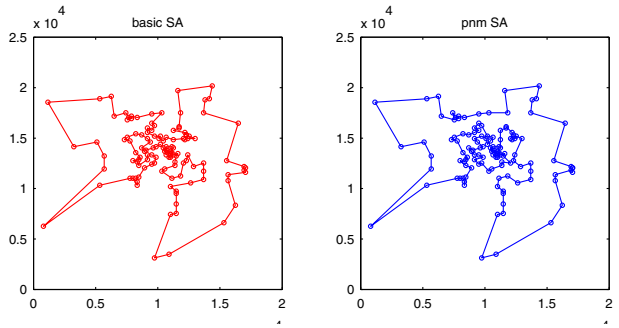
The statistical results in Table I indicate that on average, pnm SA outperforms basic SA on 9 out of all 12 test problems. For these 9 problems in which pnm SA shows better performances on mean values, most of the min, max and std. values of pnm SA are also smaller than those of basic SA. This suggests that pnm SA is very stable on these problems. As to the quality improvements, the mean values obtained by pnm SA are improved by -0.83%, 0.42%, 0.57%, 1.97%, 1.77%, -1.30%, 2.24%, 2.19%, 1.98%, 3.56%, 3.31%, and -3.75% comparing to those obtained by basic SA on the 12 instances respectively. We can see that for large-size problems such as ts225, lin318, and rd400, the improvements are bigger than small-size problems such as st70 and eil76. The reason

TABLE I
STATISTICAL RESULTS OBTAINED BY THE TWO ALGORITHMS ON 12 TEST INSTANCES OVER 100 RUNS.

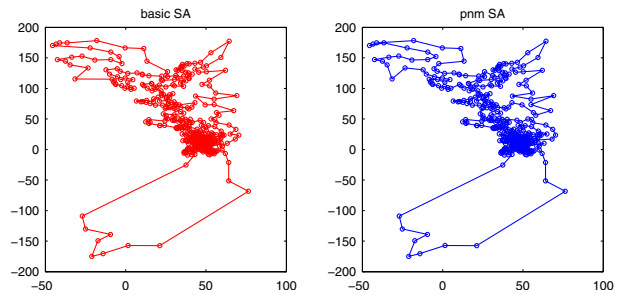
	basic SA				pnm SA			
Instance	min	max	mean	std.	min	max	mean	std.
att48	33607.700	36320.900	**34519.173**	522.312	33639.300	36531.700	34804.854	598.792
st70	677.110	732.671	694.526	10.616	677.110	716.259	**691.613**	7.904
eil76	550.129	582.208	562.827	6.272	546.503	576.378	**559.633**	6.063
rat99	1241.750	1358.450	1297.121	23.800	1219.240	1331.420	**1272.005**	26.199
ch130	6190.870	6852.050	6500.155	149.338	6150.340	6689.550	**6387.373**	119.438
bier127	120775.000	138167.000	**126804.010**	3190.333	121313.000	135739.000	128476.740	3332.543
kroA150	27534.900	29909.400	28653.069	537.495	26673.900	29785.900	**28025.954**	609.870
kroB200	30775.700	33352.700	32130.468	601.913	29998.800	32870.800	**31443.235**	528.453
ts225	127116.000	141824.000	134200.330	3133.361	126646.000	140864.000	**131588.880**	3048.688
lin318	44416.800	48292.000	46292.823	938.494	43369.700	47332.900	**44702.806**	751.894
rd400	16308.600	17366.600	16820.959	217.280	15915.300	16764.700	**16282.801**	178.421
gr431	1968.100	2072.850	**2021.432**	17.778	2013.620	2183.630	2097.142	31.618



(a) att48

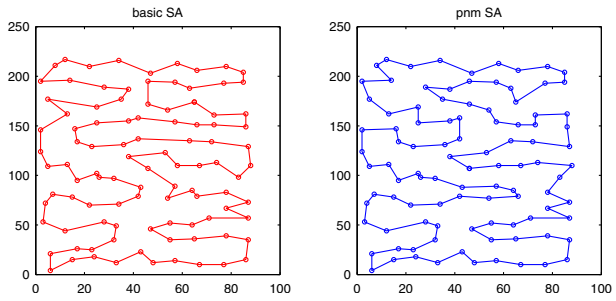


(b) bier127

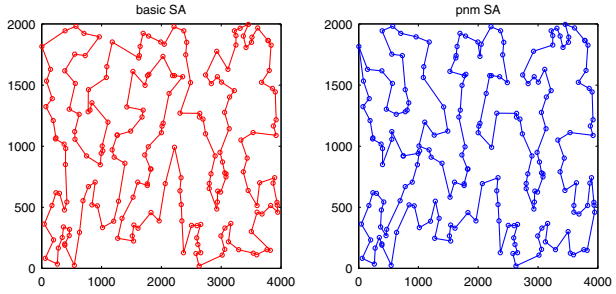


(c) gr431

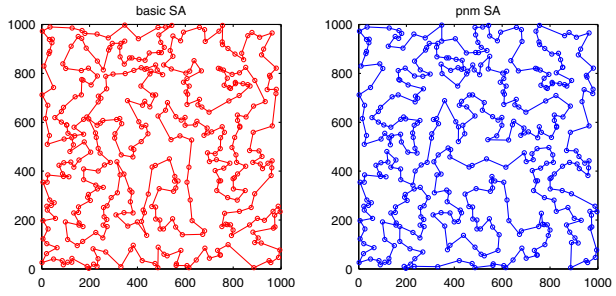
Fig. 2. The best tours found by basic SA and pnm SA on three problems. The best tours found by basic SA are better than those by pnm SA.



(a) rat99



(b) kroB200



(c) rd400

Fig. 3. The best tours found by basic SA and pnm SA on three problems. The best tours found by pnm SA are better than those by basic SA.

might be that (1) for small-size problems, basic SA could also achieve good tours in a given iterations; (2) for large-size problems, the probabilistic neighborhood could help the algorithm to find better tours quickly.

Fig. 2 plots the best tours of att48, bier127 and gr431 found by the two algorithms. The best tours found by basic SA are better than those found by pnm SA. For these three problems, the statistical results in Table I also indicate that basic SA outperforms pnm SA. Fig. 3 plots the best found tours on rat99, kroB200 and rd400. For these problems, pnm SA performs better than basic SA. Comparing Fig. 2 and Fig. 3, we can see that on bier127 and gr431, most of the cities are located in small areas. This might be the reason why pnm SA performs worse than basic SA on these problems. How to deal with the problems with this property is worth for further research in the future.

IV. Conclusions

In this paper, we proposed a probabilistic neighborhood model and applied it to an SA algorithm for TSPs. The roles of the probabilistic neighborhood model are: (1) keeping 'good' city connections, and (2) destroying 'bad' city connections. It is achieved by defining a probability which represents the 'strength' or 'closeness' of two neighbor cities. The proposed method, called pnm SA, was compared with a basic SA on a set of widely used test instances. The statistical results shown that the probabilistic neighborhood model can improve the performance of a TSP solver.

It should be noted that the results obtained in this paper are still very preliminary and much work needs to be done in the future.

- Investigate the properties of probabilistic neighborhood model further.
- Combine the probabilistic neighborhood model with other local or global search methods.
- Apply the probabilistic neighborhood model to other problems such as N-Queen problems.

Acknowledgment

This work is supported by National Basic Research Program of China (No.2011CB707104), National Science Foundation of China (No.61005050), Program for New Century Excellent Talents in University (No.NCET-08-0193), East China Normal University Innovation Program (No.78210059), and Creative Experimental Project of National Undergraduate Students (No. 101026914).

References

[1] E. L. Lawler, J. K. Lenstra, A. H. G. R. Kan, and D. B. Shmoys, *The Traveling Salesman Problem: A Guided Tour of Combinatorial Optimization*. Wiley, 1985.

[2] G. Reinelt, *The Traveling Salesman: Computational Solutions for TSP Applications*. Springer-Verlag, 1994.

[3] M. R. Garey and D. S. Johnson, *Computers and Intractability: A Guide to the Theory of NP-Completness*. W.H. Freeman, 1979.

[4] D. S. Johnson and L. A. Mcgeoch, *Local Search in Combinatorial Optimization*, 1997, ch. The Traveling Salesman Problem: A Case Study in Local Optimization, pp. 215–310.

[5] M. Held and R. M. Karp, "The traveling-salesman problem and minimum spanning trees," *Operations Research*, vol. 18, no. 6, pp. 1138–1162, 1970.

[6] G. A. Croes, "A method for solving traveling salesman problems," *Operations Research*, vol. 6, no. 6, pp. 791–812, 1958.

[7] S. Lin and B. W. Kernighan, "An effective heuristic algorithm for the traveling salesman problem," *Operations Research*, vol. 21, no. 2, pp. 498–516, 1973.

[8] C. Voudourisa and E. Tsang, "Guided local search and its application to the traveling salesman problem," *European Journal of Operational Research*, vol. 113, no. 2, pp. 469–499, 1999.

[9] J. Y. Potvin, "The traveling salesman problem: A neural network perspective," *ORSA Journal on Computing*, vol. 5, no. 4, pp. 328–348, 1993.

[10] R. Durbin, R. Szeliski, and A. Yuille, "An analysis of the elastic net approach to the traveling salesman problem," *Neural Computation*, vol. 1, no. 3, pp. 348–354, 1989.

[11] I. M. Oliver, D. J. Smith, and J. R. C. Holland, "A study of permutation crossover operators on the traveling salesman problem," in *2nd International Conference on Genetic Algorithms and Their Application*, 1987, pp. 224–230.

[12] T. Guo and Z. Michalewicz, "Inver-over operator for the TSP," in *5th Parellel Problem Solving from Nature*, 1998, pp. 803–812.

[13] H.-K. Tsai, J.-M. Yang, Y.-F. Tsai, and C.-Y. Kao, "An evolutionary algorithm for large traveling salesman problems," *IEEE Transactions on Systems Man and Cybernetics Part B*, vol. 34, no. 4, pp. 1718–1729, 2004.

[14] M. Dorigo and L. Gambardella, "Ant colony system: A cooperative learning approach to the traveling salesman problem," *IEEE Transactions on Evolutionary Computation*, vol. 1, no. 1, pp. 53–66, 1997.

[15] K.-P. Wang, L. Huang, C.-G. Zhou, and W. Pang, "Particle swarm optimization for traveling salesman problem," in *International Conference on Machine Learning and Cybernetics*, 2003, pp. 1583–1585.

[16] K. Meer, "Simulated annealing versus metropolis for a TSP instance," *Information Processing Letters*, vol. 104, no. 6, pp. 216–219, 2007.

[17] H. Duan and X. Yu, "Hybrid ant colony optimization using memetic algorithm for traveling salesman problem," in *IEEE International Symposium on Approximate Dynamic Programming and Reinforcement Learning*, 2007, pp. 92–95.

[18] R. Baraglia, J. I. Hidalgo, and R. Perego, "A hybrid heuristic for the traveling salesman problem," *IEEE Transactions on Evolutionary Computation*, vol. 5, no. 6, pp. 613–622 613–622 613–622, 2001.

[19] Z. Yan, L. Zhang, L. Kang, and G. Lin, "A new MOEA for multi-objective TSP and its convergence property analysis," in *2nd international conference on Evolutionary multi-criterion optimization*, ser. LNCS, vol. 2632, 2003, pp. 342–354.

[20] W. Peng, Q. Zhang, and H. Li, *Multi-objective Memtic Aalgorithms*, ser. SCI. Springer, 2007, vol. 171, ch. Comparison between MOEA/D and NSGA-II on the Multi-Objective Travelling Salesman Problem, pp. 309–324.

[21] W. H. Press, S. A. Teukolsky, W. T. Vetterling, and B. P. Flannery, *Numerical Recipes 3rd Edition: The Art of Scientific Computing*. Cambridge University Press, 2007.

[22] G. Reinelt, "TSPLIB - a traveling salesman problem library," *ORSA Journal on Computing*, vol. 3, no. 376-384, 1991.