Probe Machine Based Consecutive Route Filtering Approach to Symmetric Travelling Salesman Problem

Md. Rahman, Jinwen Ma

Probe Machine Based Consecutive Route Filtering Approach to Symmetric Travelling Salesman Problem

Md. Azizur Rahman and Jinwen Ma*

Department of Information Science
School of Mathematical Sciences & LMAM
Peking University, Beijing, 100871, P.R. China
mdazizur201171@pku.edu.cn, jwma@math.pku.edu.cn

Abstract. The travelling salesman problem (TSP) is one of the NPC combinatorial optimization problems and still now it remains as an interesting and challenging problem in the field of combinatorial optimization. In this paper, we propose a consecutive route filtering approach to solving the symmetric TSP with the help of probe concept such that the worse routes are filtered out step by step by using a rigorous predesigned step proportion. In this way, it is important to set up a reasonable value of the step proportion which is needed in each step during the filtering process. Actually, our proposed algorithm is implemented on the set of symmetric TSP benchmarks with both small and large numbers of cities from the TSPLIB dataset. It is demonstrated by the experimental results that our proposed algorithm can obtain the best results in some cases and generally get the approximation results close to the best known solutions.

Keywords: Probe Machine· Travelling Salesman Problem· Filtering Proportion· Discrete Optimization.

1 Introduction

In the field of combinatorial optimization, TSP is arguably the most prominent, popular, and widely studied problem. It belongs to the class of NP-complete problems [1], i.e., the most difficult problems without any exact algorithm to effectively solve it in polynomial time. In fact, with the increase of number of cities, the executive time of any existing algorithm increases super-polynomially or even almost exponentially [2]. So, its improvement has been drawn much attention to researchers due to the growing demands from vast practical applications in different areas relevant to real life such as vehicle routing, drilling holes in a circuit board, overhauling gas turbine engine, X-ray diffraction, storage and picking of stock in warehousing, computer wiring, interview scheduling, crew scheduling,

* Corresponding author.

mission planning, DNA sequencing, data association, image processing, pattern recognition and so on [3, 4].

Originally, this problem comes into being with a salesman who wants to visit every city exactly once from a list of cities to sell his products and finally return to the starting city where his purpose is to minimize the total tour cost. In graph theory, the symmetric TSP is defined by a complete undirected graph $G = (V, E)$, where $V = \{v_1, v_2, \cdots, v_N\}$ is the set of nodes and $E = \{(v_i, v_j) : v_i, v_j \in V, i \neq j\}$ represent the set of edges [3]. Moreover, a symmetric cost matrix $D_{N\times N}$ is assigned on E for representing the weight of edges. Also, the TSP can be stated as a permutation problem [24, 25] with the motive of finding a permutation ψ among N cities that minimize the following objective function:

$$f(\psi) = \sum_{i=1}^{N-1} d_{\psi(i),\psi(i+1)} + d_{\psi(N),\psi(1)} \tag{1}$$

where $\psi(i)$ indicate the city which is visited at step i, $i = 1, 2, \cdots, N$ and $f(\psi)$ is the cost of a permutation ψ. The Euclidean distance $d_{i,j}$, between any two cities $i(x_1, y_1)$ and $i(x_2, y_2)$ is calculated by:

$$d_{i,j} = \sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2} \tag{2}$$

In the case of symmetric TSP, the distance from city i to city j is the same as the distance from city j to city i, i.e.,$d_{i,j} = d_{j,i}$.

Although the TSP problem is quite simple, the main complexity comes from the large number of possible solutions. To solve an N-city symmetric TSP problem, ${}^{(N-1)!}/_{2}$ different possible routes arise so that the direct optimization procedure cannot accomplish in a polynomial time. For this reason, researchers have been trying to solve this problem in two alternative ways instead of finding the exact solution. The first way is to develop an optimization algorithm to ensure the optimal solution with longer running time, while the second way is to develop a heuristical algorithm which can reduce the computational time significantly and provide only near-optimal solution.

The objective of this work is to propose a new algorithm to solve the symmetric TSP with the help of probe machine. The proposed approach is able to filter out worse routes and keep potential routes step by step by using an appropriate choice of proportion value. The rest of the paper is organized as follows. We review some related works in Section 2. Section 3 presents our propose framework. The experimental results are summarized in Section 4 and the final section gives the conclusion and future research.

2 Related Work

During the past decades, various TSP algorithms have been proposed for finding an optimal or near-optimal solution. Branch and bound algorithm [5], dynamic programming algorithm [6] and cutting plane algorithms [7] are most popular and well known. However, these algorithms are limited to small size instances and unable to implement in large-scale problems. To get rid of this limitation, researchers have developed the heuristic methods. In fact, the nearest neighbour algorithm (NNA) [8] is a simple and easily implementable heuristic algorithm. It starts with a randomly chosen city and adds the nearest unvisited city step by step until all the cities are contained in the tour. Rosenkrantz et al. [9] developed a repetitive nearest neighbour algorithm (RNNA) and insertion algorithm (IA) to extend the NNA.

Several algorithms were reported to launch with a complete route and improve it iteratively through a simple modification. 2-opt [10] and 3-opt [11] are exoteric methods in this category, where a few edges (2 for 2-opt and 3 for 3-opt) are firstly removed from the current tour and then replace them by the corresponding number of different edges to obtain a shorter tour. Lin and Kernighan [12] enlarged this concept to k-opt where k is chosen in a reasonable way. The variable neighbourhood search (VNS) [13,14] was also based on the neighbours that are obtained iteratively by a systematic change of the node of the initial tour. Most recently, Hore et al. [15] made a significant improvement on the VNS algorithm.

In this context, population-based metaheuristic algorithms draw much more attention among the researchers in the long run. Simulated annealing (SA) [16] can be successfully applied to get a global optimal solution, but it requires longer computational time. Several types of Genetic algorithms (GAs) [17] are also introduced to rely on the principles of natural selection and genetics, and there are some developments using the assorted operator and selection methods [18]. Ant colony optimization (ACO) [19] is an alternative technique which depends on the foraging behaviour of ant colony and its modifications are also reported in some works [20-22]. Recent population-based algorithm referred to as Symbiotic organisms search (SOS) proposed by Cheng and Prayogo [23] employs the optimization scheme through the mutualism, commensalism and parasitism phase, and an extension of this algorithm developed by Ezugwu et al. [24] strengthens the use of the mutation operator in the local search process.

However, population-based methods can solve the TSP problem quickly, but might be easily trapped into a local optimum solution because there are a group of random numbers and fine-tuning parameters in the process. Certainly, some hybrid algorithms can be established to enhance the overall algorithm performance. Recently, Ezugwu et al. [25] proposed a hybrid optimization algorithm which fuses the SOS and SA algorithm together. On the other hand, Ozden et al. [26] demonstrated how the parallel computing techniques can be used for TSP and significantly decrease the overall computational time with the increase of CPU utilization.

3 Proposed Algorithm

In the proposed framework, we take an attempt to solve the symmetric TSP with the help of probe concept by using a rigorous step proportion value which filters out worse route gradually and finally reaches to an optimal route. For an N-city problem, the process needs to complete $[N/2]$ steps, where

$$[N/2] = \begin{cases} \frac{N}{2} & \text{if } N \text{ is even} \\ \frac{N-1}{2} & \text{if } N \text{ is odd} \end{cases} \tag{3}$$

The concept of probe, working steps of our proposed framework and the filtering proportion value in each step are given in details in the following subsection consecutively.

3.1 Probe Concept

Actually, the probe is a tool which is used to detect a certain substance accurately. The hypothetical probe machine was introduced by Xu [27], in which there were two types of probes: connective and transitive probes. The connective probes were used to connect two data, while the transitive probes were used to pass information from one data to another data through the data fibers. The probe machine was used to solve two types of NP-complete problems: Hamilton and the graph colouring problem. By the inspiration of his work, we try to present a new type of probe and also present a new filtering mechanism for solving the symmetric TSP using a proportion value. Here, each of the sub-route treats as a probe which is able to automatically find out two unvisited cities and connect them through their wings. In this way, each probe gradually enhances in every step and continue until all cities are included in the route. A sample of 3 cities probe and 5, 7, 9 cities probes generated from 3 cities probe in first four steps are shown in **Fig.1.** In the **Fig.1.**, the sample probes are denoted by x_{ijk}, x_{ijklm}, $x_{ijklmnp}$, $x_{ijklmnpst}$ and their wings are as follows:

$$\omega(x_{ijk}) = \{x^j_{ijk}, x^k_{ijk}\}, \omega(x_{ijklm}) = \{x^l_{ijklm}, x^m_{ijklm}\}$$

$$\omega(x_{ijklmnp}) = \{x^n_{ijklmnp}, x^p_{ijklmnp}\}, \omega(x_{ijklmnpst}) = \{x^s_{ijklmnpst}, x^t_{ijklmnpst}\}$$

3.2 Working Steps

The proposed algorithm accomplishes two tasks in each step. That is, it firstly generates the possible probes and then filters out the worse probes. The whole process consists of several steps which are described in details in the followings:

Step-1: In the first step, the process generates three city probes by considering two paths for each city position. The two path is constructed with an internal city having other two cities that are adjacent to the internal city. More formally, consider c_i, c_j and c_k are three different cities, where c_i is the internal city with c_j and c_k are two different cities both are adjacent to c_i, then the set of all two paths with internal city c_i is denoted by $F^2(c_i)$ and is defined by Eq. (4)[27]:

$$F^2(c_i) = \{c_j c_i c_k \triangleq x_{ijk} : c_j, c_k \in I(c_i); i, j, k \text{ are mutually different}\} \tag{4}$$

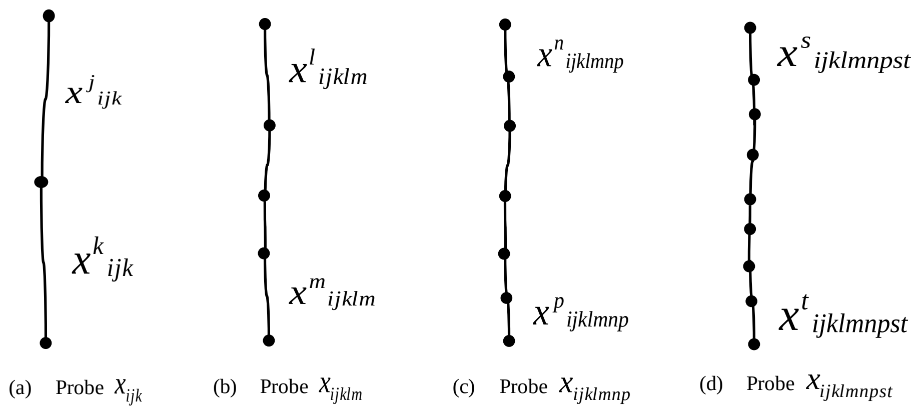

Fig. 1. A sample of 3, 5, 7 and 9 cities probes with their wings.

where $I(c_i)$ is the set of cities adjacent to c_i and x_{ijk} represent the probe that covers three cities c_i, c_j and c_k. Thus, we construct all possible three cities probe for an N-city problem as follows:

$$X_3 = \cup_{i=1}^{N} F^2(c_i) = \cup_{i=1}^{N} \{x_{ijk} : c_j, c_k \in I(c_i); i \neq j, k; k \neq j\} \tag{5}$$

So there are $N_3 = \mid X_3 \mid = \frac{N(N-1)(N-2)}{2}$ such types of 3 city probes produced in the first step for an N-city symmetric problem. Each probe x_{ijk} has exactly two types of wings x^j_{ijk} and x^k_{ijk}, and this wing includes two other unvisited cities automatically on the probe in next step.

The filtering process starts after the probes have been created. In this phase, the system keeps some potential probes by using a proportion value defined in subsection 3.3 and we consider these probes as good probes. The algorithm adopts Euclidean distance (defined in Eq. (2)) to calculate the cost of the probes, and based on the cost it keeps the best N_{G_3} probes. Consequently, those probes are good whose best order lies within the range $[1, N_{G_3}]$, where N_{G_3} indicates the total number of good probes with 3 cities. If we choose α_3 as a proportion value in the first step, then the total number of good probes is as follows:

$$N_{G_3} = \mid G_3 \mid = \alpha_3 N_3, G_3 \subset X_3 \tag{6}$$

where G_3 is the set of all good probes after completing the filtering task in first step.

Step-2: Based on good probes obtained from the first step, the algorithm produces 5 cities probe in the second step. Each good probes enlarge the route through the wings by adding two unvisited cities. In this step, the system needs to add exactly two cities in the probe from the remaining $(N-3)$ unvisited cities. So based on G_3, the probe can be built by the following ways:

$$X_5 = \{c_l c_j c_i c_k c_m \triangleq x_{ijklm} : c_j c_i c_k \in G_3; c_l, c_m \in R_{N-3}(c_j c_i c_k); i \neq j \neq k \neq l \neq m\} \tag{7}$$

where $R_{N-3}(c_j c_i c_k)$ is the set of remaining unvisited cities corresponding to good probe $c_j c_i c_k$, i.e., $R_{N-3}(c_j c_i c_k) = \{c_1, c_2, \cdots, c_N\} - \{c_i, c_j, c_k\}$. It is noted that the two city c_l and c_m are chosen in both order from $R_{N-3}(c_j c_i c_k)$. Hence, the total produced probe is estimated based on Eq. (8):

$$N_5 = | X_5 | = 2N_{G_3}\binom{N-3}{2} = N_{G_3}(N-3)(N-4) \tag{8}$$

After generating 5 cities probe the process also takes a filtering proportion value in this step which gives some good probes and this probe will be used in the next step to create 7 cities probe. Therefore, the number of good probes at the end of this stage can be calculated by the following equation:

$$N_{G_5} = | G_5 | = \alpha_5 N_5, G_5 \subset X_5 \tag{9}$$

where α_5 represent the filtering proportion value in the step and G_5 is the set of all good 5 cities probe, respectively.

***Step-(k+1)*:** The $(k+1)^{th}$ step of the procedure starts with the good probes (G_{2k+1}) and their corresponding cost which are obtained from k^{th} step, where

$$N_{G_{2k+1}} = | G_{2k+1} | = \alpha_{2k+1} N_{2k+1} \tag{10}$$

In the above Eq. (10), $N_{G_{2k+1}}$ indicates the number of total good probes, α_{2k+1} is the filtering proportion value and N_{2k+1} represent the set of possible generated probes in k^{th} step. There are $(2k+1)$ cities in each probe of the above mentioned good probes. In fact, when the algorithm arrive in this step, it needs to include $(N-2k-1)$ more cities in the probe. By adding two cities from $(N-2k-1)$, each probe becomes $(2k+3)$ cities probe. The set of creates probe in the current step based on good probes (G_{2k+1}) is denoted by X_{2k+3} and is defined by:

$$X_{2k+3} = \{\underbrace{c_s \quad \overbrace{c_l c_j \ldots c_k c_m}^{\text{(2k+1)cities good probe}} \quad c_t}_{\text{(2k+3)cities probe}} : c_l \ldots c_m \in G_{2k+1}; c_s, c_t \in R_{N-2k-1}(c_l \ldots c_m), s \neq t\} \tag{11}$$

The two cities c_s and c_t are taken in both ways, i.e., (c_s, c_t) and (c_t, c_s) both are included and $R_{N-2k-1}(c_l \ldots c_m) = \{c_1, c_2, \cdots, c_N\} - \{c_l, c_j, \ldots, c_k, c_m\}$. The total produced probes in this step as shown in Eq. (12):

$$N_{2k+3} = | X_{2k+3} | = 2N_{G_{2k+1}}\binom{N-2k-1}{2} = N_{G_{2k+1}}(N-2k-1)(N-2k-2) \tag{12}$$

Now, the algorithm adopts a filtering proportion value α_{2k+3} that filled out the potential probe. So the number of good probes with $(2k+3)$ cities obtained as:

$$N_{G_{2k+3}} = | G_{2k+3} | = \alpha_{2k+3} N_{2k+3}, G_{2k+3} \subset X_{2k+3} \tag{13}$$

where G_{2k+3} is the set of all good probes at the end of this step.

In this way, the probe construction and filtering mechanism continue until all the cities are contained in the probe. When the algorithm reaches in the last step, it generates probes of N cities and finds out the best probes from all generated probes. It is important to mention that in the last step of the even number of problems, there is one city remaining to visit, in this case the probe uses any one of its wings to include the city properly.

3.3 Filtering Proportion in Each Step

The filtering proportion plays a vital role in our proposed framework. Usually, an inappropriate choice of proportion value leads to trap the whole process and yields a worse solution as well as take longer running time. So it is very challenging task to design an efficient filtering proportion value that keeps potential probes from produced probes in each step. In this work, we address a filtering proportion value of k^{th} step through the experiments by trial and error method that is denoted as α_{2k+1} and defined as:

$$\alpha_{2k+1} = \frac{C}{N+\sqrt{k}}; k = 1, 2, 3, \cdots \cdot \tag{14}$$

where N represents the number of cities and C is the constant. The above proportion value has been chosen in such a way that it decreases slowly step by step, i.e., it maintains the relation $\alpha_3 > \alpha_5 > \cdots > \alpha_{2k+1} > \alpha_{2k+3} \cdot \cdots$. During the experiment, we found that the number of good probes increases with the gradually decreases of proportion value in each step, but after a certain step later it also starts to decrease. In some cases, we use some strategies in the remaining steps such as trade off to decrease the proportion value and sometimes it is increased by $\frac{1}{(N-2k-1)(N-2k-2)}$. By the experiment, we estimated that the value of C lie within the interval $[\frac{2(N+1)}{N(N-1)(N-2)}, \frac{800(N+1)}{N(N-1)(N-2)}]$ for our computational results.

4 Experimental Results

In this section, we conduct a number of experiments for measuring the capability of our proposed framework based on several TSPLIB [28, 29] datasets ranging from 14 up to 1432 cities. Actually, TSPLIB is a publicly available library that contains the sample of TSP instances and their corresponding optimum solutions. The technical computations in this study are performed in MATLAB R2016b software by using 4 core GPU system. The average required time is measured by running each instance ten (10) consecutive times. The experimental results from our experiments have been displayed in Table 1. In the left part of the table, the first column represents the serial number (S/N) of the instances, the second column contains the name of each instance, in the third column "Scale" denote the total number of cities in each instance, the fourth column "Results" stands for the length of obtained solution from the experiment, the fifth one represent average required times (in seconds) of each instance, the sixth column available for the best known results (BKR) obtained from the TSPLIB and the right part of the table brings similar indicator as the left part.

Table 1. The computational results of the proposed algorithm for the symmetric TSP

S/N	Instances	Scale	Results	Time(Se.)	BKR	S/N	Instances	Scale	Results	Time(Se.)	BKR
1	burma14	14	3323	2.1148	3323	38	u159	159	51624	137.3	42080
2	p01	15	291	0.0101	291	39	si175	175	22051	166.0453	21407
3	ulysses16	16	6859	4.1738	6859	40	brg180	180	1960	347.3403	1950
4	gr17	17	2085	0.0181	2085	41	rat195	195	2512.8	6.1452	2323
5	gr21	21	2707	0.3412	2707	42	d198	198	17366	2.2832	15780
6	gr24	24	1272	31.278	1272	43	kroA200	200	35420	2.1057	29368
7	fri26	26	937	0.4982	937	44	kroB200	200	35895	2.3493	29437
8	bays29	29	2093	613.2	2020	45	ts225	225	135000	958.7	126643
9	bayg29	29	1667	0.0282	1610	46	tsp225	225	4550.5	16.8851	3916
10	dantzig42	42	762	93.6	699	47	pr226	226	94850	863.8	80369
11	swiss42	42	1376	11.8	1273	48	gil262	262	2736.7	12.0015	2378
12	att48	48	10671	1265.2	10628	49	a280	280	3111.8	8.1597	2579
13	hk48	48	11461	206.8	11461	50	pr299	299	57931	43.1237	48191
14	eil51	51	455.86	17.48	426	51	lin318	318	52547	65.6026	42029
15	berlin52	52	7982.2	3.3017	7542	52	rd400	400	18581	101.6123	15281
16	brazil58	58	25649	2.2556	25395	53	fl417	417	15406	44.7756	11861
17	st70	70	762.34	0.1033	675	54	pcb442	442	61346	55.0617	50778
18	eil76	76	601.46	39.8	538	55	d493	493	42408	136.3764	35002
19	pr76	76	124740	63.4	108159	56	att532	532	34239	258.6158	27686
20	gr96	96	61741	2119.6	55209	57	si535	535	50286	322.1954	48450
21	rat99	99	1443.6	0.9600	1211	58	pa561	561	3313	248.8954	2763
22	rd100	100	9283	21.5	7910	59	u574	574	46191	290.0283	36905
23	kroA100	100	24511	0.2025	21282	60	rat575	575	8066.2	272.50	6773
24	kroB100	100	23568	0.3084	22141	61	p654	654	48380	1489.9	34643
25	kroD100	100	25767	45.3334	21294	62	d657	657	63099	282.42	48912
26	kroE100	100	24571	0.4640	22068	63	u724	724	50391	399.1771	41910
27	eil101	101	729.2	83.4142	629	64	rat783	783	11140	558.2935	8806
28	lin105	105	16638	68.2125	14379	65	pr1002	1002	314850	1566.7	259045
29	pr107	107	50448	36.26	44303	66	si1032	1032	96145	1737.3	92650
30	gr120	120	8255	59.3458	6942	67	u1060	1060	287620	2584.8	224094
31	pr124	124	70605	80.9855	59030	68	vm1084	1084	305270	2816.4	239297
32	bier127	127	132320	383.1	118282	69	pcb1173	1173	73093	2408.1	56892
33	ch130	130	7002	1.2577	6110	70	d1291	1291	64050.8	4280.1	50801
34	pr136	136	116580	2.2622	96772	71	rl1304	1304	323060	4228.5	252948
35	pr144	144	62370	356.24	58537	72	rl1323	1323	357230	4386.1	270199
36	ch150	150	7195.8	0.800	6528	73	nrw1379	1379	68157	4541.2	56638
37	kroB150	150	29319	5.7321	26130	74	u1432	1432	190400	5494.4	152970

5 Conclusion and Future Research

We have established a consecutive route filtering approach with the help of probe concept by using an appropriate filtering proportion value in each step as a new approach to solving the symmetric TSP. From our experimental results, it is found that the proposed approach can effectively reach at the optimum point to all of the tested datasets that contain up to 26 cities and also for the 48 cities dataset (hk48, Instance no. 13 in Table 1.). It is also noted that our obtained solutions are close to the best known solution of some other datasets whereas, the performance of our implemented algorithm is also satisfactory on big scale datasets. In the future, we try to investigate more effective and efficient filtering approach to implement this framework to solve the existing asymmetric travelling salesman problems (aTSPs) and the multi travelling salesman problems (mTSPs). In addition, it is also a good direction to fuse this algorithm with the other existing heuristic or meta-heuristic algorithms together in our further study.

Acknowledgment

This work is supported by the Natural Science Foundation of China for Grant 61171138.

References

1. Papadimitriou, C.H.: The euclidean traveling salesman problem is NP-complete. Theoretical Computer Science **4**(3), 237-244 (1977)
2. Garey, M.R., Johnson, D.S.: Computers and Intractability: A Guide to the Theoryof NP-Completeness. W. H. Freeman, New York (1979)
3. Matai, R., Singh, S.P., Mittal, M.L.: Traveling Salesman Problem: An Overview of Applications, Formulations, and Solution Approaches. Traveling Salesman Problem, Theory and Applications. Prof. Donald Davendra (Ed.), InTech, 1-24 (2010)
4. MacGregor, J.N., Chu, Y.: Human Performance on the Traveling Salesman and Related Problems: A Review. The Journal of Problem Solving **3**(2), Article 2 (2011)
5. Finke, G., Claus, A., Gunn, E.: A two-commodity network flow approach to the traveling salesman problem. Congressus Numerantium **41**, 167–178 (1984)
6. Held, M., Karp, R.M.: A dynamic programming approach to sequencing problems. Journal of the Society for Industrial and Applied Mathematics **10**(1), 196–210 (1962)
7. Fleischmann, B.: A cutting plane procedure for the travelling salesman problem on road networks. European Journal of Operational Research **21**(3), 307–317 (1985)
8. Bellmore, M., Nemhauser, G.L.: The traveling salesman problem: A survey. Operations Research **16**(3), 538-558 (1968)
9. Rosenkrantz, D.J., Stearns, R.E., Philip, M.L.I.: An analysis of several heuristics for the traveling salesman problem. SIAM Journal on Computing **6**(3), 563–581 (1977)
10. Croes, G.A.: A method for solving traveling-salesman problems. Operations Research **6**(6), 791–812 (1958)

11. Lin, S.: Computer solutions of the travelling salesman problem. Bell Systems Technical Journal **44**(10), 2245–2269 (1965)
12. Lin, S., Kernighan, B.W: An effective heuristic algorithm for the traveling salesman problem. Operations Research **21**(2), 498–516 (1973)
13. Hansen, P., Mladenović, N.: An introduction to variable neighborhood search. In: S. Voss, S. Martello, I. Osman, C. C. Roucairol (Eds.), Meta-Heuristics: Advances and Trends in Local Search Paradigms for Optimization, KluwerAcademic Publishers, MA, USA, 433–458 (1999)
14. Hansen, P., Mladenović, N.: Variable neighborhood search: principles and applications. European Journal of Operational Research **130**(3), 449–467 (2001)
15. Hore, S., Chatterjee, A., Dewanji, A.: Improving variable neighborhood search to solve the traveling salesman problem. Applied Soft Computing **68**, 83-91 (2018)
16. Kirkpatrick, S., Gelatt, C.D., Vecchi, M.: Optimization by simulated annealing. Science, New Series **220**(4598), 671–680 (1983)
17. Whitley, D.: A genetic algorithm tutorial. Statistics and Computing **4**(2), 65–85 (1994)
18. Guo, D., Chen, H., Wang, B.: An Improved Genetic Algorithm with Decision Function for Solving Travelling Salesman Problem. Proceeding of 12th ISKE. IEEE Conferences, Nanjing, China, pp. 1-7 (2017). https://doi.org/10.1109/ISKE.2017.8258774
19. Colorni, A., Dorigo, M., Maniezzo, V.: Distributed Optimization by Ant Colonies. In: Varela, F. and Bourgine, P., Eds., Proceedings of European Conference on Artificial Life, Paris, France, pp. 134-142 (1991)
20. Shufen, L., Huang, L., Lu, H.: Pheromone Model Selection in Ant Colony Optimization for the Travelling Salesman Problem. Chinese Journal of Electronics **26**(2), 223-229 (2017)
21. Xiong, N., Wu, W., Wu, C.: An Improved Routing Optimization Algorithm Based on Travelling Salesman Problem for Social Networks. Autonomous and Sustainable Computing for preparing the Internet of Things Environment **9**(6), 985 (2017)
22. Ratanavilisagul, C.: Modified Ant Colony Optimization with Pheromone Mutation for Travelling Salesman Problem. Proceeding of 14th ECTI-CON. IEEE Conferences, Phuket, Thailand, pp. 411-414 (2017). https://doi.org/10.1109/ECTICon.2017.8096261
23. Cheng, M-Y., Prayogo, D.: Symbiotic Organisms Search: A new metaheuristic optimization algorithm. Computers and Structures **139**, 98–112 (2014)
24. Ezugwu, A.E.-S., Adewumi, A.O.: Discrete symbiotic organisms search algorithm for travelling salesman problem. Expert Systems With Applications **87**, 70-78 (2017)
25. Ezugwu, A.E.-S., Adewumi, A.O., Frîncu, M. E.: Simulated annealing based symbiotic organisms search optimization algorithm for traveling salesman problem. Expert Systems With Applications **77**(1), 189–210 (2017)
26. Ozden, S.G., Smith, A.E., Gue, K.R.: Solving large batches of travelling salesman problems with parallel and distributed computing. Computers and Operations Research **85**, 87-96 (2017)
27. Xu, J.: Probe Machine. IEEE transations on neural networks and learning systems **27**(7), 1405-1416 (2016)
28. TSPLIB. http://elib.zib.de/pub/mp-testdata/tsp/tsplib/tsplib.html. Last accessed 16 June 2018
29. https://people.sc.fsu.edu/ jburkardt/datasets/tsp/tsp.html. Last accessed 24 June 2018