Proceedings of Science and Mathematics
Faculty of Science, Universiti Teknologi Malaysia
Vol. 2, 2021, page 97-106

Solving Travelling Salesman Problem Using Simulated Annealing

[1]Indok Zarith Sofia Dzolkepli and [2]Farhana Johar

[1,2]Department of Mathematical Sciences
Faculty of Science, Universiti Teknologi Malaysia,
81310 Johor Bahru, Johor, Malaysia.

e-mail: [1]indokzarithsofia@graduate.utm.my, [2]farhanajohar@utm.my

Abstract This research presents a metaheuristic algorithm called Simulated Annealing (SA) for solving Travelling Salesman Problem (TSP) where the aim is to obtain the best cost. Six instances' data are used which are Att48, Berlin52, St70, Pr76, Eil101 and KroA200. A comparison of the algorithm is being studied among 6 other metaheuristic algorithms which are Tabu Search (TS), Genetic Algorithm (GA), Particle Swarm Optimization (PSO), Ant Colony Optimization (ACO), PSO-ACO and GA-PSO-ACO. SA outperformed other algorithms in all test instances under investigation as it acquired the smallest best cost and the nearest optimal solution to the TSP instances.

Keywords Travelling Salesman Problem; Simulated Annealing; metaheuristic method

1 Introduction

Travelling Salesman Problem (TSP) is one of the most studied classical combinatorial optimization techniques [1]. It is very common in daily life situation especially in business world. TSP imitates the salesman moving around in the cities and wishes to visit every city on his list once and then return back to his original city in the least time. TSP is one of the most fundamental NP-complete combinatorial optimization problems. The standard TSP goes like this: given a set of cities, the goal is to identify the shortest Hamiltonian cycle that starts in one city and visits each of the others once before returning to the starting city [2]. Exact approaches, heuristics, and metaheuristics have all been presented to solve the TSP throughout the years. Heuristics and metaheuristics are preferred to precise techniques due to the significant rise in computing time as the issue size grows, since they exchange optimality for efficiency [3].

The study of TSP has captivated many researchers from diverse field such as mathematics, physics, biology, and operations research. Not only TSP shows all aspects of combinatorial optimization, but it also assists as the platform for the new algorithm ideas like Simulated Annealing (SA), Tabu Search, and Neural Networks [4]. TSP has been applied in a real-world problem such as school bus routing problem, gas turbine engine overhaul, crystallography of X-Ray, warehouse order-picking problem and computer wiring problem [5].

In this research, the focus is on solving the symmetric TSP using one of the powerful metaheuristic methods called SA to obtain the best route that can minimize the total cost travelled by salesman. Computer software called MATLAB is used to solve the TSP where the SA method

is incorporated into the software. This study focuses on 6 datasets taken from TSPLIB which is the benchmark of TSP where the number of cities is 48, 52, 70, 76, 101 and 200. In order to accomplish the research, two objectives have been identified, which are to implement the SA algorithm in solving and finding the best route that can minimize the cost for TSP and to compare the performance of SA with other metaheuristic algorithms which are Tabu Search (TS), Genetic Algorithm (GA), Particle Swarm Optimization (PSO), Ant Colony Optimization (ACO), hybrid of PSO-ACO and hybrid of GA-PSO-ACO.

2 Literature Review

2.1 Simulated Annealing

SA is a meta-heuristic algorithm capable of tackling many hard-combinatorial optimization problems through controlled randomization [6]. It can skip local minima by exploring the solution space in directions that lead to improvement in objective function. SA simulates the annealing process of metal atoms by attempting to find stable states at lower energy levels through controlled cooling.

Simulated annealing is a method of optimization that resembles the material procedure of annealing. It is heated until it gets to a point where it liquefies as a substance experiences annealing and then gradually cools down until the material becomes solid again. The material's last property is highly dependent on the cooling schedule implemented. The structure can become imperfect if the material is easily or quickly cooled down but the final structure is strong if it is slowly cooled.

SA operates by observing the physical mechanism by which a solid is slowly cooled, so that this will ultimately happen at a minimum energy configuration when the structure is frozen [7]. The structure of the material indicates the solution of the problem when solving the optimization problem using simulated annealing. In order to clarify how and when the new solution is perturbed and accepted, temperature is used. Therefore, the main concept behind SA is to use an acceptable given annealing schedule to reduce the cost function, which 'cools' the device slowly enough to achieve the approximately optimal solution.

2.2 Related Works on SA

SA has been shown to be able to solve many combinatorial optimization problems in real life, including the TSP. Two new TSP solution approaches based on SA have been suggested in a previous research [7]. The first method uses Roulette Wheel selection to join nodes chosen from the distance matrix. The second method uses a new matrix that is generated using the average of the previously visited solution's target function values. Using many TSPs from the literature, these two amended versions of SA were checked and the results of the tests were compared to the traditional SA technique.

A diversified model of TSP was proposed with hybrid parallel SA to solve transport system problems [8]. Each city has various transport vehicles in the proposed model and the cost of driving through each city depends on the type of vehicle being transported. The objective is to establish, within a restricted budget, an optimal sequence of cities visited with minimum travelling time by means of accessible transport vehicles.

SA algorithm was introduced with list-based cooling program for solving TSP [9]. To monitor the reduce in temperature parameters, this list-based geometric cooling program with variable coefficient of cooling was used to control. The Metropolis acceptance criteria utilizes the

maximum temperature list to determine whether to approve a candidate solution once a temperature list has been created. This list-based performance of the cooling schedule on a large range of parameter values is seen to be stable.

2.3 Advantages of SA

The advantages of SA are its easy implementation and its ability to find a global optimum even after a local minimum has been identified, as it accepts solutions which are worse than the best candidate [10]. In addition, with a relatively low number of iterations, it can provide acceptable performance, which makes it ideal for real-time control. It has been shown that if enough randomness is used in conjunction with very slow cooling, SA would converge to its global optimality. Essentially, as a Markov chain, SA is a search algorithm which converges under suitable conditions [11].

SA is a robust method for dealing with highly nonlinear models, chaotic and noisy data with several restrictions. Its main benefits over other local search methods are its adaptability and capacity to approach global optimality. The approach is particularly versatile since it does not rely on any model's restricting features. A given optimization algorithm can be tuned to improve its efficiency for any relatively difficult nonlinear or stochastic system, and since it takes time and effort to become familiar with a given code, the ability to tune a given algorithm for use in more than one problem should be considered an essential feature of an algorithm [12].

It can consider the non-functions, as well as multi-local optima functions. In a number of disciplines, it is also capable for parallel implementation, a very strong and significant method. With all mention advantages of SA, this research proposes to solve TSP using SA method among others metaheuristic methods.

3 Research Methodology

3.1 Assumption of TSP

The assumptions for this study are as follows:

i. The direct link between every pair of nodes or cities must be incomplete graph where the nodes *N* and arcs *A* is connected.
ii. The number of visits at each city is restricted to exactly once.
iii. The cost matrix is symmetric because from each direction, the distance between two cities is the same where $c_{ij} = c_{ji}$.
iv. The salesman must return to the node or city where he began his tour, which is commonly referred to as depot.

3.2 Notation of TSP

The TSP can be represented by a complete directed graph $G = (N, A)$. Below shows the notation in TSP that will be used in this study.

N	Set of nodes to be visited
n	Total number of nodes
A	The set of arcs connecting the nodes
D	Cost or distance matrix associated with each arc $(i, j) \in A$
c_{ij}	Cost of traveling from node i to node j

X	Set of decision variables
x_{ij}	Decision variables
d_{ij}	Distance from node i to node j
z	Objective function

The collection of places to visit is specified as $N = \{1,2, \dots, n\}$ where n denotes the total number of nodes or the size of the TSP data. The set of nodes that connects them is defined as $A = \{(i,j) : i,j \in N, i \neq j\}$, where the pair (i,j) indicates the arc between node i and node j. A set of decision variables is defined as $X = \{x_{ij} : i,j \in N, i \neq j\}$ where $x_{ij} = 1$ if the salesman travel from node i to node j and 0 otherwise. The cost matrix is defined as $D = \{c_{ij} : i,j \in N, i \neq j\}$ where c_{ij} represent the cost of traveling from node i to node j.

3.3 Model Formulation of TSP

Minimize

$$z = \sum_{i=1}^{n}\sum_{j=1}^{n} c_{ij}x_{ij} \quad (1)$$

Subject to:

$$\sum_{i=1}^{n} x_{ij} = 1, \forall\, j \in N \quad (2)$$

$$\sum_{j=1}^{n} x_{ij} = 1, \forall\, i \in N \quad (3)$$

$$x_{ij} \in \{0,1\},\ \forall\, i,j \in N \quad (4)$$

$$x_{i,i} = 0, \forall\, i \in N \quad (5)$$

From the formulation above, the objective function (1) will minimize the total cost along all the arcs that is used to complete the tour. Constraint (2), constraint (3) and constraint (4) are the standard assignment constraints. Constraint (5) will fix the diagonal in a square matrix of the binary variables equal to zero.

3.3 Simulated Annealing Flowchart

The flowchart below is the process of SA in solving TSP with the objective to minimize the cost travelled by the salesman.

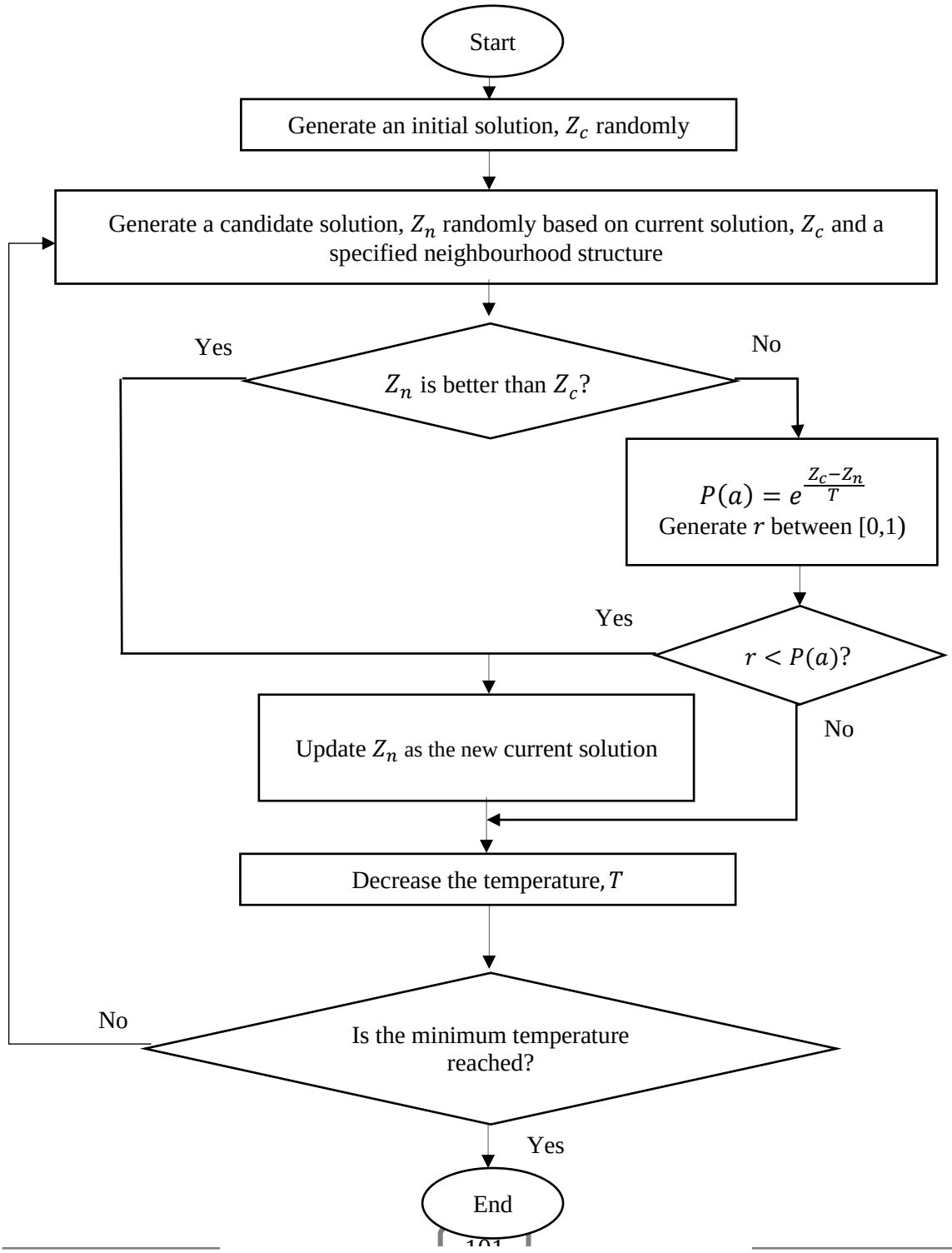

Figure 1: Simulated Annealing Flowchart

4 Result and Discussion

4.1 Parameter Estimation

A few parameters have been experimented to set the algorithm of SA. In this paper, the parameters of these algorithm are selected after thorough testing. There are two things that need to be considered in implementing SA to solve TSP, which are initial temperature and cooling rate. The initial temperature is set to be 100 and the cooling rate is set to be 0.80 throughout the execution of 6 instances which are Att48, Berlin52, St70, Pr76, Eil101 and KroA200.

4.2 Best Route of TSP

The data for this research are taken from TSPLIB which is the benchmark of TSP and solve by using MATLAB software. Six instances' data are used to compare the effectiveness and performance of the algorithms. Att48 is a set of 48 cities in United State capitals, Berlin52 is a set of 52 locations in Berlin, St70 is a set 0f 70-city problem, Pr76 is a set of 76-city problem, Eil101 is a set of 101-city problem and KroA200 is a set of 200-city problem where all datasets are taken from TSPLIB benchmark.

All datasets were run 20 times for each instance and the results presented include the best, worst and average solutions. Figure 2 shows the path of the best route for the 6 instances. All instances have a mixed of random and cluster distribution as the positions have a combination of concentrated which close to each other and random position which is far to each other.

In terms of time, the instance with a smaller number of cities tends to have a shorter running time to get an optimal solution. In this research, Att48 that consists of 48 cities does not take a longer time than the instance with a larger number of cities like KroA200, which consists of 200 cities. The smaller number of cities does not require many movements of the swapping between the cities to find the optimal cost, thus requiring less running time. Unlike the instance with a larger number of cities, it tends to have many swapping numbers between cities, requiring it to have a longer running time.

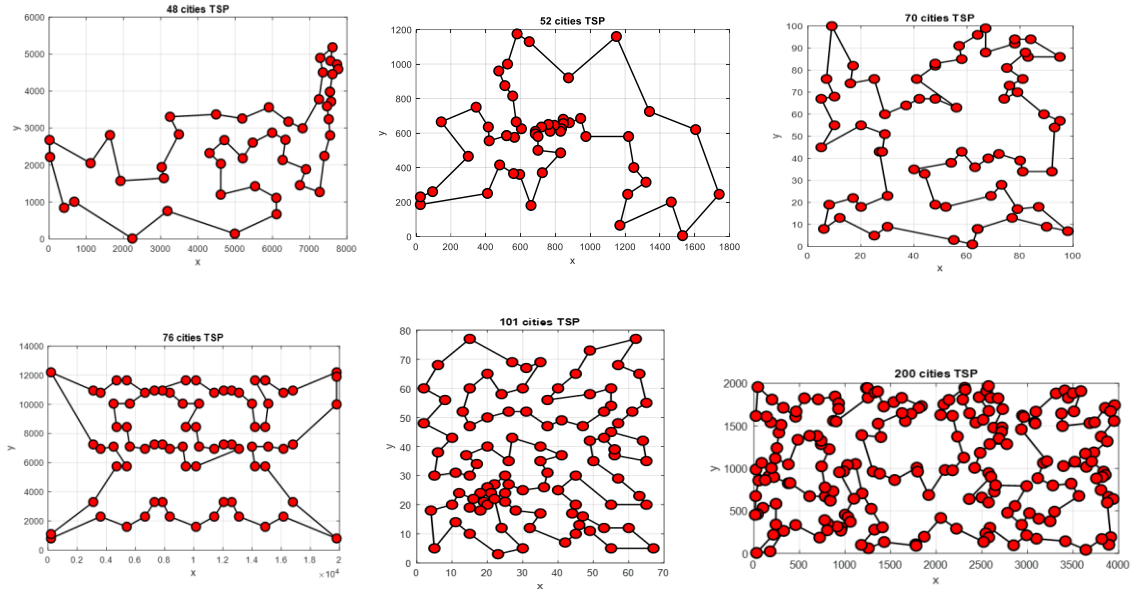

Figure 2: Best Route of TSP

4.3 Comparison of Algorithms

A comparison of the algorithm is being studied among 7 algorithms which are TS, GA, PSO, ACO, hybrid of PSO-ACO, hybrid of GA-PSO-ACO and SA. The comparison to be performed here will take into account the cost of solutions found by the algorithms, the difference and the error. The results of TS, GA, PSO, ACO, PSO-ACO and GA-PSO-ACO for Att48, Berlin52, St70, Pr76, Eil101 and KroA200 are directly taken from Deng et al. [13] and being compared with the result obtained by using SA algorithm.

Table 1, Table 2, Table 3, Table 4, Table 5 and Table 6 show the comparison for each instance. From the table, scale indicates the number of cities which include the depot and the optimal tour length, as indicated in TSBLIB, is the optimal solution. The term "best" refers to the algorithm's best answer, the poorest answer obtained by each algorithm is denoted by the word "worst" and the average value of the whole run solutions is denoted by average. Difference denotes the result of subtracting between best and optimal solution and error denotes the percent difference of the solution.

Table 1: Comparison for Att48

Algorithm	Scale	Optimal Solution	Best	Worst	Average	Difference	Error (%)
TS	48	33,522	34,198	35,886	34,978	676	2.017
GA	48	33,522	34,572	36,299	35,002	1,050	3.132
PSO	48	33,522	34,759	37,672	36,179	1,237	3.690
ACO	48	33,522	34,357	35,197	34,460	835	2.491
PSO-ACO	48	33,522	33,641	34,730	33,956	119	0.355
GA-PSO-ACO	48	33,522	33,524	34,164	33,662	2	0.006
SA	48	33,522	**33,523.71**	34,113.58	33,658.24	1.71	0.005

Table 2: Comparison for Berlin52

Algorithm	Scale	Optimal Solution	Best	Worst	Average	Difference	Error (%)
TS	52	7,542	7,976.84	8,286.68	8,014.60	434.84	5.766
GA	52	7,542	8,201.17	8,443.02	8,376.55	659.17	8.740
PSO	52	7,542	8,197.79	8,589.31	8,319.51	655.79	8.695
ACO	52	7,542	7,647.55	7,780.57	7,732.31	105.55	1.399
PSO-ACO	52	7,542	7,568.54	7,618.31	7,586.42	26.54	0.352
GA-PSO-ACO	52	7,542	7,544.37	7,544.37	7,545.37	2.37	0.031
SA	52	7,542	**7,544.37**	7,549.71	7,546.31	2.37	0.031

Table 3: Comparison for St70

Algorithm	Scale	Optimal Solution	Best	Worst	Average	Difference	Error (%)
TS	70	675	702.27	738.45	718.56	27.27	4.040
GA	70	675	715.43	744.31	731.72	40.43	5.990
PSO	70	675	720.41	753.29	741.09	45.41	6.727
ACO	70	675	697.76	716.83	705.58	22.76	3.372
PSO-ACO	70	675	684.16	710.47	698.75	9.16	1.357
GA-PSO-ACO	70	675	679.60	704.25	694.60	4.6	0.681
SA	70	675	**679.32**	706.98	694.17	4.32	0.640

Table 4: Comparison for Pr76

Algorithm	Scale	Optimal Solution	Best	Worst	Average	Difference	Error (%)
TS	76	108,159	110,941.00	130,637.00	122,104.00	2,782.00	2.572
GA	76	108,159	115,329.00	124,851.00	120,245.00	7,170.00	6.629
PSO	76	108,159	118,038.00	126,583.00	122,735.00	9,879.00	9.134
ACO	76	108,159	110,517.00	120,922.00	114,964.00	2,358.00	2.180
PSO-ACO	76	108,159	109,244.00	113,120.00	110,162.00	1,085.00	1.003
GA-PSO-ACO	76	108,159	109,206.00	112,443.00	110,023.00	1,047.00	0.968
SA	76	108,159	**108,159.44**	111,599.33	109,993.12	0.44	0.000

Table 5: Comparison for Eil101

Algorithm	Scale	Optimal Solution	Best	Worst	Average	Difference	Error (%)
TS	101	629	667.43	709.11	685.49	38.43	6.110
GA	101	629	682.37	745.33	706.25	53.37	8.485
PSO	101	629	687.32	779.11	731.58	58.32	9.272
ACO	101	629	649.87	695.18	664.07	20.87	3.318
PSO-ACO	101	629	637.65	674.07	651.36	8.65	1.375
GA-PSO-ACO	101	629	633.07	641.17	637.93	4.07	0.647
SA	101	629	**631.05**	641.60	637.84	2.05	0.326

Table 6: Comparison for KroA200

Algorithm	Scale	Optimal Solution	Best	Worst	Average	Difference	Error (%)
TS	200	29,368	31,289	33,438	32219	1,921	6.541
GA	200	29,368	32,261	34,572	33158	2,893	9.851
PSO	200	29,368	32,350	34,526	33132	2,982	10.154
ACO	200	29,368	31,669	33,839	32434	2,301	7.835
PSO-ACO	200	29,368	30,190	33,626	31,927	822	2.799
GA-PSO-ACO	200	29,368	29,731	33,228	31,015	363	1.236
SA	200	29,368	**29,492.69**	31,934.32	30,973.43	125	0.425

As can be seen in Table 1, Table 2, Table 3, Table 4, Table 5 and Table 6, the SA outperformed the other algorithms in all test instances under investigation. Besides, the SA obtained the nearest-optimal solution to the TSP instances compared to any other algorithms. In terms of the average results obtained by each algorithm, the SA still has the best performance for Att48, St70, Pr76, Eil101 and KroA200 datasets. In comparison to the error, SA produces the smallest error compared to other 6 algorithms. Specifically, SA algorithm gives a better improvement on the results obtained for the 6 datasets.

5 Conclusion

In conclusion, the objectives of this research are successfully achieved. The objectives are to implement the SA algorithm in solving and finding the best route that can minimize the cost for TSP and to compare the performance of SA with other metaheuristic algorithms in solving TSP.

SA is performed on 6 TSP benchmark instances from TSPLIB with cities scale of 48, 52, 70, 76, 101 and 200 where the algorithms were coded in MATLAB language. Experimental result and analysis also have been discussed in this research. In order to test the effectiveness of the SA, a comparison of the algorithm is being studied among 6 other metaheuristic algorithms which are TS, GA, PSO, ACO, hybrid of PSO-ACO, hybrid of GA-PSO-ACO. According to the experiment results obtained, SA can greatly improve the computing efficiency for solving TSP and outperforms the other 6 metaheuristics. SA has obtained the smallest best cost and the nearest optimal solution to the TSP instances compared to other algorithms. To conclude, we can see that the order of solution quality for the 6 algorithms is SA > GA-PSO-ACO > PSO-ACO > ACO > TS > GA > PSO.

As recommendation for future work, it is hoped that further study can be carried out to consider merging SA algorithm with any other metaheuristic algorithms, given the fact that SA is so sophisticated, easy to implement and has many advantages. The hybrid algorithm might give interesting results and might produce a better solution. Besides, researcher might want to consider more constraints in solving TSP or consider on solving other types of TSP which are asymmetric TSP (aTSP) and multi-TSP (mTSP). Researcher also might want to conduct a research on solving the applications of TSP which include vehicle routing, job scheduling and urban transportation problems, logistics, genome sequencing, scan chains, drilling problems and data clustering. In

addition, researcher can include a fair comparison on parameters for other algorithms in solving TSP.

References

[1] Gonsalves, T. and Shiozaki, T. (2015) 'Solving capacity problems as symmetric travelling salesman problems', *International Journal of Artificial Intelligence & Applications*. 6(2): 53.

[2] Mavrovouniotis, M., Müller, F. M. and Yang, S. (2016) 'Ant colony optimization with local search for dynamic traveling salesman problems', *IEEE Transactions on Cybernetics*. 47(7): 1743-1756.

[3] Mansouri, N. and Javidi, M. M. (2020) 'A review of data replication based on meta-heuristics approach in cloud computing and data grid', *Soft Computing*. 1-28.

[4] Chauhan, C., Gupta, R. and Pathak, K. (2012) 'Survey of methods of solving TSP along with its implementation using dynamic programming approach', *International Journal of Computer Applications*. 52(4).

[5] Sghaier, S. B., Guedria, N. B. and Mraihi, R. (2013) 'Solving school bus routing problem with genetic algorithm', *International Conference on Advanced Logistics and Transport*. 7-12.

[6] Du, K. L. and Swamy, M. N. S. (2016) *Simulated Annealing*. in Search and Optimization by Metaheuristics. Birkhäuser, Cham. 29-36.

[7] Bayram, H. and Şahin, R. (2013) 'A new simulated annealing approach for travelling salesman problem', *Mathematical and Computational Applications*. 18(3): 313-322.

[8] Azimi, P., Rooeinfar, R. and Pourvaziri, H. (2014) 'A new hybrid parallel simulated annealing algorithm for travelling salesman problem with multiple transporters', *Journal of Optimization in Industrial Engineering*. 7(15): 1-13.

[9] Zhan, S. H., Lin, J., Zhang, Z. J. and Zhong, Y. W. (2016) 'List-based simulated annealing algorithm for traveling salesman problem', *Computational Intelligence and Neuroscience.*

[10] Martinez, C. M. and Cao, D. (2018) *iHorizon-Enabled Energy Management for Electrified Vehicles*. Butterworth-Heinemann.

[11]Yang, X. S. (2020) 'Nature-inspired optimization algorithms: challenges and open problems', *Journal of Computational Science*. 46: 101-104.

[12] Samora, I., Franca, M. J., Schleiss, A. J. and Ramos, H. M. (2016) 'Simulated annealing in optimization of energy production in a water supply network', *Water Resources Management*. *30*(4): 1533-1547.

[13] Deng, W., Chen, R., He, B., Liu, Y., Yin, L. and Guo, J. (2012) 'A novel two-stage hybrid swarm intelligence optimization algorithm and application', *Soft Computing*. 16(10): 1707-1722.