Date of publication MAR-29, 2023, date of current version SEP-20, 2022.
www.computingonline.net / computing@computingonline.net

Print ISSN 1727-6209
Online ISSN 2312-5381
DOI 10.47839/ijc.22.1.2878

Simulated Annealing – 2 Opt Algorithm for Solving Traveling Salesman Problem

P. H. Gunawan[1], Iryanto[2]
[1]School of Computing, Telkom University, Indonesia (phgunawan@telkomuniversity.ac.id)
[2]Informatics Department, Politeknik Negeri Indramayu, Indonesia (iryanto@polindra.ac.id)

Corresponding author: Iryanto (e-mail: iryanto@polindra.ac.id).

ABSTRACT The purpose of this article is to elaborate performance of the hybrid model of Simulated Annealing (SA) and 2 Opt algorithm for solving the traveling salesman problem (TSP). The SA algorithm used in this article is based on the outer and inner loop SA algorithm. The hybrid algorithm has promising results in solving small and medium-scale symmetric traveling salesman problem benchmark tests taken from the TSPLIB reference. Results of the optimal solution and standard deviation indicate that the hybrid algorithm shows good performance in terms of reliability and stability in finding the optimal solution from the TSP benchmark case. Values of average error and standard deviation for all simulations in the medium scale are 0.0267 and 644.12, respectively. Moreover, in some cases namely KroB100, Pr107, and Pr144, the hybrid algorithm finds a better solution compared with the best-known solution mentioned in the reference. Further, the hybrid algorithm is 1.207 – 5.692 times faster than the pure outer and inner loop-based SA algorithm. Additionally, the results show that the hybrid algorithm outperforms other hybrid algorithms such as SA – nearest neighbor (NN) and NN – 2 Opt.

KEYWORDS simulated annealing; 2 opt algorithm; traveling salesman problem; hybrid simulated annealing – 2 opt algorithm.

I. INTRODUCTION

OPTIMIZATION problems have become one of the most active research areas to date. The technique of optimization is a branch of science that involves searching for parameter values in solving a particular problem. The optimization is designed to find a solution for a predetermined objective function through an iterative process toward the optimal value. Here, the mathematical representation of the objective function is defined with a clear constraint that depends on the problems [1]. One of the well-known problems in optimization is called the traveling salesman problem (TSP).

The TSP is an active research topic in optimization problems. This problem aims to find the shortest route with the constraint that all cities are visited exactly once and return to the initial place of departure. Finding the shortest route has its difficulties since the solution involves a combinatorial procedure. Therefore, TSP is also known as the NP-hard problem in combinatorial optimization [2]. There are two types of TSP problems, e.g., symmetric and asymmetric TSPs that are distinguished by the distance between cities. The characteristic of symmetric TSP is given by the distance between two cities in a constant. Meanwhile, the characteristic of asymmetric TSP is the distance between two cities that varies depending on where we start [3]. Various approaches have been studied and used by researchers to solve TSP. One of them is called the simulated annealing (SA) algorithm.

The SA algorithm was introduced to solve the combinatorial optimization problem by Kirkpatrick et al. in 1983. The main idea of this algorithm is that it is possible to accept a less than optimal solution that depends on a certain probability function. Using this algorithm will certainly be able to produce a global optimum value. However, the use of random numbers in the calculation process affects the results to be obtained. These results can be in the form of optimal or non-optimal solutions [4].

The SA algorithm is one of the efficient methods to handle optimization problems both continuously and discretely. This SA algorithm was originally formed from the problem of simulating the metal cooling schedule. The SA algorithm has the advantage of not requiring an initial solution close to the solution. Despite many benefits, this algorithm has drawbacks such as poor performance and slow convergence when applied to complex TSP [5]. Various modifications were done to improve the performance of the SA algorithm. Several studies about the modification of the SA algorithm can be found in some references [4-17].

The following are some modifications made to the SA algorithm which are proven to improve the performance of the algorithm in solving TSP:

- The use of a combination of outer and inner loops in the SA algorithm [4].
- The use of four vertices and three lines of inequality to

find the optimal Hamiltonian circuit [5].

- The use of a list-based cooling scheme to control the temperature parameters [6].
- The use of adaptive temperature control to reduce the temperature adaptively based on a certain temperature function [7].

The combination of the SA algorithm with other metaheuristic algorithms in solving TSP is also used to overcome the shortcomings of each algorithm by their advantages. Some of the combinations of these algorithms include the SA with water flow-like algorithm (WFA) [8], the SA with symbiotic organisms search optimization algorithm (SOS) [9], the SA with ant colony optimization (ACO) [10], the SA with genetic algorithm (GA) [11, 12], the SA with gene expression programming (GEP) [13], and the SA with particle swarm optimization (PSO) [14].

In addition to combining with other metaheuristic algorithms, other search algorithms are also used to overcome the shortcomings of the SA algorithm in obtaining the optimal TSP solution. For example, by adding a tabu search algorithm [15], the use of an adaptive simulated annealing algorithm with a tabu search and 2-opt algorithm [16] and a greedy search algorithm [17] has also been shown to improve the performance of the SA algorithm.

According to [4], the combination of outer and inner loops in the SA algorithm can simplify the algorithm in finding local solutions. Continuing the research, started in [4], by adding process loops at one temperature before the temperature reduction scheme is carried out and incorporating 2-opt local searches is the focus of this article. In addition, to see the performance of the proposed algorithm, the search for solutions from 43 TSP benchmark tests with several cities from 16 – 1060 obtained from TSPLIB [18] will be simulated.

II. RESEARCH METHOD

A. SIMULATED ANEALING (SA)

Kirkpatrick, Gelatt, and Vecchi introduced the simulated annealing algorithm in combinatorial optimization problems in 1983. This algorithm was inspired by the physical annealing of solid [19-22]. In general, there are two processes, namely (1) increasing the temperature to the maximum temperature so that the solid melts; (2) the decrease in temperature follows a certain temperature reduction scheme until it reaches the ground state of the solid [20, 21].

According to references, the SA algorithm is one of the simplest and most popular methods for dealing with difficult global optimization problems [20]. In addition, the SA algorithm can improve the exploitation of finding solutions without being trapped in the local optimal [8]. The possibility of obtaining the global optimum solution is inseparable from the cooling scheme used. The algorithm will be stuck in the optimal local solution if the temperature drops rapidly. Meanwhile, if the temperature is controlled properly, the algorithm will get a global optimal [22]. The performance of the SA algorithm is judged to be lacking in its poor performance and slow convergence when applied to complex TSP [5]. The main drawback of this algorithm is that it does not consider the system's state when searching for the optimal solution. Thus, it is difficult to predict the convergence of the system with the SA algorithm [22].

One way to improve the algorithm performance is to use the interaction between the outer and inner loops of the algorithm. The use of the results from the inner loop as the outer loop input is proven to improve the performance of the SA algorithm in solving small-scale symmetric TSP problems with the number of cities from 9 – 225. The use of these interactions gives good results compared to several other algorithms such as ACO (ant colony optimization), PSO (particle swarm optimization), SFLA (shuffled frog leaping algorithms), GA (genetic algorithm), BHA (black hole algorithm), STA (state transition algorithm) in completing the TSP symmetric benchmark test [4].

However, using interaction to get the optimal solution still requires much time, especially for many cities. In addition, the relative error rate compared to the best-known solution (BKS) of each of these benchmark tests tends to increase with the increase in the number of cities. In this article, the reduction of computational time in finding the optimal TSP solution and decreasing the error rate is the goal of this research. In order to achieve this goal, a two-optimization (2-opt) local search algorithm will be combined.

B. 2 OPT ALGORITHM

Croes introduced a two-optimization (2-opt) algorithm to solve the TSP case [23]. This algorithm is a simple local search algorithm. In order to decrease the total length of the route/tour, an edge swap in the tour is performed [2]. In this case, two crossing routes are identified and rearranged so that they do not cross each other. As an illustration of this algorithm, it can be seen in Fig. 1.

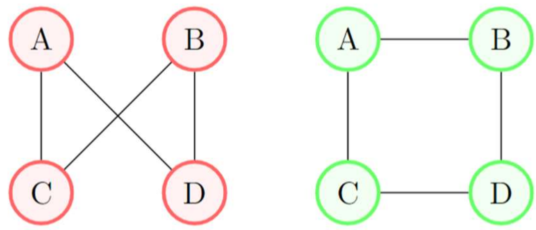

Figure 1. The illustration for the 2-opt algorithm.

Suppose there is a route that passes through cities A, B, C and D as shown in Fig. 1 (left). In this case, there are two paths that cross each other (intersect each other), namely path A to D (AD) and path C to B (CB). To minimize the overall route distance, a new path will be formed with the following conditions:

$$d(A,B) + d(C,D) < d(A,D) + d(B,C), \quad (1)$$

where $d(\cdot,\cdot)$is the distance metric. If the conditions in (1) are met, then the AD and CB paths are replaced with AB and CD paths so that the current path does not contain cross paths, as shown in Fig. 1 (right).

C. HYBRID SA – 2 OPT ALGORITHM

The combination of the SA algorithm with the 2-opt algorithm is carried out to improve the SA algorithm performance. Merging this algorithm is done directly. First, the solution of the TSP case will be searched using the SA algorithm then the results of the algorithm are optimized using the 2-opt algorithm. It should be noted that this SA algorithm uses the interaction between the inner and outer loops as described in [4]. Moreover, the addition of a loop for temperature is also carried out. Here is the hybrid algorithm:

Algorithm 1

SA – 2 Opt procedures

```
Set the initial value T, S_0 using (2), E_0,
S_opt and E_opt
For 1 to n
  Set m = 0
  While (m ≤ number of city or T ≥ 0.01) do
     Update State S, Calculate Energy E
     using (3)
     Optimize the state using 2-opt
     algorithm
     Calculate ΔE = E − E_0, ω ∈ [0,1], p using
     (1)
     If ΔE < 0 then
           Set S_0 = S, E_0 = E
           If E < E_opt then S_opt = S_0, E_opt = E_0
                 Else go to Step 11
     Else
           If p > ω then go to Step 5
           Else go to Step 11
  End while
  If satisfying Inner Loop Termination
  Criteria, then
  Do cooling schedule T = T × r
  Go to Step 3
End For
```

The initial state of the SA algorithm is generated according to the following conditions:

$$S_0 = \{1, 2, 3, 4, 5 \dots, M\}, \quad (2)$$

where M represents the number of cities of TSP cases. The calculation of energy (distance) follows the following equation

$$E = \sum_{i=1}^{N-1} d_i + d_N, \quad (3)$$

where d represents the distance between two cities in the state. At the same time, d_N is the distance from the last to the first city in that state. The acceptance criterion of a worse solution depends on the value of p, which is defined as:

$$p(\Delta E, T) = \begin{cases} e^{\frac{-\Delta E}{T}} & \Delta E > 0 \\ 1 & \Delta E \leq 0 \end{cases}. \quad (4)$$

III. RESULTS AND DISCUSSIONS

Here, a numerical simulation experiment is performed to see the performance of the modified SA algorithm. The first simulation was carried out to see the performance of the algorithm compared to the SA algorithm with the interaction of inner and outer loops as described in [4]. Meanwhile, the next experiment was conducted to see the performance of the algorithm in solving other symmetric TSP cases. All of these numerical experiments were conducted in C++ programming language and run using a PC with Windows 10 pro-64-bit OS, Intel® Core™ i7-8550U CPU @ 1.80 GHz processor, and 16 GB RAM. The values of the parameters used are the cooling rate, r = 0.9, initial temperature, T = 1000, inner loop (m) = number of cities, outer loop (n) = 20, inner loop stopping criteria: temperature < 0.01 or iteration > number of cities.

A. COMPARISON WITH PURE OUTER AND INNER LOOP-BASED SA ALGORITHM

In this case, a numerical experiment was conducted to test the performance of the SA – 2 Opt algorithm in solving symmetric TSP cases consisting of 16 – 225 cities. Comparisons were made with the results of the SA algorithm presented in reference [4]. The results of this comparison are presented in Table 1.

In the article [4], there are two cases of symmetric TSPs, namely cases taken from the benchmark data available in reference [18] and the case of the square grid TSP. SG denotes this TSP square grid case in Table 1. It can be seen in the table that the performance of the SA – 2 opt algorithm is better than that of the SA algorithm in the article [4] from the aspect of BS (best solution), Ave (average solution), and WS (worst solution), indicated in bold letters. The SA – 2 Opt algorithm can find optimal solutions in 13 of the 21 cases mentioned. While the SA algorithm [4] only found 6 of the 21 cases. In addition, the SA – 2 Opt algorithm can find a more optimal solution than BKS (the best-known solution) in the case of Gr96. The BKS of those cases was 514 [24]. While the solution generated by the algorithm SA – 2opt is 510.8863. As an illustration of the optimal route for the Gr96 case, both the results of the SA [4], BKS [18], and SA – 2 Opt algorithms are presented in Fig. 2. In addition, comparison plots for several other cases are also presented in Fig. 3 and Fig. 4.

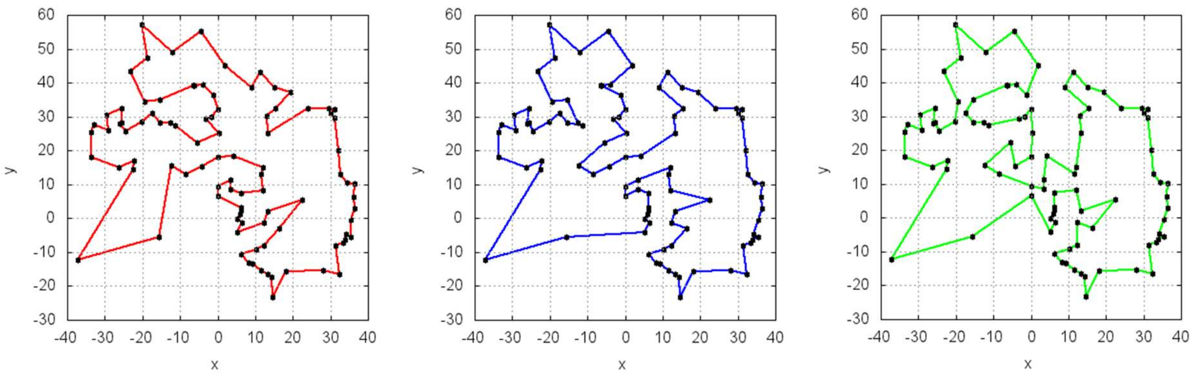

Figure 2. Results of the Gr96 TSP solution. BKS [18] (left), proposed algorithm (middle) and SA [4] (right).

Table 1. Comparison results of hybrid SA – 2 opt algorithm and SA algorithm in [4]

Case	BKS	BS [4]	Ave [4]	WS [4]	Ave.Time [4]	BS	Ave	WS	Ave.Time	Speed Up
Ulysess16	73.9876	**73.9876**	74.2239	74.4602	0.1665	**73.9876**	**73.9876**	**73.9876**	**0.138**	1.207
Ulysess22	75.3097	**75.3097**	76.2203	76.8119	0.3106	**75.3097**	**75.3097**	**75.3097**	**0.144**	2.157
Att48	33522	33882.48	34766.3	34846.67	0.5306	**33523.7**	**33523.7**	**33523.7**	**0.181**	2.931
Eil51	426	430.89	444.41	444.74	0.6051	**428.88**	**429.13**	**433.93**	**0.167**	3.623
Berlin52	7542	**7544.366**	7845.61	8341.865	0.6172	**7544.366**	**7544.37**	**7544.37**	**0.191**	3.231
St70	675	697.8861	708.807	727.8492	1.0369	**677.1096**	**677.502**	**684.953**	**0.241**	4.302
Eil76	538	563.6019	575.556	594.2049	1.2333	**545.3876**	**547.965**	**554.958**	**0.401**	3.076
Gr96	514	539.9611	550.015	558.2102	1.8726	**510.8863**	**511.354**	**512.924**	**0.329**	5.692
KroA100	21282	21632.56	22956.7	24181.94	1.8989	**21285.44**	**21289.1**	**21357.8**	**0.434**	4.375
Eil101	629	673.4284	688.984	708.6054	1.9990	**643.9111**	**647.613**	**651.378**	**0.475**	4.208
Ch130	6110	6356.304	6443.35	6657.21	3.2852	**6110.722**	**6140.74**	**6210.84**	**0.675**	4.867
SG6	36	36	36.7249	36.8284	0.3551	**36**	**36**	**36**	**0.1771**	2.005
SG7	49.4142	49.4142	50.5188	51.0711	0.4053	**49.4142**	**49.4142**	**49.4142**	**0.1958**	2.070
SG8	64	64	64.9527	66.4853	0.9037	**64**	**64.0828**	**64.8284**	**0.2314**	3.905
SG9	81.4142	81.4142	82.5740	86.3848	1.2858	**81.4142**	**81.4556**	**82.2426**	**0.2878**	4.468
SG10	100	100.8284	103.038	104.1421	1.4590	**100**	**100.124**	**100.828**	**0.4061**	3.593
SG11	121.4142	123.0711	124.562	125.5563	2.1928	**121.4142**	**121.828**	**122.243**	**0.5673**	3.865
SG12	144	147.3137	150.904	152.2843	3.6069	**144**	**144**	**144**	**0.6802**	5.303
SG13	169.4142	174.3848	176.594	177.6985	4.284	**169.4142**	**169.414**	**169.414**	**1.2243**	3.499
SG14	196	201.799	206.217	208.4264	5.6313	**196**	**196**	**196**	**1.1775**	4.782
SG15	225.4142	232.8701	235.713	237.8406	8.9315	**226.2426**	**226.698**	**228.728**	**1.8346**	4.868

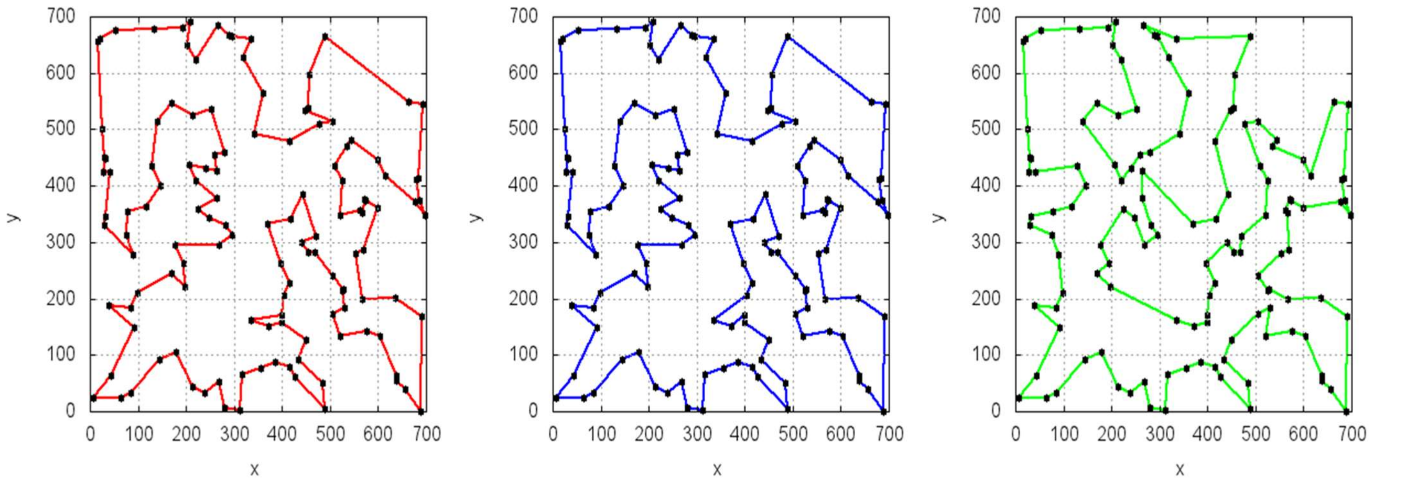

Figure 3. Results of the Ch130 TSP solution. BKS [18] (left), proposed algorithm (middle) and SA [4] (right).

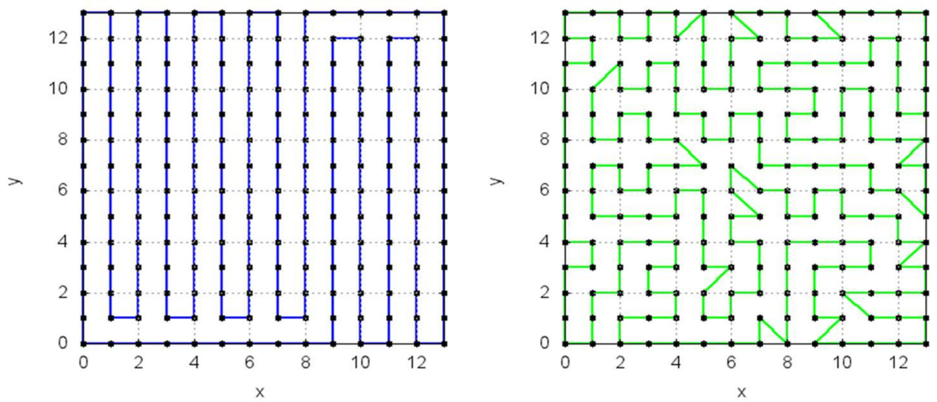

Figure 4. Results of TSP SG14 solution. The proposed algorithm (left) and SA [4] (right)

The average simulation time performance for each inner loop can be seen in Table 1. It can be seen in the table that the average computation time of the SA – 2 Opt algorithm is better than that of the SA algorithm [4] for all the cases mentioned. The results obtained for these cases show that the SA – 2 Opt algorithm has a faster computation time of 1.207 – 5.692 times than the SA algorithm [4]. Given the better results of BS, Ave, and WS, the addition of the 2 Opt algorithm can improve algorithm performance in finding optimal solutions and increase computational time.

Fig. 3 and Fig. 4 show comparison results of the proposed algorithm and SA algorithm in [4] when solving TSP with 130 cities and 196 cities, respectively. It is clearly seen that the proposed algorithm outperforms the SA algorithm in [4] for both cases. Further, the proposed algorithm gets the same route as the best-known solution mentioned in [18] (TSP with 130 cities) and [4] (TSP with 196 cities). These findings state that the proposed algorithm has good performance in solving TSP.

B. COMPARISON WITH OTHER HYBRID ALGORITHMS

To see further performance of the proposed algorithm, a comparison with other hybrid algorithms is carried out. The other algorithms are the hybrid simulated annealing – nearest neighbor (SA – NN) algorithm and the nearest neighbor – 2 optimal (NN – 2 Opt) algorithm. The numerical experiment was conducted in solving symmetric TSP cases consisting of 16 – 299 cities. Results of the hybrid algorithms in solving the benchmark cases are presented in Table 2.

Table 2 presents the numerical simulation results for the TSP cases in terms of the best solution (BS), average solution (Ave), and worst solution (WS). The simulation results are compared with the BKS in the reference [18] and the other two hybrid algorithms. The number of cities in these cases varies from 16 to 299. This selection was made to see the algorithm performance in solving TSP with a few cities (TSP size) to more (small-medium scale TSP). The solved TSP cases and their simulation results are presented in Table. 2.

From Table 2, it can be said that the SA – 2 Opt algorithm has a good match in finding the optimal solution from the BKS mentioned in reference [18]. Moreover, the proposed algorithm gets better results than the other two hybrid algorithms. This is indicated in bold writing presented in Table 2. In addition, the SA – 2 Opt algorithm also obtains a better optimal solution than the BKS written in [18] for the cases of KroB100, Pr107, and Pr144. The plot of our result of the cases can be seen in Fig. 5.

C. PERFORMANCE OF THE HYBRID ALGORITHM IN SOLVING MEDIUM SCALE TSP

The next numerical experiments are carried out to see further performance of the proposed method in solving TSP cases with more cities. The number of cities in these cases varies from 315 to 1060. This selection was made to see the algorithm performance in solving TSP with a medium-scale TSP given in [18]. The simulation results are compared with the BKS in the reference. The solved TSP cases and their simulation results are presented in Table. 3.

Table 3 shows the numerical simulation results for the TSP cases in terms of the best solution (BS), average solution (Ave), worst solution (WS), standard deviation (STD. Dev), and average computation time for one inner loop (Ave.Time). From the table, it can be said that the SA – 2 Opt algorithm has a good match in finding the optimal solution from the BKS mentioned in reference [18]. This is indicated by the relative error values as presented in Table 3.

The standard deviation is used to see the stability and reliability of the algorithm in finding the optimal solution to the problem. A smaller value indicates that the algorithm is more stable and reliable in finding the optimal solution [25]. The standard deviation of the SA – 2 Opt algorithm is presented in Table 3 in the STD column. Dev. The standard deviation value of the SA – 2 Opt algorithm varies from 79.89 – 2085.7. From Table 3, it can be said that this algorithm has a good performance in terms of reliability and stability in finding the optimal solution from the TSP benchmark cases.

The relative error for each case can be seen in Table 3. The BE column represents the best error value, AveE represents the average error value, and WE represents the worst error value obtained using Eq. 5. The BE values varied from 0.005592 to 0.04137, the AveE values varied from 0.010012 to 0.057179, and the WE values varied from 0.017458 to 0.092546. It is indicated that the proposed method has promising results in solving the symmetric TSP.

$$BE = \frac{BS - BKS}{BKS}, \quad WE = \frac{WS - BKS}{BKS}, \quad AE = \frac{Ave - BKS}{BKS}. \tag{5}$$

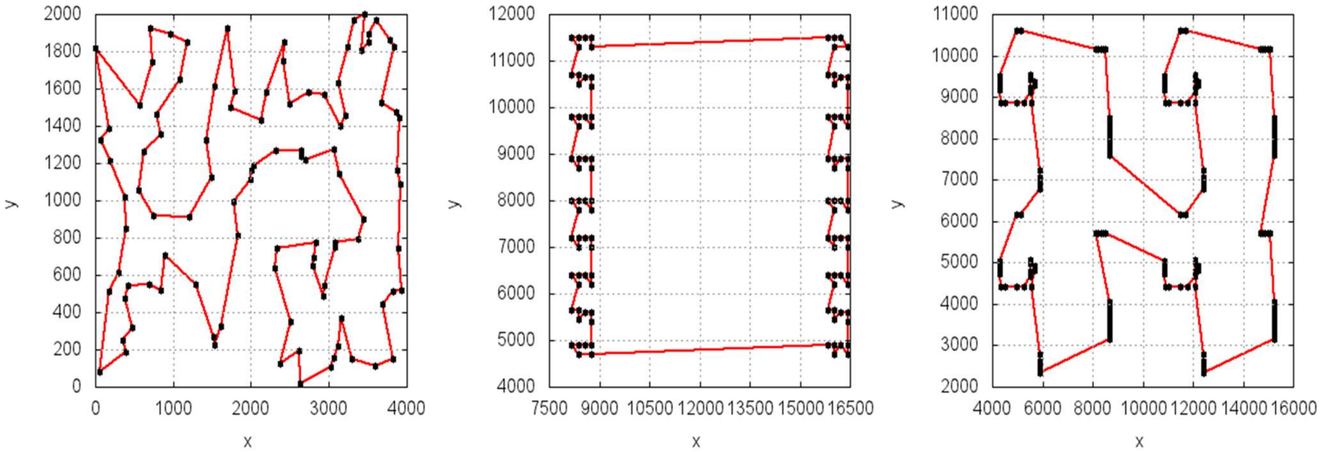

Figure 5. Optimal solution results for cases of KroB100 (left), Pr107 (middle), and Pr144 (right).

Table 2. Comparison results of hybrid SA – 2 opt, SA – NN, and NN – 2 Opt algorithm

Case	BKS	SA – 2 Opt			SA – NN			NN – 2 Opt		
		BS	Ave	WS	BS	Ave	WS	BS	Ave	WS
Ulysess16	73.9876	**73.9876**	**73.9876**	**73.9876**	**73.9876**	**73.9876**	**73.9876**	**73.9876**	**73.9876**	**73.9876**
Ulysess22	75.3097	**75.3097**	**75.3097**	**75.3097**	**75.3097**	**75.3097**	**75.3097**	**75.3097**	**75.3097**	**75.3097**
Att48	33522	**33523.7**	**33523.7**	**33523.7**	33523.7	33527.9	33607.7	33523.7	33547.6	33872.2
Eil51	426	**428.88**	**429.13**	**433.93**	428.98	429.85	432.37	430.24	431.76	435.51
Berlin52	7542	**7544.37**	**7544.37**	**7544.37**	**7544.37**	**7544.37**	**7544.37**	**7544.37**	**7544.37**	**7544.37**
St70	675	**677.1096**	677.502	684.953	**677.1096**	**677.1139**	**677.1945**	677.1945	677.910	681.534
Eil76	538	**545.3876**	**547.965**	**554.958**	549.9961	551.4574	555.1221	551.9872	555.002	560.637
Pr76	108159	**108159.4**	**108159.4**	**108159.4**	**108159.4**	**108159.4**	**108159.4**	108159.4	108229.8	108276.7
Gr96	514	**510.8863**	**511.354**	512.924	511.7315	511.409	**511.9407**	514.0526	514.672	518.113
Rat99	1211	**1219.24**	**1219.73**	**1224.08**	1224.59	1229.59	1247.36	1226.462	1227.674	1250.697
KroA100	21282	**21285.44**	**21289.1**	21357.8	21294.4	21296	**21298.98**	21380.38	21396.6	21552.1
KroB100	22141	**22139.07**	**22150.73**	**22197.33**	22191.33	22224.81	22346.61	22245.18	22255.32	22346.61
KroC100	20749	**20750.76**	20757.26	20852.28	20750.76	**20754.24**	**20820.37**	20816.37	20866.64	21159.13
KroD100	21294	**21294.29**	21328.13	21478.49	**21294.29**	**21294.29**	**21294.29**	21481.15	21500.03	21582.08
KroE100	22068	**22073.25**	**22078.74**	**22161.3**	22107.53	22156.05	22216.05	22078.67	22167.85	22565
Eil101	629	**643.9111**	**647.613**	**651.378**	652.4963	653.017	656.205	656.0167	657.184	658.934
Lin105	14379	**14382.99**	14384.15	14406.12	**14382.99**	**14382.99**	**14382.99**	14383	14390.42	14420.1
Pr107	44303	**44301.68**	44306.13	44346.19	**44301.68**	**44305.16**	**44324.84**	44337.36	44405.93	44503.03
Pr124	59030	**59030.74**	**59030.74**	**59030.74**	**59030.74**	**59030.74**	**59030.74**	59030.74	59048.36	59074.8
Ch130	6110	**6110.722**	**6140.74**	**6210.84**	6134.592	6152.313	6217.597	6176.014	6192.97	6207.90
Pr136	96772	**96875.82**	**96956.52**	**97543.82**	97547.25	97725.07	98459.30	98447.39	98772.69	98962.76
Pr144	58537	**58535.2**	58536.9	58568.77	**58535.2**	**58535.2**	**58535.2**	58535.22	58538.58	58568.77
KroA150	26524	**26524.86**	**26640.87**	26803.01	26729.99	26756.67	**27033.09**	26983.32	27095.1	27344.91
KroB150	26130	**26140.3**	**26188.5**	26551.2	26396.48	26428.94	**26468.62**	26505.17	26523.85	26878.71
Ch150	6528	**6533.81**	**6565.32**	6612.98	6593.021	6593.021	**6593.021**	6622.825	6622.825	6622.825
Pr152	73682	**73683.64**	**73701.86**	**73881.68**	73683.64	73711.98	73907.78	73843.09	73845.69	73846.56
Rat195	2323	**2351.23**	**2356.58**	2396.05	2375.385	2375.385	**2375.385**	2388.146	2395.457	2398.59
KroA200	29368	**29463.13**	**29544.56**	29756.51	29666.4	29666.4	**29666.4**	29666.4	29666.4	29666.4
KroB200	29437	**29588.3**	**29630.3**	**29986.9**	29951.92	30109	30541.03	30204.76	30312.75	30512.03
Pr226	80369	**80461.17**	80904.18	81317.47	80635.47	**80663.16**	**80751.66**	80596.9	80616.69	80676.05
Pr264	49135	**49135**	**49207**	**49352.28**	49767.9	49812.39	50138.23	49822.65	49852.21	49970.47
Pr299	48191	**48468.15**	**48819.68**	50091.91	49227.5	49310.18	**49736.69**	49071.26	49312.1	49938.31
SG6	36	**36**	**36**	**36**	**36**	**36**	**36**	**36**	**36**	**36**
SG7	49.4142	**49.4142**	**49.4142**	**49.4142**	**49.4142**	**49.4142**	**49.4142**	**49.4142**	49.4553	50.2426
SG8	64	**64**	64.0828	64.8284	**64**	**64**	**64**	**64**	**64**	**64**
SG9	81.4142	**81.4142**	81.4556	82.2426	**81.4142**	**81.4142**	**81.4142**	**81.4142**	**81.4142**	**81.4142**
SG10	100	**100**	100.124	100.828	**100**	**100**	**100**	**100**	**100**	**100**
SG11	121.4142	**121.4142**	**121.828**	**122.243**	122.243	122.243	122.243	122.243	122.864	123.899
SG12	144	**144**	**144**	**144**	**144**	**144**	**144**	**144**	**144**	**144**
SG13	169.4142	**169.4142**	**169.414**	**169.414**	170.2426	170.2426	170.2426	172.307	172.412	173.5563
SG14	196	**196**	**196**	**196**	**196**	**196**	**196**	**196**	**196**	**196**
SG15	225.4142	**226.2426**	**226.698**	**228.728**	228.7279	228.7279	228.7279	229.5563	230.136	230.385

Table 3 Results of the proposed algorithm in solving medium-scale symmetric TSP

Case	BKS	BS	Ave	WS	STD.Dev	BE	AveE	WE
Lin318	42029	42433.15	42845.78	43805.65	342.18	0.009616	0.019434	0.042272
Pr439	107217	108039.9	108290.4	109088.8	405.38	0.007675	0.010012	0.017458
Pcb442	50778	51851.2	51968.9	52709.1	239.01	0.021135	0.023453	0.03803
U574	36905	37962.51	38257.52	38732.99	196.61	0.028655	0.036649	0.049532
Rat575	6773	7049.36	7132.23	7329.89	79.89	0.040803	0.053048	0.082222
P654	34643	34836.74	35028.86	35915.18	269.24	0.005592	0.011138	0.036723
D657	48912	50284.86	50594.24	52717.38	616.42	0.028068	0.034393	0.0778
U724	41910	43144.86	43878.97	45520.22	681.43	0.029465	0.046981	0.086142
Rat783	8806	9163.44	9309.52	9620.96	126.37	0.040591	0.057179	0.092546
Pr1002	259045	269762.7	272846.8	277447.7	2043.1	0.04137	0.05328	0.07104
U1060	224094	233324.6	235565	242353.4	2085.7	0.041191	0.051188	0.081481

IV. CONCLUSIONS

Implementation of the hybrid model of simulated annealing and 2 opt algorithm has been conducted. The SA algorithm is based on the outer and inner loop SA algorithm described in [4]. Some simulations of small-scale and medium-scale symmetric traveling salesman problem benchmark tests taken from [18] have been carried out to see the performance of the hybrid algorithm. The hybrid algorithm shows good performance in terms of reliability and stability in finding the optimal solution from the TSP benchmark case. It can be seen from the values of errors and standard deviation. Values of average error and standard deviation for all simulations in the medium scale are 0.0147 and 272, respectively. Moreover, in some cases namely KroB100, Pr107, and Pr144, the hybrid algorithm finds a better solution compared with the best-known solution mentioned in [18]. Further, the proposed algorithm outperforms the SA algorithm in [4] in finding the optimal solution for the cases and computational time. The hybrid algorithm is 1.207 – 5.692 times faster than the SA algorithm in [4]. Further, the proposed algorithm is also better than the other two hybrid algorithms, i.e., NN – SA and NN – 2 Opt algorithm, in solving the TSP benchmark cases.

References

[1] S. Obadan, Z. Wang, "A hybrid optimization approach for complex nonlinear objective functions," *International Journal of Computing*, vol. 17, no. 2, pp.102-112, 2018. https://doi.org/10.47839/ijc.17.2.996

[2] K. Karagul, Aydemir, E. and S. Tokat, "Using 2-Opt based evolution strategy for travelling salesman problem," *An International Journal of Optimization and Control: Theories & Applications* (IJOCTA), vol. 6, issue 2, pp. 103-113, 2016. https://doi.org/10.11121/ijocta.01.2016.00268

[3] E. Osaba, J. Del Ser, A. Sadollah, M. N. Bilbao, and D. Camacho, "A discrete water cycle algorithm for solving the symmetric and asymmetric traveling salesman problem," *Applied Soft Computing*, vol. 71, pp. 277-290, 2018. https://doi.org/10.1016/j.asoc.2018.06.047

[4] I. Iryanto, and P. H. Gunawan, "Numerical approach of symmetric traveling salesman problem using simulated annealing," *Jurnal RESTI (Rekayasa Sistem Dan Teknologi Informasi)*, vol. 5, issue 6, pp.1090-1098, 2021. https://doi.org/10.29207/resti.v5i6.3549

[5] Y. Wang, D. Tian, and Y. H. Li, "An improved simulated annealing algorithm for traveling salesman problem," *Proceedings of the 2012 International Conference on Information Technology and Software Engineering*, 2013, pp. 525-532, Springer, Berlin, Heidelberg. https://doi.org/10.1007/978-3-642-34522-7_56

[6] L. Wang, R. Cai, M. Lin and Y. Zhong, "Enhanced list-based simulated annealing algorithm for large-scale traveling salesman problem," *IEEE Access*, vol. 7, pp. 144366-144380, 2019. https://doi.org/10.1109/ACCESS.2019.2945570

[7] N. Azizi, and S. Zolfaghari, "Adaptive temperature control for simulated annealing: a comparative study," *Computers & Operations Research*, vol. 31, issue 14, pp. 2439-2451, 2004. https://doi.org/10.1016/S0305-0548(03)00197-7

[8] Z. A. Othman, N. H. Al-Dhwai, A. Srour, and W. Yi, "Water flow-like algorithm with simulated annealing for travelling salesman problems," *International Journal on Advanced Science, Engineering and Information Technology*, vol. 7, issue 2, pp. 669-675, 2017. https://doi.org/10.18517/ijaseit.7.2.1837

[9] A. E. S. Ezugwu, A. O. Adewumi and M. E. Frîncu, "Simulated annealing based symbiotic organisms search optimization algorithm for traveling salesman problem," *Expert Systems with Applications*, vol. 77, pp. 189-210, 2017. https://doi.org/10.1016/j.eswa.2017.01.053

[10] P. Stodola, K. Michenka, J. Nohel, and M. Rybanský, "Hybrid algorithm based on ant colony optimization and simulated annealing applied to the dynamic traveling salesman problem," *Entropy*, vol. 22, issue 8, p. 884, 2020. https://doi.org/10.3390/e22080884

[11] K. Katayama and H. Narihisa, "An efficient hybrid genetic algorithm for the traveling salesman problem," *Electronics and Communications in Japan (Part III: Fundamental Electronic Science)*, vol. 84, issue 2, pp. 76-83, 2021. https://doi.org/10.1002/1520-6440(200102)84:2<76::AID-ECJC9>3.0.CO;2-O

[12] M. Xu, S. Li, and J. Guo, "Optimization of multiple traveling salesman problem based on simulated annealing genetic algorithm," *In Matec Web of Conferences*, EDP Sciences, vol. 100, p. 02025, 2017. https://doi.org/10.1051/matecconf/201710002025

[13] A. H. Zhou, L. P. Zhu, B. Hu, S. Deng, Y. Song, H. Qiu, and S. Pan, "Traveling-salesman-problem algorithm based on simulated annealing and gene-expression programming," *Information*, vol. 10, issue 1, p. 7, 2019. https://doi.org/10.3390/info10010007

[14] G. Cabrera, S. Roncagliolo, J. P. Riquelme, C. Cubillos, and R. Soto, "A hybrid particle swarm optimization-simulated annealing algorithm for the probabilistic travelling salesman problem," *Studies in Informatics and Control*, vol. 21, issue 1, pp. 49-58, 2012. https://doi.org/10.24846/v21i1y201206

[15] Y. Lin, Z. Bian, and X. Liu, "Developing a dynamic neighborhood structure for an adaptive hybrid simulated annealing–tabu search algorithm to solve the symmetrical traveling salesman problem," *Applied Soft Computing*, vol. 49, pp. 937-952, 2016. https://doi.org/10.1016/j.asoc.2016.08.036

[16] İ. Küçükoğlu, R. Dewil, and D. Cattrysse, "Hybrid simulated annealing and tabu search method for the electric travelling salesman problem with time windows and mixed charging rates," *Expert Systems with Applications*, vol. 134, pp. 279-303, 2019. https://doi.org/10.1016/j.eswa.2019.05.037

[17] X. Geng, Z. Chen, W. Yang, D. Shi, and K. Zhao, "Solving the traveling salesman problem based on an adaptive simulated annealing algorithm with greedy search," *Applied Soft Computing*, vol. 11, issue 4, pp. 3680-3689, 2011. https://doi.org/10.1016/j.eswa.2019.05.037

[18] G. Reinelt, "TSPLIB", 19 February 1997. [Online]. Available at: http://elib.zib.de/pub/mp-testdata/tsp/tsplib/tsplib.html

[19] S. Kirkpatrick, C. D. Gelatt, and M. P. Vecchi, "Optimization by simulated annealing," *Science*, vol. 220, issue 4598, pp. 671-680, 1983. https://doi.org/10.1126/science.220.4598.671

[20] D. Delahaye, S. Chaimatanan, and M. Mongeau, "Simulated annealing: From basics to applications," *In Handbook of Metaheuristics*, 2019, pp. 1-35, Springer, Cham. https://doi.org/10.1007/978-3-319-91086-4_1
[21] E. K. Burke, and G. Kendall, "Search methodologies: introductory tutorials in optimization and decision support techniques, 2014, pp. 265-285, Springer. https://doi.org/10.1007/978-1-4614-6940-7
[22] M. El Alaoui, and M. Ettaouil, "An adaptive hybrid approach: Combining neural networks and simulated annealing to calculate the equilibrium point in max-stable problem," *IAENG International Journal of Computer Science*, vol. 48, issue 4, pp. 893-8982021.
[23] G. A. Croes, "A method for solving travelling salesman problems," *Operations Research*, vol. 6, no. 6, pp. 791-812, 1958. https://doi.org/10.1287/opre.6.6.791
[24] P. H. Siqueira, S. Scheer, and M. T. A. Steiner. "A recurrent neural network to traveling salesman problem," *Travelling Salesman Problem, I-Tech Veinna*, pp. 135-156, 2008.
[25] A. Hatamlou, "Solving travelling salesman problem using black hole algorithm," *Soft Computing*, vol. 22, no. 24, pp. 8167-8175, 2018. https://doi.org/10.1007/s00500-017-2760-y

***P. H. GUNAWAN** is a Lecturer at the School of Computing, Telkom University. He graduated with a doctoral degree from Universite Paris-Est Marne la Valle and Institut Teknologi Bandung in the Mathematics department (Joint Degree Program). His research focuses on Applied Mathematics, Numerical Analysis, Machine Learning, and High-performance Computing.*

Moreover, his specialty is in the numerical method for partial differential equations (PDE) models, such as shallow water equations and sedimentation models..

***IRYANTO** is currently working as a Lecturer in Politeknik Negeri Indramayu. He received his master's degree in computational science at Institut Teknologi Bandung. Moreover, his research interests include applied mathematics, computational science, soft computing, and modeling & simulation.*