International Journal of
Intelligent Engineering & Systems

http://www.inass.org/

A New Method for Travelling Salesman Problem Relied on Growth Optimization

Quyen Thi Nguyen[1]*

[1]*Faculty of Information Technology, School of Technology, Van Lang University, Ho Chi Minh City, Vietnam*
* Corresponding author's Email: quyen.nt@vlu.edu.vn

Abstract: This paper shows a novel method relied on growth optimization (GO) algorithm for searching the shortest tour length of the travelling salesman problem (TSP). GO is a recent algorithm relied on the idea of learning and reflecting of people in the society. To enhance performance of GO, the 2-opt local search technique is applied for adjusting the candidate solutions created by GO. The effectiveness of GO is validated on five TSP instances consisting of the 14-city, 30-city, 48-city, 52-city and 76-city. The error between the optimal tour length value obtained by GO and the best-known value for these instances is 0.0000%, 0.0000%, 0.0021%, 0.0314% and 0.0004%, respectively. Furthermore, the comparisons to the methods in the literature in term of the optimal and mean tour length values have shown that GO reaches the better values compared to other prvious methods. Thus, the proposed GO approach is a potential method for the TSP problem.

Keywords: Growth optimization, Travelling salesman problem, 2-opt, Shortest tour length.

1. Introduction

The TSP is the optimization problem that determines the minimal cost Hamiltonian cycle. For a set of n cities with distance among cities, a salesman has to determine the way to visit cities once and comeback to the starting city while minimizing the length of the tour. The main challenge of the TSP is to manage cities in the optimal order. The TSP problem is used in many fields of the real life such as transportation, manufacturing and logistics [1]. But, the TSP problem is not only a complex problem but the number of possible solutions can be up to n factorial. Thus, it should be solved by effective methods.

There are two main method groups for the TSP problem including exacting method and metaheuristic algorithm. For the first one, there are some popular methods like lagrangian [2], branch-and-bound [3], branch-and-cut [4, 5] and pseudo-polynomial-time-exact algorithm [6]. The main characters of these methods is that the problem description is complex and they may take a long-time for large-scale problems [7]. For the second one, there are some popular methods like genetic-algorithm (GA) [8-10] particle-swarm-optimization (PSO) [11-13] ant-colony-optimization (ACO) [14-16] cuckoo-search (CS) [17, 18] harmony-search-algorithm (HSA) [19], artificial-bee-colony (ABC) [20], tree-seed-algorithm (TSA) [21], discrete-spider-monkey-optimization (DSMO) [22], simulated-annealing-algorithm [1], discrete-komodo-algorithm (DKA) [23]. In addition, some hybrid algorithms have been proposed for the TSP to enhance the performance of the algorithms for the considered problem. In [24], the combination of ACO and GA (ACO-GA) has gained the optimal results faster than the standard ones. In [25], the hybrid of GA-PSO-ACO has obtained the better results than those of GA and PSO. Similarly, the hybrid of harris-hawk-optimizer and ACO (HHO-ACS) has outperformed to PSO and GA [26]. In [27], the hybrid of SOS and SA (SOS-SA) has outperformed to SOS for the TSP problem.

Based on the idea of metaheuristic algorithms, the above metaheuristic algorithms can be divided into several categories such as evolutionary algorithms like GA, swarm intelligence including PSO, ACO, CS, ABC, DSMO and DKA and physics-based algorithms such as HSA and SA. The hybrid versions of these algorithms retain the main idea of the original algorithms. Most of these algorithms rely on ideas from nature. Meanwhile, compared to the physical phenomena or behavior of animals in nature, humans have superior thinking and skills. Therefore, algorithms based on human behavior will definitely be better than algorithms inspired by nature [28]. In addition, some algorithms such as PSO, ACO, ABC, SOS mainly orient the search space through the best individual, which can lead to a lack of search information, leading to an imbalance between exploration and exploitation [29]. Therefore, using an algorithm with the ability to synthesize more information during the movement orientation process for the TSP problem also needs to be considered.

In addition, as no-free-lunch-theorem, there does not exist the best algorithm for all problems. An algorithm achieves the best efficiency for a given problem, but it does not get the same performance another problem [30, 31]. Furthermore, there are many new algorithms being developed nowadays like swarm-magnetic-optimizer [32], extended-stochastic coati-optimizer [33], walk-spread-algorithm [34] and growth-optimization (GO) [29], etc. They have also demonstrated their advantages compared with many previous algorithms for classic functions. Therefore, considering their capabilities for the TSP problem to diversify methods for this problem should also be encouraged.

This paper presents a new algorithm to resolve the TSP problem relied on GO. The GO is developed based on metaphor of learning and reflecting of people in the society [29]. Learning is the way each individual gains information from the world, while reflection is the way that he identifies and corrects his shortcomings in order to improve his learning strategies and grow as individuals. GO uses these two concepts to generate new solutions in two ways. The learning technique generates new solutions relied on information from five different individuals. This supports GO avoid getting stuck in local optima. The reflection technique uses a variety of techniques to generate new solutions based on information from the best solutions. For the benchmark functions, GO has shown the high efficiency compared to other algorithms [29]. However, its performance for the TSP problem should be considered.

Therefore, this paper shows GO steps for the TSP. For solving the TSP problem, in the population, each individual represents for each tour. The order in which the elements of a solution vector are arranged in ascending order is described by an index array, which is considered as real tour of the solution vector.

Furthermore, the 2-opt technique is used to adjust individuals of GO. The efficiency of GO is demonstrated on different instances such as the 14-city, 30-city, 48-city, 52-city and 76-city. The performance of the proposed method is compared to other previous methods in literature. The contributions of this work are summarized below:

(i) GO is the first shown for searching the optimal tour of the TSP problem.

(ii) The 2-opt technique is combined to adjusted the GO's solutions.

(iii) The GO's efficiency is evaluated on the 14-city, 30-city, 48-city, 52-city and 76-city instances.

(iv) The proposed GO gains the better performance than previous methods.

The rest paper is include parts: The TSP problem is shown in the next section. Section 3 shows the application of GO for finding the optimal tour of the TSP problem. Section 4 discuss the results. The conclusions are shown in final section.

2. The TSP problem

The distance between two cities is calculated as follows:

$$d(c_i, c_{i+1}) = \sqrt{(x_i - x_{i+1})^2 + (y_i - y_{i+1})^2} \quad (1)$$

Where, (x_i, y_i) and (x_{i+1}, y_{i+1}) are respectively coordinates of the city i and $i+1$.

The tour length for D cities is determined below:

$$f(tour) = \sum_{i=1}^{D-1} d(c_i, c_{i+1}) + d(c_D, c_1) \quad (2)$$

Where, $d(c_D, c_1)$ is the distances between city c_D and the first city.

3. Application GO for TSP

In GO, each individual is considered as a candidate solution. The initial population of solutions is created as follows:

$$T_{i,j} = L_j + rand.\left(H_j - L_j\right) \quad (3)$$

Where, $T_{i,j}$ is variable j of solution i with $j = 1,2,\ldots,D$ and $i = 1,\ldots,N$. N and D is the population size and dimension. $[L_j, H_j]$ is the limit of the variable j.

The solution generated by (3) is a real number. In order to form the tour for TSP, the solution is sorted in ascending order. Then the indexes that shows the order of variables in solution is treated as the real tour. Thus, in this work, the limits of the variables are selected in ranges of [-2000, 2000].

In addition, to improve the quality of solutions, each solution is adjusted by using the 2-opt technique

```
Input: The current T_i
Set ΔD_min = 0
For i = 1 to D do
    n_1 = i
    m_1 = i − 1
    If m_1 = 0 then
        m_1 = N_city
    End
    For j = i + 2 to D do
        n_2 = j
        m_2 = n_2 − 1
        ΔD = d(m_1, m_2) + d(m_1, m_2) − d(m_1, n_1) − d(m_2, n_2)
        If ΔD < ΔD_min then
            ΔD_min = ΔD
            Save positions of m_1, n_1, m_2 and n_2
        End
    End
End
T_i (n_1: m_2) = reverse (T_i (m_2: n_1))
Result: The new tour T_i
```

Figure. 1 The 2-opt pseudo code for modifying the current tour

[35]. The operating principle of the 2-opt algorithm is to cut two edges in the current tour and join the two parts to form a new one. If the two old edges are longer than the new ones, the new tour will substitute for the old one. Its pseudo code to find the better tour than the current one based on this technique is presented in Fig. 1.

After adjusting by the 2-opt algorithm, each solution is calculated the fitness value (f_i) by using (2) and the best-so-far solution (T_{best}) with the best fitness values (f_{best}) is abtained.

In GO, each solution is updated by two phases consisting of leaning and reflection. For the first one, the solution is renewed based on the information of the neighbour in the population. Based on the fitness values, in addition to the best solution, there are P_1 solutions to be considered as the better solution (T_b) and the rest solutions are considered as the worse individuals (T_w). The distances among solutions are defined as follows:

$$\begin{cases} \theta_1 = T_{best} - T_{b,h} \\ \theta_2 = T_{best} - T_{w,k} \\ \theta_3 = T_{b,h} - T_{w,k} \\ \theta_4 = T_{r1} - T_{r2} \end{cases} \quad (4)$$

The affections of each distance (C_j) to learning ability solutions is calculated as follows:

$$C_j = \frac{\|\theta_j\|}{\|\theta_1+\theta_2+\theta_3+\theta_4\|}; j = 1, 2, 3, 4 \quad (5)$$

Depending on the quality of each solution, they have different learning level that is defined as follows:

$$S_i = \frac{f_i}{f_{worst}} \quad (6)$$

Where, S_i is the learning ability of solution i. f_{worst} is the fitness value of the worst solution.

Finally, each solution is updated based on the learning phase as follows:

$$\mathrm{T}_i^{new} = T_i + S_i.[C_1.\theta_1 + C_2.\theta_2 + C_3.\theta_3 + C_4.\theta_4] \quad (7)$$

The new solutions are modified by the 2-opt technique and their fitness value is found using (2). Then, the current solutions are updated as follows:

$$T_i = \begin{cases} T_i^{new}; & if\ f_i^{new} < f_i \\ \begin{cases} \mathrm{T}_i^{new};\ if\ \text{rand} < P_2 \\ T_i;\quad ortherwise \end{cases} & ortherwise \end{cases} \quad (8)$$

The P_2 shows the probability of knowledge retention of solution T as the process of knowlwdge update fails.

For the reflection phase, each solution in the population is updated as follows:

$$T_{i,j}^{new} = \begin{cases} \begin{cases} L_j + \vartheta_1.(H_j - L_j);\ if\ \vartheta_2 < A \\ T_{i,j} + \vartheta_3.(R_j - T_{i,j});\ orthewise \end{cases} & ;\ if\ \vartheta_4 < P_3 \\ T_{i,j} & ;\ ortherwise \end{cases} \quad (9)$$

Where, ϑ_1 to ϑ_4 are random number in [0,1]. P_3 is the reflection probability. R_j is variable of the better solution chosen randomly. A is the attenuation factor defined as follows:

$$A = 0.01 + 0.99.(it/maxIt) \quad (10)$$

Where, it and $maxIt$ are the current and maximum number of iterations.

The new solutions are modified by the 2-opt technique and their fitness value is calculated using (2). Then, the current solutions are updated using (8). Pseudocode of GO for TSP is shown in Fig. 2.

4. Numerical results

The GO for the TSP is carried out on Matlab and executed on the computer with configuration of 4GB RAM and core-i5 CPU. The instances are chosen

Table 1. The optimal results gained by GO for the instances

Instances	The optimal tour	$f_{best-known}$	Tour length	OE
Burma14	3, 7, 12, 6, 5, 4, 3, 14, 2, 1, 10, 9, 11, 8	30.8785	30.8785	0.0000%
Oliver30	17, 18, 19, 20, 21, 22, 23, 25, 24, 26, 27, 28, 29, 30, 2, 1, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16	423.7406	423.7406	0.0000%
Att48	37, 6, 28, 7, 18, 44, 31, 38, 8, 1, 9, 40, 15, 12, 11, 13, 25, 14, 23, 3, 22, 16, 41, 34, 29, 2, 26, 4, 35, 45, 10, 24, 42, 5, 48, 39, 32, 21, 47, 20, 33, 46, 36, 30, 43, 17, 27, 19	33523	33523.7085	0.0021%
Berlin52	32, 45, 19, 41, 8, 9, 10, 43, 33, 51, 11, 52, 14, 13, 47, 26, 27, 28, 12, 25, 4, 6, 15, 5, 24, 48, 38, 37, 40, 39, 36, 35, 34, 44, 46, 16, 29, 50, 20, 23, 30, 2, 7, 42, 21, 17, 3, 18, 31, 22, 1, 49	7542	7544.3659	0.0314%
Pr76	29, 30, 31, 19, 20, 26, 27, 28, 43, 42, 54, 53, 52, 55, 56, 57, 58, 59, 60, 41, 61, 62, 63, 64, 73, 72, 71, 65, 66, 51, 49, 50, 67, 70, 68, 69, 47, 48, 44, 45, 46, 24, 25, 21, 22, 23, 1, 76, 75, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 74, 15, 16, 17, 18, 37, 36, 38, 39, 40, 34, 35, 33, 32	108159	108159.4383	0.0004%

```
Input: N, D, maxIt, P_1, P_2, P_3 and It = 0
Create solutions using (3)
Modify the solutions using the 2-opt technique
Find the fitness value of each solution using (3)
Find the best-so-far individual
While It < maxIt do
[~, index] = sort(f)
T_best = T(index(1), :)
    For i = 1 : N do
        T_w = T(index(randi(N - P_1 + 1)), :)
        T_better = T(index(randi(2, P_1)), :)
        Choose two random solutions T_r1 and T_r2
        Calculate distances among solutions, affections
of each distance and learning level of each solution
using (4), (5) and (6)
        Generate the new solutions using (7)
        Modify the solutions using the 2-opt technique
        Validate the quality solution f_i using (2)
        Renew the current tour by (8)
        Renew the best-so-far individual
    End for
    For i = 1 : N do
        Create new solution using (9) and (10)
        Modify the solutions using the 2-opt technique
        Validate the quality solution f_i using (2)
        Renew the current tour by (8)
        Renew the best-so-far individual
    End for
        It = It + 1
End while
Result: The best-so-far tour
```

Figure. 2 Pseudocode of GO for the TSP

from the TSP library including Burma14, Oliver30, Att48, Berlin52 and pr76 referenced from TSPLIB for searching the shortest length [36, 37]. The control parameters of N, P_1, P_2, P_3 and maxIt are set to 30, 5, 0.001 and 0.3, respectively. For each instance, GO is independently executed 20 times and the best solution in these runs is considered as the result of the problem. The results of GO are compared to the previous methods.

Table 1 shows the length tour of cities gained by GO. The optimal tour length for the 14-city, 30-city, 48-city, 52-city and 76-city instances are 30.8785, 423.7406, 33523.7085, 7544.3659 and 108159.4383 respectively. The error of the optimal and the best-known result (OE) of instances is 0.0000%, 0.0000%, 0.0021%, 0.0314% and 0.0004%, respectively. The shortest tour of the cities is shown in Fig. 3, wherein the shortest tour for the 14-city, 30-city, 48-city, 52-city and 76-city instances are respectively shown in Figs. 3(a), 3(b), 3(c), 3(d), and 3(e). The mean, maximum and minimum convergence curves of GO for these cities in Fig. 4, wherein the convergences for the 14-city, 30-city, 48-city, 52-city and 76-city instances are respectively shown in Figs. 4(a), 4(b), 4(c), 4(d), and 4(e). These figures show that the mean curves are closed to the minimum ones. This confirms that GO is a potential method for the considered problem.

The compared results between GO with other methods for optimal and mean tour length values as well as the error of the mean and the best-known result (ME) are shown in Table 2. For the 14-city, all of GO, PSO [12] and DSMO [22] have determined the bets tour. However, based on the mean fitness value and ME value, the performance of GO is better than that of PSO [12] with the lower f_{mean} and ME value. Compare with DSMO [22], GO reaches the same results as those of DSMO [22]. For the 30-city, GO have also found the optimal tour similar to PSO [12], DTSA [21], ACO-GA [24], DABC [38] and discrete-jaya-algorithm (DJAYA) [39]. However, the f_{mean} value obtained by GO is lower than that of PSO [12], DTSA [21] and DJAYA [39]. The ME value of GO is 2.0018%, 1.1232% and 0.7409% lower than that of these methods. For the 48-city, the

Received: November 14, 2023. Revised: December 28, 2023.

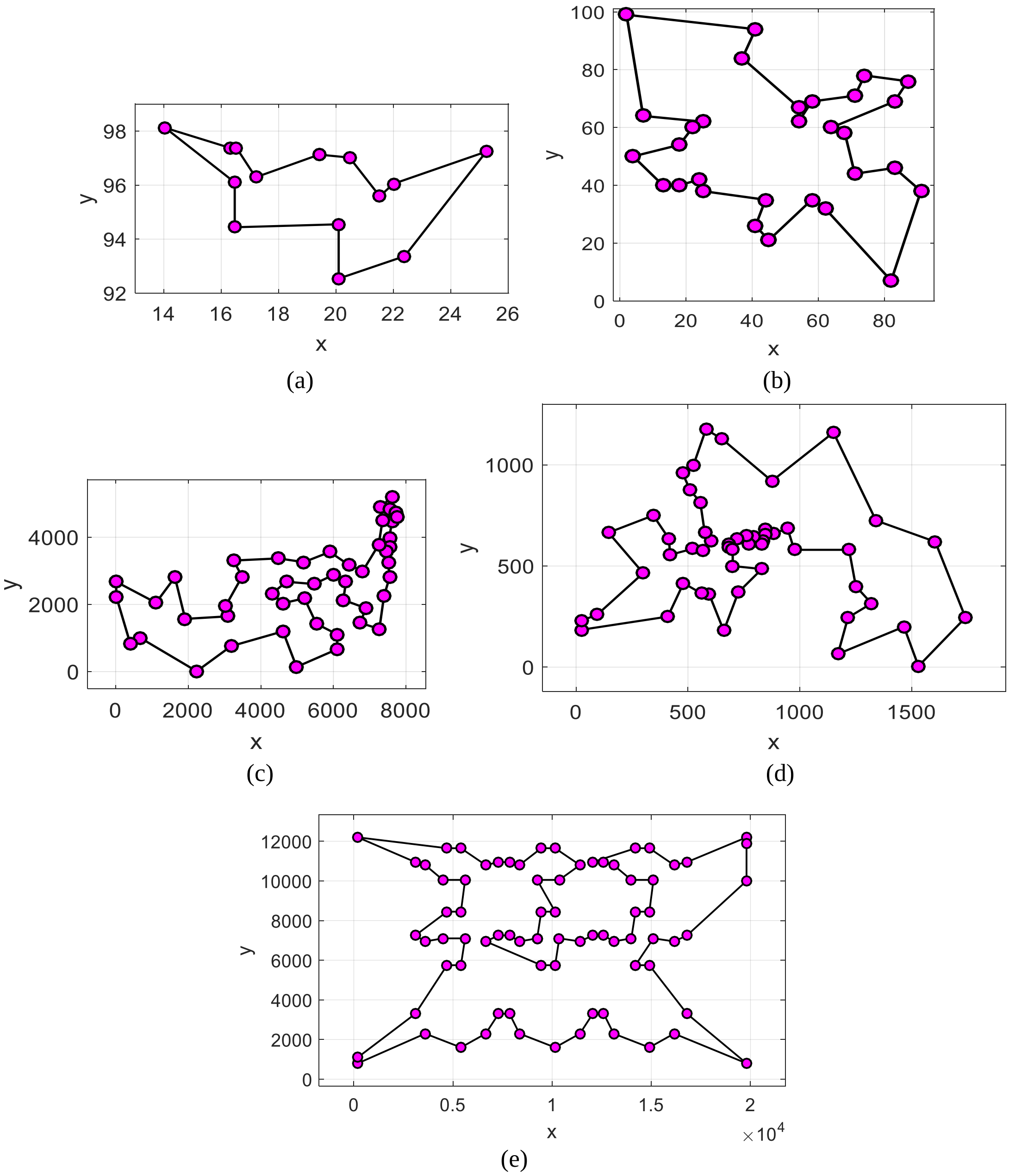

Figure. 3 The optimal tours: (a) for the 14-city, (b) 30-city and (c) 48-city instances, (d) for the 52-city, and (e) 76-city instances

optimal tour gained by GO is the better that of GA-PSO-ACO [25] and is only 0.7085, 1.7085, 0.7085 and 1.7085 higher than that of SOS-SA [27], HHO-ACS [26], discrete-grey-wolf-optimizer (DGWO) [40], and discrete-sparrow-search-algorithm (DSSA) [41]. The ME value of GO is 0.1724% and 0.0054% lower than that of GA-PSO-ACO [25] and HHO-ACS [26]. For the 52-city, the optimal tour gained by GO is equal to GA-PSO-ACO [25] and DSMO [22] and higher than that of DTSA [21], SOS-SA [27], HHO-ACS [26], DKA [23], DJAYA[39], random-walk-discrete-cuckoo-search (RW-DCS) [42] and DSSA [41]. The GO's AE value is lower than that of DTSA [21], HHO-ACS [26], DSMO [22], DKA [23], DJAYA[39] and similar to that of GA-PSO-ACO [25] as well as slightly higher than RW-DCS [42] and DSSA [41]. For 76-city, the optimal tour gained by GO is similar to DSMO [22], DBSA [43], DJAYA[39], DGWO [40], RW-DCS [42] and better than that of GA-PSO-ACO [25] and DKA [23]. These results show that GO can reach the efficiency for the TSP problem.

5. Conclusion

This paper presents the steps of the GO algorithm for the TSP problem. To enhance the quality of solution created by GO, the 2-opt local search approach is added to GO to modify generated

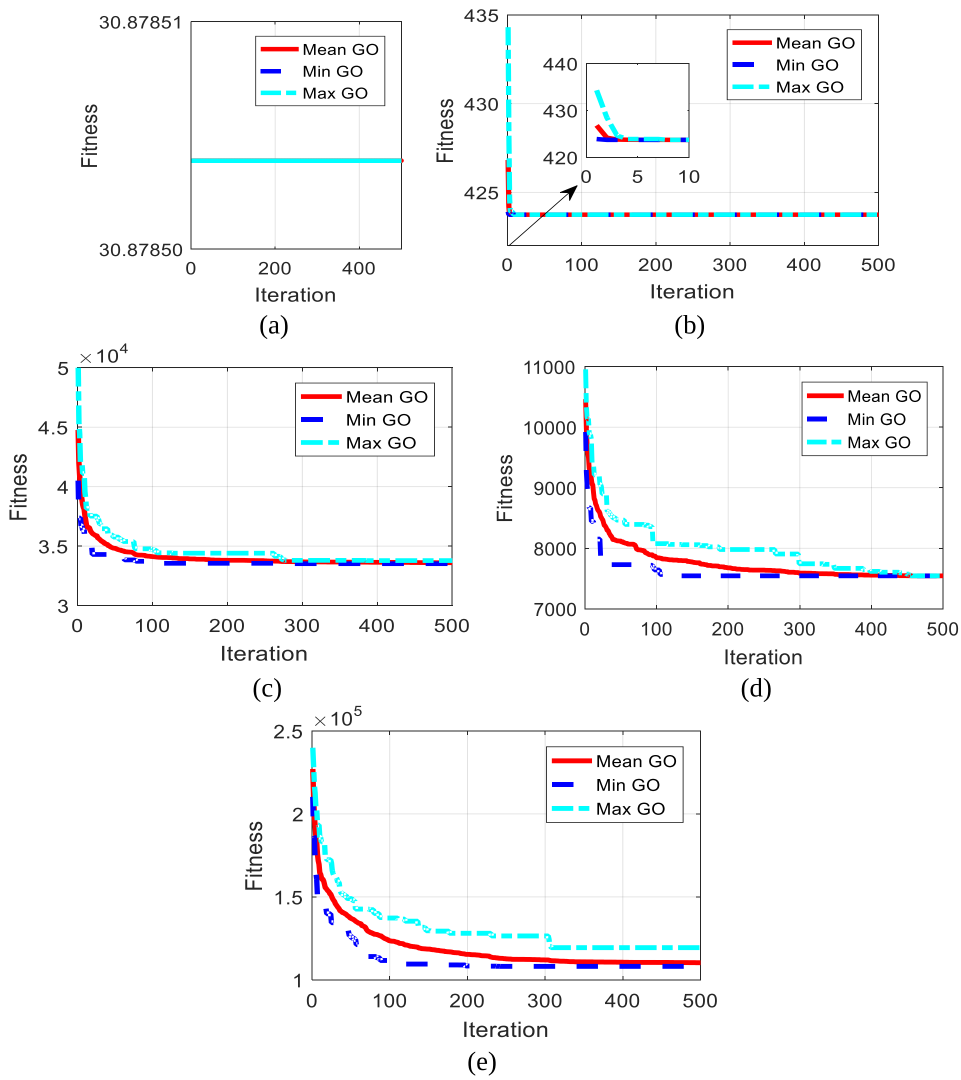

Figure. 4 The convergence of the GO: (a) for the 14-city, (b) 30-city, (c) 48-city instances, (d) for the 52-city, and (e) 76-city instances

solutions. The efficiency of GO is demonstrated on the cities consisting of 14-city, Oliver30-city, 48-city, 52-city and 76-city. The obtained results is compared to the previous approaches. The results have shown that GO has ability to search the optimal tour for the cities with the small AE value. In addition, the compared results with other methods have also shown that GO is also an effective method for the TSP problem. For the future, GO can be used for larger-scale cities as well as the practical problems.

Notation list

$d(c_i, c_{i+1})$:	Distance between two cities
(x_i, y_i):	Coordinates of the city i
(x_{i+1}, y_{i+1}):	Coordinates of the city $i+1$
$d(c_D, c_1)$:	Distance between city c_D and the first city
$\theta_1, \theta_2, \theta_3, \theta_4$:	Distances among solutions
$T_{b,h}$:	The better solution hth
$T_{w,k}$:	The worse solution kth
T_{r1}, T_{r2}:	Random solutions
f_i:	Fitness value of solution T_i
P_1:	Number of better solutions
P_2:	Probability of knowledge retention of solution T_i
P_3:	The reflection probability
it:	The current number of iterations
$maxIt$:	The maximum number of iterations

Conflicts of interest

The authors declare no conflict of interest.

Author contributions

Conceptualization, methodology, software, writing—original draft preparation, writing—review and editing, Q.T.N.

Table 2. The comparisons of GO with the previous methods for the considered cities

No. of City	Method	$f_{best-known}$	$f_{optimal}$	f_{mean}	ME
14	GO	30.8785	30.8785	30.8785	0.0000%
	PSO [12]	30.8785	30.8785	30.9245	0.1490%
	DSMO [22]	30.8785	30.8785	30.8785	0.0000%
30	GO	423.7406	423.7406	423.7406	0.0000%
	PSO [12]	423.7406	423.7406	432.2231	2.0018%
	DTSA [21]	423.7406	423.7406	428.5000	1.1232%
	ACO-GA [24]	423.7406	423.7406	-	-
	DABC [38]	423.7406	423.7406	423.7406	0.0000%
	DJAYA[39]	423.7406	423.7406	426.88	0.7409%
48	GO	33523	33523.7085	33604.1950	0.2422%
	GA-PSO-ACO [25]	33523	33524	33662	0.4146%
	SOS-SA [27]	33523	33523	33539.6800	0.0498%
	HHO-ACS [26]	33523	33522	33606.0000	0.2476%
	DGWO [40]	33523	33523	33600	0.2297%
	DSSA [41]	33522	33522	33522	0.0000%
52	GO	7542	7544.3659	7544.3659	0.0314%
	DTSA [21]	7542	7542.0000	7761.6000	2.9117%
	GA-PSO-ACO [25]	7542	7544.3700	7544.3700	0.0314%
	SOS-SA [27]	7542	7540	7541.1070	-0.0118%
	HHO-ACS [26]	7542	7542	7657	1.5248%
	DSMO [22]	7542	7544.3700	7633.6000	1.2145%
	DKA [23]	7542	7542	7646.25	1.3823%
	DJAYA[39]	7542	7542	7668.35	1.6753%
	RW-DCS [42]	7542	7542	7542	0.0000%
	DSSA [41]	7542	7542	7542	0.0000%
76	GO	108159	108159.4383	110358.2997	2.0334%
	GA-PSO-ACO [25]	108159	109206	110023	1.7234%
	DSMO [22]	108159	108159.4000	111299.3000	2.9034%
	DBSA [43]	108159	108159.4400	108293.7000	0.1245%
	DKA [23]	108159	108304	110134.95	1.8269%
	DJAYA[39]	108159	108159.4400	113258.29	4.7146%
	DGWO [40]	108159	108159.4400	108900	0.6851%
	RW-DCS [42]	108159	108159	108159	0.0000%

References

[1] X. Dong, Q. Lin, F. Shen, Q. Guo, and Q. Li, “A novel hybrid simulated annealing algorithm for colored bottleneck traveling salesman problem”, *Swarm and Evolutionary Computation*, Vol. 83, No. December 2022, p. 101406, 2023.

[2] G. Laporte, “The traveling salesman problem: An overview of exact and approximate algorithms”, *European Journal of Operational Research*, Vol. 59, No. 2, pp. 231–247, 1992.

[3] R. Salman, F. Ekstedt, and P. Damaschke, “Branch-and-bound for the Precedence Constrained Generalized Traveling Salesman Problem”, *Operations Research Letters*, Vol. 48, No. 2, pp. 163–166, 2020.

[4] Y. Yuan, D. Cattaruzza, M. Ogier, and F. Semet, “A branch-and-cut algorithm for the generalized traveling salesman problem with time windows”, *European Journal of Operational Research*, Vol. 286, No. 3, pp. 849–866, 2020.

[5] A. A. Chaves, B. L. Vianna, T. T. da Silva, and C. M. Schenekemberg, “A parallel branch-and-cut and an adaptive metaheuristic to solve the Family Traveling Salesman Problem”, *Expert Systems with Applications*, Vol. 238, No. PA, p. 121735, 2024.

[6] L. Xu, Z. Xu, and D. Xu, “Exact and approximation algorithms for the min-max k-traveling salesmen problem on a tree”, *European Journal of Operational Research*, Vol. 227, No. 2, pp. 284–292, 2013.

[7] C. Jiang, Z. Wan, and Z. Peng, “A new efficient hybrid algorithm for large scale multiple traveling salesman problems”, *Expert Systems with Applications*, Vol. 139, p. 112867, 2020.

[8] X. Dong and Y. Cai, “A novel genetic algorithm for large scale colored balanced traveling

salesman problem", *Future Generation Computer Systems*, Vol. 95, pp. 727–742, 2019.

[9] Y. Deng, Y. Liu, and D. Zhou, "An Improved Genetic Algorithm with Initial Population Strategy for Symmetric TSP", *Mathematical Problems in Engineering*, Vol. 2015, p. 212794, 2015.

[10] J. Xu, L. Pei, and R. Z. Zhu, "Application of a genetic algorithm with random crossover and dynamic mutation on the travelling salesman problem", *Procedia Computer Science*, Vol. 131, pp. 937–945, 2018.

[11] X. L. Xu, X. Cheng, Z. C. Yang, X. H. Yang, and W. L. Wang, "Improved particle swarm optimization for Traveling Salesman Problem", In: *Proc - 27th European Conference on Modelling and Simulation, ECMS 2013*, No. November, pp. 857–862, 2013.

[12] Z. Yuan, L. Yang, Y. Wu, L. Liao, and G. Li, "Chaotic particle swarm optimization algorithm for traveling salesman problem", In: *Proc of the IEEE International Conference on Automation and Logistics, ICAL 2007*, pp. 1121–1124, 2007.

[13] M. Hoffmann, M. Mühlenthaler, S. Helwig, and R. Wanka, "Discrete particle swarm optimization for TSP: theoretical results and experimental evaluations", *Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, Vol. 6943 LNAI, pp. 416–427, 2011.

[14] F. Dahan, K. E. Hindi, H. Mathkour, and H. Alsalman, "Dynamic flying ant colony optimization (DFACO) for solving the traveling salesman problem", *Sensors*, Vol. 19, No. 8, p. 1837, 2019.

[15] S. Ebadinezhad, "DEACO: Adopting dynamic evaporation strategy to enhance ACO algorithm for the traveling salesman problem", *Engineering Applications of Artificial Intelligence*, Vol. 92, No. April, p. 103649, 2020.

[16] S. Bouzbita, A. E. E. Afia, and R. Faizi, "The behaviour of ACS-TSP algorithm when adapting both pheromone parameters using fuzzy logic controller", *International Journal of Electrical and Computer Engineering*, Vol. 10, No. 5, pp. 5436–5444, 2020.

[17] A. Ouaarab, B. Ahiod, and X. S. Yang, "Discrete cuckoo search algorithm for the travelling salesman problem", *Neural Computing and Applications*, Vol. 24, No. 7–8, pp. 1659–1669, 2014.

[18] X. Ouyang, Y. Zhou, Q. Luo, and H. Chen, "A novel discrete cuckoo search algorithm for spherical traveling salesman problem", *Applied Mathematics and Information Sciences*, Vol. 7, No. 2, pp. 777–784, 2013.

[19] U. Boryczka and K. Szwarc, "The Harmony Search algorithm with additional improvement of harmony memory for Asymmetric Traveling Salesman Problem", *Expert Systems with Applications*, Vol. 122, pp. 43–53, 2019.

[20] V. Pandiri and A. Singh, "An artificial bee colony algorithm with variable degree of perturbation for the generalized covering traveling salesman problem", *Applied Soft Computing Journal*, Vol. 78, pp. 481–495, 2019.

[21] A. C. Cinar, S. Korkmaz, and M. S. Kiran, "A discrete tree-seed algorithm for solving symmetric traveling salesman problem", *Engineering Science and Technology, an International Journal*, Vol. 23, No. 4, pp. 879–890, 2020.

[22] M. A. H. Akhand, S. I. Ayon, S. A. Shahriyar, N. Siddique, and H. Adeli, "Discrete Spider Monkey Optimization for Travelling Salesman Problem", *Applied Soft Computing Journal*, Vol. 86, p. 105887, 2020.

[23] G. K. Jati, G. Kuwanto, T. Hashmi, and H. Widjaja, "Discrete komodo algorithm for traveling salesman problem[Formula presented]", *Applied Soft Computing*, Vol. 139, p. 110219, 2023.

[24] E. K. Elsayed, A. H. Omar, and K. E. Elsayed, "Smart solution for STSP semantic traveling salesman problem via hybrid ant colony system with genetic algorithm", *International Journal of Intelligent Engineering and Systems*, Vol. 13, No. 5, pp. 476–489, 2020, doi: 10.22266/ijies2020.1031.42.

[25] W. Deng, R. Chen, B. He, Y. Liu, L. Yin, and J. Guo, "A novel two-stage hybrid swarm intelligence optimization algorithm and application", *Soft Computing*, Vol. 16, No. 10, pp. 1707–1722, 2012.

[26] S. A. Yasear and K. R. K. Mahamud, "Fine-Tuning the Ant Colony System Algorithm Through Harris's Hawk Optimizer for Travelling Salesman Problem", *International Journal of Intelligent Engineering and Systems*, Vol. 14, No. 4, pp. 136–145, 2021, doi: 10.22266/ijies2021.0831.13.

[27] 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.

[28] A. C. Johnvictor, V. Durgamahanthi, R. M. P. Venkata, and N. Jethi, "Critical review of bio-inspired optimization techniques", *Wiley*

Interdisciplinary Reviews: Computational Statistics, Vol. 14, No. 1, p. e1528, 2022.

[29] Q. Zhang, H. Gao, Z. H. Zhan, J. Li, and H. Zhang, "Growth Optimizer: A powerful metaheuristic algorithm for solving continuous and discrete global optimization problems", *Knowledge-Based Systems*, Vol. 261, p. 110206, 2023.

[30] D. H. Wolpert and W. G. Macready, "No free lunch theorems for optimization", *IEEE Transactions on Evolutionary Computation*, Vol. 1, No. 1, pp. 67–82, 1997.

[31] T. Thanh, T. Trung, N. Au, and T. Long, "A novel method based on coyote algorithm for simultaneous network reconfiguration and distribution generation placement", *Ain Shams Engineering Journal*, Vol. 12, No. 1, pp. 665–676, 2021.

[32] P. D. Kusuma and F. C. Hasibuan, "Swarm Magnetic Optimizer: A New Optimizer that Adopts Magnetic Behaviour", *International Journal of Intelligent Engineering and Systems*, Vol. 16, No. 4, pp. 264–275, 2023, doi: 10.22266/ijies2023.0831.22.

[33] P. D. Kusuma and A. Dinimaharawati, "Extended Stochastic Coati Optimizer", *International Journal of Intelligent Engineering and Systems*, Vol. 16, No. 3, pp. 482–494, 2023, doi: 10.22266/ijies2023.0630.38.

[34] P. D. Kusuma and A. L. Prasasti, "Walk-Spread Algorithm: A Fast and Superior Stochastic Optimization", *International Journal of Intelligent Engineering and Systems*, Vol. 16, No. 5, pp. 275–288, 2023, doi: 10.22266/ijies2023.1031.24.

[35] G. A. Croes, "A Method for Solving Traveling-Salesman Problems", *Operations Research*, Vol. 6, No. 6, pp. 791–812, 1958.

[36] G. Reinelt, "TSPLIB. A traveling salesman problem library", *ORSA Journal on Computing*, Vol. 3, No. 4, pp. 376–384, 1991.

[37] I. M. Oliver, D. Smith, and J. R. Holland, "Study of permutation crossover operators on the traveling salesman problem", in *Genetic Algorithms and Their Applications: Proceedings of the Second International Conference on Genetic Algorithms: July 28-31, 1987 at the Massachusetts Institute of Technology, Cambridge, MA. Hillsdale, NJ: L. Erlhaum Associates*, at the Massachusetts Institute of Technology, Cambridge, MA. Hillsdale, NJ: L. Erlhaum Associates, pp. 224–230, 1987.

[38] M. S. Kiran, H. Işcan, and M. Gündüz, "The analysis of discrete artificial bee colony algorithm with neighborhood operator on traveling salesman problem", *Neural Computing and Applications*, Vol. 23, No. 1, pp. 9–21, 2013.

[39] M. Gunduz and M. Aslan, "DJAYA: A discrete Jaya algorithm for solving traveling salesman problem", *Applied Soft Computing*, Vol. 105, p. 107275, 2021.

[40] K. Panwar and K. Deep, "Discrete Grey Wolf Optimizer for symmetric travelling salesman problem", *Applied Soft Computing*, Vol. 105, p. 107298, 2021.

[41] Z. Zhang and Y. Han, "Discrete sparrow search algorithm for symmetric traveling salesman problem", *Applied Soft Computing*, Vol. 118, p. 108469, 2022.

[42] Z. Zhang and J. Yang, "A discrete cuckoo search algorithm for traveling salesman problem and its application in cutting path optimization", *Computers and Industrial Engineering*, Vol. 169, No. April, p. 108157, 2022.

[43] M. Lin, Y. Zhong, J. Lin, and X. Lin, "Discrete bird swarm algorithm based on information entropy matrix for traveling salesman problem", *Mathematical Problems in Engineering*, Vol. 2018, pp. 1-15, 2018.