International Journal of Recent Development in Engineering and Technology
Website: www.ijrdet.com (ISSN 2347-6435 (Online) Volume 15, Issue 03, March 2026)

An Ant Colony Optimization Based Algorithm to Solve TSP using Pentagonal Fuzzy Travel Cost

Moumita Mondal

Department of Computer Science, Raja N. L. Khan Women's College (Autonomous), Paschim Medinipur, WB, India

Abstract- **In this article I have depicted an ant colony-based, biologically inspired algorithm for solving travelling salesman problem with a fuzzy cost of travelling. In this research I have considered pentagonal fuzzy travelling cost. Here I utilize the "sub-interval addition" method of defuzzification to obtain a crisp equivalent of the pentagonal fuzzy objective. Few benchmark instances have been used to test the optimality of the proposed algorithm. Finally, I have generated some fuzzy instances and solved them with the proposed technique and presented the results.**

Keywords- **Ant colony optimization, Defuzzification, Pentagonal fuzzy number, Travelling Salesman Problem, sub-interval-addition defuzzification.**

I. Introduction

One of the traditional optimization problems in operations research is the Travelling Salesman Problem (TSP). It is one of the most extensively researched topics in OR and was initially examined by a number of applied mathematicians in the 1930s. The shortest path between a given collections of nodes or cities is the aim of the programming optimization problem known as the TSP. Numerous techniques are employed to get approximate solutions because it takes a lot of computer resources to find a precise solution.

II. Literature Review

Zhao et al. (2022) have developed a robust travelling salesman problem with multiple drones, where a truck works in tandem with various kinds of drones to transport goods in situations when routing is unpredictable. Zhang et al. (2022) have presented an asymptotically tight online approach to solve the m-Steiner TSP. Lei and Chen (2022) proposed an enhanced variable neighborhood search method to solve a parallel drone scheduling problem to reduce the amount of time that each truck needs to finish all delivery-related duties. Shi and Zhang (2022) have developed neural network approaches for solving TSP. In reflectance transformation imaging scenarios, a TSP with neighborhoods on a sphere has been presented by Deckerova et al. (2022).

Larasati and Wang (2022) introduced an integrated integer programming model with a simulated annealing technique for the carrier vehicle TSP. The flying sidekick TSP with multiple drops has a novel mathematical formulation, while Mara, Rifai, and Sopha (2022) have proposed a new heuristic method based on Adaptive Large Neighborhood Search. Panwar and Deep (2021) have introduced an innovative discrete Grey Wolf Optimizer (GWO) algorithm called D-GWO that is projected to address the difficult discrete TSP.

Reda et al. (2022) developed a discrete version of the cuckoo search algorithm to solve the TSP. Luo et al. (2022) have proposed a center-based approach to the TSP. Liu et al. (2022) have enhanced the flying sidekick TSP by framing the problem as a Markov decision process and employing stochastic trip timings. Cheikhrouhou and Khoufi (2021) have created a thorough survey on the Multiple TSP. This study aims to bridge the gap by providing a comprehensive review of earlier MTSP research. The hybrid optimization algorithm known as earthworm-based DHOA was formed by Kanna et al. (2021) with the intention of identifying the best solution for the TSP. Schmidt and Irnich (2022) developed a novel neighborhood search technique to solve a generalized TSP.

Listed below is an outline of the presentation's structure for the proposed research: In Section III, I addressed the mathematics necessary to comprehend the given problem. In this article's Section IV, I outline suggested problem. The soft computing method for the specified problem is presented in Section V. Section VI contains the numerical illustration. The paper is concluded at Section VII.

III. Mathematical Preliminaries

3.1 Pentagonal Fuzzy Number (PFN)

The five-tuple (p1, p2, p3, p4, p5) indicates a PFN, an order of fuzzy number in which 'p3' is the highest or most possible value, '(p1, p2)' are the left end points, and '(p4, p5)' are the right end points. According to our survey, in most situations, membership functions are assumed to be linear with symmetry at both ends.

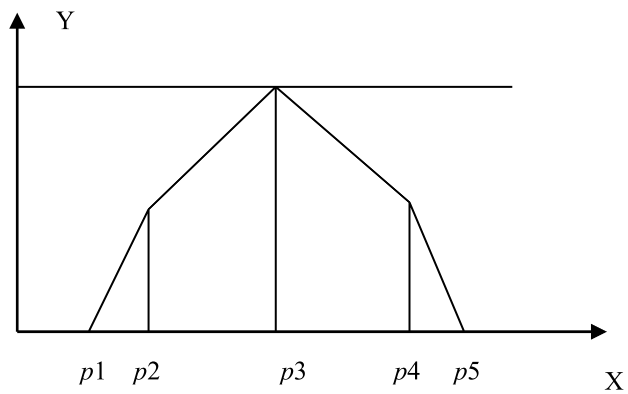

Figure 1: Pentagonal Fuzzy Number

3.2 Ranking Procedure

The x-axis has the discrete real points p_m, $m=1, 2,\ldots, n$.It has Xn number of intervals between the discrete points. Each "sub-interval" has a discrete upper and lower limit (i.e., least upper bound and greatest lower bound of each interval), say, p_r, p_s, $r <= s$, where $1 <= r <= s <= n$ and they could be averaged as $(p_r + p_s)/2$.The average of average of all intervals is our ranking function.

If $m=n$, then the number of intervals is x_n, where m is the corresponding triangular number and it is given by

$x_n = n(n+1)/2$.

Consider an interval fuzzy number $P^2 = (p_1,p_2)$. The intervals in the "interval fuzzy number" P^2 as per the above set rules are (p_1,p_2), (p_1,p_1), (p_2,p_2). The average of the lower and upper limit of each interval is as follows: $(p_1+p_2)/2$, $(p_1+p_1)/2$, $(p_2+p_2)/2$.

The average of all such intervals $= \dfrac{\frac{p_1+p_2}{2}+\frac{p_1+p_1}{2}+\frac{p_2+p_2}{2}}{3}$

$$\mathfrak{R}(P^2) = \frac{3(p_1+p_2)}{6} \ldots\ldots\ldots\ldots \quad (1)$$

In the similar fashion the procedure is extended for further fuzzy numbers.

Consider a pentagonal fuzzy number $P^5 = (p_1, p_2, p_3, p_4, p_5)$ then the ranking function using "sub-interval addition" is

$$\mathfrak{R}(P^5) = 6(p_1 + p_2 + p_3 + p_4 + p_5)/30 \ldots\ldots (2)$$

IV. Problem Formulation

4.1 Classical TSP

In a classical two-dimensional TSP, a salesman has to travel N cities using minimum cost. In his/her tour, the salesman starts from a city, visits all the cities exactly once, and comes to the starting city using minimum cost. Let *T(m ,r)* be the cost for travelling from the m-th city to the r-th city. Then the problem can be mathematically formulated as:

$$\left.\begin{array}{l} \textit{Determine a complete tour} \\ \qquad (y_1, y_2, \ldots, y_N, y_1) \\ \textit{to minimize} \quad M = \sum_{m=1}^{N-1} T(y_m, y_{m+1}) + T(y_N, y_1) \end{array}\right\} \ldots (3)$$

4.2 Proposed fuzzy TSP

In proposed problem if the cost of travel from m-th city to the r-th city is pentagonal fuzzy number, i.e. $\bar{T}(m, r)$ then the above problem can be reduced as follows.

$$\left.\begin{array}{l} \textit{Determine a complete tour} \\ \qquad (y_1, y_2, \ldots, y_N, y_1) \\ \textit{to minimize} \quad \bar{M} = \sum_{m=1}^{N-1} \bar{T}(y_m, y_{m+1}) + \bar{T}(y_N, y_1) \end{array}\right\} \ldots\ldots (4)$$

In this model pentagonal fuzzy cost of travel

$$\bar{T}(m,n) = \{\bar{T}(m,n)_1, \bar{T}(m,n)_2, \bar{T}(m,n)_3, \bar{T}(m,n)_4, \bar{T}(m,n)_5\}$$

4.3 Proposed fuzzy TSP using "sub-interval addition" method

Here I employ the defuzzification strategy to identify the crisp equivalent of the fuzzy objective as fuzzy objective is not generally understood. To produce a crisp equivalent of the pentagonal fuzzy objective, I employ the "sub-interval addition" approach of defuzzification in this study. Using "sub-interval addition" defuzzification, I can reduce the proposed TSP as follows:

$$\text{Determine a complete tour } (y_1, y_2, \dots, y_N, y_1)$$

$$\text{to minimize } \bar{M} = \left\{ \sum_{m=1}^{N-1} \begin{bmatrix} 6 * T(y_m, y_{m+1})_1 + \\ 6 * T(y_m, y_{m+1})_2 + \\ 6 * T(y_m, y_{m+1})_3 + \\ 6 * T(y_m, y_{m+1})_4 + \\ 6 * T(y_m, y_{m+1})_5 + \\ 6 * T(y_m, y_{m+1})_6 \end{bmatrix} /30 + \begin{bmatrix} 6 * T(y_N, y_1)_1 + \\ 6 * T(y_N, y_1)_2 + \\ 6 * T(y_N, y_1)_3 + \\ 6 * 4 + \\ 6 * 6 * T(y_N, y_1)_5 + \\ 6 * T(y_N, y_1)_6 \end{bmatrix} /30; \right\} \quad \dots\dots\dots\dots (5)$$

V. PROPOSED ALGORITHM

5.1 Ant Colony Optimization

The foraging strategies of certain ant species serve as the foundation for ant colony optimization (ACO). In order for the other ants in the colony to follow, these ants locate a good path and mark it with pheromones left on the ground. Ant colony optimization solves optimization problems with a similar technique.

5.2 Procedure of Ant Colony Optimization

Pseudocode is given below for the proposed ACO.

Begin procedure

Initialize necessary parameters and pheromone trials;

While (*m<= Maxiteration*) *do // Maxiteration* is the maximum number of iterations.

a) Construct path of each ant by probabilistic method
b) Determine the fitness values for every ant
c) Find best solution through selection methods
d) Renew pheromone trial for each path
e) Evaporation pheromone trial

Next m

End procedure

5.3 Description of the algorithm

Description of the proposed ACO is as below.

Construction of path: Assume that node i is the a^{ω} ant's present location. An ant locates the next city (let's say the *j-th* city) using a probabilistic selection. Accordingly, an ant will choose the next city $j \epsilon NV$ with a probability b_{ij} following the Roulette–Wheel selection procedure (Michalewicz, 1996). The below formula is used to generate the value of b_{ij} (Engelbrech, 2005):

$$b_{ij} = \frac{TAO_{ij}^{\alpha}}{\sum_{j \epsilon NV} TAO_{ij}^{\alpha}}$$

where NV is the set of locations the ant does not visit and α is a positive variable used to regulate the pheromone attentiveness. The city will be removed from the set NV of unvisited cities if it is selected. The value of α has a range of [1.6, 2.0].

Pheromone Evaporation: Evaporation causes the ants to forget previous decisions. The evaporation of pheromone is performed by the following assignment statement:

$$\tau_{ij} = (1 - \rho)\tau_{ij}$$

Where $\rho \in [0,1]$ and the constant, ρ specifies the rate at which pheromone evaporates. Here, to solve the problem, the value of ρ is taken as 0.01 (Engelbrecht, 2005).

Pheromone Updating: The pheromone on the ants travel routes is raised once every ant has built their whole tour. For this path, each pair PATH [k][i] to PATH [k][i+1] is increased by (1/COST(k)) if COST(k) is the cost of PATH[k]. The pheromone on the route and the intuitive content between city-*m* and city-*r* determine the state change probability. It is determined as follows:

$$R_{mr}^{s}(t) = \begin{cases} \dfrac{[\tau_{mr}(t)]^{\alpha}[\eta_{mr}]^{\beta}}{\sum_{\epsilon \text{ allowe } d_s}[\tau_{mr}(t)]^{\alpha}[\eta_{mr}]^{\beta}} & r \epsilon \text{ allowe } d_s \\ 0 & \text{otherwise} \end{cases} \quad \dots. (6)$$

here τ_{mr} is the intensity of pheromone trail of linking nodes m and r, and α is the parameter to control the influence of pheromone, η_{mr} is the visibility of node-r from node-m, which is always set as $1/d_s$ (d_s is the cost of travel between cities m and r), and β is another parameter to adjust the influence of η_{mr}, respectively.

International Journal of Recent Development in Engineering and Technology
Website: www.ijrdet.com (ISSN 2347-6435 (Online) Volume 15, Issue 03, March 2026)

The trail levels are updated as on a tour each ant leaves pheromone quantity given by Q/*cost*(k), where Q is a constant and *cost*(k) is the total cost of its tour, respectively. In addition, as time passes, the pheromone will evaporate. In such cases, the τ_{mr} is changed in the following ways.

$$\left.\begin{array}{l} \tau_{mr}(t+1) = \rho.\tau_{mr}(t) + \Delta\tau_{mr} \\ \Delta\tau_{mr} = \sum_{k=1}^{l} \Delta\tau_{mr} \ and \ \Delta\tau_{mr} = \\ \begin{cases} \frac{Q}{cost(k)} & if\ travers\ through\ arc\ (m,r) \\ 0 & otherwise \end{cases} \end{array}\right\} \quad \ldots\ldots\ldots (7)$$

Here t is iteration number and ρ is the parameter for pheromone change and its value is in range of [0, 1]. Now $\Delta\tau_{mr}{}^{k}$ is the increment in trail level on arc (m, r) caused by ant k.

Evaporation is made by the following ways:

$$\Delta\tau_{mr} = (1-\sigma)\Delta\tau_{mr} \ \ldots\ldots\ldots\ (8)$$

where σ is a value in the range of [0,1).

Implementation: All computational experiments are performed using a Core i3 processor, Windows XP OS, 2GB RAM, and Dev C++ 4.9.9.2.

VI. Experimental Results

6.1 Performance testing

Here I have considered five benchmark instances from TSPLIB as described by Wang et al. (2004). We have presented the best results by the proposed ACO and compared with GA and Self adoptive GA (SAGA). We have presented the results in Table 1.

No of iteration for the proposed ACO is directly given in the result table. For ACO the value of α=1.6 and other ACO parameters are given in the result Table. For evaporation the value of σ=0.01.

Table 1:
Results of TSPLIB datasets for slandered five cases.

Instance name	Optimal	Proposed ACO parameters	Results by different algorithms[Wang et al., 2004]		
			By GA	By SAGA	By proposed ACO
br17	39	# ants: 17 # iterations: 500	39	39	39
ftv33	1286	# ants: 33 # iterations: 1000	1349	1353	1343
ftv55	1608	# ants: 55 # iterations: 1500	1672	1663	1635
ry48p	14422	# ants: 48 # iterations: 1500	15024	14915	14915
ft70	38673	# ants: 70 # iterations: 2000	41056	39702	39702

6.2 Experiments with a small size real instance

In this subsection I have considered a small sized i.e., 10-city problem with pentagonal fuzzy cost of travel. The input cost matrix is randomly generated and presented in Table 2. The optimal path obtained for this dataset by the proposed ACO has been given in Table 3. The number of ants and the maximum iteration are considered as 10 and 500, respectively. The other ACO parameter values are the same as in the previous subsection.

International Journal of Recent Development in Engineering and Technology
Website: www.ijrdet.com (ISSN 2347-6435 (Online) Volume 15, Issue 03, March 2026)

Table 2:
10-city pentagonal fuzzy input time matrix for TSP.

i/j	1	2	3	4	5	6	7	8	9	10
1	∞	(2.9,3.1,4.4,5.6,6.1)	(6.1,6.2,7.6,9.3,10.0)	(6.2,6.4,7.5,9.8,10.0)	(4.5,4.6,6.5,8.2,8.7)	(4.1,4.4,6.0,7.3,7.5)	(4.3,4.6,6.0,8.1,8.3)	(4.5,4.6,6.3,8.1,8.6)	(6.8,7.2,8.5,9.5,10.8)	(5.7,6.2,7.5,9.0,9.5)
2	(5.6,6.2,7.3,8.7,9.6)	∞	(5.1,5.1,6.8,7.8,8.9)	(6.5,6.5,8.2,10.0,10.4)	(4.5,4.7,5.7,7.1,7.9)	(2.8,3.7,4.7,6.1,6.7)	(4.4,4.5,6.3,7.9,8.4)	(4.7,4.8,5.8,7.4,7.7)	(5.5,6.1,7.2,9.3,9.6)	(6.2,6.8,8.1,9.6,10.5)
3	(7.0,7.2,8.6,10.1,10.4)	(5.6,5.6,6.7,8.5,9.1)	∞	(6.5,7.0,8.1,10.1,10.2)	(6.6,7.3,8.6,10.1,10.6)	(2.4,2.6,4.3,5.4,5.7)	(6.3,6.5,8.1,9.2,9.9)	(7.5,7.7,8.8,10.4,10.8)	(6.1,6.2,7.8,8.9,8.9)	(3.9,4.7,5.8,7.9,8.3)
4	(5.0,5.1,6.2,7.2,7.7)	(4.6,5.1,6.3,7.9,8.8)	(6.7,7.3,8.5,9.8,10.4)	∞	(4.2,4.7,6.2,7.2,7.8)	(3.1,3.4,5.0,7.0,7.1)	(4.5,4.8,6.1,7.9,7.9)	(4.9,5.4,6.5,8.0,8.4)	(3.7,4.2,5.6,6.6,7.5)	(3.0,3.1,4.3,6.0,6.3)
5	(3.8,4.5,5.5,7.1,8.0)	(6.7,6.8,8.6,9.8,10.3)	(5.7,6.0,7.6,9.3,9.8)	(4.6,4.9,6.2,7.9,8.4)	∞	(3.7,4.0,5.7,7.3,7.6)	(3.8,3.9,5.4,6.8,7.6)	(3.4,4.2,5.3,7.2,7.7)	(6.0,6.3,7.5,8.6,9.0)	(5.1,5.3,6.7,8.3,8.4)
6	(3.5,3.7,5.1,7.0,7.3)	(6.4,6.7,8.0,9.0,10.1)	(3.3,3.5,4.9,6.4,6.7)	(3.2,3.2,4.4,6.4,6.8)	(3.2,3.3,5.1,6.3,6.9)	∞	(3.5,3.9,5.1,6.2,6.6)	(5.4,5.9,7.2,8.9,9.5)	(6.4,7.1,8.3,9.9,10.6)	(7.2,7.6,8.9,9.9,10.9)
7	(4.9,5.3,6.8,7.9,8.4)	(5.9,6.0,7.5,8.6,9.8)	(6.6,6.8,8.2,9.6,10.4)	(4.6,5.2,6.2,7.5,7.6)	(3.0,3.2,4.6,6.7,6.9)	(7.4,7.4,8.6,9.9,10.7)	∞	(7.4,7.5,8.8,10.4,10.8)	(5.3,5.4,6.8,8.5,8.6)	(3.2,3.8,4.9,6.1,7.0)
8	(4.2,4.9,6.2,8.0,8.2)	(4.9,5.3,6.8,8.1,9.2)	(3.7,3.8,5.1,6.3,6.5)	(5.0,5.2,6.5,7.7,8.6)	(5.8,6.2,7.5,8.9,9.7)	(4.3,4.5,5.7,6.8,8.0)	(6.1,6.2,8.0,9.5,10.1)	∞	(4.1,4.2,5.7,7.4,8.2)	(3.7,3.7,4.9,6.1,6.1)
9	(5.3,5.5,6.7,7.8,8.4)	(5.9,6.0,7.3,8.3,8.7)	(3.2,3.4,4.9,6.2,6.9)	(4.2,4.6,5.7,6.7,6.9)	(5.9,6.0,7.9,9.2,9.5)	(5.7,5.7,7.1,8.6,9.4)	(3.9,4.4,5.8,7.0,8.0)	(3.0,3.2,4.5,5.8,6.3)	∞	(6.1,6.6,7.8,9.1,9.8)
10	(7.3,7.7,8.8,10.1,10.8)	(6.4,6.9,8.0,9.8,9.9)	(6.6,7.5,8.5,9.6,9.8)	(2.7,2.8,4.2,6.1,6.3)	(2.9,3.2,4.3,5.7,5.8)	(2.6,3.3,4.5,6.0,6.1)	(7.5,7.6,8.8,10.1,11.0)	(5.6,5.8,7.5,9.0,9.5)	(5.1,5.6,6.8,8.7,8.8)	∞

Table 3:
Result of 10-city fuzzy instance using defuzzification technique.

Instance name	Obtained path by proposed ACO	Optimal cost(Crisp equivalent)
10-city pentagonal fuzzy instance	1, 2, 7, 10, 4, 9, 8, 3, 6, 5	50.13

6.3 Experiments with randomly generated instances

In this sub-section, I have generated a few random pentagonal fuzzy instances of different sizes. Say a fuzzy cost has five components ($p1$, $p2$, $p3$, $p4$, $p5$) . At first, random data is generated in the range of [20, 30] and assumed as $p3$. Then two random numbers, b and c, are generated (b and c are both in the range [1, 2]). At this point, $p2$ equals $p3$-b, and $p1$ equals $p3$-c.

International Journal of Recent Development in Engineering and Technology
Website: www.ijrdet.com (ISSN 2347-6435 (Online) Volume 15, Issue 03, March 2026)

Again, two random numbers, *d* and *e*, are generated in the range of [1, 2]. Now, $p4=p3+d$ and $p5=p3+e$. In this way, every pentagonal fuzzy cost is generated. The ACO parameters for these instances have been given in the result table. The values of α and σ are 1.6 and 0.01, respectively, for all the instances of this subsection. Due to large size the input datasets are not presented in the result section, and only optimal path and crisp equivalent cost using "sub-interval addition" of defuzzification have been presented.

Table 4:
Results of randomly generated instances.

Instance	ACO parameters	Optimal/ best path by proposed ACO	Obtained cost of tour
Randins 25	# of ants: 25 # of iteration: 1000	1, 12, 22, 23, 14, 20, 11, 16, 2, 9, 7, 18, 5, 4, 21, 15, 24, 25, 6, 13, 8, 10, 3, 19, 17	536.73
Randins 50	# of ants: 50 # of iteration: 1500	1, 31, 10, 2, 22, 15, 14, 24, 38, 6, 21, 41, 36, 43, 26, 4, 13, 40, 9, 46, 32, 50, 33, 44, 35, 39, 17, 23, 42, 5, 11, 20, 25, 45, 7, 27, 30, 29, 47, 18, 16, 37, 48, 8, 34, 12, 3, 28, 19, 49	1110.07
Randins 75	# of ants: 75 # of iteration: 2000	1, 18, 60, 56, 2, 50, 25, 52, 68, 17, 64, 10, 42, 51, 63, 62, 66, 40, 3, 28, 8, 15, 29, 5, 6, 26, 33, 23, 19, 34, 70, 54, 39, 30, 12, 31, 72, 24, 32, 47, 46, 61, 45, 74, 65, 43, 53, 44, 20, 16, 67, 22, 7, 48, 58, 75, 69, 73, 57, 35, 38, 21, 27, 4, 36, 9, 55, 14, 49, 71, 37, 11, 41, 13, 59	1638.28
Randins 100	# of ants: 100 # of iteration: 2500	1, 65, 25, 30, 28, 99, 12, 76, 40, 82, 32, 22, 49, 61, 98, 10, 48, 20, 17, 39, 91, 90, 53, 66, 29, 64, 73, 72, 11, 6, 3, 45, 67, 24, 7, 35, 41, 74, 21, 9, 42, 44, 43, 62, 52, 26, 89, 88, 19, 79, 95, 84, 68, 37, 13, 75, 70, 34, 33, 16, 80, 36, 81, 100, 96, 38, 63, 55, 59, 57, 51, 77, 8, 93, 14, 94, 27, 58, 4, 78, 23, 83, 69, 97, 54, 18, 60, 2, 50, 5, 85, 46, 56, 92, 15, 86, 87, 71, 47, 31	2148.24

VII. CONCLUSION

The proposed research solved a pentagonal fuzzy TSP using an ACO based approach. The novelty of this research is the solution of pentagonal fuzzy TSP with an ACO based algorithm. A sub-interval method of defuzzification has been used to find the crisp equivalent of the TSP. The proposed algorithm is tested with TSPLIB instances. Few randomly generated instances have also been considered for solving. One drawback is that all solved instances are asymmetric but not considered any symmetric datasets. In the future I want to solve a symmetric TSP using the bat algorithm in a hexagonal fuzzy environment.

REFERENCES

[1] Zhao, L., Bi, X., Li, G., Dong, Z., Xiao, N., & Zhao, A. (2022). Robust traveling salesman problem with multiple drones: Parcel delivery under uncertain navigation environments, Transportation Research Part E: Logistics and Transportation Review, 168,102967.https://doi.org/10.1016/j.tre.2022.102967.

[2] Zhang, Y., Zhang, Z., Liu, Z., & Chen, Q. (2022). An asymptotically tight online algorithm for m-Steiner Traveling Salesman Problem, Information Processing Letters, 174,106177. https://doi.org/10.1016/j.ipl.2021.106177.

[3] Lei, D., & Chen, X. (2022). An improved variable neighborhood search for parallel drone scheduling traveling salesman problem, Applied Soft Computing, 127,109416. https://doi.org/10.1016/j.asoc.2022.109416.

[4] Shi, Y., & Zhang, Y. (2022). The neural network methods for solving Traveling Salesman Problem, Procedia Computer Science, 199, 681-686. https://doi.org/10.1016/j.procs.2022.01.084.

[5] Deckerova, J., Faigl, J., & Kratky, V. (2022). Travelling salesman problem with neighborhoods on a sphere in reflectance transformation imaging scenarios. Expert Systems with Applications, 198, 116814. https://doi.org/10.1016/j.eswa.2022.116814.

[6] Larasati, M. R., & Wang, I-L. (2022). An integrated integer programming model with a simulated annealing heuristic for the carrier vehicle traveling salesman problem, Procedia Computer Science, 197, 301-308. https://doi.org/10.1016/j.procs.2021.12.144.

[7] Mara, S. T. W., Rifai, A.P., & Sopha, B. M. (2022). An adaptive large neighborhood search heuristic for the flying sidekick traveling salesman problem with multiple drops, Expert Systems with Applications, 205, 117647. https://doi.org/10.1016/j.eswa.2022.117647.

[8] Panwar, K., & Deep, K. (2021). Discrete Grey Wolf Optimizer for symmetric travelling salesman problem, Applied Soft Computing, 105, 107298**.** https://doi.org/10.1016/j.asoc.2021.107298

[9] Reda, M., Onsy, A., Elhosseini, M.A., Haikal, A. Y., & Badawy, M. (2022). A discrete variant of cuckoo search algorithm to solve the Travelling Salesman Problem and path planning for autonomous trolley inside warehouse, Knowledge-Based Systems, 252, 109290. https://doi.org/10.1016/j.knosys.2022.109290.

[10] Luo, Y., Golden, B., Poikonen. & Zhang, R. (2022). A fresh look at the Traveling Salesman Problem with a Center, Computers & Operations Research, 143,105748. https://doi.org/10.1016/j.cor.2022.105748.

[11] Liu, Z., Li, X., & Khojandi, A. (2022). The flying sidekick traveling salesman problem with stochastic travel time: A reinforcement learning approach, Transportation Research Part E: Logistics and Transportation Review, 164, 102816. https://doi.org/10.1016/j.tre.2022.102816.

[12] Cheikhrouhou, O., & Khoufi, I. (2021). A comprehensive survey on the Multiple Traveling Salesman Problem: Applications, approaches and taxonomy, Computer Science Review, 40, 100369. https://doi.org/10.1016/j.cosrev.2021.100369.

[13] Kanna, S.K.R., Sivakumar, K., & Lingaraj, N. (2021). Development of Deer Hunting linked Earthworm Optimization Algorithm for solving large scale Traveling Salesman Problem, Knowledge-Based Systems, 227, 107199. https://doi.org/10.1016/j.knosys.2021.107199.

[14] Schmidt, J., & Irnich, S. (2022). New neighborhoods and an iterated local search algorithm for the generalized traveling salesman problem, EURO Journal on Computational Optimization, 10, 100029. https://doi.org/10.1016/j.ejco.2022.100029.

[15] Wang, J., Huang, J., Rao, S., Xue, S., & Yin, J. (2008). An adaptive Genetic Algorithm for Solving Traveling Salesman Problem, Springer-Verlag Berlin Heidelberg 2008 , ICIC 2008, LNAI 5227, 182-189. https://doi.org/10.1007/978-3-540-85984-0_23

[16] Engelbrech AP (2005). Fundamentals of computational swarm intelligence. Wiley, Hoboke.