TWMS J. App. and Eng. Math. V.15, N.12, 2025, pp. 2742-2751

ANT COLONY OPTIMIZATION FOR SOLVING TSP WITH SUB-ROUTE ELIMINATION CONSTRAINTS ON TÜRKIYE MAP

F. NURIYEVA1*, V. ERDEMCI[2], §

Abstract. The Traveling Salesman Problem is the famous optimization problem in the NP-hard class. Many problems with applications in computer science and engineering can be modeled using the Traveling Salesman Problem. In this study, one of the artificial intelligence techniques, ant colony method, is used to solve the traveling salesman problem. In the study applied on the map of Türkiye, it is aimed to plan the best route.

Keywords: Symmetric Traveling Salesman Problem, Artificial Intelligence, Ant Colony Algorithm, Metaheuristics, Sub-Route Elimination Constraint.

AMS Subject Classification: 90C27, 68T20

1. Introduction

The Traveling Salesman Problem (TSP) is a well-known combinatorial optimization problem studied in operations research and computer science [4]. The TSP aims to find the shortest or least-cost tour that passes through each of n points (such as a city or a node) only once with known distances between them. In graph theory, TSP can be defined as finding the least-cost Hamiltonian circuit in a given weighted graph (where nodes, i.e., vertices, are cities, edges are paths between cities, and weights are the cost or length of the path) [13].

TSP has many transportation and logistics applications, such as determining the location of base stations of GSM operators, material flow system design, vehicle routing, scheduling of crane routes in warehouses, material picking in stock areas, airport routing for airplanes, electronic circuit design.

There are different variants of TSP [10]. Symmetric TSP - when the distance matrix is symmetric. Assymmetric TSP - where the distance matrix is not symmetric, i.e. the distance from city i to city j is not equal to the distance from city j to city i. Multiple TSP - in this problem there is more than one traveling salesman. Suppose we have n cities and m traveling salesmen: These n cities are divided among m traveling sellers,

[1] Yaşar University, Department of Mathematics, Izmir, Türkiye.
e-mail: fidan.nuriyeva@yasar.edu.tr; ORCHID: https://orcid.org 0000-0001-5431-8506.
[2] Denizbank Intertech, Istanbul, Türkiye.
e-mail: verdemci@gmail.com; ORCHID: https://orcid.org 0000-0002-3613-6044.
* Corresponding author.
§ Manuscript received: December 21, 2024; accepted: April 10, 2025.
TWMS Journal of Applied and Engineering Mathematics, Vol.15, No.12;

provided that they do not have a common city. There are two variants of this problem: each traveling salesman is located in the same initial city, and each traveling salesman is located in different cities at the initial time. The problem requires the optimal ordering of the cities on the sellers' tours. Each of the above problems are closed problems, in the sense that the traveling salesman must eventually return to the origin. There are also open traveling salesman problems. An open TSP is one in which the traveling salesman does not return to the starting city. There are four variants of this problem: initial and final cities are given in advance, initial city is given in advance, final city is given in advance, and neither initial nor final city is given in advance.

In this study, the symmetric traveling salesman problem is considered. In symmetric TSP, for each pair of cities, the distance from city i to city j is equal to the distance from city j to city i.

The integer linear programming model for the symmetric TSP is shown below [15]:

$$\sum_{i=1}^{n} \sum_{j=1, j\neq i}^{n} , d_{ij} x_{ij} \longrightarrow min \tag{1}$$

$$\sum_{i=1, i\neq j}^{n} x_{ij} = 1, \quad j = 1, ..., n \tag{2}$$

$$\sum_{j=1, j\neq i}^{n} x_{ij} = 1, \quad i = 1, ..., n \tag{3}$$

$$u_i - u_j + n x_{ij} \leq n - 1; \quad u_i \geq 0; \quad i, j = 2, ..., n; \quad i \neq j; \tag{4}$$

$$x_{ij} \in \{0, 1\} \quad i, j = 1, ..., n; \quad i \neq j \tag{5}$$

$$x_{ij} = \begin{cases} 1, & \text{if the salesman travels from city } i \text{ to city } j \\ 0, & \text{otherwise} \end{cases}$$

In the above equation, d_{ij} is the distance (cost) between city i and city j, n is the number of cities, x_{ij} is a binary decision variable representing the connections between cities, and u_i is the auxiliary variable (used to block sub-tours). Equation (1) represents the objective function, which defines the total travel cost of the optimal tour. Constraints (2) and (3) ensure that each city is visited exactly once and departed exactly once, respectively. Constraint (4) is the subtour elimination constraint, which prevents the formation of disconnected cycles using the Miller-Tucker-Zemlin (MTZ) formulation. Finally, Constraint (5) enforces the binary nature of the decision variables [17].

Although the problem definition is simple, its solution is difficult. As the number of cities used in the problem increases, the solution space expands and the solution time and difficulty of the problem increases [11, 12].

In this study, the TSP problem is solved with the Ant Colony method and a suitable route is found on the map of Türkiye. In the Introduction section of the study, the definition, types, application areas and mathematical model of TSP are given. In the second section; some solution approaches for TSP are mentioned. In the third section; Ant Colony Metaheuristic and some solutions with ant colony algorithm is explained. In the fourth section; TSP is solved with Ant Colony Metaheuristic. In the fifth section, the Ant Colony algorithm is applied on the map of Türkiye is given. In the last part of

the paper, the results and evaluation section summarizes the work done and presents the general results obtained for the algorithm.

2. Solution Approaches for the Traveling Salesman Problem

The Traveling Salesman Problem is a combinatorial optimization problem and is known to be NP-Hard [9]. As the problem size increases, the number of possible rounds increases and it becomes difficult or even impossible to solve the problem in an exact short time. Exact, approximate and heuristic algorithms have been developed to solve the TSP [19].

Exact Algorithms: These algorithms are usually approaches derived from the integer linear programming formula of the TSP. An example of this approach is the "Branch and Bound" algorithm. The first method that comes to mind is the enumeration method where every possibility is tried. Other methods are linear programming methods, dynamic programming methods. However, these algorithms are computationally expensive [1].

Approximation methods: Since TSP is an NP-hard problem, it is also a hard problem to solve. Therefore, heuristic and approximate algorithms are needed. Approximation algorithms do not guarantee a solution, but they differ from heuristics in that they guarantee how far we are from the optimal solution. The best known approximate algorithms for TSP are the "Christofides Algorithm" (guarantee value 3/2), "Minimum-Spanning Tree (MST)" based algorithms (guarantee value 1/2), and others [20].

Heuristic Algorithms: Exact algorithms may not work efficiently. In this case, we can find a solution close to the ideal solution without using exact algorithms. In practice, heuristic algorithms are preferred over exact algorithms. We can divide the heuristic algorithms that solve the TSP into three categories: Tour-generating heuristics, Tour-improving heuristics and heuristics that use a hybrid of these two methods [20].

- **Tour-generating heuristics:** The common feature of tour-generating algorithms is that once they find a result, they do not try to improve it. At this point, the algorithms stop working. Known tour-generating heuristics are the "Nearest Neighbor", "Greedy Technique", "Insertion Heuristic" and "Christofides" algorithms [14, 20].
- **Tour-improving heuristics:** These algorithms aim to improve the tour. Examples of these algorithms include local optimization algorithms such as "2-opt", "3-opt" and "Lin-Kernighan" as well as Artificial Intelligence methods such as "Tabu Search", "Genetic Algorithms", "Simulated Annealing" and "Ant Colony Algorithm" [20].
- **Hybrid Methods:** Algorithms that combine both tour generating and tour improving heuristics. "Iterated Lin-Kernighan" is an example. The most successful results are obtained from hybrid methods [20].

Metaheuristics: Metaheuristic Algorithms are computational techniques for solving 'hard' optimization problems that attempt to incrementally improve a candidate solution (or solutions) generated by a particular method based on a quality measure. Meta-heuristic algorithms consider the heuristic approach to the problem as a black box and are not interested in the details of the algorithm developed for the solution. All it does is to try to optimize one or more functions used in the solution. These functions are called goal functions or objective functions. Examples of Metaheuristic Algorithms are Artificial Intelligence methods such as Tabu search, Genetic algorithms, Simulated Annealing, Artificial neural networks and Ant Colony Algorithm.

3. Ant Colony Algorithm

The use of natural processes in determining heuristics for solving industrial problems is becoming increasingly common. This is because these systems, which consist of a large number of simple individuals, exhibit a very complex structure as a whole. Ant colonies are an example of such systems. The Ant Colony Algorithm (ACA) is an algorithm based on mathematical models of real ant colony behavior. The first work was done by Dorigo et al. (1991), who described their system as an "ant system" and the resulting algorithm as an "ant algorithm". The proposed algorithms are slightly different from real ant behaviors, since artificial ant colonies are considered as an optimization tool instead of modeling the behavior of ant colonies exactly [6].

For example, artificial ants have certain memory and, unlike real ants, are not completely blind. At the same time, artificial ants live in a discrete-time environment. The most basic and most important point in the description of any distributed system is to express how communication between individuals is done.

Dorigo et al. (1991) introduced the ant algorithm as an approach to distributed solution of difficult problems based on simulating the behavior of a large number of locally interacting simple individuals. There are many algorithms derived from the Ant Colony Meta heuristic that have been used to solve various problems. Many combinatorial optimization problems have been solved with these algorithms [6].

In the literature review on solving the traveling salesman problem (TSP) with ant colony optimization (ACO), there are many academic studies that present different approaches and improvements. In this literature, various versions such as the classical ant colony algorithm (Ant System (AS)), elitist ant system (EAS), Min-Max ant system (MMAS), and hybrid approaches are examined. Sample literature studies are as follows:

Dorigo, M., Maniezzo, V., and Colorni, A. (1996) introduced the original version of the Ant Colony System, which is one of the most cited works. This approach enables each ant to discover the shortest route for solving the TSP by marking solution paths with a pheromone trail. It has served as the foundation for numerous subsequent algorithms [8].

Dorigo, M., and Gambardella, L. M. (1997), in their work on the "Ant Colony System," propose a derivative of the popular Ant Colony Optimization (ACO) method. In this system, modifications to pheromone evaporation and the search process lead to shorter solution paths for the TSP [7].

Bullnheimer, B., Hartl, R. F., and Strauss, C. (1999), in their work on "A New Rank-Based Version of the Ant System," address pheromone updates using a rank-based approach. This method significantly enhances the speed of convergence to the optimal path in TSP solutions [3].

Bonabeau, E., Dorigo, M., and Theraulaz, G. (1999), in their book Swarm Intelligence: From Natural to Artificial Systems, explore a wide range of applications of the ant colony algorithm and provide a comprehensive review of the theoretical foundations of ACO for solving the TSP [2].

Stützle, T., and Hoos, H. H. (2000), in their work on the "Max-Min Ant System", demonstrate that better performance can be achieved by introducing lower and upper limits on the pheromone levels to enhance the classical Ant System (AS) method. This approach is regarded as one of the most significant updates for improving solution quality in the TSP [25].

Shang et al. (2007) evaluated the effectiveness of ant colony optimization (ACO) by applying it to larger and more complex TSP problems. They proposed performance enhancements for ACO by experimenting with different strategies for updating pheromone trails, resulting in faster solutions [21].

In Söyler H. and Keskintürk T. (2007), the Ant Colony Algorithm (ACA) is introduced as a heuristic method for solving problems such as the Traveling Salesman Problem (TSP). Inspired by the behavior of ants, who find the shortest path by leaving pheromone trails, this algorithm provides effective solutions for both symmetric and asymmetric TSPs. The shortest path is explored through iterations using artificial ants and pheromone updates. The aim of this study is to introduce the Ant Colony Algorithm in logistics and distribution and to compare the results with an example problem [23].

Zheng et al. (2023) apply the ant colony algorithm to optimize urban logistics distribution, showing via MATLAB-based analysis that it effectively reduces costs and enhances transport efficiency, contributing to the literature on urban distribution optimization [29].

Zarei et al. (2016) propose a two-phase hybrid metaheuristic algorithm (MACSGA) for the TSP, combining a modified ant colony system (MACS) for initial solutions with a modified genetic algorithm and 2-opt local search for improvement. The approach prevents premature convergence and outperforms GA, ACO, and other metaheuristics in terms of efficiency [27].

Zhang et al. (2011) introduce a hybrid artificial bee colony algorithm (ABC & PR) for the TSP, combining ABC's solution construction with path relinking (PR). This approach enhances solution quality by balancing diversification and convergence, outperforming existing methods on benchmark TSP instances [28].

Dikmen et al. (2014), in their study, compared the performance of Ant Colony Optimization (ACO) with Genetic Algorithms for solving the TSP on the map of Turkey. The research aimed to optimize route planning by evaluating both the route distance and the computational time. The study found that ACO outperformed the Genetic Algorithm in terms of both distance and computational time. After 1000 iterations, the shortest route distance found by the ant colony on the map of Turkey was 9966 km [5].

Mohsen (2016) proposes a hybrid ACO algorithm to solve the TSP, integrating the advantages of ACO, SA, mutation operator, and local search. While retaining the core benefits of ACO, SA and mutation increase the diversity of the ant population, and local search efficiently explores the search area. Experiments on 24 TSP instances show that the proposed algorithm outperforms some well-known algorithms in terms of solution quality [18].

Şenaras and Inanç (2017), in their paper, also address the TSP using ACO, although not specifically focusing on the Turkish map. However, their methodologies could be adapted to Turkey-specific data, as they explore various aspects of ACO for route optimization. In their study on ant colony optimization for solving the TSP, the shortest path found was 2529 km in 50 iterations for 20 provinces on the map of Turkey. The provinces used in the experiment are: Ankara, Düzce, Kocaeli, Uşak, Aydın, Edirne, Kütahya, Yalova, Bolu, Eskişehir, Manisa, Balıkesir, İstanbul, Muğla, Çanakkale, İzmir, Sakarya, Denizli, Kırklareli, and Tekirdağ [26].

Siemiński (2016) proposes hyper-populated ant colonies to reduce the non-determinism and computational cost of ACO for TSP. The study shows that increasing colony size and parallelizing via sockets or RMI significantly improves performance, with promising potential for dynamic TSP variants [22].

Manfrin et al. (2006) investigate the impact of communication in parallel ACO for the TSP. They compare synchronous and asynchronous strategies and find that even simple

independent runs can be surprisingly effective, offering insights into the role of minimal communication [16].

Skinderowicz (2022) proposes a novel ant colony optimization variant, Focused ACO (FACO), to improve the efficiency of solving large-scale TSP instances. FACO guides the search process by limiting differences between newly constructed and previous solutions, resulting in more focused exploration and better integration with local search. Experimental results on large TSP benchmarks demonstrate that FACO outperforms state-of-the-art ACO methods, achieving solutions within 1% of the best-known results using multi-core CPUs [24].

4. Solving the Traveling Salesman Problem with Ant Colony Algorithm

First, the number of ants in the ant colony is determined. Then, each ant is randomly placed in a node and completes its tour by visiting all nodes one by one.

The steps of the algorithm are as follows:

1. Initial pheromone values and other parameters are determined.
2. Ants are randomly placed at each point.
3. The ants choose the next point with the probability value obtained from the given equation and complete the tour.
4. The length of the paths traveled by each ant is calculated, the pheromone values are updated.
5. The best solution is calculated, saved.
6. Go to step 2 until the maximum number of iterations is reached.

4.1. Parameters of Ant Colony Algorithm.

Number of Ants: This parameter determines how many ants will be in the colony.

Number of Iterations: This parameter determines how many iterations (steps) the search process will take.

Pheromone Coefficient (α): A parameter that determines the importance of pheromone amounts between nodes.

Coefficient of Heuristic Value (β): It is the parameter that determines the importance of the distance between nodes.

Coefficient of Pheromone Evaporation (ρ): It is the parameter that determines the rate at which pheromones between nodes will evaporate at the end of each iteration.

α : pheromone coefficient

β: coefficient of the heuristic value

τ: pheromone quantity

η :heuristic information (inverse of distance)

ρ : pheromone evaporation coefficient ($0 < \rho < 1$)

Probability function to determine the next point, based on pheromone values is calculated as:

$$a_{ij}(t) = \frac{[\tau_{ij}(t))]^{\alpha} \left[\frac{1}{d_{ij}}\right]^{\beta}}{\sum_{l \in N_{ik}} [\tau_{ij}(t))]^{\alpha} \left[\frac{1}{d_{ij}}\right]^{\beta}}$$

$$p_{ij}^{k}(t) = \frac{a_{ij}(t)}{\sum_{l \in N_{tk}} a_{il}(t))}$$

To update the pheromone trails after each ant has produced a result

$$\tau_{ij}(t+1) = (1-\rho) \cdot \tau_{ij}(t) + \sum_{k=1}^{m} \Delta\tau_{ij}^{k}(t)$$

formula will be applied.

$$\Delta\tau_{ij}^{k}(t) = \begin{cases} \dfrac{1}{L^{k}(t)}, & \text{from city } i \text{ to city } j \\ 0, & \text{otherwise} \end{cases}$$

Here, k is the amount of pheromone left on the edge visited by the ant.

5. Application of Ant Colony Algorithm on Türkiye Map

The calculations on the provinces of Türkiye were based on the distances indicated in the distance table between provinces shared on the website of the General Directorate of Land Roads of the Ministry of Transportation of the Republic of Türkiye [30].

To ensure reliable performance analysis, the algorithm was executed 10 times for each instance. During these executions, the best, worst, and average costs, along with the standard deviation and running time, were recorded to assess the algorithm's consistency.

In the algorithm, each ant's position is stored within a structure in the code. The positions of the ants are represented by cities. This structure contains information to track the sequence of cities visited by each ant and the cities that have already been visited. Initially, each ant is randomly placed at a city, and a corresponding structure is created. This structure includes the sequence of cities the ant has visited, the total length of the tour taken, and a logical array indicating the visited cities. This allows the position and progress of each ant to be tracked.

The ants are initially placed at specific cities. During the transition between cities, the selection of the next city is determined using a probabilistic method, such as the roulette wheel technique.

The values of the parameters α and β, along with other parameters, are provided in Table 1. The best solution found had a cost of 9855.00, while the worst solution reached

Table 1. Parameters and values used in the Ant Colony Algorithm

Parameter	**Value**
α	1.2
β	4.5
ρ	0.6
Q	1000
Number of Ants	100
Number of Iterations	200

a cost of 11638.00. The average execution time per run was 226.48 seconds, resulting in a total execution time of 2264.77 seconds. The average cost across all runs was 10697.55, with a standard deviation of 865.30, indicating significant variation in the results.

Based on the distances between cities and the parameters listed in Table 1, the total distance is calculated as 9855 km, and the tour is as follows:

Malatya → Elazığ → Bingöl → Muş → Bitlis → Van → Hakkâri → Şırnak → Siirt → Batman → Diyarbakır → Mardin → Şanlıurfa → Adıyaman → Kahramanmaraş → Gaziantep → Kilis → Hatay → Osmaniye → Adana → Mersin → Karaman → Konya → Aksaray → Niğde → Nevşehir → Kayseri → Sivas → Tokat → Amasya → Çorum → Yozgat → Kırşehir → Kırıkkale → Çankırı → Ankara → Eskişehir → Kütahya → Afyonkarahisar → Uşak → Isparta → Burdur → Antalya → Denizli → Muğla → Aydın → İzmir → Manisa → Balıkesir → Çanakkale → Edirne → Kırklareli → Tekirdağ → İstanbul → Kocaeli → Yalova → Bursa → Bilecik → Sakarya → Düzce → Bolu → Zonguldak → Bartın → Karabük → Kastamonu → Sinop → Samsun → Ordu → Giresun → Trabzon → Rize → Artvin → Ardahan → Kars → Iğdır → Ağrı → Erzurum → Bayburt → Gümüşhane → Erzincan → Tunceli → **Malatya**

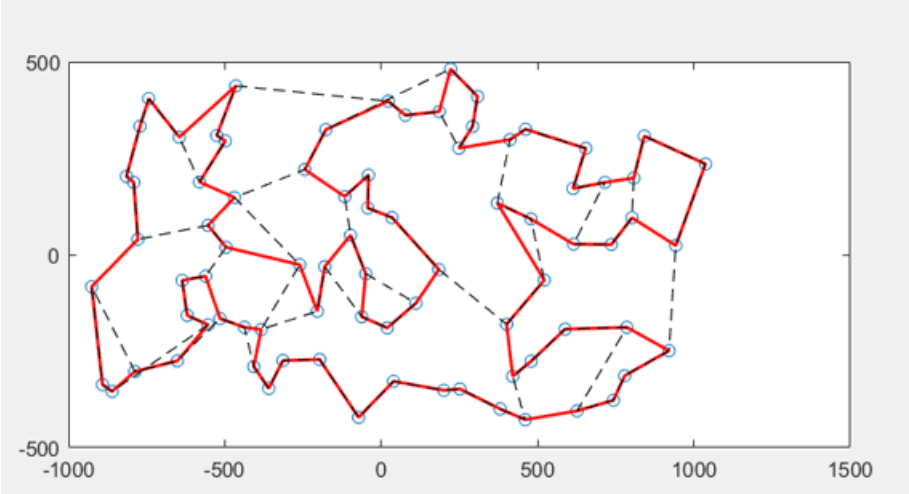

FIGURE 1. The best found tour

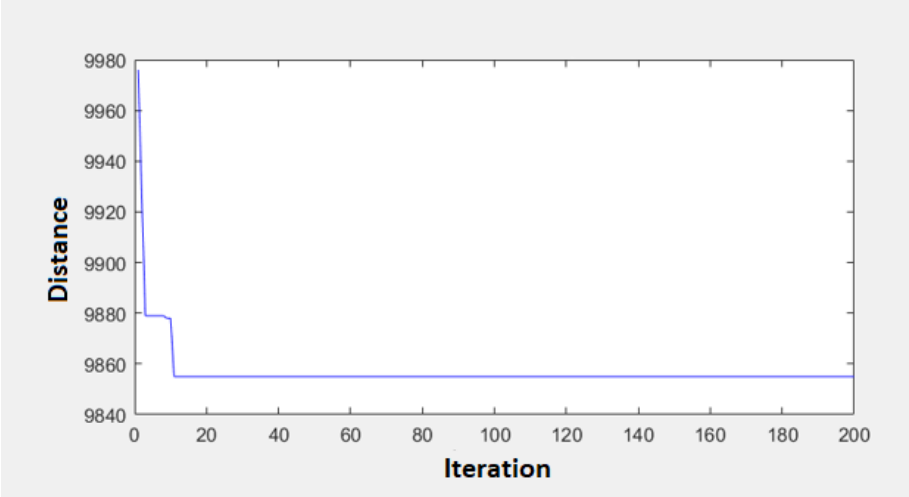

FIGURE 2. Convergence Graph

All computational experiments were performed on a system equipped with an Intel Core i7-8665U, 1.9 GHz CPU, 32 GB DDR4 2667 MHz RAM, Windows 11 Pro 64-bit 24H2

operating system, 512 GB SSD, 426.45 MB/s – 755.36 MB/s storage and a 21 nanoseconds Column Address Strobe (CAS) Latency. These hardware specifications are provided to ensure reproducibility and fair comparison of the results.

The problem was implemented using the MATLAB software package (MATLAB version R2017b).

6. Conclusions

In this study, the symmetric Traveling Salesman Problem was solved using the Ant Colony Optimization algorithm, and its application on the map of Türkiye was presented. The results demonstrate the efficiency of the ACO algorithm in solving real-world geographic optimization problems.

The best solution obtained in this study yielded a total travel distance of 9855 km, which is an improvement over several previous works that applied ACO to the TSP on the map of Türkiye. Compared to similar studies in the literature, our results indicate a more optimized route, demonstrating the effectiveness of the chosen parameters and implementation approach.

References

[1] Appligate, D. L., Bixby, R. E., Chavatal, V., Cook, W. J., (2006), The Traveling Salesman Problem, A Computational Study, Princeton University Press.

[2] Bonabeau, E., Dorigo, M., Theraulaz, G., (1999), Swarm Intelligence: From Natural to Artificial Systems, Oxford University Press.

[3] Bullnheimer, B., Hartl, R. F., Strauss, C., (1999), A new rank-based version of the ant system - computational study, Central European Journal of Operations Research, 7(1), pp. 25-38.

[4] Davendra, D., (2010), Travelling Salesman Problem, Theory and Applications, IntechOpen.

[5] Dikmen, H., Dikmen, H., Elbir, A., Ekşi, Z., Çelik, F., (2014), Gezgin satıcı probleminin karınca kolonisi ve genetik algoritmalarla eniyilemesi ve karşılaştırılması, (Optimization and comparison of Travelling Salesman Problem using ant colony and genetic algorithms), Suleyman Demirel University Journal of Natural and Applied Science, 18(1), pp. 8-13.

[6] Dorigo, M., Maniezzo, V., Colorni, A., (1991), Ant System: An Autocatalytic Optimizing Process, Technical Report 91-016.

[7] Dorigo, M., Gambardella, L. M., (1997), Ant Colony System: A cooperative learning approach to the traveling salesman problem, IEEE Transactions on Evolutionary Computation, 1(1), pp. 53-66.

[8] Dorigo, M., Maniezzo, V., Colorni, A., (1996), The ant system: Optimization by a colony of cooperating agents, IEEE Transactions on Systems, Man, and Cybernetics, Part B, 26(1), pp. 1-13.

[9] Garey, M. R., Johnson, D. S., (1979), Computers and Intractability: A Guide to the Theory of NP-Completeness, W. H. Freeman and Company.

[10] Gutin, G., Punnen, A., (2002), The Traveling Salesman Problem and Its Variations, Kluwer Academic Publishers.

[11] Johnson, D., Papadimitriou, C., (1985a), Computational Complexity, In Lawler et al., Chapter 3, pp. 37-86.

[12] Johnson, D., Papadimitriou, C., (1985b), Performance Guarantees for Heuristics, In Lawler et al., Chapter 5, pp. 145-180.

[13] Johnson, D. S., McGeoch, L. A., (1995), The Traveling Salesman Problem: A Case Study in Local Search in Combinatorial Optimization, pp. 215-310, John Wiley & Sons.

[14] Kızılateş, G., Nuriyeva, F., (2013), On the nearest neighbor algorithms for the traveling salesman problem, Advances in Computational Science, Engineering and Information Technology, Advances in Intelligent Systems and Computing, 225, Springer, Heidelberg.

[15] Lawler, E. L., Lenstra, J. K., Rinnoy Kan, A. H. G., Shmoys, D. B., (1991), The Traveling Salesman Problem: A Guided Tour of Combinatorial Optimization, Wiley Series in Discrete Mathematics & Optimization.

[16] Manfrin, M., Birattari, M., Stützle, T., Dorigo, M., (2006), Parallel Ant Colony Optimization for the Traveling Salesman Problem, In: Dorigo, M., Gambardella, L.M., Birattari, M., Martinoli, A., Poli,

R., Stützle, T. (eds) Ant Colony Optimization and Swarm Intelligence. ANTS 2006. Lecture Notes in Computer Science, 4150, Springer, Berlin, Heidelberg.
[17] Miller, C. E., Tucker, A. W., Zemlin, R. A., (1960), Integer programming formulations and Traveling Salesman Problems, Journal of the ACM, 7(4), pp. 326–329.
[18] Mohsen, A. M., (2016), Annealing ant colony optimization with mutation operator for solving TSP, Computational Intelligence and Neuroscience, 2016, pp. 1-13.
[19] Nuriyev, U., Nuriyeva, F., (2018), Practical aspects of solving combinatorial optimization problems, Advanced Mathematical Models and Applications, 3(3), pp. 179–191.
[20] Rosenkrantz, D. J., Stearns, R. E., Lewis, P. M., (1977), An analysis of several heuristics for the Travelling Salesman Problem, SIAM Journal of Computing, 6, pp. 563-581.
[21] Shang, Y., Gong, H., Zeng, Q., (2007), Enhancing ant colony optimization algorithm for solving large traveling salesman problem, Applied Mathematics and Computation, 184(2), pp. 422-431.
[22] Siemiński, A., (2016), Using hyper populated ant colonies for solving the TSP, Vietnam J Comput Sci, 3, pp. 103–117.
[23] Söyler, H., Keskintürk, T., (2007), Karınca kolonisi algoritması ile Gezen Satıcı Probleminin çözümü, (Solution of the Traveling Salesman Problem with ant colony algorithm), 8th Turkish Conference on Econometrics and Statistics, Malatya, Turkey, pp. 1-11.
[24] Skinderowicz, R., (2022), Improving ant colony optimization efficiency for solving large TSP instances, Applied Soft Computing, 120, pp. 108653.
[25] Stützle, T., Hoos, H. H., (2000), MAX–MIN ant system, Future Generation Computer Systems, 16(8), pp. 889-914.
[26] Şenaras, E. A., İnanç, Ş., (2017), GSP çözümü için karınca kolonisi optimizasyonu, (Ant colony optimization for solving Vehicle Routing Problems), Papers on Social Science, 2017(2), pp. 58-67.
[27] Zarei, H., YousefiKhoshbakht, M., Khorram, E., (2016), A hybrid modified meta-heuristic algorithm for solving the traveling salesman problem, Journal of Industrial and Systems Engineering, 9(3), pp. 57-69.
[28] Zhang, X., Bai, Q., Yun, X., (2011), A new hybrid artificial bee colony algorithm for the traveling salesman problem, 2011 IEEE 3rd International Conference on Communication Software and Networks, Xi'an, China, pp. 155-159, doi: 10.1109/ICCSN.2011.6014240.
[29] Zheng, Z., Liu, S., Zhou, X., (2023), Optimization of Logistics Distribution Route Based on Ant Colony Algorithm – Taking Nantian Logistics as an Example, In: Hu, Z., Zhang, Q., He, M. (eds) Advances in Artificial Systems for Logistics Engineering III. ICAILE 2023. Lecture Notes on Data Engineering and Communications Technologies, vol 180, Springer, Cham.
[30] `http://www.kgm.gov.tr/Sayfalar/KGM/SiteTr/Root/Uzakliklar.aspx`.

Fidan Nuriyeva received the B.Sc., M.Sc., and Ph.D. degrees in Mathematics from Ege University, Izmir, Türkiye. She works in Yaşar University, Department of Mathematics, Izmir, Türkiye. Her research has focused on combinatorial optimization and operations research.

Veysel Erdemci received the B.Sc. degree from Izmir University, Department of Mathematics and Computer Science, and M.Sc. degree in Computer Science from Dokuz Eylul University, Izmir, Türkiye. He works in Denizbank Intertech, Istanbul, Türkiye. His research has focused on Clustering, School Bus Routing Problems.