DEU FMD 26(78) (2024) 519-527

Dokuz Eylül Üniversitesi Mühendislik Fakültesi Fen ve Mühendislik Dergisi
Dokuz Eylul University Faculty of Engineering Journal of Science and Engineering
Elektronik/Online ISSN: 2547-958X

RESEARCH ARTICLE / ARAŞTIRMA MAKALESI

Ant Colony Optimization and Beam-Ant Colony Optimization on Traveling Salesman Problem with Traffic Congestion

Trafik Sıkışıklığı Olan Gezgin Satıcı Probleminde Karınca Kolonisi Optimizasyonu ve Işın-Karınca Kolonisi Optimizasyonu

Mustafa Orçun Uslu [1*], Kazım Erdoğdu [2]

[1] Yaşar University, Graduate School, Bornova, İzmir, TÜRKİYE
[2] Yaşar University, Department of Software Engineering, Bornova, İzmir, TÜRKİYE

Corresponding Author / Sorumlu Yazar *: orcunuslu6@gmail.com

Abstract

The Traveling Salesman Problem (TSP) is a well-known combinatorial optimization problem that has various implications in a variety of industries. Even the purest formulation of TSP has applications on from logistics routes to microchip manufacturing, unexpectedly, it can be used on DNA sequencing with slight modification as a sub-problem. In this paper, two versions of TSP were studied, a classical TSP and the TSP containing traffic congestion data. Two state-of-the-art solution methods were used, Ant Colony Optimization (ACO) and Beam-ACO. These algorithms were hybridized with 2-Opt local search and their performances compared on the same benchmark instances. The experimental results show the efficiency of Beam-ACO compared to ACO.

Keywords: *Traveling Salesman Problem, Ant Colony Optimization, Beam-ACO, 2-Opt, Swarm Intelligence Optimization, Traffic Congestion*

Öz

Gezgin Satıcı Problemi (GSP), çeşitli endüstrilerde çeşitli etkileri olan, iyi bilinen bir kombinatoryal optimizasyon problemidir. GSP'nin en saf formülasyonu bile lojistik yollardan mikroçip üretimine kadar çeşitli uygulamalara sahiptir. Beklenmedik bir şekilde, bir alt problem olarak DNA dizilimi için küçük değişikliklerle kullanılabilir. Bu yazıda GSP'nin iki versiyonu incelenmiştir: klasik bir TSP ve trafik sıkışıklığı verilerini içeren GSP. Son teknoloji ürünü iki çözüm yöntemi kullanıldı: Karınca Kolonisi Optimizasyonu (KKO) ve Işın-KKO. Bu algoritmalar 2-Opt yerel arama ile hibritleştirildi ve performansları aynı kıyaslama örnekleriyle karşılaştırıldı. Deney sonuçları Işın-KKO'nun KKO'ya kıyasla verimliliğini göstermektedir.

Anahtar Kelimeler: *Gezgin Satıcı Problemi, Karınca Kolonisi Optimizasyonu, Beam-KKO, 2-Opt, Sürü Zekası Optimizasyonu, Trafik Sıkışıklığı*

1. Introduction

The Traveling Salesman Problem (TSP) is a challenging optimization problem in which a salesman must find the shortest tour that visits a set of cities exactly once and returns to the starting point [1].

The challenging part of the TSP is where the problem exponentially increases the computation cost. As the number of cities increases, the number of possible routes grows at a factorial rate, making exhaustive search impractical for larger instances. Furthermore, TSP is classified as NP-hard [2], implying that there is no known polynomial-time algorithm to find the optimal solution efficiently.

TSP has implications in a variety of industries. It helps businesses optimize their delivery routes, saving them time and money on fuel [3]. It assists in the design of effective machine routes in manufacturing [4], increasing production efficiency. Tourists employ TSP to plan itineraries that cover multiple attractions while minimizing travel time [5]. In network design, TSP can be used to determine the shortest cable layout in a network [6].

In real life, traffic is known to be a problem for everyone. It causes money and time loss as well as environmental damage and negative effects on human health [7]. The proposed algorithms try to solve this problem with a different version of TSP, created a simple model and called it “TSP with Traffic Congestion”.

This paper combines the Beam Search algorithm with the Ant Colony Algorithm (ACO) and compares it with a standard Ant Colony Algorithm while adding an element to the problem as “Traffic between cities”. While the ACO is one of the swarm intelligent methods that is inspired by real ant colonies and proposed by Marco Dorigo [8, 9]. The algorithm mimics real-life ants that try to find food source and return them to nest. The Beam Search Algorithm is a method used in many Natural Language Processing (NLP) and Speech Recognition Models for decision making. The algorithm is like a tree search algorithm with a scoring system where the highest score is to one that is decided on.

The Ant Colony Optimization (ACO) algorithm was proposed by Dorigo in 1992 in his PhD thesis [8]. His first algorithm aims to mimic an ant colony where an ant searches for a path between their colony and food source and this is used to search for an

DOI: https://doi.org/10.21205/deufmd.2024267820

Geliş Tarihi / Received: 05.01.2024
Kabul Tarihi / Accepted: 07.03.2024

optimal path in a graph. In his second paper [9], he applied this method to the Travelling Salesman Problem (TSP). His search proved that ACO had very promising results and encouraged to apply this method to other optimization problems.

In 1997, Doringo improved his proposed algorithm and created a new algorithm called Ant Colony System (ACS)[10]. In this algorithm edge selection is biased towards favorability of the edge with its pheromone level. At the end of each iteration, only the best ant could update the global pheromone level.

In 2000, Stützle proposed the "Max-Min Ant System"(MMAS) [11] to overcome ACO's poor performance when the instance is large enough and demonstrated that ACO could be an effective solution for hard combinatorial optimization problems. In his paper, he proved that MMAS is currently the best-performing ACO algorithm for Quadratic Assignment Problem (QAP) and TSP.

Chu [12] proposed another approach to the ACO algorithm and added communication strategies to it in 2004. He called this algorithm Parallel Ant Colony Optimization (PACO). In the proposed algorithm, the population of ants is divided into multiple groups, and seven communication methods are applied to update the pheromone level between these groups. This method outperforms the previously proposed ACO algorithms.

To increase the efficiency and collaboration of the ants in searching the solution space, Hu and Li [13] proposed Continuous Orthogonal Ant Colony (COAC) algorithm by using an orthogonal design method where ants can explore the chosen domain more efficiently and accurately.

In order to find more accurate results in TSP, Gupta et al. [14] introduced a new algorithm called Recursive Ant Colony Optimization (RACO). Their algorithm divides the search space into multiple sub-spaces. After solving the TSP in these sub-spaces, RACO selects the best solutions for each sub-space and carries them to the next step. The algorithm runs until finds a solution, and every process mentioned above repeats itself.

The term Beam Search was first suggested by Reddy [15] in 1977. In 1987, Bisiani [16] implemented the Beam Search Algorithm as a combination of the best-first and breadth-first searches. It only searches for the best and prunes the tree which helps to reduce memory usage.

Blum [17] was the first person to use the Beam Search algorithm and ACO together to solve a Scheduling problem in 2005. He called this method “Beam-ACO” and proved that Beam-ACO is getting better results than standard ACO algorithms.

Proven that the Beam-ACO algorithm is better than the ACO algorithm, Caldeira et al. [18] applied this method to a supply chain management system. They have shown that Beam-ACO is always faster and better than ACO in supply chain problems. Although it’s slower than ACO, Beam-ACO still had better results on logistic systems.

In 2008, Blum [19] applied Beam-ACO to the Simple Assembly Line Balancing (SALB) problem and showed that Beam-ACO gives the best solutions for the benchmark instances of the problem. He mentioned that Beam-ACO optimally solved the majority of the existing benchmark instances (SALB-1 problem). The solutions obtained for the rest of the instances by Beam-ACO were nearly optimal solutions.

In 2009, López-Ibánez et al. [20] applied Beam-ACO on TSP with Time Windows (TSPTW) by replacing the bounding information in Beam-ACO by stochastic sampling. Based on their experimental results, they claim that their proposed algorithm was the state-of-the-art method for TSP.

In 2010, López-Ibánez and Blum [21] improved their study [20] by implementing one-opt local search to it. Their results outperformed the results of the existing heuristic methods and were able to find good approximations in a much shorter time.

In 2017, Simões et al. [22] applied a hybridized Beam Search algorithm with a Population-based Ant Colony Optimization algorithm and called it Beam Search with Population-based Ant Colony Optimization (P-ACO). They applied P-ACO to a Multi-rendezvous Spacecraft Trajectory Optimization problem. The P-ACO algorithm had superior worst-case performance which might be the preferable choice for the practical applications.

In general, genetic algorithms (GA) are utilized in literature. In this paper, Beam-ACO and ACO are implemented and compared to each other. In addition, studied a different variation of TSP by adding the traffic congestion information to the TSP.

In Section 2, Materials and Methods are given by the formulation of the problems, explanations of the algorithms and their pseudo codes and their respected parameters. In Section 3, Experimental Results were laid out by explaining how the test cases are done, and giving the results in the figures and tables. Section 4 contains the Discussion, where the results and the performances of the two algorithms are compared. Section 5 includes the Conclusion and Future Work, in which the concluding remarks and future work for improvement are mentioned.

A combinatorial optimization problem known as the "Traveling Salesman Problem" (TSP) [1] [17] is defined as follows. There is a salesman who must travel to a given set of cities by visiting each city exactly once and returning to the initial city from which he or she began his or her tour. The goal of the salesman is to minimize the total distance (or cost) of this tour. Mathematically, TSP is a graph problem that searches for a Hamiltonian cycle on the given nodes (i.e. cities) that has the minimum total weight (i.e. total distance). The mathematical model is given below.

TSP is defined as a graph $G = (V, A)$ where V and A are the sets of cities and arcs between the cities, respectively. Let $d_{i,j}$ define the distance between the cities i and j. Then the TSP can be mathematically formulated as follows:

$$minimize \quad \sum_{i \in V} \sum_{j \in V} d_{i,j} x_{i,j} \tag{1}$$

Subject to

$$\sum_{i \in V \; i \neq j} x_{i,j} = 1, \qquad \forall j \in V \tag{2}$$

$$\sum_{j \in V \; i \neq j} x_{i,j} = 1, \qquad \forall i \in V \tag{3}$$

$$\sum_{i,j \in S \; i \neq j} x_{i,j} \leq |S| - 1, \qquad 2 \leq |S| \leq |V| \;\; and \;\; \forall S \subset V \tag{4}$$

$$x_{i,j} = \begin{cases} 1, & if\ the\ path\ from\ i\ to\ j\ is\ in\ the\ tour \\ 0, & otherwise \end{cases} \tag{5}$$

Equation (1) is the objective function of the TSP which aims to minimize the tour distance. Equations (2) – (5) are the constraints of the TSP. Equations (2) and (3) ensure that each city is visited by the salesman exactly once. Equation (4) eliminates the sub-tours in the TSP tour and Equation (5) is the decision variable for selecting the arcs in the tour.

In this paper, two types of TSP are studied. The first one is the classical TSP with the mathematical definition given above. We proposed a second type of TSP in addition to the classical TSP. We included traffic congestion information on each arc and updated the objective function with Equation (6). In the function, $c_{i,j}$ represents the traffic congestion unit which was randomly generated for each arc in the problem from the set {1,2,3,4,5}. Each value in the set represents the traffic density as "no traffic, less traffic, normal, semi-heavy traffic, and heavy traffic", respectively.

$$minimize \quad \sum_{i \in V} \sum_{j \in V} c_{i,j} d_{i,j} x_{i,j} \tag{6}$$

2. Material and Methods

Since TSP is an NP-Hard problem, we used two well-known meta-heuristic methods to solve the TSP in this study. We both implemented the classical Ant Colony Optimization (ACO) algorithm and the Beam Search ACO, as known as the Beam-ACO algorithm.

ACO is a swarm intelligence method that was proposed by Marco Dorigo in 1992 in his PhD thesis [8]. ACO mimics the collective intelligent foraging behavior of ants. In the natural world, ants begin their exploration of food randomly. They leave pheromone hormones on the trails they traverse. Once they locate a food resource then they keep adding more pheromones on the path between the nest and the food. The pheromone hormones assist other ants to follow the same trail for the same food source. The shorter the distance between the food source and the nest the greater the amount of pheromone. This results in shortening the travel distances of the ants and directs the routes of the ants more intelligently. Nevertheless, the pheromones are volatile meaning it gradually evaporates in time. It means that more ant travels are needed on the same path to attract the other ants to follow that path. Consequently, the food sources that are farther away from the nest are less preferred by the ants. ACO simulates all this behavior in its search for the optimum tour for the TSP.

Beam search is a popular search algorithm in AI, particularly in natural language processing. It keeps track of a collection of potential sequences, building on an initial sequence by taking into account a variety of potential following moves. Each choice is given a likelihood score, and the highest-scoring sequences are kept while the others are pruned.

The beam-ACO algorithm takes these 2 algorithms and combines them. While the ant colony is working as usual, each and every ant runs the Beam Search algorithm to decide which city to go to next. When an ant is selecting the next city to move between unvisited cities, calculates their probability and distances. Then according to beam width, the amount of the best cities is selected among the unvisited cities. The ant will select the next city randomly within the selected cities, but the more probability the city has the more likely the ant will choose that city and vice versa.

2.1. Pseudocode of the Beam-Aco Algorithm

The GA procedure in this study is described below:

```
Initialize the Ant Population
Repeat
   Construct the Tour of Each Ant
   Evaluate Fitness Values of Each Ant Tour
   Apply 2-Opt on Each Tour
   Update Best Solution
   Update Pheromone
Until (termination condition is satisfied)
Return Best Solution
```

2.2. Initialize the Ant Population

Several ants are randomly located at the cities in the initializing of the ant population. The initial pheromone amounts on each arc is set to 1. Every ant has a random path assigned by giving cities and shuffling them. By shuffling the path for each ant, the initial generation will have a uniformly distributed solution pool.

2.3. Constructing the Tour of Each Ant

Each ant starts moving from their first city to the next unvisited city and continues from the current city to next unvisited city until it finishes constructing its tour. Each unvisited city has a probability of selection. This probability is highly influenced by the pheromone amount on the arc to that city and the length of (and traffic congestion on) that arc. In this study, the probability of city selection is calculated as in [23]. The formula is given in Equation (7).

$$p_{i,j}^{k}(t) = \begin{cases} \dfrac{[\tau_{i,j}(t)^{\alpha}]\,[\eta_{i,j}(t)^{\beta}]}{\sum_{s \in allows_k}[\tau_{i,j}(t)^{\alpha}]\,[\eta_{i,j}(t)^{\beta}]}, & j \in allows \\ 0, & j \notin allows \end{cases} \tag{7}$$

where,

- $p_{i,j}^{k}(t)$ is the probability of selecting a city by ant k at time t
- $\tau_{i,j}(t)$ is the amount of pheromone on the arc between city i and j at time t
- $\eta_{i,j}(t)$ is the desirability of the arc between city i and j at time t (typically $1/d_{i,j}$)
- α is a parameter that controls the influence of $\tau_{i,j}$

β is a parameter that controls the influence of $\eta_{i,j}$

In the classical ACO algorithm, at each city, an ant decides the next unvisited city by considering the probabilities of all unvisited cities. Although this approach explores all the possible candidate cities, it is a time and memory-consuming process. Beam-ACO speeds up and saves up from memory by limiting the choice of the ants in a beam width. Beam width is a predefined number that limits the unvisited city pool. Beam includes the number of cities according to Beam Width. The probabilities (Eq. 7) of each city are calculated beforehand, and the cities that have the best probabilities are selected for the beam. Within the selected unvisited cities, the city is selected randomly but the more probability the city has, the more likely the ant chooses it as the next city. The beam search procedure in this study is described below:

```
Beam Search
    Calculate Probabilities of Each City
    Get Cities with Best Probabilities [Beam Width]
    Choose Next City Randomly According to
    Probability Ratio
```

2.4. Evaluate Fitness Values of Each Ant Tour

Once the ant tours are constructed, the fitness values of each tour is also calculated. To calculate fitness values of each solution, Equation (1) is used for the classical TSP scenario and Equation (6) is used for TSP with traffic congestion scenario.

2.5. Apply 2-Opt on Each Tour

The current solutions obtained by the ants are improved by 2-Opt. This algorithm was proposed by Croes [24]. 2-Opt local search is a well-known and often used local search heuristic, especially in permutation-based optimization problems such as TSP. In 2-Opt method, two connection points (or edges in TSP) in the permutation are successively broken and cross-connected with the aim of reducing the fitness of the solution (i.e. total distance or cost of the tour). If the new permutation's fitness is better than the current one then the original permutation is replaced by the new permutation; otherwise, it remains unchanged. 2-Opt repeats this process consecutively on each edge in the tour and results in finding the best permutation possible on the ant tour.

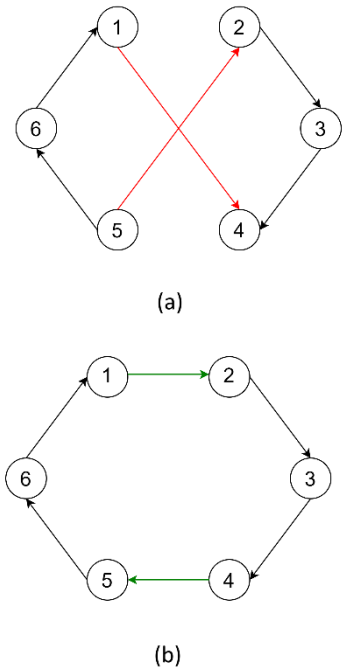


Figure 1. 2-Opt Local Search

In Fig. 1, the breaking and cross-connection of edges of a tour in 2-Opt local search algorithm is represented. Cities are shown in circles with their indices. Directed arrows demonstrate the order of the city sequence in the tour, and the lengths of these arrows represent the distance between 2 cities. In the first diagram (a) of Fig. 1, the path that an ant follows is changed by breaking the connections between city 1 and 4, and the cities between 2 and 5. Then, city 1 is connected to city 2, and city 4 is connected to city 5. This way the total distance of the ant's tour is decreased, which in turn, improves the fitness of the solution obtained by that ant. This breaking and cross-connecting process is repeatedly applied to all edges of the tour, until there is no improvement occur on the total distance. In the end, 2-Opt returns the tour with the best fitness that is found for the given tour.

2.6. Update Best Solution

After applying 2-Opt on each ant tour, the current best solution is updated according to these newly obtained tours. The solution with the best fitness value is assigned as the new best solution.

2.7. Update Pheromone

The final step in a generation of Beam-ACO is updating the amount of pheromones on each arc. In other words, some amount of pheromone evaporates before the next generation of ant moves. This pheromone update is done with Equation 8.

$$\tau_{i,j} = (1-\rho)\tau_{i,j} + \triangle\,\tau_{i,j} \tag{8}$$

where,

- $\tau_{i,j}$ is the amount of pheromone on the arc between the cities i and j
- ρ is the rate of pheromone evaporation
- $\triangle\,\tau_{i,j}$ is the amount of pheromone deposited, typically given by

$$\triangle^{k}_{\tau_{i,j}} = \begin{cases} \frac{1}{L_k}, & \textit{if ant } k \textit{ travels on the arc between } i \textit{ and } j \\ 0, & \textit{otherwise} \end{cases} \tag{9}$$

where L_k is the distance of the k^{th} ant's tour.

2.8. Termination Criteria

In our study, our Beam-ACO algorithm terminates either when it finds the known optimal solution in literature for the related problem instance or when it runs predefined number of generations. In the latter case the overall best solution of all generations is the result of Beam-ACO.

2.9. Beam-Aco Parameters

The values of the parameters in pheromone calculations and updates in the Beam-ACO used in this paper are adapted from the study [15]. The ant population size, beam width and iteration size values are determined by an experimental study done on one of the TSP instances used in this study which is "ch150.tsp". The values of these parameters that yielded to best results are given in Table 1 below. Same parameters and values are also used for ACO.

Table 1. Beam-ACO Parameters

Parameter	Value
m: number of ants in the Beam-ACO population	200
α: pheromone factor	1
β: heuristic function factor	5
Q: pheromone constant	100
ρ: evaporation rate	0.5
bw: beam width.	30% of n (n is the number of the cities)

N: maximum number of iterations of Beam-ACO	100

3. Experimental Results

The ACO and Beam-ACO algorithms used in this study are coded in Java programming language on a laptop computer with i7 processor and 16 GB RAM. The graphics of the best results are coded in Python programming language.

Four well-known TSP instances were selected in the experimental studies: berlin52.tsp, ch130.tsp, ch150.tsp, a280.tsp. These instances can be found at http://comopt.ifi.uni-heidelberg.de/software/TSPLIB95/tsp/. The numbers on the files are represent the number of the cities that instance contains.

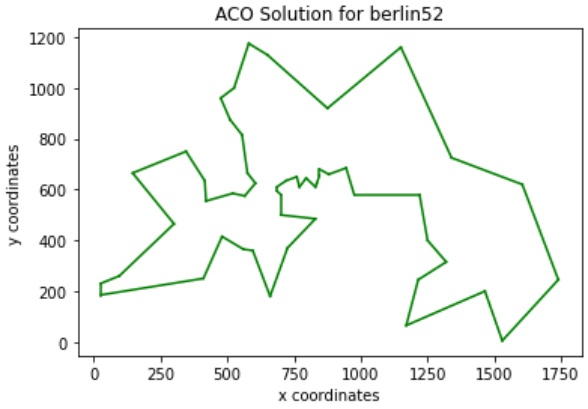
Two scenarios of the TSP were taken into consideration: classical TSP and TSP with traffic congestion. For each scenario, each instance was run 30 times for each ACO and Beam-ACO algorithms.

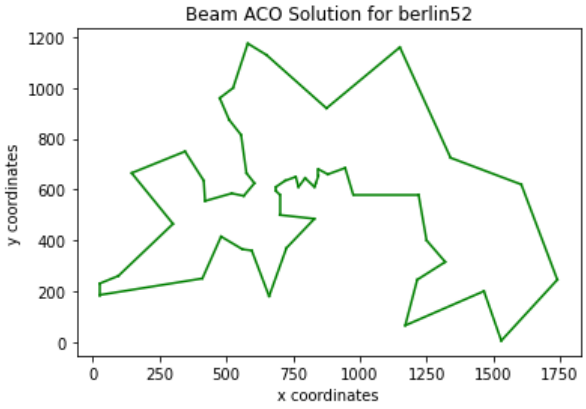
In the first scenario, the performances of ACO and Beam-ACO algorithms were compared against each other and the optimum distances known in TSP literature for the related instance. Table 2 contains the experimental results of the first scenario. In Table 2, n is the number of cities that the instances have and provided. Literature Optimum values are obtained from TSBLIB95 at http://comopt.ifi.uni-heidelberg.de/software/TSPLIB95/tsp/. ACO and Beam-ACO best values are calculated according to equation (1) in Table 2 and equation (6) in Table 3.

Table 2. The performances of ACO and Beam-ACO for classical TSP

Instance	n	Literature Optimum	ACO Best	ACO's Convergence to the Optimum	Beam-ACO Best	Beam-ACO's Convergence to the Optimum	ACO Average	Beam-ACO Average
berlin52	52	7542	**7542**	0.000%	**7542**	0.000%	**7542**	**7542**
ch130	130	6110	**6144**	0.556%	**6144**	0.556%	6167.8	**6165.43**
ch150	150	6528	6544	0.245%	**6540**	0.184%	**6577.56**	6579
a280	280	2580	2649	2.674%	**2647**	2.597%	2674.5	**2672.8**

ACO and Beam-ACO could find optimum values of berlin52.tsp instance. They, however, could not perform the same success on the other three instances. It is because, as the number of cities increases, the search space expands exponentially. On the other hand, while both ACO and Beam-ACO found the same best value for the ch130.tsp instance, Beam-ACO outperformed ACO with regards to its convergence to the optimum values for the other two instances, ch150.tsp and a280.tsp. Figure 2 shows the graphical results of both ACO's and Beam-ACO's best solutions in Euclidean space for the first scenario.





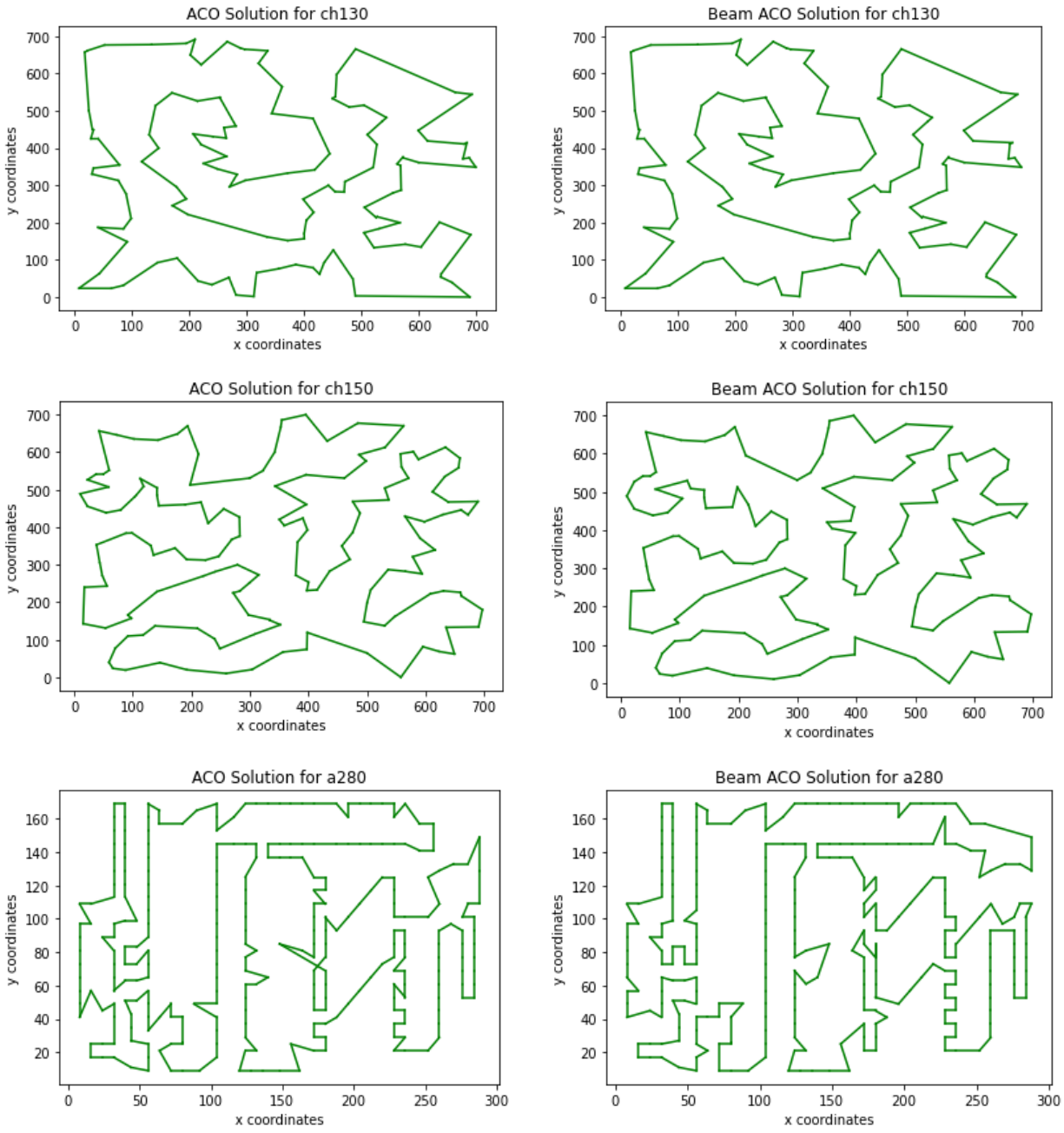


Figure 2. The graphics of ACO and Beam-ACO result in Euclidean space for the first scenario

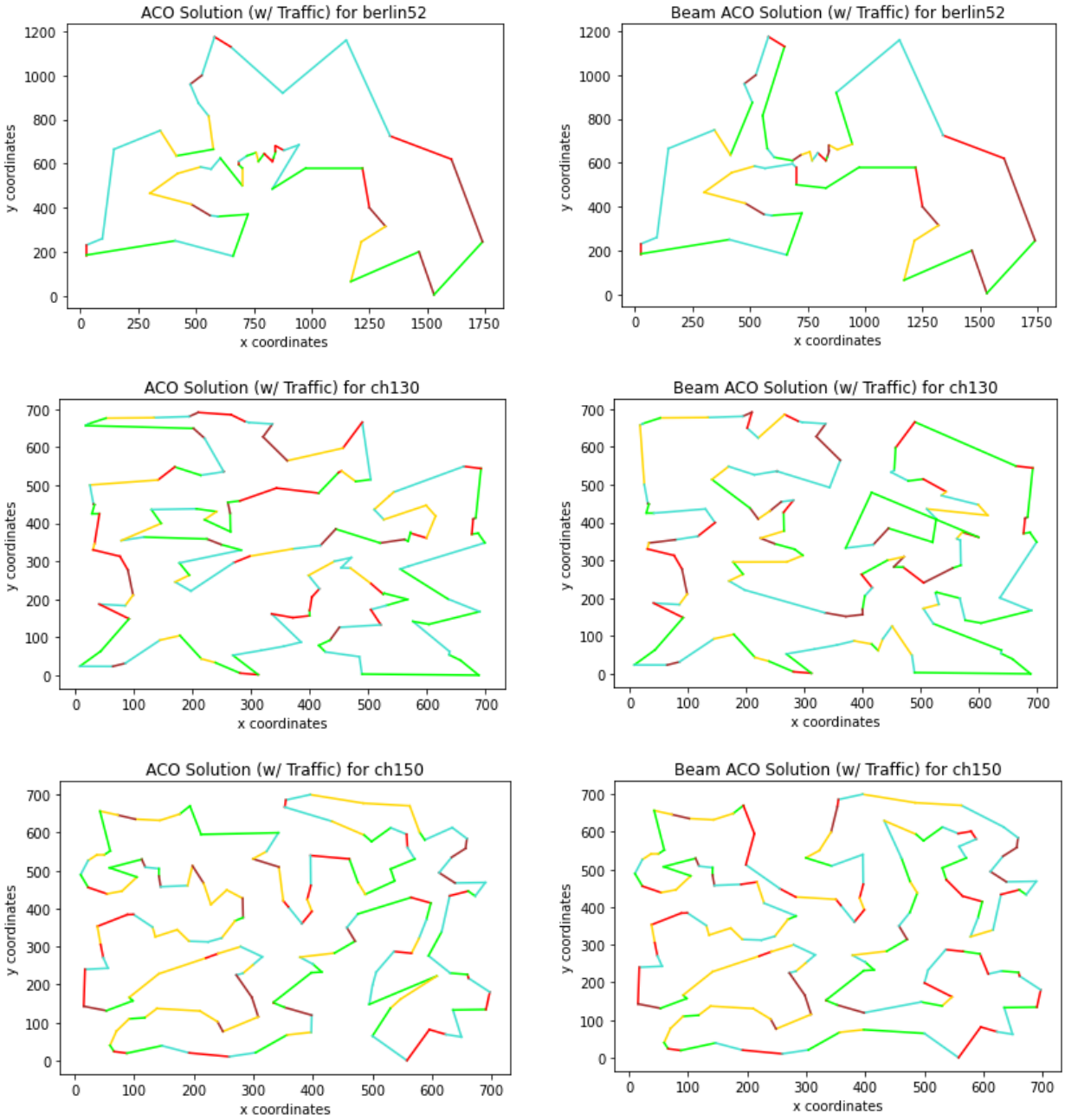
The second scenario includes the traffic congestion information on the arcs between each city. This information consists of numbers from 1 to 5 representing the intense of traffic on that arc. Each color in Figure corresponds to the density of traffic congestion such as "no traffic, less traffic, normal, semi-heavy traffic, and heavy traffic" with colors "1-lime, 2-turquoise, 3-gold, 4-red, 5-brown" respectively. These values are randomly assigned for each instance. To the best of our knowledge, this is the first study that proposed a TSP model which includes traffic congestion information on the arcs in the selected TSP instances and that the ACO and Beam-ACO algorithms are applied on such a setup. Since the optimum values for this TSP model do not exist in literature, the performances of ACO and Beam-ACO algorithms in this second scenario were compared against each other. Table 3 contains the experimental results of the second scenario.

Table 3. The performances of ACO and Beam-ACO for the TSP model that includes traffic congestion

Instance	n	ACO Best	Beam-ACO Best	ACO Average	Beam-ACO Average
berlin52	52	18942	**18786**	19156.2	**19117.1**
ch130	130	15191	**14953**	15524.6	**15516.2**

ch150	150	17583	**17545**	17955.1	**17883**
a280	280	**7472**	7562	**7677.1**	7704.46

As the result shows in Table 3, Beam-ACO algorithm has a better performance on the majority of the instances both in best and average optimum values. ACO has better results on a280.tsp because as it is seen in the figures, some of the location has multiple option with exact same distance which is challenging for the algorithms. Adding traffic levels to the process, solving the TSP become harder since the objective function of this new TSP model contains travel times along with distances on each path. Both ACO and Beam-ACO algorithms try to find the fastest path to complete TSP while avoiding high-level traffic on the path. As it is seen in Figure 3 below, ACO and Beam-ACO aim to minimize the selection of high distance and traffic paths. The chosen high traffic paths are mostly short distances that can be travelled faster. Since Beam-ACO is utilizing the Beam Search Algorithm for selecting the next city to move, it has better results than ACO in general. Figure 3 shows the graphical results of both ACO's and Beam-ACO's best solutions in Euclidean space for the second scenario.



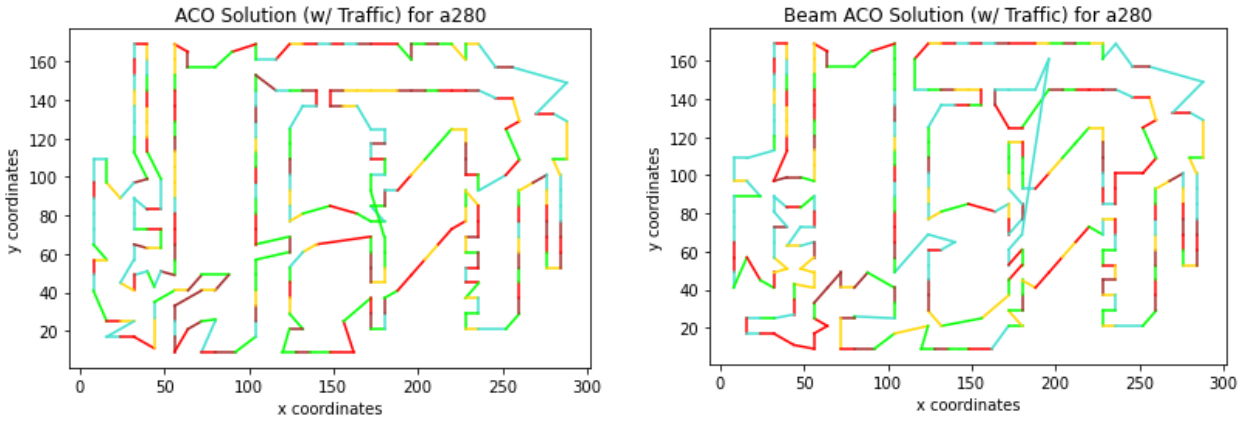


Figure 3. The graphics of ACO and Beam-ACO results in Euclidean space for the second scenario

4. Discussions

For the classical TSP version studied in this paper, the literature optimum values for the berlin52.tsp instance was found by ACO and Beam-ACO as seen in Table 2. However, these two algorithms could not perform the same success for other instances. It's because the search space grows exponentially with the number of cities. While both algorithms found the same best value for the ch130.tsp instance, in terms of convergence to the optimal values for the other two instances of ch150.tsp and a280.tsp, Beam-ACO performed better than ACO.

For the TSP with traffic congestion version of the experimental results as shown in Table 3, the Beam-ACO algorithm performs better than ACO in terms of both average and best values for the majority of instances. When traffic levels are added to the equation, the TSP becomes more difficult to solve since the new TSP model's objective function includes travel times and path lengths. To find the fastest route, both algorithms avoid high-level traffic paths. It's visible in Figure 3, both algorithms are avoiding such path and the chosen high-level traffic paths are mostly short distances. Beam-ACO has better results in general because in the city selection process, Beam-Search performed better search than classical ACO.

5. Conclusions and Future Work

In this study, a TSP was studied with two versions. The first version is the classical TSP while the second one includes traffic congestion data on the arcs of the classical one. As solution methods, we used state-of-the-art ACO and Beam-ACO enhanced by 2-Opt local search heuristic. The experimental studies of this paper indicate that Beam-ACO performed more successful results than ACO in both TSP versions on the same problem instances.

The main contribution of this paper is to proposing a TSP version with traffic congestion data and comparing the performance of two state-of-the-art swarm intelligent optimization methods, i.e. ACO and Beam-ACO. This version of TSP brings a new dimension to the conventional TSP by changing the objective of the problem from distance minimization to travel time minimization of the salesman.

As a future work, the proposed TSP model can be modified by replacing the randomly generated traffic congestion values with real-life case scenarios. This real-life data can even be arranged dynamically, that is, the traffic congestion data may vary in time as the salesman travels his or her tour.

Ethics committee approval and conflict of interest statement

The prepared article does not require ethical committee approval. There is no conflict of interest with any person or institution in the prepared article.

Author Contribution Statement

Mustafa Orçun Uslu made contributions in coding the main algorithms, running the experimental studies, and writing of the article. Kazım Erdoğdu made contributions in developing the model, coding the heuristics, improvement of the parameters, writing and proof reading the article.

References

[1] 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, Jun. 1992, doi: 10.1016/0377-2217(92)90138-Y.

[2] S. H. Rubin, T. Bouabana-Tebibel, Y. Hoadjli, and Z. Ghalem, "Reusing the NP-Hard Traveling-Salesman Problem to Demonstrate That P~NP (Invited Paper)," in 2016 IEEE 17th International Conference on Information Reuse and Integration (IRI), Pittsburgh, PA, USA: IEEE, Jul. 2016, pp. 574–581. doi: 10.1109/IRI.2016.84.

[3] Y. S. Chang and H. J. Lee, "Optimal delivery routing with wider drone-delivery areas along a shorter truck-route," Expert Systems with Applications, vol. 104, pp. 307–317, Aug. 2018, doi: 10.1016/j.eswa.2018.03.032.

[4] C. H. Cheng, Y. P. Gupta, W. H. Lee, and K. F. Wong, "A TSP-based heuristic for forming machine groups and part families," International Journal of Production Research, vol. 36, no. 5, pp. 1325–1337, May 1998, doi: 10.1080/002075498193345.

[5] V. Shinkarenko, S. Nezdoyminov, S. Galasyuk, and L. Shynkarenko, "Optimization of the tourist route by solving the problem of a salesman," Journ. Geol., Geogr., and Geoec.., vol. 29, no. 3, pp. 572–579, Oct. 2020, doi: 10.15421/112052.

[6] E. Duman, M. H. Ozcelik, and A. N. Ceranoglu, "A TSP (1,2) application arising in cable assembly shops," Journal of the Operational Research Society, vol. 56, no. 6, pp. 642–648, Jun. 2005, doi: 10.1057/palgrave.jors.2601850.

[7] A. Meijer, M. A. J. Huijbregts, E. Hertwich, and L. Reijnders, "Including human health damages due to road traffic in life cycle assessment of dwellings," Int. J. Life Cycle Assess., vol. 11, pp. 64–71, Apr. 2006, doi: 10.1065/lca2006.04.013.

[8] A. Colorni, M. Dorigo, and V. Maniezzo, "An Investigation of Some Properties of an Ant Algorithm," in PARALLEL PROBLEM SOLVING FROM NATURE, 2, R. Manner and B. Manderick, Eds., Amsterdam: Elsevier Science Publ B V, 1992, pp. 509–520. Accessed: Nov. 20, 2023. [Online]. Available: https://www.webofscience.com/wos/woscc/full-record/WOS:A1992BX92H00051(overlay:export/exp)

[9] A. Colorni, M. Dorigo, and V. Maniezzo, "Distributed Optimization by Ant Colonies," in TOWARD A PRACTICE OF AUTONOMOUS SYSTEMS: PROCEEDINGS OF THE FIRST EUROPEAN CONFERENCE ON ARTIFICIAL LIFE, F. Varela and P. Bourgine, Eds., Cambridge: M I T Press, 1992, pp. 134–142. Accessed: Nov. 20, 2023. [Online]. Available: https://www.webofscience.com/wos/woscc/full-record/WOS:A1992BW87V00017

[10] M. Dorigo and L. M. Gambardella, "Ant colony system: a cooperative learning approach to the traveling salesman problem," IEEE Transactions on Evolutionary Computation, vol. 1, no. 1, pp. 53–66, Apr. 1997, doi: 10.1109/4235.585892.
[11] "MAX–MIN Ant System - ScienceDirect." Accessed: Nov. 20, 2023. [Online]. Available: https://www.sciencedirect.com/science/article/abs/pii/S0167739X00000431?via%3Dihub
[12] S.-C. Chu, J. F. Roddick, and J.-S. Pan, "Ant colony system with communication strategies," Information Sciences, vol. 167, no. 1, pp. 63–76, Dec. 2004, doi: 10.1016/j.ins.2003.10.013.
[13] X.-M. Hu, J. Zhang, and Y. Li, "Orthogonal Methods Based Ant Colony Search for Solving Continuous Optimization Problems," J. Comput. Sci. Technol., vol. 23, no. 1, pp. 2–18, Jan. 2008, doi: 10.1007/s11390-008-9111-5.
[14] D. K. Gupta, Y. Arora, U. K. Singh, and J. P. Gupta, "Recursive Ant Colony Optimization for estimation of parameters of a function," in 2012 1st International Conference on Recent Advances in Information Technology (RAIT), Mar. 2012, pp. 448–454. doi: 10.1109/RAIT.2012.6194620.
[15] "Speech Understanding Systems. Summary of Results of the Five-Year Research Effort at Carnegie-Mellon University." Accessed: Dec. 21, 2023. [Online]. Available: https://apps.dtic.mil/sti/citations/ADA049288
[16] R. Bisiani, "Beam search," Encyclopedia of Artificial Intelligence. Wiley & Sons, pp. 56–58, 1987.
[17] C. Blum, "Beam-ACO—hybridizing ant colony optimization with beam search: an application to open shop scheduling," Computers & Operations Research, vol. 32, no. 6, pp. 1565–1591, Jun. 2005, doi: 10.1016/j.cor.2003.11.018.
[18] J. L. Caldeira, R. C. Azevedo, C. A. Silva, and J. M. C. Sousa, "Supply-Chain Management Using ACO and Beam-ACO Algorithms," in 2007 IEEE International Fuzzy Systems Conference, Jul. 2007, pp. 1–6. doi: 10.1109/FUZZY.2007.4295615.
[19] C. Blum, "Beam-ACO for Simple Assembly Line Balancing," INFORMS J. Comput., vol. 20, no. 4, pp. 618–627, FAL 2008, doi: 10.1287/ijoc.1080.0271.
[20] M. Lopez-Ibanez, C. Blum, D. Thiruvady, A. T. Ernst, and B. Meyer, "Beam-ACO Based on Stochastic Sampling for Makespan Optimization Concerning the TSP with Time Windows," in EVOLUTIONARY COMPUTATION IN COMBINATORIAL OPTIMIZATION, PROCEEDINGS, C. Cotta and P. Crowling, Eds., Berlin: Springer-Verlag Berlin, 2009, pp. 97-+. Accessed: Nov. 20, 2023. [Online]. Available: https://www.webofscience.com/wos/woscc/full-record/WOS:000265680900009
[21] M. López-Ibáñez and C. Blum, "Beam-ACO for the travelling salesman problem with time windows," Computers & Operations Research, vol. 37, no. 9, pp. 1570–1583, Sep. 2010, doi: 10.1016/j.cor.2009.11.015.
[22] L. F. Simões, D. Izzo, E. Haasdijk, and A. E. Eiben, "Multi-rendezvous Spacecraft Trajectory Optimization with Beam P-ACO," in Evolutionary Computation in Combinatorial Optimization, B. Hu and M. López-Ibáñez, Eds., in Lecture Notes in Computer Science. Cham: Springer International Publishing, 2017, pp. 141–156. doi: 10.1007/978-3-319-55453-2_10.
[23] [T. Fei et al., "Research on improved ant colony optimization for traveling salesman problem," MBE, vol. 19, no. 8, pp. 8152–8186, 2022, doi: 10.3934/mbe.2022381.
[24] G. A. Croes, "A Method for Solving Traveling-Salesman Problems," Operations Research, vol. 6, no. 6, pp. 791–812, Dec. 1958, doi: 10.1287/opre.6.6.791.