Improvement of the Bees Algorithm for Solving the Traveling Salesman Problems

Araştırma Makalesi/Research Article

Murat ŞAHİN

Control Systems Department, Roketsan A. S., 06780 - Elmadag, Ankara, Turkey
msahin@roketsan.com

(Geliş/Received:06.09.2021; Kabul/Accepted:02.01.2022)
DOI: 10.17671/gazibtd.991866

Abstract**—** The traveling salesman problem (TSP) has been a popular problem studied in the optimization field for a long time. The most successful methods used in solving these difficult problems are metaheuristic algorithms. In this study, an improved version of the bees algorithm is used in the solution of TSPs. In addition to the classical bees algorithm, two different city selection and relocation functions have been developed. With these functions, it is possible to change the location of multiple and variable numbers of cities. These new functions have been added to the continuation of the classical bees algorithm and are used only on the elite site, ensuring that the elite site becomes more elite. Thus, better results could be obtained with less number of iterations and the number of the total evaluation compared to the existing bees algorithm.

Keywords**—** The bees algorithm, the traveling salesman problem, multiple insertion, optimization.

Gezgin Satıcı Problemlerini Çözmek İçin Arı Algoritmasının İyileştirilmesi

Özet**—** Gezgin satıcı problemi (GSP), optimizasyon alanında uzun süredir çalışılan popüler bir problemdir. Bu problemlerin çözümünde kullanılan en başarılı yöntemler ise metasezgisel algoritmalardır. Bu çalışmada, GSP'lerin çözümünde Arı Algoritmasının geliştirilmiş bir versiyonu kullanılmıştır. Klasik Arı Algoritmasına ek olarak iki farklı şehir seçimi ve yer değiştirme fonksiyonu geliştirilmiştir. Bu işlevler ile birden fazla ve değişken sayıdaki şehrin yerini değiştirmek mümkündür. Bu yeni fonksiyonlar klasik Arı Algoritmasının devamına eklenerek ve sadece elit bölgede kullanılmış ve bu bölümün daha elit hale gelmesini sağlamıştır. Böylece mevcut Arı Algoritmasına göre, daha az iterasyon ve arama ile daha iyi sonuçlar elde edilmiştir.

Anahtar Kelimeler**—** Arı Algoritması, gezgin satıcı problemi, çoklu ekleme, optimizasyon.

1. INTRODUCTION

The traveling salesman problem (TSP); is based on the idea that a salesman who has to visit a large number of cities should find a route to follow in order to make these trips at a minimum distance. Usage areas have increased over time, and it has begun to be widely used in computer networks, logistics distribution, and other fields [1]. TSP is also used for path planning in unmanned aerial vehicles today [2]. Another TSP is the multiple traveling salesman problems (MTSP), in which the minimum total cost is tried to be determined for more than one salesman. Each city should only be visited by one salesman. MTSP is also used in different areas such as production planning, hot rolling planning, designing routing of many buses, satellite systems, and workforce planning [3]. Billions of attempts may be required to resolve large TSPs. A long time and high processing power is needed for such trials. Heuristic algorithms have been preferred by researchers to find optimum/close to optimum solutions in a short time.

Heuristic algorithms have been successfully applied in many different fields of science [4]. With many published papers, it has been observed that heuristic algorithms are effective for TSP. Heuristic algorithms do not guarantee to find the best solution due to their nature. However, with the developed algorithms, results very close to the best solution can be obtained. Researchers have been working on designing algorithms for TSP for many years. Better results are obtained every year due to improvements in algorithms. For this reason, papers published especially in recent years were examined within the scope of this study.

Daoqing and Mingyan developed a discrete lion swarm optimization (DLSO) and a parallel discrete lion swarm optimization (PDLSO) for TSP. They used ring topology to transfer information between sub-populations in PDLSO. In their experiments for TSPLIB (TSB Library) problems with 50 to 1000 cities, they achieved results with error values ranging from 0.03% to 6.47% [1]. Mavrovouniotis et al. used the ant colony optimization (ACO) method for a dynamic TSP. The difference in DTSP is that cities or distances between cities can change over time. An example of this is the change in some of the addresses during cargo distribution [5]. ACO is one of the most popular methods used for TSP. It is inspired by the methods of real ants to find the shortest way to reach food. Therefore, successful results are also obtained within the scope of TSP. Some other studies using ACO, multi-robot systems [6] implementation, a hybrid algorithm that using ACO for TSPLIB [7], ant colony optimization algorithm with Levy Flight [8]. Another successful study is the Combinatorial Artificial Bee Colony Algorithm (CABC), developed by Karaboga and Gorkemli. ABC has been used for many years for optimization purposes in many different areas. It is an algorithm inspired by bees' search for food. In this study, the researchers developed quick CABC algorithms for TSPLIB together with CABC. They have shown that they can achieve successful results for 15 different TSPLIB when compared with other studies with the genetic algorithm and ACO [9]. A different version of ABC has also been successfully used by Dong et al in large scale colored TSP [10]. Some other algorithms used for TSP and MTSP; simulated annealing algorithm for large-scale TSP [11] and state transition simulated annealing algorithm for TSP [12], bird mating optimizer for TSP [13], genetic algorithm [14].

Another popular heuristic method is the bees algorithm (BA). The BA was developed by Pham et al. in 2005 which is a population-based search algorithm. The algorithm mimics bees' nectar search behavior [15]. This algorithm has some advantages over other optimization methods, like simplicity, flexibility, and robustness [16]. Because of these advantages, the bees algorithm has been used in the scope of optimization in different areas. Some of these, the interline power flow controller system [17], the dynamic economic dispatch problem [18], design optimization of permanent magnet synchronous motor [19], multi-sensor task allocation [20]. The bees algorithm has proven itself in many areas from mechanical design to energy optimization. But it is not used much in applications for combinatorial problems, especially TSP. Some of the highlights from the few applications; single machine scheduling problem [21], combinational circuit design [22], vehicle routing problem [23], a multi-objective supply chain optimization [24].

The basic version of the bees algorithm may be the easiest to implement among heuristic algorithms. The main reason for this is that there is no formulation in the local search strategy. More random trials are made to the neighborhoods of the best solutions, while fewer trials are made to the worse solutions area. With the random new values generated in the global search section, it is avoided to stick to local minimums. As such, the algorithm, which is very effective in numerical problems, cannot be effective in more complex problems such as combinatorial problems. To make the algorithm more effective against combinatorial problems, Koc used improved versions of the BA for problems up to 51 cities in 2010 [25], and Otri used another improved version of the BA for problems up to 100 cities in 2011 [26]. In 2015, Zeybek and Koç added the vantage point approach to the bees algorithm, again working on problems up to 100 cities [27]. Ismail et al. tried the bees algorithm in the scope of problems up to 200 cities within TSPLIB in 2020 [28]. Finally, Zeybek et al. developed the bees algorithm by adding vantage points and tried it on problems up to 318 cities [29].

When the studies in the literature are examined, the bees algorithm is used successfully in combinatorial problems such as production planning and vehicle routing. However, the bees algorithm needs to be improved to solve TSPs with a large number of cities. A new method is proposed in this study in order to increase the probabilities, i.e. to improve the results, without increasing the total number of evaluations. In the second part of the paper, the general structure of the bees algorithm and the structure of the TSP was examined in detail. In the third part, firstly the basic structure of the bees algorithm was created by taking the study numbered [28] as an example. Later, an elitism strategy was developed to improve the algorithm. This algorithm has been applied to different problems within TSPLIB. In the fourth section, test results and analyzes are available.

2. THE TRAVELING SALESMAN PROBLEM AND THE BEES ALGORITHM

TSP is a difficult problem that aims to find the Hamilton path at minimum cost. In TSP, the seller starts from one city and returns to the starting city by visiting all other cities only once. The aim is to complete this traveling with minimum cost. The cost of the tour is directly dependent on the length of the tour. The distance between two successive cities is defined as follows [9]:

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

x and y show the coordinates of the city. The following equation gives the total tour distance [9]:

$$\sum_{i=1}^{n-1} d\big(T(i),T(i+1)\big)+d\big(T(n),T(1)\big) \qquad (2)$$

n represents the total number of cities.

The bees algorithm is a population-based search algorithm. The algorithm mimics the nectar source search behavior of honey bees. Basically, it does some kind of neighbor region search along with random search and can be used for both integrated and functional optimization. Detailed explanations about the algorithm were provided in references [29,30]. The algorithm starts with the determination of some parameters, such as number of scout bees (n), number of the best site (m) for local search, number of the elite site (e) for local search, number of recruited bees in the elite site (nep), number of recruited bees in the best site (nsp), the size of the neighborhood for each patch (ngh), number of iterations (I). After that, the cost function, constraints of the problem, and ranges of the variables are defined. Then, the steps of the basic bees algorithm given in Figure 1 are applied [15].

```
1: Generate random population and evaluate fitness.
2: Sort population
3: While (stopping criterion not met)
4:      For i=1:EliteSite
5:              Recruit bees for elite site and evaluate fitness
6:              Select the best bee from each site
7:      End
8:      For i=EliteSite+1:BestSite
9:              Recruit bees for best site and evaluate fitness
10:             Select the best bee from each site
11:     End
12:     For i= BestSite+1:ScoutBee
13:             Generate random population and evaluate fitness.
14:     End
15:     Sort population
16: End While
```

Figure 1. Pseudo code of the bees algorithm

Search sections and neighborhood definitions in TSPs are different than numerical problems. In general, by changing the location of the cities on the route according to certain rules, it is tried to find new routes with less cost. For the bees algorithm, Ismail et al. used three different operators in the local search section [28]. These processes are similarly used in other heuristic algorithms. In the swap operation, the order of two randomly selected cities is replaced with each other. In the insertion operation, the ranking of a randomly selected city is inserted to a different order, which is also randomly selected. In reversion, the order between two randomly determined cities is reversed. The pseudo-codes of these processes are shown in detail in the related article [28]. It is simply shown on the figures below for reminder purposes. These processes are used randomly in the local search section, with more numbers in elite sites and fewer numbers in the best sites.

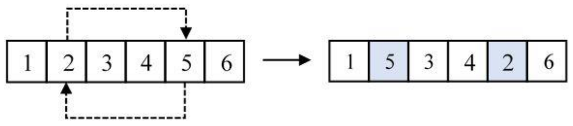


Figure 2. Swap operation

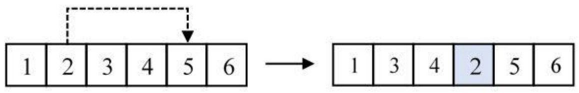


Figure 3. Insertion operation

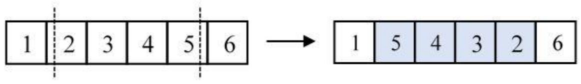


Figure 4. Reversion operation

3. PROPOSED NEW METHOD

When the bees algorithm shown in the previous section is examined, it is seen that single city changes are made in the swap and insertion operations. In problems where the number of cities is low, it is possible to reach the best values with these changes. However, it is clear that in problems with 100 or more cities, the number of iterations or recruited bees must increase in order to reach the best values. As can be seen from the sample study, while there is a deviation of 2-3% in the average values in 100-city problems, it goes up to 5% in 150 cities and 10% in 200 cities [28]. A new method is proposed in this study in order to increase the probabilities, ie to improve the results, without increasing the total number of evaluations. This method consists of two parts. In the first part, the elite and best site searches are made as in the classic bees algorithm. In these searches, 3 operators given in the previous section are used. Before each trial, one of these operators is chosen randomly. In the second part, a proposed variable neighborhood search is made only in the elite site. In this section, a random number between 2 and 5% of the colony (the number of the colony (nc)/20) is determined first. This will change the location of the selected number of cities. The concept of variable comes from here.

Then one of the 2 different search operators is randomly selected. The first operator is the variable mixing and reverse operator. Two different areas are selected as in the reverse operation. But this time, only one area is selected. The order of the cities in that area for the previously determined number is reversed. There may be many cities between two different areas determined in the classical reverse process. This process, which is done with a large number of cities in the first part of the search, helps to find better routes. However, in the following iterations, it is thought that these big changes are not very beneficial as the routes approach the optimum value. Therefore, this operator is used for fewer cities here. And also, this time, there is no reverse operation only. There are two options. These procedures are again determined randomly. The first option of these is the reverse operation. The effect of this process on the city order is shown in Figure 5.

The second option of these is the mixing operator. When this operator is selected, the reverse operation is not performed and random mixing is performed on the same data. The order of the cities is randomly mixed within itself and placed in the same area. This is why it is mixing. Figure 6. shows this process on the example. The effect of this process on the city order is shown in Figure 7.

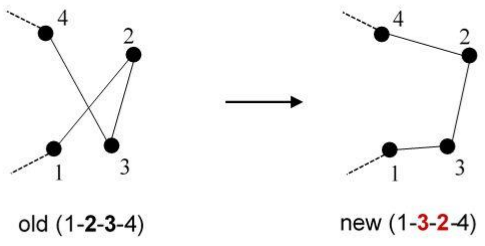


Figure 5. Variable reversion effect

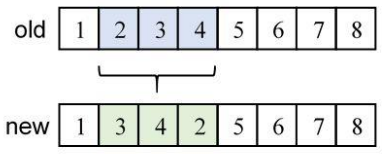


Figure 6. Mixing

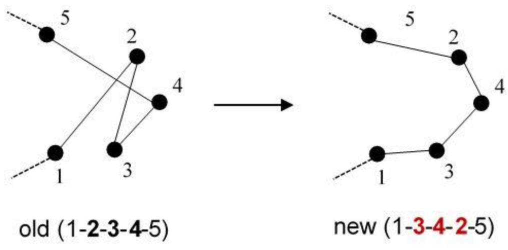


Figure 7. Mixing effect

The second operator is the multiple insertion. Unlike the insertion process described in the previous section, instead of a single city, a multiple variable cities are taken from the selected area. These cities are placed in the second area determined randomly. This process is shown in Figure 8. The effect of this process on the city order is shown in Figure 9.

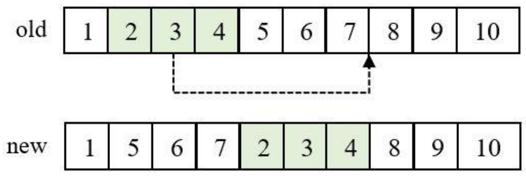


Figure 8. Multiple insertion

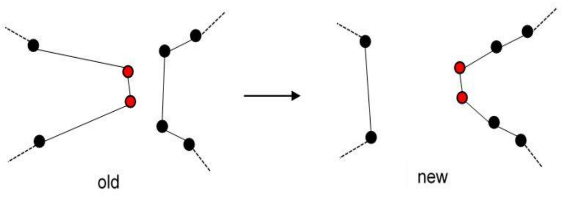


Figure 9. Multiple insertion effect

This process is also applied in two different ways. In the first alternative, selected cities are inserted into the other area in the same order. In the second alternative, the order of the cities is reversed and then inserted. So, different variations can be tried with two different alternative methods. The decision for this procedure is determined randomly. The purpose of this method is to make good routes more elite. Therefore, the number is limited to 5% of the colony when making multiple selections. Especially in the insert process, it is aimed to transfer the cities located very close to each other to another place together. As can be seen from Figure 10., while the number of clusters is less in problems with 100 cities (such as 4-5), it is higher in problems with 200 cities (around 10). Therefore, the number of multiple selections is limited to 5%. And also it is not effective to change the location of many cities on routes that reach a certain maturity.

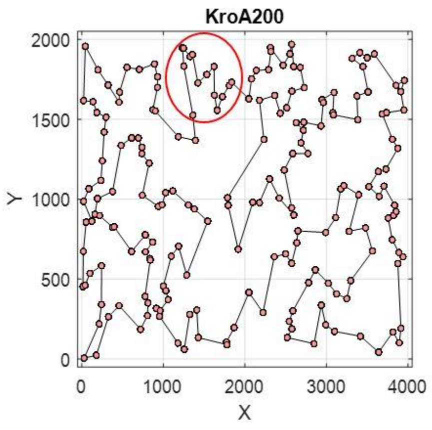


Figure 10. TSPs with 200 cities

The pseudo code of the bees algorithm with variable neighborhood method is given in Figure 11. The pseudo codes of the functions described in this section are given in the appendix. Population sorting is done in the 28th step. Thus, the elite 5 sites are determined and the next operations are carried out only in these sites. The number of recruited bees in the best site section is used in this section. Thus, the total number of evaluations is not increased. This situation directly affects the working time and processing load. The algorithm parameters used in this study are given in Table 1. While determining the parameters, care was taken not to exceed the total number of evaluations used in the other study [28]. The number of elite and best sites was reduced and the number of recruited bees increased. For TSPs with 100 or more cities, a large number of trials are required to make changes to a solution in as many different regions as possible. With the increase in the number of recruited bees, the opportunity to experiment more on fewer regions has been obtained. Reducing the number of sites further may result in local minimums. In order to make a complete comparison, experimental studies have been carried out with the classical bees algorithm, using the determined new parameters. If you look at the table, while the best site is 15 in the classic bees algorithm, this number is 10 in the proposed new method. Thus, the total number of evaluations was the same for both algorithms.

```
while (stopping criterion not met)
    for i=1:EliteSite
        for j=1:EliteSiteBee
            Generate random number n (1-3)
            if n==1
                Swap(EliteSite(i))
            elseif n==2
                Reverse(EliteSite(i))
            else
                Insertion(EliteSite(i))
            end
        end
    Selecet the best bee from each site
    end
    for i=EliteSite+1:BestSite
        for j=1:BestSiteBee
            Generate random number n (1-3)
            if n==1
                Swap(BestSite(i))
            elseif n==2
                Reverse(BestSite(i))
            else
                Insertion(BestSite(i))
            end
        end
    Selecet the best bee from each site
    end
    Sort population
    for i=1:EliteSite
        for j=1: BestSiteBee
            Generate random number m (1-2)
            if m==1
                MixReverse(EliteSite(i))
            else
                MultipleInsertion(EliteSite(i))
            end
        end
    Selecet the best bee from each site
    end
    for i=BestSite+1:ScoutBee
        Generate random population and
evaluate fitness
    end
    Sort population
end while
```

Figure 11. Pseudo code of the bees algorithm with variable neighborhood method

Table 1. Parameters of the algorithms

Parameter	Classic Bees with New Parameters	Proposed New Method
Number of runs	10	10
Number of iterations	1000	1000
Number of scout bees (n)	40	40
Number of elite bees (nep)	200	200
Number of best bees (nsp)	100	100
Number of elite sites (e)	5	5
Number of best sites (m)	15	**10**

4. EXPERIMENTAL ANALYSIS AND RESULTS

In this study, firstly nine problems were selected from TSPLIB [31] for comparison with the classical bees algorithm [28]. The experimental results, together with the results of the other study, are shown in Table 2. The first column of Table 2 gives the TSPLIB dataset names and the lap length for the best known solution (BKS). The next 4 columns contain the results of the other study [28]. The last eight columns show the results of this study. Davg expression gives the deviation of the mean value from the best known value. Dbst shows the deviation rate of the best value found in the result of 10 runs in the study from the best known value. Davg and Dbest equations are defined in (3) and (4) [28].

$$Davg = \frac{(Avg-BKS)}{(BKS)} * 100\% \tag{3}$$

$$Dbst = \frac{(Best-BKS)}{(BKS)} * 100\% \tag{4}$$

As can be seen from the Table 2, with the proposed new method, the best value can be found (Approached with a deviation of 0.02% for KroA100) for problems with 100 cities. If attention is paid to the table, it seems that better values were found for KroB100 and KroD100 than the best known value. However, these values were found to be reached by other researchers [32, 33]. For problems with 150 cities, values with a deviation of 0.8-1.25% from the best known value were found. The average for 10 runs has deviations of about 2.6-2.9%. For problems with 200 cities, there are deviations around 2.5% in the best value and around 5-5.5% in the average value.

As can be seen from the results, with the proposed new method, much better results than the classical bees algorithm have been obtained. Moreover, while these values are obtained, the maximum number of evaluations is less than the other study. The best values found for each problem are shown in the Appendix -2. Also, as can be seen from the second part of the table (Classic Bees with New Parameters), increasing the number of recruited bees of the bees algorithm for TSPs has a positive effect. Although the number of sites, the number of iterations, and the number of the total evaluation were reduced, better results were obtained.

Table 2. Experimental results and comparison with the classic bees algorithm

Dataset names (BKS)	Classic Bees Algorithm [28]				Classic Bees with New Parameters				**Proposed New Method**			
	Best	Avg	Davg	Dbst	Best	Avg	Davg	Dbst	Best	Avg	Davg	Dbst
KroA100 (21282)	21469	21712	2.02	0.88	21322	21521	1.12	0.19	**21285**	**21466**	**0.86**	**0.02**
KroB100 (22141)	22618	22879	3.33	2.15	22269	22735	2.69	0.58	**22139**	**22467**	**1.47**	**0**
KroC100 (20749)	20969	21339	2.85	1.06	20813	21071	1.55	0.31	**20750**	**20923**	**0.84**	**0**
KroD100 (21389)	21392	21896	2.37	0.01	21613	21830	2.07	1.05	**21384**	**21597**	**0.98**	**0**
KroE100 (22068)	22266	22496	1.94	0,90	**22078**	22394	1.47	**0.05**	**22078**	**22336**	**1.22**	**0.05**
KroA150 (26524)	27527	27871	5.08	3.78	27340	27778	4.73	3.08	**26740**	**27299**	**2.92**	**0.82**
KroB150 (26130)	27095	27458	5.08	3.69	26733	27093	3.69	2.31	**26457**	**26833**	**2.69**	**1.25**
KroA200 (29368)	31550	32214	9.69	7.43	31266	31609	7.63	6.46	**30098**	**30899**	**5.22**	**2.49**
KroB200 (29437)	32024	32519	10.47	8.79	31254	31946	8.52	6.17	**30218**	**31104**	**5.66**	**2.65**

Finally, experiments were conducted with 7 different TSPs to test the algorithm. The results are shown in Appendix. Similar to other experiments, deviation values of less than 1% for the best values and 2% for the average values were obtained for problems up to 150 cities. For 200 and above problems, these values vary between 1.5% and 6%. The comparison study was made with other metaheuristic algorithms. All Davg and Dbest data are shown in Table 3. For comparison, the Genetic Algorithm (GA) [34] and Tabu Algorithm (GA) [34], which has been used in this field for many years, and Improved Vantage Point Bees Algorithm (IVP-BA) [29] which is an improved version of the Bees Algorithm, were chosen. The results obtained with the proposed method were more successful than the results of other studies. It is much better, especially at average values. It can be said that these results are strong in terms of repeatability. In the biggest problem of 318 cities, it was able to calculate lower routes than the others.

Table 3. Comparison with other algorithms

Datasets	New Method		IVP – BA [29]		GA [34]		TA [34]	
	Davg	Dbst	Davg	Dbst	Davg	Dbst	Davg	Dbst
KroA100	**0.86**	**0.02**	1.70	0.42	2.22	0.12	2.69	0.06
KroB100	**1.47**	**0**	1.74	0.26	2.14	0.89	4.28	2.62
KroC100	**0.84**	**0**	1.81	0.09	-	-	-	-
KroD100	**0.98**	**0**	2.84	0.86	-	-	-	-
KroE100	**1.22**	**0.05**	1.39	0.23	-	-	-	-
KroA150	**2.92**	**0.82**	4.70	3.06	5.18	2.10	4.48	2.85
KroB150	**2.69**	**1.25**	3.79	2.73	-	-	-	-
KroA200	5.22	2.49	8.10	7.16	5.22	**2.18**	**4.43**	2.58
KroB200	**5.66**	**2.65**	9.67	8.05	-	-	-	-
Lin318	**5.89**	**4.20**	7.57	7.02	13.05	9.89	6.87	4.59

5. CONCLUSIONS

In this study, it is aimed to improve the bees algorithm towards TSP. First of all, the number of parameters used in previous applications has been changed. The number of elite sites and best sites, which is the basis of the bees algorithm, was kept small, and the number of recruited bees to search in these sites was increased. Thus, it is aimed to develop these sites by making more searches on fewer solutions. In order to improve and diversify search methods, two new functions have been prepared. Unlike the single membership insertion process, a multiple insertion function has been prepared. By determining the number of variables, a different number of choices were allowed in each trial. The reverse process has also been handled differently and the second function has been developed here. With the new method, it is seen that much better results are obtained compared to the other studies performed with the bees algorithm. It is seen that the error values decrease to 0% of the best values in 100 cities. In cities of 150 and 200, the error values are around 2.5%.

Acknowledgments

The author would like to thank Roketsan A.S. for their financial support for this work.

REFERENCES

[1] Z. Daoqing, J. Mingyan, "Parallel Discrete Lion Swarm Optimization Algorithm for Solving Traveling Salesman Problem", *Journal of Systems Engineering and Electronics*, 31(4), 751 – 760, 2020.

[2] J. Xie, L. R. G. Carrillo, L. Jin, "An Integrated Traveling Salesman and Coverage Path Planning Problem for Unmanned Aircraft Systems", *IEEE Control Systems Letters*, 3(1), 67-72, 2019.

[3] X. Meng, J. Li, X. Dai, J. Dou, "Variable Neighborhood Search for a Colored Traveling Salesman Problem", *IEEE Transactions on Intelligent Transportation Systems*, 19(4), 1018-1026, 2018.

[4] E. Çelik, N. Öztürk, "Doğru Akım Motor Sürücüleri için PI Parametrelerinin Simbiyotik Organizmalar Arama Algoritması ile Optimal Ayarı", *Bilişim Teknolojileri Dergisi*, 10(3), 311-318, 2017.

[5] M. Mavrovouniotis, F. M. Müller, S. Yang, "Ant Colony Optimization with Local Search for Dynamic Traveling Salesman Problems", *IEEE Transactions on Cybernetics*, 47(7), 1743-1756, 2017.

[6] X. Chen, P. Zhang, G. Du, F. Li, "Ant Colony Optimization Based Memetic Algorithm to Solve Bi-Objective Multiple Traveling Salesmen Problem for Multi-Robot Systems", *IEEE Access*, 6, 21745-21757, 2018.

[7] E. Liao, C. Liu, "A Hierarchical Algorithm Based on Density Peaks Clustering and Ant Colony Optimization for Traveling Salesman Problem", *IEEE Access*, 6, 38921-38933, 2018.

[8] Y. Liu, B. Cao, "A Novel Ant Colony Optimization Algorithm with Levy Flight", *IEEE Access*, 8, 67205-67213, 2020.

[9] D. Karaboga, B. Gorkemli, "Solving Traveling Salesman Problem by Using Combinatorial Artificial Bee Colony Algorithms", *International Journal on Artificial Intelligence Tools*, 28(1), 1-28, 2019.

[10] X. Dong, Q. Lin, M. Xu, Y. Cai, "Artificial bee colony algorithm with generating neighbourhood solution for large scale coloured traveling salesman problem", *IET Intelligent Transport Systems*, 13(10), 1483-1491, 2019.

[11] L. Wang, R. Cai, M. Lin, Y. Zhong, "Enhanced List-Based Simulated Annealing Algorithm for Large-Scale Traveling Salesman Problem", *IEEE Access*, 7, 144366-144380, 2019.

[12] X. Han, Y. Dong, L.Yue, Q. Xu, "State Transition Simulated Annealing Algorithm for Discrete-Continuous Optimization Problems", *IEEE Access*, 7, 44391-44403, 2019.

[13] A. Arram, M. Ayob, G. Kendall, A. Sulaıman, "Bird Mating Optimizer for Combinatorial Optimization Problems", *IEEE Access*, 8, 96845-96858, 2020.

[14] I. K. Gupta, A. Choubey, S. Choubey, S., "Randomized bias genetic algorithm to solve traveling salesman problem", **8th International Conference on Computing, Communication and Networking Technologies (ICCCNT)**, Delhi, India, 1-6, 2017.

[15] D. T. Pham, A. Ghanbarzadeh, E. Koc, S. Otri, S. Rahim, M. Zaidi, "The Bees Algorithm – A Novel Tool for Complex Optimisation Problems", **Intelligent production machines and systems**, **2nd I*PROMS Virtual International Conference**, 454-459, 2006.

[16] M. Zarchi, B. Attaran, "Performance improvement of an active vibration absorber subsystem for an aircraft model using a bees algorithm based on multi-objective intelligent optimization", *Engineering Optimization Volume*, 49(11), 1905-1921, 2017.

[17] S. Rajagopalan, V. C. Thippana, A. M. Parimi, "Optimal Placement of the Interline Power Flow Controller using the Bees Algorithm to minimize power loss", **4th International Conference on Electrical Energy Systems (ICEES)**, Chennai, India, 212-217, 2018.

[18] M. K. Sharma, P. Phonrattanasak, N. Leeprechanon, "Improved bees algorithm for dynamic economic dispatch considering prohibited operating zones", **IEEE Innovative Smart Grid Technologies - Asia (ISGT ASIA)**, Singapore, 1-6, 2015.

[19] N. Y. Braiwish, F. J. Anayi, A. A. Fahmy, E. E. Eldukhri, "Design optimisation of Permanent Magnet Synchronous Motor for electric vehicles traction using the Bees Algorithm", **49th International Universities Power Engineering Conference (UPEC)**, Cluj-Napoca, Romania, 1-5, 2014.

[20] I. Tkach, Y. Edan, A. Jevtic, S. Y. Nof, "Automatic Multi-sensor Task Allocation Using Modified Distributed Bees Algorithm", **IEEE International Conference on Systems, Man, and Cybernetics**, Manchester, United Kingdom, 1401-1406, 2013.

[21] M. S. Packianather, B. Yuce, E. Mastrocinque, F. Fruggiero, D. T. Pham, A. Lambiase, A., "Novel Genetic Bees Algorithm applied to single machine scheduling problem", **World Automation Congress (WAC)**, Kona, Big Island of Hawaii, 906-911, 2014.

[22] N. Mollabakhshi, M. Eshghi, M., "Combinational circuit design using bees algorithm", **IEEE Conference Anthology**, Piscataway, NJ, 1-4, 2013.

[23] M. Alzaqebah, S. Jawarneh, H. M. Sarim, S. Abdullah, “Bees Algorithm for Vehicle Routing Problems with Time Windows”, *International Journal of Machine Learning and Computing,* 8(3), 236-240, 2018.

[24] B. Yuce, E. Mastrocinque, A. Lambiase, M. S. Packianather, D. T. Pham, “A multi-objective supply chain optimisation using enhanced Bees Algorithm with adaptive neighbourhood search and site abandonment strategy”, *Swarm and Evolutionary Computation*, 18(1), 71-82, 2014.

[25] E. Koc, **Bees algorithm: theory, improvements and applications**, Phd thesis, Cardiff University, UK, 2010.

[26] S. Otri, **Improving the bees algorithm for complex optimisation problems,** Phd thesis, Cardiff University, UK, 2011.

[27] S. Zeybek, E. Koc, “The vantage point bees algorithm”, **7th International Joint Conference on Computational Intelligence (IJCCI),** Lisbon, Portugal, 340–45, 2015.

[28] A. H. Ismail, N. Hartono, S. Zeybek, D. T. Pham, “Using the Bees Algorithm to solve combinatorial optimisation problems for TSPLIB”, **IOP Conf. Series: Materials Science and Engineering 847**, Indonesia, 1-9, 2020.

[29] Zeybek, S., Ismail, A. H., Hartono, N., Caterino, M., & Jiang, K., “An Improved Vantage Point Bees Algorithm to Solve Combinatorial Optimization Problems from TSPLIB”, *In Macromolecular Symposia,* 396(1), 1-4, 2021.

[30] L. Baronti, M. Castellani, D. T. Pham, “An analysis of the search mechanisms of the bees algorithm”, *Swarm and Evolutionary Computation*, 59, 1-30, 2020.

[31] G. Reinelt, “Tsplib - a traveling salesman problem library”, *ORSA journal on computing,* 3(4), 376–84, 1991.

[32] W. Tan, J. Gao, J. Rao, “Ant colony algorithm based on data classification”, **IOP Conf. Series: Materials Science and Engineering,** 768(7), Shanghai, China, 072099, 2020.

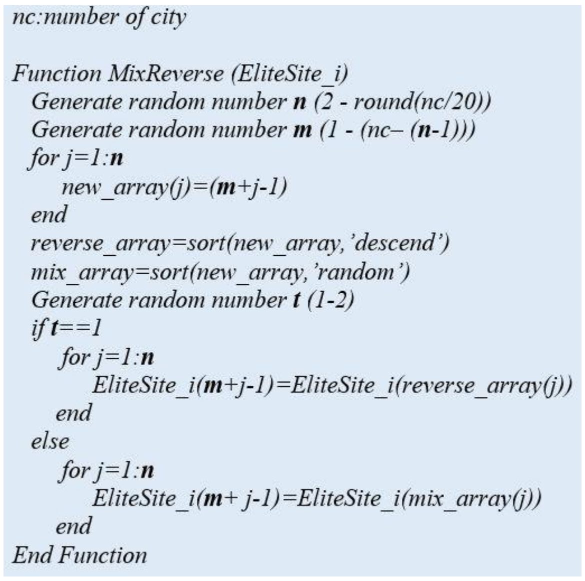
[33] M. R. Othman, Z. A. Othman, “Ayman Ibraheem Srour, Nor Samsiah Sani, A Hybrid Water Flow-Like Algorithm and Variable Neighbourhood Search for Traveling Salesman Problem”, *International Journal on Advanced Science, Engineering and Information Technology*, 9(5), 1505-1511, 2019.

[34] Y. Şahin, “Sezgisel Ve Metasezgisel Yöntemlerin Gezgin Satıcı Problemi Çözüm Performanslarının Kıyaslanması”, *Bolu Abant İzzet Baysal Üniversitesi Sosyal Bilimler Enstitüsü Dergisi*, 19(4), 911-932, 2019.

APPENDİX – 1

Table 4. Experimental results for different TSPs

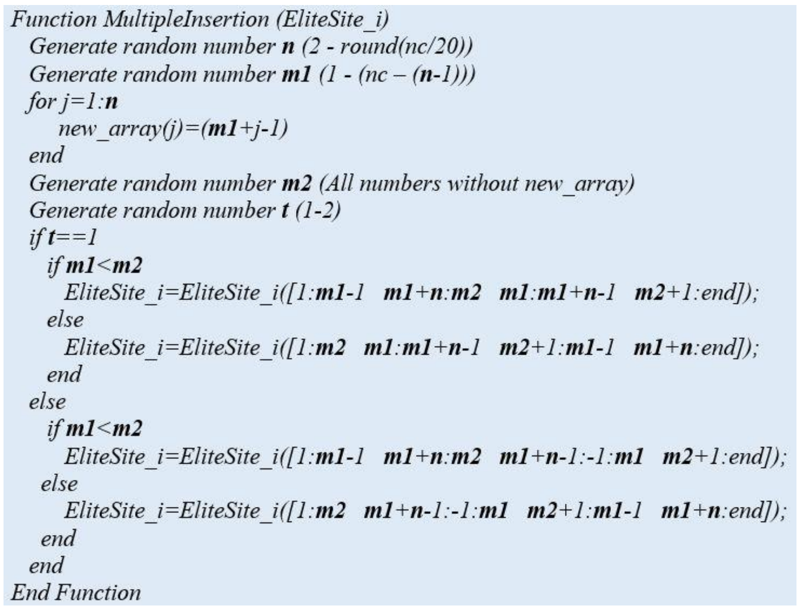
Dataset names (BKS)	Proposed New Method			
	Best	Avg	Davg	Dbst
berlin52 (7542)	7544	7578	0.48	0.03
pr144 (58537)	58813	59349	1.39	0.47
pr152 (73682)	74366	75065	1.88	0.93
D198 (15780)	16021	16190	2.60	1.53
ts225 (126643)	128648	130931	3.39	1.58
pr226 (80369)	82264	82818	3.05	2.36
Lin318 (42029)	43796	44505	5.89	4.20



```
nc:number of city

Function MixReverse (EliteSite_i)
  Generate random number n (2 - round(nc/20))
  Generate random number m (1 - (nc– (n-1)))
  for j=1:n
     new_array(j)=(m+j-1)
  end
  reverse_array=sort(new_array,'descend')
  mix_array=sort(new_array,'random')
  Generate random number t (1-2)
  if t==1
     for j=1:n
        EliteSite_i(m+j-1)=EliteSite_i(reverse_array(j))
     end
  else
     for j=1:n
        EliteSite_i(m+ j-1)=EliteSite_i(mix_array(j))
     end
End Function
```

Figure 12. Pseudo code of MixReverse Function



```
Function MultipleInsertion (EliteSite_i)
  Generate random number n (2 - round(nc/20))
  Generate random number m1 (1 - (nc – (n-1)))
  for j=1:n
     new_array(j)=(m1+j-1)
  end
  Generate random number m2 (All numbers without new_array)
  Generate random number t (1-2)
  if t==1
    if m1<m2
      EliteSite_i=EliteSite_i([1:m1-1  m1+n:m2  m1:m1+n-1  m2+1:end]);
    else
      EliteSite_i=EliteSite_i([1:m2  m1:m1+n-1  m2+1:m1-1  m1+n:end]);
    end
  else
    if m1<m2
      EliteSite_i=EliteSite_i([1:m1-1  m1+n:m2  m1+n-1:-1:m1  m2+1:end]);
    else
      EliteSite_i=EliteSite_i([1:m2  m1+n-1:-1:m1  m2+1:m1-1  m1+n:end]);
    end
  end
End Function
```

Figure 13. Pseudo code of MultipleInsertion Function

APPENDİX – 2

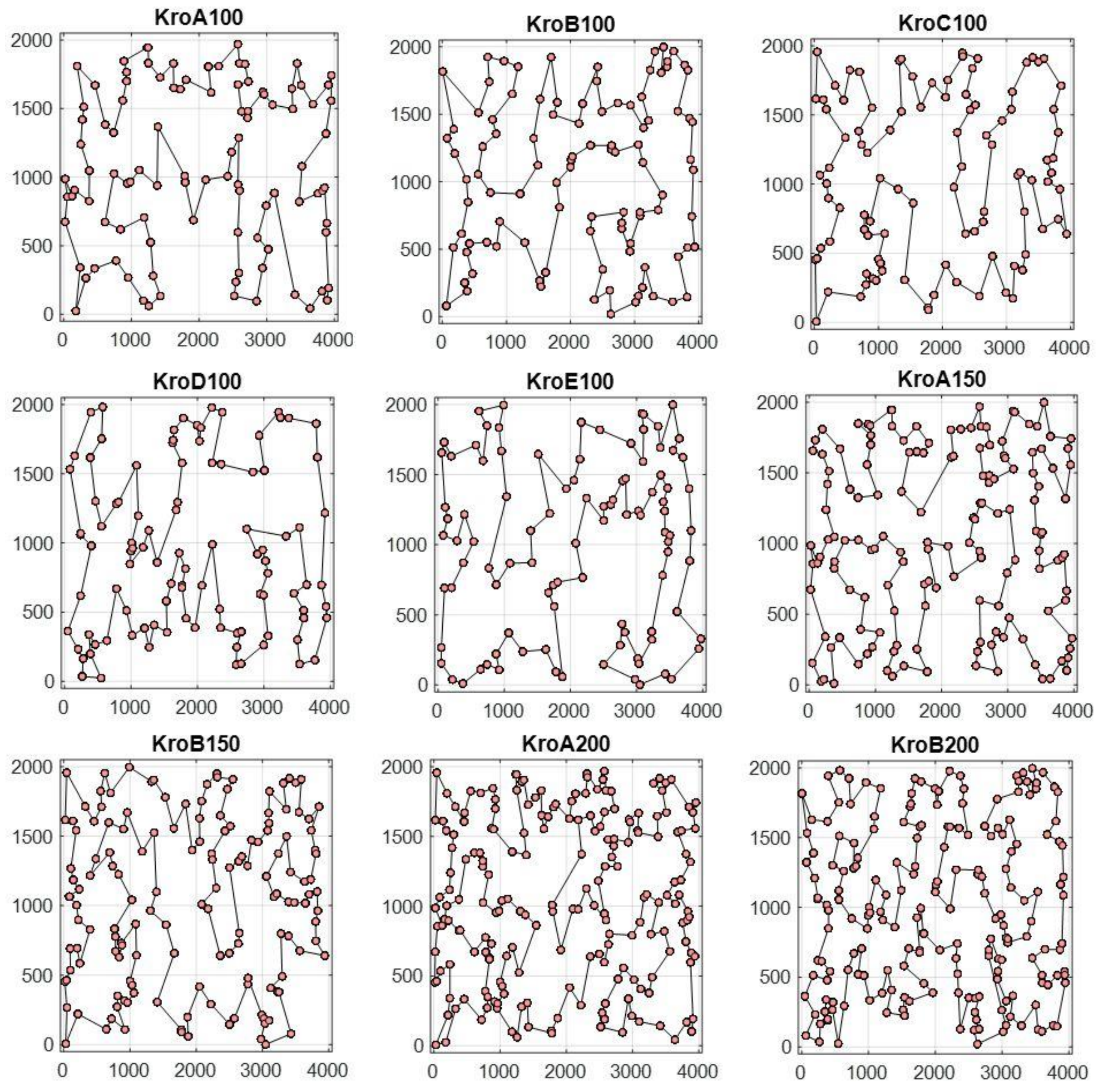


Figure 14. Best solution of TSPs