A Novel Crossover based Discrete Artificial Algae Algorithm for Solving Traveling Salesman Problem

Refik Nureddin
Department of Computer Engineering
Konya Technical University, Turkey
refik.nureddin@gmail.com

Ismail Koc
Department of Software Engineering
Sciences, Konya Technical University
Turkey
ismailkoc@ktun.edu.tr

Sait Ali Uymaz
Department of Computer Engineering
Sciences, Konya Technical University
Turkey
sauymaz@ktun.edu.tr

Abstract: *The Artificial Algae Algorithm (AAA) is a newly proposed metaheuristic algorithm that is inspired by microalgae behaviors. This algorithm has been proposed for solving continuous optimization problems and achieved good results for the continuous problems. In addition, binary versions of AAA are proposed in the literature. This paper presents a discrete version of AAA, which is named Discrete Artificial Algae Algorithm (DAAA). For discretization of AAA, Crossover operators (one-point and uniform) are used in the processes (helical movement, evolutionary process, and adaptation). In this study, in addition to crossover operators, transformation operators such as swapping, insertion, symmetry, and reversion are also used. DAAA's performance was analyzed on a well-known discrete optimization problem called the Traveling Salesman Problem (TSP). DAAA was tested on thirty-two Benchmark instances of the TSP. These instances were small-sized, medium-sized, and large-sized. Firstly, the AAA processes (evolutionary process, adaptation, and helical movement) with the combination of nearest neighbor and transformation operators were tested for selected benchmark instances and this testing was called Process Analysis. After this process Analysis the best processes with which to continue were selected, and after this decision comparisons with other algorithms were started. The main comparison is between discrete Social Spider Algorithm (DSSA) and DAAA, and DAAA outperformed DSSA on most of the problems. Further, DAAA's performance on some of the benchmark instances was compared with some of the well-known algorithms for TSP. In this comparison, DAAA has achieved better results than many other algorithms. Experimental results show that DAAA has the capability of solving discrete optimization problems and outperforming other algorithms.*

Keywords: *Artificial algae algorithm, traveling salesman problem, crossover operator, transformation operators, discrete optimization.*

Received February 14, 2024; accepted September 2, 2024
https://doi.org/10.34028/iajit/21/5/14

1. Introduction

The focus of much scientific research in optimization is on Routing problems. Their popularity and importance stem from their complexity and these problems are difficult to solve. Therefore, they attract more interest and study than many other problems. To solve this problem, classical mathematic and heuristic methods have been used. The solutions to these problems are not only of interest in business and logistics, but also are valuable for tourism and entertainment sector Osaba *et al*. [42].

The most popular routing problem is the Traveling Salesman Problem (TSP). In 1930, the TSP was introduced as a mathematical problem, and its popularity increased after 1950 [6].

TSP is well known in the scientific community for discrete optimization problems. Regarding this problem, the goal is to find the shortest tour path. Other than for business, logistics, and tourism routing problems also have been used in a lot of engineering applications as designing hardware devices, radio-electronic systems, and computer networks [21].

TSP is based on visiting every city and returning to the starting city but with the condition of visiting each city just one time. TSP is NP-hard problem, and the complexity of the TSP is the main reason it is a commonly used problem in the scientific world.

With an increasing number of cities, it is more difficult to find the optimum solution for TSP with classical mathematical methods. Researchers have started to use heuristic methods or discrete optimization algorithms for achieving better results and improving the optimal solution in computational and time complexity [21].

TSP has been used for comparison in many optimization methods; for example, Nearest Neighborhood Search (NNS), Simulated Annealing (SA), Tabu Search (TS), Neural Networks (NNs), Ant Colony System (ACS) and Genetic Algorithm (GA).

Bas and Ulker [5] proposed a new Discrete Social Spider Algorithm (DSSA) for TSP. Mahi *et al*. [38] proposed a new hybrid method based on Particle Swarm Optimization (PSO), Ant Colony Optimization (ACO) and 3-Opt algorithms for TSP. The proposed algorithm uses beneficial sides of these three algorithms. Rokbani *et al*. [47] are using bi-heurtic method for combining

different methods and mechanisms. They have proposed bi-heuristic ACO-based approaches for TSP. Cinar *et al*. [10] proposed a new discrete tree-seed algorithm for solving symmetric TSP. This proposed algorithm uses some techniques of tree-seed algorithm to solve discrete problem. Pedro *et al*. [44] proposed a TS approach for the prize collecting TSP. This method is one of the very useful methods for solving TSP. Dorigo and Gambardella [15] proposed that ACO has the capacity to achieve good solutions in both symmetric and asymmetric instances of the TSP. Mavrovouniotis and Yang [39] proposed an ACO framework for dynamic environments. Karaboga and Gorkemli [31] proposed a new Artificial Bee Colony (ABC) algorithm called Combinatorial ABC for the TSP. Additionally, Hijazi *et al*. [25] proposed ABC algorithm for the purpose of feature selection.

Artificial Algae Algorithm (AAA) is proposed for continuous optimization problems. AAA has three processes. In the Evolutionary Process, an algal colony which receives enough light grows and reproduces itself to generate two new algal cells in time, similar to the real mitotic division.

The second process is the Adaptation process, in which an insufficiently grown algal colony tries to resemble itself to the biggest algal colony in the environment. And the last process is the helical movement process in which the movements of algal cells differ. As the friction surface of a growing algal cell gets larger, the frequency of helical movements increases by increasing their local search. For Discrete Artificial Algae Algorithm (DAAA), the process of crossover operators has been successfully implemented. The results for TSP became better after applying the crossover operator in the method mentioned above. Adding 2-opt, roulette wheel methods, and transform operators also improved the results. With the combination of crossover operators and these methods DAAA achieved better results.

1.1. Literature Review

TSP is a well-known problem in the scientific world, the complexity is NP-Hard but there are a lot of proposed algorithms which nowadays provide an optimal solution. Gunduz and Aslan proposed [22] A newly discrete Jaya algorithm achieves good results for TSP. This algorithm was used in random permutations and neighborhood to solve TSP, and after using these methods 2-opt also was implemented for achieving better results. Pandiri and Singh [43] implemented a k-interconnected method on ABC that achieved better results than standard ABC for the multi-depot multi-TSP. Osaba *et al*. [42] proposed a discrete water cycle algorithm that was tested on 33 problem datasets covering both symmetric and asymmetric TSP problems. Eskandari *et al*. [18] proposed a new hybrid algorithm which is a modified and enhanced ACO for TSP. This algorithm is the hybridization of ACO and Bean Optimization Algorithm (BOA); results of this algorithm were better than ACO and GA. For avoiding local minima 3-opt was used in the newly proposed by Gulcu *et al*. [21] parallel cooperative hybrid method based on ACO and the 3-Opt algorithm for solving TSP; this algorithm gets optimum solutions for Eil51, Berlin52, Eil76, KroA100, Lin105, and Eil101. Ulder *et al*. [52] implemented genetic local search, GA and Multi-Search GA with Lin-Kerhinghan and 2-opt. Choong *et al*. [9] proposed An ABC algorithm with a modified choice function for the TSP, and it was tested on 64 problem datasets using Lin-Kerhinghan. The proposed method solves approximately within 2.7 minutes which is better than other algorithms.

Archetti *et al*. [2] proposed an iterated local search for the TSP with release dates and completion time minimization. The proposed model used destroy and repair method, which means one method is simple and fast, the other method is using mathematical programming formulation. Dodig and Smith [13] proposed a new apple carving algorithm to approximate traveling salesman problem from compact triangulation of planar point sets and was tested with some of TSP benchmark datasets. In applying metaheuristic for time-dependent traveling salesman problem in postdisaster for time-dependent TSP (TDTSP). Ban [4] proposed a model for combining GA and LS to solve TDTSP, and this model gets for many problems better results than others. For handling ACO's disadvantages proposed Adopting dynamic evaporation strategy to enhance ACO algorithm for the TSP [16]. Dynamic evaporation strategy is implemented on ACO and tested on 10 TSP benchmark datasets. A novel metaheuristic algorithm, the domino algorithm, proposed by Ismail [27] for the solution of the TSP, was tested on Eil51, Berlin52, St70, Eil76, Pr76, and Rat99 and results were compared with nearest neighbour approach. Domino algorithm's results were better than nearest neighbour approach. Daoqing and Mingyan [11] proposed for solving the TSP a new parallel Discrete Lion Swarm Optimization (DLSO) algorithm. Firstly, crossover operators on DLSO were used, after that complete 2-opt was used for improving local search abilities. The modification of the hybrid method of ACO, PSO, and 3-OPT algorithm in TSP Hertono *et al*. [24] proposed model utilized the good sides of 3 different algorithms. ACO is used for finding the best solution, PSO for implementing the best value of parameters, and 3-opt for getting better results on local minima. Zhong *et al*. [58] proposed a discrete comprehensive learning PSO algorithm with Metropolis acceptance criterion for TSP that used SA on PSO and used Metropolis acceptance criterion on particles. This model was tested on 6 TSP benchmark methods, and the results were better than the compared algorithms. Shang *et al*. proposed [49] ACO for solving the TSP. Saraei *et al*. [48] proposed solving of TSP using firefly algorithm with greedy approach, on this model is used greedy

approach for solving the TSP. Snyder and Daskin [50] proposed a random-key GA for solving the TSP, on this model GA was combined with local tour improvements. This was tested from 42 to 442 nodes benchmark tests and was published quality and computation time. A hybridization model was proposed by Gunduz *et al*. [23] to solve the TSP, this model has used a hierarchic approach based on swarm intelligence. For achieving good solutions was used ACO, for improving the path was used ABC; this hybridization achieves better results than ACO and ABC individually. Global Local Search (GLS), C2opt, and Smallest Square (SS) were used together to solve the TSP on the proposed model in [19] and Yang *et al*. [56] proposed ACO to solve the generalized TSP, this model for escaping from local minima was used, a mutation technique, and this technique implemented on ACO very well. Gharib *et al*. [20] have a comparison between PSO and GA to the solving the TSP, results were acceptable for both algorithms. Improving variable neighborhood search to solve the TSP [26] is used for the symmetric and asymmetric problem, and the results were acceptable for the symmetric and asymmetric problem. For achieving better results on TSP, proposed analysis of discrete ABC was used in neighborhood operator [33] with combined 2-opt and 3-opt. African Buffalo Optimization (ABO) algorithm was proposed by Odili *et al*. [40] for solving the TSP, African Buffalo's organization abilities was used on this proposed model. The results were better than other algorithms.

Odili *et al*. [41] have a comparative study between ABO and Randomized Insertion Algorithm (RIA). In the proposed model ABO used Karp-Steele approach, while RIA used random insertion method, and the RIA achieved better results were achieved than ABO. Liu and Zeng [36] proposed GA with reinforcement learning to solve the TSP, which uses heterogonous pairing selection, and the proposed model was tested on small and medium TSP benchmark methods. Wang *et al*. [55] proposed multi-offspring GA was to solve the TSP; this proposed model implements biological evolutionary and mathematical ecological theory methods. This proposed method was faster than other methods. Rokbani *et al*. [46] proposed the model combined Particle Swarm Optimization-Ant Colony Optimization-Local Search (PSO-ACO-LS) with gravitational search to solve the TSP, and results were acceptable for some of the TSP benchmark datasets. Taillard and Helsgaun [51] proposed Partial Optimization Metaheuristic Under Special Intensification Conditions (POPMUSIC) for solving the TSP, this proposed model was used in large size problems, and results were acceptable. Beam search and ACO [37] were combined to solve the TSP with a time window, and the proposed algorithm was implemented very well on the TSP with time window. Akhand *et al*. [1] proposed a newly algorithm Discrete Spider Monkey Optimization (DSMO) was proposed to solve the TSP, which implements Swap Sequence (SS) and Swap Operator (SO), and the results were better than other compared algorithms. Khan and Maiti [32] implemented K-opt with SS and SO on ABC and tested this method on asymmetric and symmetric TSP benchmark instances.

Uymaz *et al*. [53] proposed a novel bio-inspired AAA for nonlinear global optimization for continuous problems, and the proposed algorithm was tested with CEC 05' instances and the results were compared with other algorithms, achieving acceptable results. With modified multi-light source movement AAA was implemented to the multi-light source for numerical optimization and applications [54] and tested in real-world optimization problems IEEE-Congress on Evolutionary Computation (IEEE-CEC) 11'. Multidimensional Knapsack Problem was solved with Binary AAA (BAAA), the proposed [57] algorithm was tested on 94 benchmark instances and results were better than other algorithms. AAA was implemented to solve multi-objective continuous problems. Babalik *et al*. proposed [3] the newly multi-objective algorithm was tested on 36 different multi-objective problems, and the results were better than compared algorithms. AAA and Simplex Search Method (SSM) were hybridized for economic load dispatch problems, on the proposed [34] algorithm AAA was for global searching and SSM for local searching. The proposed method was tested on CEC 05' for small, medium, and large size problems. Results were better than other algorithms. Every day renewable energy sources become more popular and more important in our world. For getting this energy some tools are needed, and one of the most important tools is Wind Turbine; BAAA was proposed by Beskirli *et al*. [7] for solving this wind turbine problem. The proposed algorithm achieved the best results for this problem.

TSP algorithms can be applied in the field to optimize multiple gas turbine engines. Given that this is a large-scale problem, traditional algorithms often fall into local optima when solving such expansive TSP challenges. The newly proposed algorithm, ITO [14] addresses this issue and achieves a better solution quality than the compared state-of-the-art algorithms. Hybrid algorithms achieve better solutions for large-scale TSP problems. Kanna *et al*. [30] proposed an EW-DHOA developed by integrating two well-performing meta-heuristic algorithms: the Deer Hunting Optimization Algorithm (DHOA) and the Earthworm Optimization Algorithm (EWA). Experimental results show that the convergence of the proposed hybrid optimization is superior when solving TSP, with reduced computational complexity, and it offers significant improvement in attaining optimal results.

Problems involving Multiple Drones (MD) can be formulated as TSP problems Cavani *et al*. [8] proposed exact methods to address this issue, using Mixed-Integer Linear Programming (MILP) and a branch-and-cut

algorithm. These approaches effectively solved the TSP-MD problem and achieved excellent results. The exact method Flying Assistant Traveling Salesman Problem (FSTSP) Dell'Amico *et al*. proposed [12] to address the drone problem, and the results obtained were competitive with other algorithms.

1.2. Contribution

This paper covers a novel discrete algorithm of AAA for solving symmetric TSP, and this study presents the first version of the AAA which was coded and implemented to the most popular discrete problem TSP. The parameters of AAA (evaluation process, adaptation, and helical motion) were applied for discrete optimization problems with crossover operators (one-point and uniform). The main goal of the research work presented in this manuscript is to prove efficiency of the DAAA in solving the TSP. Achieving results for TSP of DAAA are compared with DSSA. Experiments have been adapted to TSP datasets exporting from publicly available benchmark instances. As the obtained results clearly show, the proposed DAAA achieves better results compared to DSSA.

There are some differences between our study and the studies that are available in the literature. The differences are listed below.

DAAA is a novel and alternative discrete optimization method for discrete optimization problems. The new method works in a crossover based discrete space, and includes the application of transformation operators, and the nearest neighbor method. Also process analysis has been performed for the proposed method, and this process analysis is a combination between AAA processes (evolutionary process, adaptation, and helical movement) and Nearest Neighbor and Transformation operators. This combination is tested on selected Benchmark instances.

Experimental results show that DAAA achieves acceptable and competitive results for TSP.

The remaining of this work is organized as listed below.

- Section 2 contains a description of AAA.
- Section 3 contains a description of the proposed discrete version of AAA.
- Section 4 contains the experimental Results.
- Section 5 contains a comparison of the DAAA results with DSSA results, and with other algorithms' results.
- Section 6 contains the conclusion of the paper.

2. Artificial Algea Algorithm (AAA)

Uymaz *et al*. [53] proposed AAA for solving continuous optimization problems. This algorithm was developed with the inspiration of the behavior of microalgae. It was also inspired from characteristic properties. The newly proposed method has become a part of biological metaheuristic methods. AAA consists of three parts and those parts are helical movement, evolutionary process, and adaptation [53]. In the proposed algorithm the population consists of several algal colonies and each colony has several individual algae each of which plays an important role in the algorithm. The following equation represents the population:

$$Population\ of\ Algal\ Colony = \begin{bmatrix} x_1^1 \cdots \cdots x_1^d \\ x_2^1 \cdots \cdots x_2^d \\ \vdots \cdots \cdots \vdots \\ x_n^1 \cdots \cdots x_n^d \end{bmatrix} \quad (1)$$

$x_i = \left(x_i^1, x_i^2, \ldots., x_i^d\right) i = 1,2, \ldots. N$, x_i is every viable solution in solution space. x_i^j is algal cell, N is the number of algal colony and D is the number of decision variables. The problem dimension is equal to the number of algal cells in each algal colony.

2.1. Helical Movement

Algae perform typical movements to reach adequate light and other nutrients in the water areas. In AAA, all algal colonies move toward the best algal colony. The algal colony moves in three dimensions like its real-world movements. The movement randomly selecting three distinct algal cells and changing their positions.

$$x_{im}^{t+1} = x_{im}^t + \left(x_{jm}^t - x_{im}^t\right)(sf - \omega_i)p \quad (2)$$

$$x_{ik}^{t+1} = x_{ik}^t + \left(x_{jk}^t - x_{ik}^t\right)(sf - \omega_i)\cos\alpha \quad (3)$$

$$x_{il}^{t+1} = x_{il}^t + \left(x_{jl}^t - x_{il}^t\right)(sf - \omega_i)\sin\beta \quad (4)$$

where m, k and l are random integers between 1 and d, x_{im}, x_{ik} and x_{il} simulate x, y and z coordinates of the *ith* algal colony, j is the index of a neighbor algal colony, p is an independent random number between -1 and 1, $\alpha$and β are random degrees of arc between 0 and 2, sf is shear force, andω_i is the friction surface area of ith algal colony.

2.2. Evolutionary Process

An algal colony tries to reach sufficient food source, growing rapidly if it reaches the source and dying if it fails. The same is true for AAA: if the algal colony moves to an ideal position that x_i becomes bigger than its starting position. The smallest algal colony is changed by a bigger algal colony's cell. The equations below show the process:

$$biggest = arg\ max(size(x_i)), \quad i = 1,2,\ldots.,n \quad (5)$$

$$smallest = arg\ min\left(size(x_i)\right), \quad i = 1,2,\ldots.,nb \quad (6)$$

$$smallest_j = biggest_j, \quad j = 1,2,\ldots.,n \quad (7)$$

where *smallest* and *biggest* are the biggest and smallest algal colony, and j is a randomly selected algal cell.

2.3. Adaptation

An algal colony bears starvation in the growing process

when there are insufficient light and nutrients. After starvation the algal colony tries to become part of the biggest colony and adapts itself to the environment. Initially, the value of starvation is zero and it is changed with helical movement, it can get worse or better. Below is the equation of adaptation process:

$$x_s = arg\,max(starvation(x_i)),\ \ i = 1,2,\ldots.n \quad (8)$$

$$x_{sj}^{t+1} = \{x_{sj}^t + (biggest_j - x_{sj}^t) \times rand1, if\ rand2 < A_p;\ \ j = 1,2,\ldots d \quad (9)$$

where s is the index of starvation with the highest value in the algal colony.

The flowchart for AAA is shown in Figure 1.

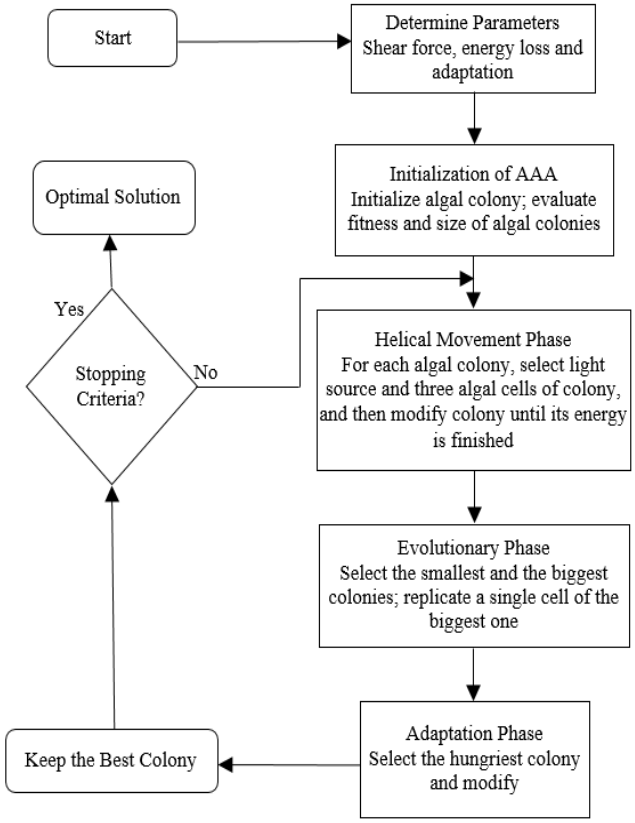

Figure 1. Flow chart of AAA.

3. Discrete Artificial Algea Algorithm (DAAA)

AAA is developed for continuous optimization problems,and AAA was reorganized and modified to solve discrete optimization problems. This novel developed discrete optimization algorithm was named DAAA. The organization of the DAAA processes is as follows:

The first phase was implementing crossover operators [29] on the AAA's methods (Helical movement, Evaluation process, and Adaptation), this operation is described in the following subsections.

Other than crossover operators nearest neighbour search was applied in the first phase [28].

Transformation operators were used in the second phase [59] for improving the results. These transformation operators implemented on the AAA's methods were Swapping, Reversing, Inserting and Symmetrization.

2-opt [17] and roulette-wheel selection [35] were implemented in the last phase for achieving better results.

3.1. Evolutionary Process with Crossover

In Table 1 the evolutionary process if an algal colony does not reach a sufficient food source then a random selected cell of the smallest algal colony is changed by a bigger algal colony's cell. We implement crossover here in the biggest and smallest algal colony. One point crossover was implemented on randomly chosen cells from these colonies with these cells changing places. To improv the results we use uniform crossover,where each cell is chosen from either parent with equal probability. These operations are described in Figures 2 and 3.

Table 1. Example algal colony of evalutanory process.

Xworst	3	7	6	5
Xbest	2	4	1	8

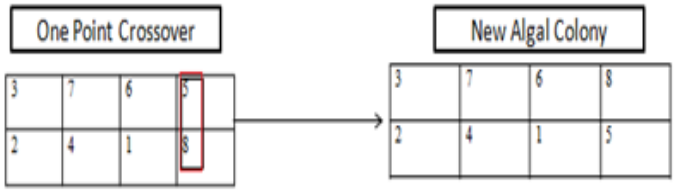

Figure 2. One-point crossover for evolutionary process.

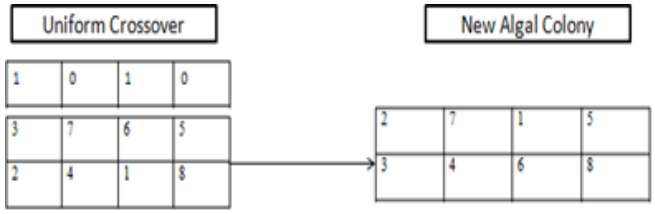

Figure 3. Uniform crossover for evalutanory process.

3.2. Adaptation Process with Crossover

In Table 2, algal colony starvation is increasing when there are insufficient light and nutrients, and the algal colony is becoming part of the biggest colony and it is adapting to the environment. We use one point and uniform crossover on that process. These operations are described in Figures 4 and 5.

Table 2. Example algal colony of adaptation process.

Xstarving	2	3	7	6
Xbest	1	4	5	8

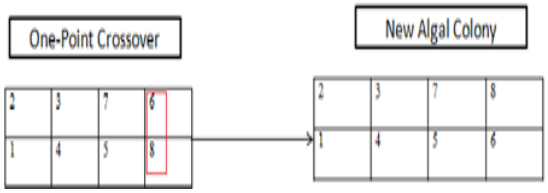

Figure 4. One-point crossover for adaptation process.

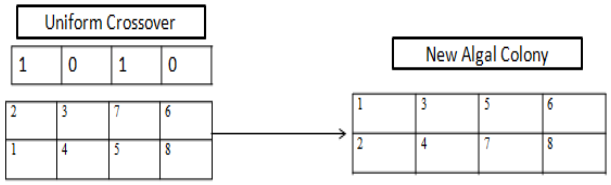

Figure 5. Uniform crossover for adaptation process.

3.3. Helical Movement Process with Crossover

In Table 3 to reaching suificient light and other nutriment in the water areas, algal colony is doing helical movements. In these movements cells are chosen randomly and their positions are changed. After these movements the crossover operations (one point and uniform) are implemented in DAAA to improve the actual solutions as showen in Algorithm (1). These operations are described in Figures 6 and 7.

Table 3. Example algal colony of adaptation process.

X1	13	2	5	11
X2	1	4	3	12

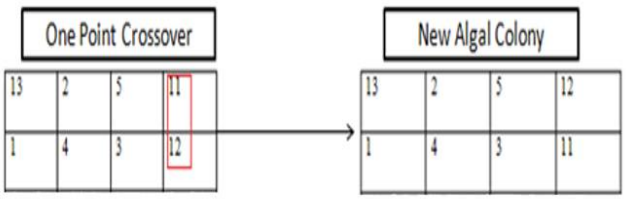

Figure 6. One-point crossover for helical movement.

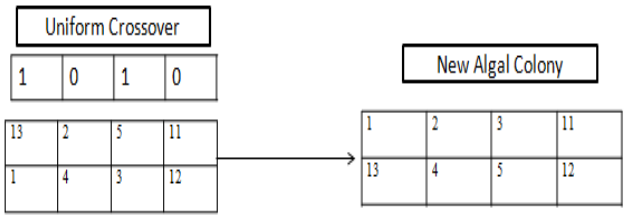

Figure7. Uniform crossover for helical movement.

Algorithm 1: The Pseudo Code of DAAA.

Initialize a population of n algal colonies with random solutions
Evaluate size (G) of n algal colonies
Define the parameters (shear force∆,loss of energy e and
Adaptation parameter A_p)
Create algal colonies with a random permutation
Set Maximum Fitness Calculation Number as MaxFEVs

While Fes is smaller than MaxFEVs

If rand < 0.25
NNS algae= Apply Nearest Neighbour Search
Replace the worst algae with NNS algae
End

Start with evolutionary process
Evaluate the worst and best algal colony
Implement crossover to the algal colony and get new algal colony
If rand <0.90 than One point crossover
XworstTemp=Xworst;
OnePointCRIndex=randi(number of dimension)
Apply OnePointCR for Xbest
Apply OnePointCR for Xworst
Xworst=Xbest;
End
If new cost < current cost of the algae
Replace new algae with the current algae
Starve Counter of the current algae =0;
else
Starve Counter of the current algae ++;
end
Else
Uniform crossover
End

Continue with adaptation process
Evaluate the starvation of algal colony
Maxstarving= Xstarving
for j =1: CrossoverSize
Temporary Xstarving =Xstarving;
OnePointCRIndex=randi(number of dimension-CrossoverSize)
Apply OnePointCR for Xbest
Apply OnePointCR for Xstarving
Xstarving=Xbest;

End
...
If new cost < current cost of the algae
Replace new algea with the current algae
Starve Counter of the current algae =0;
else
Starve Counter of the current algae ++;
end
Else
Uniform crossover
End

Continue with helical movement
Select a RandomAlgae
RandomAlgae = Algs
If rand <0.90 than One point crossover
Temporary RandomAlgae = RandomAlgae;

OnePointCRIndex=randi(number of dimension)
Apply OnePointCR for currentAlgae
Apply OnePointCR for Random Algae
RandomAlgae=currentAlgae;

End

If new cost < current cost of the algae
Replace new algae with the current algae
Starve Counter of the current algae =0;
else
Starve Counter of the current algae ++;
end
Else
Uniform crossover
End

For each algae in algae population
Select one operation by Roulete wheel selection
Case 1:
Do Swap
Case 2:
Do Reversion
Case 3:
Do Insertion
Case 4:
Do Symmetry
End

Change the positions of the algae according to the selected
operation.

If new cost < current cost of the algae
Replace new algae with the current algae
Starve Counter of the current algae =0;
else
Starve Counter of the current algae ++;
end

End of while

Apply the 2-Opt algorithm to the global best solution

Report the best as the obtained best solution
End

2-opt, Roulette-wheel selection and transformation operators were used in the proposed method for improving the solutions achieved from crossover operations implemented on the evolutionary process, adaptation and helical movement, but 2-opt was used only once.

4. Traveling Salesman Problem

The TSP involves a salesman starting his tour from one city and finishing the tour back at the starting city, aiming to visit all necessary cities in the shortest route possible. All cities (which are necessary to visit) are included in that tour. The main achievement is to complete this tour on the shortest route. The number of possible tours is the hardest part of finding the shortest paths: There are *(n-1)!* Possible tours for n cities. This term signifies the total number of permutations possible when visiting n distinct cities. Each permutation represents a unique order of visiting all cities exactly once. The euclidean distance is used for calculating distance between city *i* and city *j*. There are two types of TSP problems, symmetric and asymmetric; this work uses benchmark tests of symmetric problems [7].

5. Experimental Results and Analysis

The benchmark tests which are used in this study are obtained from TSPLIB [45]. There are 32 benchmark tests used for our experiments. These are: Eil51, Berlin52, Eil76, Pr76, Rat99, Rd100, KroA100, KroB100, KroC100, KroD100, KroE100, Lin105, Pr107, Pr124, Pr136, Pr144, KroA150, Pr152, Rat195, KroA200, Ts225, Tsp225, Pr226, Pr264, Pr299, Lin318, Linhp318, Rd400, Pr439, Rat575, Rat783, Pr1002.

The used benchmark tests are small-sized, middle-sized, and large-sized dimensions. The best results obtained for each problem are marked in boldface font type to ease the comparison. Each problem was tested 10 times. Experiments were run on a Windows 10 professional OS laptop using Intel Core i7 2.80 GHz CPU, 16GB RAM and Windows 10 64-bit operating system, and MATLAB was used for implementing the codes.

The first experimental benchmark instances with DAAA processes were tested, the general results are in Table 4. For the naming processes in this experiment the first letter of every process was used: N-nearest neighbor, E-evolutionary process, H-helical movement, A-adaptation, T-transformation operators. The purpose of this analysis was to identify the most effective combination of processes for solving the TSP benchmark instances (through this analysis, we sought to identify the most effective process for solving benchmark instances of the TSP. This analysis starts with three combinations Nearest Neighbor, Evolutionary Process, Adaptation (NEA) and finishes with five combinations NEA-Helical Movement, Adaptation, Transformation Operators (NEAHT); every possible combination between the processes was tested. Measurements for comparing are the following values: Mean (the average result obtained after testing the problem ten times), Standard Deviation (in optimization measures how much individual solutions vary from the average solution, indicating the consistency of optimization results (Std)), Best (the best result obtained after testing the problem ten times), Worst (the worst result obtained after testing the problem ten times). For Linhp318 the NEA process obtained best results in Mean, Std, and Worst, for Rat575 NEH process obtained best Std,for Linhp318 NHTEM process obtained best results in the Best value, for Berlin52 Nearest Neighbor, Helical Movement, Adaptation (NAH) process obtained best Std, for tsp225 NAT process obtained best results in Mean and Worst, for KroA100 NEAT process obtained best results in Mean, for Tsp225 in the Best, for KroA100 NEAHT process obtained best results in the best value.

Tables 5 and 6 show the results of ranking the process analyzing of Mean and the Best values. The NET and Nearest Neighbor, Evolutionary Process, Adaptation, Transformation Operators (NEAT) processes obtained best ranking for the Mean value. In the Best value, the best ranking results are achieved from NEAT and NAHT processes. The NEAT process achieved better results than other processes in both tables; only in Table 5 NET is better than NEAT. Following these results, it was decided to continue with NEAT, so nearest neighbor, evolutionary process, adaptation, and transformation operators were used for the next tests and comparisons. Following the results in process analysis, general comparisons were continued with NEAT.

Table 7 shows the result of small-sized problems for 5000 iterations, the instances for this experiment are Burma 14, Ulysses16, Ulysses22, and Bays29. The error rate is 0 for all instances, Std is 0 for Burma 14, Ulysses16, and Ulysses 22.

Table 8 has general results in the medium-sized and large-sized problems for 100,000 and 50,000 iterations. By comparing the results, the Error rate is lower for 100.000 iterations, especially for the large-sized problems.

Table 4. General results of process analysis.

Processes	Problem	Mean	Std	Best	Worst
NEA	Berlin52	7911.40	**0.00**	7911.40	7911.40
	KroA100	22144.75	**37.31**	22120.22	**22229.24**
	Tsp225	4333.88	9.47	4314.81	4341.10
	Linhp318	**46099.76**	**11.00**	46068.44	**46103.24**
	Rat575	7297.34	20.12	**7277.55**	**7334.53**
NEH	Berlin52	7911.40	**0.00**	7911.40	7911.40
	KroA100	22440.14	485.23	22049.39	23403.30
	Tsp225	4274.45	49.78	4191.56	4343.97
	Linhp318	46360.99	224.79	46177.97	46784.95
	Rat575	7342.51	**15.39**	7316.01	7361.66
NET	Berlin52	**7555.18**	22.80	**7544.37**	**7598.44**
	KroA100	21955.08	264.98	21643.34	22572.68
	Tsp225	4228.47	55.15	4146.62	4316.16
	Linhp318	46162.55	316.37	**45631.33**	46591.38
	Rat575	7352.40	27.89	7325.95	7417.55
NAH	Berlin52	7911.40	**0.00**	7911.40	7911.40
	KroA100	22321.35	282.86	22120.22	22933.20
	Tsp225	4272.97	65.26	4118.62	4349.84
	Linhp318	46259.50	176.02	46141.51	46752.13
	Rat575	7351.03	35.38	7305.29	7435.27
NAT	Berlin52	7578.10	60.67	7544.37	7713.03
	KroA100	22085.43	457.82	21712.37	23291.65
	Tsp225	**4196.48**	42.62	4124.53	**4257.60**
	Linhp318	46205.37	271.99	45823.55	46752.92
	Rat575	7354.11	30.10	7319.30	7412.39
NHT	Berlin52	7628.18	70.90	7544.37	7716.63
	KroA100	22174.50	290.63	21622.30	22738.69
	Tsp225	4209.43	37.50	4150.60	4268.75
	Linhp318	46400.35	291.99	45999.13	46969.28
	Rat575	7358.14	50.34	7313.72	7463.24
EAH	Berlin52	8619.16	256.41	8250.59	8968.58
	KroA100	27539.26	1187.26	26349.69	29738.17
	Tsp225	5497.64	231.13	4958.49	5802.25
	Linhp318	62390.17	2424.13	56619.23	64715.46
	Rat575	10430.64	202.01	9934.01	10730.37
EAT	Berlin52	8042.15	209.04	7669.62	8408.26
	KroA100	24700.81	965.69	23420.86	25913.52
	Tsp225	5196.09	141.12	5004.44	5460.52
	Linhp318	61868.30	1944.54	58288.12	64140.80
	Rat575	10448.19	213.07	10088.54	10786.88
EHT	Berlin52	8011.56	205.57	7760.07	8427.79
	KroA100	25825.42	1078.48	24097.45	27602.09
	Tsp225	5379.07	183.84	5126.16	5655.10
	Linhp318	61443.75	2151.57	57839.60	66127.09
	Rat575	10399.18	259.19	10127.14	10962.35
AHT	Berlin52	8083.31	226.07	7606.95	8385.30
	KroA100	26277.18	1727.69	24346.70	29479.48
	Tsp225	5450.53	267.32	5255.86	6093.93
	Linhp318	62701.98	2880.93	59127.96	68967.38
	Rat575	10564.55	339.64	10308.28	11426.90
NEAH	Berlin52	7902.92	19.37	7853.25	7911.40
	KroA100	22386.42	386.29	22120.22	23280.51
	Tsp225	4255.50	37.88	4180.79	4327.08
	Linhp318	46375.29	241.31	46177.97	46784.95
	Rat575	7344.56	26.47	7316.67	7404.06
NEAT	Berlin52	7574.20	53.60	7544.37	7713.03
	KroA100	**21907.48**	251.60	21626.07	22385.22
	Tsp225	4209.94	52.64	**4115.87**	4283.04
	Linhp318	46101.07	254.20	45656.50	46439.31
	Rat575	7386.69	63.86	7317.56	7540.22
NEHT	Berlin52	7642.72	88.34	7544.37	7805.41
	KroA100	22180.92	205.21	21807.79	22401.74
	Tsp225	4240.39	50.63	4183.38	4352.48
	Linhp318	46225.27	316.08	45659.16	46750.15
	Rat575	7347.75	46.99	7291.45	7452.66
NAHT	Berlin52	7637.10	55.66	7544.37	7713.03
	KroA100	22078.31	239.30	21607.16	22355.23
	Tsp225	4245.66	60.29	4147.30	4313.86
	Linhp318	46263.68	261.62	45632.33	46468.20
	Rat575	7353.94	29.59	7305.73	7410.22
EAHT	Berlin52	8118.54	120.61	7855.50	8292.64
	KroA100	26403.94	712.19	25115.57	27227.13
	Tsp225	5359.01	139.47	5079.38	5632.13
	Linhp318	63497.47	1746.88	61162.83	66747.49
	Rat575	10438.94	237.64	10075.53	10865.96
NEAH	Berlin52	7664.34	78.24	7544.37	7800.20
	KroA100	22258.45	299.03	**21591.29**	22655.73
	Tsp225	4257.74	41.07	4200.59	4324.02
	Linhp318	46248.18	401.58	45693.22	46837.52
	Rat575	7353.63	39.67	7325.95	7461.64

Table 5. Rank results of process analysis according to mean values.

Process Names	Berlin52	KroA100	Tsp225	Linhp318	Rat575	Total rank	Average rank	Final rank
NEA	7911.40	22144.75	4333.88	46099.76	7297.34			
	9	5	11	1	1	27	3.4	4
NEH	7911.40	22440.14	4274.45	46360.99	7342.51			
	10	11	10	9	2	42	5.3	11
NET	7555.18	21955.08	4228.47	46162.55	7352.40			
	1	2	4	3	6	**16**	**2.0**	**1**
NAH	7911.40	22321.35	4272.97	46259.50	7351.03			
	11	9	9	7	5	41	5.1	10
NAT	7578.10	22085.43	4196.48	46205.37	7354.11			
	3	4	1	4	9	21	2.6	3
NHT	7628.18	22174.50	4209.43	46400.35	7358.14			
	4	6	2	11	10	33	4.1	7
EAH	8619.16	27539.26	5497.64	62390.17	10430.64			
	16	16	16	14	13	75	9.4	16
EAT	8042.15	24700.81	5196.09	61868.30	10448.19			
	13	12	12	13	15	65	8.1	13
EHT	8011.526	25825.42	5379.07	61443.75	10399.18			
	122.4	13	14	12	12	63	7.9	12
AHT	8083.3191	26277.18	5450.53	62701.98	10564.55			
	1411	14	15	15	16	74	9.3	15
NEAH	7902.292	22386.42	4255.50	46375.29	7344.56			
	83	10	7	10	3	38	4.8	9
NEAT	75741.20	21907.48	4209.94	46101.07	7386.69			
	2	1	3	2	11	**19**	**2.4**	**2**
NEHT	7642.72	22180.92	4240.39	46225.27	7347.75			
	6	7	5	5	4	27	3.4	5
NAHN	7637.10	22078.31	4245.66	46263.68	7353.94			
	5	3	6	8	8	30	3.8	6
EAHT	8118.54	26403.94	5359.01	63497.47	10438.94			
	15	15	13	16	14	73	9.1	14
NEAHT	7664.34	22258.45	4257.74	46248.18	7353.63			
	7	8	8	6	7	36	4.5	8

Table 6. Rank results of process analysis according to best values.

Process Names	Berlin52	KroA100	Tsp225	Linhp318	Rat575	Total rank	Average rank	Final rank
NEA	7911.40	22120.22	4314.81	46068.44	7277.55			
	13	9	11	8	1	42	5.3	9
NEH	7911.40	22049.39	4191.56	46177.97	7316.01			
	14	8	9	10	6	47	5.9	10
NET	7544.37	21643.34	4146.62	45631.33	7325.95			
	2	5	4	1	10	22	2.8	3
NAH	7911.40	22120.22	4118.62	46141.51	7305.29			
	15	10	2	9	3	39	4.9	8
NAT	7544.37	21712.37	4124.53	45823.55	7319.30			
	1	6	3	6	9	25	3.1	4
NHT	7544.37	21622.30	4150.60	45999.13	7313.72			
	7	3	6	7	5	28	3.5	6
EAH	8250.59	26349.69	4958.49	56619.23	9934.01			
	16	16	12	12	12	68	8.5	14
EAT	7669.62	23420.86	5004.44	58288.12	10088.54			
	9	12	13	14	14	62	7.8	12
EHT	7760.07	24097.45	5126.16	57839.60	10127.14			
	10	13	15	13	15	66	8.3	13
AHT	7606.95	24346.70	5255.86	59127.96	10308.28			
	8	14	16	15	16	69	8.6	15
NEAH	7853.25	22120.22	4180.79	46177.97	7316.67			
	11	11	7	11	7	47	5.9	11
NEAT	7544.366	21626.07	4115.869	45656.5	7317.561			
	3	4	1	3	8	**19**	**2.4**	**2**
NEHT	7544.37	21807.79	4183.38	45659.16	7291.45			
	4	7	8	4	2	25	3.1	5
NAHN	7544.37	21607.16	4147.30	45632.33	7305.73			
	5	2	5	2	4	**18**	**2.3**	**1**
EAHT	7855.50	25115.57	5079.38	61162.83	10075.53			
	12	15	14	16	13	7	8.8	16
NEAHT	7544.37	21591.29	4200.59	45693.22	7325.95			
	6	1	10	5	11	33	4.1	7

Table 7. Results of small sized problems for 5,000 iterations.

Problem	Best	Worst	Mean	Std	Error %
Burma14	3323.0	3323.0	3323.0	0.00	0.00
Ulysses16	6747.0	6747.0	6747.0	0.00	0.00
Ulysses22	6901.0	6901.0	6901.0	0.00	0.00
Bays29	2020.0	2026.0	2021.2	2.53	0.00

Table 8. General results of the TSP problems for 50,000 and 100,000 iterations.

	100,000 iterations					**50,000 iterations**				
Problem	**Best**	**Worst**	**Mean**	**Std**	**Error**	**Best**	**Worst**	**Mean**	**Std**	**Error**
Eil51	430.4	434.3	432.6	1.33	**1.04**	430.7	438.3	433.6	2.24	1.11
Berlin52	7,544.4	7,544.4	7,544.4	0.00	**0.00**	7,544.4	7,544.4	7,544.4	0.00	**0.00**
Eil76	553.2	560.6	557.0	2.53	**2.82**	552.6	564.5	557.6	3.53	2.72
Pr76	108,882.4	110,633.1	109,898.2	538.47	**0.67**	109,352.5	110,512.2	109,977.9	423.32	1.10
Rat99	1,221.0	1,230.4	1,223.7	3.23	**0.83**	1,221.0	1,234.3	1,227.4	5.55	**0.83**
Rd100	8,038.3	8,237.6	8,149.8	66.47	**1.62**	8,122.2	8,233.8	8,183.2	34.17	2.68
KroA100	21,316.4	21,497.7	21,411.7	64.08	0.16	21,294.4	21,703.0	21,455.2	125.64	**0.06**
KroB100	22,337.2	22,697.0	22,448.8	110.94	0.89	22,248.0	22,800.5	22,462.8	149.82	**0.48**
KroC100	20,930.1	21,244.4	21,104.9	102.06	0.87	20,923.6	21,424.0	21,168.8	160.54	**0.84**
KroD100	21,974.4	22,359.4	22,184.5	149.45	3.20	21,731.7	22,343.2	22,086.0	190.38	**2.06**
KroE100	22,204.0	22,413.6	22,319.8	86.78	0.62	22,136.7	22,420.2	22,308.7	95.74	**0.31**
Lin105	14,475.4	14,687.5	14,576.1	71.91	**0.67**	14,489.1	14,779.6	14,623.6	99.03	0.77
Pr107	44,301.7	44,324.8	44,306.3	9.76	**0.00**	44,301.7	44,391.7	44,323.9	33.83	**0.00**
Pr124	59,245.9	59,416.8	59,312.7	86.24	**0.37**	59,245.9	60,052.8	59,409.5	252.46	0.37
Pr136	101,663.2	103,913.2	103,247.1	691.02	**5.05**	102,743.9	104,180.0	103,674.4	504.49	6.17
Pr144	59,021.4	60,704.1	60,173.5	663.23	0.83	58,721.3	60,719.9	60,251.2	706.06	**0.31**
KroA150	27,173.4	27,585.0	27,404.1	149.87	**2.45**	27,258.0	27,816.5	27,472.5	188.05	2.77
Pr152	74,255.8	74,785.9	74,479.8	181.33	**0.78**	74,300.5	74,710.3	74,538.2	144.46	0.84
Rat195	2,369.8	2,415.4	2,387.2	12.35	**2.01**	2,376.5	2,411.5	2,394.2	12.50	2.30
KroA200	29,578.8	29,986.6	29,825.9	117.89	**0.72**	29,937.4	30,961.2	30,274.5	346.51	1.94
Ts225	127,397.8	128,958.4	128,201.8	441.01	**0.60**	128,297.1	132,182.8	129,325.6	1,154.57	1.31
Tsp225	3,978.5	4,096.2	4,034.3	32.84	**1.52**	4,014.4	4,151.1	4,081.6	38.56	2.43
Pr226	81,416.5	82,638.6	82,238.0	403.22	**1.30**	81,828.1	83,857.8	82,674.5	547.12	1.82
Pr264	50,835.8	52,312.1	51,815.8	485.30	**3.46**	51,568.1	52,552.5	52,135.8	300.17	4.95
Pr299	50,563.7	51,196.4	50,984.6	237.19	**4.92**	50,847.6	52,135.1	51,579.7	462.29	5.51
Lin318	44,156.8	45,585.5	44,700.2	402.74	**5.06**	44,482.5	45,554.8	45,010.3	346.91	5.84
Linhp318	44,141.2	45,354.7	44,620.2	328.80	**6.76**	44,708.0	45,856.4	45,198.4	403.14	8.13
Rd400	16,107.4	16,533.0	16,294.1	136.49	**5.41**	16,237.6	16,689.8	16,521.7	136.63	6.26
Pr439	112,135.0	115,489.3	113,979.4	986.27	**4.59**	115,230.3	117,834.1	116,219.6	900.27	7.47
Rat575	7,260.9	7,342.1	7,297.8	28.44	**7.20**	7,238.7	7,423.9	7,323.8	61.40	6.88
Rat783	9,559.4	9,767.0	9,686.6	74.66	**8.56**	9,680.5	9,871.7	9,757.0	62.88	9.93
Pr1002	280,340.8	286,863.3	283,424.5	2,137.71	**8.22**	281,839.8	287,653.1	284,594.3	2,017.77	8.80

Table 9. The comparison of the proposed algorithm with DSSA in the TSP benchmark.

				DSSA			**DAAA**		
	Problem	**Size**	**Optimal**	**Best**	**Worst**	**Error%**	**Best**	**Worst**	**Error%**
1	Burma14	14	3323	**3323.0**	7702.0	**0.00**	**3323.0**	**3323.0**	**0.00**
2	Ulysses16	16	6859	**6859.0**	14230.0	**0.00**	**6859.0**	**6859.0**	**0.00**
3	Ulysses22	22	7013	**7013.0**	18325.0	**0.00**	**7013.0**	**7013.0**	**0.00**
4	Bays29	29	2020	**2020.0**	6709.0	**0.00**	**2020.0**	**2026.0**	**0.00**
5	Eil51	51	426	431.9	1835.0	1.38	**430.5**	**434.3**	**1.04**
6	Berlin52	52	7544	7659.0	31432.0	1.55	**7544.4**	**7544.8**	**0.00**
7	Eil76	76	538	559.3	2720.4	3.96	**552.6**	**564.5**	**2.72**
8	Pr76	76	108159	**108880.0**	621210.0	**0.67**	108882.4	**110633.1**	**0.67**
9	Rat99	99	1211	**1221.0**	9237.0	**0.83**	**1221.0**	**1230.4**	**0.83**
10	Rd100	100	7910	8120.0	59865.0	2.65	**8038.3**	**8237.6**	**1.62**
11	KroA100	100	21282	**21363.0**	189380.0	0.38	**21294.4**	**21703.00**	**0.06**
12	KroB100	100	22141	22347.0	188300.0	0.93	**22248.0**	**22800.5**	**0.48**
13	KroC100	100	20749	20997.0	191830.0	1.19	**20923.6**	**21424.0**	**0.84**
14	KroD100	100	21294	21552.0	147160.0	**1.21**	21731.7	**22343.2**	2.06
15	KroE100	100	22068	22407.0	198040.0	1.53	**22136.7**	**22420.2**	**0.31**
16	Lin105	105	14379	14502.0	135670.0	0.85	**14475.5**	**14687.5**	**0.67**
17	Pr107	107	44303	44346.0	68720.0	**0.09**	**44303.0**	**44324.8**	**0.00**
18	Pr124	124	59030	**59087.0**	76722.0	0.09	59245.9	**59416.8**	0.37
19	Pr136	136	96772	103460.0	914060.0	6.91	**101663.2**	**103913.2**	**5.05**
20	Pr144	144	58537	58669.0	87320.0	**0.22**	58721.4	**60719.9**	0.31
21	KroA150	150	26524	**27027.0**	359350.0	**1.90**	27173.4	**27585.0**	2.45
22	Pr152	152	73682	74462.0	112480.0	1.05	**74255.8**	**74786.0**	**0.78**
23	Rat195	195	2323	**2353.0**	24826.0	**1.29**	2369.8	**2415.4**	2.01
24	KroA200	200	29368	29666.0	373590.0	1.01	**29578.8**	**29986.6**	**0.72**
25	Ts225	225	126643	**127230.0**	1578500.0	**0.46**	127397.8	**128958.4**	0.60
26	Tsp225	225	3919	**3933.0**	43032.0	**0.35**	3978.5	**4096.2**	1.52
27	Pr226	226	80369	82186.0	180840.0	2.26	**81416.5**	**82638.6**	**1.30**
28	Pr264	264	49135	**50739.0**	61355.0	**3.26**	50835.8	**52312.1**	3.46

The main comparison in this study is between DSSA and the proposed method DAAA. The results of this comparison are presented in Table 9. Results for DSSA are taken from DSSA for the travelıng salesman problem [5]. For having a fair comparison, we use the same numbers from the above-mentioned work: the iteration number is 100000 and dimension is the number of the cities. In the presented table we have a comparison of the three values (Best, Worst, and Error). The Mean values are not taken because in the DSSA work Mean and Best values were the same. DAAA is over performing the DSSA in most of the problems. The

proposed method has better results in all the areas (Best, Worst and Error) for the following problems: Eil51, Berlin52, Eil76, Rd100, KroA100, KroB100, KroC100, KroE100, Lin105, Pr107, Pr136, Pr152, KroA200 and Pr226. For all problems, DAAAs Worst values are better than DSSA's. DAAA and DSSA achieved similar Best and Error rate values in the following instances: Burma14, Ulysses16, Ulysses22, Pr76, and for Rat99.

Table 10. Comparison of the other algorithms.

Method	Criteria	Bays29	Fri26	Gr17	Eil51	Berlin52	Eil76	KroA100	KroB100	KroC100
DVNS	Mean	-	-	-	-	-	-	-	-	-
	SD	-	-	-	-	-	-	-	-	-
	Error (%)	0.00	0.00	0.00	0.69	0.03	1.37	-	-	-
DWCA	Mean	-	-	-	-	-	-	-	-	
	SD	-	-	-	2.00	-	3.30	47.90	164.40	124.60
	Error (%)	-	-	-	-	-	-	-	-	-
GA	Mean	-	-	-	440.80	7542	565.40	21812.40	22687.40	21510.40
	SD	-	-	-	7.30	0.00	9.80	420.80	407.70	390.20
	Error (%)	-	-	-	-	-	-	-	-	-
IDGA	Mean	-	-	-	434.4	7542	557.70	21731.80	22712.60	21298.70
	SD	-	-	-	4.50	0.00	6.80	340.70	312.80	290.70
	Error (%)	-	-	-	-	-	-	-	-	-
Discrete ESA	Mean	-	-	-	431.60	7542	553.70	21481.70	22602.20	21170.40
	SD	-	-	-	2.90	0.00	4.20	150.10	210.20	188.70
	Error (%)	-	-	-	-	-	-	-	-	-
Discrete BA1	Mean	-	-	-	438.30	7676	558.80	21884.20	22842.90	21476.60
	SD	-	-	-	2.50	104.40	9.00	213.60	231.20	235.10
	Error (%)	-	-	-	-	-	-	-	-	-
Discrete BA2	Mean	-	-	-	436.80	7681.90	560.50	21989.40	22946.70	21631.10
	SD	-	-	-	5.30	112.30	11.60	305.20	291.30	325.00
	Error (%)	-	-	-	-	-	-	-	-	-
Discrete IBA	Mean	-	-	-	428.10	7542.00	548.10	21445.30	22506.4	21050.0
	SD	-	-	-	1.60	0.00	3.80	116.50	221.3	164.70
	Error (%)	-	-	-	-	-	-	-	-	-
DFA	Mean	-	-	-	430.80	7542	556.80	21483.60	22604.80	21096.30
	SD	-	-	-	2.30	0.00	4.90	163.70	243.90	148.30
	Error (%)	-	-	-	-	-	-	-	-	-
DICA	Mean	-	-	-	432.30	7542	557.60	21500.30	22599.70	21103.90
	SD	-	-	-	3.10	0.00	5.80	183.40	244.90	161.10
	Error (%)	-	-	-	-	-	-	-	-	-
Discrete ACO with ABC	Mean	-	-	-	443.39	7544.37	557.98	22435.31	-	-
	SD	-	-	-	5.25	0.00	4.10	231.34	-	-
	Error (%)	-	-	-	4.08	0.03	3.71	5.42	-	-
Discrete PSO-ACO-3Opt	Mean	-	-	-	426.45	7543.20	538.30	21445.10	-	-
	SD	-	-	-	0.61	2.37	0.47	78.24	-	-
	Error (%)	-	-	-	0.11	0.02	0.06	0.77	-	-
DABC	Mean	16751	-	-	-	21338	-	129960	-	-
	SD	-	-	-	-	-	-	-	-	-
	Error (%)	4.54	-	-	-	-	-	-	-	-
DPSO	Mean	17953	-	-	-	21827	-	132400	-	-
	SD		-	-	-	-	-	-	-	-
	Error (%)	5.64	-	-	-	-	-	-	-	-
Discrete DMRSA	Mean	-	-	-	426	7542	540.36	21282	-	-
	SD	-	-	-	0.19	0.00	1.16	0.00	-	-
	Error (%)	-	-	-	-	-	-	-	-	-
Basic DCS	Mean	-	-	-	439	7836	565.70	22419.96	23417.06	-
	SD	-	-	-	-	-	-	-	-	-
	Error (%)	-	-	-	3.05	3.90	5.14	5.34	5.76	-
Improved DCS	Mean	-	-	-	426	7542	538.03	21282	22141.53	-
	SD	-	-	-	0.00	0.00	0.00	0.00	0.00	-
	Error (%)	-	-	-	0.00	0.00	0.00	0.00	0.00	-
DSSA	Mean	2020	937	2085	431.87	7659	559.31	21363	22347	20997
	SD	0.00	0.00	0.00	2.80	117	0.40	81	206	248
	Error (%)	0.00	0.00	0.00	1.38	1.55	3.96	0.38	0.93	1.19
Proposed algorithm	Mean				432.6	7544.4	557.0	21,411.7	22,448.8	21,104.9
	SD				1.33	0.00	2.53	64.08	110.94	102.06
	Error (%)				1.04	0.00				
							2.82	0.16	0.89	0.87

The results in Table 10 are comparison between DAAA and other algorithms for these instances: Bays29, Fri26, Gr17, Eil51, Berlin52, Eil76, KroA100, KroB100, and KroC100. The Mean, SD, and Error rate values are presented in this table. For Eil51 DAAA had a better result for all values (Mean, SD and Error) than GA, IDGA, DiscreteBA1, DiscreteBA2, DiscreteIBA, Discrete ACO with ABC, Basic DCS. For Berlin52 DAAA is better than DiscreteBA1, DiscreteBA2, and DSSA. For Eil76 DAAA achieved better results than GA, DiscreteBA1, DiscreteBA2, BasicDCS and DSSA. For KroA100 DAAA is better than all of the listed methods except Discrete ACO with ABC, Improved DCS and DSSA. In KroB100, Improved DCS and

DSSA are better than DAAA, but DAAA is better than other methods. For KroC100 DAAA have better results than GA, IDGA, Discrete ESA, DiscreteBA1, and DiscreteBA2.

6. Conclusions

AAA is a newly proposed method for solving continuous optimization problems, and this method has achieved good results for continuous optimization problems. To the best of our knowledge, a discrete version of AAA has not been proposed. We develop a discrete version of AAA, and the name of this newly proposed discrete method is DAAA. AAA has three processes (Helical movement, Evaluation Process, and Adaptation) and these processes were used for discretization, but crossover operations (one-point and uniform) were applied to these processes. In addition to crossover operators, this study used transformation operators (swapping, insertion, symmetry, and reversion). These steps and details are explained in the paper. Additionally, Roulette wheel selection and 2-opt(but 2-opt was used only once), common tools for solving TSP, were used to improve the results.

For analyzing the success of the proposed algorithm, DAAA was tested on thirty-two TSP symmetric benchmark instances. Before starting the comparison of DAAA with other algorithms we had a process analysis, the processes (Evolutionary process, adaptation, and helical movement) with the combination of nearest neighbor and transformation operators were tested for selected Benchmark instances. This analysis starts with three combinations NEA and finishes with five combinations NEAHT. Benchmark instances included small, medium and large-scale problems. NEA obtained the best results on large-scale problems, while NEAHT performed best on medium-scale problems. After this Process Analysis, we selected which processes will be continued, and after this decision comparisons with other algorithms were started. The main comparison was between DAAA and DSSA, and our proposed method outperformed DSSA in most of the benchmark instances. To analyze the performance of DAAA it has been compared with some of the well-known methods in the literature for TSP, and DAAA has achieved better results than other methods for some problems. DAAA was superior for small and medium-scale problems, but for large-scale problems, it achieved competitive results with other algorithms. In future studies Hybrid DAAA will be developed by using different methods for discrete problems.

7. Acknowledgement

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

References

[1] Akhand M., Ayon S., Shahriyar S., Siddique N., and Adeli H., "Discrete Spider Monkey Optimization for Travelling Salesman Problem," *Applied Soft Computing*, vol. 86, pp. 105887, 2020. https://doi.org/10.1016/j.asoc.2019.105887

[2] Archetti C., Feillet D., Mor A., and Speranza M., "An Iterated Local Search for the Traveling Salesman Problem with Release Dates and Completion Time Minimization," *Computers and Operations Research*, vol. 98, pp. 24-37, 2018. https://doi.org/10.1016/j.cor.2018.05.001

[3] Babalik A., Ozkis A., Uymaz S., and Kiran M., "A Multi-Objective Artificial Algae Algorithm," *Applied Soft Computing*, vol. 68, pp. 377-395, 2018. https://doi.org/10.1016/j.asoc.2018.04.009

[4] Ban H., "Applying Metaheuristic for Time-Dependent Traveling Salesman Problem in Postdisaster," *International Journal of Computational Intelligence Systems*, vol. 14, no. 1, pp. 1087-1107, 2021. DOI:10.2991/ijcis.d.210226.001

[5] Bas E. and Ulker E., "Discrete Social Spider Algorıthm for the Travelıng Salesman Problem," *Artificial Intelligence Review*, vol. 54, no. 2, pp. 1063-1085, 2021. https://link.springer.com/article/10.1007/s10462-020-09869-8

[6] Bello R., Gomez Y., Nowe A., and Garcia M., "Two-Step Particle Swarm Optimization to Solve the Feature Selection Problem," *in Proceedings of the 7th International Conference on Intelligent Systems Design and Applications*, Rio de Janeiro, pp. 691-696, 2007. DOI: 10.1109/ISDA.2007.101

[7] Beskirli M., Koc I., Hakli H., and Kodaz H., "A New Optimization Algorithm for Solving Wind Turbine Placement Problem: Binary Artificial Algae Algorithm," *Renewable Energy*, vol. 121, pp. 301-308, 2018. https://doi.org/10.1016/j.renene.2017.12.087

[8] Cavani S., Iori M., and Roberti R., "Exact Methods for the Traveling Salesman Problem with Multiple Drones," *Transportation Research Part C: Emerging Technologies*, vol. 130, pp. 103280, 2021. https://doi.org/10.1016/j.trc.2021.103280

[9] Choong S., Wong L., and Lim C., "An Artificial Bee Colony Algorithm with a Modified Choice Function for the Traveling Salesman Problem," *Swarm and Evolutionary Computation*, vol. 44, pp. 622-635, 2019. https://doi.org/10.1016/j.swevo.2018.08.004

[10] Cinar A., Korkmaz S., and Kiran M., "A Discrete Tree-Seed Algorithm for Solving Symmetric Traveling Salesman Problem," *Engineering Science and Technology, an International Journal*, vol. 23, no. 4, pp. 879-890, 2020. https://doi.org/10.1016/j.jestch.2019.11.005

[11] Daoqing Z. and Mingyan J., "Parallel Discrete Lion Swarm Optimization Algorithm for Solving Traveling Salesman Problem," *Journal of Systems Engineering and Electronics*, vol. 31, no. 4, pp. 751-760, 2020. DOI:10.23919/JSEE.2020.000050

[12] Dell'Amico M., Montemanni R., and Novellani S., "Exact Models for the Flying Sidekick Traveling Salesman Problem," *International Transactions in Operational Research*, vol. 29, no. 3, pp. 1360-1393, 2022. https://doi.org/10.1111/itor.13030

[13] Dodig M. and Smith M., "Apple Carving Algorithm to Approximate Traveling Salesman Problem from Compact Triangulation of Planar Point Sets," *International Journal of Advanced Computer Science and Applications*, vol. 11, no. 3, pp. 1-7, 2020.

[14] Dong X., Xu M., Lin Q., Han S., Li Q., and Guo Q., "ITO Algorithm with Local Search for Large Scale Multiple Balanced Traveling Salesmen Problem," *Knowledge-Based Systems*, vol. 229, pp. 107330, 2021. https://doi.org/10.1016/j.knosys.2021.107330

[15] Dorigo M. and Gambardella L., "Ant Colonies for the Travelling Salesman Problem," *Biosystems*, vol. 43, no. 2, pp. 73-81, 1997. https://doi.org/10.1016/S0303-2647(97)01708-5

[16] Ebadinezhad S., "DEACO: Adopting Dynamic Evaporation Strategy to Enhance ACO Algorithm for the Traveling Salesman Problem," *Engineering Applications of Artificial Intelligence*, vol. 92, pp. 103649, 2020. https://doi.org/10.1016/j.engappai.2020.103649

[17] Englert M., Roglin H., and Vocking B., "Worst Case and Probabilistic Analysis of the 2-Opt Algorithm for the TSP," *Algorithmica*, vol. 68, no. 1, pp. 190-264, 2014. https://link.springer.com/article/10.1007/s00453-013-9801-4

[18] Eskandari L., Jafarian A., Rahimloo P., and Baleanu D., *Mathematical Methods in Engineering*, Springer, 2019. https://link.springer.com/chapter/10.1007/978-3-319-91065-9_13

[19] Gang P., Iimura I., and Nakayama S., "An Evolutionary Multiple Heuristic with Genetic Local Search for Solving TSP," *International Journal of Information Technology*, vol. 14, no. 2, pp. 1-11, 2008. https://intjit.org/cms/journal/volume/14/2/142_1.pdf

[20] Gharib A., Benhra J., and Chaouqi M., "A Performance Comparison of PSO and GA Applied to TSP," *International Journal of Computer Applications*, vol. 975, no. 15, pp. 34-39, 2015. DOI:10.5120/ijca2015907188

[21] Gulcu S., Mahi M., Baykan O., and Kodaz H., "A Parallel Cooperative Hybrid Method Based on Ant Colony Optimization and 3-Opt Algorithm for Solving Traveling Salesman Problem," *Soft Computing*, vol. 22, no. 5, pp. 1669-1685, 2018. https://link.springer.com/article/10.1007/s00500-016-2432-3

[22] Gunduz M. and Aslan M., "DJAYA: A Discrete Jaya Algorithm for Solving Traveling Salesman Problem," *Applied Soft Computing*, vol. 105, pp. 107275, 2021. https://doi.org/10.1016/j.asoc.2021.107275

[23] Gunduz M., Kiran M., and Ozceylan E., "A Hierarchic Approach Based on Swarm Intelligence to Solve the Traveling Salesman Problem," *Turkish Journal of Electrical Engineering and Computer Sciences*, vol. 23, no. 1, pp. 103-117, 2015. https://journals.tubitak.gov.tr/cgi/viewcontent.cgi?article=2946&context=elektrik

[24] Hertono G., Ubadah., and Handari B., "The Modification of Hybrid Method of Ant Colony Optimization, Particle Swarm Optimization and 3-OPT Algorithm in Traveling Salesman Problem," *in Proceedings of the International Conference on Mathematics: Pure, Applied and Computation*, Surabaya, pp. 1-8, 2018. DOI:10.1088/1742-6596/974/1/012032

[25] Hijazi M., Zeki A., and Ismail A., "Utilizing Artificial Bee Colony Algorithm as Feature Selection Method in Arabic Text Classification," *The International Arab Journal of Information Technology*, vol. 20, no. 3A, pp. 536-547, 2023. https://doi.org/10.34028/iajit/20/3A/11

[26] Hore S., Chatterjee A., and Dewanji A., "Improving Variable Neighborhood Search to Solve the Traveling Salesman Problem," *Applied Soft Computing*, vol. 68, pp. 83-91, 2018. https://doi.org/10.1016/j.asoc.2018.03.048

[27] Ismail A., "Domino Algorithm: A Novel Constructive Heuristics for Traveling Salesman Problem," *in Proceedings of the 11th International Seminar on Industrial Engineering and Management, Technology and Innovation Challenges towards Industry*, Makasar, pp. 1-10, 2019. DOI:10.1088/1757-899X/528/1/012043

[28] Izzo D., Getzner I., Hennes D., and Simoes L., "Evolving Solutions to TSP Variants for Active Space Debris Removal," *in Proceedings of the Annual Conference on Genetic and Evolutionary Computation*, Madrid, pp. 1207-1214, 2015. https://dl.acm.org/doi/10.1145/2739480.2754727

[29] Jones T., "Crossover, Macromutation, and Population-Based Search," *in Proceedings of the 6th International Conference on Genetic Algorithms*, Pittsburgh, pp. 73-80, 1995. https://dl.acm.org/doi/10.5555/645514.657932

[30] Kanna S., Sivakumar K., and Lingaraj N., "Development of Deer Hunting Linked Earthworm Optimization Algorithm for Solving Large Scale Traveling Salesman Problem,"

Knowledge-Based Systems, vol. 227, pp. 107199, 2021. https://doi.org/10.1016/j.knosys.2021.107199

[31] Karaboga D. and Gorkemli B., "A Combinatorial Artificial Bee Colony Algorithm for Traveling Salesman Problem," *in Proceedings of the International Symposium on Innovations in Intelligent Systems and Applications*, Istanbul, pp. 50-53, 2011. DOI:10.1109/INISTA.2011.5946125

[32] Khan I. and Maiti M., "A Swap Sequence Based Artificial Bee Colony Algorithm for Traveling Salesman Problem," *Swarm and Evolutionary Computation*, vol. 44, pp. 428-438, 2019. https://doi.org/10.1016/j.swevo.2018.05.006

[33] Kiran M., Iscan H., and Gunduz M., "The Analysis of Discrete Artificial Bee Colony Algorithm with Neighborhood Operator on Traveling Salesman Problem," *Neural Computing and Applications*, vol. 23, no. 1, pp. 9-21, 2013. https://link.springer.com/article/10.1007/s00521-011-0794-0

[34] Kumar M. and Dhillon J., "Hybrid Artificial Algae Algorithm for Economic Load Dispatch," *Applied Soft Computing*, vol. 71, pp. 89-109, 2018. https://doi.org/10.1016/j.asoc.2018.06.035

[35] Lipowski A. and Lipowska D., "Roulette-Wheel Selection via Stochastic Acceptance," *Physica A: Statistical Mechanics and its Applications*, vol. 391, no. 6, pp. 2193-2196, 2012. https://doi.org/10.1016/j.physa.2011.12.004

[36] Liu F. and Zeng G., "Study of Genetic Algorithm with Reinforcement Learning to Solve the TSP," *Expert Systems with Applications*, vol. 36, no. 3, pp. 6995-7001, 2009. https://doi.org/10.1016/j.eswa.2008.08.026

[37] Lopez-Ibanez M. and Blum C., "Beam-ACO for the Travelling Salesman Problem with Time Windows," *Computers and Operations Research*, vol. 37, no. 9, pp. 1570-1583, 2010. https://doi.org/10.1016/j.cor.2009.11.015

[38] Mahi M., Baykan O., and Kodaz H., "A New Hybrid Method Based on Particle Swarm Optimization, Ant Colony Optimization and 3-Opt Algorithms for Traveling Salesman Problem," *Applied Soft Computing*, vol. 30, pp. 484-490, 2015. https://doi.org/10.1016/j.asoc.2015.01.068

[39] Mavrovouniotis M. and Yang S., "Ant Colony Optimization with Immigrants Schemes for the Dynamic Travelling Salesman Problem with Traffic Factors," *Applied Soft Computing*, vol. 13, no. 10, pp. 4023-4037, 2013. https://doi.org/10.1016/j.asoc.2013.05.022

[40] Odili J., Kahar M., and Anwar S., "African Buffalo Optimization: A Swarm-Intelligence Technique," *Procedia Computer Science*, vol. 76, pp. 443-448, 2015. https://doi.org/10.1016/j.procs.2015.12.291

[41] Odili J., Kahar M., Anwar S., and Azrag M., "A Comparative Study of African Buffalo Optimization and Randomized Insertion Algorithm for Asymmetric Travelling Salesman's Problem," *in Proceedings of the 4th International Conference on Software Engineering and Computer Systems*, Kuantan, pp. 90-95, 2015. DOI: 10.1109/ICSECS.2015.7333089

[42] Osaba E., Del Ser J., Sadollah A., Bilbao M., and Camacho D., "A Discrete Water Cycle Algorithm for Solving the Symmetric and Asymmetric Traveling Salesman Problem," *Applied Soft Computing*, vol. 71, pp. 277-290, 2018. https://doi.org/10.1016/j.asoc.2018.06.047

[43] Pandiri V. and Singh A., "A Hyper-Heuristic Based Artificial Bee Colony Algorithm for *k*-Interconnected Multi-Depot Multi-Traveling Salesman Problem," *Information Sciences*, vol. 463-464, pp. 261-281, 2018. https://doi.org/10.1016/j.ins.2018.06.027

[44] Pedro O., Saldanha R., and Camargo R., "A Tabu Search Approach for the Prize Collecting Traveling Salesman Problem," *Electronic Notes in Discrete Mathematics*, vol. 41, pp. 261-268, 2013. https://doi.org/10.1016/j.endm.2013.05.101

[45] Reinelt G., "TSPLIB-A Traveling Salesman Problem Library," *ORSA Journal on Computing*, vol. 3, no. 4, pp. 376-384, 1991. https://pubsonline.informs.org/doi/10.1287/ijoc.3.4.376

[46] Rokbani N., Kromer P., Twir I., and Alimi A., "A New Hybrid Gravitational Particle Swarm Optimisation-ACO with Local Search Mechanism, PSOGSA-ACO-Ls for TSP," *International Journal of Intelligent Engineering Informatics*, vol. 7, no. 4, pp. 384-398, 2019. https://doi.org/10.1504/IJIEI.2019.101565

[47] Rokbani N., Kumar R., Abraham A., Alimi A., Long H., Priyadarshini I., and Son L., "Bi-Heuristic Ant Colony Optimization-based Approaches for Traveling Salesman Problem," *Soft Computing*, vol. 25, pp. 3775-3794, 2021. https://www.softcomputing.net/soft2021.pdf

[48] Saraei M., Analouei R., and Mansouri P., "Solving of Travelling Salesman Problem Using Firefly Algorithm with Greedy Approach," *Cumhuriyet University Faculty of Science Journal*, vol. 36, no. 6, pp. 267-273, 2015. https://arastirmax.com/sites/default/files/filefield_paths/5000142870-5000238948-1-pb.pdf

[49] Shang G., Lei Z., Fengting Z., and Chunxian Z., "Solving Traveling Salesman Problem by Ant Colony Optimization Algorithm with Association Rule," *in Proceedings of the 3rd International Conference on Natural Computation*, Haikou, pp. 693-698, 2007. DOI:10.1109/ICNC.2007.675

[50] Snyder L. and Daskin M., "A Random-Key Genetic Algorithm for the Generalized Traveling

Salesman Problem," *European Journal of Operational Research*, vol. 174, no. 1, pp. 38-53, 2006. https://doi.org/10.1016/j.ejor.2004.09.057

[51] Taillard E. and Helsgaun K., "POPMUSIC for the Travelling Salesman Problem," *European Journal of Operational Research*, vol. 272, no. 2, pp. 420-429, 2019. https://doi.org/10.1016/j.ejor.2018.06.039

[52] Ulder N., Aarts E., Bandelt H., Van Laarhoven P., and Pesch E., "Genetic Local Search Algorithms for the Traveling Salesman Problem," *in Proceedings of the International Conference on Parallel Problem Solving from Nature*, Dortmund, pp. 109-116, 1990. https://link.springer.com/chapter/10.1007/BFb0029740

[53] Uymaz S., Tezel G., and Yel E., "Artificial Algae Algorithm for Nonlinear Global Optimization," *Applied Soft Computing*, vol. 31, pp. 153-171, 2015. https://doi.org/10.1016/j.asoc.2015.03.003

[54] Uymaz S., Tezel G., and Yel E., "Artificial Algae Algorithm with Multi-Light Source for Numerical Optimization and Applications," *Biosystems*, vol. 138, pp. 25-38, 2015. https://doi.org/10.1016/j.biosystems.2015.11.004

[55] Wang J., Ersoy O., He M., and Wang F., "Multi-Offspring Genetic Algorithm and its Application to the Traveling Salesman Problem," *Applied Soft Computing*, vol. 43, pp. 415-423, 2016. https://doi.org/10.1016/j.asoc.2016.02.021

[56] Yang J., Shi X., Marchese M., and Liang Y., "An Ant Colony Optimization Method for Generalized TSP Problem," *Progress in Natural Science*, vol. 18, no. 11, pp. 1417-1422, 2008. https://doi.org/10.1016/j.pnsc.2008.03.028

[57] Zhang X., Wu C., Li J., Wang X., Yang Z., Lee J., and Jung K., "Binary Artificial Algae Algorithm for Multidimensional Knapsack Problems," *Applied Soft Computing*, vol. 43, pp. 583-595, 2016. https://doi.org/10.1016/j.asoc.2016.02.027

[58] Zhong Y., Lin J., Wang L., and Zhang H., "Discrete Comprehensive Learning Particle Swarm Optimization Algorithm with Metropolis Acceptance Criterion for Traveling Salesman Problem," *Swarm and Evolutionary Computation*, vol. 42, pp. 77-88, 2018. https://doi.org/10.1016/j.swevo.2018.02.017

[59] Zhou X., Gao D., Yang C., and Gui W., "Discrete State Transition Algorithm for Unconstrained Integer Optimization Problems," *Neurocomputing*, vol. 173, pp. 864-874, 2016. https://doi.org/10.1016/j.neucom.2015.08.041

Refik Nureddin received his B.Sc. and M.Sc. degrees in Computer Engineering from FON University in 2012 and 2015, respectively. He is currently working on his Ph.D. dissertation in the Department of Computer Engineering, Faculty of Engineering and Natural Sciences, at Konya Technical University. His research interests include Discrete and Binary Optimization Problems, Swarm Intelligence, and Evolutionary Computation Algorithms.

Ismail Koc gained his BSc. and MSc. degree of computer engineering from Selçuk University in 2010 and 2016. He received the Ph.D. degree in Computer Department from Konya Technical University in 2020. He is an associate professor at the Software Engineering Department at Konya Technical University. His areas of interests are Classification Methods, Artificial Intelligence, Machine Learning and Optimization Techniques.

Sait Ali Uymaz received his B.Sc., M.Sc., and Ph.D. degrees in Computer Engineering from Selçuk University in 2000, 2007, and 2015, respectively. He is currently an Associate Professor in the Department of Computer Engineering at Konya Technical University. His research focuses on Nature-Inspired Optimization, Machine Learning, and Data Science.