International Journal of Applied Mathematics Electronics and Computers

IJAMEC

ISSN:2147-8228 http://dergipark.gov.tr/ijamec Original Research Paper

# Meta-Heuristic Solution Approaches for Traveling Salesperson Problem

**Omar Mohammed Ahmed AHMED[1], Humar KAHRAMANLI[2]**

**Accepted : 27/08/2018 Published: 30/09/2018**

***Abstract:*** The traveling salesperson problem (TSP) is the NP-hard optimization problems which have been widely studied over the past years. TSP creates a Hamiltonian cycle where each node is visited once and only once to minimize the total traveled distance. TSPs are difficult to be solved using classical mathematical methods. Even with nowadays computers solving TSP problems with these methods takes very plenty of time. Therefore, many efficient optimization methods have been focused for academic proposes for the TSP all the times. Most of the TSP problems are now solved by meta-heuristic methods, that provides a satisfactory solutions in real-time. Meta-heuristic algorithms were inspired from behaviors of animals and insects such as ants, bees, fish schools, bird flocks and mammals.This paper focuses on three meta-heuristic methods: Whale Optimization Algorithm (WOA), Particle Swarm Optimization (PSO) algorithm and Grey Wolf Optimizer (GWO). The problem for application was selected from TSPLIB. Probably the best implemented solutions were Whale Optimization Algorithm and Grey Wolf Optimizer which can be recommended as primary algorithm to solve the TSP or to start with the meta-heuristic solution.

***Keywords:*** *Travelling salesperson problem, Meta-heuristic optimization, Whale Optimization Algorithm, Grey Wolf Optimizer, Particle Swarm Optimization.*

## 1. Introduction

The Travelling Salesperson Problem (TSP) is one of the complex and well-known NP-hard combinatorial optimization problems. It is easy to understand the TSP where it remains at the list of the one of the challenging problems of operational research. Its purpose is finding the shortest path for a salesperson who must visit N cities. Solving TSP has both of practical importance and academic interest, and it is an important topic of active research.

A great number of methods have been invented to solve TSP problems. Some of them are Genetic Algorithms (GA) [1], Simulated Annealing (SA) [2], Tabu-Search Algorithm (TSA) [3], Ant Colony Optimization (ACO) Algorithm [4], Memetic Algorithm (MA) [5], Bee Colony Optimization (BCO) Algorithm [6], Firefly Algorithm (FA) [7], Cuckoo Search Algorithm (CSA) [8]. In spite of classical algorithms such as TS and SA are not that efficient to be used for solving optimization problems, Evolutionary Algorithms (EA) such as MA and GA gives appropriate solutions for complex optimization problems.

TSP problem can be explained as follow: Give the shortest path that covers all cities along. Let's assume that R = (N; S) be a graph where N is a collection of vertices and S is a collection of edges. Let C=($C_{ij}$) be a cost (or distance) matrix related with S. In the TSP problem minimum distance loop (Hamiltonian loop or cycle) determination required, which is all the vertices are visited just one time. Assume that salesperson already knows $C_{ij}(i,j \in \{1,2,3,\dots,N\})$ which indicates the distance between the *i*th and *j*th cities. The salesperson must select the route with the minimum travel distance. Besides it this tour must include all of the cities moreover each city must appear only one time. The salesperson could begin his route from any city, while he must return to the city where he began his tour.

The need of quick find of satisfactory solutions to TSP has caused to the development of numerous methods such as meta-heuristics. Meta-heuristic algorithms have showed effective performance in solving a large set of optimization problems. They have many advantages over classical methods such as flexibility and simplicity. Meta-heuristic methods are generally easy to implement and procced. In addition, these methods are very simple and flexible, and they are able to deal with many problems, both continuous and discrete moreover mixed.

Nowadays, the techniques which is using for TSP divides into two main groups: approximation algorithm, and exact methods which guarantees obtaining the optimal solution.

Approximation algorithms have the ability to obtain more accurate, therefore they are very appropriate to solve large-scale problems. These algorithms also divide into two groups: heuristic optimization techniques and local search algorithms. Heuristic optimization techniques search around the optimal solution. GA [1], SA [2], ACO [4], PSO [30], Artificial Neural Network (ANN) [9,10,11], Marriage in Honey Bees Optimization Algorithm [12] and Artificial Immune Algorithm (AIS) [13] are examples for heuristic optimization techniques. 2 - Opt [14], 3 - Opt [15], LK [16], LKH [17] and Inver-over [18] are the examples for the local search algorithms.

The second main category for solving TSP problems is exact methods which have the ability to obtain guarantee optimal solutions, but it leads to increasing in the problem's scale, the required time for solving exponentially increases. The common exact techniques include dynamic branch and bound method [19], programming method [20].

In the past years, many researches could combine meta-heuristic algorithms with local search to develop a novel hybrid algorithm to solve TSP, such as LK and genetic operators [21], combined ant colony optimization algorithm with mutation strategy [22], combined technique of a 2-Opt and genetic algorithm [23]. These

[1] *Graduate School of Natural Sciences, Selcuk University, Konya, TÜRKİYE*

[2] *Department of Computer Engineering, Technology Faculty, Selcuk University, Konya, TÜRKİYE*

** Corresponding Author: hkahramanli@selcuk.edu.tr*

combined algorithms can get satisfactory solution in less iteration. In addition, the above mentioned heuristic, meta-heuristic algorithms and exact algorithms have been tested by number of developers on TSP successfully.

This paper examines three nature-inspired (meta-heuristic) algorithms to solve TSP. Six benchmarks problem were selected to test the algorithms, and the obtained results show that the WOA and GWO achieve better results than PSO. The rest of the article is organized as follows: Section 2 gives detailed information of the meta-heuristic algorithms; section 3 gives brief explanation about applications, in section 4 simulation and comparisons are presented. In section 5 the work is concluded.

## 2. Method

Three nature-inspired algorithms: GWO, WOA and PSO have been used in this study for comparison.

### 2.1. Grey Wolf Optimizer

GWO has been proposed by Seyedali Mirjalili, Seyed Mohammad Mirjalili and Andrew Lewis. The algorithm is inspired by grey wolves which belong to Canidae family [24]. Grey wolves mostly live in groups of size 5–12. Hunting strategy of grey wolves and the leadership hierarchy in nature is mimicked by the GWO algorithm. The grey wolves group consists of four types: alpha, beta, delta, and omega. The groups are used to organize the hierarchy. Dominance decreases from alpha to omega

The leader wolf called alpha. The alpha member decides about sleeping place, waking time and hunting. The alpha member is the dominant member since group should follow his/her decisions [25]. The betas are at the second level of the hierarchy. Beta members are alpha's assistants and help him/her to make decisions. The third level of wolves in the hierarchy is called subordinate (or delta). Their jobs are Scouting, sentinels, hunting, caretakers and elders. If the wolves are not alpha, beta, or delta then it's called omega that are at the last level of grey wolf hierarchy, their job is to be scapegoat and sometimes babysitters. The main strategies of hunting for the grey wolves are [26]:

- Tracking, chasing and approximating to the prey.
- Pursuing and encircling the prey. Harassing the prey until the moving stops.
- Attacking the prey.

Hunting strategies of grey wolves is modeled mathematically for designing GWO:

During the hunting process grey wolves surround their prey. The encircling strategy is mathematically modeled as equations (1), (2).

$$D = |A.X(l) - C.X_p(l)| \quad (1)$$
$$X(l+1) = X_p(l) - A.D \quad (2)$$

Where l is the current iteration, C and A are coefficient vectors, Xp and X indicate the preys' and grey wolfs' position vectors, respectively. Calculation of the vectors C and A are shown in (3) and (4).

$$A = 2c.r_1 - c \quad (3)$$
$$C = 2.r_2 \quad (4)$$

where c linearly decreases from 2 to 0 during the iteration process (in both exploitation and exploration phases) and r1, r2 are random vectors in [0, 1].

The hunting process is generally directed by the alpha member. Occasionally the beta agent and delta agent also take a part in the process. When the hunting strategy of grey wolves mathematically modeled, it assumed that the prey's potential location is better known by the alpha, beta and delta members. The first, second and third solutions are saved and the other members are obliged to update positons according to the best search members' position as shown in the equations (5) - (7).

$$D_\alpha = |C_1.X_\alpha - X|$$
$$D_\beta = |C_2.X_\beta - X| \quad (5)$$
$$D_\delta = |C_1.X_\delta - X|$$
$$X_1 = |X_\alpha - A_1(D_\alpha)|$$
$$X_2 = |X_\beta - A_2(D_\beta)| \quad (6)$$
$$X_3 = |X_\delta - A_3(D_\delta)|$$
$$X(l+1) = (X_1 + X_2 + X_3)/3 \quad (7)$$

The hunting process for grey wolves finishes by attacking when the prey stops change location. The A is a random number in the interval [-2c, 2c], and during the iterations also c is decreased from 2 to 0.

The operation of exploration in GWO is used according with the positon, and that leads the wolves to move of from each other for searching prey and converge to attacking the prey. The exploration process modeled mathematically by utilizing A with random numbers where A <-1 or A>1 to oblige the search members to diverge from the prey.

The GWO is described below.

Step 1: Initialize the wolf's population.
Step 2: Initialize $A, C$ and a
Step 3: Calculate agent's fitness and define the best three first agents $X_\alpha, X_\beta$ and $X_\delta$.
Step 4: Using Eq (7) update the current agents position.
Step 5: Update $A, C$ and a
Step 6: Calculate fitness for all agents and update $X_\alpha, X_\beta$ and $X_\delta$
Step 7: Apply pair-wise swap mutation
Step 8: Optimize population.
Step 9: Go to step 3 until termination criterion is met.
Step 10: Take the best solution $X_\alpha$ as a result.

### 2.2. The Whale Optimization Algorithm (WOA)

The WOA proposed by Seyedali Mirjalili [27]. WOA is considered as population based method mimicking the humpback whales social behaviors. The method is inspired by the strategy of bubble-net hunting of humpback whales when hunting their preys. The whales are the world's greatest mammals. There are seven different main types of whales and humpback is one of them. Whales are mainly considered to be predators. Whales live in groups or alone and they are able to communicate, think, learn, make judgments and even become emotional as a human but in low level of smartness. Hunting method of Humpback whales is called bubble-net feeding method [28]. They create special bubbles in a spiral shape as foraging behavior. Goldbogen et al. [29] discovered two main maneuvers related with the bubble. He called these maneuvers as 'upward-spirals' and 'double-loops'. Humpback whales first start diving down about 12 m, then swim up toward the surface with creating bubble around the prey in a spiral shape. The second maneuver includes the following three stages: coral loop, lobtail, and capture loop.

Hunting strategy of humpback whales is modeled mathematically for designing WOA and performs optimization.

Humpback whales have the ability to locate their prey and surround them. They consider the current best candidate solution is best obtained solution and near to the optimal solution. After assigning the best member, the other search members will start updating their positions towards the Best member as shown in the equation (9).

$$D = |X(l) - C.X^*(l)| \quad (9)$$
$$X(l+1) = X^*(l) - A.D \quad (10)$$

Where l is the current iteration, C and A are the coefficient vectors, X is the position vector, $X^*$ is the best solution's position vector obtained so far.

Calculation of the vectors C and A as equation (11).

$$A = 2c.r_1 - c \quad (11)$$
$$C = 2.r_2 \quad (12)$$

Where components of c are linearly decreased from 2 to 0 during the iteration process (in both exploitation and exploration phases) and r1, r2 are random vectors in [0, 1].

Bubble-net attacking method (exploitation phase)

In shrinking encircling mechanism A is a random number in the [-c, c] interval and the value of c is decreased from 2 to 0 during the course of iterations as shown in equation (11).

In Spiral updating position mechanism first the distance between the prey and whale location is calculated then the helix-shaped movement of humpback whales is calculated using the equation (13).

$$X(l+1) = D'X^*(l).e^{bl}.\cos(2\pi t) + X^*(l) \quad (13)$$

where $D' = |X(l) - X^*(l)|$ is the distance between the prey and $i^{th}$ whale (best solution), t is a random number in the [-1, 1] interval, b is a constant.

The humpback whales used the above mentioned mechanisms when they swim around the prey. We set the mathematical model of these two mechanisms; it assumed that there is a probability of 50% to choose between these two mechanisms to update the whale's position as equation (14).

$$X(l+1) = \begin{cases} X^*(l) - A.D & if\ p < 0.5 \\ D'X^*(l).e^{bl}.\cos(2\pi t) + X^*(l) & if\ p \geq 0.5 \end{cases} \quad (14)$$

where p is a random value in [0,1].

In the exploration phase the each humpback whale search randomly for prey (best solution) and change their positions in accordance with the positions of other whales. To force the agents to move far away from the reference whale, A <-1or A >1 was used in this study.

The mathematical model for the exploration phase is as equations (15), (16).

$$D = |C.X_{rand} - X| \quad (15)$$
$$X(l+1) = X_{rand} - A.D \quad (16)$$

where $X_{rand}$ is a position vector which was randomly chosen from the population.

The WOA is described below

Step 1: Initalize the population of whales.
Step 2: Calculate fitness of agents and define $X^*$ as the best agent.
Step 3: Update $a, A, C, l$ and p for each agent.
Step 4: Using equations (9), (13) and (16) update the current agent's position.
Step 5: Calculate fitness for all agents and update $X^*$
Step 6: Apply pair-wise swap mutation
Step 7: Optimize population.
Step 8: Go to step 3 until termination criterion is met.
Step 9: Take the best solution $X^*$ as a result.

### 2.3. Particle Swarm Optimization (PSO)

The PSO has been developed in 1995 by Eberhart and Kennedy [30]. PSO mimics the social behaviours of fish schooling and birds flocking. The system of PSO initializes with a population of random agents. The population is hailed as "swarm", while, the potential solutions are termed as "particles". The Particles flow in the multidimensional search space for searching of the optimal solution by updating each particles position depending on the experience of the neighboring particles and its own experience. During the flow, the current position of the ith particle is defined by a vector $X_i = (X_{i1}, X_{i2}, X_{i3}, \dots, X_{iD})$, where D indicates the dimensions of the search space. The ith particles velocity is defined as $V_i = (V_{i1}, V_{i2}, V_{i3}, \dots, V_{iD})$. The particles previous best position is saved as the personal best position and named as pbest. The best position acquired by the population so far saved as global best and named as gbest. Optimal solution in PSO is searched by updating the velocity and the position of each particle according to the equations (17), (18).

$$v_{id}^{t+1} = w * v_{id}^t + c_1 * r_{1i} * (p_{id} - x_{id}^t) + c_2 * r_{2i} * (p_{gd} - x_{id}^t) \quad (17)$$
$$x_{id}^{t+1} = x_{id}^t + v_{id}^{t+1} \quad (18)$$

where $t$ is the current iteration. $d \in D$ indicates the dth dimension in the search space. w is the inertia weight, which is applied to control the effect of the previous velocities on the current velocity. $c_1$ and $c_2$ are acceleration constants. $r_{1i}$ and $r_{2i}$ are random vectors in interval [0, 1]. $p_{id}$ and $p_{gd}$ represent the pbest and gbest in the dth dimension. Steps of PSO are described below.

Step 1: Initialization.
Step 2: Calculate the velocity according to Eq (17).
Step 3: Update position of particles according to Eq (18).
Step 4: Update $p_i$ if the new $x_i^t$ is better than $p_i$.
Step 5: Update $p_g$ if the new $x_i^t$ is better than $p_g$.
Step 6: Go to step 2 until termination criterion is met.
Step 7: Take the global best solution $p_g$ as a result.

## 3. Application

In recent years PSO technique developed successfully to be used for discrete and continuous optimization problems to find optimal solutions through local and global models. The method mentioned above is appropriate for problems of continuous value and it can't be used directly to solve problems of discrete value such as TSP. In many studies developers could redefine the basic PSO algorithm by suggesting new concepts one of them is inspired by the 'Swap operator' and 'Swap sequence', as in [31].

Therefore, solving TSP by PSO in this paper is done in different way.

Solving TSP by GWO and WOA is done by adding pair-wise swap mutation (PSM) for improving the whole position of population as shown in Fig 1.

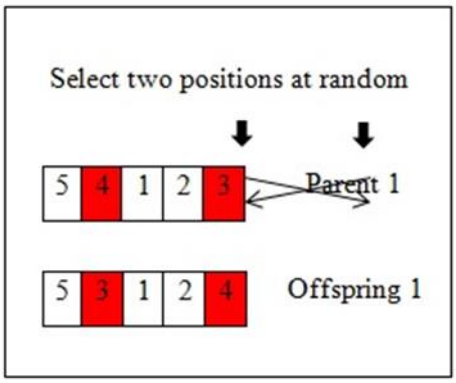


**Fig. 1.** Pair-wise swap mutation (PSM)

## 4. Simulation Experiments and Results Analysis

All three algorithms are implemented to TSP. Population size set to 200, Maximum number of iterations set to 2000 iterations. In this paper, 6 benchmark problems are used from TSPLIB [32] where number of cities varied from 30 to 100. Table 1 summarizes the experiments results and they are averaged with 30 runs of all the models for each data set. The first column of the table shows the name of the instance with the optimal solution length. In the second column the 'Best value' shows the best solution length achieved after 30 runs, the 'Worst' shows the worst solution lengths found after 30 runs, Err is the percentage of error, 'time' shows the average time used by algorithms. The percentage error of a solution is given by the equation (19).

$$\text{error} = \frac{\frac{\text{Best value}+\text{Worst value}}{2} - \text{Opt}}{\text{Opt}} \quad (19)$$

Table 1 shows the results obtained by GWO, WOA and PSO. It can be observed from the Table 1 that the results of instance with size 30 cities obtained by GWO and WOA is the optimal result while PSO was the worst and the results of instances with the size of greater than 30 cities is close to the optimal results and the error percentages of the results are smaller. For Berlin52, Eil51, Eil76, St70, and KroA100 the error percentages are less than 1 for GWO and WOA which indicate that GWO and WOA can obtain better results than PSO.

In order to simplify observation, the curve evolution diagram for Oliver30 and Berlin52 with GWO and WOA are given in Figs. 2–5. It can be observed from the figures that GWO and WOA can achieve convergence in very short iterations in Oliver30, the iteration number for convergence is less than 50, and the iteration number for convergence of Berlin52 is less than 400. Figs. 6-8 show the tour obtained by GWO and WOA of Oliver30 and Berlin52.

**Table 1.** Computational results of GWO, WOA and PSO for 6 TSP benchmark instances of TSPLIB

| *Instance* | *Calculation index* | *GWO* | *WOA* | *PSO* |
|---|---|---|---|---|
| | Best value | 423 | 423 | 801 |
| Oliver30(423) | Worst | 423 | 423 | 927 |
| | Average | 423 | 423 | 862 |
| | Err(%) | 0 | 0 | 1.9 |
| | Time (sec) | 1699 | 1141 | 9 |
| | Best value | 429 | 437 | 907 |
| Eil51(429) | Worst | 454 | 464 | 1374 |
| | Average | 442 | 448 | 1235 |
| | Err(%) | 0.02 | 0.05 | 2.7 |
| | Time (sec) | 3035 | 2731 | 18 |
| | Best value | 7680 | 7661 | 16144 |
| Berlin52(7542) | Worst | 8505 | 8323 | 23068 |
| | Average | 8030 | 7940 | 22206 |
| | Err(%) | 0.07 | 0.05 | 1.6 |
| | Time (sec) | 3051 | 2853 | 19 |
| | Best value | 684 | 679 | 2001 |
| st70(675) | Worst | 736 | 739 | 3496 |
| | Average | 718 | 713 | 2790 |
| | Err(%) | 0.05 | 0.05 | 4.5 |
| | Time (sec) | 4705 | 4746 | 40 |
| | Best value | 569 | 569 | 1662 |
| Eil76(538) | Worst | 602 | 614 | 2179 |
| | Average | 575 | 587 | 2008 |
| | Err(%) | 0.088 | 0.099 | 2.5 |
| | Time (sec) | 5446 | 5257 | 42 |
| | Best value | 21984 | 21958 | 114001 |
| KroA100( 21282) | Worst | 25475 | 25776 | 191895 |
| | Average | 23215 | 23334 | 134460 |
| | Err(%) | 0.1 | 0.1 | 6.1 |
| | Time (sec) | 8660 | 9305 | 108 |

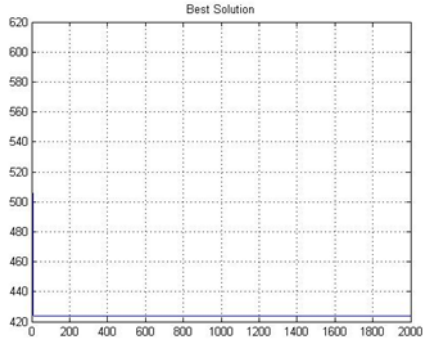


**Fig. 2.** Curve evolution diagram for Oliver30 by WOA

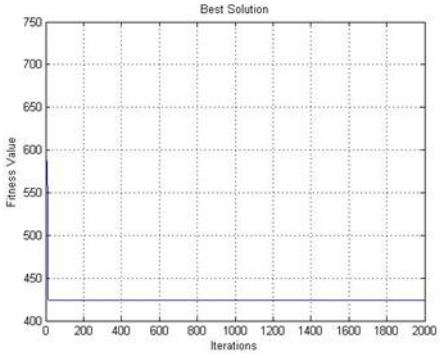


**Fig. 3.** Curve evolution diagram for Oliver30 by GWO

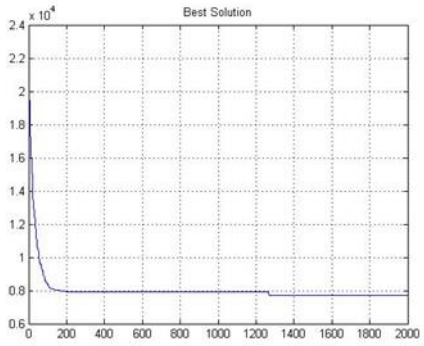


**Fig. 4.** Curve evolution diagram for Berlin52 by WOA

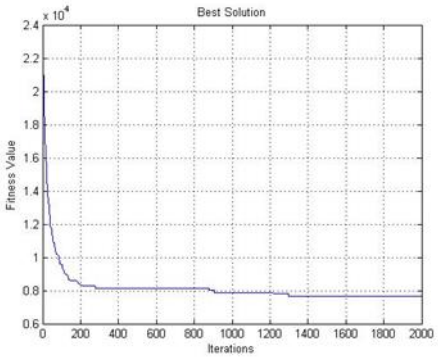


**Fig. 5.** Curve evolution diagram for Berlin52 by GWO

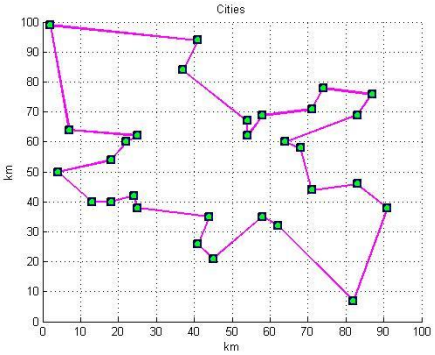


**Fig. 6.** Tour of Oliver30 obtained by WOA and GOW

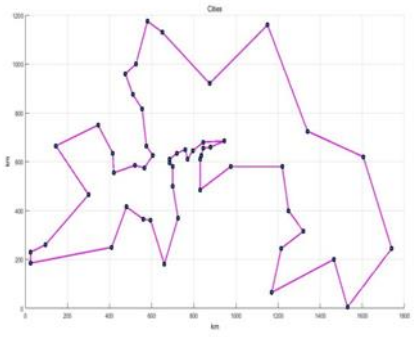

**Fig. 7**. Tour of Berlin52 obtained by WOA

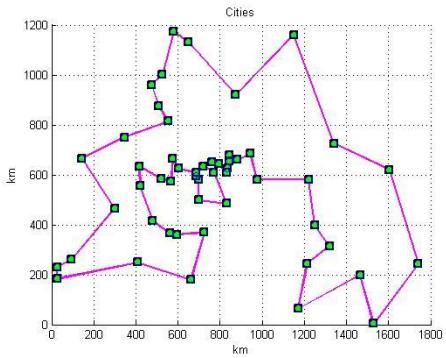


**Fig. 8.** Tour of Berlin52 obtained by GWO.

Table 2 shows comparisons of the performance of WOA, GWO and PSO to other existing algorithms: adapted harmony search algorithm, improved discrete bat algorithm, discrete penguins search optimization, discrete cat swarm optimization, hunting search algorithm, cycle crossover and order crossover with inversion mutation shuffled frog leaping algorithm and multi-population discrete firefly algorithm

**Table 2:** Results Obtained By Different Metaheuristic Algorithms

| ***Method*** | ***Instance (optimal solution)*** | | | | | |
|---|---|---|---|---|---|---|
| | ***Oliver30 (423)*** | ***Eil51 (426)*** | ***Berlin52 (7542)*** | ***st70 (675)*** | ***Eil76 (538)*** | ***KroA100 (21282)*** |
| HS [33] | - | 426 | - | 675 | 538 | 21282 |
| IBA [34] | 420 | 426 | 7542 | 675 | 539 | 21282 |
| PeSOA [35] | - | 426 | 7542 | 675 | 538 | 21282 |
| CSO [36] | - | 426 | 7542 | 675 | 538 | 21282 |
| HUS [37] | | 426 | 7542 | 675 | 538 | 21282 |
| OXIMSFLA [38] | 434 | 534 | 8362 | 892 | 733 | 37212 |
| CXIMSFLA [38] | 556 | 671 | 12266 | 1355 | 1072 | 58069 |
| MDFA [39] | - | 432 | 7681 | 682 | - | - |

## 5. Conclusion

Determining an optimal route is very important to save travel cost and time. In this study, three meta-heuristic algorithms were applied to find the best route for TSP. It was observed that the Grey Wolf Optimizer and Whale Optimization Algorithm provide feasible results for TSP and reached better performance than PSO, while WOA is bit better than GWO.

## References

J.H. Holland, “Adaptation in Natural and Artificial Systems”, The MIT Press, 1992.

D. Bookstaber, “Simulated Annealing for Traveling Salesman Problem”, 1997.

F. Glover, E. Tilliard, D.E. Werra, “A user’s guide to Tabu Search, Annals of Operations Research”, 41, 3-28,1993.

M. Doroto, M. Gambardella, “Ant colonies for the traveling salesman problem”, IRIDIA, Université Libre de Bruxelles, Belgium.

P. A. Moscato, “On Evolution, Search, Optimization, Genetic Algorithms and Martial Arts: Toward Memetic Algorithms,” Caltech, Pasadena, CA, Tech. Rep. California Technical Institute concurrent computation program report 826, 1989.

K.V. Frisch, "Decoding the language of the bee," Science, vol. 185, no. 4152, pp. 663 - 668, 1974.

X.S. Yang, X.-S., “Firefly algorithms for multimodal optimization”, International symposium on stochastic algorithms, 169-178, 2009.

X.S. Yang,, “Nature-inspired Metaheuristic Algorithm”. Luniver Press,

2nd Edition 2010.

D.S.Huang, Ji-XiangDu, “A constructive hybrid structure optimization method ology for radial basis probabilistic neural networks”, IEEET.NeuralNetw19 (2008) 2099–2115.

X. Liu, C. Xiu, “A novel hysteretic chaotic neural network and its applications”, Neurocomputing, 70 (13-15), 2561-2565, 2007.

F. Han, Qing-Hua Ling, D.S. Huang, “An improved approximation approach incorporating particle swarm optimization and a priori information into neural networks”, NeuralComput.Appl19, 255–261, 2010.

H. A. Abbass, “MBO: Marriage in honey bees optimization –a haplometrosis polygynous swarming approach”, In: Proceedings of the 2001 congress on evo- lutionary computation, p. 207–14, 2001.

J. Huant, D. Cooke, “Learning using an artificial immune system”, J.Netw. Comput. Appl, 189–212, 1996.

G.A. Croes, “A method for solving traveling salesman problem”, Oper.Res6, 791–812, 1958.

S. Lin, “Computer solutions of the traveling salesman problem”, BellSyst.Tech.J 44, 2245–2269, 1965.

S. Lin, B.W. Kernighan, “An effective heuristic algorithm for the traveling salesman problem”, Oper.Res 21 498–516, 1973.

K. Helsgaun, “An effective implementation of the lin-lemighan traveling salesman heuristic”, Eur.J.Oper.Res12, 106–130, 2000.

G. Tao, Z. Michalewicz,”Evolutionary Algorithms for the TSP, Parallel Problem Solving from Nature”, 1498, 803-812, 1998.

E.L. Lawler, D.E Wood, “Branch-and-bound methods: A survey”, Operations research, 14 (4), 699-719, 1966.

R.E. Bellman, S.E. Dreyfus, “Applied Dynamic Programming”, Princeton University Press, Princeton, New Jersey, 1962.

P. Merz, B Freisleben, “Genetic local search for the TSP: new results”, Proceedings of the 1997 IEEE International Conference on Evolutionary Computation, pp.159-163, 1997.

J.H. Yang, X.H.Shi, M.Marchese, “An ant colony optimization method for generalized TSP problem”, Prog.Nat.Sci18, 1417–1422, 2008.

F. Samanlioglu W.G.Ferrell Jr, M.E.Kurz, “A memetic random key genetic algorithm for a symmetric multi-objective traveling salesman problem”, Computers & Industrial Engineering Volume 55, Issue 2, September 2008, Pages 439-449

S. Mirjalili, S.M. Mirjalili, A. Lewis, “Grey wolf optimizer”, Adv Eng Softw 2014;69:46–61.

L.D. Mech “Alpha status, dominance, and division of labor in wolf packs”. Can J Zool,77:1196–203, 1999.

C. Muro, R. Escobedo, L. Spector, R. Coppinger, “Wolf-pack (Canis lupus) hunting strategies emerge from simple rules in computational simulations”, Behav Process, 88:192–7, 2011.

S. Mirjalili, A. Lewis, ”The whale optimization algorithm”, Advances in Engineering Software, 95, 51-67, 2016.

W. A. Watkins, W.E. Schevill,”Aerial observation of feeding behavior in four baleen whales: Eubalaena glacialis, Balaenoptera borealis, Megaptera novaean-gliae, and Balaenoptera physalus”, J Mammal, 155–63, 1979 .

J.A. Goldbogen, A.S. Friedlaender, J. Calambokidis, M.F. Mckenna, M. Simon, D.P. Nowacek, “Integrative approaches to the study of baleen whale diving behavior, feeding performance, and foraging ecology”. BioScience, 63:90–100, 2013.

J. Kennedy, R. Eberhart, “Particle swarm optimization,” in IEEE International Conference on Neural Networks, vol. 4, pp. 1942–1948, 1995.

K.P. Wang, L. Huang, C.-G. Zhou, W. Pang, ”Particle swarm optimization for traveling salesman problem,” in Machine Learning and Cybernetics, 2003 International Conference on, pp. 1583-1585, 2003.

G. Reinelt,”TSPLIB http://www.iwr.uni-heidelberg.de/groups/comopt/software,” TSPLIB95, 1995, last accessed December, 2016.

M. Bouzidi, M.E. Riffi, “Adaptation of the Harmony Search Algorithm to Solve the Travelling Salesman Problem”, Journal of Theoretical & Applied Information Technology, 62 (1).

E. Osaba, X.S. Yang, F. Diaz, P. Lopez-Garcia, R. Carballedo, “An improved discrete bat algorithm for symmetric and asymmetric traveling salesman problems”, Engineering Applications of Artificial Intelligence, 48, 59-71, 2016.

I. Mzili, M.E. RIFFI, “Discrete penguins search optimization algorithm to solve the traveling salesman problem”, Journal of Theoretical & Applied Information Technology, 72 (3), 2015

A. Bouzidi, M.E. Riffi, “Discrete cat swarm optimization to resolve the traveling salesman problem”, International Journal, 3 (9), 2013.

A. Agharghor, M.E. Riffi, “Hunting Search Algorithm to Solve the Traveling Salesman Problem”, Journal of Theoretical & Applied Information Technology, 74 (1), 2015.

S. Saud, H. Kodaz, I. Babaoğlu, “Solving Travelling Salesman Problem by Using Optimization Algorithms”, KnE Social Sciences, 3 (1), 17-32, 2018.

L. Zhou, L. Ding, X. Qiang,”A multi-population discrete firefly algorithm to solve TSP”, In: Bio-Inspired Computing-Theories and Applications, Eds: Springer, p. 648-653, 2014.