Solving A Combinatorial Optimization Problem Using Artificial Fish Swarm Algorithm

Article *in* International Journal of Engineering Trends and Technology · May 2020

DOI: 10.14445/22315381/IJETT-V68I5P206S

2 authors, including:

Nitesh Sureja
Krishna School of Diploma Studies, KPGU, Vadodara, Gujarat, INDIA.
19 PUBLICATIONS **70** CITATIONS

Solving A Combinatorial Optimization Problem Using Artificial Fish Swarm Algorithm

Dr. Nitesh M Sureja[1], Dr. Sanjay P Patel[2]

[1]Director, OM Engineering College, Junagadh, Junagadh Dist, Gujarat, India
[2]Assistant Professor, Government Engineering College, Patan, Patan Dist, Gujarat, India

Abstract

In recent times, nature inspired optimization algorithms, which are normally inspired from the food foraging, security and breeding behavior of social species have been widely used for solving a variety of combinatorial optimization problems. In this context, the social behavior of fish colonies has been explored to develop a novel algorithm, the so called Artificial Fish Swarm Algorithm (AFSA), based on the behavior of fish swarm in search for food. In this paper, the AFSA is applied to a variant of the benchmark problem known as Travelling Salesman Problem (TSP). The results obtained are then compared with other nature inspired algorithms to analyse the performance of Artificial Fish Swarm Algorithm. Comparison shows that the algorithm has better convergence performance than other algorithms.

Keywords: *Combinatorial Optimization, Nature Inspired Algorithms, Random Traveling Salesman Problem, Artificial Fish Swarm Algorithm*

I. INTRODUCTION

A combinatorial optimization problem $P = (S, f)$ can be defined by:

—a set of variables $X=\{x_1, \ldots, x_n\}$;

—variable domains $D_1, \ldots, D_n$;

—constraints among variables;

—an objective function f to be minimized, where $f: D_1 \times \cdots \times D_n \rightarrow IR^+$; The set of all possible feasible assignments is $S=\{s=\{(x_1, v_1), \ldots, (x_n, v_n)\} \mid v_i \in D_i,\ s \text{ satisfies all the constraints}\}$. S is usually called a search (or solution) space, as each element of the set can be seen as a candidate solution. To solve a combinatorial optimization problem one has to find a solution $s^* \in S$ with minimum objective function value, that is, $f(s^*) \leqslant f(s) \quad \forall\ s \in S$. s^* is called a globally optimal solution of (S, f) and the set $S^* \subseteq S$ is called the set of globally optimal solutions[1].

Combinatorial optimization problems exist widely in the fields of economic management, transportation, communication network and other fields. Its main purpose is to find optimal scheduling, grouping, order or filtering of discrete events. Many combinatorial optimization problems of both practical and theoretical importance are known to be NP-hard, such as the Knapsack problem, the Traveling Salesman Problem (TSP), and Timetabling and Scheduling problems.

Since exact algorithms are not feasible in such cases, heuristics are the main approach to tackle these problems. The significant part of heuristics comprises metaheuristic methods, which differs from the classical methods in that they combined the stochastic and deterministic composition. It means that they are focused on global optimization, not only for local extremes. The big advantage of metaheuristics is that they are built not only for solving a concrete type of problem, but they describe general algorithm in that, they show only the way, how to apply some procedures to become solution of the problem. This procedure is defined only descriptively, by black-box, and the implementation depends from the specific type of problem. The group of the most known metaheuristics includes: evolutionary algorithms[2], genetic algorithms[3], particle swarm optimization[4], differential evolution[5], ant colony optimization[6], harmony search[7], artificial immune algorithms[8], bat algorithms[9], firefly algorithms[10], cuckoo search[11], artificial buffalo optimization[12] and so on.

AFSA, which was presented by X. L. Li [13], is a new swarm intelligence optimization method developed by simulating fish swarm behavior. It is becoming a prospective method because of its good performances in solving combinatorial optimization problems such as routing optimization problem [14], complex function optimization problem [15], and so on.

In this paper, an improved AFSA is applied to a combinatorial problem known as Random Traveling Salesman Problem. The focus of the paper is on analyzing the performance of the algorithm based on time required for the convergence, cost and quality of the solution achieved.

The basic traveling salesman problem consists of a salesman and a set of cities. The salesman has to visit each one of the cities starting from a certain one and returning to the same city. The challenge of the problem is that the traveling salesman wants to

minimize the total length of the trip [16-17]. Graph theory defines the problem as finding the hamilton cycle with the least weight for a given complete weighted graph. The traveling salesman problem can be described as follows:

TSP = { *(G, f, t): G = (V, E)* a complete graph,
F is a function, $V \times V \rightarrow Z, t \in Z$,
G is a graph that contains a traveling salesman tour with cost that does not exceed t }.

Consider the example shown in figure 1 where a set of 5 cities is shown. The problem lies in finding a minimal path passing from all vertices once. For example the path 1 {A, B, C, D, E, A} and the Path 2 {A, B, C, E, D, A} pass all the vertices but Path 1 has a total length of 22 and Path 2 has a total length of 27.

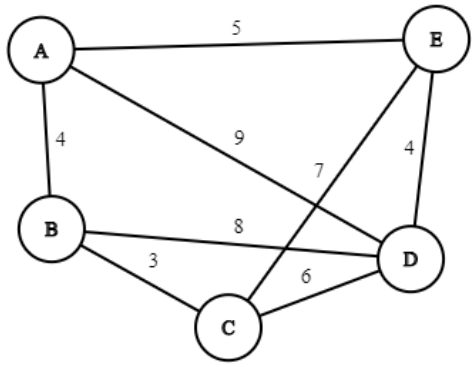

Fig 1: A graph with weights on its edges

For the practical relevance, it is necessary to solve the combinatorial optimization problems with the help of heuristics. Heuristic methods vary from exacts methods in that they give no guarantee to find the optimal solution to the given problem, but in many cases this is the solution of good quality and we can obtain it in acceptable time. Heuristic methods are usually focused on solving the special type of problems.

The rest of the paper is organized as follows. Section II introduces the Random travelling salesman problem. Section III Introduces AFSA. Section IV presents proposed AFSA-RTSP model. Section V discusses implementation and results followed by conclusion in section VI.

II. RANDOM TRAVELING SALESMAN PROBLEM

In a Random Traveling Salesman Problem, city problems are generated randomly. This is done to explore more search space to address the problem of local optima. A random problem instance generator is used to generate the city problems randomly [18-19].

Other than random TSP, there are many types of traveling salesman problem (TSP) described in the literature. To list a few; Symmetric TSP, Asymmetric TSP, Dynamic TSP, Spherical TSP etc. Symmetric TSP is a tsp where distance between the cities is same from the either side. Asymmetric TSP is a TSP where distance between the cities from the either side is not same. Dynamic TSP is a TSP where the problem changes itself at run time. In a spherical TSP all cities lie on a sphere. This paper presents an artificial fish swarm algorithm to solve Random Traveling Salesman Problem. All city problems are generated in the range of 10 to 100.

III. ARTIFICIAL FISH SWARM ALGORITHM

Artificial fish (AF) is a fictitious entity of true fish. Generally, fish moves to a position with better food consistency by performing social search behaviours [20]. AF has approximately four social behaviours: prey behavior, follow behavior, swarm behavior, and leap behavior [21]. We use these behaviours to conduct the analysis and explanation of the problem.

Suppose the state of individual artificial fish is vector X. We can denote the vector $X = (x_1, x_2, \ldots, x_n)$, where $x_i (i = 1, 2, \ldots, n)$ is the optimization variable of fishes. The food concentration in the current position of the artificial fish can be expressed as $Y = f(X)$, where Y is the value of target function. Visual is the range in which AFs can search and step is the maximum length which an AF can move. The distance between two AFs X_i and X_j can be expressed by Euclidean distance as the following equation:

$$D_{i,j} = X_i - X_j \qquad (1)$$

The crowd factor δ $(0 < \delta < 1)$ is a control parameter to control AFs' crowd around a position and the best position which AFs ever found will be loaded in bulletin. In the following subsection, behaviours of AFs will be described in detail [22].

A. Pray Behaviour

In nature, prey behavior is a basic biological behavior for fish to find food. We can determine the position X_j in the visual scope of the AF_i randomly. X_i is the current position of AF_i, X_j is a random state of its visual distance, and Y is the food concentration which can be expressed as the objective function $Y = f(X)$. The position X_j can be calculated by the following equation:

$$X_j = X_i + \text{Visual} \times \text{rand}(0, 1) \qquad (2)$$

Y_j and Y_i determine the food concentration of X_i and X_j; if $Yi<Yj$, AF moves forward a step from its current position to X_j, which is done by -

$$X_i(t+1) = X_i(t) + \frac{X_j(t) - X_i(t)}{\| X_j(t) - X_i(t) \|} \times \text{Step} \times \text{rand}(0,1) \qquad (3)$$

If $Yi>Yj$, we select a state X_j randomly again and judge whether its food consistence satisfies the forward condition. When after a specified number of try, AF_i is not satisfied with the forward condition, the concerned AF performs leap behavior.

B. Swarm Behaviour

In order to keep swarm generality, AFs attempt to move towards the centre position in every time of iterations. The central position can be determined as the following equation:

$$X_c = \frac{1}{N}\sum_{1}^{N} X_i \qquad (4)$$

where Xc is the arithmetic average of all AF swarm. And N is the size of the population. Denote n_j as the number of AF swarms in the visual scope of Xc. If $n_j/N < \delta$ and $Yc>Y_i$, which means the centre position of the swarm has better food consistence and population is not a crowd, AF_i moves forward a step to the companion centre by

$$X_i(t+1) = X_i(t) + \frac{X_c - X_i(t)}{\| X_c - X_i(t) \|} \times \text{Step} \times \text{rand}(0,1) \qquad (5)$$

Otherwise, AF performs prey behavior.

C. Follow Behaviour

During the moving process of the AF, when a single fish or several ones find food, the neighbour fishes will follow and reach the position quickly. Suppose the current position of AF_i is X_i, and position X_j is the neighbour in its visual scope. Denote n_f as the number of AF swarms in the visual scope of X_c; if $Yi<Y_j$ and $n_f/N < \delta$, AFi moves forward a step to the neighbour X_j.The expression is determined as follows:

$$X_i(t+1) = X_i(t) + \frac{X_j(t) - X_i(t)}{\| X_j(t) - X_i(t) \|} \times \text{Step} \times \text{rand}(0,1) \qquad (5)$$

If there are no neighbors around X_i or all of them dissatisfied the condition, then AF_i executes the prey behavior.

D. Leap Behavior

Leap behavior is the basic behavior to seek food or companions in large ranges, which can effectively prevent local optimum. AF_i , performs the leap behavior and changes the parameter to leap out of the current position. It chooses a state in the visual and moves towards this state to avoid the local extreme values. Hence,

$$X_i\,(t+1) = X_i\,(t) + \text{Visual} \times \text{rand}\,(0, 1)$$

The artificial fish swarm algorithm is shown in figure 2 [23].

Input: *Initialization of population for the problem, visual, try number, crowd factor. Initialization of X_i for each artificial fish AF_i (i=1, 2, …, n)*
Output: *Best Solution*
Evaluate each AF_i, $F(X_i)$ (i =1, 2, …, n)
bulletin = min $F(X_i)$
while (*t <Max Generation*)
  for *each AF_i* do
    Perform Follow Behavior on $X_i(t)$ and compute $X_{i,follow}$
    Perform Swarm Behavior on $X_i(t)$ and compute $X_{i,swarm}$
    if $F(X_{i,follow}) < F(X_{i,swarm})$
      $X_i(t+1) = X_{i,follow}$
    else
      $X_i(t+1) = X_{i,swarm}$
    end if
  end for
    if $F(X_{best\_AF}) < F(bulletin)$
    bulletin$= X_{best\_AF}$
    end if
end while

Fig 2: AFSA Algorithm [23]

IV. PROPOSED AFSA-RTSP MODEL

Traveling salesman problem is to find such a tourist routes: from a city, through each city once and only once, finally returned to the departure city, and find out the shortest route. The mathematical model [24] of the TSP problem can be described as:

$$\min f(X), s.t. g(X) \geq 0,\ \ X \in D$$

In formula, $f(X)$ act as the objective function, $g(X)$ act as the constraint function, X act as decision variable, D represent assemble that set composed of a finite number of points. Usually, a combinatorial optimization problem can be used to represent the three parameters, D which say the decision variable domain, and $F = \{X|X \in D, g(X) \geq 0\}$ f represents the target function, meet the feasible solution is called the optimal solution of the problem.

To improve our AFSA for TSP problem, we introduce the following relevant definitions [24].

Definition 1 to combinatorial optimization problem, the distance between the decision variables can be represented as:

$$distance(X_1, X_2) = |X_1 - X_2| + |X_2 - X_1|$$

It refers to the number of elements which do not belong to X_1 and X_2 at the same time.

Definition 2 for combinatorial optimization problems D, F, f

$$N(X,k) = \{ X' | \text{distance}(X, X') < k, X' \in D \}$$

Called—distance field, $X' \in N(X,k)$ known as a neighbour of X

Definition 3 for combinatorial optimization problem D, F, f;

$$\text{center}(X_1, X_2, \ldots\ldots, X_m) = \bigcup_{i=1}^{m} \bigcup_{\substack{j-1 \\ j \neq i}}^{m} (X_i \cap X_j)$$

Known as the centre of the decision variables $X_1, X_2, \ldots\ldots, X_m$.

A flowchart of AFSA-RTSP model developed is shown in figure 3.

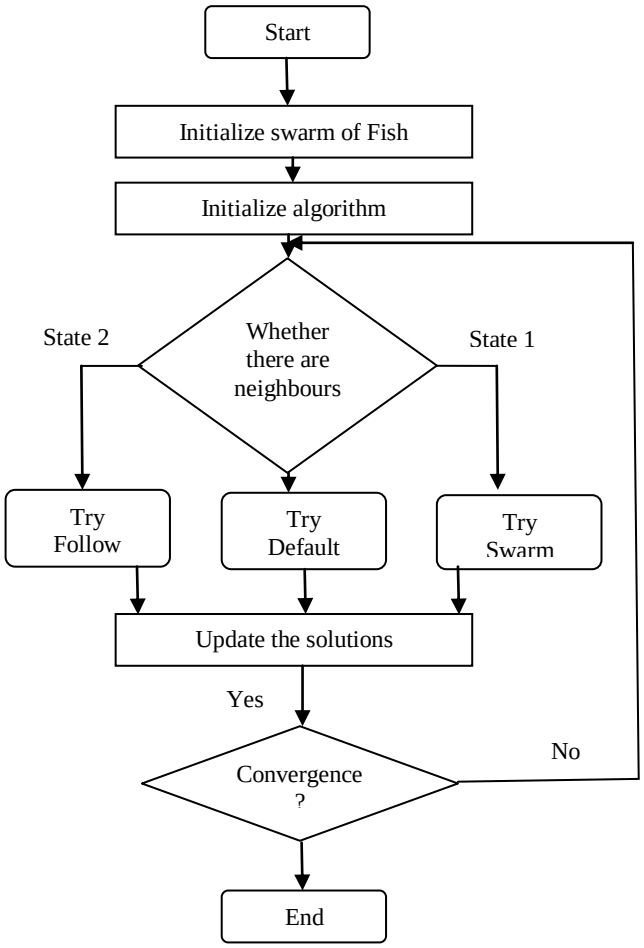

Fig 2: FASA Algorithm [23]

V. IMPLEMENTATION AND RESULTS

Proposed AFSA-RTSP model is implemented in Matlab15. Program developed is run on a dual core machine with four GB RAM. As mentioned, all tsp problems are generated randomly in the range of 10 to 100. Algorithm runs itself based on the termination criteria. Termination criteria used here is number of iterations. Table 1 shows the results obtained by the algorithm. Results are also shown graphically in the figure 4 to 13.

TABLE I. RESULTS OBTAINED FROM PROPOSED AFSA-RTSP MODEL

City Problem	Proposed AFSA-RTSP Model		
	Length	Iterations	Time
10	271.628	15	5.286396
20	361.379	32	7.712861
30	457.8638	57	12.838100
40	486.882	25	14.036483
50	570.5149	80	16.439230
60	652.7529	33	20.407480
70	674.6509	59	189.705185
80	723.445	84	213.085134
90	752.5401	248	183.455019
100	811.7461	160	299.975574

TABLE II. RESULTS OBTAINED FROM GA,MA AND ACO

City Problem	GA		MA		ACO	
	Length	Iter	Length	Iter	Length	Iter
10	300.75	116	300.75	101	319.95	100
20	375.38	215	372.34	124	373.70	102
30	485.44	266	408.57	189	422.27	102
40	527.85	395	537.95	205	524.17	107
50	587.18	677	575.36	156	578.80	109
60	631.26	675	616.31	340	639.01	115
70	693.26	1077	723.90	183	663.95	110
80	783.68	1084	768.69	216	729.85	111
90	802.38	1431	757.38	216	754.92	120
100	847.57	1683	875.87	245	809.51	119

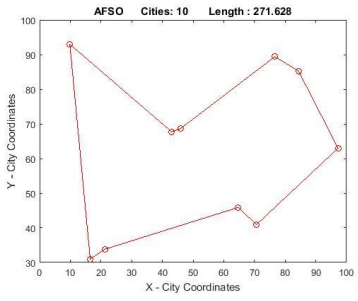

Fig 3: 10 City Problem Results

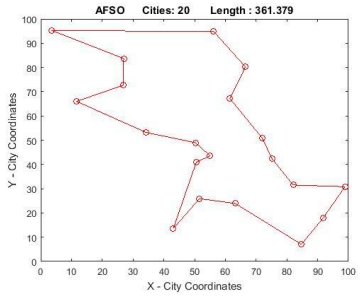

Fig 4: 20 City Problem Results

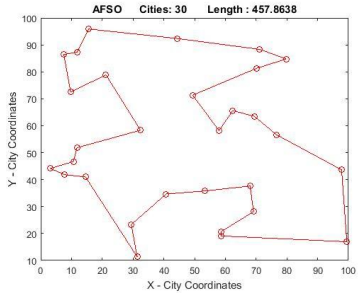

Fig 5: 30 City Problem Results

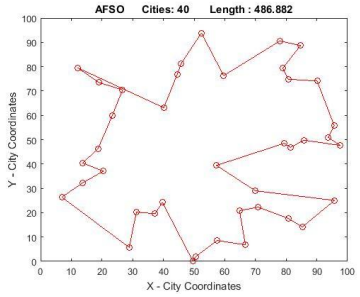

Fig 6: 40 City Problem Results

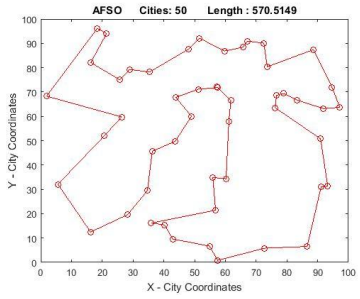

Fig 7: 50 City Problem Results

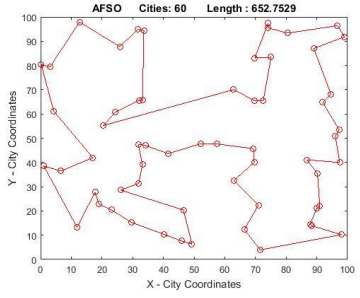

Fig 8: 60 City Problem Results

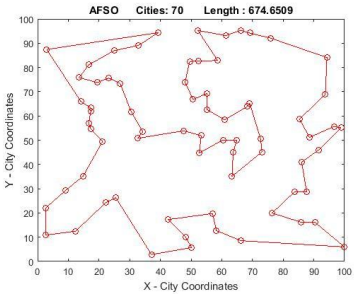

Fig 9: 70 City Problem Results

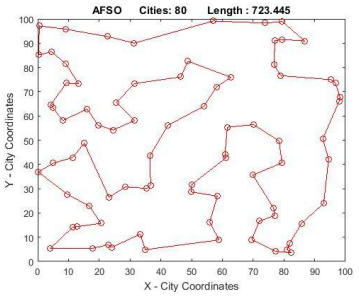

Fig 10: 80 City Problem Results

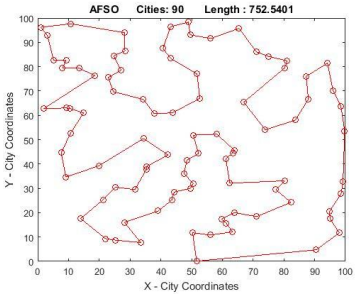

Fig 11: 90 City Problem Results

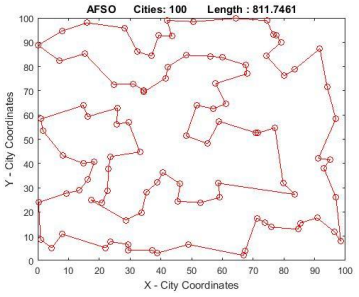

Fig 12: 100 City Problem Results

VI. CONCLUSION

A FASA model based on the food searching behavior has been introduced to solve RTSP. The model has been tested on a set of randomly generated TSP problems. The model is implemented in Matlab 15. Program is run on a Pentium dual machine with 4 GB RAM. Results obtained are compared with the results obtained from other algorithms like GA, MA and ACO. Comparisons show that AFSA outperforms all algorithms in terms of convergence

time and quality. AFSA also performs well in terms of tour costs obtained for the Random TSP problems. Future enhancements in our proposed model that will be investigated

ACKNOWLEDGMENT

The authors would like to thank OM Engineering college-Junagadh, Government Engineering college-Patan and all those who have supported directly or indirectly to carry out this research.

REFERENCES

[1] Yun Cai,"*Artificial Fish School Algorithm Applied in a Combinatorial Optimization Problem*", International Journal of Intelligent Systems and Applications (IJISA), vol.2, no.1, pp.37-43, 2010. DOI: 10.5815/ijisa.2010.01.06

[2] Tsai, H.-K., Yang, J.-M., Tsai, Y.-F., & Kao, C.-Y. (2004). *An Evolutionary Algorithm for Large Traveling Salesman Problems. IEEE Transactions on Systems, Man and Cybernetics, Part B (Cybernetics)*, 34(4), 1718–1729

[3] Abid Hussain, Yousaf Shad Muhammad, M. Nauman Sajid, Ijaz Hussain, Alaa Mohamd Shoukry, and Showkat Gani, "*Genetic Algorithm for Traveling Salesman Problem with Modified Cycle Crossover Operator*," Computational Intelligence and Neuroscience, vol. 2017, Article ID 7430125, 7 pages, 2017.

[4] G. Goldbarg, E. F., C., M., & de Souz, G. R. (2008). *Particle Swarm Optimization Algorithm for the Traveling Salesman Problem*.

[5] Mi, M., Huifeng, X., Ming, Z., & Yu, G. (2010). *An Improved Differential Evolution Algorithm for TSP Problem.* 2010 International Conference on Intelligent Computation Technology and Automation.

[6] M. Dorigo, L. Gambardella, "*Ant colonies for the Traveling salesman problem*." Biosystems 43 (1997): (73-81).

[7] X.S. Yang, "*Harmony Search as a Metaheuristic Algorithm*", Studies in Computational Intelligence, Springer Berlin, Vol. 191, pp. 1-14 (2009)

[8] M. Bakhouya, Jaafar Gaber,(2007), "*An Immune Inspired-based Optimization Algorithm: Application to the Traveling Salesman Problem*", Advanced Modeling and Optimization, Volume 9, Number 1, 105-116.

[9] Saji, Y., Riffi, M.E. *A novel discrete bat algorithm for solving the travelling salesman problem. Neural Computing & Applications*, 27, (2016) :(1853–1866)

[10] Jati, G. K., & Suyanto. (2011*). Evolutionary Discrete Firefly Algorithm for Travelling Salesman Problem.* Lecture Notes in Computer Science, 393–403.

[11] Jati GK, Manurung HM, Suyanto. "*Discrete cuckoo search for traveling salesman problem*", In Proceedings – 7th International Conference on Computing and Convergence Technology, 2012. p. 993-997.

[12] Odili, J. B., & Mohmad Kahar, M. N. (2016). *Solving the Traveling Salesman's Problem Using the African Buffalo Optimization.* Computational Intelligence and Neuroscience, 2016, 1–12.

[13] X. L. Li, Z. J. Shao, and J. X. Qian, "*An optimizing method based on autonomous animats: Fish-swarm Algorithm*," System Engineering Theory and Practice, vol. 22, pp.32-38, November 2003.

[14] X. J. Shan, M. Y. Jiang, "*The routing optimization based on improved artificial fish swarm algorithm*," Proc. of IEEE the 6th World Congress on Intelligent Control and Automation, Dalian China, pp.3658-3662, October 2006.

[15] P. Li, Modelling *and Optimization of Berth Allocation and Quay Scheduling System*. Dissertation, Tianjin, China: Tianjin university, 2007.

[16] G. Dantzig, R. Fulkerson, S. Johnson, *Solution of a Large-Scale Traveling Salesman Problem*, J. Oper. Res. Soc. 2 (1954) 393–410.

[17] Gerhard Reinelt. "*The Traveling Salesman: Computational Solutions for TSP Applications.*", Springer-Verlag, (1994),Berlin, Heidelberg

[18] Nitesh M Sureja, Bharat V Chawda, *Random Travelling Salesman Problem using Genetic Algorithms*", IFRSA's international Journal Of Computing, Volume 2, Issue 2, April 2012.

[19] Nitesh M. Sureja, Bharat V. Chawda, "*Memetic Algorithm a Metaheuristic Approach to Solve RTSP*", IJCSEITR, ISSN 2249- 6831, Vol. 3, Issue 2, pp. 183-186, June 2013

[20] R. Azizi, "*Empirical study of artificial fish swarm algorithm," Computer Science,* vol. 17, no. 6, pp. 626–641, 2014.

[21] Y. Gao, L. Guan, and T. Wang, "*Triaxial accelerometer error coefficients identification with a novel artificial fish swarm algorithm*," Journal of Sensors, vol. 2015, Article ID 509143, 17 pages, 2015.

[22] Zhang, Y., Guan, G., & Pu, X. (2016). *The Robot Path Planning Based on Improved Artificial Fish Swarm Algorithm.* Mathematical Problems in Engineering, 2016, 1–11. doi:10.1155/2016/3297585

[23] A. T. S. Alobaidi and S. A. Hussein, "*An improved Artificial Fish Swarm Algorithm to solve flexible job shop*," in New Trends in Information \& Communications Technology Applications (NTICT), 2017 Annual Conference on, 2017.

[24] Cheng, C., Li, H.-F., & Bao, C.-H. (2015). *Hybrid Artificial Fish Algorithm to Solve TSP Problem.* Proceedings of the 6th International Asia Conference on Industrial Engineering and Management Innovation, 275–285