A Hybrid Firefly Algorithm – Ant Colony Optimization for Traveling Salesman Problem

Olief Ilmandira Ratu Farisi[1], Budi Setiyono[2], R. Imbang Danandjojo[3]
[1,2]Study Program Mathematics, Faculty of Mathematics and Science, Institut Teknologi Sepuluh Nopember
Kampus ITS Keputih, Sukolilo, Surabaya, 60111, Jawa Timur
[3]Badan Penelitian dan Pengembangan Perhubungan, Kementerian Perhubungan Republik Indonesia
Kementerian Perhubungan Republik Indonesia, Jakarta
E-mail: [1]olief.ilmandira@gmail.com, [2]budi@matematika.its.ac.id, [3]dj_imbang@yahoo.co.id

Masuk: 13 Juli 2015; Direvisi: 3 Agustus 2015; Diterima: 3 Agustus 2015

Abstrak. *Pada penelitian ini dikembangkan suatu metode baru yaitu hybrid firefly algorithm-ant colony optimization (hybrid FA-ACO) untuk menyelesaikan masalah traveling salesman problem (TSP). ACO memiliki komputasi terdistribusi sehingga dapat mencegah konvergensi dini dan FA memiliki kemampuan konvergensi yang cepat dalam pencarian solusi. Untuk memperbaiki solusi dan mempercepat waktu konvergensi, digunakan metode kombinasi. Pendekatan kombinasi ini meliputi pencarian solusi dengan FA dan pencarian global dengan ACO. Solusi lokal dari FA dinormalisasi dan digunakan untuk menginisialisasi feromon untuk pencarian global ACO. Hasil dari hybrid FA-ACO dibandingkan dengan FA dan ACO. Hasil penelitian menunjukkan bahwa metode yang diusulkan dapat menemukan solusi yang lebih baik tanpa terjebak lokal optimum dengan waktu komputasi lebih pendek.*
Kata kunci: *Traveling Salesman Problem, Firefly Algorithm, Ant Colony Optimization, metode hybrid.*

Abstract. *In this paper, we develop a novel method hybrid firefly algorithm-ant colony optimization for solving traveling salesman problem. The ACO has distributed computation to avoid premature convergence and the FA has a very great ability to search solutions with a fast speed to converge. To improve the result and convergence time, we used hybrid method. The hybrid approach involves local search by the FA and global search by the ACO. Local solution of FA is normalized and is used to initialize the pheromone for the global solution search using the ACO. The outcome are compared with FA and ACO itself. The experiment showed that the proposed method can find the solution much better without trapped into local optimum with shorter computation time.*
Keywords: *Traveling Salesman Problem, Firefly Algorithm, Ant Colony Optimization, hybrid method.*

1. Introduction

The traveling salesman problem (TSP) is a problem in combinatorial optimization studied in operational research, computational mathematics, and artificial intelligence. It can be described simply: a salesman need to find the shortest tour of visiting a set of cities and return to starting city such that each city is visited exactly once. An exact solution can be obtained by finding the possibility of all existing solutions. When a large number of cities is given, it will be hard to get the exact solution, thus, the TSP is NP hard problem. There has not been a reliable method to ensure the optimal solution.

Many methods have been developed to solve TSP such as branch and bound (Land and Doig, 1960), Lin Kernighan local search (Lin & Kernighan, 1973), heuristic search (Jiang et al., 2005), and dynamic programming (Jellouli, 2001). Besides, there are meta-heuristic methods have been proposed, such that GA (Braun, 1991), PSO (Clerc, 2004), ACO (Dorigo et al., 1996), and FA (Jati & Suyanto, 2011). These minimization problems of meta-heuristic methods allow solutions to be found closer to the optimum but with high cost in time.

One of meta-heurisctic methods is Ant Colony Optimization (ACO). ACO is an algorithm that mimics the behavior of ant colonies. ACO was first developed to solve TSP. ACO has distributed computation to avoid premature convergence. But, the more number of cities, the more number of ants and iterations required. Therefore, it converges to the optimal solution slowly.

Several methods are proposed to improve the convergence time of ACO method. Hlaing and Khine (Hlaing & Khine, 2011) adopted candidate set strategy and a dynamic updating rule to improve the convergence time of conventional ACO. While Hassan, et al (Hassan, et al., 2013) proposed a method that uses the concept of ant colony system together with the parallel search of genetic algorithm to obtain the optimal solution quickly for larger TSP problems.

Another method to solve TSP is using Firefly Algorithm (FA). FA is based on the flashing behavior of fireflies. On the application of TSP, each firefly represents one permutation solution. Each firefly moves toward a brighter firefly. Since, a firefly moves with no direction, FA easily trapped into local optimum.

In this regards to improve results and convergence time for solving TSP, we developed a practical combination strategy for two evolutionary algorithms (FA-ACO) based on the ant colony algorithm (ACO) and firefly algorithm (FA). FA is used for local search because of its fast convergence time to find the local solution. While ACO is used for global search to avoid the local optimum and find the optimal solution based on the local solutions by FA. In this way, we not only avoid local optimum situation but also get the better result with faster convergence time.

2. Traveling Salesman Problem

TSP can be stated formally as follows (Lenstra & Rinnooy, 1975). Given a finite set of cities *N* and element of distance matrix $c_{ij}(i,j \in N)$ be the distance from city-*i* to city-*j*. The objective function is formulated by Equation (1), where π runs over cyclic permutation of *N* and $\pi^k(i)$ is k-th city reached by the salesman from city-*i*. If $N=\{1,2,...,n\}$ then an equivalent formulation is Equation (2), where *v* runs over all permutation of *N* and *v(k)* is k-th city of a tour.

$$\min_{\pi} \sum_{i \in N} c_{i\pi(i)} \tag{1}$$

$$\min\left(\sum_{i=1}^{n-1} c_{v(i)v(i+1)} + c_{v(n)v(1)} \right) \tag{2}$$

3. Firefly Algorithm

Firefly algorithm (FA) is a meta-heuristic optimization algorithm and nature-inspired algorithm based on the flashing behavior of fireflies. FA was first developed by Yang (Yang, 2010) for solving continuous optimization problem. FA has adapted by discretizing to solve permutation problem. The evolutionary discrete firefly algorithm (EDFA) has been developed for solving TSP by Jati and Suyanto (2011).

FA uses the following three idealized rules: (1) All firefly are unisex. (2) Attractiveness is proportional to their brightness and decreases as the distance increases. For any two flashing fireflies, the less bright one will move towards the brighter one. If there is no brighter one, then a firefly will move randomly. (3) The brightness (light intensity) of a firefly depends on the objective function.

The intensity of the light emitted by fireflies is affected by distance and the light absorption by air. Light intensity will decrease if the distance is farther. Similarly, the air will also absorb the light so that the light intensity will decrease again. These two factors that cause fireflies limited vision.

Figure 1 illustrated fireflies's movement. Each firefly represents one different solution initially. The closer to the optimum solution, the brighter the light emmited by fireflies. A firefly moves toward the brightest one. In the end, fireflies gather at the same solution.

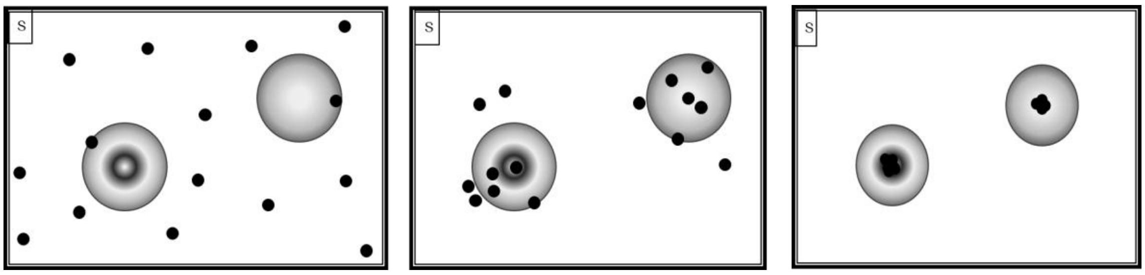

Figure 1. Illustration of fireflies' movement

3.1. Representation of firefly

A firefly represents one permutation solution of TSP as illustrated in Figure 2. In the representation, an element of an array represents a city and the index represents the order of a tour.

3	4	2	8	9	7	5	10	1	6	City
1	2	3	4	5	6	7	8	9	10	Order

Figure 2. Permutation representation of TSP solution

3.2. Light intensity

Light intensity is a value that represents the objective function of a problem. Since the objective function of TSP is to find a route with minimum distance, the light intensity is the invers of total distance produced by a firefly. A firefly which has less distance route, will have a brighter light intensity. Light intensity of a firefly *x* is calculated as Equation (3).

$$I(x) = \frac{1}{\text{totaldistance}(x)} \tag{3}$$

3.3. Distance

The distance between firefly-*i* and firefly-*j* can be defined as the number of different edges between them. The distance can be calculated by using Equation (4), where *r* is the distance between any two fireflies, *A* is the total number of different edges between two fireflies, and *N* is the number of cities. Equation (4) scales *r* in the interval [0,10] (Jati, et al., 2013). In Figure 3, four edges 2-8, 8-9, 9-7, 5-10 in firefly-*i* do not exist in firefly-*j*. By using Equation (4), we get the distance between firefly-*i* and firefly-*j* is 4.

$$r = \frac{A}{N} \times 10 \tag{4}$$

firefly-i	3	4	2	8	9	7	5	10	1	6
firefly-j	3	4	2	7	5	9	8	10	1	6

Figure 3. Permutation representation of TSP solution

3.4. Attractiveness

The attractiveness of firefly-*j* seen by firefly-*i* $\beta(i,j)$ can be any monotonic decreasing function shown as Equation (5), where $\beta(i,j)$ is the attractiveness of a firefly when seen by other firefly at distance *r*, β_0 is the brightness (light intensity) of a brighter firefly, and γ is a fixed light absorption coefficient.

$$\beta(i,j) = \beta_0 e^{-\gamma r^2} \tag{5}$$

3.5. Light Absorption

Light absorption coefficient γ determine how fast the convergence of EDFA. It also determines the characteristics of the problems. If $\gamma \rightarrow 0$, then $\beta(i,j) = \beta_0$. Thus, the attractiveness of a firefly will not decrease when viewed by another. Whereas if $\gamma \rightarrow \infty$, then the attractiveness of a firefly will be close to zero. In this case, a firefly cannot be seen. Hence, the selection of γ plays an important role because it will affect the attractiveness of fireflies. In the EDFA, the coefficient γ is in the interval [0.01, 0.15] (Jati, et al., 2013).

3.6. Movement

The movement of a firefly-*i* attracted to another more attractive firefly-*j* is determined by Equation (6), where x_i is the step that must be taken by firefly-*i* to move toward firefly-*j* and *A* is the total number of different edges between two fireflies i and *j*.

$$x_i = random(1, A) \tag{6}$$

Since a firefly in EDFA has no direction to move and represents a permutation solution, it moves using inversion mutation. Thus, it does not deform the previous permutation. This movement makes existing solutions in the firefly are changed. Each firefly will move using inversion mutation for *m* times. It means, each firefly has *m* new solutions. After *p* fireflies move and produce *pm* new solutions, then the *p* best fireflies will be selected as the new population.

Figure 4 shows the inversion mutation of a firefly. Point P_1 is start point of the inversion mutation. If a firefly moves randomly, P_1 is determined randomly. If a firefly move towards another, P_1 is determined by the first different edges of both fireflies.

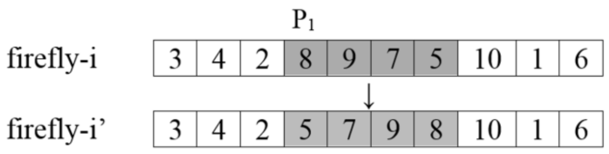

Figure 4. Inversion mutation with length movement (step) = 3

4. Ant Colony Optimization

Ant Colony Optimization (ACO) is an optimization algorithm that mimics the behaviorof ant colonies. ACO was first developed by Dorigo (Dorigo, et.al, 1996) to find the shortest path. In the ACO, a total of *m* ants cooperate and communicate using pheromones. In order to solve TSP, each artificial ant has the following characteristics: (1) Ant chooses the city that will be visited based on the probability function called transition rules. The function depends on the cities distance and the amount of trail present on the connecting edge. (2) Ant is not allowed to visit the same city before a tour is completed. This will be controlled by the tabu list. (3) When an ant finish a tour, it lays a trail of pheromones on each edge *(i, j)* which have been visited.

Ants use substances called pheromones in communication among individuals. Ants would leave pheromones on the ground thus marking the path with pheromones trail. When there are other ants running randomly, the ants will be able to detect pheromones trail and decide a way that will be passed through the magnitude of probability. Then, the ants also left a trail of pheromones that magnifies the pheromone levels on the path. The more number of ants follow the trail, the more attractive the trail to be followed. Probability ants choose the path increase with the number of ants before choosing the path. Figure 5 represents the ants' movement.

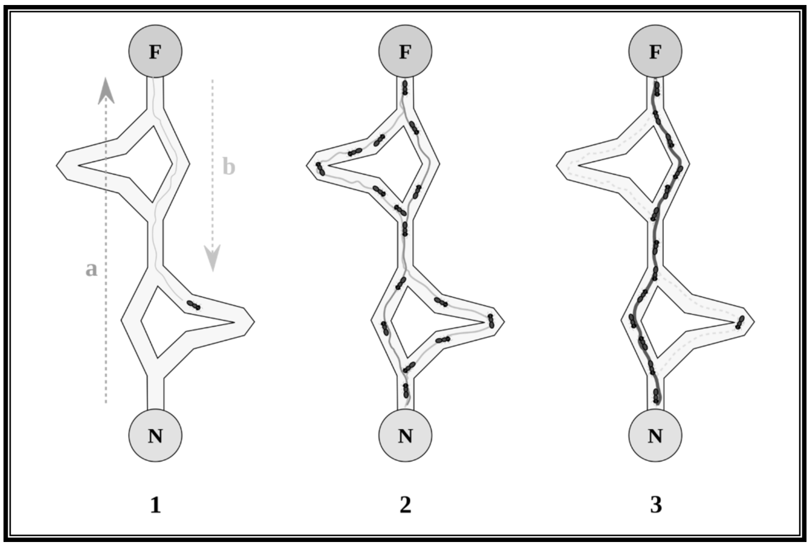

Figure 5. Illustration of ants' movement

4.1. Transition rule

Each ant is placed in a random departure city. Ants will visit one by one city using transition rule and produce a tour. Given *N* is the set of cities that N = {1, 2, …, *n*} and *m* be the total number of ants. Probability for ant-*k* from the city-*i* to city-*j* is defined as Equation (7), where τ_{ij} (t) is the intensity of trail on edge *(i,j)* at time *t*, visibility η_{ij} is the inverse distance ($\eta_{ij} = 1/d_{ij}$) and U_k is the set of cities that have not been visited by ant-*k*.

$$p_{ij} = \begin{cases} \dfrac{\left[\tau_{ij}(t)\right]^{\alpha}\left[\eta_{ij}(t)\right]^{\beta}}{\sum\limits_{u \in U_k}\left[\tau_{iu}(t)\right]^{\alpha}\left[\eta_{iu}(t)\right]^{\beta}}, \; j \in U_k \\ 0, \text{otherwise} \end{cases} \tag{7}$$

The visited cities of each ant are saved in the tabu list. When a tour is completed, the tabu list is used to compute the ant's current solution. The tabu list is then emptied and the trail intensity is updated using global pheromone update rule.

4.2. Pheromone update

The pheromone update rule includes the evaporation on all edges and the addition of pheromones on the edges that are part of the tour. The shorter tour is produced, the more pheromones left by ants on the edges. It implies the edges with a lot of pheromones would be more desirable in the next tours.

After each ant produces a tour of *n* cities, the intensity of trail will be updated with the global pheromone update rules defined as Equation (8), where ρ is the coefficient such that (1-ρ) represents the evaporation of trail between time *t* and *t+n,* and $\Delta\tau_{ij}^{k}$ is the quantity of pheromones laid by ant-*k* on the edge *(i, j)* is formulated by Equation (9), where *Q* is a constant and L_k is the tour length of ant-*k*. The shortest route found by the ants is saved and the tabu list is emptied to save the cities on the next tour. This process will be repeated until maximum condition.

$$\tau_{ij}\left(t+n\right)=\rho\tau_{ij}\left(t\right)+\sum_{k=1}^{m}\Delta\tau_{ij}^{k} \tag{8}$$

$$\Delta\tau_{ij}^{k}=\begin{cases}\dfrac{Q}{L_k},\text{if ant-}k\text{ uses }(i,j)\\ 0,\text{otherwise}\end{cases} \tag{9}$$

5. Hybrid FA-ACO

To minimize the time of convergence which is due to the high number of the agents and iterations, we proposed a hybrid method with the combination of FA and ACO with a lower number of ants and fireflies as possible. In this method, FA plays as local search and ACO used to find the global solution. FA is implemented to find a local solution because the FA has fast convergence capability.

Figure 6 shows the phase of the proposed method and explained as follows. Firstly, FA is implemented to find *p* local solutions of *pm* + *p* solutions. The local solutions is obtained by generating *p* random firefly as first population. Then, calculate the attractiveness of each firefly when seen by another firefly. Before that, calculate the light intensity and the distance between two fireflies. Each firefly finds the most attractive firefly and moves toward. A firefly will move *m* times using inversion mutation. Finally, there are *pm* + *p* solutions. Choose *p* best fireflies from *pm* + *p* solutions as a new population for next iteration.

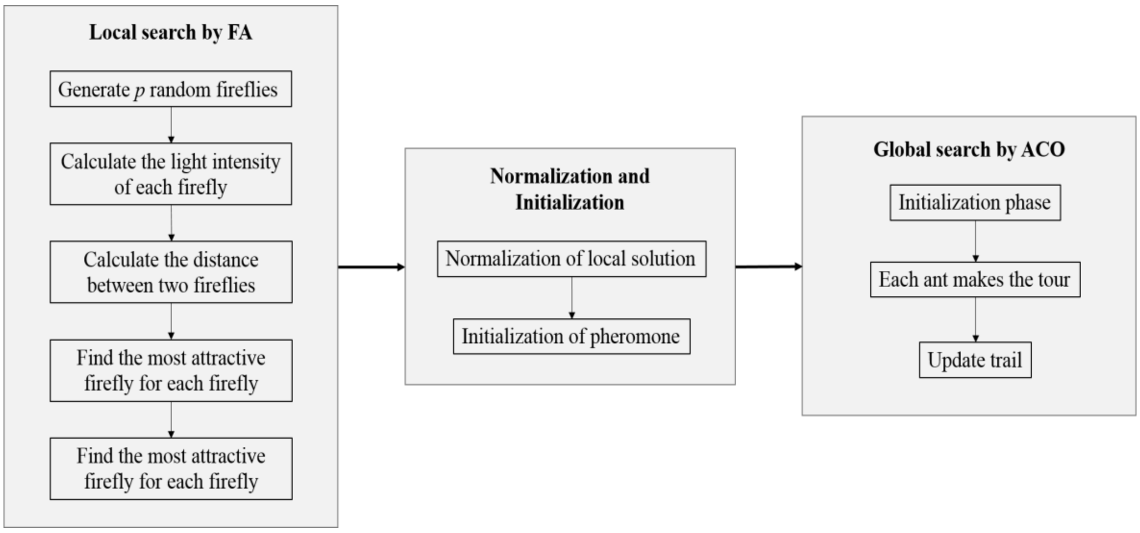
Local search by FA
Generate p random fireflies
Calculate the light intensity of each firefly
Calculate the distance between two fireflies
Find the most attractive firefly for each firefly
Find the most attractive firefly for each firefly
Normalization and Initialization
Normalization of local solution
Initialization of pheromone
Global search by ACO
Initialization phase
Each ant makes the tour
Update trail

Figure 6. Phase of proposed method

Secondly, in the end of FA's iteration, there will be *p* local solutions. From the *p* local solutions, do normalization by finding *q* different representations of firefly as candidate tours. Fireflies which have the same permutation solution will be counted as a single solution. Then, set initial trail of pheromone by adding the pheromone on the edges which are part of the *q* candidate tours. Edges that belong to best solution normalization will get the most pheromone addition. Edges on second best solution get less pheromone than the first one. Thus, the most frequently passed edges on candidate tour will get the most pheromone addition. This initial trail will be used to run ACO. By setting the initial trail, it will be minimize the agents and iterations for ACO. Third, ACO implemented to find the global solution. The output is the shortest tour. The pseudocode of the proposed method is shown as Code 1.

Code 1. Pseudocode hybrid FA-ACO for TSP

```
Input:
FA: p number of fireflies, light absorption $\gamma$, m moves.
ACO: n number of ants, parameter $\alpha$, $\beta$, $\rho$ coefficient, Q constant, $\Delta\tau_{ij}^{k} = 0$
Begin
  % Local search by FA
  Generate p random fireflies.
  Calculate light intensity of p fireflies based on the objective function.
  Repeat
    temp = p firefly;
    Calculate the distance of two fireflies using Equation (4);
    Find attractiveness varies at distance r using Equation (5);
    for i = 1 to p do (p fireflies)
      Get most attractive firefly j, where i≠j
      if there is most attractive firefly j, move firefly i toward j for m times; add
      solution to temp;
      else move firefly i random for m times; add solution to temp;
      end
    end
    Evaluate new solution and update light intensity;
    Rank the fireflies in temp and select p best firelies.
  until (stopping condition is satisfied)
  % initialize pheromone for ACO based on the best solution found by FA
  Add pheromone on each edge of p firefly.
  % Global search by ACO
  Place n ants on n city
  While (stopping condition is not satisfied)
    for k = 1 to n do
      Place the starting town of ant-k on tabu_k(1)
    end
    repeat
    for k = 1 to n do
      Choose city-j with probability in Equation (7)
      Move ant-k to the city j
      Insert city j in $tabu_k$(s)
    end
    until tabu list is full ($tabu_k$(n))
    for k = 1 to n do
      Move the ant-k from $tabu_k$ (n) to $tabu_k$(1)
      Compute the length $L_k$ of the tour described by ant-k
      Calculate pheromone addition produced by ant-k
    end
    Update the shortest tour found
    Update trail set $\Delta\tau_{ij}^{k} = 0$
    Empty all tabu list
  end
  Print the shortest tour
End
```

6. Experimental Results

The experiment is examined for four TSP instances obtained from TSPLIB. The type of TSP instances in TSPLIB is based on Euclidian distances, wherein a TSP instance provides some cities with their coordinates. The number in the name of an instance represents the number of provided cities. For example, ulysses16 provides 16 cities with their coordinates.

The experiment is implemented using Matlab and run on computer with specification Processor Intel(R) Core(TM) i7 CPU 2.20 GHz and 4.00 GB RAM. Table 1 and Table 2 present the result of FA, ACO, and hybrid FA-ACO applied to the problems. The results reported are averaged after running the experiment 10 times.

In order to evaluate the performance of the proposed method, we compare the solution obtained by the method with best solution from dataset. From Table 1, hybrid FA-ACO obtains the nearest solution to the best solution for ulysses16, oliver30, berlin52, and pr76. For ulysses16, oliver30, berlin52, hybrid FA-ACO get the exact solution. It is noticed that hybrid FA-ACO yield better solution than FA and ACO for all problems.

Table 1. Comparison of best solution known hybrid FA-ACO with FA and ACO

Problem	Best Solution	Best solution known		
		FA	ACO	Hybrid FA-ACO
ulysses16	6859	6870	6909	**6859**
oliver30	412	412	412	**412**
berlin52	7542	7718	7570	**7542**
pr76	108159	129550	117174	**110337**

Table 2. Comparison hybrid FA-ACO with FA and ACO. Results are averaged over 10 runs.

Problem	FA		ACO		Hybrid FA-ACO	
	Average solution	Average computation time	Average solution	Average computation time	Average solution	Average computation time
ulysses16	6981.4	**1.30057**	6925.5	5.8429	**6890**	3.61989
oliver30	434.1	**1.39531**	416.4	7.6767	**415**	3.98507
berlin52	8549.4	**8.4653**	7720.1	78.053	**7719.8**	23.8794
pr76	137561.3	**15.2506**	118951.9	126.709	**115339.5**	46.1076

Table 2 shows the average solution and average time converge after running the experiments 10 times. From Table 2, average solution by FA is higher than ACO. To the fact that FA is easily trapped into local optimum. Meanwhile, hybrid FA-ACO get the lowest average solution for all problems. Related to average computation time, hybrid FA-ACO runs faster than ACO, but slower than FA.

For oliver30 problem, the parameter settings are described as follows: light absorption $\gamma = 0.05$, $\alpha = 1$, $\beta = 5$, $\rho = 0.05$, and Q = 100. For running FA, we used 7 fireflies with 7 movements and 700 iterations. For runnning ACO, 30 ants and 500 iterations are required. While for hybrid FA-ACO, the local search by FA needs 4 fireflies with 4 movements and 400 iterations, the global search by ACO requires 20 ants and 300 iterations.

Figure 7 shows the comparison of convergence time among three methods for oliver30 problem and more detailed shown as Figure 8. FA has the fastest time convergence but the worst solution while hybrid FA-ACO get the best solution and faster convergence time than ACO. It means that FA has fast speed to converge but easy trapped into local optimum. These results clearly showed that hybrid FA-ACO can find the solution much better without trapped into local optimum with shorter computation time.

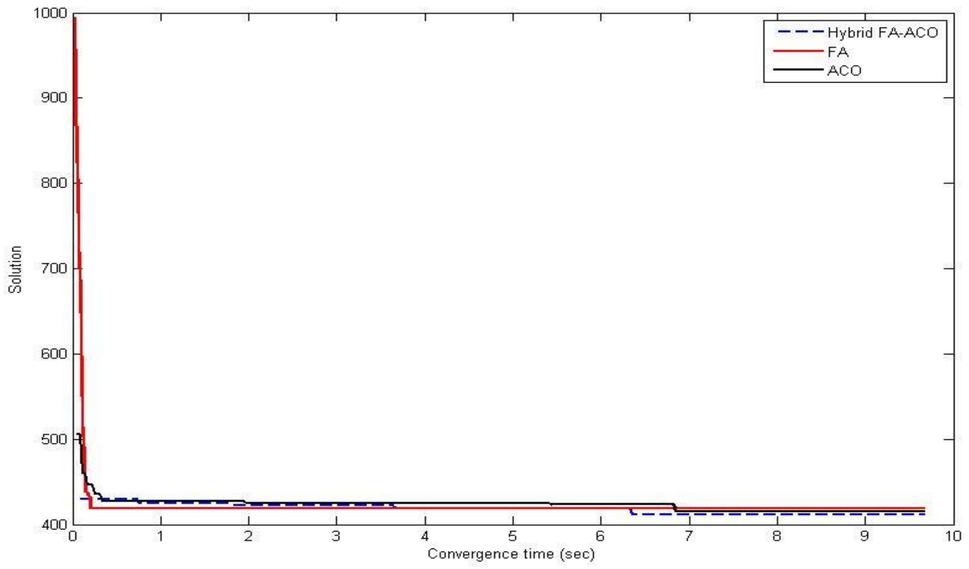

Figure 7. Comparison of convergence time for oliver30

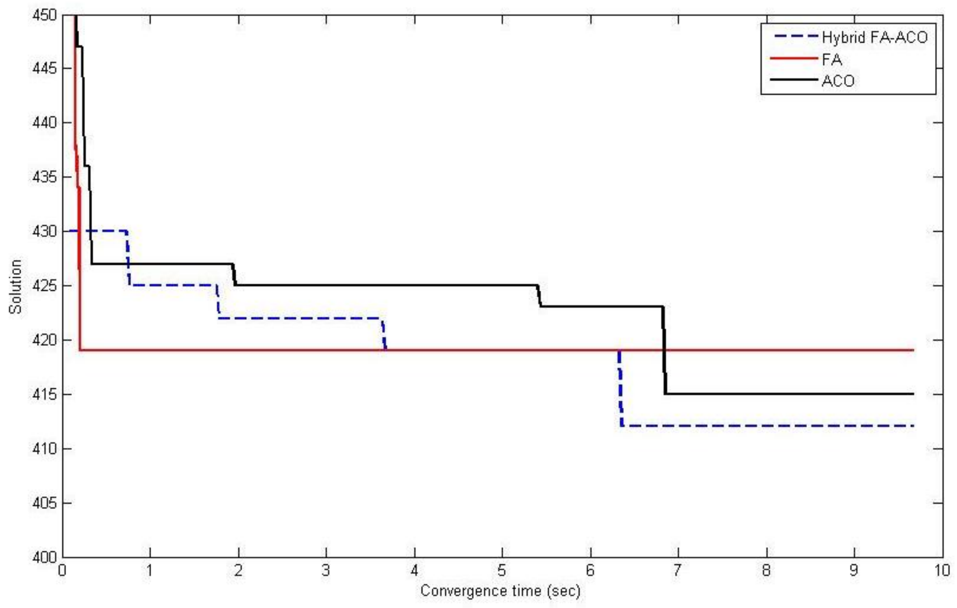

Figure 8. More detailed comparison of convergence time for oliver30

7. Conclusion

In this paper, we presented the hybrid FA-ACO method for solving the traveling salesman problem. The proposed method tries to combine FA for local search and ACO for global search. The local solution obtained by FA is used to initialize the pheromone for global search by ACO. The goal is to minimize the time of convergence with minimum number of the agents and iterations.

The experiment results showed that the hybrid FA-ACO is able to provide the better solution than FA and ACO. In our experiment, the proposed method can find the best solution without trapped into local optimum. In addition, it performs well with shorter computation time.

References

Braun, H. 1991. On Solving Travelling Salesman Problems by Genetic Algorithm. In: *1st Workshop, PPSN I Dortmund, FRG, October 1–3, 1990 Proceedings, vol. 496*, Springer-Verlag, Berlin, Heidelberg, Lecture Notes in Computer Science, 129-133.

Clerc, M. 2004. Discrete Particle Swarm Optimization, illustrated by Traveling Salesman Problem. *New Optimization Techniques in Engineering*, Springer-Verlag, Berlin, Heidelberg, Studies in Fuzziness and Soft Computing vol. 141, 219-239.

Dorigo, M., Maniezzo, V, & Colorni, A. 1996. The Ant System: Optimization by a Colony of Cooperating Agents. *IEEE Transactions on System, Man, and Cybernetics-Part B: Cybernetics*, 26(1): 29-41.

Hassan, M.R., Hasan, M.K., & Hashem, M.M.A. 2013. An Improved ACS Algorithm for the Solutions of Larger TSP Problems. In: *Proceedings of the ICEECE December 22-24, Dhaka, Bangladesh.*

Hlaing, Z.C.S.S. & Khine, M.A. December 2011. Solving Traveling Salesman Problem by Using Improved Ant Colony Optimization Algorithm. *International Journal of Information and Education Technology,* 1(5): 404-409.

Jati, G.K. & Suyanto. 2011. Evolutionary Discrete Firefly Algorithm for Travelling Salesman Problem. In: *Proceedings of Second International Conference ICAIS 2011, LNAI 6943*, Eds. Bouchachia, A., University of Klagenfurt, Klagenfurt, 393-403.

Jati, G.K., Manurung, R., & Suyanto. 2013. Discrete Firefly Algorithm for Traveling Salesman Problem: A New Movement Scheme. *Swarm Intelligence and Bio-Inspired Computation*, 295-312.

Jellouli, O., 2001. Intelligent dynamic programming for the generalized traveling salesman problem. In: *Proceedings of the IEEE International Conference on Systems, Man, and Cybernetics*, vol. 4. IEEE Computer Society, Washington, DC, 2765-2768.

Jiang, H., Zhou, Z., Zhou, P., & Chen, G.L. 2005. Union Search: A New Metaheuristic Algorithm to the Traveling Salesman Problem. J. *Univ. Sci. Technol. China*, 35: 367-375.

Land, A., & Doig, A.1960. An Automatic Method of Solving Discrete Programming Problems. *Econometrica*. 28 (3), 497-520.

Lenstra, J.K. & Rinnooy Kan, A.H.G. 1975. Some Simple Applications of Travelling Salesman Problem. *Opl Res. Q., Pergamon Press*, 26(4): 717-733.

Lin, S., & Kernighan, B. 1973. "An Effective Heuristic Algorithm for the Traveling-Salesman Problem. *Oper. Res*, 21(2): 498-516.

TSPLIB95: Ruprecht—Karls—Universitat Heildelberg, 2011. (Online), (http://www.iwr.uini-heidelberg.de/groups/comopt/software/TSPLIB95/, accessed May 10th, 2015).

Yang, X.S. 2010. *Nature-Inspired Metaheuristic Algorithm*, 2nd edition. United Kingdom: Luniver Press.