Metaheuristic Algorithms for Solving Multiple-Trips Vehicle Routing Problem with Time Windows (MTVRPTW)

A B Pratiwi¹, A Sasmito², Q S Istiqomah³, M R Kurniawan⁴, and H Suprajitno⁵
¹,²,³,⁴,⁵Department of Mathematics, Faculty of Science and Technology Universitas Airlangga, Surabaya, Indonesia

Email: ¹asri.be1kt@fst.unair.ac.id

Abstract. Multiple-Trips Vehicle Routing Problem with Time Windows (MTVRPTW) is problem of determining vehicles routes involving depot and number of customers in order to minimize the number of vehicles used and travel time. MTVRPTW has objective to minimize the total costs of transportation which are the number of tours, total tour duration time and range of duration time. This paper provides metaheuristic algorithms to solve the MTVRPTW. Firefly Algorithm, Cuckoo Search Algorithm and Flower Pollination Algorithm are applied and compared in exchange for having the performances. Based on the experimental results using different size of data shows that Firefly Algorithm performs better than Cuckoo Search Algorithm and Flower Pollination Algorithm in solving MTVRPTW.

1. Introduction
Logistics costs are the second largest costs after purchasing materials, goods and services in industries. The activities are intended to serve demands of customers. The costs incurred to serve the customers are not cheap. A company needs to budget more money for the logistics expenses. Therefore the company is required to make a specific strategy to minimize the logistics costs. One strategy that can be made is to control the distribution costs in order to fulfill demands of the customers. A set of distribution routes of vehicles must minimize costs in transportation. A problem concerning in determination of the routes by considering the capacities of the vehicles with minimum costs of transportations is called the Vehicle Routing Problem (VRP) [1].

VRPs are combinatorial problems where locations of depots and customers also the number of demands are known. The objective of the problems is determining the set of routes which minimize the total mileages. But in many applications of these problems, not only the capacities and distances that are considered for determining the vehicles routes. Each customer has service interval time which is called as time windows. Limited vehicles and vehicles operational time are also taken into consideration to determine the routes. Cases like this are an extension of VRP namely Multiple-Trips Vehicle Routing Problem with Time Windows (MTVRPTW) [2]. MTVRPTW is a variation of the classic VRP where each vehicle can be used more than once for a certain period of time [3]. In addition, MTVRPTW also has restrictions in operational time from the depot to customers. The time limits cause the vehicles can only operate at certain duration time.

Several methods had been used to solve this problem, such as branch and price algorithm [3] and an exact algorithm [4]. In this paper, three metaheuristic algorithms are applied to solve the MTVRPTW, Firefly Algorithm (FA), Cuckoo Search Algorithm (CSA) and Flower Pollination Algorithm (FPA).
This paper also provides the comparison results of the three algorithms in solving MTVRPTW. The first algorithm, Firefly Algorithm (FA), is an algorithm inspired by the behavior of fireflies in patterns of flashing. The attractiveness of the fireflies are based on brightness of their flashing. The less bright firefly will move towards the brighter one [5]. Previous research had tried to compare FA to others algorithms, Simulated Annealing (SA), Tabu Search (TS), Genetic Algorithm (GA), Memetic Algorithm, Ant Colony Optimization (ACO), Bee Colony Optimization. The results said that FA gave better performance in finding solution compared with those algorithms [6].

Cuckoo Search Algorithm (CSA) is inspired by the behavior of cuckoo species where lay their eggs in others nests [7]. There is a possibility that others bird which owned the nest will discover the egg and abandon the egg. According to previous research, CSA has better performance in optimizing times where faster than Genetic Algorithm (GA) and Particle Swarm Optimization (PSO) where CSA has advantages in random steps using Lévy Flight [8]. Last algorithm, Flower Pollination Algorithm (FPA), is inspired by flower pollination process between plants [9]. Based on the simulation results from previous research show that FPA is more efficient than GA and PSO [9] and more superior than Grasshopper Optimization Algorithm [10].

This paper is organized as follows: The next section describes formulation of the problem (MTVRPTW), the third section overviews the metaheuristic algorithms, FA, CSA and FPA. The following section is experimental result and the last section is conclusion.

2. Multiple-Trips Vehicle Routing Problem with Time Windows

The Multiple-Trips Vehicle Routing Problem with Time Windows (MTVRPTW) is different from VRP in two things. In the VRP, each vehicle is allowed to travel not more than one route while each vehicle is allowed to travel more than one route in MTVRPTW for a specified time. In addition, each customer must be served in a certain time interval called time windows [11]. The time windows denoted by \([e_i, t_i]\) is time interval where customer \(i\) must be served at that interval of time. \(e_i\) is the earliest time to be able to serve customer \(i\), while \(t_i\) is the latest time customer \(i\) can be served. If a vehicle comes to customer \(i\) before time \(e_i\), then the vehicle have to wait until time \(e_i\) [11].

In MTVRPTW, there are there objectives. The first objective is to minimize the number of vehicles which is considered as the number of tours. Because the vehicles are operated by labors then the capital cost, i.e labor costs and other operating costs, is represented as the number of tour. Then the second objective is to minimize total tour duration time. The vehicle operational cost, e.g. fuel consumption, is represented as total duration time. The third objective is to minimize the interval of duration time of the longest and shortest tour. The priority of each objective is determined by weights. Each objective is weighted by \(W_1\), \(W_2\), and \(W_3\) where \(W_1 > W_2 > W_3\) means that the number of tours is as the first priority because greater than other objectives. The objective function of MTVRPTW is given as follows [12]:

\[
\text{Min } Z = NV \cdot W_1 + TDT \cdot W_2 + RDT \cdot W_3
\]  

where

- \(NV\) : number of vehicles
- \(TDT\) : total duration time
- \(RDT\) : range duration time

Mathematical Representation

- \(NT\) : number of tours
- \(NR(t)\) : number of routes in tour \(t\)
- \(NL(t, r)\) : number of locations in route \(r\) from tour \(t\)
- \(L(t, r, k)\) : location at position \(k\) in route \(r\) from tour \(t\)
- \(a(t, r, k)\) : time of arrival at position \(k\) in route \(r\) from tour \(t\)
- \(a^0(t, r, k)\) : earliest time of arrival at position \(k\) in route \(r\) from tour \(t\)
\[ \sigma(t, r, k) : \text{time of start to service at position } k \text{ in route } r \text{ from tour } t \]

\[ \sigma^e(t, r, k) : \text{earliest time of start to service at position } k \text{ in route } r \text{ from tour } t \]

\[ \delta(t, r, k) : \text{time of departure at position } k \text{ in route } r \text{ from tour } t \]

\[ \delta^e(t, r, k) : \text{earliest time of departure at position } k \text{ in route } r \text{ from tour } t \]

\[ \delta^l(t, r, k) : \text{latest time of departure at position } k \text{ in route } r \text{ from tour } t \]

\[ w(t, r, k) : \text{time of waiting at position } k \text{ in route } r \text{ from tour } t \]

\[ TD(t, r) : \text{total of delivery load in route } r \text{ from tour } t \]

\[ d(L(t, r, k)) : \text{demand of the customer } i \text{ at position } k \text{ in route } r \text{ from tour } t \]

\[ T(i, j) : \text{travel time from customer } i \text{ to } j \]

\[ s_i : \text{service duration of customer } i \]

\[ Q : \text{maximum capacity} \]

\[ e_i : \text{earliest time window for customer } i \]

\[ l_i : \text{latest time window for customer } i \]

Constraints:

1. Each customer can only be served exactly once, every \( i \) appears once at certain \( t, r \) and \( k \), except for \( i = 0 \).

\[
L(t, r, 0) = 0 \quad \text{if } 1 \leq t \leq NT, \ r = 1, ..., NR(t)
\]

\[
L(t, r, NL(tr)) = 0 \quad \text{if } 1 \leq t \leq NT, \ r = 1, ..., NR(t)
\]

\[
L(t, r, k) = i \quad \text{if } 1 \leq t \leq NT, \ r = 1, ..., NR(t), k = 1, ..., NL(tr)
\]

2. Total delivery loads cannot exceed the maximum capacities in tours.

\[
TD(t, r, k) = \sum_{k=1}^{NL(tr)} d(L(t, r, k)) \leq Q, t = 1, ..., NT, \ r = 1, ..., NR(t)
\]

3. Earliest time of arrival and departure also starting time for service can be determined as follows:

\[
\alpha^e(t, r, k) = \delta^e(t, r, k) + T[L(t, r, k - 1), L(t, r, k)]
\]

\[
\sigma^e(t, r, k) = \max\{\alpha^e(t, r, k), e(L(t, r, k))\}
\]

\[
\delta^e(t, r, k) = \sigma^e(t, r, k) + s(L(t, r, k)) \leq l(t, r, k), t = 1, ..., NT, \ r = 1, ..., NR(t), k = 1, ..., NL(tr)
\]

4. Time windows visibility.

\[
\delta^l(t, r, NL(tr)) = l(t, r, k), t = 1, ..., NT, \ r = 1, ..., NR(t), k = 1, ..., NL(tr)
\]

5. Service time of any customers can be determined as follows:

\[
s(L(t, r, k)) = d(L(t, r, k))
\]

6. Waiting time for any customers can be described as follows:

\[
w(t, r, k) = \begin{cases} e(L(t, r, k)) - \alpha(t, r, k), & \text{if } \alpha(t, r, k) < e(L(t, r, k)) \\ 0, & \text{if } \alpha(t, r, k) \geq e(L(t, r, k)) \end{cases}
\]

7. The total tour duration time is stated as:

\[
TDT = \sum_{t=1}^{NT} DT(t)
\]

with \( DT(t) = \delta[t, NR(t), NL(tr)] - \alpha(t, 1, 0) \)

8. Range of duration time is defined as follows:

\[
RDT = \begin{cases} \text{Max}\{DT(t)\} - \text{Min}\{DT(t)\}, & \text{NT } \geq 2 \\ 0, & \text{NT } = 1 \end{cases}
\]

3. Overview of the Algorithms

Firefly Algorithm (FA), Cuckoo Search Algorithm (CSA) and Flower Pollination Algorithm (FPA) are used to solve MTVRPTW. This section overviews those metaheuristic algorithms.
3.1. Firefly Algorithm

Firefly Algorithm (FA) is an algorithm based on the behavior the fireflies to attract others using their flashing lights. There are three rules: (1) All fireflies are unisex, (2) Attractiveness of the firefly depends on the brightness of its lights, and (3) The objective function is determined by the brightness of the firefly. Each firefly will move towards other which has the brighter one. Solutions of the problems are represented as the positions of the fireflies. The first stage of FA is started with initialization parameters according to the optimization problems. Then following by the iterative stages until the stopping condition is reached [13]. Flowchart of this algorithm is given in Figure 1. The procedures can be described as follows:

1. Generate initial population of $n$ fireflies.
2. Define $\gamma$ as the light absorption coefficients.
3. Evaluate the objective function values.
4. Determine the light intensity $(I_i, i = 1, 2, \ldots, n)$
5. For each firefly and until all fireflies compared, if $I_i < I_j$ then firefly $i$ will move toward to firefly $j$ for $i, j = 1, 2, \ldots, n$. Get the distance $(d_{i,j})$, attractiveness $(\beta)$, new position $(x_i')$ and update the light intensity of firefly $i$.

$$d_{i,j} = \sqrt{\sum_{k=1}^{m} (x_i^k - x_j^k)^2} \quad (14)$$

$$\beta = I_0 e^{-\gamma d_{i,j}^2} \quad (15)$$

$$x_i' = (1 - \beta)x_i + \beta x_j + \alpha \left( rand(0,1) - \frac{1}{2} \right) \quad (16)$$

where $x_i^k$ is the position of firefly $i$ dimension $k$ ($k = 1, 2, \ldots, m$) and $\alpha$ is random step size $[0,1]$.
6. Find the current global best.
7. Check the stopping condition: maximum iteration. If the stopping condition is not satisfied, then for next iteration, global best will move randomly according to this equation below:

$$x_{best}' = x_{best} + \alpha \left( rand(0,1) - \frac{1}{2} \right) \quad (17)$$

and do step 5-7.

3.2. Cuckoo Search Algorithm

Cuckoo Search Algorithm (CSA) inspired by the behavior of cuckoo species in order to lay their eggs in other nests. There are three rules in CSA: (1) Each cuckoo lays one eggs on certain time in chosen nest randomly, (2) The best nest with good quality of egg will move to the next generation, and (3) Number of host nest is fixed, and there is a possibility that the egg will be discovered by the host bird then the host bird can abandon the nest and build the new one or get rid of the egg. In CSA, solution is represented as the position of nests or eggs. Initialization parameters according to the optimization problems will be the first stage of CSA. Then following by the iterative stage until the stopping condition is reached [14]. The procedures can described as below:

1. Generate an initial population of $n$ host nests.
2. Evaluate the objective function values $(f_i, i = 1, 2, \ldots, n)$.
3. Get a cuckoo randomly via Lévy flights according to equation below and evaluate the quality $(f_i)$.

$$x_i^{t+1} = x_i^t + L(x_{best} - x_i^t) \quad (18)$$
Figure 1. Flowchart of Firefly Algorithm.

where $x_i^t$ is position of cuckoo $i$ at iteration $t$ and $x_{best}$ is current best solution found among all solutions. The scale factor, $L$, is suggested by Mantegna [15] has been used in order to generate $L$ values:

$$L = s \cdot \kappa | \kappa \sim N(0,1), s = 0.01, \frac{u}{|v|^\beta}$$ (19)

where step size parameter $\beta \in [1,2)$, $u \sim N(0, \sigma_u^2)$, $v \sim N(0, \sigma_v^2)$, $\sigma_v = 1$, and $\sigma_u = \left[ \Gamma(1+\beta) \sin \left( \frac{\pi \beta}{2} \right) \right]^{\frac{1}{\beta}} \left[ \Gamma \left( \frac{1+\beta}{2} \right) \beta^{\frac{\beta-1}{2}} \right]$. 


4. Choose a nest $j$ randomly among $n$. If $f_i < f_j$ then replace $j$ by the new solution.
5. Define a fraction of abandon nests, $p_a \in (0,1)$.
6. Generate new solution by equation below
   \[ x_{i}^{t+1} = x_{i}^{t} + F(x_{j}^{t} - x_{k}^{t}) \]  
   where $F \sim U(0,1)$, and $x_j$ and $x_k$ are two different nests were chosen randomly.
7. Keep the best solutions
8. Find the current global best
9. Do step 5-6 until maximum iteration is reached.

3.3. Flower Pollination Algorithm
Flower pollination is the process of pollen transfer which is usually assisted by insects or other animals. There are certain flowers and insects that make a relationship commonly called a flower-pollinator partnership. These flowers will only attract insects involved in the relationship, and the insect is considered the main pollinator for the flowers. Flower pollination occurs through cross-pollination or self-pollination. In cross pollination, pollen will fall into another flower. Cross pollination and biotic occur at long distances through the help of insects. This pollination is commonly referred to as global pollination. The movement of insects will be considered as discrete movements that follow Lévy distribution. In self-pollination, pollen will fall into the same flower or similar. Self-pollination usually does not require insect assistance. In the initialization stage of FPA, each flower, part of the population of flowers, is initialized with a solution of the optimization problem then followed by iterative stage.

The procedure is described as follows [9]:
1. Generate initial population of $n$ flowers.
2. Evaluate the objective function values.
3. Define a switch probability, $p \in [0,1]$.
4. For each flower, if $rand_i < p$ then do global pollination according equation:
   \[ x_{i}^{t+1} = x_{i}^{t} + L(x_{i}^{t} - x_{best}^{t}) \]  
   otherwise do local pollination according equation below:
   \[ x_{i}^{t+1} = x_{i}^{t} + F(x_{j}^{t} - x_{k}^{t}) \]  
5. Evaluate new solutions and update.
6. Keep the best solutions.
7. Do step 4-6 until maximum iteration is reached.

4. Experimental Results
This section provides the experimental results based on three algorithms, Firefly Algorithm (FA), Cuckoo Search Algorithm (CSA) and Flower Pollination Algorithm (FPA), to solve different size of data for MTVRPTV.

4.1. Data
Data used in this paper is obtained from [http://neo.lcc.uma.es/vrp/vrp-instances/capacitated-vrp-with-time-windows-instances/](http://neo.lcc.uma.es/vrp/vrp-instances/capacitated-vrp-with-time-windows-instances/). Data consists of time windows, demands, and locations of the customers. The data taken classified into two types, namely small data (25 customers), and big data (100 customers).

4.2. Parameters
To obtain the performance comparison of each algorithm, the weights of each objective is given: $W_1 = 100000$, $W_2 = 100$ and $W_3 = 0.00005$. Parameters used in the process are presented in Table 1. Stopping condition for this process is maximum iteration. The variations of maximum iteration are 10; 250; and 500. Number of population is varied 10; 25; and 50.
### Table 1. Parameters

| Algorithm | Parameters       |
|-----------|-----------------|
| FA        | $\gamma = 0.9$, $\alpha = 0.2$ |
| CSA       | $\alpha = 0.01$, $p_a = 0.25$, $\beta = 1.5$ |
| FPA       | $p = 0.8$, $\gamma = 0.1$, $\beta = 1.5$ |

### 4.3. Results

The algorithms are applied to solve the problems using 25 customers and 100 customers. The number of maximum capacity for each vehicle in 25 customers (small data) and 100 customers (big data) are 75 and 100, respectively. The experiment results are given in Table 2 and Table 3. The values in each table are explained the best objective values reached so far from each algorithm based on the data problems given. According to Table 2, all algorithms shows the same patterns. A high number of population and iteration effect the algorithms to obtain better solution. As the number iteration goes up, the objective values tend to decrease. Based on the results of the objective function values using 100 customers in Table 3 show that FA can obtain better solution than CSA and FPA. The best solution of FA found so far is better than CSA and FPA where the number of fireflies is 50 and number of maximum iteration is 500.

#### Table 2. Performance for 25 customers

| Number of population | Number of iteration | $Z$ |
|----------------------|---------------------|-----|
|                      | FA                  | CSA | FPA |
| 10                   | 10                  | 827883 | 824343 | 947018 |
|                      | 250                 | 676735 | 659765 | 692537 |
|                      | 500                 | 706068 | 700275 | 678445 |
| 25                   | 10                  | 713043 | 837375 | 905362 |
|                      | 250                 | 669514 | 689085 | 689579 |
|                      | 500                 | 658958 | 666857 | 682068 |
| 50                   | 10                  | 648140 | 812941 | 813362 |
|                      | 250                 | 648629 | 670516 | 666426 |
|                      | 500                 | 570714 | 645439 | 620626 |

#### Table 3. Performance for 100 customers

| Number of population | Number of iteration | $Z$ |
|----------------------|---------------------|-----|
|                      | FA                  | CSA | FPA |
| 10                   | 10                  | 2771890 | 2890740 | 2989580 |
|                      | 250                 | 2764760 | 2627950 | 2760960 |
|                      | 500                 | 2623300 | 2615380 | 2717460 |
| 25                   | 10                  | 2764760 | 2778730 | 2940180 |
|                      | 250                 | 2608840 | 2609850 | 2732330 |
|                      | 500                 | 2600740 | 2580140 | 2614680 |
| 50                   | 10                  | 2655680 | 2799120 | 2903910 |
|                      | 250                 | 2600740 | 2591930 | 2723390 |
|                      | 500                 | 2541110 | 2566200 | 2569180 |
5. Conclusion
This paper proposed an overview of metaheuristic algorithms: Firefly Algorithm, Cuckoo Search Algorithm and Flower Pollination Algorithm, and apply to solve the Multiple-Trips Vehicle Routing Problem with Time Windows (MTVRPTW). Based on the overall experimental results in two types of data, 25 customers and 100 customers, indicates that Firefly Algorithm performs better than Cuckoo Search Algorithm and Flower Pollination Algorithm for solving MTVRPTW. Firefly Algorithm can find better solution as the number of iteration and population increase.

References
[1] Toth P and Vigo D 2001 The Vehicle Routing Problem (SIAM) pp 1-10
[2] Brando J and Mercer A 1998 The Multi-trip Vehicle Routing Problem Journal of the Operational Research Society 49 pp 799-805.
[3] Hernandez F, Feillet D, Giroudeau R and Naud O 2015 Branch-and-price algorithm for the solution of the multi-trip vehicle routing problem with time windows European Journal of Operational Research 000 pp 1-9.
[4] Azi N, Gendreau M and Potvin J Y 2007 An exact algorithm for a single-vehicle routing problem with time windows and multiple routes European Journal of Operational Research 178 pp 755-766
[5] Yang S X 2008 Firefly Algorithm for Multimodal Optimization Proc of the 5th International Symposium on Stochastic Algorithms : Foundations and Applications 5 pp 169-178
[6] Gupta D 2013 Solving TSP using Various meta-Heuristic Algorithms iJES I pp 22-26
[7] Yang X S and Deb S 2009 Cuckoo Search via Lévy Flights In Nature and Biologically Inspired Computing World Congress on NaBIC pp 201-214
[8] Fister D, Fister I, Fister I Jr and Yang S X 2014 Cuckoo Search : A Brief literature Review Studies in Computational Intelligence
[9] Yang S X 2012 Flower Pollination Algorithm for Global Optimization Unconventional Computation and Natural Computation pp 240-249
[10] Abdel-Basset M and Shawky L 2018 Flower pollination algorithm : a comprehensive review Artificial Intelligence Review pp 1-25
[11] Wang Z, Liang W and Hu X 2014 A metaheuristic based on a pool of routes for the vehicle routing problem with multiple trips and time windows Journal of the Operational Research Society 65 pp 37-48
[12] Ong J O and Suprayogi 2011 Vehicle Routing Problem with Backhaul, Multiple Trips and Time Windows Jurnal Teknik Industri 13 pp 1-10.
[13] Yang X S 2008 Nature-Inspired Metaheuristic Algorithms Luniver Press Frome United Kingdom
[14] Civicioglu P and Besdok E 2013 Comparative Analysis of the Cuckoo Search Algorithm In: Yang XS (eds) Cuckoo Search and Firefly Algorithm Studies in Computational Intelligence 516 pp 85-113
[15] Mantegna R N 1994 Fast, Accurate Algorithm for Numerical Simulation of Lévy Stable Stochastic Processes Phys Rev E 49 pp 4677-4683