Discrete Flower Pollination Algorithm to Solve Vehicle Routing Problem with Time Windows

Yayuk Ismi Rahayu[1], Budi Santosa[1,*], and Erwin Widodo[1]

[1] Industrial and System Engineering, Institut Teknologi Sepuluh Nopember, 6011
Kampus ITS Surabaya, Indonesia

`Corresponding author's email: budi_s@ie.its.ac.id`

Abstract. Flower Pollination Algorithm (FPA) is a nature- inspired optimization algorithm based on the pollination process of flowers. Originally FPA was created to solve continuous optimization problems. Here FPA was modified to solve discrete problems. In this paper, FPA was modified by adding Chaotic Maps function to present a different strategy in generating solutions. The modified algorithm was used to solve Vehicle Routing Problem with Time Window (VRPTW). Logistic, Iterative, Sine, Tent, and Singer Maps are the types of Chaotic Maps used in the experiment. Small, medium and large datasets were tested. Three algorithms compared here are original FPA, hybrid FPA-Chaotic Maps and Simulated Annealing. To evaluate the performance of the algorithms, we use total distance, running time and Average Relative Percentage Deviation (ARPD). Based on the experiment results, the hybrid FPA - Chaotic Maps produced the best performance, both for total distance and ARPD. The challenge of hybrid FPA - Chaotic Maps is how to improve the running time.

Keywords: Metaheuristics, Flower Pollination, Discrete Optimization, Vehicle Routing Problem, Chaotic Maps.

1 INTRODUCTION

This paper employs metaheuristic methods to address the Vehicle Routing Problem with Time Windows (VRPTW). The choice of a metaheuristic approach arises from its foundation in heuristics, which serve as alternative solutions for complex problems that are challenging to resolve using exact methods [12]. The Flower Pollination Algorithm (FPA), introduced by Xin She Yang in 2012, is a metaheuristic inspired by natural processes, particularly the pollination of flowering plants. FPA is based on four structured pollination concepts: (1) biotic and cross-pollination are considered global pollination, facilitated by pollen-carrying organisms performing Lévy Flights; (2) abiotic and sel pollination represent local pollination; (3) flower constancy relates to the reproductive chance proportional to the similarity between two flowers; and (4) the transition between local and global pollination is governed by a parameter known as the switch probability, which ranges from 0 to 1 [15].

FPA has demonstrated superior performance and faster computation times compared to Genetic Algorithms (GA), Particle Swarm Optimization (PSO), Ant Colony Optimization (ACO), and Grey Wolf Optimization (GWO), and is effective for solving lowdimensional optimization problems [1]. However, FPA also has limitations, particularly in higher-dimensional optimization scenarios, where solutions may

M. A. Muflikhun et al. (eds.), *Proceedings of the 10th International Conference on Science and Technology (ICST 2024)*, Advances in Engineering Research 268,
https://doi.org/10.2991/978-94-6463-772-4_18

converge too prematurely. This is due to the use of random real number updates ((0,1)) in successive iterations, increasing the risk of local optima entrapment and leading to higher memory usage and longer computation times. Recently, chaotic methods have emerged as a potential substitute for traditional random number generation in various applications, enhancing stochastic search capabilities [6]. Chaotic Maps offer the benefits of producing random sequences with extended periods and better consistency, resulting in reduced storage and computation time while I proving accuracy. Moreover, the application of chaotic methods in diverse optimization areas has proven effective in escaping local optima and enhancing global convergence over other stochastic optimization algorithms. This paper combines FPA and Chaotic Maps (hereinafter referred to as FPACM) to tackle VRPTW, marking a novel application of this combination in solving the problem. Performance evaluations will compare the original FPA with the FPA-CM.

VRPTW is one type of goods distribution. Distribution involves delivering products to customers through a series of repeated activities, with a shared goal of achieving swift and cost-effective delivery that benefits both the company and consumers. Given that consumers are often dispersed across various areas, transportation vehicles need well-planned routes to minimize costs and travel distances. Additionally, the transport vehicles' capacity must align with consumer demand, as transport vehicles are subject to maximum load limitations. Determining a route that considers the transport vehicle's capacity and travel distances is referred to as the Vehicle Routing Problem (VRP) [14]. The Vehicle Routing Problem (VRP) is a distribution system challenge that aims to optimize routes with the least distance for a fleet of vehicles, whose capacities are already known, in order to satisfy consumer demand at specified locations and quantities [17]. In product distribution, consumers have specific time windows during which they can accept deliveries. This creates a new challenge: the Vehicle Routing Problem with Time Windows (VRPTW), which extends the VRP by introducing time constraints.

The Vehicle Routing Problem with Time Windows (VRPTW) involves multiple vehicles with defined capacities operating within specific time frames. Here, time windows are defined as intervals [a_i, b_i], which dictate the service time required for each consumer indexed by i. A vehicle can arrive prior to a_i and wait for the consumer's approval, during which no service is provided. The service process commences once a_i begins, and vehicles arriving after b_i are not permitted (where i represents the consumer index) [13]. In this paper FPA-CM was compared to the original FPA and another metaheuristics, Simulated Annealing (SA). The remainder of this paper is structured as follows: Section 2 outlines the formulation of VRPTW, Section 3 provides a brief overview of FPA and FPACM, Section 4 presents the experiment and discussion and Section 5 conclude the paper.

2 VEHICLE ROUTING PROBLEM WITH TIME WINDOWS

Vehicle Routing Problem with Time Windows is VRP problem with additional time, where service from each customer starts within certain time interval [8]. Time windows is defined as time span [ai,bi] which is needed by a customer to obtain service. A vehicle is allowed to arrive before ai, and wait until that customer is available, however the arrival after bi is not permitted [13]. The aim of VRPTW is minimizing the total cost of travel without ignoring vehicle capacity restriction and time windows depots. The route formation is carried out so that each customer is only visited once by one vehicle, and each vehicle starts and ends its route at the depots [4]. According to [1] mathematical model for VRPTW is as follows:

$$Min\ Z = \sum_{k=1}^{K} \sum_{i=1}^{L} \sum_{j=1}^{L} c_{ij} x_{ijk} \qquad (1)$$

With some constraints:

1. Total of request from each city in a route which is passed every vehicle is not more than the capacity of vehicle.

$$\sum_{i=1}^{L} g_i y_{ik} \le q\ \forall = 1,2,\dots,K \qquad (2)$$

2. Every customer is exactly visited one time by one vehicle.

$$\sum_{k=1}^{K} \mathbf{y_{ik} = 1}, \quad \forall = \mathbf{1,2,\dots,L} \qquad \mathbf{(3)}$$

3. Make sure every route that is passed through is complete.

$$\sum_{i=1}^{L} x_{ijk} = y_{jk,} \forall k,j, \quad k = 1,2,\dots, j = 1,2,\dots,L \qquad \mathbf{(4)}$$

$$\sum_{k=1}^{L} x_{ijk} = y_{ik,} \forall k,i,k = 1,2,\dots,K, i = 1,2,\dots,L \qquad (5)$$

$$\sum_{i,j} \sum_{\in S\, x\, S} x_{ijk} \le |s| - 1, s \subset \{1,2,\dots,L\}, \forall k = 1,2,\dots,L \ (6)$$

4. Make sure that two close vehicles are lied on the same route.

$$sl_i + t_{ij} \le s_{ej}, \forall k,i,j, \qquad k = 1,2 \dots,K,$$
$$i = 1,2,\dots,L, \quad j = 1,2,\dots,L \qquad (7)$$

5. Service restrictions toward the customer *(time windows)*:

$$\mathrm{e_i} \le \mathrm{se_i} \le \mathrm{s_i}\ \forall\, \mathrm{i} = 1,2,\dots,\mathrm{L} \qquad (8)$$

$$x_{ijk}, y_{ik} \epsilon (0,1) \forall k,i,j,k = 1,2,\dots,K, i = 1,2,\dots,L, j = 1,2,\dots,L$$

With:

$$x_{ijk} = \begin{cases} 1, k's \text{ directly vehicle request from } i \ to \ j \\ 0, \quad \text{if it is not such way} \end{cases}$$

$$y_{ik} = \begin{cases} 1, \text{i's request consument is fulfilled by k vehicle} \\ 0, \quad \text{if it is not such way} \end{cases}$$

Note:

x_{ijk} : Vehicle k that takes care of customer i after visiting customer j

c_{ij} : Distance cost from customer i to customer j (can be interpreted as distance)

q : Vehicle capacity

g_i : Request from customer i

$[e_i, l_i]$: *Time windows* toward customer i

e_i : The most early time to do service to customer i

l_i : The most end time to do service to customer i

se_i : Time when service in customer's node i is started

sl_i : Time when service in customer's node i end

s_i : Total amount of time service from customer's node i

t_{ij} : Time travelling from customer's node i to customer's node j (in this case, is exampled that $t_{ij} = c_{ij}$)

i : Early city index

j : Destination city index

K : The amount of vehicle

L : The amount of customer

S : Total number of customer

3 FLOWER POLLINATION ALGORITHM (FPA)

Flower Pollination Algorithm (FPA) is a natureinspired optimization algorithm. It mimics the natural pollination process of flowering plants, where pollinators like bees, birds, or insects transfer pollen. The FPA is particularly used for solving optimization problems and is based on two key concepts: Global pollination and local pollination. Global pollination is modelled as a Lévy flight process, where pollinators transfer pollen over long distances. It is analogous to cross-pollination, which ensures global search by exploring distant regions of the search space. Local pollination mimics self-pollination, where flowers in the same plant or nearby plants exchange pollen. It represents local search to fine-tune solutions. The algorithm switches between global and local pollination using a switch probability *p* to balance exploration and exploitation. The FPA is known for being simple to implement and effective for a wide range of optimization problems. The step of local pollination is done by using the second and third approaches which can depict mathematically as below:

$$x_i^{t+1} = x_i^t + \epsilon\left(x_j^t - x_k^t\right) \quad (9)$$

With informations,

x_i^t : i − flower to t-iteration

x_i^{t+1} : i − flower to t + 1-iteration

ε : real random numbers that uniformly distributed between 0 and 1

j, k : index of two different flowers which still place in one plant

Global pollination is seen as biotic pollination and cross pollination. Biotic pollination means that the transfer of pollen to the stigma is carried out by pollinating organisms such as bees, bats, birds and flies. Whereas cross pollination occurs when pollen falls from flowers from different plants, so that in the process of global pollination, pollinating organisms can fly in long distance and do *Lèvy Flights* movement based on *Lèvy* distribution. This process can be presented mathematically as follows:

$$x_i^{t+1} = x_i^t + \gamma L(\lambda)(g_* - x_i^t) \quad (10)$$

With ,

x_i^t : i − flower to t-iteration

x_i^{t+1} : i − flower to $t + 1$-iteration

$L(\lambda)$: *stepsize* or flying distance that is based on *Lévy* distribution with $L > 0$

G^* : The best flower element in the iteration to t

With γ is a *stepsize* controller parameter at intervals with $\gamma > 0$, and λ is a parameter at interval [1,2] . The *Lèvy* distribution is represented by the following equation :

$$L \sim \frac{\lambda A(\lambda)\sin(\pi.\lambda/2)}{\pi} \frac{1}{s^{1+\lambda}} \quad (11)$$

With Γ(λ) is a gamma function, and the distribution is applied to *step length* > 0. The function of Chaotic Maps is deterministic, random-like behavior found in nonlinear dynamical systems that are sensitive to dependence on initial conditions [7]. A discrete time dynamic system for Chaotic conditions can be formulated as follows:

$$x_{k+1} = f(x_k), 0 < x_k < 1, k = 0,1,2, \quad (12)$$

where xk $\{k = 0,1,2, \ldots\}$ is a chaotic state or chaotic sequence, which can be used as a spread spectrum sequence as a sequence of random numbers. This chaotic sequence is easy to create and computationally fast requiring less memory because it only requires Chaotic Maps and initial conditions. The Chaotic Maps function has several types as shown in Table 1 [6].

Table 1. Chaotic Maps Used in Experiment

Chaotic Map	**Equation**	**Parameter**
Logistic	$x_{k+1} = ax_k(1 - x_k)$	$a = 4, x_0 = 0.3$
Iterative	$x_{k+1} = \sin(a\pi\, x_k)$	$a = 0.7, x_0 = 0.3$
Sine	$x_{k+1} = \frac{a}{4}\sin(\pi x_k)$	$a = 4, x_0 = 0.3$
Tent	$x_{k+1} = \begin{cases} \frac{x_k}{0.7}, & x_k < 0.7 \\ \frac{10}{3}(1 - x_k), & x_k \geq 0.7 \end{cases}$	$x_0 = 0.3$
Singer	$x_{k+1} = \mu(7.86x_k - 23.31x_k^2 + 28.75x_k^3 - 13.3x_k^4)$	$\mu = 1.07, x_0 = 0.3$

The procedure for FPA-Chaotic maps combination is shown in Figure 1.

```
Algorithm 1 : Chaotic Flower Pollination Algorithm
Objective min or max $f(x), x = (x_1, x_2, x_3, \dots, x_n)$
Initialize a population of n flower with random values
Sort these random values ascendingly and use their indices as routes
Evaluate the routes using total distance
Find the best solution g* in the initial population
Define a switch probability $p \in [0,1]$
while ($t < MaxGeneration$)
    for $i = 1{:}n$ (all n flowers in the population)
    if $r(Chaotic\ Maps) < p$,
        Draw a (d-dimensional) step vector L
        which obeys a L´evy distribution
        Do global pollination via
        $x_i^{t+1} = x_i^t + \gamma L(\lambda)(g_* - x_i^t)$
        Sort $x_i$ ascendingly and use its indices as routes
        Evaluate the routes using total distance
    else
        Draw ϵ from a uniform distribution
        in [0,1]
        Randomly choose j and k among all the
        solutions
        Do local pollination via
        $x_i^{t+1} = x_i^t + \varepsilon(Chaotic\ Maps)(x_j^t - x_k^t)$
        Sort $x_i$ ascendingly and use their
    indices as routes

    end if
    Evaluate new solutions
    If new solutions are better,update
        them in the population
    end for
    Find the current best solution g*
end while
```

Figure 1. Algorithm of FPA-Chaotic Maps Combination

4 EXPERIMENT RESULTS AND DISCUSSION

Experiment were done on three different algorithms: FPA, FPA-Chaotic Maps and Simulated Annealing (SA). SA was chosen as it is quite simple and in general gives good results on discrete problems. The experiment were performed using Phyton. VRPTW data were taken from https://sites.google.com/view/datavrptw. Data include information about of X-Y coordinates of nodes, Customer Demand, Service Start Time, Service End Time, and Service Length. The data consist of three types, namely small (25 Customers), medium (50 Customers), and large datasets (100 Customers).

The parameters in FPA and hybrid of FPA-Chaotic Maps are number of flowers, maximum iterations, stepsize, switch probability, value λ = 1,5 and Chaotic Maps function as shown in Table 1. The parameters used for Simulated Annealing (SA) are Initial temperature, number of iterations, Temperature Reduction Factor. The final results obtained are listed in Tables 2 & 3.

Table 2. The Objective Values For Small Dataset

Rep	**Total Distance**						
	FPA	**Logistic**	**Iterative**	**Sine**	**Tent**	**Singer**	**SA**
1	801.05	**789.43**	803.59	824.98	800.14	808.99	891.54
2	795.92	**758.77**	784.01	802.56	817.58	810.22	890.94
3	810.5	**806.69**	800.2	819.31	801.95	790.75	907.44
4	803.96	**804.78**	780.08	784.08	793.96	785.83	896.15
Avg	801.626	**789.676**	791.984	801.352	800.36	797.272	901.364

Table 3. Result of Running Time for Small Datasets

Rep	**Avg. Running Time**						
	FPA	**Logistic**	**Iterative**	**Sine**	**Tent**	**Singer**	**SA**
1	1104.11	**708.72**	779.46	786.87	802.46	798.67	**156.84**
2	1440.36	**747.71**	839.47	773.78	823.54	815.63	**119.86**
3	1037.06	**739.11**	821.73	712.45	784.65	756.78	**122.51**
4	1393.36	**813.78**	830.62	831.85	840.87	754.67	**118.17**
5	1397.42	**816.99**	800.58	799.79	836.38	803.56	**121.85**
Avg	1274.462	**765.262**	814.372	780.948	817.58	785.862	**127.85**

The final results obtained for medium datasets are shown in Tables 4 to 7 a KPI, such as PR.1.1 Earnings Before Interest and Taxes (EBIT) as a percentage of revenue. The environmental attribute has a KPI, such as EV.2.3 Renewable Energy Consumed. Social attributes have KPIs, such as SC.1.3 Training.

Table 4. Objective Values for Medium Dataset

Rep	**Avg. Running Time**						
	FPA	**Logistic**	**Iterative**	**Sine**	**Tent**	**Singer**	**SA**
1	2510.25	**2401.64**	2385.09	2449.09	2474.52	2390.35	2884.69
2	2486.26	**2369.45**	2398.3	2430.38	2380.5	2471.38	2874.89
3	2477.28	**2407.91**	2437.35	2454.65	2425.94	2423.42	2849.87
4	2508.27	**2389.96**	2474.18	2380.04	2486.48	2396.47	2830.28
5	2471.44	**2375.58**	2462.99	2396.08	2387.53	2482.07	2839.81
Avg	2490.7	**2388.908**	2431.582	2422.048	2430.994	2432.738	2884.69

Table 5. Result of Avg. Running Time for Medium Dataset

Rep	**Avg. Running Time**						
	FPA	**Logistic**	**Iterative**	**Sine**	**Tent**	**Singer**	**SA**
1	2210.58	**1465.15**	1506.02	1412.42	1438.93	1731.37	**318.56**
2	2253.21	**1337.14**	1881.71	1435.18	1401.98	1887.07	**311.93**
3	2351.67	**1464.97**	1734.56	1438.98	1483.93	1831.37	**305.66**
4	2108.76	**1470.61**	1514.89	1445.13	1496.55	1679.71	**314.84**
5	2134.89	**1512.56**	1578.43	1438.34	1441.71	1661.72	**304.31**
Avg	2211.822	**1450.086**	1643.122	1434.01	1452.62	1758.248	**311.06**

The final results obtained for large datasets are shown in Tables 6 and 7.

Table 6. Objective Values for Large Datasets

Rep	Avg. Running Time						
	FPA	**Logistic**	**Iterative**	**Sine**	**Tent**	**Singer**	**SA**
1	4567.99	**4412.4**	4505.07	4533.38	4510.19	4519.22	5514.86
2	4600.4	**4506.67**	4539.15	4504.73	4529.48	4535.89	5500.47
3	4633.85	**4451.87**	4532.25	4521.51	4556.95	4548.26	5516.23
4	4593.4	**4434.6**	4454.97	4524.34	4520.68	4491.66	5414.18
5	4582.48	**4289.28**	4471.19	4432.6	4497.7	4509.01	5449.84
Avg	4595.624	**4418.964**	4500.526	4503.312	4523	4520. 808	5514.86

Table 7. Result of Avg. Running Time for Large Datasets

Rep	Avg. Running Time						
	FPA	**Logistic**	**Iterative**	**Sine**	**Tent**	**Singer**	**SA**
1	5184.61	**2842.23**	3755.01	3566.31	4074.97	3493.76	**530.75**
2	5458.71	**2720.48**	4158.78	3469.83	3901.32	3955.34	**534.44**
3	5523.83	**3194.38**	3671.83	3027.47	3572.22	2945.71	**528.92**
4	5342.78	**3027.57**	3659.27	2893.76	3876.78	3092.57	**531.76**
5	5198.72	**2956.78**	4217.24	2976.06	3790.75	3087.26	**541.65**
Avg	5341.73	**2948.288**	3892.426	3186.686	3843.208	3314.928	**533.5**

From the results presented in Tables 2, 4, and 6 it can be seen that for all datasets, the hybrid FPA with Chaotic Maps provided the best results in term of total distance. While, for the running time for SA is still better compared to both FPA and hybrid FPA with Chaotic Maps as shown in Tables 3, 5 and 7. However, based on the average Running Time, Simulated Annealing has lower running time than those of FPA and FPA-CM. It can be understood because in FPA, there is step to convert continuous values into discrete values to get routes then again re-convert to continuous values to update the solutions. These steps need of course more computing time.

The experiment results can be analysize further. Here Average Relative Percentage Deviation (ARPD) is used to evaluate the results. ARPD is a metric commonly used in algorithm performance evaluation to compare the effectiveness of metaheuristic algorithms. It measures how far the solution found by an algorithm is from the best-known solution, typically expressed as a percentage. The ARPD is formulated as follows[11]:

$$ARPD = \left(\sum_{i=1}^{R} \frac{(S_i - B) * 100}{B} \right) \Big/ R$$

where Si shows the best result produced by the algorithm in run-i, B shows the best solution achieved among all the compared algorithms, and R shows the number of runs. A lower ARPD value indicates that the algorithm performs well and is close to best-known solution. From the results of the experiment of each algorithm using three datasets, the ARPDs were obtained, shown in Table 8.

From Table 8, it can be seen that for small, medium and large datasets, FPA-CM provided lower ARPD compared to the original FPA. Additionally, FPA with Chaotic Maps also outperform Simulated Annealing. It occurred for all types of Chaotic Maps. Specifically, Logistic function gave lowest ARPD for all datasets. But, it needs more investigation why Logistic function provided the best results.

Table 8. ARPD Value for Each Algorithm

Method	**Small Dataset**	**Medium Dataset**	**Large Dataset**
FPA	5.65	5.12	4.70
Logistic	**4.07**	**1.94**	**1.13**
Iterative	4.38	2.62	2.53
Sine	5.61	2.22	2.60
Tent	5.48	2.60	3.05
Singer	5.07	2.67	3.00
SA	18.79	20.53	24.83

The use of chaotic maps might improve the generation of solutions. The variability of the solutions generated is higher compared to the original FPA. The process of finding solutions could spans more search space. This can prevent from convergence too early. In turn the solution obtained in average is lower than those got by original FPA.

5 CONCLUSION

Based on the Average Relative Percentage Deviation (ARPD) value from the experimental results on section 4, it can be concluded that combination FPA and Chaotic Maps (FPA-CM) produces best solutions compared to the Original FPA and Simulated Annealing to solve VRPTW. However, FPA-CM is still needs improvement to shorten the computational time. Considering the promising result of using FPA-CM, Discrete FPA with modification of how generating solutions can be good alternative to solve discrete optimization problems.

This research was supported by Department of Industrial and System Engineering, Institut Teknologi Sepuluh Nopember Surabaya.

REFERENCES

1. Abdel-Basset, M., & Shawky, L. A. Flower pollination algorithm: a comprehensive review. In Artificial Intelligence Review (Vol. 52, Issue 4, pp. 2533–2557). Springer Netherlands. (2019) https://doi.org/10.1007/s10462-018-9624-4
2. A. Fattah Sayed, S. Nabil, E., & Badr, A. A binary clonal flower pollination algorithm for feature selection. Pattern Recognition Letters, 77, pp. 21–27 (2016). https://doi.org/10.1016/j.patrec.2016.03.014

3. A. Raouf, O., Abdel-Baset, M., & El-Henawy, I., A New Hybrid Flower Pollination Algorithm for Solving Constrained Global Optimization Problems. International Journal of Applied Operational Research, Vol. 4, Issue 2 (2014)
4. Braysy, O., A Fast Evolutionary Metaheuristic For The Vehicle Routing Problem With Time Windows, International Journal on Artificial Intelligence Tools, Vol. 12, Issue 2 (2003)
5. El-Shorbagy, M. A., Mousa, A. A., & Nasr, S. M., A chaos-based evolutionary algorithm for general nonlinear programming problems. Chaos, Solitons and Fractals, 85, pp. 8–21 (2016)
6. Garip, Z., Erhan Çimen, M., Karayel, D., Fuat Boz, A., Batik Garip, Z., & Erhan Cimen, M., The chaos-based whale optimization algorithms global optimization, Chaos Theory and Applications, 1(1), pp 51–63 (2019)
7. Heidari-Bateni, G., & Mcgillem, C. D. A Chaotic Direct-Sequence Spread-Spectrum Communication System, In IEEE Transactions On Communications, Vol. 42, (1994)
8. Kallehauge, B., Larsen, J., dan Marsen, O.B.G., Lagrangean Duality Applied on Vehicle Routing with time windows, Technical Report. IMM: Technical University of Denmark (2001)
9. Kallehauge, B., Larsen, J., Madsen, O. B. G., & Solomon, M. M., Chapter 3 Vehicle Routing Problem With Time Windows (2005)
10. Mantegna, R. N., Fast, accurate algorithm for numerical simulation of Lévy stable stochastic processes, Physical Review E, 49(5), 1994, pp.4677–4683
11. Mishra and D. Shrivastava, "A TLBO and a Jaya heuristics for permutation flow shop scheduling to minimize the sum of inventory holding and batch delay costs", Computers & Industrial Engineering, Vol 124, 2018, pp. 509-522.
12. B. Santosa, & T. Jin Ai, Pengantar Metaheuristik - Implementasi dengan Matlab (1st ed.), (ITS Tekno Sains, Surabaya 2017)
13. Savelsbergh, M., Vigo, D., Cordeau, J.-F., Laporte, G., & Savelsbergh, M. W. P., Chapter 6: Vehicle Routing, Hdanbook in OR & MS, Elsavier,Vol, 14, 2007, pp. 367-417
14. Toth, P., dan Vigo, D., An Overview of Vehicle Routing Problem. IN: Toth, P., Vigo, D, The Vehicle Routing Problem, Philadelphia: Society for Industrial and Applied Mathematics, 2002, pp. 1-6
15. Yang, X.-S., Flower Pollination Algorithm for Global Optimization. In Unconventional Computation and Natural Computation. Lecture Notes in Computer Science. Vol. 7445, pp. 240- 249 (2012)
16. Yeun, L. C., Ismail, R.W., Omar, K, dan Zirous, M., Vehicle Routing Problem: Model and solution, Journal of Quality Measurement and Analysis, vol. 4, no. 1, pp. 205-218 (2008)
17. Yuniarti, R., & Astuti, M,Penerapan Metode Saving Matrix Dalam Penjadwalan Dan Penentuan Rute Distribusi Premium Di SPBU Kota Malang, Jurnal Rekayasa Mesin, Vol. 4, Issue 1 (2013)