Contemporary Mathematics and Applications
Vol. 1, No. 1, 2019, pp. 46-55.

Penerapan Algoritma Kunang-Kunang pada *Open Vehicle Routing Problem* (OVRP)

Ihda Septiyafi[1], Herry Suprajitno[2,*] & Asri Bekti Pratiwi[3]

[1,2,3] Departemen Matematika, Fakultas Sains dan Teknologi, Kampus C
Universitas Airlangga, Jl. Mulyorejo, Surabaya

*Corresponding author: herry-s@fst.unair.ac.id

Abstract. This paper aims to solve Open Vehicle Routing Problem using Firefly Algorithm. Open Vehicle Routing Problem (OVRP) is a variant of Vehicle Routing Problem (VRP) where vehicles used to serve customers do not return to the depot after serving the last customer on each route. The steps of the Firefly Algorithm to handle OVRP are data input and initialization parameters, generating the initial population for each firefly, sorting population sources, calculating the value of the objective function and light intensity, comparing the intensity of light, performing movement, setting the best fireflies as g-best, doing random movement in the best fireflies as long as the maximum number of iterations has not been met. The program used to complete OVRP using the Firefly Algorithm is Borland C ++ and implemented in 3 case examples, namely small data with 18 customers, moderate data with 50 customers, and large data with 100 customers with the best total mileage of 211, 344 , 970.62, and 2531.83. The results obtained from the program output indicate that the more the number of iterations and the number of fireflies, then the results of the objective function (total mileage) obtained tend to be better so that these parameters affect the value of the objective function. While the absorption coefficient value (γ) does not give effect to the value of the objective function.

Keywords: *Vehicle Routing Problem (VRP), Open Vehicle Routing Problem (OVRP), Firefly Algorithm (FA).*

1 Pendahuluan

Perkembangan ilmu pengetahuan dan teknologi yang sangat pesat di negeri ini berdampak pada banyak hal, salah satunya mempengaruhi perkembangan industri. Pengaruh pertumbuhan industri yang paling menonjol yaitu persaingan masing-masing perusahaan. Menurut Siswanto [1] perusahaan bersaing secara bebas, sehingga mereka cenderung menggunakan sumber daya yang ada secara efisien guna meningkatkan kualitas dan kuantitas dari produk yang diproduksi. Tak hanya meningkatkan kualitas dan kuantitas produk tetapi juga memperbaiki sistem distribusi salah satunya dapat meminimalkan biaya operasional.

Menurut Li dkk [2] VRP dapat menghasilkan urutan rute untuk setiap kendaraan dan jarak total yang di tempuh dapat diminimalkan dengan melibatkan satu depot dan beberapa

pelanggan. Karenanya, setiap pelanggan memiliki permintaan dan lokasi yang diketahui. Permintaan masing-masing pelanggan dilayani tepat satu kunjungan dari satu kendaraan dengan adanya batasan kapasitas dan jarak tempuh maksimum. Selain itu, setiap kendaraan diharuskan berangkat dan kembali ke depot. Namun untuk beberapa kasus tertentu kendaraan tidak diharuskan kembali ke depot setelah melayani pelanggan terakhir. Kasus seperti ini merupakan perluasan dari VRP yaitu *Open Vehicle Routing Problem* (OVRP).

Menurut Mirhassani dan Abolghasemi *Open Vehicle Routing Problem* (OVRP) merupakan varian dari *Vehicle Routing Problem* (VRP) klasik dengan masing-masing kendaraan tidak kembali ke depot [3]. OVRP menghasilkan suatu urutan rute yang dimulai dari depot untuk melayani semua pelanggan dengan armada kendaraan yang tersedia dengan meminimalkan total jarak tempuh kendaraan. Dalam kehidupan nyata OVRP serupa dengan masalah penentuan rute ada kendaraan antar jemput sekolah [4], pengiriman surat kabar, susu, dan pengiriman logististik lainnya [5].

OVRP dapat diselesaikan dengan beberapa metode. Metode yang pernah digunakan untuk menyelesaikan kasus OVRP yaitu *Tabu Search Algorithm* [6], *Ant Colony Optimization* [7], *Particle Swarm Optimization* [3], metode heuristik [8], *Hybrid Genetic Algorithm with Tabu Search Algorithm* [9], *Imperalist Competitive Algorithm* [10], dan *Memetic Algorithm* [5].

Algoritma Kunang-Kunang merupakan salah satu algoritma yang dapat digunakan untuk menyelesaikan permasalahan kombinatorial. Algoritma ini dikembangkan oleh Dr. Xin-She Yang dari Universitas Cambridge di tahun 2008. Algoritma ini juga terinspirasi dari pola dan perilaku berkedip kunang-kunang. Setiap kunang-kunang tertarik untuk bergerak menuju kunang-kunang lainnya yang memiliki intensitas cahaya yang lebih terang [11]. Keunggulan algoritma ini dari algoritma lain seperti *Artificial Bee Colony* (ABC), *Particle Swarm Optimization* (PSO), *Genetic Algorithm* (GA) yakni lebih sederhana dalam pengaplikasiannya. Dalam beberapa kasus juga dapat ditemukan solusi optimal dengan cepat dengan tingkat keberhasilan yang cukup tinggi [12].

Oleh karena itu, untuk mengembangkan OVRP dibuatlah penerapan Algoritma Kunang-Kunang dalam menyelesaikan *Open Vehicle Routing Problem* (OVRP). Diharapkan penggunaan algoritma ini dapat memberikan hasil yang lebih baik dari algoritma yang telah digunakan sebelumnya.

2 *Open Vehicle Routing Problem* (OVRP)

Menurut Sariklis dan Powell [8] *Open Vehicle Routing Problem* (OVRP) merupakan perluasan masalah dari VRP. Sama halnya dengan VRP, OVRP dapat digambarkan sebagai permasalahan manajemen distribusi kepada pelanggan dengan permintaan tertentu yang dilayani dengan kendaraan yang memiliki kapasitas muatan tertentu yang

harus dimulai dari depot. Hal tersebut bertujuan untuk meminimalkan jarak dalam menentukan rute yang harus ditempuh untuk masing-masing kendaraan dalam memenuhi permintaan pelanggan. Namun yang membedakan diantara keduanya yaitu setiap kendaraan yang digunakan untuk melayani setiap rute tidak diakhiri di depot melainkan diakhiri di tempat pelanggan terakhir selesai dilayani.

Model matematika dari *Open Vehicle Routing Problem* (OVRP) dapat dirumuskan sebagai berikut [3]:

$$\min f = \sum_{k=1}^{K} \sum_{i=0}^{n} \sum_{j=0}^{n} d_{ij} x_{ij}^{k} \quad (1)$$

dengan :

$$x_{ij}^{k} = \begin{cases} 1, jika\ kendaraan\ k\ berjalan\ langsung\ dari\ pelanggan\ i\ ke\ pelanggan\ j \\ 0, yang\ lainnya \end{cases}$$

dan batasan-batasan:

1. Setiap pelanggan dan depot dikunjungi tepat satu kali oleh satu kendaraan pengangkut

$$\sum_{k=1}^{K} \sum_{i=0}^{n} x_{ij}^{k} = 1 ; \qquad \forall j = 1,2, \dots, n \quad (2)$$

$$\sum_{k=1}^{K} \sum_{j=1}^{n} x_{ij}^{k} = 1 ; \qquad \forall i = 1,2, \dots, n \quad (3)$$

2. Setiap kendaraan pengangkut yang mengunjungi suatu pelanggan, setelah selesai melayani akan meninggalkan pelanggan tersebut

$$\sum_{i=0}^{n} x_{iu}^{k} - \sum_{j=0}^{n} x_{uj}^{k} = 0 ; \qquad \forall k = 1,2, \dots, K ; \forall u = 1,2, \dots, n \quad (4)$$

3. Hanya ada satu kendaraan yang akan berangkat dari depot untuk melayani suatu urutan pelanggan

$$\sum_{j=1}^{n} x_{0j}^{k} = 1 ; \qquad \forall k = 1,2, \dots, K \quad (5)$$

4. Total permintaan dari setiap konsumen dalam rute yang dilalui setiap kendaraan pengangkut tidak boleh melebihi kapasitas muatan kendaraan

$$\sum_{j=1}^{n} q_j \left(\sum_{i=0}^{n} x_{ij}^{k} \right) \leq Q ; \qquad \forall k = 1,2, \dots, K \quad (6)$$

5. Tidak ada kendaraan pengangkut yang kembali ke depot setelah melayani pelanggan terakhir

$$\sum_{i=1}^{n} x_{i0}^{k} = 0 ; \qquad \forall k = 1,2, \dots, K \quad (7)$$

6. Memastikan bahwa setiap rute yang dilalui telah lengkap

$$\sum_{(i,j)\epsilon S\times S} x_{ij}^{k} \leq |S| - 1\ ;\ \ \forall S \subseteq V : 1 \leq |S| \leq n\ ;\ \forall k = 1,2,\dots,K \quad (9)$$

Keterangan :

d_{ij} = Jarak dari pelanggan *i* ke pelanggan *j*
x_{ij}^{k} = Kendaraan k yang melayani pelanggan i setelah mengunjungi pelanggan j
q_j = Permintaan dari pelanggan *j*
n = Jumlah pelanggan
K = Jumlah kendaraan
Q = Kapasitas maksimum kendaraan
V = Sekumpulan pelanggan dan depot
S = Sub himpunan dari himpunan pelanggan
$|S|$ = Kardinalitas himpunan *S*
i, j, u = Indeks pelanggan
k = Indeks kendaraan

3 Algoritma Kunang-Kunang

Kunang-kunang adalah kumbang bersayap kecil yang dapat menghasilkan cahaya untuk menarik pasangan. *Firefly Algorithm* atau Algoritma Kunang-Kunang merupakan algoritma metaheuristik yang terinspirasi dari perilaku berkedipnya kunang-kunang yang membuat kunang-kunang lain tertarik. Algoritma ini menganggap bahwa setiap kunang-kunang diamati pada titik posisinya, yaitu ketika kunang-kunang mencoba untuk pindah menuju menuju sumber cahaya yang lebih besar dari dirinya sendiri.

Algoritma ini diasumsikan sebagai berikut:

1. Semua kunang-kunang tidak tertarik berdasarkan jenis kelaminnya karena kunang-kunang merupakan hewan yang *unisex*.
2. Ketertarikan antar kunang-kunang bergantung pada tingkat kecerahan cahayanya, dimana kunang-kunang yang cahayanya lebih redup akan menuju kunang-kunang dengan tingkat cahaya yang lebih terang.
3. Fungsi tujuan sangat mempengaruhi kecerahan kunang-kunang.

Langkah-langkah Algoritma Kunang-Kunang yang digunakan dalam menyelesaikan permasalahan pada penelitian ini adalah sebagai berikut :

1. Menginputkan data dan parameter algoritma kunang-kunang, antara lain banyaknya kunang-kunang (*firefly*) (m), maksimal iterasi, ukuran untuk langkah acak (α), dan koefisien penyerapan cahaya/absorpsi (γ).

2. Membangkitkan populasi awal
   (i) Membangkitkan bilangan real acak pada masing-masing *firefly* sebanyak n pelanggan pada interval (0,1).
   (ii) Mengurutkan bilangan real acak sehingga diperoleh nomor urutan bilangan bulat acak.
   (iii) Menampilkan nomor urutan sesuai dengan posisi bilangan real acak awal.
   (iv) Menentukan rute yaitu dimulai dari depot dan berakhir apabila kapasitas maksimum terpenuhi.
3. Menghitung total jarak yang dilalui oleh kendaraan yang akan menjadi nilai fungsi tujuan *f(x)*.
4. Fungsi tujuan ditransformasikan menjadi intensitas cahaya dengan rumus *I(x)* = 10000/*f(x)*.
5. Membandingkan intensitas cahaya *firefly* i dengan *firefly* j (i = 1,2,...,m; j = 1,2,...,m). Jika $I_i > I_j$ maka intesitas *firefly* i dibandingkan dengan *firefly* j yang lain sampai semua *firefly* j selesai dibandingkan dengan *firefly* i, tetapi apabila $I_j > I_i$maka
   (i) Terjadi pergerakan yaitu *firefly* i bergerak menuju *firefly* j.
   (ii) Setelah pergerakan terjadi kemudian menghitung *distance*, *attractiveness*, dan *movement* dengan formula sebagai berikut :

$$d_{i,j} = distance\,(x_i, x_j) = \sqrt{\sum_{n=1}^{m}\left(x_i^n - x_j^n\right)^2} \qquad (9)$$

$$\beta = I_0 e^{-\gamma d^2} \qquad (10)$$

$$x_i' = (1-\beta)x_i + \beta x_j + \alpha\left(random\,(0,1) - \frac{1}{2}\right) \qquad (11)$$

   sehingga didapatkan posisi baru dari *firefly* i. Berikutnya menghitung intensitas cahaya baru dari posisi baru *firefly* i.
   (iii) Setelah mendapatkan intensitas cahaya baru dari *firefly* i kemudian melakukan perbandingan intensitas cahaya dengan *firefly* lainnya dan apabila $I_j > I_i'$ maka kembali pada proses (i) dan (ii) sampai semua *firefly* telah selesai dibandingkan. Jika *firefly* mempunyai posisi dan intensitas cahaya baru, maka yang dibandingkan adalah intensitas cahaya *firefly* yang baru.
6. Menentukan *G-best* dari *firefly* dengan intensitas cahaya yang paling besar pada iterasi tersebut.
7. Meng-*update G-best* yaitu dengan membandingkan *G-best* pada iterasi sebelumnya dengan *G-best* pada iterasi saat itu, lalu dipilih yang paling besar intensitas cahayanya yang kemudian menjadi *G-best'*.
8. Ketika iterasi maksimal belum terpenuhi, maka *firefly* yang menjadi *G-best'* melakukan *random movement* dengan formula sebagai berikut :

$$x'best = x' + \alpha\left(random\,(0,1) - \frac{1}{2}\right) \qquad (12)$$

   sehingga didapatkan posisi dan intensitas cahaya baru untuk *firefly* tersebut. Intensitas cahaya baru yang didapatkan akan langsung dibandingkan dengan intensitas cahaya dari *firefly* lainnya pada iterasi berikutnya.

9. Mengulangi proses e) hingga maksimal iterasi terpenuhi.
10. Selanjutnya cetak *G-best'* untuk menjadi solusi

Keterangan :

m	= banyak dimensi kunang-kunang
x_i	= posisi dari kunang-kunang *i*
x_j	= posisi dari kunang-kunang *j*
n	= banyak dimensi pelanggan
β	= nilai *attractiveness*
α	= ukuran untuk langkah acak nilai [0,1]
γ	= koefisien absorpsi nilai [0,1]
d	= jarak antar kunang-kunang
x'	= posisi dari kunang-kunang yang menjadi *G-best* setelah mengalami pergerakan
$x'best$	= posisi baru dari kunang-kunang yang menjadi *G-best* setelah mengalami pergerakan
x_i'	= posisi baru kunang-kunang yang bergerak
I_0	= intensitas cahaya di titik awal

4 Hasil dan Pembahasan

Data yang digunakan pada permasalahan ini merupakan data sekunder yang didapatkan dari https://www.coin-or.org/SYMPHONY/branchandcut/VRP/ data/index.htm.old. Data terdiri atas data permintaan masing-masing pelanggan, posisi depot dan pelanggan, jumlah kendaraan yang tersedia, dan kapasitas maksimal masing-masing kendaraan. Terdapat 3 jenis data yang digunakan untuk permasalahan ini, yaitu data dari data kecil (18 pelanggan), data sedang (50 pelanggan), dan data besar (100 pelanggan).

Pengujian performansi dengan menggunakan data kecil, yaitu 18 pelanggan menggunakan parameter : jumlah kunang-kunnag (*m*) = 10; 25; 50, jumlah iterasi maksimal (*max_iterasi*) = 10; 250; 500, kapasitas kendaraan (*max_kpk*) untuk data ini yaitu 160, jumlah kendaraan (*kendaraan*) = 3, koefisien langkah acak (α) = 0,2 [13], koefisien absorpsi (γ) = 0,1; 0,5; 0,9. Hasil akhir running data kecil yang diperoleh disajikan pada Tabel 1.

Dari Tabel 1 dapat diambil kesimpulan bahwa semakin banyak iterasi dan *firefly* hasil yang diperoleh cenderung lebih baik serta dengan adanya γ juga mempengaruhi hasil *running* program dan dengan jumlah kendaraan yang tersedia maka didapatkan nilai fungsi tujuan (total jarak tempuh) terbaik dari 18 pelanggan adalah 211,344 dengan rute sebagai berikut :

Rute 1 $\rightarrow$ 0 – 10 – 12 – 4 – 11 – 14 – 16 – 8 – 3 – 17

Rute 2 → 0 – 6 – 18 – 5 – 13 – 15 – 7 – 9
Rute 3 → 0 – 2 – 1

Tabel 1 Nilai Fungsi Tujuan Data Kecil (18 pelanggan)

Jumlah firefly	10			25			50		
Γ	0,1	0,5	0,9	0,1	0,5	0,9	0,1	0,5	0,9
Iterasi 10	295,063	277,212	276,584	300,096	280,004	280,87	278,012	249,935	232,604
Iterasi 250	270,679	234,592	223,075	242,911	249,932	235,882	256,207	221,125	232,087
Iterasi 500	248,969	236,017	232,046	238,376	231,05	211,344	238,323	218,686	227,276

Pengujian performansi dengan menggunakan data kecil, yaitu 18 pelanggan menggunakan parameter : jumlah kunang-kunang (*m*) = 10; 25; 50, jumlah iterasi maksimal (*max_iterasi*) = 10; 250; 500, kapasitas kendaraan (*max_kpk*) untuk data ini yaitu 80, jumlah kendaraan (*kendaraan*) = 10, koefisien langkah acak (α) = 0,2 [13], koefisien absorpsi (γ) = 0,1; 0,5; 0,9.

Hasil *running* program menggunakan data ini tidak ditemukan solusi karena jumlah kendaraan yang tersedia tidak dapat memenuhi semua rute untuk melayani pelanggan maka di setiap parameternya ditambahkan jumlah kendaraan yang berbeda-beda agar didapatkan solusi yang optimal. Nilai fungsi tujuan dari keseluruhan proses *running* dengan adanya tambahan kendaraan menggunakan data ini dapat dilihat pada Tabel 2.

Tabel 2 Nilai Fungsi Tujuan Data Sedang (50 pelanggan)

Jumlah firefly	10			25			50		
Γ	0,1	0,5	0,9	0,1	0,5	0,9	0,1	0,5	0,9
Iterasi 10	1269,83	1226,87	1255,45	1201	1158,32	1165,45	1144,92	1176,39	1160,68
Iterasi 250	1182,03	1151,65	1125,76	1098,43	1162,22	1127,49	1103,73	1112,35	1090,35
Iterasi 500	1135,3	1156,46	1135,3	1085,33	1015,48	1090,07	970,62	1089,88	986,595

Dari Tabel 2 dapat diambil kesimpulan bahwa semakin banyak iterasi dan *firefly* hasil yang diperoleh cenderung lebih baik serta dengan adanya γ juga mempengaruhi hasil *running* program dan didapatkan nilai fungsi tujuan (total jarak tempuh) terbaik dari 50

pelanggan dengan adanya tambahan kendaraan sebanyak 1 kendaraan adalah 970,62 dengan rute sebagai berikut :

Rute 1 → 0 – 18 – 14 – 43
Rute 2 → 0 – 49 – 10 – 47 – 15 – 17 – 26 – 4
Rute 3 → 0 – 6 – 1 – 24 – 23 – 7
Rute 4 → 0 – 27 – 32 – 11 – 13
Rute 5 → 0 – 48 – 25 – 41
Rute 6 → 0 – 2 – 50 – 20 – 31
Rute 7 → 0 – 46 – 35 – 3 – 36 – 28 – 29 – 21
Rute 8 → 0 – 5 – 30 – 33 – 39
Rute 9 → 0 – 44 – 19 – 42 – 40 – 38 – 16
Rute 10 → 0 – 8 – 9 – 34
Rute 11 → 0 – 12 – 22 – 45 – 37

Pengujian performansi dengan menggunakan data kecil, yaitu 18 pelanggan menggunakan parameter : jumlah *firefly* (*m*) = 10; 25; 50, jumlah iterasi maksimal (*max_iterasi*) = 10; 250; 500, kapasitas kendaraan (*max_kpk*) untuk data ini yaitu 400, jumlah kendaraan (*kendaraan*) = 4, koefisien langkah acak (α) = 0,2 [13], koefisien absorpsi (γ) = 0,1; 0,5; 0,9. Hasil akhir yang diperoleh disajikan pada Tabel 3.

Tabel 3 Nilai Fungsi Tujuan Data Besar (100 pelanggan)

Jumlah firefly	10			25			50		
γ	0,1	0,5	0,9	0,1	0,5	0,9	0,1	0,5	0,9
Iterasi 10	2846,23	2936,39	2856,5	2758,7	2837,24	2743,91	2652,23	2791,03	2731,88
Iterasi 250	2755,04	2739,1	2787,79	2749,29	2531,83	2736,05	2644,18	2606,2	2671,79
Iterasi 500	2713,58	2687,62	2715,14	2652,85	2574,41	2682,94	2579,86	2564,45	2541,69

Dari Tabel 3 dapat diambil kesimpulan bahwa semakin banyak iterasi dan *firefly* hasil yang diperoleh cenderung lebih baik serta dengan adanya γ juga mempengaruhi hasil *running* program dan dengan jumlah kendaraan yang tersedia maka didapatkan nilai fungsi tujuan (total jarak tempuh) terbaik dari 100 pelanggan adalah 2531,83 dengan rute sebagai berikut :

Rute 1 → 0 – 48 – 50 – 53 – 16 – 40 – 31 – 65 – 25 – 56 – 77 – 85 – 90 – 51 – 63 – 11 – 47 – 42 – 57 – 97 – 83 – 17 – 99 – 96 – 41 – 75 – 23 – 2 – 71

Rute 2 → 0 – 1 – 81 – 58 – 29 – 24 – 37 – 92 – 26 – 66 – 34 – 55 – 79 – 33 – 15 – 46 – 5 – 36 – 72 – 9 – 13 – 94 – 32 – 19 – 8 – 20 – 84 – 18 – 10

Rute 3 → 0 – 86 – 43 – 91 – 82 – 45 – 38 – 61 – 6 – 21 – 59 – 93 – 14 – 87 – 89 – 52 – 49 – 70 – 30 – 74 – 68 – 88 – 12 – 78 – 62 – 64

Rute 4 → 0 – 100 – 4 – 76 – 28 – 27 – 80 – 35 – 39 – 67 – 54 – 22 – 73 – 98 – 60 – 44 – 95 – 7 – 3 – 69

5 Kesimpulan

Berdasarkan uraian diatas, maka dapat disimpulkan bahwa Algoritma Kunang-Kunang dapat diterapkan untuk menyelesaikan *Open Vehicle Routing Problem* (OVRP). Program untuk menyelesaikan OVRP dengan menggunakan Algoritma Kunang-Kunang dapat dibuat menggunakan bahasa pemrograman C++ menggunakan *software* Borland C++. Implementasi program untuk menyelesaikan contoh kasus dengan menggunakan parameter yang berbeda-beda menghasilkan nilai fungsi tujuan yang bervariasi pada data kecil (18 pelanggan), data sedang (50 pelanggan), dan data besar (100 pelanggan). Dari hasil implementasi tersebut dapat disimpulkan bahwa baik dari jumlah iterasi maksimal, jumlah kunang-kunang, dan nilai koefisien absorpsi (γ) semuanya dapat mempengaruhi nilai fungsi tujuan (total jarak tempuh) dengan hasil yang cenderung lebih baik apabila masing-masing parameter tersebut semakin besar.

6 Daftar Pustaka

[1] Siswanto, Arie. 2002. *Hukum Persaingan Usaha*. Bogor: Ghalia Indonesia.

[2] Li, F., Golden, B., dan Wasil, E. 2007. *The Open Vehicle Routing Problem: Algorithms, Large-Scale Test Problems, and Computational Results*. Journal of Computers & Operations Research 34: 2918-2930.

[3] Mirhassani, S.A. dan Abolghasemi, N. 2011. *A Particle Swarm Optimization Algorithm for Open Vehicle Routing Problem*. Journal of Expert Systems with Applications 38: 11547-11551.

[4] Aksen, D., Ozyurt, Z., dan Aras, N. 2007. *Open Vehicle Routing Problem with Driver Nodes with Time Deadlines*. Journal of The Operational Research Society, Vol. 58 (9): 1223-1234.

[5] Hosseinabadi, A. A. R., Vahidi, J., dan Balas, V. E. 2018. *OVRP_GELS: Solving Open Vehicle Routing Problem using The Gravitational Emulation Local Search Algorithm*. Journal of Neural Comput & Applic 29: 955-968.

[6] Brandao, J. 2004. *A Tabu Search Algorithm for The Open Vehicle Routing Problem*. European Journal of Operational Research 157: 552-564.

[7] Gurpreetsingh, Er. dan Dhir, Dr. Vijay. 2014. *Open Vehicle RoutingProblem by Ant Colony Optimization*. International Journal of Advanced Computer Science and Applications (IJACSA), Vol. 5, No. 3.

[8] Sariklis, D. dan Powell, S. 2000. *A Heuristic Method for Open Vehicle Routing Problem*. The Journal of the Operational Research Society, Vol. 51, No. 5: 564-573.

[9] Yu, Sh., Ding, Ch., dan Zhu, K. 2011. *A Hybrid GA-TS Algorithm for Open Vehicle Routing Optimization of Coal Mines Material*. Journal of Expert Systems with Applications 38: 10568-10573.

[10] Yousefi, K. M., Didaver, F., Rahmati, F., dan Sedighpour, M. 2012. *An Effective Imperialist Competitive Algorithm for Solving The Open Vehicle Routing Problem*. Journal of Transportation Research, Vol. 9, No. 1 (30): 83-95.

[11] Yang, X. S. 2009. *Firefly Algorithm for Multimodal Optimization, in Stochastic Algorithms: Foundations and Application*, Vol. 5792: 169-178.

[12] Sayadi, M. K., Ramezanian, R., dan Ghafari-Nasab, Nader. 2010. *A Discrete Firefly Meta-Heuristic with Local Search for Makespan Minimazation in Permutation Flow Shop Scheduling Problems*. International Journal of Industrial Engineering Computations 1: 1-10.

[13] Sadjadi, S. J., Ashtiani, M. G., Ramezanian, R., & Makui, A. 2016. *A Firefly Algorithm for Solving Competitive Location-Design Problem: A Case Study*. Journal of Industrial Engineering International 12: 517-527.