View metadata, citation and similar papers at core.ac.uk

brought to you by CORE

provided by Rumah Jurnal Online - Fakultas Sains dan Teknologi UIN Sunan Ampel Surabaya

Jurnal Matematika MANTIK
Volume 5, Nomor 2, October 2019, pp. 100-111

ISSN: 2527-3159 (*print*) 2527-3167 (*online*)

Use of Ant Colony Optimization Algorithm for Determining Traveling Salesman Problem Routes

Bib Paruhum Silalahi[1], Nurul Fathiah[2], Prapto Tri Supriyo[3]

Department of Mathematics, IPB University, bibparuhum@gmail.com[1]
Department of Mathematics, IPB University, nurulfathiah393@gmail.com[2]
Department of Mathematics, IPB University, praptotrisupriyo@gmail.com[3]

doi: https://doi.org/10.15642/mantik.2019.5.2.100-111

Abstrak: Optimisasi Koloni Semut merupakan salah satu metode *meta-heuristic* yang digunakan untuk menyelesaikan masalah optimisasi kombinatorial yang cukup sulit. Algoritma Optimisasi Koloni Semut diinspirasi dari perilaku semut dalam dunia nyata untuk membangun jalur terpendek antara sumber makanan dan sarangnya. *Traveling Salesman Problem* adalah suatu permasalahan dalam optimisasi. *Traveling Salesman Problem* merupakan permasalahan untuk mencari jarak minimal dari node awal menuju node selanjutnya, dengan setiap node harus didatangi persis satu kali dan harus kembali ke node awal. *Traveling Salesman Problem* merupakan suatu masalah yang *non-deterministic polynomial-time complete*. Pada penelitian ini didiskusikan penyelesaian *Traveling Salesman Problem* menggunakan algoritma Optimisasi Koloni Semut dan juga menggunakan algoritma eksak. Hasil penelitian menunjukkan bahwa semakin besar ukuran kasus *Traveling Salesman Problem*, waktu eksekusi yang dibutuhkan semakin lama. Kemudian diperoleh hasil bahwa waktu eksekusi yang dihasilkan Optimisasi Koloni Semut jauh lebih cepat dibandingkan waktu eksekusi metode eksak.

Kata kunci: *Algoritma, Optimisasi , Optimisasi Koloni Semut, Traveling Salesman Problem*

Abstract: Ant Colony Optimization is one of the meta-heuristic methods used to solve combinatorial optimization problems that are quite difficult. Ant Colony Optimization algorithm is inspired by ant behaviour in the real world to build the shortest path between food sources and their nests. Travelling Salesman *Problem* is a problem in optimization. Travelling Salesman Problem is a problem to find the minimum distance from the initial node to the whole node with each node must be visited exactly once and must return to the initial node. Travelling Salesman Problem is a non-deterministic polynomial-time complete problem. This research discusses the solution of the Traveling Salesman Problem using the Ant Colony Optimization algorithm and also using the exact algorithm. The results showed that the greater the size of the Traveling Salesman Problem case, the longer the execution time required. The results also showed that the execution times of the Ant Colony Optimization are much faster than the execution time of the exact method.

Keywords*: Algorithm, Optimization, Ant Colony Optimization, Traveling Salesman Problem*

How to cite: *B. P. Silalahi, N. Fathiah, and P. T. Supriyo, "Use of Ant Colony Optimization Algorithm for Determining Traveling Salesman Problem Routes", Mantik, vol. 5, no. 2, pp. 100-111, October 2019.*

1. Introduction

Optimization is the study of techniques for finding the maximum or minimum value of a problem. Some examples of studies of optimization techniques are modifying of steps in the steepest descent method [1] - [3], computational test of Newton's method variance algorithm on nonlinear optimization problems without constraints [4], comparison of sensitivity analysis on linear optimization using optimal partition and optimal basis [5], elaborating a combination method for solving nonlinear equations [6], overview of Newton's optimization method [7]. Optimization is applied in many aspects of life such as generating star catalogue for satellite attitude navigation using density-based clustering [8], but many of them are applied to transportation and distribution systems [9] - [11].

Travelling Salesman Problem (TSP) is an example of problems in optimization. Problems associated with TSP are often encountered in distribution systems. Basically, TSP is a problem to find the minimum distance from the initial node to the next node with the rule that each node must be visited exactly once. Furthermore, after all the nodes visited exactly once, they must return to the initial node. One of the characteristics of TSP is non-deterministic polynomial-time complete (NPC) which implies that there is no optimal solution other than having to try all possible solutions [12].

The application of TSP in the real world is often used in the case of distributing goods and services. Problems that are often encountered relating to distribution such as determining travel routes that minimize travel time, distance travelled, or operational costs.

TSP can be solved by exact, heuristic or meta-heuristic methods. One of the meta-heuristic methods that can be used to solve the TSP is the Ant Colony Optimization algorithm. Some other meta-heuristic methods such as genetic algorithm [13], particle swarm optimization [14], firefly algorithm [15][16] and interior point method [17], [18]. Ant Colony Optimization (ACO) is adapted from the behaviour of the ant colony as a system for finding and collecting food. Ants have the ability to find food effectively by finding the shortest path between the food source and the nest without using the sense of sight at the beginning of the ant, walking random looking for food. Along the way, the ant will leave a trail called pheromones. After successfully obtaining food, the ant will return to the nest with a trail guide pheromone left behind. Other ants will take food by following in the footsteps of pheromones. Pheromones can evaporate. The longer the distance of a route, the longer the ants are on the route, so that the pheromones on the route will be reduced faster because it evaporates when compared to the shorter distance route. This causes the ants to tend to choose the shortest route [19].

ACO is widely used as a solution to various optimization problems, one of which is TSP. This algorithm is compiled using distance data between locations, some ants that communicate with their colonies indirectly using the help of pheromones.

2. Research Method

In this research, TSP cases will be solved using the Integer Linear Programming (ILP) model and the Ant Colony Optimization (ACO) model.

Furthermore, the results of the two models will be compared specifically in the aspects of execution time and total travel distance. Both models were executed using a laptop with the specifications of ASUS A 455LD i5 8GB RAM.

3. Results and Discussion

3.1. Travelling Salesman Problem

TSP is a problem of determining the route when someone will come to all locations that have been determined. Starting from the first location and then visiting all the locations that have been determined exactly once. Then after that, the person/object must return to the first location. What we want to achieve is the minimum total travel distance [20].

TSP can be formulated as follows, for example:

n : number of nodes to be visited,

c_{ij} : distance of node i to node j,

u_i : additional variables when visiting node i.

Decision variable:

$$x_{ij} = \begin{cases} 1, & \text{if there is a trip from node } i \text{ to node } j \\ 0, & \text{else} \end{cases}$$

The objective function of the TSP is to find a value so that the total distance traveled is minimum, that is:

$$\min Z = \sum_{i=1}^{n} \sum_{\mathrm{j}=1}^{n} c_{ij}\, x_{ij}.$$

Constraints:

1. Nodes must be visited exactly once.
$$\sum_{i=1}^{n} x_{ij} = 1, \qquad \forall j = 1,2,\dots,n$$
$$\sum_{j=1}^{n} x_{ij} = 1, \qquad \forall i = 1,2,\dots,n$$
2. No sub tour formed on the journey in the TSP. This means that there is only one tour from TSP. Because returning to the initial location will form a cycle, $u_i - u_j + nx_{ij} \leq n - 1, i \neq j, \forall i = 2,3,\dots,n;\ \forall j = 2,3,\dots,n;\ u_j \geq 0$.

3.2. Ant Colony Optimization Algorithm

Ant Colony Optimization (ACO) is one of the meta-heuristic methods used to find a combinatorial optimization problem that is quite difficult. The ACO algorithm is represented by ant behaviour in the real world to build the shortest path between food sources and their nests. Each ant randomly starts its route from a node. Each node is visited by ants to form a route, and this is done repeatedly. The ant selects the nodes to be visited using the probability function, based on the distance of the node, and the number of pheromones found on the side connecting the node. Nodes that have smaller distances and have higher levels of pheromones will be more likely to be visited by ants [21].

Nodes that have been visited by ants will be recorded in memory which we call by the name tabulist. This tabulist prevents ants from going to the nodes that have been visited. The tabulist will be full when all ants have visited all nodes. Then the pheromone update rules are applied. Calculation of the loss of

pheromone levels on each side is done. For shorter routes, the loss of pheromone levels will be relatively longer compared to the longer routes.

The process will be repeated until the tour reaches the maximum value or the system is in a stagnant situation, which is a situation where the system is no longer looking for other alternative solutions [22]. Figure 1 shows a simple ACO flow.

3.3. Steps of Ant Colony Optimization Algorithm

The flow of the ACO algorithm is shown in Figure 1. The steps taken are as follows.

1. Data The initial data presented is only in the form of x and y coordinate points, so the distance between nodes must be calculated using the Euclidian formula:

$$d_{ij} = \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2} \ ,$$

where

d_{ij} : the distance between node i and node j

x_i : x coordinate of node i, x_j : x coordinate of node j,

y_i : y coordinate of node i, y_j : y coordinate of node j.

2. Determine the next node the ant will pass by calculating the probability using the formula:

$$P_{ij}^k(t) = \frac{\left[\tau_{ij}(t)\right]^\alpha \cdot \left[\eta_{ij}(t)\right]^\beta}{\sum_{j \in N_i^k}\left[\tau_{ij}(t)\right]^\alpha \cdot \left[\eta_{ij}(t)\right]^\beta} \ , \text{if } j \in N_i^k$$

where:

$P_{ij}^k(t)$: the probability of ant k run from node i to node j at time t,

$\tau_{ij}(t)$: number of ant pheromones from node i to node j at time t,

$\eta_{ij}(t)$: distance inverse between node i and node j at time t,

N_i^k : the set of points to be visited by ant k which is at point i,

α: ant pheromones parameter,

β: distance control parameter,

and informing the route, the next node selection criteria use the Roulette Wheel Selection algorithm, with the following steps:

a. determine one random value between 0 and 1,
b. calculate the cumulative probability value $P_{ij}^k(t)$,
c. compare the random value in step a with each cumulative probability value $P_{ij}^k(t)$ in step b,
d. if the cumulative probability value $P_{ij}^k(t)$ is greater than the random value, then it will be selected as the next node.

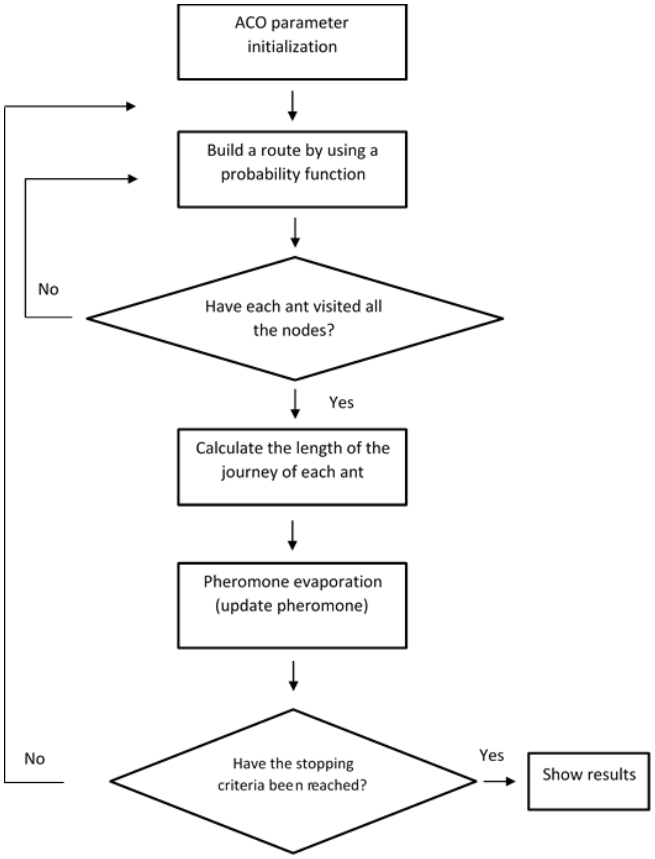

Figure 1. The flow of Ant Colony Optimization Algorithm.

3. Repeat step 2 for each ant until all the nodes are selected and form a route.
4. Calculate the total distance of each node that each ant passes.
5. Choose the best route from all the routes produced, i.e. the route with the minimum total distance obtained in step 4.
6. Update the pheromone with the following formula:

$$\tau_{ij}(t) = (1-\rho)\tau_{ij}(t) + \Delta\tau_{ij}(t)$$

with $\Delta\tau_{ij}(t) = \begin{cases} \frac{1}{L_k}, & \text{if ant } k \text{ run from node } i \text{ to node } j \\ 0, & \text{else} \end{cases}$

where:

$\tau_{ij}(t)$: number of ant pheromones from node i to node j at time t,

$\Delta\tau_{ij}(t)$: changes in pheromone values in the node (i,j) made by ant k,

L_k : travel length of ant k,

ρ : global pheromone evaporation parameter with $0 < \rho \leq 1$,

pheromone updates are carried out with the aim of updating the amounts of pheromones left by ants as a result of evaporation. The number of pheromones that have been updated will be used as information for ants in the next iteration.

7. This algorithm is terminated if the stopping criteria have been reached, i.e. when the number of iterations has been reached.
8. Return to step 2 if the stopping criteria have not been reached.

3.4. Solving by Using Integer Linear Programming (ILP)

Accomplishment by using ILP is carried out with the help of LINGO 11.0 software. The data used are x and y coordinate which indicate the location of the nodes in Cartesian coordinates. The data are hypothetic, obtained from a random function. The x and y coordinate data are presented in Table 1. From the data in Table 1, the distance between nodes is searched using the euclidian formula to produce a distance matrix.

Case I

The results of execution using ILP for the case of 30 nodes required execution time of 1 minute 34 seconds with the number of iterations of 436.649 and obtained a minimum total travel distance 388.371 km.

Case II

The results of execution using ILP for the case of 35 nodes required 11 minutes 04 seconds of execution time with the number of iterations 3.351.017 and obtained a minimum total travel distance 427.584 km.

Case III

The results of execution using ILP for the case of 38 nodes required execution time of 6 hours 38 minutes 14 seconds with the number of iterations 142.214.825 and the result obtained a minimum total travel distance 430.251 km.

3.5. Solving by Using Ant Colony Optimization Algorithm

In this work, the helping of a mathematics software is used to find the solution of the TSP using the ACO algorithm. Each case uses different parameter values α, β, and *m* (number of ants). The value of α is at the interval $0<\alpha\leq1$, $\beta>0$, and $m>0$ [23]. The values of α, β, dan *m*, which can produce the smallest total approach distance will be presented.

Case I

In case I, the experiments are carried out using parameters with values as presented in Table 2. After conducting several experiments, the smallest total approach distance is 392.8014 km with execution time 11.836 s in 500 iterations. The route obtained is as shown in Figure 2.

Case II

In case II, the experiments are carried out using parameters with values, as shown in Table 3. After conducting several experiments, the smallest total approach distance was 463.4509 km with an execution time 21.340 s in 700 iterations. Figure 3 shows the route obtained.

Case III

In case III, experiments will be conducted by using parameters with the values presented in Table 4. After conducting several experiments, the smallest total approach distance is 464.7083 km with an execution time of 38.00 s in 800 iterations. The route obtained is presented in Figure 4.

Table 1. The data in each case

Case I		Case II		Case III	
x	y	x	y	x	y
81.91	37.47	81.91	37.47	81.91	37.47
83.16	49.92	83.16	49.92	83.16	49.92

Case I		Case II		Case III	
x	y	x	y	x	y
84.76	40.37	84.76	40.37	84.76	40.37
75.27	63.61	75.27	63.61	75.27	63.61
38.10	28.80	38.10	28.80	38.10	28.80
23.28	95.00	23.28	95.00	23.28	95.00
77.49	15.56	77.49	15.56	77.49	15.56
39.22	22.32	39.22	22.32	39.22	22.32
62.88	32.60	62.88	32.60	62.88	32.60
36.63	65.29	36.63	65.29	36.63	65.29
17.02	41.47	17.02	41.47	17.02	41.47
89.50	30.82	89.50	30.82	89.50	30.82
17.02	52.23	17.02	52.23	17.02	52.23
13.25	100.12	13.25	100.12	13.25	100.12
73.20	52.75	73.20	52.75	73.20	52.75
34.08	89.16	34.08	89.16	34.08	89.16
55.29	90.36	55.29	90.36	55.29	90.36
41.42	25.66	41.42	25.66	41.42	25.66
53.31	90.58	53.31	90.58	53.31	90.58
42.66	31.08	42.66	31.08	42.66	31.08
54.49	56.85	54.49	56.85	54.49	56.85
99.85	60.28	99.85	60.28	99.85	60.28
63.39	100.56	63.39	100.56	63.39	100.56
21.86	77.04	21.86	77.04	21.86	77.04
35.09	36.91	35.09	36.91	35.09	36.91
56.86	61.57	56.86	61.57	56.86	61.57
83.89	52.24	83.89	52.24	83.89	52.24
53.04	56.22	53.04	56.22	53.04	56.22
34.06	37.48	34.06	37.48	34.06	37.48
66.57	25.84	66.57	25.84	66.57	25.84
		34.56	20.48	34.56	20.48
		63.63	63.48	63.63	63.48
		56.74	22.22	56.74	22.22
		15.90	62.84	15.90	62.84
		88.86	79.06	88.86	79.06
				73.48	82.00
				85.58	55.79
				90.92	54.49

Table 2. The results of case I.

The Number of Iterations	α	β	m	Total Distance (in km)
300	1	5	50	425.1656
350	1	5	50	417.1519
400	1	5	50	407.1231
450	1	5	50	399.8285
500	0.9	5	50	420.6379
500	1	5	50	392.8014

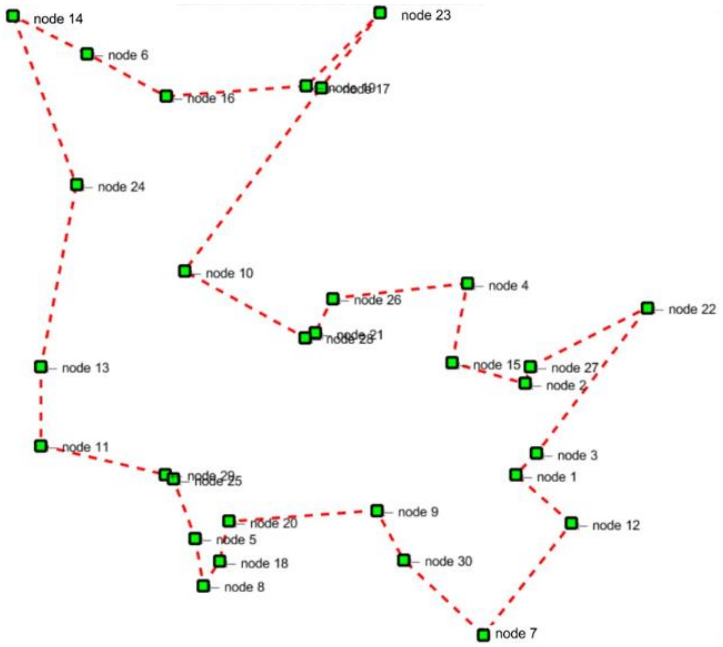

Figure 2. Route approach in case I.

Table 3. The results of case II.

The Number of Iterations	α	β	m	Total Distance (in km)
300	0.7	1	50	535.0748
300	1	5	50	497.4685
400	1	5	50	481.1290
500	1	5	50	472.6510
600	1	5	50	470.1542
700	1	5	50	463.4509

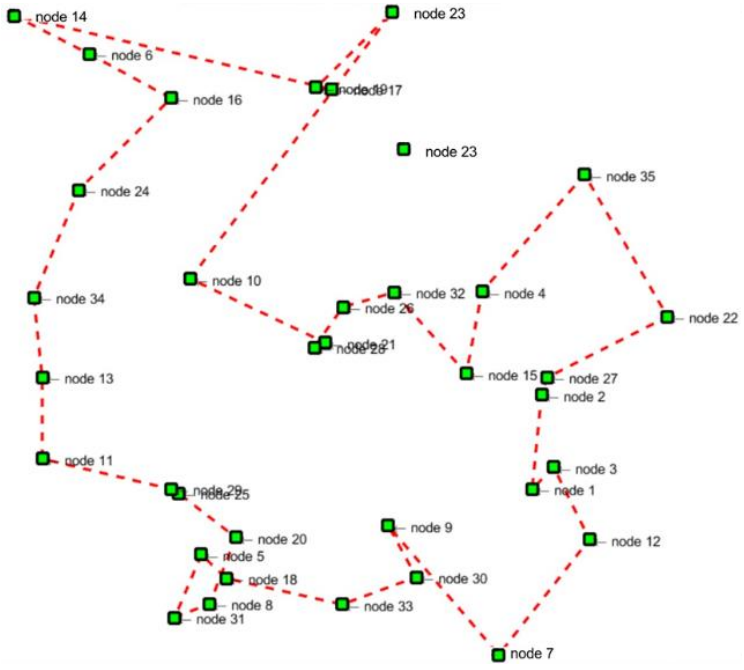

Figure 3. Route approach in case II.

Table 4. The results of case III.

The Number of Iterations	α	β	m	Total Distance (in km)
400	0.9	5	50	543.4537
400	0.9	5	70	494.852
500	0.9	5	70	487.6729
600	1	5	70	488.1252
700	0.9	5	70	481.1320
800	0.9	5	70	464.7083

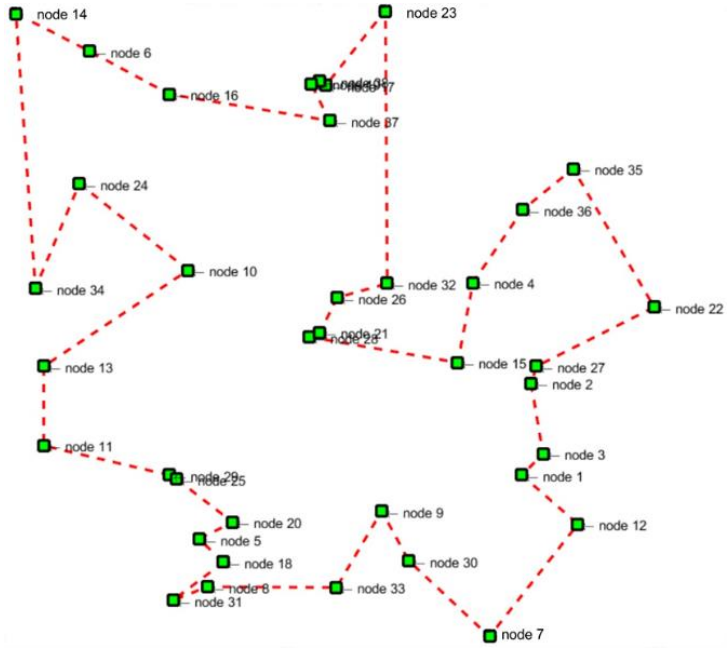

Figure 4. Route approach in case III.

3.6. Results comparison

ILP and ACO execution times are shown in Table 5. From the table, it can be seen that ACO execution time is much faster than ILP execution time. For Case I, ACO execution time is eight times faster than ILP. In Case II, ACO execution time is 31 times faster than ILP execution time. ACO execution time for Case III is 629 times faster than ILP.

Table 6 shows the total distance produced by each method. ILP method produces optimal values whereas ACO produces a value approach. In Case I, the percentage of distance difference is 1%. While in Case II and Case III have a percentage difference of 8%.

Table 5. Comparison of execution time.

Case	Execution Time (hours: minutes: seconds)	
	ILP	ACO
I	0:01:34	0:0:11.836
II	0:11:04	0:0:21.340
III	6:38:14	0:0:38.000

Table 6. Comparison of the total distance.

Case	Total Distance (in km)	
	ILP	ACO
I	388.371	392.8014
II	427.584	463.4509
III	430.251	464.7083

4. Conclusion

Based on the results of this study, it can be seen that the larger the cases that must be solved, the longer the execution time required. It can also be seen that the execution time produced by ACO is much faster than the execution time of the exact method. From the three cases, the percentage of distance difference of the first case is 1 %, while the two other cases have a percentage difference of 8%.

References

[1] F. Fadhillah, B. P. Silalahi, and M. Ilyas, "Modifikasi stepsize pada metode steepest descent dalam pengoptimuman fungsi: kasus fungsi kuadratik diagonal", *Jurnal Matematika dan Aplikasinya,* vol. 13, no. 1, pp. 47-60, Juli 2014, doi:10.29244/jmap.13.1.47-60.

[2] S. Idaman, B. P. Silalahi, and S. Guritman S, "Penyelarasan arah vektor gradien untuk menentukan step size metode steepest descent pada fungsi nonlinear kuadratik banyak variable", *Journal of Mathematics and Its Applications,* vol. 17, no. 1, pp. 47-59, Juli 2018, doi: 10.29244/jmap.17.1.47-60.

[3] B. P. Silalahi, D. Wungguli, and S. Guritman, "Steepest descent method with new step sizes", *World Academy of Science, Engineering, and Technology, International Journal of Mathematical and Computational Sciences*, vol. 9, no. 7, pp. 378- 384, 2015

[4] N. Haqueqy, B. P. Silalahi, and I. S. Sitanggang, "Uji komputasi algoritme varian metode Newton pada permasalahan optimasi nonlinear tanpa kendala", *Journal of Mathematics and Its Applications,* vol. 15, no. 2, pp. 63-76, Desember 2016, doi: 10.29244/jmap.15.2.63-76

[5] B. P. Silalahi and M. S. Dewi, "Comparison of Sensitivity Analysis on Linear Optimization Using Optimal Partition And Optimal Basis (In The Simplex Method) At Some Cases", *Published by Indonesian Mathematical Society*, pp. 82-90, April 2014.

[6] B. P. Silalahi, R. Laila, and I. S. Sitanggang, "A combination method for solving nonlinear equations", *IOP Conference Series: Materials Science and Engineering*, vol. 166, issue 1, 012011, 2017, doi:10.1088/1757-899X/166/1/012011

[7] B. P. Silalahi, Siswandi, A. Aman, "Tinjauan terhadap Metode Pengoptimuman Pendekatan Newton", *Journal of Mathematics and Its Applications*, vol. 17, no. 2, pp. 141-155, Desember 2018.

[8] M. A. Saifudin, B. P. Silalahi, and I. S. Sitanggang, "Star catalog generation for satellite attitude navigation using density-based clustering", *Journal of Computer Science*, vol. 11, no. 12, pp. 1082- 1089, 2015
[9] D. Lalang, B. P. Silalahi, and F. Bukhari, "Vehicle Routing Problem TIME Windows dengan Pengemudi Sesekali", *Journal of Mathematics and Its Applications, vol. 17, no. 2*, pp. 87-99, Desember 2018, doi: 10.29244/jmap.17.2.87-99.
[10] S. R. M. Making, B. P. Silalahi BP, and F. Bukhari, "Multi depot vehicle routing problem dengan pengemudi sesekali", *Jurnal Matematika dan Aplikasinya*, vol. 17, no. 1, pp. 75-86, Juli 2018, doi: 10.29244/jmap.17.1.75-86
[11] H. Mayyani, B. P. Silalahi BP, and A. Aman, "Frequency determination of bus rapid transit (BRT) applied on service system of trans mataram metro bus to minimize the operational cost", *International Journal of Engineering and Management Research (IJEMR)*, vol. 7, issue 6, pp. 134-140, 2017
[12] J. Heizer and B. Render, "*Operations Management 9th Edition*", Jakarta: Salemba Empat, 2010
[13] F. D. Wihartiko, A. Buono, and B. P. Silalahi, "Integer programming model for optimizing bus timetable using genetic algorithm", *IOP Conference Series: Materials Science and Engineering*. Vol. 166, 012016, 2017
[14] F. Y. Bisilisin, Y. Herdiyeni, B. P. Silalahi, "Optimasi K-Means clustering menggunakan particle swarm optimization pada sistem identifikasi tumbuhan obat berbasis citra", *Jurnal Ilmu Komputer dan Agri-Informatika*, vol. 3, no. 1, pp.37-46, 2014
[15] D. Rahmalia and A. Rohmah, "Optimisasi Perencanaan Produksi Pupuk Menggunakan Firefly Algorithm", *mantik*, vol. 4, no. 1, pp. 1-6, May 2018.
[16] D. Rahmalia and A. Rohmah, "Optimisasi Perencanaan Produksi Pupuk Menggunakan Firefly Algorithm", *mantik*, vol. 4, no. 1, pp. 1-6, May 2018.
[17] B. P. Silalahi, "Evaluation of interior-point method in Scilab", *IOP Conference Series: Earth and Environmental Science*, vol. 299, number 1, *012040*, 2019, doi:10.1088/1755-1315/299/1/012040.
[18] B. P. Silalahi, "Sharper analysis of upper bound for the iteration complexity of an interior point method using primal-dual full-Newton step algorithm", *Far East Journal of Mathematical Sciences*, vol. 95, issue 1, pp. 69-80, 2014.
[19] M. Dorigo and T. Stützlem, "The Ant Colony Optimization Metaheuristic: Algorithms, Applications, and Advances. In: Glover F., Kochenberger G.A. (eds) Handbook of Metaheuristics. *International Series in Operations Research & Management Science*, vol. 57. Springer, 2003.
[20] G. Nemhauser and L. Wolsey, "*Integer and Combinatorial Optimization"*, New York: A Wiley-Interscience, 1999
[21] M. Dorigo and L. M. Gambardella, "Ant colony system: a cooperative learning approach to the traveling salesman problem," *in* IEEE Transactions on Evolutionary Computation, vol. 1, no. 1, pp. 53-66, April 1997. doi: 10.1109/4235.585892.
[22] W. Liu, S. Li, F. Zhao, and A. Zheng, "An ant colony optimization algorithm for the Multiple Traveling Salesmen Problem," *2009 4th IEEE*

Conference on Industrial Electronics and Applications, Xi'an, 2009, pp. 1533-1537. doi: 10.1109/ICIEA.2009.5138451.

[23] M. Dorigo and K. Socha, "*An Introduction to Ant Colony Optimization"*, Belgia (BE), IRIDIA-Technical Report Series, April 2007