JAREE-Journal on Advanced Research in Electrical Engineering
Volume 1, Number 1, April 2017

Metaheuristics for The Solution of Dynamic Vehicle Routing Problem With Time Windows (DVRPTW) With Travel Time Variable

Nurlita Gamayanti [1)]
Department of Electrical Engineering
Sepuluh Nopember Institute of Technology
Surabaya, Indonesia
Email : lita@ee.its.ac.id

Abdullah Alkaff [2)]
Department of Electrical Engineering
Sepuluh Nopember Institute of Technology
Surabaya, Indonesia
Email : abdullah.alkaff@gmail.com

Ahmad Rusdiansyah [3)]
Department of Industrial Engineering
Sepuluh Nopember Institute of Technology
Surabaya, Indonesia
Email : rusdiansyah@gmail.com

Abstract—This research focuses on the development of metaheuristic algorithm to solve Dynamic Vehicle Routing Problem With Time Windows (DVRPTW) for the service provider of Inter-city Courier. The algorithm is divided into two stages which is static stage and dynamic stage. In the static stage, modified Ant Colony System is developed and in the dynamic stage, Insertion Heuristic is developed. In DVRPTW, vehicle's routes are raised dynamically based on real time information, for example the reception of new order. In order to test the performance of the proposed meta-heuristic algorithm, the authors compared the developed algorithm with the nearest neighbor algorithm and with the combination between the nearest neighbor and insertion heuristics algorithm. Experiments have been done using Chen's standard data test. The developed metaheuristic algorithm was applied on the network data of the roads in Surabaya, where the routes generated not only determine the order that the consumer must visit but also determine the routes that must be passed. Based on the results, the authors concluded that the developed algorithm generates a better travel time total than the nearest neighbor and the combination between the nearest neighbor and insertion heuristics. In addition, this algorithm can also generate route dynamically to respond to the new order well.

Keywords— Inter-City Courier, Dynamic Vehicle Routing Problem dengan Time Window (DVRPTW), Ant Colony System, Insertion Heuristics

I. Introduction

In the last few years, communication and information technology undergo a rapid development like Global Position System (GPS), Geographic Information System (GIS). These developments cause the real time information greatly affect the decision-making in vehicle routing problem. In vehicle routing problem, variety of requests is divided into 2 which is offline request, the type that come in before the route of the vehicles is built and online request, the type that come in after he vehicles already exit the depo and the route is already built. Based on the classification of these requests, vehicle routing problem is divided into static vehicle routing problem and dynamic vehicle routing problem (DVRP). In the static vehicle routing problem, all requests are categorized as offline requests while in the dynamic vehicle routing problem, besides offline request, the development of communication and information technology also enable online requests.

In this research, we solve the routing and scheduling problem in dynamic vehicle routing problem with time windows (DVRPTW) in Inter-City Courier, where vehicles leave from depo, visit a group of consumer to collect packages, and end in one of the consolidation point. It is assumed that there is communication system between dispatcher and the driver, Dispatcher can inform periodically about the next consumer that must be visited.

DVRP problems had been discussed by some researcher long ago, but the formulation of their problems is different with the problem of this research. M. Gendreau, 1996, used tabu search parallel to solve DVRP, S. Ichoua (2000) combined tabu search algorithm and diversion strategy to solve DVRP. Practically, diversion strategy is used to change the direction of vehicles from previous route to respond to the new order. We also used diversion strategy in this research.

In this research, we uses modified ant colony system to solve DVRPTW. Ant Colony Algorithm has been used to solve many kinds of combinatiorial problems such as travelling salesman problem (Dorigo and Gambardella, 1997), quadratic assigment problem (Maniezzo, 1994; Dorigo, 1996; Gambardella, 1999; Talbi, 2001), job shop schedulling (Colorni, 1994; Dorigo, 1999), sequential ordering problem (Gambardella and Dorigo, 1997). Gambardella, 1999, used multiple ant colony system to solve vehicle routing problem with time windows (VRPTW).

The developed algorithm will be tested by using Chen's standard data and will be implemented in road networks of Surabaya.

II. DVRPTW of Inter - City Courier

Inter-City Courier is a provider of delivery service of both package and documents. The operation area of Inter-City Courier is between city. In Inter-City Courier, vehicles and courier leaves from depo, visit some consumer to get their packages in certain time window that spread in many locations widely, to be gathered in one of the consolidation point that fulfill the time window and produce the least travel time. In consolidation point, the packages will be allocated based on each package destinations. And then send to each destination by the vehicles. In this research, area of the problem is restricted only for the delivery until it reaches the consolidation point, while for the package delivery to the city destination is ignored.

A. Static Problems

In static phase, all the recieved requests are offline request. In Inter-City Courier, V vehicles collect packages in N consumer that already known. All the vehicles must leave from depo and ends in one of the 4 consolidation point that exist. Each of the consolidation point has time windows. In each of the consumer node, service must be done in each of the time windows' interval. with nearest neighbor + node insertion with the optimization rate of 26,62%. Vehicles are permitted to get to the consumer node before the starting point of the time windows and must wait until the starting point of the time windows come to do service but not permitted to get to the consumer node after the deadline of the time windows. Based on the description above, the problems could be denoted as follows:

V : set of vehicles

N : set of customer nodes

C : set of consolidation points

Q : Vehicle maximum capacity

d_i : demand of node i, $i \in N$

t_{ij} : travel time of edge(i,j) includes the

w_j^k : waiting time of vehicle $k \in V$ at node

x_{ij}^k = 1 : if vehicle k visits node j immediately after node $i \in N, i \neq j$

= 0 : otherwiswe

b_j^k : the time vehicle $k \in V$ start service at node $j \in N$

e_i : the earliest start time for node $i \in N$

l_i : the latest start time for node $i \in N$

g_c : the beginning of time window for consolidation point $c \in C$

h_c : the end of time window for consolidation point $c \in C$

m_c^k : the time vehicle $k \in V$ arrive at consolidation point $c \in C$

The problem above could be formulated in the following mathematical model.

The objective function is to minimize the travel time total and waiting time in every consumer node.

$$\sum_{k \in V} \sum_{i \in N} \sum_{j \in N} t_{ij} x_{ij}^k + w_j^k \qquad (1)$$

And as for the constraints of DVRPTW explained as follows.

a. Flow Constraint

- every consumer node is visited exactly one time by one vehicle.

$$\begin{aligned} \sum_{k \in V} \sum_{i \in N} x_{ij}^k &= 1 \qquad \forall j \in N \backslash \{0\} \\ \sum_{k \in V} \sum_{j \in N} x_{ij}^k &= 1 \qquad \forall i \in N \backslash \{0\} \end{aligned} \qquad (2)$$

- every vehicles mus leave consumer node imidiatelly after visiting.

$$\sum_{i \in N} x_{ih}^k - \sum_{j \in N} x_{hj}^k = 0 \quad \forall h \in N \backslash \{0\}, \forall k \in V \qquad (3)$$

- every vehicles must leave from depo

$$\sum_{i \in N \backslash \{0\}} x_{i0}^k = 1 \qquad \forall k \in V \qquad (4)$$

- every vehicles must end in the same consolidation point

$$\sum_{i \in N \backslash \{0,c\}} x_{ic}^k = 1 \qquad \forall k \in V \qquad (5)$$

b.Capacity Constraint

- the amount of the package by every vehicle cannot exceed the amounts of vehicles.

$$\sum_{i \in N} d_i \sum_{j \in N} x_{ij}^k \leq Q \qquad \forall k \in V \qquad (6)$$

c. Time Windows Constraint

- vehicles k cannot go into node j before time $b_i^k + t_{ij}$ if passes from node i to node j

$$x_{ij}^k \left(b_i^k + t_{ij}\right) \leq b_j^k \qquad \forall i \in N, \forall j \in N, \forall k \in V \qquad (7)$$

$$b_j^k = max\left\{e_j, b_i^k + t_{ij}\right\} \qquad (8)$$

- time to start the service in every consumer node by every vehicle must be in the interval time window each node

$$e_i \leq b_i^k \leq l_i \qquad \forall i \in N, \forall k \in V \qquad (9)$$

- Every vehicles must get in the consolidation point in their interval time window.

$$g_c \le m_c^{\ k} \le h_c \qquad \forall c \in C, \forall k \in V \tag{10}$$

B. Dynamic Problems

In the description of the static problem, route and scheduling that has been made consist of all the known requests (before the vehicles leaves depo). This kind of problem in known as Static Vehicle Routing Problem. But in reality, consumer request could be happened if the vehicles have already leaves depo and route is already be determined (online request), so the route must be updated to respond to the new request.
This kind of problem is known as Dynamic Vehicle Routing Problem. This problem can be illustrated like in Fig 1.

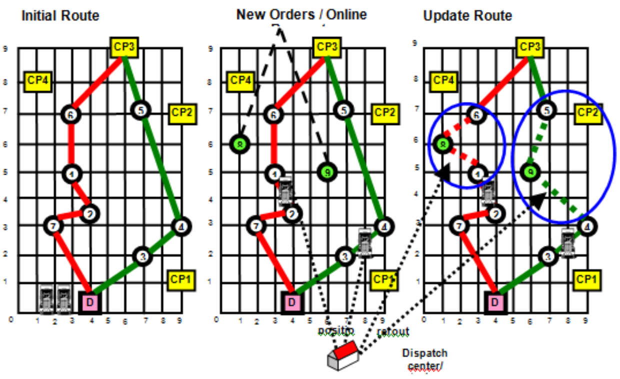

Fig.1. DVRPTW of Intercity Courier

III. Metaheuristics Algorithm

In this research, the development of the algorithm is divided into 2 stages which is algorithm to solve static problems and algorithm to solve dynamic problems.

C. Algorithm to Solve Static Problems

In the static phase, to get the route with the least total of travel time and waiting time we used modified ant colony system algorithm, of improved ant colony system (IACS). IACS algorithm is an ACS algorithm (Dorigo & Gambardella, 1999) that has already been through fixes in route construction rule and pheromone update rule.
Procedure of IACS algorithm is explained as follows :

Step 1 :
Set the parameter

Step 2 :
Generate InitialSolution using Nearest Neighbor

Step 3 :
Apply Insertion Heuristics on InitialSolution and saved as 1stSolution.
Iteration= 1, Ant = 2

Step 4 :
Create solution based on state transition rule and dolocal pheromone update on every arcs which has been passed.
Ant = Ant + 1

Step 5 :
If Ant > MaxAnt then Ant = 2 and do step 6.
If Ant <= MaxAnt then do step 4

Step 6 :
Sort the second solution until JumlahMaxAnt solution.
Do insertion heuristics on the best solution and saved as 2ndSolution

Step 7 :
Do Global Pheromone Update on arcs that create 1stSolution and 2ndSolution

Step 8 :
Compare the 1stSolution and the 2ndSolution, save the best solution as TheBestSolution.
In the next iteration, 1stSolution = TheBestSolution.
Iterasi = Iterasi + 1

Step 9 :
Stop.

Pheromone Initialization

Initialization of pheromone in every arc is :

$$\tau_0 = (N\,x\,L_{NN})^{-1} \tag{11}$$

Where N is the amount of node and L_{NN} is travel time total from the result of Nearest Neighbor.

State Transition Rule

Every ant in node I will choose node j arc (i,j) based on state transition rule as follows :

- If $q \le q_0$ (Exploitation)

$$j := \underset{j \in S'(i)}{arg\ max} \left\{ (\tau_{ij})^{\alpha} (\eta_{ij})^{\beta} \right\} \tag{12}$$

- If $q > q_0$ (Exploration)

$$P_{ij} = \frac{(\tau_{ij})^{\alpha}(\eta_{ij})^{\beta}}{\sum\limits_{j \in S'(i)} (\tau_{Cj})^{\alpha}(\eta_{Cj})^{\beta}} \quad , \ j \in S^{'}(i) \tag{13}$$

where

$$\eta_{ij} = \frac{1}{t_{ij} + w_j^k} \tag{14}$$

If (capacity requirement) and (time windows requirement) is not fulfilled then $\eta_{ij} = 0$.

Local Pheromone Update

Every time passing the arc(i,j), ants will update the amount of their pheromone in arc(i,j) based on Local Pheromone Update :

$$\tau_{ij}^{\ new} = \tau_{ij}^{\ old} + (1-\rho)\tau_0 \tag{15}$$

Global Pheromone Update

After all the ants had already create their own solution, then the arcs that create 1st Solution and 2nd Solution in every iteration will change the amount of pheromone based on Global Pheromone Update :

$$\tau_{ij}^{\ new} = (1-\gamma)\tau_{ij}^{\ old} + \gamma\Delta\tau_{ij} \tag{16}$$

where $(i, j) \in$ tour in the 1st Solution and 2nd Solution

$$\Delta\tau_{ij} = \frac{(A-B)+(A-C)}{A} \qquad (17)$$

Where A is the third best travel time total in every iteration, B is the overall best travel time total and C is the best travel time total in every iteration

D. Algorithm to solve dynamic problems

The procedure of the dynamic problems algorithm is as follows:

Step 1 :

Get the position of the vehicles when online request is accepted.

Step 2 :

Identified each of the online request service time.

Slip the online request to the initial route by considering the time window constraint and capacity constraint. If not possible to slip online request node in the initial route, then generate a new route.

Step 3 :

Applied insertion heuristic node so that the route could be obtained with minimum travel time total.

IV. Experimental Results

IACS algorithm will be implemented on the road networks of Surabaya. In this research, to test the performances of IACS algorithm, IACS will be compared to the nearest neighbor algorithm and nearest neighbor + node insertion heuristic algorithm. IACS parameters that will be used are: $\alpha = 1$, $\beta = 1$, $\rho = 0{,}1$, $q_0 = 0{,}5$, the amount of ant = 5, maximum number of iteration = 50.

Table 1 and table 2 each show the comparison of travel time total and the amount of vehicles produced by IACS with the nearest neighbor and nearest neighbor + node insertion in static phase for 15 times experiments using Chen's standard data.

TABLE 1. Comparison of total travel time IACS, nearest neighbor, and nearest neighbor + node insertion in static phase using Chen's standard data

No	Number of off.req	Total Travel Time (s)			IACS optimation (%)	
		IACS	NN+NI	NN	NN	NN+NI
1	6	24648	41592	33018	40,74	25,35
2	8	31794	44478	37980	28,52	16,29
3	8	31740	44718	38100	29,02	16,69
4	10	38784	50430	41304	23,10	6,10
5	10	24750	47418	44040	47,80	43,80
6	12	24894	64578	54642	61,45	54,44
7	12	25812	42852	35412	39,76	27,11
8	14	37590	44562	41442	15,65	9,30
9	14	37968	63672	51456	40,37	26,21
10	16	39126	78900	69288	50,42	43,53
11	16	44088	79062	67032	44,24	34,23
12	18	46974	56430	54174	16,76	13,30
13	18	41484	61314	53562	32,34	22,55
14	20	43344	80568	73650	46,20	41,15
15	20	45702	63072	56598	27,54	19,25

Based on table 1 we can see that IACS algorithm produced a minimum travel time total in static phase compared to nearest neighbor with the average optimization rate of 36,26% and also with nearest neighbor + node insertion with the optimization rate of 26,62%.

TABLE 2. Comparison of number of vehicles IACS, nearest neighbor, and nearest neighbor + node insertion in static phase using Chen's data standar

No	Number of Online Req	Number of Vehicles		
		IACS	NN+NI	NN
1	6	2	4	3
2	8	3	5	5
3	8	3	5	4
4	10	4	5	4
5	10	2	5	4
6	12	3	9	8
7	12	3	5	4
8	14	4	5	4
9	14	4	7	6
10	16	4	9	9
11	16	4	8	7
12	18	5	7	6
13	18	5	8	7
14	20	5	11	10
15	20	5	8	7

Based on table 2 we can see that IACS algorithm produced the least amounts of vehicles compared to nearest neighbor and nearest neighbor + node insertion in dynamic phase.

Table 3 and table 4 each show the comparison of travel time total and the amounts of vehicles produced by IACS compared with nearest neighbor and nearest neighbor + node insertion in dynamic phase for 15 times experiments using Chen's standard data.

TABLE 3. Comparison of IACS travel time total, nearest neighbor, and nearest neighbor + node insertion in dynamic phase using Chen's standard data

No	Number of off.req	Total Travel Time (s)			IACS optimation (%)	
		IACS	NN+NI	NN	NN	NN+NI
1	2	24828	41772	33198	40,56	25,35
2	2	32280	44478	44592	27,42	16,29
3	3	42528	55410	48792	23,25	16,69
4	2	44796	59992	50796	25,33	6,10
5	3	35076	54780	48006	25,97	43,80
6	2	28872	64578	57312	55,29	54,44
7	3	39300	47112	47100	16,58	27,11
8	2	39762	54054	50934	26,44	9,30
9	4	48294	65826	54660	26,63	26,21
10	3	50718	83748	76674	39,44	43,53
11	4	46674	79080	67050	40,98	34,23
12	3	54552	63174	60918	13,65	13,30
13	5	55188	69618	61974	20,73	22,55
14	3	55314	83190	76272	33,51	41,15
15	5	59526	71838	65364	17,14	19,25

Based on table 3 we can see that IACS algorithm produced a minimum travel time total in dynamic phase compared to nearest neighbor with the average optimization rate of 33,86% and also with nearest neighbor + node insertion with the optimization rate of 26,62%.

TABLE 4. Comparison of IACS vehicles, nearest neighbor, and nearest neighbor + node insertion in dynamic phase with the same offline request as table 2 with Chen's data.

No	Number of Online Req	Number of Vehicles		
		IACS	NN+NI	NN
1	2	2	4	3
2	2	3	5	6
3	3	4	6	5
4	2	5	6	5
5	3	3	5	5
6	2	3	9	8
7	3	5	5	5
8	2	4	6	5
9	4	5	7	6
10	3	5	10	9
11	4	5	8	7
12	3	6	7	7
13	5	7	10	9
14	3	6	11	10
15	5	6	8	7

After metaheuristics algorithm is tested in Chen's standard data, the next step is to implement the metaheuristics algorithm on the networks data of the roads in Surabaya.

TABLE 5. Comparison of IACS travel time total, nearest neighbor, and nearest neighbor + node insertion in static phase using the networks data of Surabaya's road

No	Number of off.req	Total Travel Time (s)			IACS optimation (%)	
		IACS	NN+NI	NN	NN	NN+NI
1	10	6635	13197	33198	49,72	36,40
2	30	24838	36048	44592	31,10	28,43
3	50	36049	50981	48792	29,29	26,17

Based on table 5 we can see that IACS algorithm produced a faster travel time total compared to the nearest neighbor on static phase with the average optimization rate of 33,8% and with nearest neighbor + node insertion with the optimization rate of 26,62%.

TABLE 6. Comparison of IACS vehicles, nearest neighbor, and nearest neighbor + node insertion in dynamic phase

No	Number of Online Req	Number of Vehicles		
		IACS	NN+NI	NN
1	10	2	2	2
2	30	3	4	4
3	50	4	5	5

Based on table 6 we can see that IACS algorithm produced the least amount of vehicles compared to nearest neighbor and nearest neighbor + node insertion in static phase.

TABLE 7. Comparison of IACS travel time total, nearest neighbor, and nearest neighbor + node insertion in dynamic phase using the networks data of Surabaya's road

No	Number of Online.Req	Total Travel Time (s)			IACS optimation (%)	
		IACS	NN+NI	NN	NN	NN+NI
1	2	10987	13197	17198	36,11	16,75
2	2	27178	32012	34192	20,51	15,10
3	3	50129	54388	56792	11,73	7,83

Based on table 7 we can see that IACS algorithm produced a faster travel time total in dynamic phase compared to nearest neighbor with the average optimization rate of 22,78% an also with nearest neighbor + node insertion with the optimization of 13,23%

TABLE 8. Comparison of IACS vehicles, nearest neighbor, and nearest neighbor + node insertion in dynamic phase

No	Number of Online Req	Number of Vehicles		
		IACS	NN+NI	NN
1	2	2	4	3
2	2	3	5	6
3	3	4	6	5

Based on table 8 we can see that IACS algorithm produced the least amounts of vehicles compared to nearest neighbor and nearest neighbor + node insertion in dynamic phase.

V. Conclusions

Based on the results, the developed algorithm produced a better travel time total in dynamic phase compared to nearest neighbor with the average optimization rate of 22,78% an also with nearest neighbor + node insertion with the average optimization rate of 13,23%. In addition, the developed algorithm produced the least amounts of vehicles compared to nearest neighbor and nearest neighbor + node insertion in dynamic phase.

References

[1] Champbell,A.M., Martin S (2004), "Efficient Insertion Heuristics for Vehicle Routing and Schdulling Problems".

[2] Chen, C.H., Ting, C.J. (2004), " An Improved Ant Colony Systems Algorithm for The Vehicle Routing Problem", Working Paper 2004-001, Department of Industrial Engineering and Management, Yuan Ze University, Taiwan.

[3] Chen, C.H., Ting, C.J. (2005), "A Hybrid Ant Colony System for Vehicle Routing Problem with Time Windows," Journal of the Eastern Asia Society for Transportation Studies, Vol.6, pp.2822-2836.

[4] Donati, Gambardella,M., Rizzoli, A.E.,Casagrande,N.,Montemanni, R. (2002), "Time Dependent Vehicle Routing Problem with an Ant Colony System," Instituto Dalle Molle di Studi sull'Intelligenza Artificiale (IDSIA).

[5] Gambardella,L.M., Dorigo,M. (2000), " An Ant Colony System Hybridized with a New Local Search for the Sequential Ordering Problem," INFORMS Journal on Computing Vol.12, No.3.

[6] Gambardella, L.M, Tai, E., Agazzi, G.,"A Multiple Ant Colony System for Vehicle Routing Problem with Time Window", McGraw Hill, London

[7] Gendreau, M., Guertin, F., Potvin, J., Seguin, " Neighborhood Search Heuristic for a Dynamic Vehicle Dispatching Problem with Pick-Up and Delivery", Universite de Montreal

[8] Gendreau, M., Guertin, F., Potvin, J., Taillard, E. (1999) ,"Parallel Tabu Search for Real Time Vehicle Routing and Dispatching", Transportation Sience, 33(4):381-390

[9] Ichoua, S., Gendreau, M., Potvin, J. (2000),"Diversion Issues in Real Time Vehicle Dispatching", Transportation Science, 34(4):426-438

[11] Kim,S., Lewis,M.E., Chelsea C.White III, " Dynamic Vehicle Routing with Real-Time Traffic Information," In Proceedings of the 10th World Congress on Intelligent Transport Systems, Madrid, Spain.

[12] Kim,S., Lewis,M.E., Chelsea C.White III, " Dynamic Vehicle Routing with Real-Time Traffic Information," IEEE Transactions on Intelligent Transportation Systems, Vol.6, No.2.

[13] Montemanni, R.,Gambardella, L.M., Rizolli, A.E., Donati, A.V, "Instituto Dalle Molle Studi sull'Intelligenza Artificiale (IDSIA), "A New Algorithm for a Dynamic Vehicle Routing Problem based on Ant Colony System"

[14] Renaud, J., Boctor, F.F., Ouennice,J., (1998), "A Heuristics for The Pick Up and Delivery Traveling Salesman Problem.

[15] Uchimura, K., Takahashi, H., Saitoh, T. (2002), "Demand Responsive Services in Hierarchical Public Transportation System",IEEE Transactions on Vehicular Technology, Vol. 51, No. 4, pp 1-6