Dynamic pickup and delivery problem with transfer in ridesharing to reduce congestion

Mela Ayu Novia Andini\textsuperscript{1, a}, Yudi Satria\textsuperscript{2, b}, Helen Burhan\textsuperscript{3, c}

\textsuperscript{1, 2, 3}Department of Mathematics and Natural Science, Universitas Indonesia, Depok, West Java, 16424, Indonesia

E-mail: \textsuperscript{a}mela.ayu@sci.ui.ac.id, \textsuperscript{b}satria@sci.ui.ac.id, \textsuperscript{c}helen.burhan@sci.ui.ac.id

Abstract. Pickup and delivery problem with transfer (PDPT) consists of defining a set of optimum routes between a set of pickup points and delivery points of the request. During travel in the vehicle, the passenger may change vehicle in a fixed transfer point. Transfer passenger occurs when a passenger’s delivery in a vehicle results in long travel route, while there are other vehicles which may take the passenger on a shorter travel route. Based on research conducted by Cortés et al (2010), this paper considers PDPT where customer location and service time are random variables that are realized dynamically during the plan execution. In this paper will use the insertion heuristic method to obtain the optimal solution of Dynamic PDPT. The objective solution is to minimize the operational cost of the transportation company by reducing the travel distance of each vehicle. This problem will be discussed with ridesharing approach to improve existing ridesharing services by using transfer point. Based on the result of simulation experiment obtained Dynamic PDPT can be used to optimize the existing ridesharing system that can fulfill more requests for 16.7% followed by increasing income.

1. Introduction
The development of the transportation system is currently growing rapidly. This resulted in an increase in the volume of vehicles on the road which exacerbated congestion. Various solutions have been offered to tackle this congestion problem, which aims to reduce the use of private vehicles, such as increasing public transportation, both mass — such as buses, trains, and most recently the MRT and LRT, as well as private-public transportation such as taxis, “ojek” online, and taxi online.

One way to reduce congestion is using a ridesharing system on private vehicle, whereas “Ridesharing is a joint-trip of more than two participants that share a vehicle and requires coordination with respect to itineraries” (Furuhatu et al. 2013) [1]. This paper will discuss ridesharing with a transfer based on research conducted by Cortés et al (2010) [2], where the difference lies in the dynamic case and uses insertion heuristic in solving the problem.

2. Literature review
2.1. Ridesharing
“Ridesharing is a joint-trip of more than two participants that share a vehicle and requires coordination with respect to itineraries” (Furuhatu et al. 2013) [1]. The foundation of the ridesharing theory is based on Dial A Ride Problem (DARP). DARP is one of the cases of Pickup and Delivery Problems (PDP) with the passenger object being a person. The PDP can be conceptually described as finding the
optimal way of assigning a set of transportation requests to vehicles (initially located at several depots), by minimizing a specific purpose objective function, subject to a variety of constraints. The objective function may include components such as operational costs, number of vehicles, customer’s level of service, and so on (Cortés et al. 2010) [2]. Dial-A-Ride-Problem (DARP) consists of defining the set of routes with a minimum cost to meet the set of transportation requests. Each request involves transporting the user from the original set (pickup point) to the destination set (delivery point) (Renaud Masson et al., 2013) [3]. In this paper, we will focus on PDP which is a generalization of DARP.

One of the problems in the ridesharing practice is finding the right combination of several requests that meet the criteria, one of them is time windows. This constraint requires that each request is served when or after the earliest time of the request and before the latest time of the request. Sometimes the solution obtained is not optimal, it can be a long route or an inefficient route which results in a lot of costs incurred by the company (driver). Based on these problems and research conducted by Cortés et al (2010) [2], this study offers Dynamic Pickup and Delivery Problem with Transfer in ridesharing to reduce congestion. By using a transfer point, users can change vehicles (transfer) at a transfer point. The use of transfer points is expected to optimize all requests that can be served, minimize travel distance and time, and minimize the travel costs incurred by the transportation company (driver). The problems will be reviewed dynamically (real-time), location and service time are random variables that are realized dynamically during the plan execution.

In this study, we will examine the efficiency of using transfer points (PDPT) in dynamic (real-time) cases. So that if this solution is proven to be efficient, then this solution can be applied not only for online transportation but also for other public transportation, such as school buses, travel, and other shuttle modes which are expected to be one solution to congestion.

2.2 Insertion Heuristic
To solve the problem in this paper, the insertion heuristic method will be used. Insertion heuristic is commonly used for vehicle routing problems because they are efficient, easy to implement and produce good results (Wang et al., 2016) [4]. The objective is to improve the routing quality by reducing passenger ride time which has an impact on reducing travel cost (Wang et al., 2016) [4].

The insertion heuristic is starting with selecting the first customer to enter the route, the selection of the first customer to enter the route is seen based on distance or customers who must be served immediately (Joubert & Claassen, 2006) [5], and then entering other customers who have not entered the route repeatedly until all customers enter the route.

According to Solomon (1987) [6], there are three ways to calculate the insertion cost, and in this paper will use the third formula, as follows:

\[ c_{11}(i, u, j) = d_{iu} + d_{uj} - d_{uj}, \mu \geq 0 \]  
\[ c_{12}(i, u, j) = v_{ju} - v_{j} \]  
\[ c_{1}(i, u, j) = \alpha_{1}c_{11}(i, u, j) + \alpha_{2}c_{12}(i, u, j), \alpha_{1} + \alpha_{2} = 1, \alpha_{1} \geq 0, \alpha_{2} \geq 0 \]  
\[ c_{2}(i, u, j) = \beta_{1}R_{d}(u) + \beta_{2}R_{c}(u), \beta_{1} + \beta_{2} = 1, \beta_{1} \geq 0, \beta_{2} > 0 \]

where \( c_{1}(i, u, j) \) is used to compute request \( u \)’s best feasible insertion place in the route be. \( R_{d}(u) \) is the total route distance and \( R_{c}(u) \) is the total travel time.

However, the method above obviously uses for a static case which all request has been known before finding the route. This paper will be a dynamic case which the request is real-time. So, the way to insert the request is based on the time request. The case will be seen dynamically per minute.

3. Formulation of the problem
The main objective of dynamic PDPT is minimization the transportation cost by minimizing the travel time but still satisfy all the constraints. For PDPT in static case we can define \( G = (V, A) \) be a complete graph where \( V = P \cup D \cup O \cup O' \cup T \) is set of vertex or node and \( A = \{(i, j)|i, j \in V, i \neq j\} \) is set of all arch [ronald masson]. Since \( G \) is a weighted graph, there is travel time from node \( i \) to node \( j \) which is denoted by \( t_{ij} \). For the complete mathematical formulation of PDPT which can be found in [2], Cortész et al (2010) define 31 constraints for PDPT, we added dynamic constraint
because we provide dynamic (real-time) case, and the passenger is people, so we added maximum ride time and route time. The objective function is to minimize the total ride time spent by the vehicle [2].

\[
\text{Min } \sum_{k \in M} \sum_{(i,j) \in E} t_{ij}x_{ij}^k
\]  

Subject to:
\[
\sum_{i \in V^+(k^+)} x_{k^+i}^k = 1 \quad \forall k \in M
\] (6)
\[
\sum_{i \in V^-(k^-)} x_{ki}^k = 1 \quad \forall k \in M
\] (7)
\[
\sum_{m \in M^+} - \sum_{i \in V^+(m)} x_{mi}^k \leq 1 \quad \forall k \in M
\] (8)
\[
\sum_{j \in V^+(i)} x_{ij}^k - \sum_{j \in V^-(i)} x_{ji}^k = 1 \quad \forall i \in N, \forall k \in M
\] (9)
\[
\sum_{i \in V^-(s(r))} x_{s(r)}^k = x_{s(r)f(r)}^r \quad \forall r \in T, \forall k \in M
\] (10)
\[
\sum_{j \in V^+(f(r))} x_{f(r)j}^k = x_{s(r)f(r)}^r \quad \forall r \in T, \forall k \in M
\] (11)
\[
\sum_{k \in M} \sum_{i \in V^-(i^-)} x_{ji}^{k-1} = 1 \quad \forall i \in C
\] (12)
\[
\sum_{k \in M} - \sum_{j \in V^+(i^+)} x_{ji}^{k+} = 1 \quad \forall i \in C
\] (13)
\[
x_{k^+i}^k = 1 \Rightarrow t_{k^+i}^k \leq D_{k^+} \quad \forall k \in M, \forall i \in C
\] (14)
\[
x_{k^+s(r)}^k = 1 \Rightarrow t_{k^+s(r)} \leq D_{k^+} \quad \forall k \in M, \forall i \in C
\] (15)
\[
x_{ij}^k = 1 \Rightarrow D_i + t_{ij} \leq D_j \quad \forall k \in M, \forall (i, j) \in N^2 \cap E
\] (16)
\[
x_{k^+(r)}^k = 1 \Rightarrow D_{k^+(r)} + t_{k^+(r)} \leq D_{k^+(r)} \quad \forall k \in M, \forall i \in N, \forall r \in T
\] (17)
\[
x_{k^+(r)f(r)}^k = 1 \Rightarrow D_{k^+(r)} + t_{s(r)f(r)} \leq D_{k^+(r)} \quad \forall k \in M, \forall r \in T
\] (18)
\[
x_{k^+(r)j}^k = 1 \Rightarrow D_{k^+(r)} + t_{f(r)j} \leq D_{k^+(r)} \quad \forall k \in M, \forall j \in N, \forall r \in T
\] (19)
\[
x_{k^+(r)s(u)}^k = 1 \Rightarrow D_{k^+(r)} + t_{f(r)s(u)} \leq D_{k^+(r)} \quad \forall k \in M, \forall r \in T, \forall u \in T \setminus \{r\}
\] (20)

\[D_{k^+} : \text{Time at which node } i \in N \text{ is attended}
\] (21)
\[D_{k^+(r)} : \text{Time at which vehicle } k \in M \text{ arrives at transfer } r \in T
\] (22)
\[D_{k^+(r)s(u)} : \text{Time at which vehicle } k \in M \text{ leaves transfer } r \in T
\] (23)

**IOP Publishing**

International Conference on Mathematics: Pure, Applied and Computation

IOP Conf. Series: Journal of Physics: Conf. Series 1218 (2019) 012010
doi:10.1088/1742-6596/1218/1/012010
\[ z^k_i + z^v_i = 2 \Rightarrow D^s_{s(r)} + \Gamma \leq D^v_{f(r)} \quad \forall k, v \in M, \forall r \in T, \forall i \in C \]  

(36)

Table 1. List index

| Object | Meaning | Comment |
|--------|---------|---------|
| M      | Set of vehicles |         |
| C      | Set of requests |         |
| T      | Set of transfer points |         |
| M⁺     | Set of origin depots for vehicles | \( M⁺ = \{ k⁺ : k \in M \} \) |
| M⁻     | Set of destination depots for vehicles | \( M⁻ = \{ k⁻ : k \in M \} \) |
| N⁺     | Set of origin nodes for vehicles | \( N⁺ = \{ i⁺ : i \in C \} \) |
| N⁻     | Set of destination nodes for vehicles | \( N⁻ = \{ i⁻ : i \in C \} \) |
| N      | Set of nodes associated with requests | \( N = N⁺ \cup N⁻ \) |
| s(r)   | Start node of transfer \( r \in T \) |         |
| f(r)   | Finish node of transfer \( r \in T \) |         |
| s(T)   | Set of start nodes of transfers | \( s(T) = \{ s(r) : r \in T \} \) |
| f(T)   | Set of finish nodes of transfers | \( f(T) = \{ f(r) : r \in T \} \) |
| V      | Set of nodes | \( V = M⁺ \cup M⁻ \cup N \cup s(T) \cup f(T) \) |
| qᵢ     | Size of request \( i \in C \) |         |
| Qₖ     | Capacity of vehicle \( k \in K \) |         |
| tᵢⱼ    | Minimum ride time from node \( i \) to node \( j \) |         |

According to Solomon (1987) [6], to calculate the insertion cost, it can be done using formula from Solomon (1987) [6].

4. Example of dynamic PDPT

We created not to scale graph, that represented 23 node locations with the following linkage:

![Example of dynamic PDPT](image-url)
Table 2. List Order.

| Order | Capacity | Node Pickup | Node Delivery | Time Windows Pickups |
|-------|----------|-------------|---------------|----------------------|
| o1    | 1        | N3          | N8            | 1 64.5               |
| o2    | 2        | N6          | N11           | 4 55                 |
| o3    | 1        | N4          | N9            | 5 65                 |
| o4    | 1        | N18         | N10           | 23 80                |
| o5    | 1        | N7          | N12           | 46 59                |

Table 2 gives information about the detail of the order’s information that consist of order, capacity, node pickup, node delivery, time windows for earliest pickup and latest arrival (time windows in minutes).

Table 3. Order Flow Per Minute.

| Time (minute) | Order | Node Capacity | Node Capacity | Node Capacity | Node Capacity | Node Capacity |
|---------------|-------|---------------|---------------|---------------|---------------|---------------|
| 0             | o1    | N1            | 0             | N2            | 0             | ping          |
| 4             | o2    | N3            | 0             | N2            | 0             | ping          |
| 5             | o3    | N3,4          | 1             | N2-6          | 0             | ping          |
| 6             |       | N3,4          | 1             | N6            | 1             | v1            |
| 10            |       | N4            | 2             | N6,21         | 1             | v2            |
| 13            |       | N5            | 3             | N21,N15       | 1             | v1            |
| 18            |       | N15           | 2             | N21,N15       | 1             |               |
| 23            | o4    | N16,19        | 2             | N16           | 2             | ping          |
| 28            |       | N10           | 2             | N11           | 1             |               |
| 36            |       | N18           | 0             | N15,N16       | 2             | v2            |
| 37            |       | N18           | 2             | N16,N18       | 0             | v1            |
| 44            |       | N10           | 1             | N11,7         | 1             | done          |
| 46            | o5    | N8            | 0             | N7            | 1             | ping          |
| 50            |       | N19,N10       | 0             | N7            | 2             | v2            |
| 53            |       | N8,13         | 0             | N7            | 2             | done          |
Table 3 gives information about the detail of order flow per minute that consist of time, order, node and capacity of vehicle 1 (v1), node and capacity of vehicle 2 (v2), and the flow for each order.

| Order | Pickup | Transfer | Arrival | Private | Delay (%) |
|-------|--------|----------|---------|---------|-----------|
| o1    | 6      | no       | 58      | 51      | 14%       |
| o2    | 10     | no       | 44      | 40      | 10%       |
| o3    | 13     | v1 to v2 | 62      | 47      | 32%       |
| o4    | 37     | no       | 53      | 41      | 29%       |
| o5    | 50     | no       | 57      | 46      | 24%       |
| average |        |          |         |         | 22%       |

Table 4 gives information about the detail of summary order that give information about delay between private and sharing system.

Table 5. Vehicle Route with Transfer

| Vehicle | Route with Transfer | Time with Transfer | Info |
|---------|---------------------|--------------------|------|
| V1      | N1-N3-N4-N15-N16-N18-N10-N8-N13 | 61                | o1, o3, o4(dropping o3 at N15) |
| V2      | N2-N6-N16-N11-N7-N12-N9-N14 | 66                | o3, o2, o5 (picking up o3 at N16) |
Table 6. Vehicle Route without Transfer

| Vehicle | Route                | Time | Info |
|---------|----------------------|------|------|
| V1      | N1, N3, N18, N10, N8, N13 | 55   | o1, o4 |
| V2      | N2, N6, N11, N7, N12, N14 | 58   | o2, o5 |

The delay in vehicle 1 (v1) with transfer and without transfer is 11%, and for V2 is 14%. The order/request that can be fulfilled if using transfer is five requests with total 6 passengers, but without transfer only 4 order that can be fulfilled with a total 5 passengers. We apply a flat rate of Rp15.000 for each passenger fee, so total revenue we get from using transfer point is Rp90.000, and without transfer is Rp75.000.

5. Conclusion
Dynamic Pickup and delivery problem with transfer can be used to optimize the existing ridesharing system can fulfill more requests for 16.7% followed by increasing income. These results are expected to improve the existing ridesharing system and furthermore can be a solution to traffic congestion.

Acknowledgment
This research was supported by Publikasi Internasional Terindeks untuk Tugas Akhir Mahasiswa (PITTA) research grant of Universitas Indonesia 2018.

Reference
[1] Furuhatat al. (2013). Ridesharing: The state-of-the-art and future directions. Transportation Research Part B 57 (2013) 28–46.
[2] Cortés CE, Matamala M, and Contardo C (2010) The Pickup and Delivery Problem with Transfer: Formulation and a Branch and Cut Solution Method (Journal of Operational Research) pp 711-724.
[3] Renaud M, Fabien L, and Olivier P (2014) The Dial a Ride Problem with Transfer (Computers & Operations Research41) pp 12–23.
[4] Wang X, Dessouky M, and Ordonez Fernando. (2016). A Pickup and Delivery Problem for Ridesharing Considering Congestion. Transportation Letters, 8(5), 259-269.
[5] Joubert JW, Claasen SJ. (2006). A sequential insertion heuristic for the initial solution to a constrained vehicle routing problem. Volume 22 (1), pp. 105–116.
[6] Solomon M. M. (1987). Algorithms for the Vehicle Routing and Scheduling Problems with Time Window Constraints. Operations Research 35(2), 254-265. Maryland: INFORMS.