Data & Knowledge Engineering 114 (2018) 67–85

Contents lists available at ScienceDirect

Data & Knowledge Engineering

journal homepage: www.elsevier.com/locate/datak

A time-dependent model with speed windows for share-a-ride problems: A case study for Tokyo transportation

Phan-Thuan Do*, Nguyen-Viet-Dung Nghiem, Ngoc-Quang Nguyen, Quang-Dung Pham

School of Information and Communication Technology Hanoi University of Science and Technology, 01 Dai Co Viet Road, Hanoi, Vietnam

ARTICLE INFO

Keywords:
Passenger and parcels sharing
Share-a-ride
Tokyo Taxi
Dynamic graphs
Heuristic algorithms

ABSTRACT

This paper introduces a new fully time-dependent model of a public transportation system in the urban context that allows sharing a taxi between one passenger and parcels with speed widows consideration. The model contains many real-life case features and is presented by a mathematical formulation. We study both static and dynamic scenarios in comparison to traditional strategies, i.e., the direct delivery model. Moreover, we classify speed windows by different zones and congestion levels during a day in the urban context. Different speed windows induce the dynamic graph model for road networks and make the problem much more difficult to solve. Because of the complex model, the preprocessing steps on data as well as on dynamic graphs are very important. We use a greedy algorithm to initiate the solution and then use some local search techniques to improve the solution quality. The experimental data set is recorded by Tokyo-Musen Taxi company. The data set includes more than 20000 requests per day, more than 4500 used taxis per day and more than 130000 crossing points on the Tokyo map. Experimental results are analyzed on various factors such as the total benefit, the accumulating traveling time during the day, the number of used taxis and the number of shared requests.

1. Introduction

In common transportation systems, passengers and parcels requests are always served separately because of their different characteristics. Everyday, there are a huge number of empty trips driving around to pickup passengers and parcels. This significantly increases the operation costs and causes many social problems such as traffic jams and the environmental pollution. In the urban context, parcels usually have small dimensions and low weights so that they can be transferred along with passengers without significantly affecting the private space and the comfortability. On the contrary, passengers are willing to share a taxi with parcels if both the comfortability and the security are guaranteed. These problems have been intensively studying over the last few years under the share-a-ride problems.

One of the biggest challenges while considering the share-a-ride problem between passengers is the privacy, comfortability and the security of passengers. Usually, passengers have different requirements and objectives such as the traveling time, the private space, etc. These induce the model of sharing between different passengers inapplicable in the real-life situation. Therefore, we propose an applicable share-a-ride model between a unique passenger and parcels. In respect of setting the highest priority for the passenger and lower priorities for parcels, in this model, the passenger is always delivered directly without any interruption for other pickup or delivery stops. Thereby, for a scheduled parcel, the following scenarios are permitted: (1) collected and dropped before

* Corresponding author.
E-mail address: thuan.dophan@hust.edu.vn (P.-T. Do).

http://dx.doi.org/10.1016/j.datak.2017.06.002
Received 25 January 2017; Received in revised form 1 June 2017; Accepted 12 June 2017
Available online 15 June 2017

picking the passenger up; (2) collected and dropped after delivering the passenger; (3) collected before picking the passenger up and dropped after delivering the passenger.

In the research community, our problem is classified as a Dial-A-Ride problem (DARP for short) [1,2] which is a generalization of the Vehicle Routing Problem family [3–8]. The DARP provides shared rides in response to advanced requests of trips between any origin and destination within a specific area (see [9] for a survey). In this case, the travel time variation could be a critical factor to determine and schedule vehicle routes. Some papers deal with this time issue nicely for VRPs problems [10–12] and for and DARP problems [13,14]. In [14], Hu et al. formulated a time-dependent Dial-A-Ride problems by a mixed integer programming and conducted the experiments in a Kaohsiung network. In [13], Fu considered a stochastic time-dependent model to construct the heuristic algorithms for scheduling a Dial-A-Ride paratransit.

Some existing models are proposed to handle passengers and parcels transportation with the same vehicle to reduce the congestion and improve the revenue. We refer to some comprehensive surveys on sharing transportation systems in [15–17]. Trentini and Malhene in the survey [15] have given concepts on sharing resources through different transportation modes (e.g., buses, tramways, subways, cars and bikes) in order to improve urban mobility. In [18], Ma et al. proposed a large-scaled dynamic ridesharing version and solved by a greedy algorithm. Besides, Furuhata et al. [17] have surveyed on ride-sharing (more precisely, car-sharing) systems. Through the researches, the term of car-sharing is considered as matching [19] and scheduling problems [20]. Advantages of car-sharing systems for participants to the society, and to the environment can be found at [21–24]. Recently, based on the DARP model, Li et al. developed a new class of models called Share-a-Ride Problem (SARP) in [20] that allowed different passengers and parcels to share the same taxis. Extended the model in [20], Li et al. [25] considered the share-a-ride problem with time windows constraints and solved by an adaptive large neighborhood search. Both models still allow sharing a taxi between passengers and make assumption that the traveling time between two points are constant. However passengers are usually not willing to share the taxi with others because of different requirements such as the shortest transportation time, the private space, the security and so on. Therefore, these models are hard to apply to real-life situations.

In this paper, inspired from the model in [20], we first improve the applicability of the model by considering some new realistic factors and constraints. Apart from the fuel cost, we take into account two other operation costs including the taxi drivers hiring cost and the cost of using taxis. Because of the urban congestion, taxis usually takes more time and spend more operation costs to serve requests than usual during peak hours. When the vehicle speed is lower than a small given value, a small extra fee will be active while riding passengers. The waiting fee of the taxi is a part of this fee. Furthermore, to the best of our knowledge, all proposed transportation models in the literature are based on the distance calculation. In our opinion, the distance-based models are not suitable in crowded cities with congestion consideration when the fuel consumption is always proportional to the traveling time. We propose indeed a fully time-dependent model with many realistic factors and constraints, which is an innovative contribution.

In the urban context, vehicle speeds are dramatically affected by the traffic congestion. The congestion levels are different from zones to zones in the city. We consider therefore in the model three zones for a city including inner city, buffer zone and suburban. We also limit vehicle speeds by different speed windows during a day within a zone. Hence, we use dynamic graphs to handle variable speeds with the *k*-core concept and techniques introduced in [26].

In the literature, share-a-ride models are classified by request's properties: static and dynamic. In the static scenario, all requests are known beforehand. In the opposite scenario, the information about requests is revealed during the execution of the transport system. Here we introduce a generic model with flexible periodic processes, i.e.,requests are processed periodically with a given periodic interval. From the beginning of a periodic interval, upcoming received requests will be added to a processing list. After ending the periodic interval, the requests in the list will be processed to be served. The vehicles will be scheduled to obtain the maximum total benefit. The periodic interval is configured depending on the model scenario as input parameters. Following are some scenario's examples. For the static scenario, the period interval can be set to one day from 00 h 00 min to 23 h 59 min. Thereby, all requests in a day will be received and processed in the next day. For the dynamic scenario, the periodic interval can be set to one minute, i.e.,requests are processed in real-time from the beginning of the day. Another example for the flexible periodic interval is the airport delivery system. Normally, customers have to arrive the airport at least 2 h before the flight to be able to check-in on time. So the booking calls for the airport delivery are usually very early. In this case, our model can be applied to collect requests during certain periodic time and process them after that.

Besides, one of the big challenges in the research community is the lack of real-case data for experimentation. There are very few research papers on sharing systems dealing with the real-case data. As far as we are aware, the transportation data from the Atlanta Metropolitan area is the first to use in sharing systems [19]. Another case is the taxi data in San Francisco city for the model in [20]. The problem we are encountering is complex so that it is reasonable to implement some heuristic algorithms to solve on some real-case data. We first do preprocess the large-scale data on maps and requests. After that we propose some heuristics on dynamic graphs induced from the map data. We initiate a solution by a greedy algorithm and then use local search techniques to improve the solution quality. The feasibility and the efficiency of the model and proposed algorithms are proved by our experiment on real-case requests data provided by Tokyo-Musen taxi company. The Tokyo road map is queried from Open Street Maps with more than 130000 crossing points. The experimented data includes more than 38800 requests, and around 5600 used taxis per day. We have run algorithms on both static and dynamic scenarios. The experimental results are analyzed on all objective factors such as the accumulating traveling time, the accumulating benefit, the number of added taxis every time window and the number of shared requests. The algorithms can be easily applied to other periodic interval scenarios.

This paper is organized as follows. Section 2 shows the problem description. Section 3 provides the mathematical formulation of the model. Proposed algorithms are discussed in Section 4. Section 5 presents the way to process data and to analyze the experimental results. Finally, we conclude the work and discuss future research in Section 6.

2. Problem description

From the set of input requests, the system plans taxis to serve requests to obtain the best total benefit while satisfying all constraints. The system supports two type of requests: *passenger request*, *parcel request*. Each order includes a pair of requests (*pickup-request* and *delivery-request*). All requests provide pickup and delivery location with corresponding time windows, i.e.,each request has to be served within its time window.

As the most cases in real-life, we assume that each taxi has to depart and return to the same depot at the end of a day. On the trajectory, a taxi can serve either a passenger or parcels or both within the permitted capacity. The capacity is verified by the weight of the riding parcels and the weight of the riding passenger request. The weight of every passenger request is set to be constant. The system considers following transportation scenarios for a taxi: (1) carry a passenger only; (2) carry parcels only; (3) carry both a passenger and parcels. Each passenger must be served directly, i.e., no stop is allowed while serving the passenger. During the trajectory, taxis can stay at available parkings so that they can move to the next request destination within its time window. Taxis are also allowed to wait for a limited time at pickup or delivery points. For the safety reason, the total traveling time of a taxi in a day can not exceed a given value.

In a specific zone of a city, the vehicle speed varies and is limited by different speed windows during a day. Our model considers three zone types: inner city, buffer zone, suburban. Within a zone, the system considers three speed window types: *off-peak hour*, *partial-peak hour* and *peak hour*. Each speed window is bounded by the given minimum and maximum speed values.

The problem can be defined on a weighted and directed graph $G = (V, E)$. V is the set of nodes that includes subsets $V^{po} \cup V^{fo} \cup V^{pd} \cup V^{fd} \cup D \cup V^{pa}$, where V^{po} is the set of passenger origins; V^{fo} is the set of parcel origins; V^{pd} is the set of passenger destinations; V^{fd} is the set of parcel destinations; D is the set of depots and V^{pa} is the set of parkings.

Each node $i \in V^{po} \cup V^{fo} \cup V^{pd} \cup V^{fd}$ has a pair of parameters $(\phi_i, \omega_i^{\max})$ that represent the weight and the maximum waiting time to serve request i. Each node i has also a pair values (e_i, ℓ_i) of a corresponding time window. K is the set of taxis. The taxi k is permitted to work in limited time $\tau_k^{\max}$ and has its own capacity σ_k. Each edge (u, v) in the edge-set E is associated to two values $\bar{t}_{u,v}$ and $\tau_{u,v}^t$. $\bar{t}_{u,v}$ corresponds to the borderline traveling time from u to v. If a taxi spends more time than $\bar{t}_{u,v}$ to move from u to v, an extra fee will be counted based on the time difference. $\tau_{u,v}^t$ indicates the traveling time to move from point u to point v if the taxi leaves point u at time point t.

The problem is to find a set of valid taxi trajectories that serves possible requests to achieve the maximum total benefit. The total benefit is the difference between the total revenue and the total cost. The total revenue includes the revenue of traveling passenger, the revenue of traveling parcels and the extra revenue of traveling passenger. The total cost consists of the transportation cost, the hiring taxi drivers cost and the cost of using taxis.

Fig. 1 illustrates the trajectories of 3 taxis. Taxis start from and end at depots $D = \{D_1, D_2, D_3\}$. The trajectory of Taxi 1 is represented by the solid lines. The trajectory of Taxi 2 is represented by the dashed-dotted lines. The trajectory of Taxi 3 is represented by the dashed line. Striped and empty circles represent pickup and delivery requests respectively. The list of passenger pickup and delivery requests are $V^{po} = \{r_1, r_2, r_3, r_4, r_5, r_6\}$ and $V^{pd} = \{r_{12}, r_{13}, r_{14}, r_{15}, r_{16}, r_{17}\}$. There are 6 passenger requests including the pairs (r_1, r_{12}), (r_2, r_{13}), (r_3, r_{14}), (r_4, r_{15}), (r_5, r_{16}) and (r_6, r_{17}). There are 5 parcel requests including the pairs (r_7, r_{18}), (r_8, r_{19}), (r_9, r_{20}), (r_{10}, r_{21}), (r_{11}, r_{22}) with $V^{fo} = \{r_7, r_8, r_9, r_{10}, r_{11}\}$ and $V^{fd} = \{r_{18}, r_{19}, r_{20}, r_{21}, r_{22}\}$. Taxi 1 serves the list of requests $\{r_1, r_{12}, r_7, r_{18}, r_8, r_2, r_{13}, r_{19}\}$. Taxi 2 serves $\{r_{11}, r_4, r_{15}, r_{22}, r_5, r_{16}, r_{10}, r_{21}\}$. Taxi 3 serves $\{r_3, r_{14}, r_9, r_6, r_{17}, r_{20}\}$. All passenger requests are served directly without any interruption. Two parcel requests $\{(r_7, r_{18}), (r_{10}, r_{21})\}$ are served directly without any interruption. The others are shared between parcels or between a passenger and parcels. $V^{pa} = \{P_1, P_2\}$. Taxi 2 stops at the parkings P_1 and P_2 before going to the points r_5 and r_{10} respectively. Taxi 3 stops at the parking P_2 before going to the point r_9.

3. Mathematical formulation

Based on the problem description, we introduce a mathematical formulation for our fully time-dependent model. We inherit

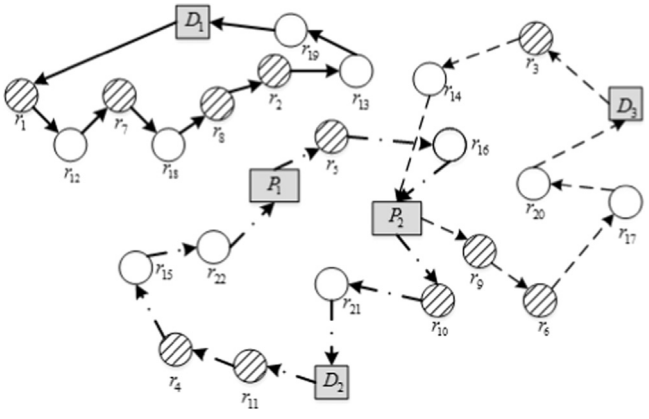

Fig. 1. An share-a-ride example between a passenger and parcels.

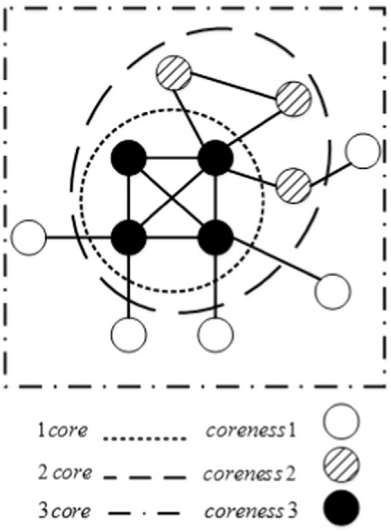

Fig. 2. An example of k-core parts of a graph.

some features of the mathematical formulation in [20].

Following the model, the traveling time between two points varies and depends on the departure time at the origin point. To indicate the limited speed, we propose $\tau^{\min}_{u,v,t}$ and $\tau^{\max}_{u,v,t}$ for the minimum and maximum travel time from point u to point v if a taxi departs point u at time point t. The traveling time between two points for the departure time point t is represented by the variable $\tau^{t}_{u,v}$ Fig. 2.

This section includes 4 parts: 1) Input definitions; 2) Variable declarations; 3) Objective functions; 4) Constraints.

3.1. Input definitions

- A weighted and directed graph $G = (V, E)$;
- n, m: number of passengers and parcels, let $s = n + m$;
- p: number of physical parkings. Each physical parking might be visited several times by a taxi. To ease the formulation (each point is visited at most once), we create, for each physical parking, a set of logical parkings;
- $K = \{1,\ldots,|K|\}$: set of taxis;
- $V^{po} = \{1,\ldots,n\}$: set of passenger origins;
- $V^{fo} = \{n + 1,\ldots,n + m\}$: set of parcel origins;
- $V^{pd} = \{s + 1,\ldots,s + n\}$: set of passenger destinations, i and $i + s$ are respectively the pickup and delivery points of the passenger request i, $\forall\ i = 1,\ldots,n$;
- $V^{fd} = \{s + n + 1,\ldots,2\,s\}$: set of parcel destinations, $i + n$ and $i + n + s$ are respectively the pickup and delivery points of the parcel request i, $\forall\ i = 1,\ldots,m$;
- $V^{pa} = \{2\,s + 1,\ldots,2\,s + pM\}$: set of logical parkings ($\forall\ i = 1,\ldots,p$, M is a big constant), in which parkings $2\,s + (i - 1)M + 1,\ldots,2\,s + iM$ are associated with the i^{th} physical parking with a capacity c_i which is the maximum number of taxis staying at a time ($\forall\ i = 1,\ldots,p$);
- $D = \{2\,s + Mp + 1,\ldots,2\,s + Mp + |K|\}$: set of depots of K taxis in which we denote $\theta_k = 2\,s + Mp + k$ the depot of taxi k, $\forall\ k \in K$;
- $V^r = V^{po} \cup V^{pd} \cup V^{fo} \cup V^{fd}$: set of requests;
- $V = V^{po} \cup V^{pd} \cup V^{fo} \cup V^{fd} \cup D \cup V^{pa}$;
- $\Gamma^{+}(i) = \{j|(i, j) \in E\}$, $\forall\ i \in V$;
- $\Gamma^{-}(i) = \{j|(j, i) \in E\}$, $\forall\ i \in V$;
- $T^p = \{1, 2,\ldots,|T^p|\}$: sequence of time points;
- E: set of arcs (u, v), $\forall\ u \in V$, $\forall\ v \in V$;
- Each arc $(u, v) \in E$ is associated with two following constants:
  1. $\tau^{\min}_{u,v,t}$: the minimum traveling time from point u to point v if taxi leaves point u at time $t \in T^p$;
  2. $\tau^{\max}_{u,v,t}$: the maximum traveling time from point u to point v if taxi leaves point u at time $t \in T^p$;

The values of $\tau^{\min}_{u,v,t}$ and $\tau^{\max}_{u,v,t}$ are analyzed and calculated in Section 4.1;.

- $\omega^{\max}_{v}$: the maximum waiting time allowed at point $v \in V$ (by convention $\omega^{\max}_{v} = \infty$, $\forall\ v \in V^{pa}$);
- Each point $i \in V$ is associated with a weight ϕ_i representing the weight of the corresponding request. By convention, each pickup point of a request is associated with a positive weight ϕ_i and the corresponding delivery point is associated with a negative weight $-\phi_i$. The weights of points of depots and parkings are 0. More precisely:

- $\phi_i + \phi_{i+s} = 0 \wedge \phi_i > 0, \forall\, i \in V^{fo} \cup V^{po}$:
- $\phi_i = 0\ \forall\, i \in D \cup V^{pa}$
- $[e_i, \ell_i]$: the time window of request i, $\forall\, i \in V^{po} \cup V^{pd} \cup V^{fo} \cup V^{fd}$;
- $\sigma_k^{\max}$: the maximum capacity of taxi k, $\forall\, k \in K$;
- $\tau_k^{\max}$: the maximum duration for taxi k, $\forall\, k \in K$;
- $\bar{t}_{i,j}$: the borderline time for traveling from point i to point j. If the traveling time of taxis from point i to point j is higher than this borderline time, an extra fee will be counted based on the time difference;
- α: the initial fare charged for delivering one passenger;
- γ_1: the fare charged for delivering one passenger per unit time;
- γ_2: the fare charged for delivering one kilogram parcel;
- γ_3: the average cost (e.g., fuel) per minute for the request delivery;
- γ_4: the cost per exceeded minute for the passenger delivery;
- γ_5: the cost per minute for hiring a taxi driver. The total hiring time of a taxi drive in a day is calculated from his start-working time at the depot and his end-working time at that depot of the day;
- γ_6: the cost of using a taxi per day.

3.2. Variable declarations

- $X_{i,j}^k$: equal 1 if taxi k goes from point i to point j; equal 0: otherwise, $\forall\, (i, j) \in E$;
- a_i^k: the time point when taxi k arrives at point i, $\forall\, i \in V$;
- d_i^k: the time point when taxi k departs from point i, $\forall\, i \in V$;
- $\tau_{i,j}^t$: the traveling time from point i to point j if a taxi starts from point i at time point t, $\forall\, (i, j) \in E$, $\forall\, t \in T^p$;
- w_i^k: the load of taxi k right after visiting point i, $\forall\, k \in K$, $\forall\, i \in V$. The value is the sum of all passenger and parcel weights in car;
- $Z_{t,p}^k$: equals to 1 if taxi k is at parking p at time point t, and equals to 0 otherwise, $\forall\, t \in T^p$, $\forall\, k \in K$, $p \in V^{pa}$;
- Y_i^k: equals to 1 if request i is served by taxi k, and equals to 0 otherwise, $\forall\, i \in V^{po} \cup V^{fo}$, $k \in K$;

3.3. Objective functions

- The revenue from all passenger deliveries by traveling time:

$$f_1 = \sum_{k \in K} \sum_{i \in V^{po}} (\alpha + \gamma_1 \tau_{i,i+s}^{d_i^k}) Y_i^k \tag{1}$$

- The extra revenue from passenger deliveries when the traveling time exceeds the borderline time:

$$f_2 = \gamma_4 \sum_{k \in K} \sum_{i \in V^{po}} \max((a_{i+s}^k - d_i^k) Y_i^k - \bar{t}_{i,i+s}, 0) \tag{2}$$

- The revenue from all parcel deliveries:

$$f_3 = \gamma_2 \sum_{k \in K} \sum_{i \in V^{fo}} \phi_i Y_i^k \tag{3}$$

- The transportation cost by traveling time:

$$f_4 = \gamma_3 \sum_{k \in K} \sum_{i \in V} \sum_{j \in \Gamma^+(i)} (a_j^k - d_i^k) X_{i,j}^k \tag{4}$$

- The hiring cost of taxi drivers:

$$f_5 = \gamma_5 \sum_{k \in K} (a_{\theta_k}^k - d_{\theta_k}^k) \tag{5}$$

- The cost of using taxis:

$$f_6 = \gamma_6 \sum_{k \in K} \sum_{j \in \Gamma^+(\theta_k)} X_{\theta_k,j}^k \tag{6}$$

- The total benefits:

$$f = f_1 + f_2 + f_3 - f_4 - f_5 - f_6 \rightarrow \max \tag{7}$$

3.4. Constraints

- Each point is visited at most once in a taxi trajectory:

$$\sum_{j \in \Gamma^+(i)} \sum_{k \in K} X_{i,j}^k \leq 1, \forall\ i \in V \tag{8}$$

- Consistency of orders, i.e., if a passenger/parcel is picked up, he/it must be delivered:

$$\sum_{j \in \Gamma^+(i)} X_{i,j}^k = \sum_{j \in \Gamma^-(i+s)} X_{j,i+s}^k, \forall\ i \in V^{po} \cup V^{fo}, k \in K \tag{9}$$

- Flow conservation constraints:

$$\sum_{j \in \Gamma^+(i)} X_{i,j}^k = \sum_{j \in \Gamma^-(i)} X_{j,i}^k, \forall\ i \in V, k \in K \tag{10}$$

- Out-flow at the depots of used taxis:

$$\sum_{i \in V^{po} \cup V^{fo}} Y_i^k = \sum_{j \in \Gamma^+(\theta_k)} X_{\theta_k,j}^k, \forall\ k \in K \tag{11}$$

- Constraints over the arrival, departure and traveling time between two points:

$$a_j^k = (d_i^k + \tau_{i,j}^{d_i^k}) X_{i,j}^k, \forall\ (i,j) \in E, \forall\ k \in K \tag{12}$$

- Bound of the direct traveling time between two adjacent points:

$$\tau_{i,j,t}^{\min} \leq \tau_{i,j}^t \leq \tau_{i,j,t}^{\max}, \forall\ (i,j) \in E, \forall\ t \in T^p \tag{13}$$

- Constraints on the departure and arrived times of each used taxi:

$$\sum_{i \in V^{po} \cup V^{fo}} Y_i^k \geq 1 \Rightarrow d_{\theta_k}^k < a_{\theta_k}^k \tag{14}$$

- Constraint over the maximum riding time of taxis:

$$\tau_k^{\max} \geq a_{\theta_k}^k - d_{\theta_k}^k, \forall\ k \in K, \theta_k \in D \tag{15}$$

- Constraint over the time window of requests:

$$e_i \leq a_i^k Y_i^k \leq \ell_i, \forall\ i \in V^{po} \cup V^{fo} \cup V^{pd} \cup V^{fd}, k \in K \tag{16}$$

item Constraint over the maximum waiting time at stops:

$$d_i^k - a_i^k \leq \omega_i^{\max}, \forall\ i \in V \backslash D, \forall\ k \in K \tag{17}$$

- Constraints over the load of taxis:

$$\begin{cases} w_j^k = (w_i^k + \phi_j) X_{i,j}^k, & \forall\ k \in K,\ (i,j) \in E \\ w_i^k \geq \max\{0, \phi_i\}, & \forall\ i \in V,\ k \in K \\ w_i^k \leq \min\{\sigma_k^{\max}, \sigma_k^{\max} + \phi_i\}, & \forall\ i \in V,\ k \in K \end{cases} \tag{18}$$

- Constraints over the capacity of parking:

$$\begin{cases} t < a_p^k \vee t > d_p^k \Rightarrow Z_{t,p}^k = 0, & \forall\, p \in V^{pa}, \forall\, k \in K, \quad t \in T^p \\ a_p^k \le t \wedge t \le d_p^k \Rightarrow Z_{t,p}^k = 1, & \forall\, p \in V^{pa}, \forall\, k \in K, \quad t \in T^p \\ \sum_{k \in K} \sum_{j=1}^{M} Z_{t,2\,s+(i-1)M+j}^k \le c_i, & \forall\, t \in T^p, \qquad\qquad i = 1, \ldots, p \end{cases} \tag{19}$$

- Direct passenger delivery:

$$X_{i,i+s}^k = Y_i^k, \forall\, i \in V^{po}, k \in K \tag{20}$$

- Channeling constraint:

$$\sum_{j \in \Gamma^+(i)} X_{i,j}^k = Y_i^k, \forall\, i \in V^{po} \cup V^{fo}, k \in K \tag{21}$$

4. Proposed solutions

The problem is an extension of the SARP, which is known to be an NP-hard problem [20]. We implement some heuristic methods to solve this dynamic complex system. After pre-processing steps on the data and on the dynamic graphs, we initiate the solution by a greedy algorithm and then use local search techniques to improve the solution quality.

This section is organized as follows. We first introduce the path finding technique to find out the shortest path in dynamic graphs in Section 4.1. Then, we apply this technique to the heuristic algorithms for both direct and sharing models in Section 4.2 and Section 4.3 respectively.

4.1. Path finding in dynamic graphs

We use the graph representation for the network. The vertex set V is the set of all pickup and delivery points, parking and depots. The characteristics of transportation in the urban context crucially depend on different time spot during a day. In order to model this dependence, the time that a taxi traverses an edge in the graph has to be a time-variant function. A graph of these functions is called a dynamic graph satisfying the FIFO property, i.e., for two cars going the same trajectory, the car that starts later will always finishes later. The congestion issue is prepared as the constant factors which are determined by the collected vehicle speeds by hours from Tokyo data. In this paper, a continuous piecewise linear function is used to calculate the traveling time based on this congestion issue:

$$\tau_{u,v}^t = \begin{cases} C_0 + a_0 t, & \text{if} t \in p_0 \\ \quad \vdots & \\ C_n + a_n t, & \text{if} t \in p_n \end{cases}$$

where,

- $\tau_{u,v}^t$ is the traveling time from point u to v if leaving from point u at the time point t;
- $p_0, p_1, \ldots, p_n$ are time periods;
- $C_0, C_1, \ldots, C_n, a_0, a_1, \ldots, a_n$ are constant parameters,
  1. $a_i = 1$, the velocity is constants during the period p_i;
  2. $a_i > 1$, the velocity is decreasing during the period p_i;
  3. $a_i < 1$, the velocity is increasing during the period p_i.

It is clear that this function satisfies the FIFO properties. Therefore, we use the Dijkstra's algorithm to calculate the fastest traveling time for each taxi' trajectory. Note that each edge is labeled by a time-variant function.

4.1.1. Contraction hierarchies

If a simple Dijkstra's algorithm is used for a large data, it will lead overload of computation. Therefore, we use the contraction hierarchies technique on the dynamic graph in [26], specifically the technique on the k-core of a graph. A k-core of an undirected graph G is a maximal connected subgraph of G in which every vertex has degree at least k. First, we decompose the graph into k-core parts with k at last 3. Then, we find the shortest paths on each part of graph and between those parts.

4.1.2. Effectiveness

The worst-case complexity of the finding shortest distances is mainly the complexity of finding shortest distances on the 3-core parts. The shrinking factor, denoted as S_k, is the ratio between the number of vertices in the whole graph and those in 3-core parts. The S_k in Tokyo dataset is about 10 to 11. Since the complexity of the Dijkstra's algorithm is $O(|V| \log |V|)$, it reduces the operations

up to 33 (≈10 log 10) times. In practice, it speeds up the queries 20 times faster in average.

This path finding technique in the dynamic graph will be applied to calculate the traveling times in order to check related traveling time constraints and to calculate the objective factors in the following algorithms.

4.2. The heuristic algorithm for the direct delivery model

We define a beginning time of a request as the earliest time a taxi can pickup this request. For each request in the direct serving strategy, we find the nearest taxi at the time to serve this request. After delivering a passenger or a parcel, the taxi will move to the nearest available parking to wait for the next pickup request. It is noticeable that each passenger or parcel request is served separately in the direct serve strategy. In this strategy, we do not reject requests that are not profitable. This strategy allows more non-free taxis running on the road and makes more chances to serves up-coming requests. This would make much better benefits finally. A taxi can serve one or many request sequentially. The pseudo-code for the direct delivery model is presented in Algorithm 1.

Algorithm 1. Heuristic algorithm for the direct delivery model.

```
1  Sort all pickup requests incrementally by the beginning time into list V';
2  foreach request i ∈ V' do
3  |  Update all status of taxis in K and parking in V^pa at the beginning
   |   time of request i;
4  |  if found the nearest taxi k ∈ K that can serve request i then
5  |  |  Insert request i into the end of the trajectory of taxi k;
6  |  |  Update the total benefit;
7  |  else
8  |  |  Reject request i;
9  |  end
10 end
```

The algorithm 1 consists of two main modules: requests arrangement and the inner loop to search and update available taxis. The module of the requests arrangement takes $O(s \log s)$ where s is the number of pickup requests. The updating and searching steps in the inner loop cost $O(|K|)$. Therefore, the complexity of the heuristic algorithm for the direct delivery model is $O(\max(|K|*s, s \log s))$.

4.3. The heuristic algorithm for the dynamic sharing model

4.3.1. General structure

For the dynamic sharing model, our heuristic algorithm includes two major steps:

(1) Preparing the graph from the data and preprocessing the graph using the hierarchies contraction technique;
(2) At the end of each periodic interval,
   - Collect received requests and update all taxi status;
   - Initiate the solution by using the greedy algorithm described in Section 4.3.2;
   - Apply local search techniques to improve the current best solution (see Section 4.3.3).

The periodic interval is configured depending on the model scenario as an input parameter. This technique makes our model generic and applicable. We do experiment the algorithms on both static and dynamic scenarios which are the most practical cases. In the static scenario, the period interval is set to one day. In the dynamic scenario, the period interval is set to 10 minutes.

4.3.2. The greedy algorithm

To initiate the solution by using the greedy algorithm, we first sort pickup requests by a flexibility value. We introduce a function f_r that represents the flexibility of a pickup request:

$$f_r(i) = \mu(\ell_{i+s} - e_i) - \nu\lambda_i^{\min} \tag{22}$$

where,

- e_i is the earliest time to serve pickup request i;
- ℓ_{i+s} is the latest time to serve the corresponding delivery request of pickup request i;
- $\lambda_i^{\min}$ is the distance from the nearest parking to the location of pickup request i;
- μ, ν are tuning parameters.

The parameter ν depends mostly on the density of taxis over the traffic map, and the parameter μ depends mostly on the density of requests over time. However, both parameters μ and ν have to be tuned by the experimentation. The higher flexibility value of

pickup request i, the more cases to accept the pickup request i. Because of wide interval time windows, parcel pickup requests always have higher flexibility values than passenger pickup requests. For each order, we try to add the pair of pickup and delivery points into every trajectory of taxis and evaluate the total benefit to find out the best one. The pseudo-code of the greedy algorithm is presented in Algorithm 2:

Algorithm 2. The greedy algorithm for the dynamic sharing model.

```
1 Sort all pickup requests incrementally by the flexibility f_r into list R';
2 foreach pickup request i ∈ R' do
3     if found a taxi k ∈ K to serve the order i that gains the best total
       benefit then
4         Insert the order i into the trajectory of taxi k;
5         Update the total benefit;
6     else
7         Reject the pickup request i;
8     end
9 end
```

The module for sorting requests takes $O(s \log s)$. The updating and searching steps in the inner loop take $O(|K|)$. Therefore, the complexity of Algorithm 2 is $O(|K|*s \log s)$.

4.3.3. Local search improvement

To improve the quality of the current solution, the system first removes a small subset of *less effectiveness requests* and then tries to re-insert each of them by a greedy strategy until there is no increment of the total benefit. Similar to the greedy algorithm 4.3.2, we propose a function to represent the effectiveness of a request. Given a request r which is served by a taxi k. Suppose that the itinerary of taxi k is a sequence of points $u_0, u_1, \ldots, u_\ell, u_{\ell+1} \equiv u_0$ ($u_0, \ldots, u_{\ell+1} \in V$ and u_0 is the depot of taxi k). Denote $\lambda^{min}_{u,v}$ the shortest distance from point u to point v. Suppose u_i and u_j ($0 < i < j < \ell + 1$) are pickup and delivery points of a request r, the effectiveness of request r is measured by the following function:

$$g(r) = (\lambda^{\min}_{u_{i-1},u_i} + \lambda^{\min}_{u_i,u_j} - \lambda^{\min}_{u_{i-1},u_j}) + (\lambda^{\min}_{u_i,u_j} + \lambda^{\min}_{u_j,u_{j+1}} - \lambda^{\min}_{u_i,u_{j+1}}) \tag{23}$$

In the experimentation of the dynamic scenario, the number of request to remove and re-insert is 10% of all requests that have not started serving yet. The pseudo-code of the local search for improving the solution in the dynamic sharing model is presented in Algorithm 3.

Algorithm 3. A local search algorithm for improving the solution in the dynamic sharing model.

```
1  Let p_size is the percentage of request numbers to remove and re-insert at
     each iteration;
2  repeat
3      Calculate the effectiveness value of all pickup requests;
4      Sort all pickup requests incrementally by the effectiveness value and
        take top p_size pickup requests into list V' ;
5      foreach pickup request i ∈ V' do
6          Update all status of taxis in K and parking in V^pa by the
            beginning time of request i;
7          Let k ∈ K is the current taxi served the order i;
8          if found a taxi k' ∈ K to serve the order i that improved the best
            total benefit then
9              Insert the order i into the trajectory of taxi k';
10             Update the best total benefit;
11         end
12     end
13 until not improve the best total benefit;
```

The module for sorting requests takes $O(s \log s)$. The inner loop at line 5 takes $O(|K|*|V| \log |V|)$. Therefore, the complexity of Algorithm 3 is $O(i*|K|*s \log s*|V| \log |V|)$ where i is the number of iteration steps and K is the set of taxis.

5. Experimentation

For the experimentation, datasets are processed and generated from the probe data set provided by Tokyo-Musen Taxi Company.

This data set was recorded in 2009 from real-case taxi trajectories in Tokyo. Each day data from this set is an input case for our algorithms. The Tokyo road map is queried from Open Street Maps.

As pointed out in the Section 4, four solutions corresponding to four algorithms are experimented. We collect different information of the solutions and analyze their feasibility and the efficiency. The algorithms are implemented in the Java environment. Experiments are run on a 2.4 GHz Intel Core i7 processor with 16GB of RAM. We execute the algorithms on one month data of January 2009. For the sake of the brevity, we show here the experimental results for 4 continuous days (22/01/2009–25/01/2009). With working days and the weekend, the experimental days handle all common circumstances of the Tokyo's traffic.

This section is organized as follows. Section 5.1 introduces the processing of the dataset generation. Section 5.2 analyzes the experimental results.

5.1. Dataset generation

The datasets are created by 4 processing steps: 1) Tokyo's road network; 2) Taxi requests conversion; 3) Speed windows; 4) Parking and depot locations.

5.1.1. Tokyo's road network

Tokyo's road map is a square map with 80 km in length. It has been divided into 8 × 8 regions. Tokyo's road map includes about 130000 crossing points, 15000 roads and 1000 customers per hour at proximity. We processed data in two steps: the *dataset pre-processing* step and the *map reducing* step.

Step 1 - Dataset pre-processing**.** We consolidated 64 separated regions into one map. A location on the map is represented by a pair of integers in range of [0, 80000]. For the convenience and standardization, we converted those Cartesian coordinate points into geometry points *(longitude, latitude)*. Since the bottom left map is (35°20′*N*, 139°00′*E*) and the top right one is (36°00′*N*, 140°00′*E*), the latitude (g_x) and the longitude (g_y) of a considered point are calculated by the following conversion formula:

$$\begin{cases} g_x = 35°20' + \dfrac{x}{80000} * (36°00' - 35°20') \\ g_y = 139°00' + \dfrac{y}{80000} * (140°00' - 139°00'), \end{cases} \tag{24}$$

where x, y are the coordinates of the considered point.

For calculating the distance of each edge, we use Haversine formula in [27]:

$$d_{x,y} = 2r\arcsin\sqrt{\sin^2(\frac{g_{x_2} - g_{x_1}}{2}) + \cos(g_{x_1})\cos(g_{x_2})\sin^2(\frac{g_{y_2} - g_{y_1}}{2})}, \tag{25}$$

where,

- $d_{x,y}$ is the distance between the two points;
- r is the radius of the sphere;
- (g_{x_1}, g_{y_1}) is the latitude and longitude of the first point;
- (g_{x_2}, g_{y_2}) is the latitude and longitude of the second point.

Step 2 - Map reducing. In Tokyo's road map, roads are represented by a number of continuous straight lights. Hence, a point on a road is considered as a connecting point if it is neither a stop nor an intersection. We consider only connecting points to reduce the map. For each 2-degree point that are not pickup or delivery points, we remove this point and then connect its two neighbor points by a new edge with the same distance. This step helps to reduce the redundancy points without affection to the technique of routing finding in the dynamic graph 4.1. We removed such 117000 points that took 90% of the vertices from the map to generate the Tokyo dataset. This helps to run at least 100 time faster than on the original graph.

5.1.2. Taxi requests conversion

The probe data contains only the location in (longitude, latitude), serving time (pickup time, delivery time) of passenger requests and the taxi's speeds. Therefore, we have to match request's locations to Tokyo's road network and generate reasonable time windows of requests. To match a location of a request to the map's location, we replace the request's location with its nearest point that sorted by the Haversine distance 25. Then, we take 70% of passenger requests to be parcel requests. For all parcel requests, their time windows are either the morning shift [8 h30, 12 h00] or the afternoon shift [12 h00, 21 h00]. For each passenger request, either pickup or delivery request, the time window is set to be 10 min from the real serving time in the probe data.

5.1.3. Speed windows processing

Vehicle speeds are dramatically affected by the traffic congestion and by specific zones in the city. Therefore, we handle speed windows in three zones for a city including inner city, buffer zone and suburban. Within a zone, we classify the speed windows by the congestion levels. We have analyzed the daily traffic data of Tokyo city of the continuous days from 20/01/2009 to 31/01/2009. The average speed of taxis by time windows of these periods are illustrated in Table 1. Then, we divided the map into 3 different regions:

Table 1
Average speeds by hours of taxis in Tokyo-Musen Taxi in January, 2009.

Hour (hh:mm)	Average speed (km/h)	Hour (hh:mm)	Average speed (km/h)
00:00	30.87	00:30	31.64
01:00	36.01	01:30	35.11
02:00	36.51	02:30	34.95
03:00	34.71	03:30	33.93
04:00	35.47	04:30	35.27
05:00	32.98	05:30	31.96
06:00	29.65	06:30	26.51
07:00	23.42	07:30	20.99
08:00	19.82	08:30	18.95
09:00	19.44	09:30	18.63
10:00	18.23	10:30	18.38
11:00	18.92	11:30	19.82
12:00	20.28	12:30	20.73
13:00	20.28	13:30	19.80
14:00	19.28	14:30	18.67
15:00	18.79	15:30	18.85
16:00	18.70	16:30	18.73
17:00	18.26	17:30	18.15
18:00	18.98	18:30	19.15
19:00	21.11	19:30	22.03
20:00	22.33	20:30	22.94
21:00	23.53	21:30	24.61
22:00	25.39	22:30	24.73
23:00	26.34	23:30	29.15

Table 2
Congestion levels of Tokyo-Musen Taxi (2009).

Level	Time windows(hh:mm)
Off-peak	[00:00, 05:30]; [19:30, 00:00]
Partial-peak	[05:30, 07:30]; [09:00, 17:30]
Peak	[07:30, 09:00]; [17:30, 19:30]

Table 3
Speed windows of Tokyo-Musen Taxi in 3 regions (2009).

Level	Inner city (km/h)	Buffer zone (km/h)	Suburban (km/h)
Off-peak	[30, 50]	[30, 50]	[30, 50]
Partial-peak	[14, 30]	[30, 50]	[30, 50]
Peak	[5, 14]	[14, 30]	[30, 50]

inner city, buffer zone and suburb by maximum speeds in peak hour. Table 2 shows the six different time windows classified into three congestion levels. By analyzing data, we propose time windows with speed limit bounds for three regions during a day in Table 3.

5.1.4. Parking, depots locations, and the other parameters

By assumption, a depot is the place where a taxi starts working and returns after finishing its trajectory in a day. We retrieved all starting and finishing points of taxis throughout a working day in the probe data to determine the depots. Besides, a parking is the place that a taxi can stop to wait for the next request. From these probe data, we calculated the frequencies of the stopping points of taxis, i.e.,where taxis have stopped for more than 3 minutes. We summarize in Table 4 the dataset information for 4 continuous days from Thursday, 22th January 2009 to Sunday, 25th January 2009. The number of requests in a weekend day is about 30% smaller than the number of requests in a working day. Table 5 and 6 show a taxicab fare rate of Tokyo-Musen Taxi and a parcel fare rate referred in the websites [28] and [29]. Table 7 shows the parameters of proposed algorithms for experimentation.

5.2. Computational results

The results of our algorithms are reported in Tables 8–11. The total benefit, the revenue, and the cost are calculated by the unit of

Table 4
Dataset information.

Dataset	1 (22/01/2009)	2 (23/01/2009)	3 (24/01/2009)	4 (25/01/2009)
Number of passenger requests	13483	14213	11842	9395
Number of parcel requests	29404	30972	25762	20361
Number of depots	6032	6226	5615	4766
Number of parking	128	120	133	114

Table 5
A rate of taxicab passenger fare in Japan (2009).

Distance (km)	¥
First 2 km	730
Each additional 280 m	90

Table 6
A rate of taxicab parcel fare in Japan (2009).

Type	Maximum dimensions (cm)	Weight (kg)	¥
60	60	2	756
80	80	5	972
100	100	10	1188
120	120	15	1404
140	140	20	1620
160	160	25	1836

Table 7
Parameters used for experimentation.

Parameters	Value	Parameters	Value	Parameters	Value
α	730	γ_1	322	γ_2	27
γ_3	11	γ_4	113	γ_5	28
μ	0.7	ν	0.3		

Table 8
Computational results of Dataset 1.

	Direct delivery	Static scenario	Dynamic scenario (Greedy+LS)	Dynamic scenario (Greedy)
Total benefit	**4761.98**	**5402.70**	**5304.24**	**5265.26**
Passenger revenue	2958.27	2958.06	2957.87	2957.70
Extra revenue of passenger	133.68	133.72	133.68	133.68
Parcel revenue	3802.09	3802.09	3802.09	3802.09
Hiring taxi driver cost	1253.70	934.32	944.82	970.69
Usage taxi cost	251.50	89.33	172.17	172.17
Transportation cost	626.85	467.15	472.41	485.35
Traveling time (hour)	11282.44	8408.08	8502.66	8735.53
Served passenger requests	13483	13483	13483	13483
Served parcel requests	29404	29404	29404	29404
Shared requests	0	29802	29740	28905
Number of used taxis	3848	1440	2322	1287
Execution time(second)	1594	9784	4613	3218

10000¥. Each algorithm is summarized by the total benefit, the revenue of passengers, the extra revenue of passenger, the revenue of parcels, the cost of hiring drivers, the cost of using taxi, the transportation cost, the traveling time, the number of served requests, the number of shared requests, the number of used taxis, and the execution time of algorithms.

We claim the performance of our algorithms by showing the experimental results on both *direct delivery* and *sharing* models. The sharing model consists of the *static scenario* and the *dynamic scenario*. According to the experimental results, all factors

Table 9
Computational results of Dataset 2.

	Direct delivery	Static scenario	Dynamic scenario (Greedy+LS)	Dynamic scenario (Greedy)
Total benefit	**5019.32**	**5710.87**	**5601.77**	**5576.08**
Passenger revenue	3115.06	3116.00	3116.00	3116.00
Extra revenue of passenger	138.10	138.10	138.10	138.10
Parcel revenue	4005.51	4005.51	4005.51	4005.51
Hiring taxi driver cost	1325.23	972.94	990.45	1007.58
Usage taxi cost	251.50	89.33	172.17	172.17
Transportation cost	662.62	486.47	495.23	503.79
Traveling time (hour)	11926.14	8755.76	8913.35	9067.46
Served passenger requests	14213	14213	14213	14213
Served parcel requests	30972	30972	30972	30972
Shared requests	0	31622	31362	30639
Number of used taxis	4182	1465	2364	1337
Execution time(second)	1712	10526	5439	4981

Table 10
Computational results of Dataset 3.

	Direct delivery	Static scenario	Dynamic scenario (Greedy+LS)	Dynamic scenario (Greedy)
Total benefit	**4059.70**	**4772.88**	**4683.51**	**4620.48**
Passenger revenue	2759.34	2759.19	2759.19	2759.19
Extra revenue of passenger	81.73	81.73	81.73	81.73
Parcel revenue	3337.00	3337.00	3337.00	3337.00
Hiring taxi driver cost	1244.59	877.14	881.50	923.51
Usage taxi cost	251.50	89.33	172.17	172.17
Transportation cost	622.29	438.57	440.75	461.76
Traveling time (hour)	11200.36	7893.59	7932.84	8310.97
Served passenger requests	11842	11842	11842	11842
Served parcel requests	25762	25762	25762	25762
Shared requests	0	25819	25895	24591
Number of used taxis	3883	1429	2221	1281
Execution time(second)	1357	7915	4108	3823

Table 11
Computational results of Dataset 4.

	Direct delivery	Static scenario	Dynamic scenario (Greedy+LS)	Dynamic scenario (Greedy)
Total benefit	**3073.46**	**3655.42**	**3565.09**	**3515.03**
Passenger revenue	2055.82	2055.45	2055.45	2055.45
Extra revenue of passenger	75.98	75.98	75.98	75.98
Parcel revenue	2633.45	2633.45	2633.45	2633.45
Hiring taxi driver cost	960.18	680.08	685.08	718.46
Usage taxi cost	251.50	89.33	172.17	172.17
Transportation cost	480.09	340.04	342.54	359.23
Traveling time (hour)	8640.97	6120.23	6165.25	6465.59
Served passenger requests	9395	9395	9395	9395
Served parcel requests	20361	20361	20361	20361
Shared requests	0	20396	20386	19224
Number of used taxis	3040	1152	1736	982
Execution time(second)	1218	7142	3705	3476

acquired from the sharing model are clear better than the traditional model such as better benefits, less total traveling time and fewer used taxis.

In general, the total benefit of the sharing model is about 1.15 times better than the one of the direct delivery model. The requests are served fully in either the sharing model or the direct delivery model. Especially, the number of shared requests takes 69.5% of the total number in the sharing model. It affects strongly on the number of used taxis. The number of used taxis in sharing model reduces to about 0.47 times than in the direct delivery. Due to a big number of shared requests and the fewer number of used taxis,

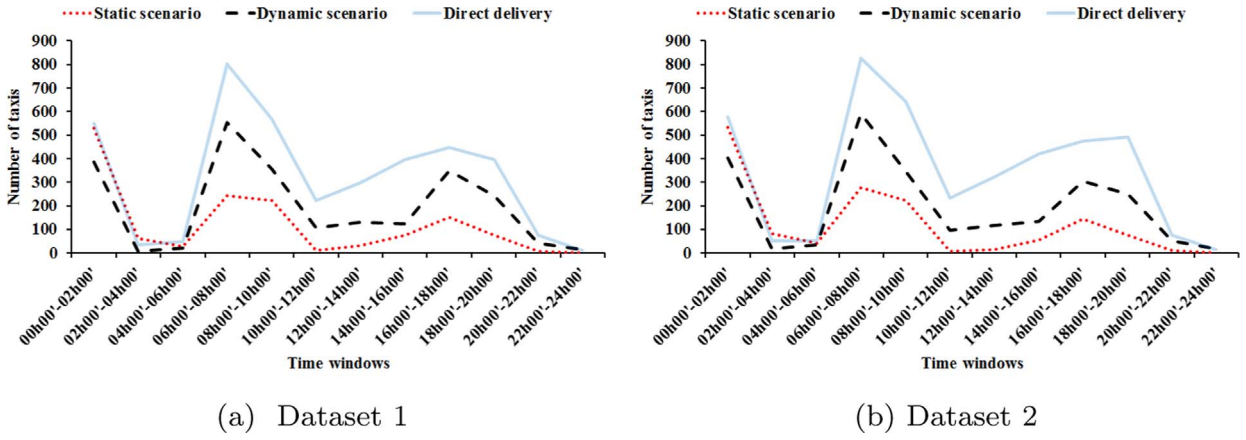

Fig. 3. Number of added taxi by time windows in Dataset 1 and Dataset 2.

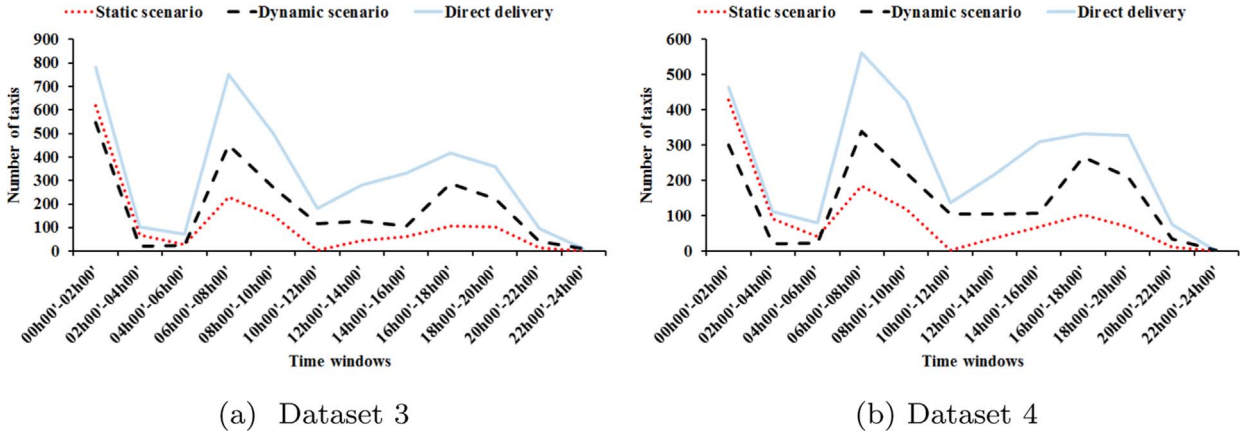

Fig. 4. Number of added taxi by time windows in Dataset 3 and Dataset 4.

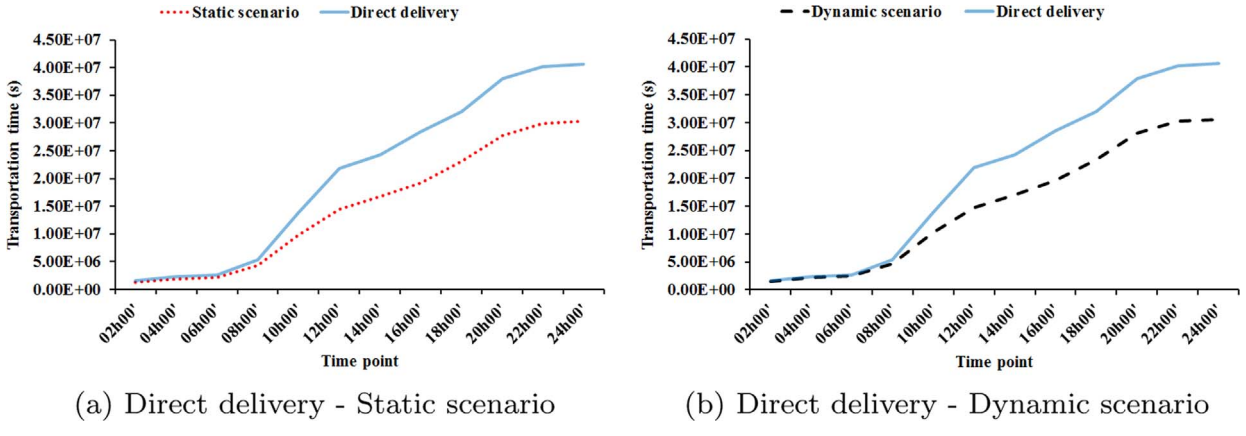

Fig. 5. Accumulating transportation times every 2 h in Dataset 1.

the total traveling times of the sharing model are about 0.27 times smaller than the one of the direct model.

Within the sharing model, the static scenario has a slightly better total benefit. Because all requests in a day are known beforehand, the algorithms have more chance to match parcels together or with a passenger in a taxi. So the static scenario takes fewer taxis to serve. The more number of parcel requests are, the more static scenario gains. The tables also show the performance of the heuristic algorithms. Although the number of used taxis after using the local search process augments about 1.77 times than after using the greedy algorithm only, the total benefit is slightly improved by the local search process.

For more details, we present following figures on accumulating data and over the time windows of the total benefits, the traveling times, and the number of added taxis. In these figures, the dashed lines indicate the results of the dynamic scenario; the round dot

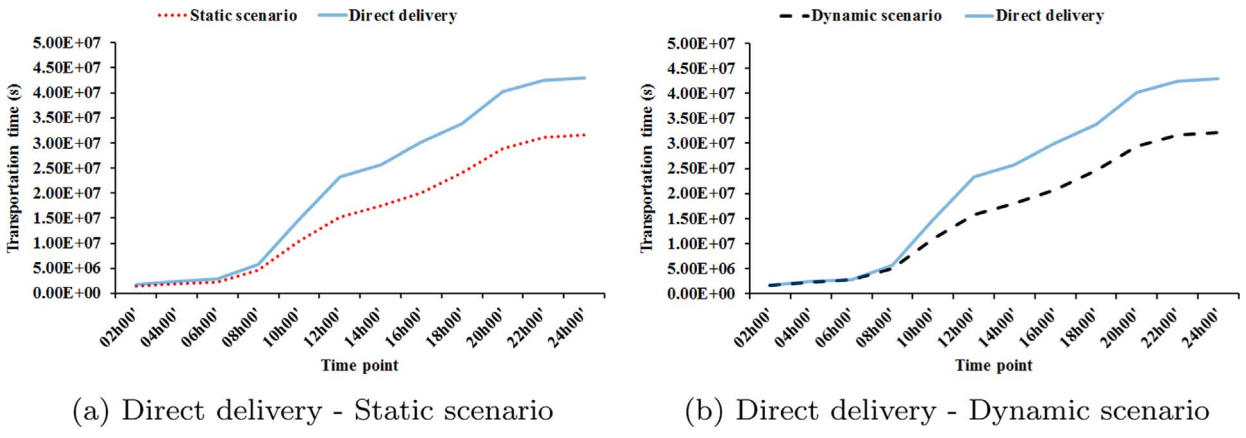

Fig. 6. Accumulating transportation times every 2 h in Dataset 2.

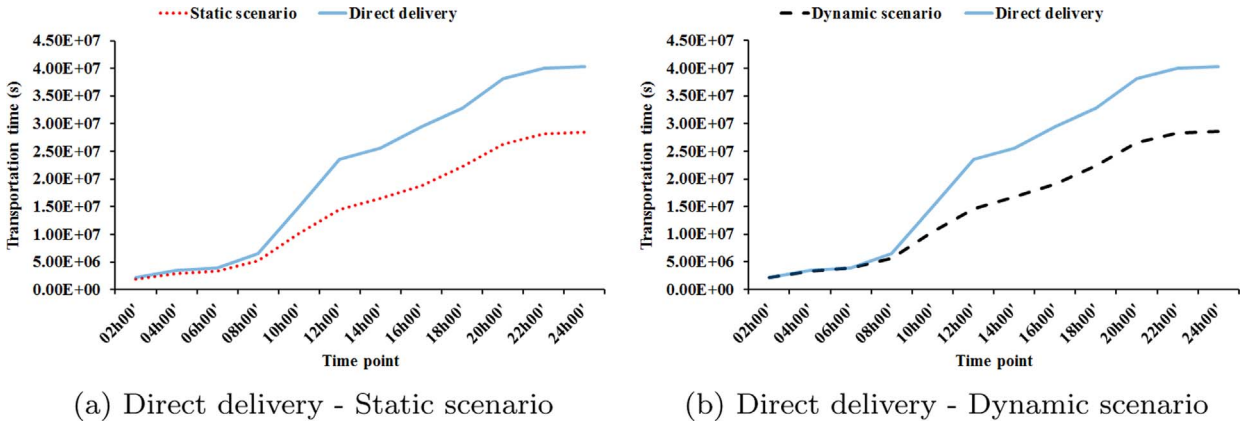

Fig. 7. Accumulating transportation times every 2 h in Dataset 3.

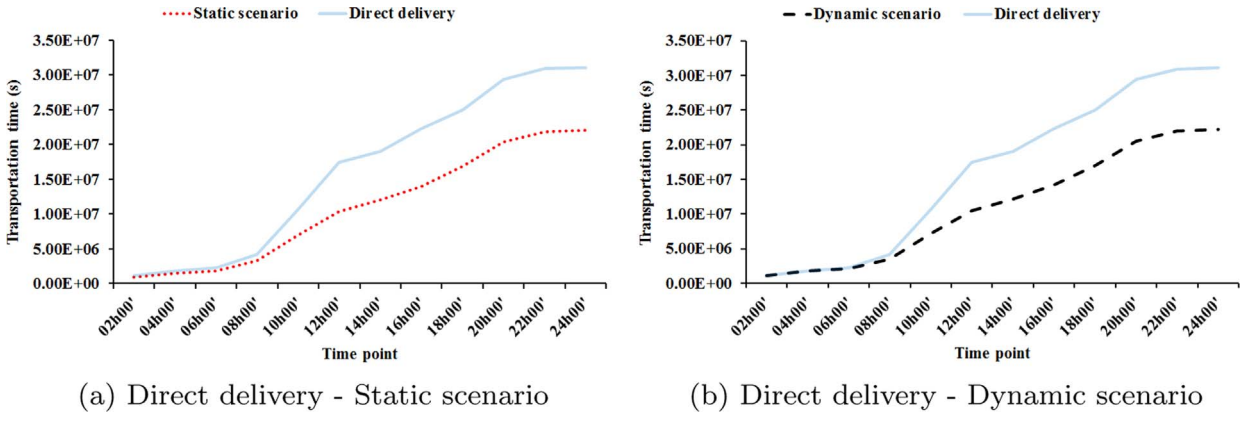

Fig. 8. Accumulating transportation times every 2 h in Dataset 4.

lines indicate the results of the static scenario; and the solid lines represent the results of the direct delivery.

Throughout a day, in the direct delivery model, the number of used taxis are dramatically increasing over time. We illustrate this effect by showing over each time window the number of new added taxis to serve upcoming requests in Figs. 3 and 4. Regarding these figures, a huge number of taxis are added to serve requests in peak-hours, especially in the direct delivery. The sharing model needs much fewer added taxis than the direct delivery in all over time windows. In the peak and partial peak hours, it is remarkable that the number of taxis used in the static scenario are clear smaller than in the dynamic scenario. That could effectively reduce the congestion in the peak and partial peak hours.

In the sharing model, a big number of requests are shared the trajectories, which considerably reduces the total traveling times. The experimentation shows that the direct delivery model totally spends more 1.36 times slower than the sharing model. The details

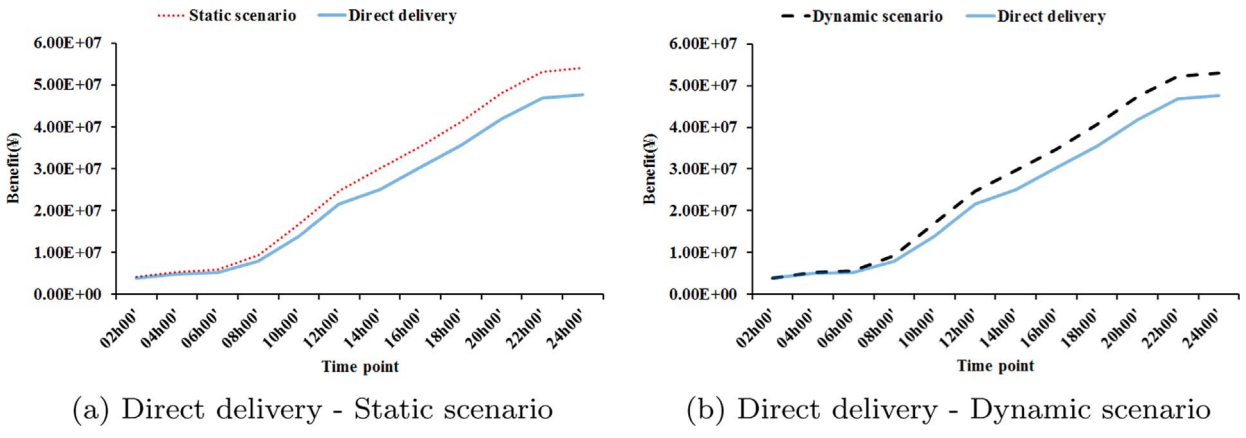

Fig. 9. Accumulating the total benefit by time every 2 h in Dataset 1.

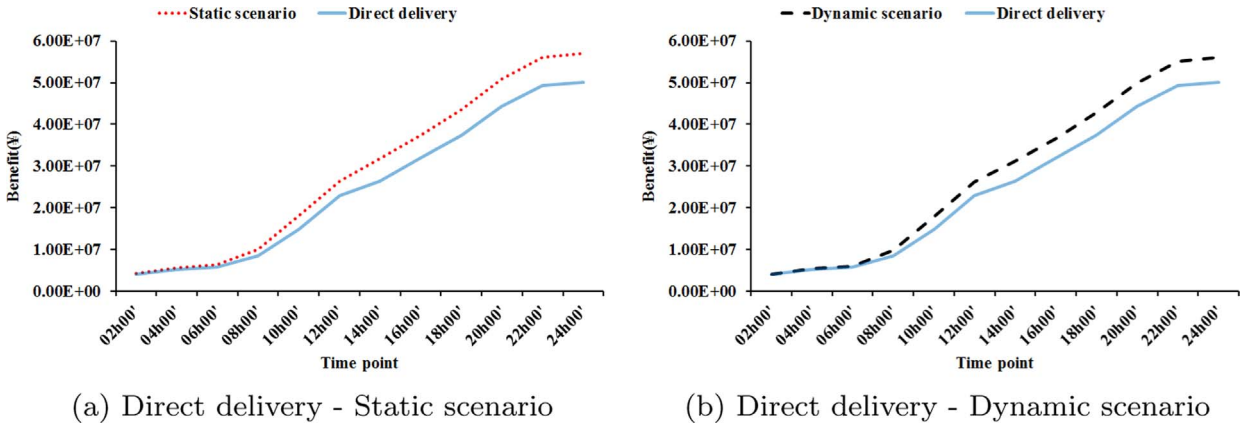

Fig. 10. Accumulating the total benefit by time every 2 h in Dataset 2.

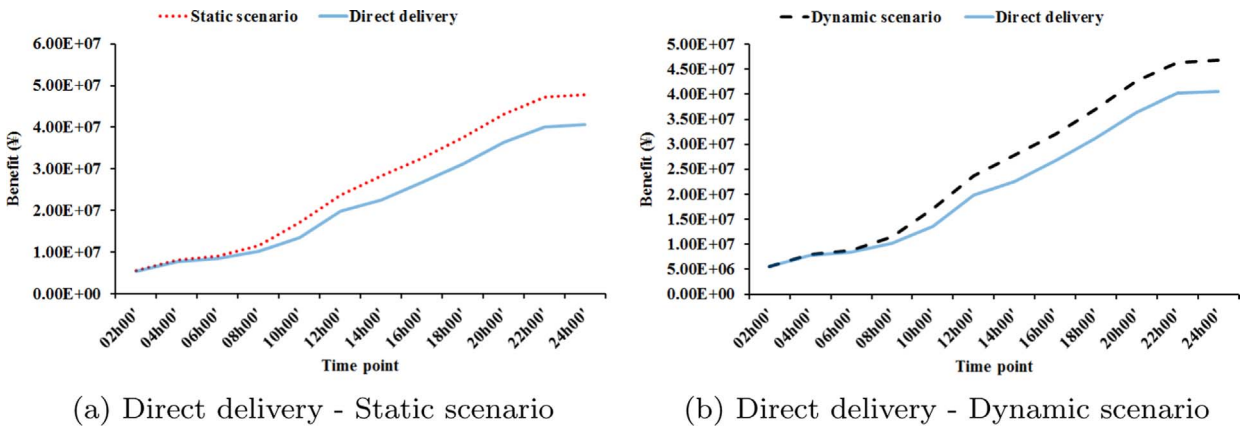

Fig. 11. Accumulating the total benefit by time every 2 h in Dataset 3.

of the accumulating traveling time are shown in Figs. 5–8.

Because of the fewer used taxis and the fewer traveling times, the sharing model gains better total benefits than the direct model. Figs. 9–12 indicate the total of benefit accumulated every 2 h during a day of both models. Regarding these figures, the accumulating total benefit of the sharing model is always bigger than that of the direct delivery in a day. The details of the total benefit over time windows, i.e., for every 2 h, are illustrated in Figs. 13–16.

The execution time reported in Tables 8–11 show the feasibility of our solution. In average, each request is processed in 0.11 s, that leads our system to be applicable directly to real-life situations.

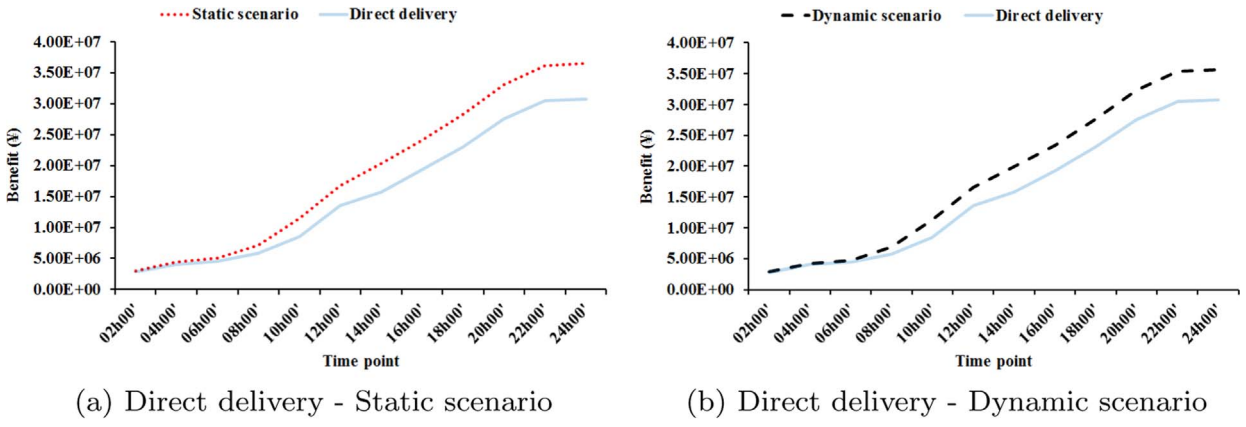

Fig. 12. Accumulating the total benefit by time every 2 h in Dataset 4.

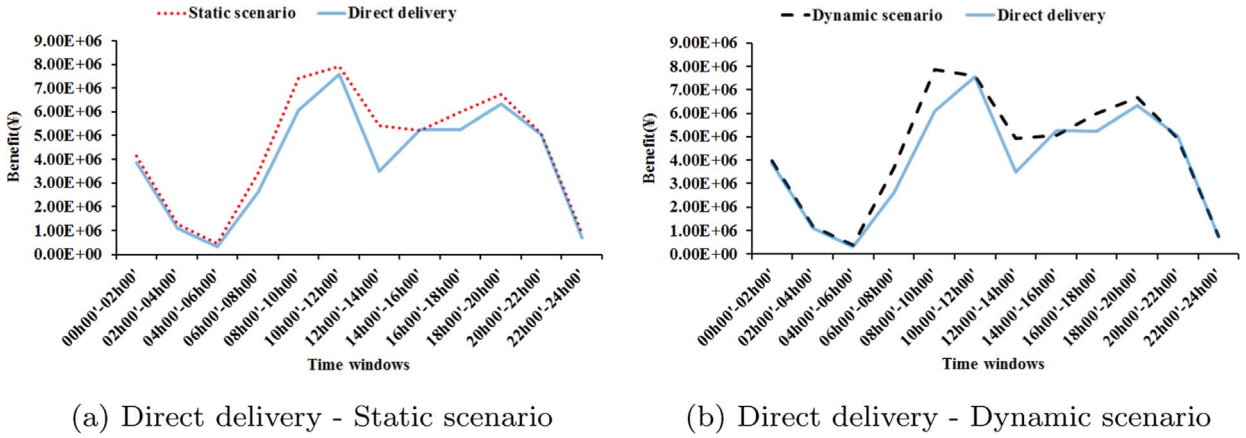

Fig. 13. The total benefit by time every 2 h in Dataset 1.

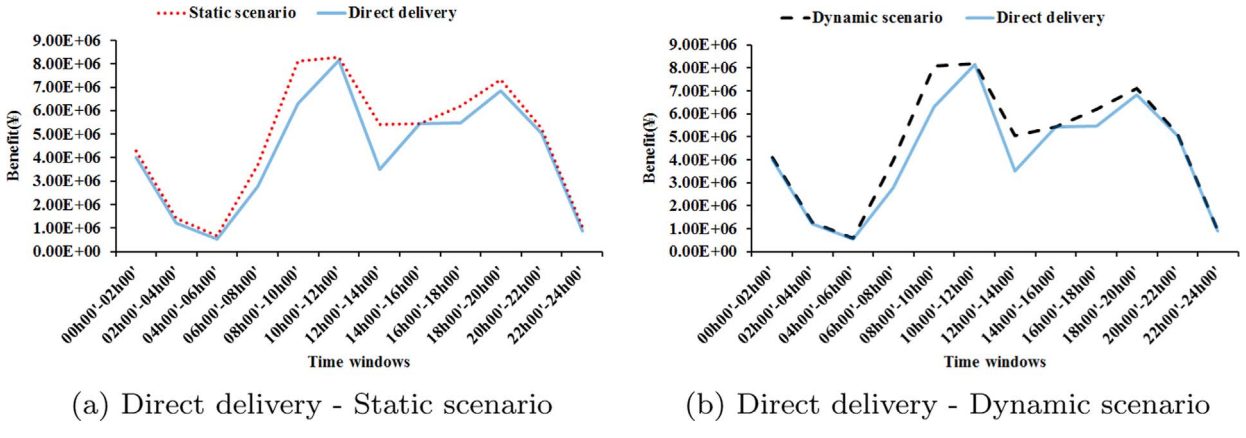

Fig. 14. The total benefit by time every 2 h in Dataset 2.

6. Conclusion and future work

This paper introduces a fully time-dependent model that allows sharing a taxi between a passenger and parcels in the urban context. Besides new realistic factors and constraints consideration, we handle congestion levels by speed windows in specific zones of a city. We presented a mathematical formulation for the problem and proposed heuristic algorithms to handle the direct delivery and the dynamic sharing model. We tested the feasibility and the efficiency of the model on a real-case dataset of Tokyo-Musen taxi requests and the Tokyo road map. The experimental results show the better performance of the sharing taxi model on all considered factors. Therefore, the sharing model has great society advantages such as saving travel cost, reducing traveling time, mitigating

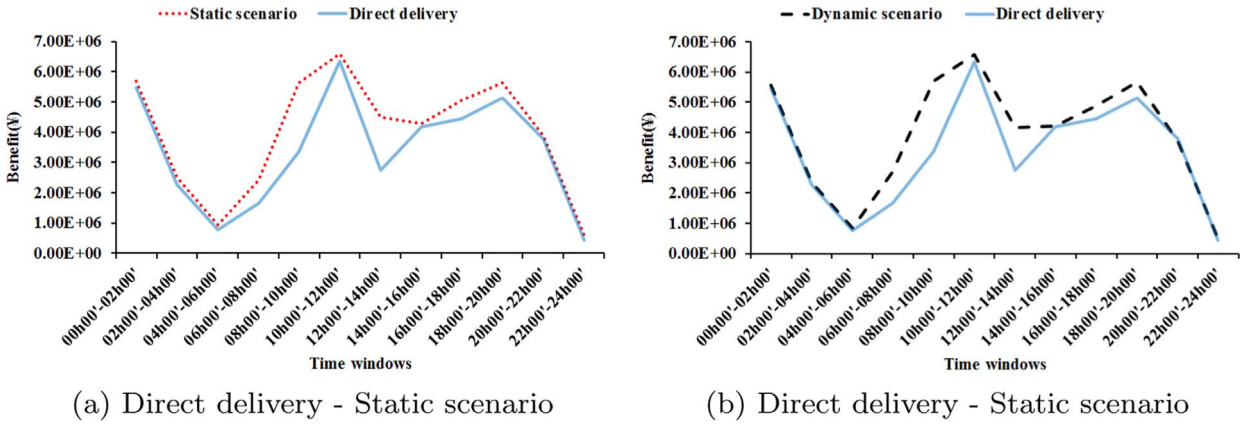

Fig. 15. The total benefit by time every 2 h in Dataset 3.

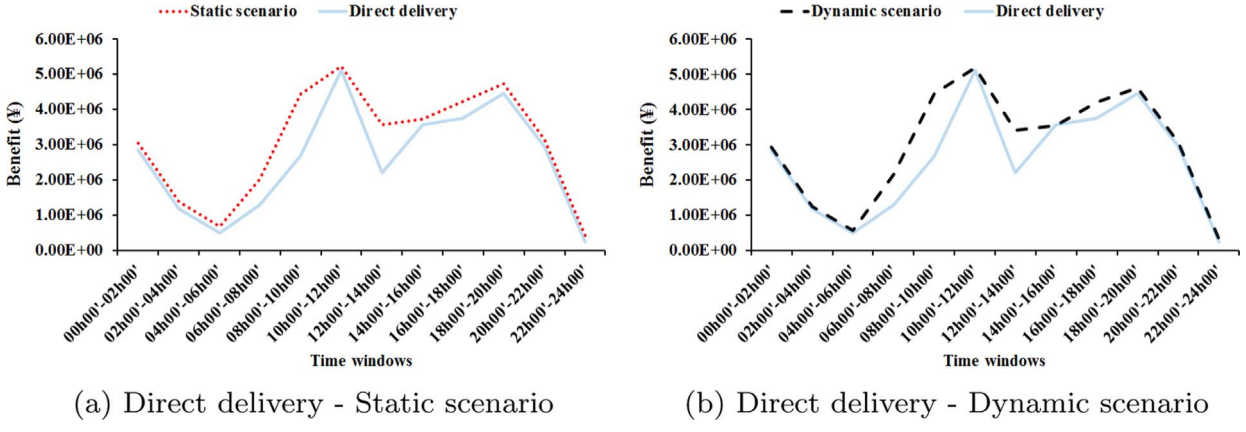

Fig. 16. The total benefit by time every 2 h in Dataset 4.

traffic congestion, conserving fuel and reducing air pollution.

This model is promising and open for some research such as extending the model (multiple objectives, city-to-city transportation, etc.); and applying meta-heuristics algorithms to improve the experimental results.

Acknowledgement

This research is funded by Vietnam National Foundation for Science and Technology Development (NAFOSTED) under grant number FWO.102.2013.04.

We would like to thank System Origin Co., Ltd., Japan for providing us the real-case data of Tokyo-Musen taxi requests.

References

[1] J.-F. Cordeau, G. Laporte, The dial-a-ride problem (darp): variants, modeling issues and algorithms, Q. J. Belgian French Ital. Oper. Res. Soc. 1 (2) (2003) 89–101. http://dx.doi.org/10.1007/s10288-002-0009-8.

[2] J.-F. Cordeau, G. Laporte, The dial-a-ride problem: models and algorithms, Ann. Oper. Res. 153 (1) (2007) 29–46. http://dx.doi.org/10.1007/s10479-007-0170-8.

[3] G.B. Dantzig, J.H. Ramser, The truck dispatching problem, Manag. Sci. 6 (1) (1959) 80–91. http://dx.doi.org/10.1287/mnsc.6.1.80.

[4] G. Clarke, J.W. Wright, Scheduling of vehicles from a central depot to a number of delivery points, Oper. Res. 12 (4) (1964) 568–581. http://dx.doi.org/10.1287/opre.12.4.568.

[5] V. Pillac, M. Gendreau, C. Guéret, A.L. Medaglia, A review of dynamic vehicle routing problems, Eur. J. Oper. Res. 225 (1) (2013) 1–11. http://dx.doi.org/10.1016/j.ejor.2012.08.015.

[6] O. Bräysy, M. Gendreau, Vehicle routing problem with time windows, part i: route construction and local search algorithms, Transp. Sci. 39 (1) (2005) 104–118. http://dx.doi.org/10.1287/trsc.1030.0056.

[7] J. Lysgaard, The pyramidal capacitated vehicle routing problem, Eur. J. Oper. Res. 205 (1) (2010) 59–64. http://dx.doi.org/10.1016/j.ejor.2009.11.029.

[8] R. Baldacci, A. Mingozzi, R. Roberti, Recent exact algorithms for solving the vehicle routing problem under capacity and time window constraints, Eur. J. Oper. Res. 218 (1) (2012) 1–6. http://dx.doi.org/10.1016/j.ejor.2011.07.037.

[9] G. Berbeglia, J.-F. Cordeau, G. Laporte, Dynamic pickup and delivery problems, Eur. J. Oper. Res. 202 (1) (2010) 8–15. http://dx.doi.org/10.1016/j.ejor.2009.04.024.

[10] C. Malandraki, M.S. Daskin, Time dependent vehicle routing problems: formulations, properties and heuristic algorithms, Transp. Sci. 26 (3) (1992) 185–200. http://dx.doi.org/10.1287/trsc.26.3.185 (URL ⟨http://dx.doi.org/10.1287/trsc.26.3.185⟩).

[11] A. Haghani, S. Jung, A dynamic vehicle routing problem with time-dependent travel times, Comput. Oper. Res. 32 (11) (2005) 2959–2986. http://dx.doi.org/10.1016/j.cor.2004.04.013 (URL ⟨http://www.sciencedirect.com/science/article/pii/S0305054804000887⟩).
[12] H.-K. Chen, C.-F. Hsueh, M.-S. Chang, The real-time time-dependent vehicle routing problem, Transp. Res. Part E: Logis. Transp. Rev. 42 (5) (2006) 383–408. http://dx.doi.org/10.1016/j.tre.2005.01.003 (URL ⟨http://www.sciencedirect.com/science/article/pii/S1366554505000293⟩).
[13] L. Fu, Scheduling dial-a-ride paratransit under time-varying, stochastic congestion, Transp. Res. Part B: Methodol. 36 (6) (2002) 485–506. http://dx.doi.org/10.1016/S0191-2615(01)00014-5 (URL ⟨http://www.sciencedirect.com/science/article/pii/S0191261501000145⟩).
[14] T.-Y. HU, C.-P. CHANG, Time-dependent dial-a-ride problems: formulation development and numerical experiments, J. East Asia Soc. Transp. Stud. 9 (2011) 690–701. http://dx.doi.org/10.11175/easts.9.690.
[15] A. Trentini, N. Mahléné, Toward a shared urban transport system ensuring passengers & goods cohabitation, Tema. J. Land Use Mobil. Environ. 3 (2) http://dx.doi.org/10.6092/1970-9870/165.
[16] N. Agatz, A. Erera, A. Savelsbergh, X. Wang, Optimization for dynamic ride-sharing: a review, Eur. J. Oper. Res. 223 (2) (2012) 295–303.
[17] M. Furuhata, M. Dessouky, F. Ordóñez, M.-E. Brunet, X. Wang, S. Koenig, Ridesharing: the state-of-the-art and future directions, Transp. Res. Part B: Methodol. 57 (2013) 28–46. http://dx.doi.org/10.1016/j.trb.2013.08.012.
[18] S. Ma, Y. Zheng, O. Wolfson, T-share: A large-scale dynamic taxi ridesharing service, in: 2013 IEEE Proceedings of the 29th International Conference on Data Engineering (ICDE), 2013, pp. 410–421 http://dx.doi.org/10.1109/ICDE.2013.6544843.
[19] N.A. Agatz, A.L. Erera, M.W. Savelsbergh, X. Wang, Dynamic ride-sharing: a simulation study in metro atlanta, Transp. Res. Part B: Methodol. 45 (9) (2011) 1450–1464. http://dx.doi.org/10.1016/j.trb.2011.05.017 (select Papers from the 19th{ISTTT).
[20] B. Li, D. Krushinsky, H.A. Reijers, T.V. Woensel, The share-a-ride problem: people and parcels sharing taxis, Eur. J. Oper. Res. 238 (1) (2014) 31–40. http://dx.doi.org/10.1016/j.ejor.2014.03.003.
[21] E. Ferguson, The rise and fall of the american carpool: 1970–1990, Transportation 24 (4) (1997) 349–376. http://dx.doi.org/10.1023/A:1004928012320.
[22] K. Kelley, Casual carpooling enhanced, J. Public Transp. 10 (4) (2007) 119–130.
[23] C. Morency, The ambivalence of ridesharing, Transportation 34 (2) (2007) 239–253. http://dx.doi.org/10.1007/s11116-006-9101-9.
[24] N.D. Chan, S.A. Shaheen, Ridesharing in North America: past, present, and future, Transp. Rev. 32 (1) (2012) 93–112. http://dx.doi.org/10.1080/01441647.2011.621557.
[25] B. Li, D. Krushinsky, T.V. Woensel, H.A. Reijers, An adaptive large neighborhood search heuristic for the share-a-ride problem, Comput. Oper. Res. 66 (2016) 170–180. http://dx.doi.org/10.1016/j.cor.2015.08.008 (URL ⟨http://www.sciencedirect.com/science/article/pii/S0305054815002051⟩).
[26] G.V. Batz, R. Geisberger, P. Sanders, C. Vetter, Minimum time-dependent travel times with contraction hierarchies, J. Exp. Algorithmics (2013) 1.4:1.1–1.4:1.43. http://dx.doi.org/10.1145/2444016.2444020.
[27] G.V. Brummelen, Heavenly Mathematics: The Forgotten art of Spherical Trigonometry, Princeton University Press, 2013.
[28] Musen Taxi Company, ⟨http://www.tokyomusen.or.jp/⟩, Website (2009).
[29] L. Japan Post Co., ⟨http://www.post.japanpost.jp/fee/in/dex.html⟩, Website (2009).