Prediction-based optimization for online People and Parcels share a ride taxis

NGUYEN Van Son
Hanoi University of Science and Technology
Academy Of Cryptography Techniques
HaNoi, VietNam
Email: sonnv188@gmail.com

Behrouz Babaki
Dept. of Computer Science
KULeuven, Belgium
Email: behrouz.babaki@cs.kuleuven.be

Anton Dries
Dept. of Computer Science
KULeuven, Belgium
Email: anton.dries@cs.kuleuven.be

PHAM Quang Dung
Hanoi University of Science and Technology
HaNoi, VietNam
Email: dungpq@soict.hust.edu.vn

NGUYEN Xuan Hoai
Hanoi University
HaNoi, VietNam
Email: nxhoai@hanu.edu.vn

***Abstract*—One of the major problems faced in the urban environment is the efficient public transportation of people and goods. Clients make a call to the company to request a transportation service for people or parcels. A good transport scheduling will bring better profits to companies while satisfying people demands and reducing negative social impact such as traffic jam and pollution. We extend the work in [10] on a people and parcel share-a-ride taxis transportation model and propose algorithms to schedule taxis that exploit prediction on requests in an online scenario.**

1. Introduction

The integration of passenger and parcel transportation has potentially large economical benefits. Such an integration reduces, for example, the impact of congestion and air pollution (Lindholm and Behrends [1]). The annual cost of congestion in the US in terms of lost hours and wasted fuel was estimated to be $124 billion in 2013 [2]. Private automobile usage is the dominant transportation mode producing carbon dioxide emissions [3]. Effective usage of empty car seats by ride-sharing may represent an important opportunity to increase occupancy rates, hence substantially increase the efficiency of urban transportation systems, potentially reducing the afore mentioned problems. Most ride-sharing models are based on the well-known Dial-a-Ride Problem [6]. A Dial-a-Ride Problem (DARP) consists of designing vehicle routes and schedules for a number of users who request pickup and drop-off points. The description of the dynamic Shared-a-Ride Problem (SARP) was proposed recently by Li et al. [10], [12] explaining the conceptual and mathematical models in which people and parcels are served by the same network. In addition, two heuristic algorithms [11], [12] are proposed to solve this problem.

In this paper, we propose a share-a-ride taxi transportation model in which people and parcels can share a ride. Our model alleviates the deficiencies of the model in [10] by adding some real-world factors. The main contributions of our new transportation model are as follows:

- We extend the model in [10] by allowing taxi to travel and stop at a parking after delivering people/parcels instead of stopping indefinitely at the last delivery location. Thus, a set of capacitated parking places is added in our model.
- In the dynamic setting, Li et al. assume that all parcel requests are known beforehand. In our model, the requests of parcels are also dynamic/on-demand. Moreover, passengers often feel uncomfortable if they have to go to pick and drop-off some parcels which are not theirs. Hence, no parcel is picked up by a taxi that already contains a customer.
- An algorithm for predicting locations of the future requests is proposed.
- We propose a new anticipatory algorithm for scheduling taxis exploiting the predicted future requests. The algorithm is experimented on a data set of real taxis in San Francisco and shown to be competitive with the one in [10].

The paper is organized as follows. In section 2, the new model for the static people-parcel share-a-ride (PPSARP) problem is given. Section 3 details our proposed method for predicting the location of future requests. Our new anticipatory method for solving the dynamic version of PPSARP is described in section 4. The experiments and comparisons are given in section 5. Section 6 concludes the paper.

2. Static PPSARP problem

The static (i.e all requests are known beforehand) PPSARP problem is formulated as follows.

Input.

- A sequence of time points of the planning $t_b, t_b + 1, \ldots, t_e$ (e.g. accurate to 1 second)

- n people requests $r_1, \ldots, r_n$ and m parcel requests $r_{n+1}, \ldots, r_{n+m}$, request r_i has pickup and drop-off points p_i, d_i ($\forall i = 1, \ldots, n+m$)
- There are K taxis $1, \ldots, K$, each taxi k starts and terminates at points s_k and t_k which refer to a physical depot ($\forall k = 1, \ldots, K$).
- Denote $P = \{p_1, d_1, \ldots, p_{n+m}, d_{n+m}\}$, $S = \{s_1, \ldots, s_K\}$, $T = \{t_1, \ldots, t_K\}$
- $\bar{d}(p)$: service duration at point $p, \forall p \in P$
- $\underline{t}(p)$ and $\bar{t}(p)$: earliest and latest allowed arrival time at point $p, \forall p \in P$
- There are Q physical parkings $1, \ldots, Q$, parking q has capacity $c(q)$. To ease the formulation in which each route visits each point at most once, we introduce logical points associated with each physical parking: $LP(q)$ is the set of logical points associated with parking q (each point of $LP(q)$ refers to the physical parking (location) q).
- Denote PK the set $LP(1) \cup \cdots \cup LP(Q)$, and $\mathcal{P} = P \cup S \cup T \cup PK$
- $d(p_1, p_2)$: the distance from p_1 to $p_2, \forall p_1, p_2 \in \mathcal{P}$
- $t(p_1, p_2)$: travel time from p_1 to $p_2, \forall p_1, p_2 \in \mathcal{P}$: $t(p_1, p_2) = \frac{d(p_1, p_2)}{\gamma}$, γ is the average speed

Variables.

- $x(p)$: the successor of point p in the routes, $\forall v \in \mathcal{P} \setminus T$
- $r(p)$: the index of taxi (route) containing point $p, \forall p \in \mathcal{P}$
- $o(p)$: the order of point p in the route $r(p), \forall p \in \mathcal{P}$
- $l(p)$: accumulated distance of the route from the start point of route $r(p)$ to $p, \forall p \in \mathcal{P}$
- $ta(p), td(p)$: arrival and departure time of taxi at point $p, \forall p \in \mathcal{P}$
- $b(q_i, t) = 1$ if there is a taxi at parking q_i at time point t, and $b(q_i, t) = 0$, $\forall q_i \in PK, t_b \leq t \leq t_e$ otherwise

Constraints.

- (1) $x(p) \neq p, \forall p \in \mathcal{P} \setminus T$
- (2) $l(x(p)) = l(p) + d(p, x(p)), \forall p \in \mathcal{P} \setminus T$
- (3) $r(p) = r(x(p)), \forall p \in \mathcal{P} \setminus T$
- (4) $o(x(p)) = o(p) + 1, \forall p \in \mathcal{P} \setminus T$
- (5) $r(p_i) = r(d_i), \forall i = 1, \ldots, n+m$
- (6) $o(p_i) < o(d_i), \forall i = 1, \ldots, n+m$
- (7) $o(p_i) = o(d_i) - 1, \forall i = 1, \ldots, n$
- (8) $ta(x(p)) = td(p) + t(p, x(p)), \forall p \in \mathcal{P}$
- (9) $td(p) = ta(p) + \bar{d}(p), \forall p \in P$
- (10) $td(p) > ta(p), \forall p \in PK$
- (11) $\underline{t}(p) \leq ta(p) \wedge ta(p) \leq \bar{t}(p)$
- (12) $b(q_i, t) = ta(q_i) \leq t \wedge t \leq td(q_i), \forall q_i \in PK, t \in \{t_b, \ldots, t_e\}$
- (13) $\sum_{q_i \in LP(q)} b(q_i, t) \leq c(q), \forall q = 1, \ldots, Q, t \in \{t_b, \ldots, t_e\}$

Constraint (1) specifies that the route must go from one point to another point. Constraint (2) relates the accumulated distance between two subsequent points on a route. This constraint also eliminates the existence of subtours. Constraint (3) specifies that a point and its successor must be on the same route. Constraint (4) relates the indices of two successive points on a route. Constraint (5) states that the pickup and delivery points must be con the same route and constraint (6) specifies that the delivery point of a request must be after the pickup point of that request. Constraint (7) states that the delivery point of a people request must be right-after the pickup point. It means that the people request must be delivery in a direct way without interruption. In the fashion, passengers do not feel any inconvenience (they travel like in traditional taxis). Constraints (8)–(10) relates the arrival and departure times from each point of the routes. Constraints (12)–(13) present the constraint on the capacity of parkings.

Objective. The objective of the problem is to maximize the total benefit. In our model, the passenger requests are served in a direct way as in traditional taxis (no interruption during the passenger delivery), thus there are no discounts for passengers. The total benefit is equal to the total revenue minus the fuel cost (the fuel cost is proportional to the total travel distance). Hence, the objective to be minimized is the total travel distance: $f = \sum_{k=1}^{K} l(t_k)$

3. Request Prediction

We propose a predictive algorithm to determine the best location to park a taxi when no immediate requests need to be handled. It estimates the time and location of new requests based on past information and can also be used to learn distributions of parcel and people requests.

First, the city is partitioned into a set of n areas - $R = \{r_1, \ldots, r_n\}$. Each 24-hour is binned into $P = \{p_1, \ldots, p_m\}$ of m time periods with equal lengths. Let $Z_{ij}^{(d)}$ be the random variable representing the number of requests coming from area $i \in R$ during period p_j in day d. The training data for the request distribution is the observed values of $Z_{ij}^{(d)}$ for all periods and areas in a number of past days.

Since each $Z_{ij}^{(d)}$ is discrete and non-negative, it is assumed that it follows a Poisson distribution, which is common for count data [13]. Therefore, the probability of observing k requests is given by:

$$P(Z_{ij}^{(d)} = k; \lambda) = \frac{e^{-\lambda} \lambda^k}{k!} \tag{1}$$

where λ is the parameter (i.e. rate) of the distribution.

To learn the unknown rate of a Poisson distribution from past observations, we use the *maximum likelihood* principle which states that the optimal parameter for a distribution is the one that maximizes the likelihood of observed data. For a Poisson variable Z with past observations $z_1, \ldots, z_n$, the maximum likelihood estimate for the rates is:

$$\lambda^* = \frac{\sum_{i=1}^{n} z_i}{n} \tag{2}$$

To simplify the problem, we can assume that the rates only depend on period and area, and hence are fixed over all days. The optimal rates can be easily computed by:

$$\lambda_{ij}^{*} = \frac{\sum_{d=1}^{D} Z_{ij}^{(d)}}{D} \quad (3)$$

To capture the variations of traffic in different days, an extra dimension is added to the above model to differentiate between the seven days of a week. Denote $\delta(d)$ as the day of week for day d. In this model we have a rate λ_{ij}^{w} for each triplet of area i, period j, and day of week w. Again, the optimal rates can be computed as:

$$\lambda_{ij}^{*(w)} = \frac{\sum_{d \in D_w} Z_{ij}^{(d)}}{|D_w|} \quad (4)$$

where $D_w = \{d \in \{1, \ldots, D\} | \delta(d) = w\}$.

4. Online PPSARP problem

In online scenarios, information on requests is revealed online during the plan execution. A control center manages status and locations of taxis, receiving information on incoming requests and making decisions. Algorithm 1 details the framework for online execution. The time horizon T contains discrete time points (e.g, 1 second). Line 2 initializes the prediction information in which P_f is the set of predicted request pickup points in the f^{th} time frame ($\forall f = 0, \ldots, \overline{f}$). These points, called popular points, are taken randomly based on the information from the prediction method in Section 3. Each day is divided into a sequence of time frames of equal length (e.g, every 15 minutes). Line 3 updates the queues Q of people and parcel requests arriving within $[T, T + \Delta T]$. Line 6 updates status of taxis Lines 7–13 assign an appropriate taxi for each people/parcel request in Q and update the itinerary of the taxi. Whenever a request is received, the algorithm will insert it into the itinerary of a selected taxi and change this itinerary (i.e., re-order request points and exchange sub-itineraries with other itineraries), and finally, use prediction information about requests to direct the taxi to a chosen parking place.

```
1  T ← t_b;
2  Initialize P_f, ∀f = 0, f̄;
3  Q ← receiveRequests(T);
4  T ← T + ΔT;
5  while T < maxSimulationTime do
6  |   Update status of taxis;
7  |   foreach request r ∈ Q do
8  |   |   ⟨k, j⟩ ← FindNearestAvailableTaxi(r, T);
9  |   |   if ⟨k, j⟩ ≠ ⊥ then
10 |   |   |   InsertRequest(r, k, j, P);
11 |   |   |   ExchangeImprovement();
12 |   |   end
13 |   end
14 |   Q ← receiveRequests(T);
15 |   T ← T + ΔT;
16 end
```

Algorithm 1: Framework for Online Plan Execution

4.1. Itinerary representation

A schedule s_k of a taxi k is represented by a sequence $\langle v_0^k, r_1^k, r_2^k, \ldots, r_{l_k}^k, p_k \rangle$ in which v_0^k is the current point, $r_1^k, \ldots, r_{l_k}^k$ are request points (pickup or delivery) and p_k is a parking. A service plan of a taxi k is an itinerary I_k which consists of a sequence of taxi traversal points including the points of s_k and the road points between any two consecutive points in s_k. I_k also contains information about arrival time, departure time and action at each point of the sequence. In this paper, we consider the problem of large-size realistic road networks. The computation of shortest paths between two points takes time. Hence, in our algorithm, we use approximation distances (e.g, the manhattan distance) for deciding the schedule (i.e., the sequence of request points), but the itinerary employs shortest path and real distance on the road network. We denote:

- $l(I_k)$: the length (number of points) of itinerary I_k
- $p(I_k, i)$: the i^{th} point of the itinerary I_k
- *prefix*(I_k, i): First i points of I_k
- *suffix*(I_k, i): Last $l(I_k) - i + 1$ points of I_k
- $d(u, v)$: the length of the shortest path from u to v in the road network
- $t(u, v)$: the minimum travel time along the shortest path from u to v.
- *idx*(I_k, p): index of point p in itinerary I_k
- *pickup*(r): the pickup point of request r
- *delivery*(r): the delivery point of request r
- $pk(I_k)$: the parking of taxi k.
- For each index i in I_k: *ta*(I_k, i) and *td*(I_k, i) is arrival and departure time of taxi at $p(I_k, i)$

4.2. Scheduled point indices

When an a request arrives at each time point, the system performs a decision to reschedule itineraries of taxis. The decision takes a time Δ. After Δ, the status of a taxi (e.g., location) changes. When rescheduling, there exists a minimum index j of I_k, called the scheduled point index, such that *prefix*(I_k, j) cannot be changed. In the addition, if there are people on board, then they must be delivered before picking up other people as they do not want to share a ride.

4.3. Schedule Re-optimization

A schedule of a taxi k might be re-optimized by re-ordering some points of s_k. We denote *opt*$(v_0, \langle x_1, \ldots, x_k \rangle)$ the function that returns the permutation $x_1^*, x_2^*, \ldots, x_k^*$ of $x_1, \ldots, x_k$ (in which $x_1, \ldots, x_k$ are request points and v_0 is a road point or a request point) such that $d'(\langle v_0, x_1^*, x_2^*, \ldots, x_k^* \rangle)$ is minimal w.r.t. the problem constraints. It returns $\perp$ if the constraints are violated.

4.4. Itinerary Establishment

With a schedule s_k, we denote *itinerary*(s_k) as the function that computes and returns I_k from s_k containing full information: sequence of points, arrival and departure times at each point. The path between two consecutive points of s_k is the shortest path on the road network. The arrival and departure times are computed using the speed of taxis. The service duration is assumed to be 1 minute (pickup and delivery).

4.5. Algorithms

We describe in this section the details of algorithms for making decision of schedule of taxis.

When a new request appears, we find the nearest taxi and the maximal index of the taxi's itinerary that can pick up the new request r. Our algorithm find the nearest point of *suffix*(I_k, j) of taxi k which can pick up within the time window of r (where, j is the scheduled point index of taxi k at this time point). This is different from what the available taxis in [11] are defined. Our algorithm lead to a diversification of the search process.

1 $v_0^k \leftarrow p(I_k, j)$;
2 $I_k^1 \leftarrow$ *prefix*(I_k, j);
3 $i \leftarrow$ minimal index such that *idx*$(I_k, r_i^k) > j$;
4 *seq* $\leftarrow$ *opt*$(v_0^k, \langle r_i^k, \ldots, r_{l_k}^k,$ *pickup*$(r),$ *delivery*$(r)\rangle)$;
5 **if** *seq* $=\perp$ **then**
6 | **return** I_k;
7 **else**
8 | $I_k^2 \leftarrow$ *itinerary(seq)*;
9 | $p_l \leftarrow p(I_k^2, l(I_k^2))$;
10 | $(p, p_k) \leftarrow$ FindAPopularPointAndParking$(p_l, td(p_l), P)$;
11 | **if** $p \neq \perp$ **then**
12 | | $I_k^3 \leftarrow$ *itinerary*(p_l, p, p_k);
13 | **else**
14 | | $I_k^3 \leftarrow$ *itinerary*(p_l, p_k);
15 | **end**
16 | **return** $I_k^1 :: I_k^2 :: I_k^3$;
17 **end**

Algorithm 2: InsertRequest(r, k, j, P)

Algorithm 2 inserts a request r into the itinerary of taxi k with scheduled point index j. The new itinerary of taxi k consist of the prefix, suffix, a near popular point and a new parking. Lines 3–4 of Algorithm 2 compute the optimized sequence consisting of the suffix part and new serving points. If no constraint satisfying permutation is found, the new request is rejected. Lines 10 finds a near popular point p and the parking p_k which is not too far from p, has the highest score. The so-called near point has a short unload travel distance (see Algorithm 3). However, we do not assign a popular point and a neighbourhood of this point to two taxis. Therefore, either the popular point and nearest parking or only nearest parking are added into taxi's itinerary.

The score of a parking p_k at the period f is:

$$score(f, p_k) = \frac{nDT(f, p_k)}{nPP(f+1, p_k)}$$

where:

- $nDT(f, p_k)$: the number of taxis which have departed from the parking up to the current period f.
- $nPP(f+1, p_k)$: the number of popular points which are near the parking from period 0 to $f+1$ (the distance is less than θ_2).

The detailed computation is shown in Algorithm 3.
Lines 2–3 in Algoritm 3 find the nearest parking pk^* and

1 $f \leftarrow$ next frame of the frame containing time point t;
2 $pk^* \leftarrow$ argMin$_{pk \in PK}(d(p, pk))$;
3 $dk^* \leftarrow d(p, pk^*)$;
4 $pk^* \leftarrow$ argMax$_{pk \in PK}(score(pk) | d(pk, p) \leq \theta_1 * dk^*)$;
5 $dk^* \leftarrow d(p, pk^*)$;
6 $p^* \leftarrow$ argMin$_{pp \in P_f}(d(p, pp) + d(pp, pk^*))$;
7 **if** $d(p, p^*) + d(p^*, pk^*) > \theta_1 \times dk^*$ **then**
8 | **return** $(\perp, pk^*)$;
9 **else**
10 | $R \leftarrow \{pp \in P_f \mid d(pp, p^*) \leq \theta_2\}$;
11 | $P_f \leftarrow P_f \setminus R$;
12 | **return** (p^*, pk^*);
13 **end**

Algorithm 3: FindAPopularPointAndParking(p, t, P)

the distance from the last point in taxi's itinerary to the parking dk^*. Lines 4–5 update the parking pk^* with the high score of pk, $\forall pk \in PK$ such that the distance is not greater than $\theta \times dk^*$. Line 6 finds a popular point where the unload distance is minimal. If the total unload travel distance is less than $\theta_1 \times dk^*$, the taxi is allowed to go to the popular point. However, we cannot give some taxis to the same a popular point. Thus, Lines 10–11 remove some points in considered popular area (in a circle area has the radius θ_2).
After we establish the itinerary of the taxi, we used a greedy algorithm to re-organize the remaining requests of all taxis. In the exchange improvement function, we try to find the best exchange of the subsequence between two taxis in the hope of improving the benefit. An example is described in Figure 1.

5. Experiments

In our work, trace of taxis of San Francisco was used. For training the prediction model, we use the trace from 07-2005 to 07-2006. For experimenting the algorithms, we use the trace in 03-2010. The pickup and delivery points were extracted from the taxi traces. One half of the people

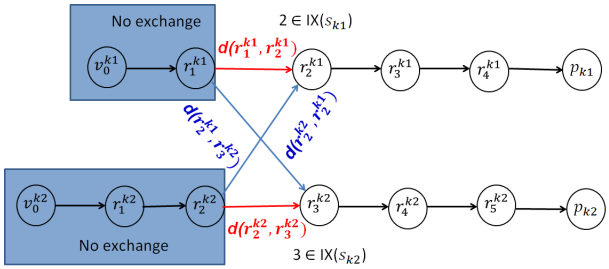

Figure 1. The exchange improvement algorithm.

requests is randomly chosen and converted into parcel requests with relaxed time window. The time call of a request is specified to be the pickup time subtracting 10 minutes. The late pickup time is specified to be the pickup time plus 15 minutes. The late delivery time is specified to be the time call plus the shortest travel time from the pickup to the delivery plus 30 minutes.

5.1. Simulation design

The simulation is designed to be generic to test the efficiency of different algorithms and can be extended for other dynamic VRPs. We experiment with 4 algorithms:

- Our algorithms are PPSARP and PPSARP+prediction: PPSARP will direct the taxi toward a nearby parking, and PPSARP+prediction will score parkings based on predicted information and direct the taxi toward the predicted point and the parking of the highest score.
- The LiSARP is the dynamic algorithm in [10]. Furthermore, we constructed a variation of LiSARP by combining with the prediction - (LiSARP+prediction). Based on Li's model, we increased the number of parkings for the making schedule. The taxis are directed to the predicted point and the parking of the highest score.

5.2. Settings

This parameter setting is as follows, where the passenger request positions are based on the given taxi traces.

- Road network: San Francisco (from OpenStreetMap with 131245 nodes and 259792 arcs).
- 34 parkings are randomly chosen based on GoogleMaps. Parkings capacity is 39 vehicles.
- 1000 taxis: depots are at parkings
- Start working time: 0h
- Terminate working time: 24h
- Speed 40km/h
- $\theta_1 = 1.5$
- $\theta_2 = 5$km

5.3. Experimental results

Test instances are solved on an Intel(R) Core(TM) i7-4790 CPU @ 3.660 GHz, CPU 16 GB RAM computer. PPSARP, PPSARP+prediction, LiSARP and LiSARP+prediction were implemented in JAVA. We have two experiment scripts. For the first script, the system serves all requests in day. However, time window of parcel requests has tight restriction. For the second, total served requests are less but time frame of parcel requests is wider. Each parcel request is associated with a time window: the lower bound is drawn uniformly at random from 8:00 - 16:00 h (accurate to 1 minute), the upper bound equals to 24:00.

5.3.1. Script 1: Tight restriction of parcel requests. The number of all requests is listed in Table 1.

Day	nPeopleRequests	nParcelRequests
Day 1	6167	6167
Day 2	5768	5768
Day 3	6646	6646
Day 4	6673	6673
Day 5	7210	7210
Day 6	8229	8229
Day 7	8238	8238
Day 8	5166	5166
Day 9	4147	4147

TABLE 1. THE NUMBER OF ALL REQUESTS IN SCRIPT 1.

Although parcels are all known beforehand, the numbers of served requests of LiSARP, LiSARP+prediction were less than those of PPSARP and PPSARP+prediction. Thus, the total benefits of PPSARP and PPSARP+prediction are greater than both LiSARP and LiSARP+prediction. The results are shown in Figure 2.

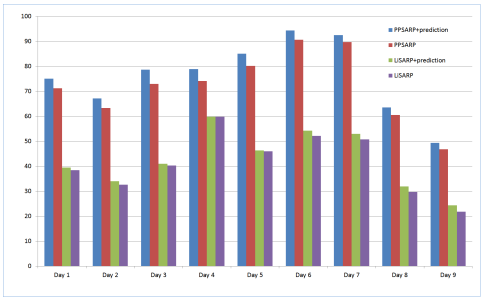

Figure 2. The total benefit of four planners of the first script.

5.3.2. Script 2: Wider time window script. Table 2 shows the number of all requests (setting time of Script 2). Although this wider time window value is of great advantage to LiSARP and LiSARP+prediction, their benefits were still less than the others PPSARP and PPSARP+prediction (see Figure 3). At a time point, we only make schedule for a location and re-optimize taxi's itinerary. If parcel requests are known beforehand, the people serving location is inserted to the taxi's itinerary

Day	nPeopleRequests	nParcelRequests
Day 1	2352	1366
Day 2	2692	1535
Day 3	2784	1601
Day 4	2600	1517
Day 5	2718	1633
Day 6	2730	1876
Day 7	2760	2037
Day 8	2317	1334
Day 9	895	299

TABLE 2. THE NUMBER OF ALL REQUESTS IN SCRIPT 2.

before we make schedule some suitable parcels for the taxi.

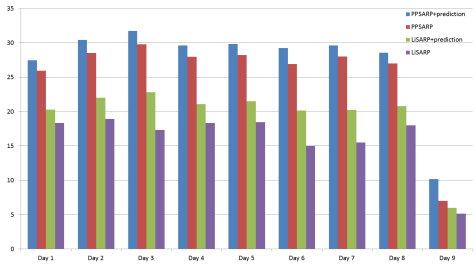

Figure 3. The total benefit of four planners of the second script.

5.3.3. Growth rate of benefit. Figures 4 depicts the accumulated benefits of all four planners. It can be seen that, PPSARP and PPSARP+prediction have higher growth rates. Locally, we see that the benefit of LiSARP and LiSARP+prediction has a declining phase. Globally, LiSARP and PPSARP without prediction capability often accept new requests located too far from taxi's position resulting in lower growth rates.

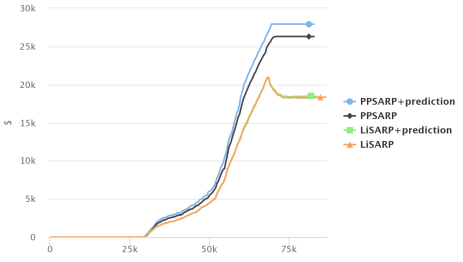

Figure 4. the growth rate of four planners (Day 1 of the second script).

6. Conclusion

In this paper, we consider the problem of scheduling people and parcel sharing a ride with taxis. We extend the model proposed in [10] by taking into account the reality that a taxi should go to a specified parking place after delivering the last request of its scheduled itinerary. We derive a model for predicting regions of the road network where requests are likely to appear in each period. New algorithms for solving the online people and parcel share a ride problem are designed utilizing the prediction information. We evaluate our proposed algorithms on data sets extracted from taxi traces of San Francisco, combined with road network of San Francisco (from OpenStreetMap.org). Experimental results show that our algorithms compute better solutions than the algorithm in [10] in term of total benefits. The results also confirm the benefit of using prediction information in the context of dynamic PPSARP.

For future works, we investigate other prediction methods as well as different approaches for exploiting prediction information. We also analyse more deeply the successive and failure cases of prediction.

Acknowledgments

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

References

[1] Lindholm, M. and Behrends, S. (2012). Challenges in urban freight transport planning: a review in the Baltic Sea Region. Journal of Transport Geography, 22:129–136

[2] http://inrix.com/press-releases/2739/

[3] Hensher, D. A., "Climate change, enhanced greenhouse gas emissions and passenger transport what can we do to make a difference?," Transportation Research Part D: Transport and Environment, vol. 13, no. 2, pp. 95–111, 2008.

[4] Brunekreef, B. and Holgate, S. T., "Air pollution and health," Lancet, vol. 360, no. 9341, pp. 1233–1242, 2002

[5] Kunzli, N., Kaiser, R., Medina, S., Studnicka, M., Chanel, O., Filliger, P., Herry, M., Horak, F., Puybonnieux-Texier, V., Quenel, P., Schneider, J., Seethaler, R., Vergnaud, J. C., and Sommer, H., "Public-health impact of outdoor and traffic-related air pollution: a european assessment," Lancet, vol. 356, no. 9232, pp. 795–801, 2000.

[6] Cordeau, J., & Laporte, G. (2003). The dial-a-ride problem (DARP): Variants, modeling issues and algorithms. 4OR: A Quarterly Journal of Operations Research, 1, 89–101

[7] Trentini, A., & Malhéné, N. (2010). Toward a shared urban transport system ensuring passengers & goods cohabitation. TeMA – Trimestrale del Laboratorio Territorio Mobility Ambiente, 3(2), 37 – 44.

[8] Agatz, N., Erera, A., Savelsbergh, M., & Wang, X. (2012). Optimization for dynamic ride-sharing: A review. European Journal of Operational Research, 223(2), 295303

[9] Furuhata, M., Dessouky, M., Ordonez, F., Brunet, M., Wang, X., & Koenig, S. (2013). Ridesharing: The state-of-the-art and future directions. Transportation Research Part B: Methodological, 57, 2846

[10] Li, B., Krushinsky, D., Reijers, H.A., Van Woensel, T., 2014. The share-a-ride problem: people and parcels sharing taxis. Euro. J. Operat. Res. 238 (1), 31–40

[11] Ngoc Quang Nguyen, Nguyen Viet Dung Nghiem, Phan Thuan DO, Khac Tuan LE, Minh Son Nguyen, Naoto MUKAI. People and parcels sharing a taxi for Tokyo city. Proceeding SoICT 2015 Proceedings of the Sixth International Symposium on Information and Communication Technology. 90-97

[12] B. Li, D. Krushinsky, W. Van, and H. Reijers. An Adaptive Large Neighborhood Search Heuristic for the Share-a-Ride Problem. Computers & Operations Research, 66:170180, 2016

[13] Cameron, A. C. & Trivedi, P. K., 2013, Regression Analysis of Count Data (Econometric Society Monographs), Cambridge University Press.