Sample-based Prophet for Online Ride-sharing with Fairness

Baoju Li, En Wang, Funing Yang*, Yongjian Yang, Wenbin Liu, Zijie Tian, Junyu Liu and Wanbo Zheng
College of Computer Science and Technology, Jilin University, China
Email: {libj21,tianzj21,junyu19}@mails.jlu.edu.cn, {wangen,yfn,yyj,liuwenbin,zwb}@jlu.edu.cn

2022 18th International Conference on Mobility, Sensing and Networking (MSN) | 978-1-6654-6457-4/22/$31.00 ©2022 IEEE | DOI: 10.1109/MSN57253.2022.00021

***Abstract*—The prosperity of industrialization urges modern ride-sharing platforms to gain profit from efficient management of their resources. Although ride-sharing allows sharing costs and promotes the traffic efficiency by making better use of vehicle capacities, dealing with large amounts of online taxi orders is an inevitable challenge in the current transportation systems, where all drivers have to make immediate and irrevocable decisions about whether to accept current order in a parallel way. Furthermore, in order to achieve global fairness, it is critical for an algorithm to function whenever the first order gets online without any observation stage. In this paper, we formulate this online user selection problem as a prophet inequality for independent identically distributed random variables from an unknown distribution. We construct a sample set to avoid the observation stage in an online decision process. Considering the driver-centered ride-sharing scenario, a route schedule algorithm and a sample-driven algorithm with a guarantee of lower bound are proposed to concurrently guide taxi drivers to accept taxi orders and achieve global fairness at the meantime. Finally, we conduct extensive evaluations based on three real-world data sets. The results verify the effectiveness of our proposed algorithm on improving the overall profit, increasing accepted orders and reducing the unoccupied time of the vehicle under the valid ride-sharing constraints.**

***Index Terms*—Online algorithm, Prophet inequality, Ride-sharing, Route planer**

I. Introduction

Taxi industry ride-sharing[1], which means the collaborative commuting of two or more people in a single taxi is a burgeoning mode of sharing economy. Recently, with the start of industrialization and the economic society developing rapidly in some developing countries, massive populations have been moving from rural to urban settings[2]. Inevitably, the clustered urbanization development arouses some major concerns like rising petrol prices, limited public transport resources, and traffic congestion in rush hours. As a consequence, the demand of taxi ride-sharing services, which aims to pickup taxi seekers with similar itineraries and time schedules and dynamically rearrange the taxi route to benefit both taxi drivers and passengers, has grown dramatically in recent years[3]

In public transportation system, a major characteristic of the participants is that they are clustered in some predefined locations waiting for vehicles to pick up and then drop them off in desired locations[4]. Therefore, it is convenient for passengers to participate in ride-sharing if we can construct

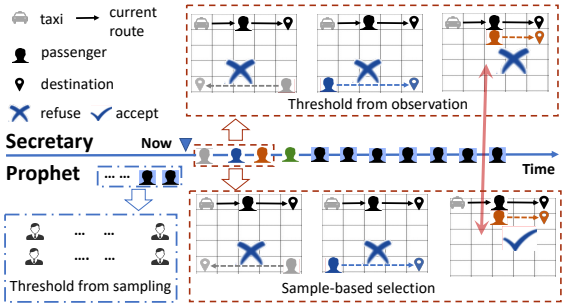


Fig. 1: Prophet inequality VS secretary problem

an understandable ride-sharing system and make it profitable to taxi drivers. Existing ride-sharing service providers like Uber and Didi are platform-centered[5], which focuses on the benefits of service platform. Orders are dispatched to the taxi drivers, taking them to a dilemmatic situation about whether to accept the order. We argue that a **driver-centered** ride-sharing system that the platforms provides parallel and rapid guidance for taxi drivers is in harmony with current economic society. In the drivers' points, apart form increasing total profit, ride-sharing can reduce the unoccupied time of their taxis. In order to let drivers make independent decision based on the guidance provided by platform, the ride-sharing system should have an understandable selection threshold and the proposed method must have a guarantee of lower bound to ensure drivers' income. However, huge difficulties must be faced to build an understandable public ride-sharing system to concurrently guide taxi drivers to select online taxi orders, which is an online decision problem.

In order to pick up proper passengers to maximize the total profit of taxi drivers, there are still several challenges. Some of the existing research[6],[7] considers the passenger selection problem in an offline scenario and using greedy matching approach to solve it. To broaden this appeal in a real-life scenario, **the first challenge** is that we must deal with a large amount of online taxi calls concurrently, which means that the taxi seekers get online in real time, that is, taxi drivers have to make instantaneous decisions without any future knowledge at a same time point. Drivers whose taxi is not totally occupied must desire to pick up a profitable passenger that could maximize their working profit. Some existing works[8],[9] formulate the online passenger selection problem as a secretary

* Funing Yang is the corresponding author.

978-1-6654-6457-4/22/$31.00 ©2022 IEEE
DOI 10.1109/MSN57253.2022.00021

Authorized licensed use limited to: JILIN UNIVERSITY. Downloaded on April 02,2024 at 07:16:34 UTC from IEEE Xplore. Restrictions apply.

problem and by discarding the first 1/e fraction of the online candidates as the observation stage, the algorithm selects the first online seeker that is greater than any of the discarded part. The optimal solution of this algorithm guarantees a result within a factor 1/e of the expected maximum among all the initially online unobserved values. However, we argue that although the solution of secretary problem achieves a tight bound, it has an observation stage that is not fair for these early coming. **The second challenge** is that, in order to guarantee global fairness for all taxi seekers, the proposed algorithm for online passenger problem should function whenever the first call get online. Consequently, we use the same temporal and spatial data in the past to calculate a threshold.

Fig.1 indicates the fairness issue in an online ride-sharing problem. Suppose a sequence of passengers getting online one by one. If the problem is formulated as secretary problem, it firstly observes the three taxi orders to set a proper threshold for later selection stage. In this case, the third one is the optimal but it is set as a threshold in the observation stage. Then it goes over every passenger starting from the fourth and begin to select, which unfortunately makes the first three taxi orders no chance to be accepted. However, our proposed method starts comparing to the threshold in a sample set at the beginning of the selection process and accepts the third order. Aiming at providing a rapid guidance for the drivers in a parallel way, we use organized history data to construct a sample set and make every order have a chance to be accepted.

Given the above, we are considering a real-world online taxi ride-sharing scenario, which involves a passenger selection standard based on a complicated passenger's utility including the passenger's discounted payment as the revenue part and the taxi's dynamically rearranged total route distance as the cost part. Moreover, in such a dynamic situation where taxi drivers have to make rapid decisions concurrently to pick up the selected passenger if there is enough capacity and constantly update their route according to current passengers on board, the passengers to be picked up and to be dropped off. As consequence, an online passenger selection algorithm that takes the complicated object function and the dynamic scenario into consideration, and most importantly, without the observation stage, a global fair method is urged to provide concurrent guidance for drivers to accept orders so as to maximize the total profit and minimize the unoccupied time during their working hours.

In this paper, we formulate this problem into the prophet inequality, which samples from history data to set a selection standard so that drivers can be ready to concurrently chose whenever the first taxi call get online and then dynamically re-adjust the selection standard along with the whole ride-sharing progress. Our goal is dynamically select a passenger to maximize taxi driver's profit in the real world taxi ride-sharing scenario. We summarize our contributions as follows:

- We formulate the online taxi ride-sharing problem as a prophet inequality for independent identically distributed random variables from an unknown distribution.
- To address the formulated problem, we propose a sample-based online selection algorithm without any observation stage to parallelly guide drivers to accept orders, and it is proved that the proposed algorithm has a guarantee of lower bound.
- We conduct several examinations based on real world dataset, proving that our algorithm can increase drivers benefits and decrease unoccupied rate.

TABLE I: Main notations

Notation	Meaning
c_i,p_i,r_i	the ith taxi, passenger, route
$p_i.ol$,$p_i.dl$	pick-up point, destination of the ith passenger
x_i	taxi event
PR, CR	passed route, current route
$CP(c)$	current passengers in car c
$DP(c)$	passengers already be dropped off
$CR(c)$	current route schedule of car c
REV	the revenue given by passengers in $DP(c)$
$PR(c)$	sequence of finished ride-sharing event
$dis(a,b)$	euclidean distance between a and b
$DIS(c)$	the total travel cost of taxi c
$PROFIT$	the profit of taxi driver

II. Problem Formulation

We proceed to define the online passenger selection problem in taxi ride-sharing. The main notations are listed in Table I.

Definition 1 (Taxi seeker): A taxi seeker p=(i,ot,ol,dl) is getting online at time p.ot on the origin location of p.ol with the destination of p.dl. Both the descriptions of p.ol and p.dl is in the form of coordinates of latitude and longitude.

Definition 2 (Taxi): A taxi c=(v,t,l,r,p,c) is an entity that is able to pick up passengers and drop them off on requested locations with a supposed constant velocity c.v. Taxi c submits its live update of current location c.l corresponding to current time c.t. Further more, taxi c dynamically records its ongoing route schedule c.r and current passengers c.p in car within the passenger capacity c.c.

Definition 3 (valid route schedule): A valid taxi's route schedule $c.r$ is comprised of some locations, which spontaneously separate the route into several segments. In each successive segment, the taxi continuously receives orders from taxi seekers p. Knowing distance between the origin location and destination, which also means the predicted fee, each taxi decides whether to accept the order based on the proposed system. Note that every taxi receives taxi calls in its region and every independent driver concurrently decides to accept taxi orders according to our proposed system. As to the competition among taxi drivers, since it is an online decision problem, independent drivers make immediate decisions, no competition should be considered so that the proposed algorithm can be easily generalized to the multiple taxis situation.

Online selection solutions[10] usually let the subject one minimize the total cost. However, the actual income of a ride-sharing taxi is affected by the taxi fare which is proportional to the travel cost. For instance, selecting a passenger with a long travel distance may lead to much more income compared

Authorized licensed use limited to: JILIN UNIVERSITY. Downloaded on April 02,2024 at 07:16:34 UTC from IEEE Xplore. Restrictions apply.

to a passenger with a rather short one. As a result, it is unreasonable to simply set the optimization objective as minimizing the total travel distance.

Assume a taxi c decides to select several passengers (not exceeding the capacity c.c). Some of the selected passengers may be already picked up and others remain on-site waiting for being picked up. A sequence of ride-sharing events is denoted as the current route $\mathcal{CR}(c) = \langle x_1, ..., x_n \rangle$, where each $x_i \in \mathcal{CR}(c)$ is either a pick-up event of being picked up passengers or drop-off event of both about to picked up and already picked up passengers. Note that the pick-up events ought to be scheduled before the corresponding drop-off events. A set of passengers is denoted as $\mathcal{CP}(c) = \{p_1, ..., p_c\}$, where each $p_i \in \mathcal{CP}(c)$ is the passengers already picked up. Furthermore a set of passengers already dropped off is denoted as $\mathcal{DP}(c) = \{p_1, ..., p_n\}$. In the meantime, we also maintain a sequence of events that the taxi already passed by, which is denoted as $\mathcal{PR}(c) = \langle y_1, ..., y_n \rangle$. Specifically, when an event l_i is removed from $\mathcal{CR}(c)$, it is added into $\mathcal{PR}$(c) immediately.

For example, assume taxi c_1 is going to pick passenger p_5 with passenger p_3 in car and p_1 already dropped off. The shortest route schedule is to picked up p_5 and then drop off p_3 and p_5 successively. Then the current route $\mathcal{CR}(c) = \langle p_5.o1, p_3.dl, p_5.dl \rangle$, the passed route $\mathcal{PR}(c) = \langle p_1.ol, p_3.ol, p_1.dl \rangle$, current passengers in car $\mathcal{CP}(c) = \{p_3\}$ and passengers dropped off $\mathcal{DP}(c) = \{p_1\}$.

Next, we can calculate the total travel cost DIS(c) as follows:

$$DIS(c) = \sum_{i=0}^{n} dis(y_i.l, y_{i+1}.l) \tag{1}$$

where $y_i.l$ is the location of event y_i. Note that we use $dis(y_i.l, y_{i+1}.l)$ to indicate euclidean distance and it can be easily generalized to real road net distance. Next, we define payment of passengers $\mathcal{DP}(c)$ ever picked up as follows:

$$REV(\mathcal{DP}(c)) = \sum_{i=1}^{n} dis(p_i.ol, p_i.dl) \times c \times discount \tag{2}$$

where c is price coefficient and passengers can get a proper discount if sharing their journey with someone. Then the profit of taxi c is calculated as follows:

$$PROFIT(c) = REV(\mathcal{DP}(c)) - DIS(c) \tag{3}$$

Valid Route Schedule. A dynamic taxi route schedule $\mathcal{CR}(c)$ is valid if it meets the following constraints:

1) Route schedule order constraint: For any selected passenger p_i in the route schedule $\mathcal{CR}(c)$, let x_j be the pick-up event and x_k be the drop-off event. Then we must have $j < k$, i.e., the pick-up event must happen before the drop-off.

2) Taxi capacity constraint: At ant time, the number of passengers in car(the passengers have already picked up but not yet dropped off) does not exceed the capacity c.c.

3) Profitable order constraint: When a taxi decides to accept an online taxi order, for the individual rationality, the order must be profitable, i.e., $\mathcal{PROFIT} > 0$.

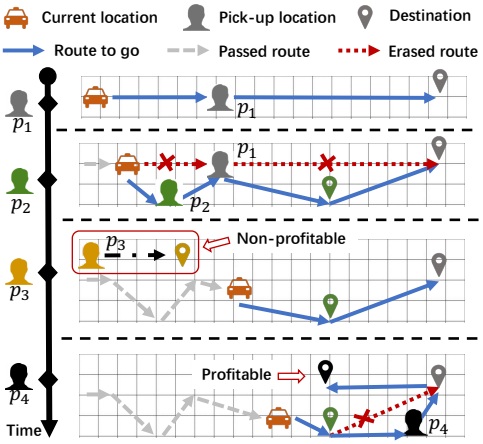


Fig. 2: Running example

Example 1: Fig.2 depicts a running example containing four temporal successive parts. Four taxi seekers are denoted as $p1, p2, p3$, and $p4$. The first part indicates a beginning situation where an empty taxi is driving in the street. At time t_1, a taxi seeker $p1$ submits a taxi order from the pick-up location to the destination point along the predefined route. As the taxi is empty and the first order is usually profitable, our method chooses to accept it and plans the taxi's following route as is shown in the picture.

After that, the taxi gets into a sharable state, the planned route could be separated into two sharable segments. In the following sharable segment, our method random samples from the history data set and calculates a set of profits as sample set, where the maximum is set as the threshold. On its way to picking up p_1 along the 'route to go', the taxi driver receives another order from p_2 at time t_2. After comparison with the threshold, our method decides to accept this taxi order. As a result, the route is rearranged and the outdated route is erased.

Then, at time t_3, taxi has already picked up passengers p_1 and p_2 and hasn't received taxi requirement in the passed two sharable route segments. In the third sharable route segment, $p3$ gets online and submits a taxi order. However, our method indicates this order is not as profitable as the expected maximum profit in the sample set so it is refused. The taxi route remains unchanged.

At last, an order is submitted by p_4 at t_4. At that time, the taxi is on its way of dropping off p_2 with p_1, p_2 in the car and certainly not exceeds the capacity. Note that our system allows drivers to continuously accept orders. In the current sharable route segment, our method decides to pick up $p4$ and rearranges the route.

To sum up, given a set of online taxi seekers $\mathcal{P}$ arrive in real-time, the Online Passenger Selection in Ride-Sharing (OPSRS) problem is to find a relative profitable taxi order submitted by passenger p_i in every sharable route segment and build a route schedule with small cost, where $p_i \in \mathcal{P}$, and for a taxi c such

Authorized licensed use limited to: JILIN UNIVERSITY. Downloaded on April 02,2024 at 07:16:34 UTC from IEEE Xplore. Restrictions apply.

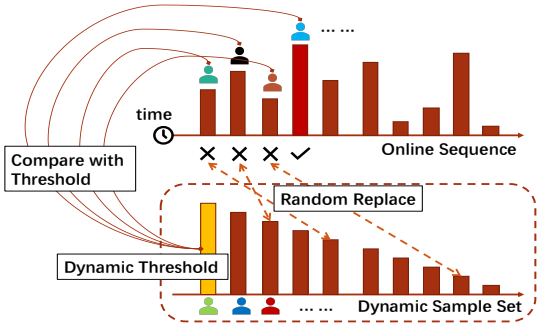


Fig. 3: Sample-based prophet inequality

that the overall profit, i.e.,

$$\sum_{p_i \in \mathcal{DP}(c)} PROFIT(c) \tag{4}$$

is maximized, subject to all $\mathcal{CR}(c)$ being valid.

III. Method

A. Overview

To efficiently solve the online ride-sharing problem, we build a city map based on real altitude and longitude. Taxi drivers can get altitude and longitude pairs corresponding to the start location and destination that is included in each taxi seeker's requirement. As we are facing a ride-sharing problem, whenever a taxi seeker submits taxi requirement, taxi driver will attempt to add it to the current route schedule and find the most profitable route arrangement. A gap between the driver's profit of the new route schedule and the old one is defined as X_i, and our ride-sharing problem can be formulated by the theory of prophet inequality as follows:

During the driver's current route schedule, several taxi seekers submit taxi requests at times T_i. Between every two stopping event, We separate current route into several sharable route segments. In each sharable segment, the taxi driver may receive some taxi requests. We pretend to accept each request, arrange virtual route schedule and calculate the extra profit as random variables $X_1, ..., X_n$. Obviously, these variables are drawn from an identical but unknown distribution. Based on prophet inequality, we use random sampling to evaluate the level of expected maximum $\mathbb{E}[max_{t \leq n} X_t]$, which guarantees a competition ration bound of $1 - 1/e$.

In addition, the system only maintains information on current taxi orders, which does enable optimized route schedules over a long time span. Thus, when multiple orders are submitted at exactly the same time, prophet inequality still processes them one by one, but in drivers' points, it is still a parallel way. Fig.3 describes the insight of sample based prophet inequality algorithm. By maintaining a dynamic sample set, we have a dynamic threshold. When variables are revealed in the online sequence, we compare them with the threshold one by one. If it exceeds the threshold, we accept it. If not, we replace it with a random one in the sample set.

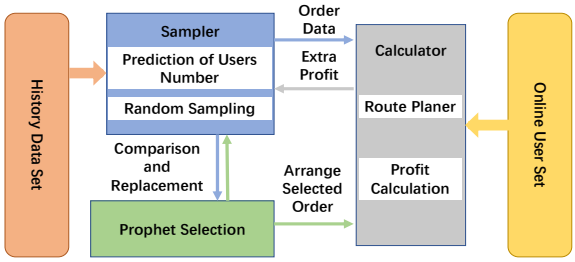


Fig. 4: System architecture

B. Proposed Algorithm

Generally, our proposed algorithm contains four parts: random sampling, route schedule (RS), profit calculator, and sample-based prophet inequality (SBP).

Fig.4 presents the system architecture of our proposed algorithm. Sampler predicts the expected number of users that will submit taxi requests in the time period of a sharable route segment based on the history data set and random samples from it. Then sampler sends sampled order data to the calculator to arrange route schedule and outputs a sequence of extra profit as a sample set. Then a user from the online user set submits a taxi order, it is also sent to calculator and generates profit to prophet selection. Then prophet selection compares this profit of accepting the current order with the maximum in the sample set. Drivers will accept this order if the profit exceeds the maximum in sample set and re-arrange its current route schedule.

We proceed to introduce a route Schedule algorithm to tackle the problem of choosing the route with the lowest cost when coming into a taxi requirement. In RS, we process the route scheduling tasks in a greedy manner after calculating all the possible routes. Note that each taxi has a passenger capacity of up to 3. So that calculating all the possible routes cost is not complex.

Given a taxi c, we have a set of current route point CR and a set passengers in car P. When the taxi receives a taxi requirement with pick-up locating and destination, we obtain a range of route costs if we accept this order. At last, this algorithm outputs the route with the lowest cost. We further illustrate RS using a running example.

Example 2: Taxi c_1 has a current route $CR = \{CL, P_1.dl\}$ and receives a taxi requirement $P_2 = \{P_2.ol, P_2.dl\}$. The first element CL represents the timely updated current location of taxi c_1, and the other denotes the following location points where taxi c_1 intends to pick up or drop off passengers. Each taxi requirement is formed by the pick-up location and destination and each location point is consist of latitude and longitude. After that, the algorithm splits the CR set into several location points and re-permutates them with the addition of two location points in the taxi requirement. Note that it is valid for the re-permutated route only if the taxi arrives at the pick-up location first and then goes to the destination. In this case, there are two location points in the original route

Authorized licensed use limited to: JILIN UNIVERSITY. Downloaded on April 02,2024 at 07:16:34 UTC from IEEE Xplore. Restrictions apply.

Algorithm 1 Algorithm 1: Route Schedule Algorithm (RS)

Input $CR(currentroute), u_s(selecteduser)$
Output CR
Initialization:
$CD \leftarrow Distance(CR)$,
$Permu \leftarrow permutation(CR)$
Repeat until $permu = \oslash$
$permu1 \leftarrow$ first element in $permu$
if $permu1 \subseteq validroute$ **then**
  if $Distance(permu1) < CD$ **then**
    $CD \leftarrow Distance(permu1)$
  end if
end if
$CR \leftarrow U_s$

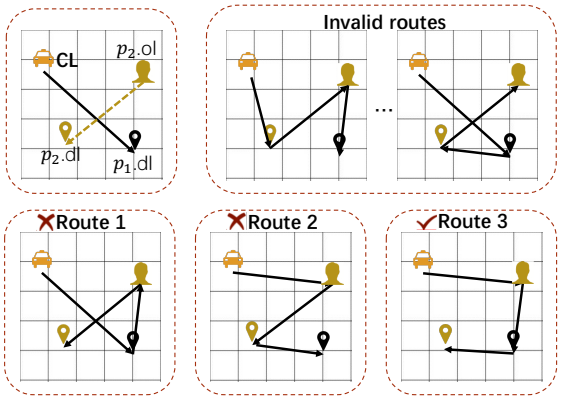


Fig. 5: An exemple of route planer

set. Adding two new location points from the received taxi requirement, there are six possible route schedules, of which three are invalid and the other three are also shown in the figure.

And then, we calculate all these route distances. For instance, we use latitude and longitude to directly calculate Euclidean distance with consideration of the radian of earth. At last, we choose the shortest route (route 3) as the taxi's current route if a driver decides to take this taxi order.

Next, we continue to explain our main algorithm SBP. As we describe above, the algorithm RS outputs the shortest route according to the taxi's current location when a driver receives a taxi requirement. Due to the theory of sample-based prophet inequality, the insight of this algorithm is to compare the output of algorithm RS with the maximum of sample set, accepting if it exceeds the maximum.

We set each time slot to 1 second, and update the current state of the taxi in each time slot. First, if CR and CP are empty, which means that the taxi is waiting for its first order, then it accepts the first order that is profitable and updates its state in the next time slot. At the beginning time slot of entering a new sharable route segment, SBP makes the preparation of constructing a sample set based on the current route set CR. When the taxi receives an order, and if it still has enough space, SBP uses RS and calculates the extra profit

Algorithm 2 Sample based prophet inqeuality(SBP)

Input: online user set U, history user set, car state c
Output: CP
Initialization:
$CL \leftarrow$ current location
$SP \leftarrow$ sample passengers
$RSP \leftarrow$ revenue of sample passengers
Repeat until $|U| = 0$
$U_c \leftarrow$ first element from U
UPDATE CL, CP
if $|CP| = 0$ *and* $|CR| = 0$ **then**
  if $(R(CP + U_c) - R(CP)) > 0$ **then**
    add $U_c \rightarrow CP$
    UPDATE SP, RSP
  end if
end if
if $|CP| = 1$ *or* $|CP| = 2$ **then**
  if route changed **then**
    UPDATE SP, RSP
  end if
  if $(R(CP + U_c) - R(CP)) > Max(RSP)$ **then**
    add $U_c \rightarrow CP$
  else {random select r from RSP}
    $r \leftarrow R(CP + U_c) - R(CP)$
  end if
end if
if $|CP| = 3$ **then**
  pass
end if

if the taxi accepts this order. After prophet inequality selection progress, SBP updates car state if accepts this order. SBP repeats these steps until the online user set is empty, which means the driver decides to get off duty.

IV. THEORETICAL ANALYSIS

In the section, we present the theoretical analysis of the problem hardness, the non-submodular property of online ride-sharing problem and then we provide a theoretical guarantee of our method.

Theorem 1: The online taxi ride-sharing problem is NP-hard.

Proof. Without loss of generality, we consider an offline version of the ride-sharing problem where the driver receives several orders including picking up point L_p, dropping off point L_d and corresponding value v_i. The driver has to decide which orders to accept to maximize the total profit. This problem can be generally reduced to the traveling salesman problem[11]: given a 'value matrix' $D = (v_{ij})$, where v_{ij} = value of going from city L_i to city L_j, $(i, j = 1, 2, ..., n)$, find a permutation $S = i_1, i_2, i_3, ..., i_n$ of the integers from 1 through n that maximizes the total value

$$v_{i_1 i_2} + v_{i_2 i_3} + \ldots + v_{i_n i_1} \tag{5}$$

Authorized licensed use limited to: JILIN UNIVERSITY. Downloaded on April 02,2024 at 07:16:34 UTC from IEEE Xplore. Restrictions apply.

Undoubtedly, it is a classic NP problem. Consequently, further considering online scenario, the online taxi ride-sharing problem is NP-hard. Theorem 1 holds. □

Theorem 2: The online ride-sharing problem is non-submodular.

$Proof$. Consider a specific situation that driver already picked up p_1 and on the way to destination, he will receive all kinds of ride-sharing orders, even one with negative profit. Naturally, online taxi ride-sharing problem is non-submodular. □

Theorem 3 [Prophet Inequality]: There are n random variables $X_1, X_2, X_3, ..., X_n$ with known distribution. The realizations of these variables are revealed one by one, and once upon seeing the value of X_i, we need to decides either to choose i or we pass to the next realization. There is a strategy with expected reward $\frac{1}{2}\mathbb{E}[X_{max}]$.

$Proof$. Suppose that m is the median of the distribution of X_{max}, that is, $Pr[X_{max} \geq m] = \frac{1}{2}$. Let m be the prophet and the strategy is to pick the first realization of X_i if exceeds m. First, the probability of picking a realization is exactly $\frac{1}{2}$ in that there are two possibilities of picking a realization: one is getting on a realization that exceeds m and the other is picking the last one without any realization exceeds m before. Based on the two possibility, the outcome ALG of proposed strategy:

$$ALG \geq m \cdot Pr[X_{max} \geq m] + \sum_{i=1}^{n} \mathbb{E}[(X_i - m)^+] \cdot Pr[\bigwedge_{j \leq i} X_j \leq m] \tag{6}$$

Since $Pr[\bigwedge_{j \leq i} X_j \leq m] \geq Pr[X_{max} \leq m]$, and both the two probability equal to $\frac{1}{2}$, we have $ALG \geq \frac{1}{2}E[X_{max}]$. □

Theorem 4 [Lower Bound of Sample based Prophet Inequality]: Theorem 4. Let $X_1, X_2, ..., X_n$ be i.i.d. random variables from an unknown distribution F. Then there exists a stopping rule with stopping time i such that

$$\mathbb{E}[X_i] \geq (1 - (1 - \frac{1}{n})^n) \cdot \mathbb{E}[max\{X_1, ..., X_n\}] \tag{7}$$

$Proof$. we use $n-1$ samples in n rounds to approximate the distribution of the expected maximum in the value sequence, which means that we divide the whole $n(n-1)$ samples into n sample sets of size $n-1$ and we use the maximum of each as the selecting threshold in algorithm SBP. Once algorithm SBP decides to choose a random variable, which is the extra profit in our case, it should have a value equal to the expected maximum of n independent variables, which is equal to $E[maxX_1, ..., X_n]$. Firstly, the conditional probability of accepting the ith random variable if reaching it is $\frac{1}{n}$ in that algorithm SBP random replaces the maximum in the sample set, and then an overall probability of successful selection is $p = \sum_{i=1}^{n}(1 - 1/n)^i \cdot 1/n = 1 - (1 - 1/n)^n$. After n rounds, since our policy is to select the one that exceed the maximum in the sample set, there must exists a stopping rule so that $\mathbb{E}[X_i] \geq (1 - (1 - \frac{1}{n})^n) \cdot \mathbb{E}[max\{X_1, ..., X_n\}]$ □

Algorithm 2 applies to this approach, but instead of using n-1 fresh samples for each of the n random variables, we constructs n samples with one variable from a fresh user set$\{P_1, ...P_(n-1)\}$. For the every random variable X_i the algorithm uses a threshold equal to the maximum of the n samples. If $X_1 \geq max\{S_1, ..., S_(n-1)\}$, the algorithm stops. Otherwise it adds X_1 to the sample set, picks one of the elements in $\{S_1, ..., S_(n-1)\}$ uniformly at random, and drops this element from the set. The algorithm then continues in the same way, by using the maximum of the set thus obtained as a threshold for the next random variable, and updating the set when a random variable fails to exceed its threshold.

V. Performance Evaluation

A. Datasets and Settings

The three real-world data sets are used for the evaluation:

- *NY Taxi*[1] includes fields capturing pick-up and drop-off dates/times, pick-up and drop-off locations, trip distances, itemized fares, rate types and payment types.
- *GeoLife*[12] was collected from phones carried by 182 users, which record a broad range of users' outdoor movements. We only use those taxi trajectories.
- *HK Taxi* is a real-world ride-sharing date set collected by the GAIA Open[2]dataset, which contains masked user ID, order ID, pick-up location, drop-off location, order received time, and traffic type (i.e.ride-sharing or not).

For taxi ride-sharing, we split the data set of GeoLife, NY Taxi, and Haikou taxi into two parts according to the time. We use the first half as the pool when randomly sampling, and the other half as an online passenger set, where the taxi seekers get online one by one submitting taxi orders to all drivers.

B. Comparison Algorithm and Metrics

The online ride-sharing problem with ready to share constraints and fair concerns is quite different from the existing works, so we mainly compare our proposed online user selection algorithm (referred as "PHT") with the following algorithms:

NOC, which forbidden taxi drivers to share their customers' ride. Note that not all taxi orders are profitable so the taxi driver may get a negative benefit using this algorithm.

OPT, based on the local beam algorithm, outputs relatively optimal results.

FIR, which lets the taxi driver simply accept the first ride-sharing order in each sharable route segment without any selection standard.

ADT, which is named as a sample based adaptive threshold online selection algorithm. It uses a random sample set to generate the maximum possible profit.

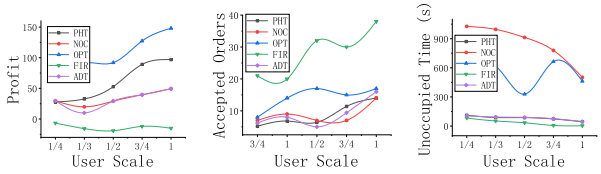
SEC, which observes the first 1/e part of the online taxi orders and then accepts the first order which exceeds the maximum of the observation part.

Generally speaking, PHT and ADT depend on the sample set while FIR and SEC only vary with the online user sequence. NOC is a non ride-sharing algorithm and OPT is

[1]https://www.nyc.gov/assets/tlc

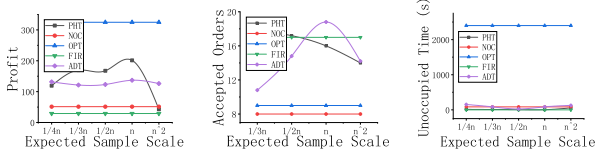
[2]https://outreach.didichuxing.com/app-vue/DataList02

Authorized licensed use limited to: JILIN UNIVERSITY. Downloaded on April 02,2024 at 07:16:34 UTC from IEEE Xplore. Restrictions apply.



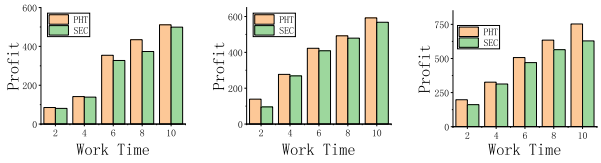
(a) Profit (b) Accepted orders (c) Unoccupied time

Fig. 6: Evaluation on user scale



(a) Profit (b) Accepted orders (c) Unoccupied time

Fig. 7: Evaluation on sample scale



(a) NY Taxi (b) Geolife (c) HK taxi

Fig. 8: Evaluations on work time

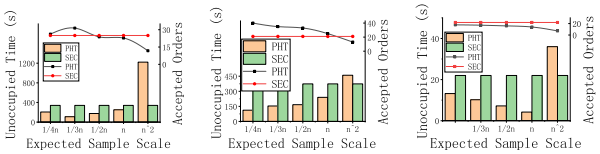
a relative optimal algorithm that is irrelevant to the order of online user sequence.

We evaluate our proposed algorithm on the following metrics:

- *Expected sample scale*: In the time period between each sharable route segment, we search through the sample pool to calculate the expected number of taxi orders that the driver is about to receive, which is the default expected sample scale n. We change the expected sample scale from $\frac{1}{4}n$ to n^2 to analyze the corresponding performance.
- *Online user scale*: Between each sharable route segment, the taxi driver is supposed to receive several taxi orders from online users, which is denoted as online user scale m. Note that this metric is transparent to the proposed algorithm, we use this metric to evaluate the performance when the number of online taxi orders becomes smaller.
- *Work time*: We change the total work time of the taxi driver, which default number is 2 hours, to evaluate the corresponding results.

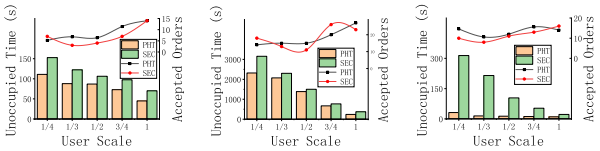
Based on the above variables, we observe the following aspects as the performance of algorithms:

- *Profit*, which is the total net profit taxi drivers get during their work time. The coefficient of cost and payment is set according to the local price.
- *Accepted order*, which is the total taxi orders driver accepted during the work time.
- *Unoccupied time*, which denotes the time of waiting for taxi orders. Note that the time that drivers use to pick up passengers is not included.



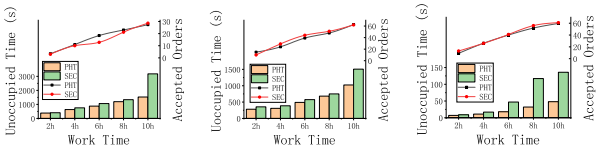
(a) Ny Taxi (b) Geolife (c) HK taxi

Fig. 9: Comparison between PHT and SEC



(a) NY Taxi (b) Geolife (c) HK taxi

Fig. 10: Comparison between PHT and SEC



(a) NY Taxi (b) Geolife (c) HK taxi

Fig. 11: Comparison between PHT and SEC

C. Evaluation Results

As shown in figure 6 and 7, We first evaluate the algorithms on sample scale and user scale. We do not consider SEC here because SEC doesn't involve sample set and we will conduct several comparison between PHT and SEC later. OPT accepts a reasonable amount of orders and generates the most profit because it is an offline algorithm that goes over all the taxi orders and works out an optimal solution. As a result, it uses much unoccupied time to wait for the most profitable order. The accepted orders of NOC are unstable since it heavily depends on the user set and generates a rather low profit and high unoccupied time. FIR simply accepts the first ride-sharing order even though the order sometimes takes a long distance to pick up the passenger and brings negative profit. We observe that only PHT and ADT are influenced by sample scale in that PHT sets the dynamically maximum as the threshold while ADT sets a random threshold from an exponent divided sample set. User scale influences all algorithm but PHT outperforms others except OPT. Figure 8 presents the evaluation on work time between PHT and SEC. It turns out that PHT brings more profit to taxi driver than SEC in the three dataset.

Figure 9 to 11 indicate the comparison between SEC and PHT. As is shown in the experimental results, PHT generates less unoccupied time for taxi driver than SEC in the basic sample scale or less, but performs much worse than SEC when the sample scale comes to n^2. The reason is that when we set the sample scale to n^2, the system hardly exchanges the maximum in the sample set and consequently, it is rare to select successfully. We notice that the orders accepted by PHT is close to that of SEC, but the unoccupied time is less than

Authorized licensed use limited to: JILIN UNIVERSITY. Downloaded on April 02,2024 at 07:16:34 UTC from IEEE Xplore. Restrictions apply.

SEC(except when the sample scale is n^2), which means that under accepting a similar amount of taxi orders, PHT generates less unoccupied time and verifies that our proposed algorithm guarantees global fairness for all taxi orders. We notice that in the dataset HK Taxi, PHT almost has no unoccupied time. It is because that this dataset contains a great amount of clustered orders in the city zone, PHT can continuously pick up passengers to share a ride while SEC have to spend some time observing.

VI. Related work

A. Dynamic Ride-sharing Problem

Dynamic ride-sharing, also named real-time ride-sharing, is a modern paradigm of carpooling and Mobility on Demand(MOD)[13]. Traditional ride-sharing[14] supposes that taxi seekers act in some fixed time schedule and static origin and destination locations. Rather, dynamic ride-sharing focuses on each trip respectively and dynamically rearranges trip routes to accommodate the taxi seekers getting online at a random time and locations one by one. Taxi drivers must make an immediate and irrevocable decision to maximize their profit or minimize their total route distance.

B. Online Decision Problem

In an online decision problem, one has to make a sequence of decisions related to a collection of objects that arrive and depart over time without knowledge of the future. One representative case is called "secretary problem", where specifically n candidates apply for a unique position and came one by one, the boss only compares the current applicant to those already seen. In each interview, the boss either rejects, but not recalls after this interview, this applicant or accepts him. An optimal stopping theory[15] points out that by waiting until 1/e (approximately 37%) of the candidates have been interviewed and then selecting the first one better than all those in the observation stage, it can guarantee to obtain the global optimum with a probability of 1/e.

Another case is the prophet inequality problem[16] that one is showed n non-negative numbers from an unknown distribution one at a time and the goal is to maximize the expected value of the number selected. Krengel et al. [16] state that by setting the threshold at half of the optimum, the maximal expected reward is at least half of the global optimum. Correa et al. [17] propose a sample-based prophet inequality which is much more adoptable in online decision problems since the observation stage is replaced by the sampling stage so as to ensure fairness for all candidates.

VII. conclusion

Taxi ride-sharing has many challenges facing online scenarios and an enormous amount of taxi requirements. We formulate the online taxi ride-sharing problem as a prophet inequality for independent identically distributed random variables from an unknown distribution. Aiming at providing rapid and parallel guidance for drivers, We propose a sample-driven algorithm with a guarantee of lower bound. Experimental results show that the proposed algorithm can effectively meet all of the above-mentioned requirements. Compared with the major online algorithm, our proposed algorithm can enhance profit and accepted orders, and reduce unoccupied time.

Acknowledgment

This work is supported in part by National Key R&D Program of China under Grant Nos. 2021ZD0112501 and 2021ZD0112502, and National Natural Science Foundation of China under Grant Nos. 62102161, 61772230 and 61972450, and CCF-Baidu Open Fund (No.2021PP15002000), and Key R& D projects of Science and Technology Department of Jilin Province No.20200404222YY, and NSF grants CNS 1824440, CNS 1828363, CNS 1757533, CNS 1629746, CNS 1651947, and CNS 1564128.

References

[1] N. A. A, A. E. B, M. S. C, and X. W. B, "Optimization for dynamic ride-sharing: A review - sciencedirect," *European Journal of Operational Research*, vol. 223, no. 2, pp. 295–303, 2012.

[2] C. Xiang, Y. Zhou, H. Dai, Y. Qu, S. He, C. Chen, and P. Yang, "Reusing delivery drones for urban crowdsensing," *IEEE Transactions on Mobile Computing*, pp. 1–17, 2022 (Early Access).

[3] T. Çetin, *The Rise of Ride Sharing in Urban Transport: Threat or Opportunity?* Urban Transport Systems, 2017.

[4] H. J. W. J. Liu, Yunhao, "Rumor riding: Anonymizing unstructured peer-to-peer systems," *IEEE Transactions on Parallel and Distributed Systems: A Publication of the IEEE Computer Society*, no. 3, p. 22, 2011.

[5] A. Fielbaum, X. Bai, and J. Alonso-Mora, "On-demand ridesharing with optimized pick-up and drop-off walking locations," *Transportation Research Part C Emerging Technologies*, vol. 126, no. 3, p. 103061, 2021.

[6] U. Oftransportation and FederalTransitAdministration, *Advanced public transportation systems*. Advanced public transportation systems, 1995.

[7] S. Martin, S. J. Taylor, and J. Yan, "Trading flexibility for adoption: Dynamic versus static walking in ridesharing," *Social Science Electronic Publishing*.

[8] Y. S. Chow, S. Moriguti, H. Robbins, and S. M. Samuels, "Optimal selection based on relative rank (the 'secretary problem')," *Israel Journal of Mathematics*, vol. 2, no. 2, pp. 81–90, 1964.

[9] M. H. Bateni, M. T. Hajiaghayi, and M. Zadimoghaddam, "Submodular secretary problem and extensions," *Springer Berlin Heidelberg*.

[10] R. Schenkendorf and M. Mangold, "Online model selection approach based on unscented kalman filtering," *Journal of Process Control*, vol. 23, no. 1, pp. 44–57, 2013.

[11] S. Lin, "Computer solutions of the traveling salesman problem," *Bell Labs Technical Journal*, vol. 44, no. 10, pp. 2245–2269, 1965.

[12] Y. Zheng, Q. Li, Y. Chen, X. Xie, and W.-Y. Ma, "Understanding mobility based on gps data," *UbiComp 2008 - Proceedings of the 10th International Conference on Ubiquitous Computing*, pp. 312–321, 01 2008.

[13] X. Y. Yang, "Beyond trilateration: On the localizability of wireless ad hoc networks," *IEEE/ACM Transactions on Networking: A Joint Publication of the IEEE Communications Soceity, the IEEE Computer Society, and the ACM with Its Special Interest Group on Data Communication*, no. 6, p. 18, 2010.

[14] M. Furuhata, M. Dessouky, F. Ordó?Ez, M. E. Brunet, X. Wang, and S. Koenig, "Ridesharing: The state-of-the-art and future directions," *Transportation Research Part B*, vol. 57, pp. 28–46, 2013.

[15] E. B. Dynkin, "The optimum choice of the instant for stopping a markov process," *Soviet Mathematics*, vol. 4, pp. 627–629, 1963.

[16] U. Krengel and L. Sucheston, "Semiamarts and finite values," *Bulletin of the American Mathematical Society*, vol. 83, no. 1977, pp. 745–747, 1977.

[17] J. R. Correa, P. Dütting, F. Fischer, and K. Schewior, "Prophet inequalities for independent random variables from an unknown distribution," 2018.

Authorized licensed use limited to: JILIN UNIVERSITY. Downloaded on April 02,2024 at 07:16:34 UTC from IEEE Xplore. Restrictions apply.