The Rich and the Poor: A Markov Decision Process Approach to Optimizing Taxi Driver Revenue Efficiency

Huigui Rong[1]*, Xun Zhou[2]†, Chang Yang[1], Zubair Shafiq[3], Alex Liu[4]
[1]College of Computer Science and Electronic Engineering, Hunan University, Changsha, HN 410082
[2]Department of Management Sciences, [3]Department of Computer Science
University of Iowa, Iowa City, IA 52242
[4]Department of Computer Science and Engineering, Michigan State University, East Lansing, MI 48824
{ronghg, yangchangchn}@hnu.edu.cn, {xun-zhou, zubair-shafiq}@uiowa.edu
alexliu@cse.msu.edu

ABSTRACT

Taxi services play an important role in the public transportation system of large cities. Improving taxi business efficiency is an important societal problem since it could improve the income of the drivers and reduce gas emissions and fuel consumption. The recent research on seeking strategies may not be optimal for the overall revenue over an extended period of time as they ignored the important impact of passengers' destinations on future passenger seeking. To address these issues, this paper investigates how to increase the revenue efficiency (revenue per unit time) of taxi drivers, and models the passenger seeking process as a Markov Decision Process (MDP). For each one-hour time slot, we learn a different set of parameters for the MDP from data and find the best move for a vacant taxi to maximize the total revenue in that time slot. A case study and several experimental evaluations on a real dataset from a major city in China show that our proposed approach improves the revenue efficiency of inexperienced drivers by up to 15% and outperforms a baseline method in all the time slots.

Keywords

Taxi Driver, Markov Decision Process, Revenue Efficiency

1. INTRODUCTION

Taxi services are playing an important role in the public transportation system in modern cities. Taxi drivers are a big social group in many major cities in the world, and improving the business efficiency helps increase taxi drivers' annual income and thus contributes to the development of urban economy. Also, higher efficiency means less driving time and cruising distance needed, and thus leads to lower gas emission and fuel consumption.

*Co-first author
†Co-first author and corresponding author.

CIKM '16, October 24–28, 2016, Indianapolis, IN, USA.
© 2016 ACM. ISBN 978-1-4503-4073-1/16/10. . . $15.00
DOI: http://dx.doi.org/10.1145/2983323.2983689

Recent research [1–3, 8, 11] have focused on developing recommendation systems for taxi drivers. These work usually learn knowledge about passenger demand distribution from data and recommend the best route or location for a driver to optimize one or more of the following measures: the profit margins for the next trip [1, 8], the chance of finding the next passenger [4, 7], or energy consumption before finding the next passenger [3]. Some other work learns knowledge from taxi data for other types of recommendation scenarios such as fast routing, ride-sharing, or fair-recommendation [9, 10, 13, 14], which is not directly related with our topic.

The above work, although effective in improving the proposed measures, still have some limitations. First of all, some of the above work (e.g., [3, 8, 12]) simply aggregated the historical data over the entire study period and ignored the temporal variation of the passenger distribution. This may lead to inaccuracy in the recommendations. More importantly, all of the existing work focus on optimizing the measures for the immediate next trip. However, they do not consider the impact of the driver's move on the overall revenue in the next few trips.

To address the limitations of the related work, this paper investigates how to learn business strategies from historical data to increase the revenue efficiency (revenue per unit time) of taxi drivers. This paper models the passenger-seeking process as a Markov Decision Process (MDP). The study area is partitioned into grids. Each state in MDP is a combination of current grid, time, and the driving direction. For each state, a vacant taxi may choose to travel to a subset of its neighboring grids and cruise through that grid. At a probability p_{seek}, the driver may find a passenger during the cruising, and the destination of the passenger may follow another probability distribution p_{dest}.We learn the necessary MDP parameters (e.g., p_{seek}, p_{dest}) for each time slot from historical data. The MDP optimization problem is solved by a dynamic programming algorithm. The output, which contains the best action to take for each state, will be recommended to taxi drivers.

The major contributions of this paper are summarized as follows: (1) We perform a detailed analysis to quantify taxi business efficiency and identify the key strategic differences between the most successful and the least successful drivers. (2) We model the passenger-seeking process as a Markov Decision Process (MDP). For each time slot, we learn a different set of input parameters for MDP from historical data. A dynamic programming approach is employed to solve the MDP.(3) We propose a recommendation mechanism based on our MDP solution in each time slot. Computer simulations show that our solution can improve the revenue efficiency of drivers by up to 15% in real data and up to 8.4% over a baseline method.

2. PROBLEM FORMULATION

In this section we present our data analysis to quantify the success of a driver and identify the more (less) successful drivers. Then we identify the most important factors for drivers to improve their business. Based on these analysis we formulate the problem as an optimal decision problem. The dataset we are using in this study contains taxi operation record for a whole year from the capital city of a central province in China. There are approximately 19 million taxi trip records (53,000 per day), where each record has the latitude-longitude coordinates and timestamps of the pick-up and drop-off events, along with total traveled distance and the fare of the corresponding trip. There are totally about 1400 taxi cabs in the data.

2.1 Who Are More Successful?

We first calculate the total business time of each taxi cab to better estimate the drivers' time commitment. The total business time (denoted as T_{bus}) is the sum of two parts, the total operating time (T_{drive}) and the total seeking time (T_{seek}) as shown in Eq. 1. The total operating time T_{drive} is the sum of all the trip duration of a taxi. Calculating total seeking time is tricky since the gap between two consecutive trips vary and some drivers may choose to take a rest between trips. We considered all the gaps between consecutive trips in the dataset and call these gaps "seeking trips". Figure 2 shows the distribution of the length of all the seeking trips. As can be observed, 90% of the gaps are shorter than 25 minutes, so, we thus use 25 minutes as the threshold.

$$T_{bus} = T_{drive} + T_{seek} \tag{1}$$

Revenue Efficiency. An obvious way to increase revenue is by driving taxi for longer time duration. However, given time constraints, it makes sense to maximize revenue per unit of time. To this end, we define a revenue efficiency (E_{rev}) metric: revenue earned divided by total taxi driver's business time. Formally it can be expressed as follows:

$$E_{rev} = \frac{M}{T_{bus}} = \frac{M}{T_{drive} + T_{seek}} \tag{2}$$

where M denotes the total money earned by the taxi driver. The revenue efficiency measure could be calculated for an hour, a day, or a whole year.

Identifying Top and Bottom Drivers. Based on the above proposed measure, we first rank all the day-time drivers and find the top and bottom ones. Figure 1 plots the histogram of overall revenue efficiency for all taxi drivers in our data set. About 80% drivers have a revenue efficiency between 0.72 and 0.80. We find the top 10% and bottom 10% drivers as successful and not-so-successful drivers.

We also further zoom into each time slot to show the difference in efficiency between the top and bottom drivers. We calculate the average revenue efficiency of the top 10% and bottom 10% drivers in each time slot. Figure 3(a) shows the average revenue efficiency of the top 10% drivers (yellow bar) and the bottom 10% drivers (red bar) during different time slots of a day. As can be seen, there is a 15%-20% difference between the two groups.

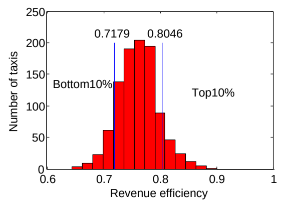


Figure 1: Distribution of Revenue Efficiency for all the day-time drivers

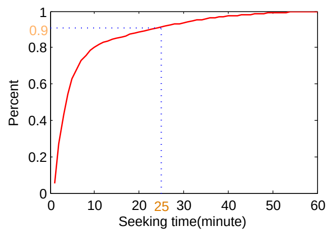


Figure 2: Distribution of seeking time

2.2 Why Are the Top Drivers Successful?

As noted previously, the total business time includes driving time (occupied taxi) and seeking time (vacant taxi). A taxi driver has positive money income when driving with passenger but no income while seeking. The revenue efficiency of a driver depends on (1) how much money a driver can make while driving for one minute, i.e., driving efficiency, and (2) how quickly can one driver find the next passenger (seeking efficiency). Taking slower routes, running into traffic congestions will be likely to lower his/her driving efficiency and revenue efficiency. First we define driving efficiency E_{drive} as follows:

$$E_{drive} = \frac{M}{T_{drive}} \tag{3}$$

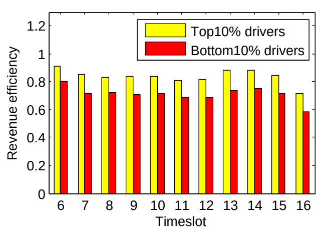
The best way to improve the revenue efficiency is to increase E_{drive} while reducing the total seeking time. Figure 3(b) shows the comparison of driving efficiency between the top and bottom drivers. The difference is about 10% to 13%, which is smaller compared to the difference in the overall revenue efficiency (approximately 17% difference).

We also compare the average seeking time of the top 10% and bottom 10% drivers. Figure 3(c) shows the average seeking time (the time gap between consecutive trips that are less than 25 minutes) of the top (yellow bar) and bottom drivers (red bar) in each time slot. Results show that top drivers on average save up to 25% to 35% time between trips, respectively.

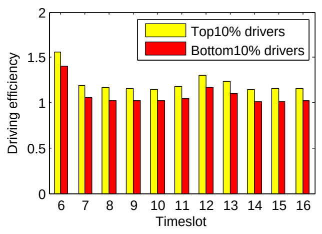
2.3 What is the Optimal Strategy?

A number of related work have used taxi data to calculate the chance of finding a passenger in each region or along each road segment. Based on this probability and the current location of a driver, one can choose the route or destination with the highest probability to find the next passenger. So where passengers want to go is also a key issue to drivers. However, drivers don't have much freedom in choosing where to go after they have found a passenger. In China, most cities will fine taxi drivers who refuse to take a passenger. Taking this issue into consideration, we believe that the driver's decision on where to find the next passenger is very important, not only because it determines the seeking time of this iteration, but also to some degree the next few trips and the overall revenue efficiency. To this end, we formulate the best taxi seeking strategy as follows:

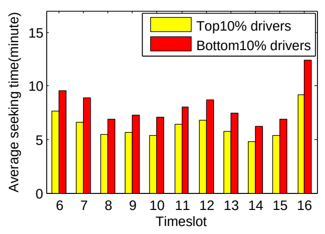
Given the historical passenger trips and seeking trips for each time slot, the current status of an vacant taxi, we aim to **find** the next movement for the driver. **The objective**



(a) Average Revenue efficiency



(b) Average Driving efficiency



(c) Average Seeking time

Figure 3: Comparison between top 10% and bottom 10% drivers on (a) Revenue efficiency (b) Driving Efficiency and (c) Average seeking time

Table 1: Major variables in this paper

Variable	Meaning
l, L	The current location and the location set
t	Number of minutes into a time slot
d,D	Incoming direction to the current grid and the set of all the possible directions
s	A state of the MDP model, $s = (l, t, d)$
S	The collection of all the states. $s \in S$
a	An action taken by a vacant taxi
A	The set of all the possible actions, $a \in A$
$A_{allowed}(s)$	The allowed actions for state s
$\pi(s)$	the optimal action for state s
$t_{seek}(j)$	The time needed for a taxi to cruise grid j
$t_{drive}(i,j)$	The time needed to drive from grid i to j
$P_{dest}(j,k)$	The probability a passenger picked up in grid j wishes to go to grid k
$P_{find}(j)$	The probability that a passenger can be found in grid j during the vacant cruise
$r(i,j)$	The expected reward (trip fare) from grid i to j
E_{drive}	Driving efficiency (Yuan per minute while occupied)

is to maximize the total expected revenue for this taxi in the rest time of the current time slot.

3. A MARKOV DECISION PROCESS APPROACH

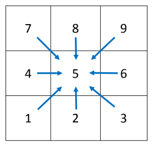
A Markov Decision Process (MDP) [5] is a stochastic decision process with a set of states (S) and a set of possible actions (A) that transfer the states from one to another. A MDP must have the Markov property, which requires that the next state of the process only depends on the current state and the action, but not any previous state or action. Each action will transfer the process from the current state to a new state at a probability (P) with a corresponding reward (R). Usually the optimal action for each state (often denoted as "policy") is desired, which maximizes the total reward over a finite or infinite number of steps.

We model the passenger seeking strategy of a taxi as a Markov Decision Process. Solving this MDP model will give us the best seeking strategy for a taxi at each different state. Table 1 shows all the parameters we use in the MDP model.

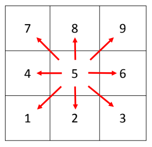
3.1 System States

It is obvious that the best seeking strategy of a taxi is dependent on the current location and the remaining time of the business time window. In our model, each state of a vacant taxi is described by three parameters: current location (grid cell number)$l \in L = \{1, 2, ..., 128 \times 128\}$, current time (minutes into the time slot) $t \in T = \{1, 2, ..., 60\}$, and the direction from which the taxi arrived at the current location $d \in D$, where $D = \{\emptyset, \nearrow, \uparrow, \nwarrow, \leftarrow, \circlearrowleft, \rightarrow, \searrow, \downarrow, \swarrow\}$.

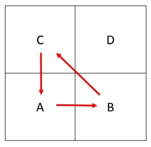
The "incoming direction" is needed for each state to ensure that the taxi does not fall into a infinite loop among a small number of grid cells or staying in the same grid forever. The set of arrival directions for a location includes the eight possible directions from the queen-connectivity neighbors, plus from the grid itself. We also include a NULL direction $\emptyset$ in the set, which indicates that the taxi just dropped off a passenger and does not have any arriving direction. We use ten numbers (0-9) to index these directions, which is illustrated in Figure 4(a). 0 is for the NULL direction and 5 is for the direction to the current grid itself. Formally, a state in our MDP model can be represented as $s = (l, t, d)$. The maximum number of states in our problem setting is $|L| \times |T| \times |D| = 16384 \times 60 \times 10 = 9830400$. However, the actual number of states is much smaller than this.



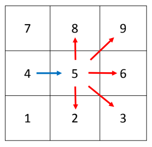
(a) Arrival Direction



(b) Possible Actions



(c) Unallowable actions



(d) Allowable actions

Figure 4: An illustration of the direction of MDP model

3.2 Actions

In our model, each vacant taxi at one of the states has nine possible actions to choose. Each action (a) is to move from the current grid to one of the eight neighbors or stay in the current grid. Formally, it can be expressed as: $a \in A$, $A = \{\swarrow \downarrow, \searrow, \rightarrow, \circlearrowleft, \leftarrow, \nwarrow, \uparrow, \nearrow\}$. Similar to D, we also use numbers (1-9) to index the directions. Figure 4(b) illustrates the mapping of the directions. The arrow directions are the opposite compared to the mapping of the d parameter in the states. The same direction in D and A are indexed with different numbers.

For each grid cell i, we examine all the seeking trips that origin from i and end at each of i's eight neighbors. For each of i's neighbors j, we calculate how many percent of these seeking trips are between i and j. If the percentage of such trips between i and j is lower than a minimum threshold then we could conclude that there is no road connecting these grids. In this case, we do not allow the taxi to move from i to j. Taken into account the precision errors of GPS signal, the threshold is set to 5%.

Finally, we also need to prevent a taxi from cruising the same grid or a small number of grids in a loop. Figure 4(c) illustrate such a scenario. To this end, we require that in a state s, a taxi may only choose actions from a subset of A, denoted as the allowed action set $A_{allowed}(s)$. For a taxi that just dropped off a passenger, the incoming direction is 0 (no direction). The taxi may choose any action.After the taxi searched the current grid, it must leave the current grid.Also a taxi must follow approximately the same direction and should not make a very sharp turn. Specifically, if the taxi came to the current grid l at time t from direction d, then $A_{allowed}(l, t, d)$ should be within the range of $[-90°, 90°]$ of d. Figure 4(d) illustrates an example. A taxi coming to the current state from left (4) can only go to the top, top-right, right, bottom-right, or bottom grid (2,3,6,8,9).

3.3 State Transition and Objective Function

Assuming the current state of the taxi is $s = (i, t, d)$. An action a is taken to move the taxi from grid i to its neighbor j. As a result, the taxi will cruise to the next destination j and cruise the entire grid j in $t_{seek}(j)$ minutes. There will be two possible consequences.

(1) The taxi successfully finds a passenger in grid j after cruising the grid for $t_{seek}(j)$ minutes. In this case, the passenger may choose to go to one of the grid cells as destination (denoted as k) at a probability $p_{dest}(j, k)$. We will discuss how to obtain this probability later. Upon finishing this trip, the taxi will arrive at location k. We use $t_{drive}(j, k)$ to represent the total time needed to travel from j to k. The driver will receive a fare of $r(j, k)$ Yuan, where $r(j, k)$ represents the expected fare between grids j and k. The taxi will start seeking from k again. The state of the taxi is thus transitioned to $s' = (k, t + t_{seek}(j) + t_{drive}(j, k), 0)$.

(2) The taxi did not find any passenger after $t_{seek}(j)$ minutes in j. Then the taxi must leave the current grid and move to the next one. Assume the taxi took action $a = 6$ $(\rightarrow)$. Then the taxi will end up in the next state $s' = (j, t + t_{seek}(j), 4)$ (from the left grid, $\rightarrow$).

In the above process, an important parameter needed is the probability that the taxi can find a passenger during the cruising in grid j, denoted as $P_{find}(j)$. We will discuss how we estimate this parameter in the next subsection.

To summarize, a vacant taxi in any state $s_0 = (i, t, d), s_0 \in S$ may take one of the possible actions $a \in A_{allowed}(s_0)$ to cruise to a nearby grid j. With the probability $1 - P_{find}(j)$ the taxi will transition to state $s_1 = (j, t + t_{seek}(j), 10 - a)$ with no reward. With the probability of $P_{find}(j) \times P_{dest}(j, k)$, $(k = 1, 2, ..., |L|)$, the taxi will end up in the next state $s_2 = (k, t + t_{seek}(j) + t_{drive}(j, k), 0)$ and receive a reward of $r(j, k)$ Yuan. The state transition diagram of the proposed MDP model is shown in Figure 5. Each circle represents a state with the three parameters listed beside it.

The objective of the MDP model is to maximize the total expected reward (taxi fare) in the current time slot. The MDP has a set of terminal states with $t = 60$. Once the system reaches these states, no more actions can be taken.

For every a, the $V^*(s, a)$ function represents the maximal expected revenue in the current time slot if action a is taken at state s. $V(s)$ is the maximum expected revenue for state s. Formally the objective can be expressed as follows:

$$V^*(s, a) = (1 - P_{find}(l)) \times V(l_a, t + t_{seek}(l_a), 10 - a) + \\ \Sigma_{k=1}^{|L|} P_{find}(j) \times P_{dest}(j, k) \\ \times (r(j, k) + V(k, t + t_{seek}(j) + t_{drive}(j, k), 0)) \quad (4)$$

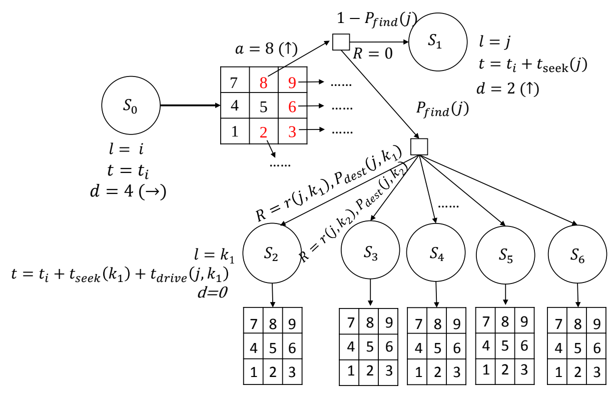


Figure 5: An illustration of the MDP model

where $s = (l, t, d)$ is a state and a is an action that moves the taxi from l to l_a. The optimal policy (π) is defined as follows:

$$\pi(s) = arg \max\{V^*(s, a)\} \quad (5)$$
$$V(s) = V^*(s, \pi(s)) \quad (6)$$

3.4 Learning Parameters for MDP

Now we discuss how to decide the necessary parameters for the MDP. All the parameters are learned from our historical passenger trips and seeking trips.

Learning the passenger pickup probability P_{find}. First we estimate the probability that a vacant taxi successfully picks up a passenger during its cruise in each grid. For each of the time slots, we calculate a 128×128 matrix P_{find} with the pick up probability of each grid.

The basic idea is: the probability of picking up a passenger in grid i can be approximated by dividing the number of successful pickups in i by the total number of times this grid is visited by a vacant taxi. Since the data only contains the origin and destination of each trip, we have to estimate the route taken by each taxi when seeking for a passenger. For each "seeking trip", we use the API provided by Baidu Map to calculate the shortest path between the origin (the previous drop-off) and the destination (the next pick-up). Then we map each estimated seeking trip obtained to corresponding grids and count how many times each grid is passed by a vacant taxi during each time slot (denoted as n_{seek}). We also calculate the pickup counts n_{pickup} in each grid. The pick up probability is thus calculated as follows:

$$P_{find}(i) = \frac{n_{find}(i)}{n_{find}(i) + n_{pickup}(i)} \quad (7)$$

Learning passenger destination probability P_{dest}
To estimate the destination probability for a time slot, we calculate the number of trips between each pair of source and destination in that time slot and get a 128 by 128 matrix W. Each value W_{ij} represents the total number of trips from i to j in the corresponding time slot. We normalize the data in each row (each value divided by the sum of the entire row, except for zero sums). In the resulting matrix P_{dest}, each row i has the empirical probability distribution of a passenger choosing destination j when picked-up in grid i.

Estimating the reward function $r(i, j)$. In our model, the reward (r) is the fare income for a trip. We simply calculate the average fare of trips between each pair of source and destinations in the same time slot as the expected fare.

Estimating the driving time t_{drive}. Here we consider the average driving efficiency of top 10% and bottom 10% drivers. Specifically, we calculate the total fare income of the two groups of drivers and divide them by the total driving time, respectively. The driving efficiency of top 10% and bottom 10% drivers are 1.3061 and 1.1648. Using the driving efficiency, we estimate the travel time for a trip by dividing the estimated fare by the corresponding driving efficiency. Formally we have $t_{drive}(i,j) = r(i,j)/E_{drive}$

Seeking time for each grid t_{seek}. We calculated the average speed of seeking trips and find it is about 300m/min. Since the grid size in our analysis is fairly small, we set the seeking time t_{seek} to 1 minute for every grid.

3.5 Solving MDP

For each time slot, we build a MDP with parameters learned from our data. Now we solve the MDP to find the optimal policy. For each state in the MDP model, the result is the best action to take for that state to maximize total expected rewards in the remaining time of the time slot. The MDP does not have a fixed number of steps. Instead, there are a few terminal states. Once the system reaches a state with t=60 the system terminates. The MDP can be solved by employing a dynamic programming approach.

The pseudo code of the algorithm is presented in Algorithm 1.The algorithm starts from the states with $t = 60$ and find out the maximum expected reward for them. Then the algorithm traces back along time to $t = 1$. For each state s, the algorithm examines all the possible actions and calculates the maximum expected reward for each of them (Lines 2-4). The total expected reward of state s after taking action a, $V^*(s,a)$, is calculated using Equation 4 (Line 5). Here $V(l_a, t+t_{seek}(l_a), 10-a)$ and $V(k, t+t_{seek}(j)+t_{drive}(j,k), 0)$ must have been calculated already since they have larger t value than s. Then the action with the maximum V^* for s (denoted as a_{max}) is selected (Line 6). The maximum expected reward of s is thus set to $V^*(s, a_{max})$ (Line 7). The output of this algorithm is one action for each state.

The algorithm has time complexity of $O(|T| \times |D| \times |L| \times |A|)$. Since $|D|$ and $|A|$ are small constant numbers, the complexity can be simplified to $O(|T| \cdot |L|)$. Similarly, the space complexity is $O(|T| \cdot |L|)$.

Algorithm 1 Solving MDP using Dynamic Programming

```
Input: $L, A, D, T, P_{find}, P_{dest}, R, t_{seek}$
Output: The best policy $\pi$
1: $V$ is a $|L| \times |T|$ matrix; $V \leftarrow 0$
2: for $t = |T|$ to 1 do
3:     for $l = 1$ To $|L|$ do
4:         for $d$ in $D$ do                    ▷ $s = (l, t, d)$
5:             $a_{max} \leftarrow a$ that maximizes $V^*(s, a)$
6:             $\pi(s) \leftarrow a_{max}$
7:             $V(s) \leftarrow V^*(s, a_{max})$
8: Return $\pi$
```

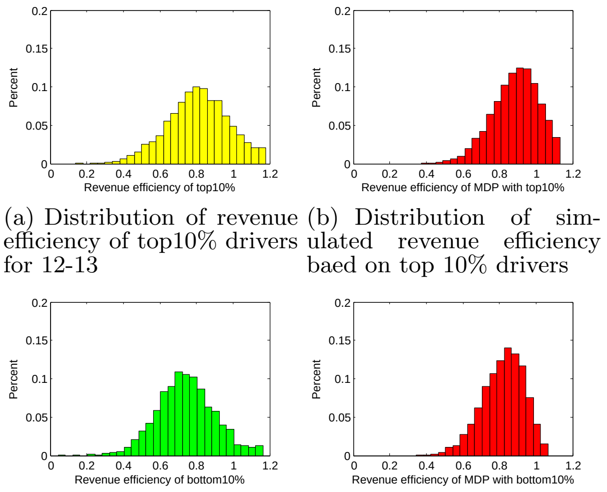
4. EVALUATION

In this section, we perform a number of experiments to evaluate the quality of our result. Specifically, we hope to answer the following questions: (1) How much more revenue can be generated if a driver uses our strategy? (2) Is our proposed method better than methods in related work?

4.1 Revenue Improvement

In order to verify the effectiveness of our recommendation, We carry out a simulation based on our MDP model, and compare our revenue efficiency of simulated results with that of the top 10% and bottom 10% drivers.

Figure 6 (a)-(b) show the revenue efficiency distribution of the top 10% drivers (left, yellow) and our simulated results using the top 10% drivers' driving efficiency (right, red) for time slot 12-13. The average revenue efficiency is improved from 0.828 yuan/min to 0.89 yuan/min, yielding a 7.6% improvement. Figure 6 (c)-(d) show the revenue efficiency distribution of the bottom 10% drivers (left, green) and our simulated results using the bottom 10% drivers' driving efficiency (right, red) for the same time slot. The average revenue efficiency is improved from 0.75 yuan/min to 0.82 yuan/min, yielding a 9.8% improvement. We also perform the same simulations for all the day-time slots. Figure 7 (a) show the average revenue efficiency of the top drivers in reality compared to the simulated revenue efficiency using the bottom 10% drivers' driving efficiency for all the day-time slots. The improvement is between 3.6% and 12%. Figure 7 (b) shows the comparison of revenue efficiency between the bottom drivers and the simulated results using the bottom 10% drivers' driving efficiency for the day-time slots. The corresponding improvement are between 4.2% and 15.5%. As can be observed, our approach has more improvements during morning and evening rush hours because there are more data records and our parameters learned from the data are more accurate.



(a) Distribution of revenue efficiency of top10% drivers for 12-13

(b) Distribution of simulated revenue efficiency baed on top 10% drivers

(c) Distribution of revenue efficiency of bottom10% drivers

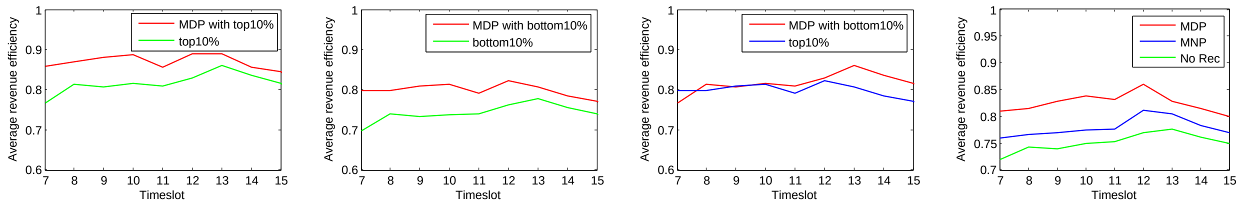
(d) Distribution of simulated revenue efficiency based on bottom 10% drivers

Figure 6: Comparison between simulated and real revenue efficiency in time slot 12-13

Figure 7 (c) shows the simulated revenue efficiency (red line) is higher than the top 10% drivers' revenue efficiency for almost all the time slots. This result validates the effectiveness of our proposed method: the bottom drivers with limited driving efficiency can still make the same amount of money as the top drivers if they follow our driving strategy.

4.2 Comparison with Baseline Method

Next we compare the output of our method with a baseline



(a) Simulated MDP vs. real revenue efficiency with top 10% drivers' driving efficiency (b) Simulated MDP vs. real revenue efficiency with bottom 10% drivers' driving efficiency (c) Simulated MDP revenue efficiency using bottom 10% drivers' driving efficiency vs. real top10% drivers (d) Simulated revenue efficiency of MDP vs. baseline and real data (all drivers)

Figure 7: Revenue efficiency comparison between our method, real data, and a baseline method (best viewed in color).

algorithm called MNP algorithm in related work [8], which is the most relevant and similar to ours. Given the current location of a taxi, the baseline method MNP recommends the next five road segments to seek passengers. The objective of the MNP algorithm is to maximize the total expected net profit along the seeking trip. If no passenger is found, the algorithm will recommend another five road segments based on the new location of the taxi.

Here we use grids to replace road segments to make a fair comparison. The experiment settings are the same as described in 4.1. We calculate the average simulated revenue of all the 6000 taxis in each of the one-hour time slots based on the two methods, respectively. Figure 7(d) shows the average revenue in the simulated results. Our method (red curve) outperformed the baseline method (blue curve) in average revenue efficiency (Yuan per minute) in all the time slots. The maximum improvement is about 8.4%.

5. CONCLUSION AND FUTURE WORK

This paper investigated how to learn the best taxi seeking strategy from historical data to improve taxi drivers' business efficiency over an extended period of time. This problem is of great societal importance.This paper proposed to model the passenger-seeking process as a Markov Decision Process (MDP)to get the best move for a vacant taxi at any state. Case study and experiment results showed that our approach effectively improved the revenue efficiency of inexperienced drivers by up to 15% and outperformed a baseline by up to 8.4%.

For future work, we plan to improve our MDP model to incorporate the total revenue and driving time covering the taxi shift time, and recommend the best strategy to improve the overall revenue efficiency. Finally, we plan to include other attributes such weather condition and traffic conditions to improve the results of our method.

6. ACKNOWLEDGMENTS

Xun Zhou is supported in part by the National Science Foundation under Grant No.1566386. Huigui Rong and Chang Yang are supported in part by the National Natural Science Foundation of China under Grant No.61672221, No.61273232 and No.61304184. Alex Liu is partially supported by the National Science Foundation under Grant Numbers CNS-1318563, CNS-1524698, and CNS-1421407, and the National Natural Science Foundation of China under Grant Numbers 61472184 and 61321491, and the Jiangsu High-level Innovation and Entrepreneurship (Shuangchuang) Program.

7. REFERENCES

[1] J. Yuan, Y. Zheng, L. Zhang, X. Xie, and G. Sun. Where to find my next passenger. The 13th international conference on Ubiquitous computing (UbiComp), pp. 109-118, ACM, 2011.

[2] Y. Zheng, J. Yuan, W. Xie, X. Xie, and G. Sun. Drive smartly as a taxi driver. The 7th International Conference on Ubiquitous Intelligence & Computing (UIC), pp. 484-486, IEEE, 2010.

[3] Y. Ge, H. Xiong, A.Tuzhilin, K. Xiao, M. Gruteser, An energy-efficient mobile recommender system. the 16th International Conference on Knowledge Discovery and Data Mining (SIGKDD), pp. 899-908, ACM, 2010.

[4] D. Ziebart, L. Maas, K. Dey, and J. Bagnell. Navigate like a cabbie: probabilistic reasoning from observed context-aware behavior. The 10th international conference on Ubiquitous computing (UbiComp), pp. 322-331, Springer Berlin Heidelberg,2008.

[5] Bellman, Richard. A Markovian decision process. No. P-1066. RAND CORP SANTA MONICA CA, 1957.

[6] Lee J, Shin I, Park G L. Analysis of the passenger pick-up pattern for taxi location recommendation. The 4th International Conference on Networked Computing and Advanced Information Management (NCM), pp. 199-204, IEEE, 2008.

[7] Y. Ge, C. Liu, et al. A Taxi Business Intelligence System. The 17th International Conference on Knowledge Discovery and Data Mining, pp. 735-738, ACM, 2011.

[8] M. Qu, H. Zhu et al. A Cost-Effective Recommender System for Taxi Drivers. The 20th international conference on Knowledge discovery and data mining (SIGKDD), pp. 45-54, ACM, 2014.

[9] Castro P S, Zhang D, Li S. Urban traffic modelling and prediction using large scale taxi GPS traces. The 10th International Conference on Ubiquitous Computing (UbicomP), pp. 57-72, Springer Berlin Heidelberg, 2012.

[10] Chawla S, Zheng Y, Hu J. Inferring the Root Cause in Road Traffic Anomalies. The 12th International conference on Data Mining, pp. 141-150, IEEE, 2012.

[11] D. Zhang, L. Sun, B. Li, C. Chen. Understanding Taxi Service Strategies from Taxi GPS Traces. IEEE Transactions on Intelligent Transportation Systems, 2015,16(1): 123-135.

[12] J. Huang, X. Huangfu, H. Sun, H. Li. Backward path growth for efficient mobile sequential recommendation. IEEE Transactions on Knowledge and Data Engineering (TKDE), 2015, 27(1):46-60.

[13] S. Ma, Y. Zheng, Wolfson, O. T-share: A large-scale dynamic taxi ridesharing service. The 29th International Conference on Data Engineering (ICDE), pp. 410-421, IEEE, 2013.

[14] S. Qian, J. Cao, et al. A Sharing Considered Route Assignment Mechanism for Fair Taxi Route Recommendations. The 21th Iinternational conference on SIGKDD, pp. 955-964, ACM, 2015.