IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received December 2, 2019, accepted January 5, 2020, date of publication January 9, 2020, date of current version January 17, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.2965171

Waiting or Moving? A Crossroad Network-Based Markov Decision Process Approach to Catch Vacant Taxis

HUIGUI RONG[1], (Member, IEEE), XUDONG ZHANG[1], ZHUO LI[2], AND ZHAOYANG AI[3], (Member, IEEE)

[1]College of Computer Science and Electronic Engineering, Hunan University, Changsha 410082, China
[2]Ophthalmology Department, The Second Xiangya Hospital of Central-South University, Changsha 410011, China
[3]College of Foreign Languages, Institute of Cognition and Language, Hunan University, Changsha 410082, China

Corresponding author: Xudong Zhang (zhangxudong@hnu.edu.cn)

This work was supported in part by the National Natural Science Foundation of China under Grant 61672221 and Grant 61872130, and in part by the Hunan Key Research Development Program under Grant 2017GK2272.

ABSTRACT Taxi services play a critical role in the public transportation system in our cities. However, we usually find it difficult to catch vacant taxis based on our experience alone in the random taxi-waiting mode, especially on the streets unfamiliar to us, which may greatly influence users' taxi service experience. Therefore, how to recommend appropriate waiting locations for passengers becomes meaningful, and the available large-scale taxi trajectory data have helped with the right recommendation. Recent researches have focused on the one-location recommendation for the passengers without considering the recommendation failure situation where they find no vacant taxis available after waiting a long time. In response to this deficiency, we designed a **C**rossroad **N**etwork-based **M**arkov **D**ecision **P**rocess (CN-MDP) scheme to recommend a waiting location sequence for a current passenger whose cumulative probability of catching a vacant taxi getting close to 100%. Further, our scheme changes the recommended locations from the road segments to the crossroads, as we discovered that passengers are more likely to catch vacant taxis at the crossroads connected to multi-road segments. In addition, the multi-passengers competing strategy for vacant taxis at the same location and in the same time slot is also involved in our scheme by dynamically updating the pass rate of vacant taxis at each crossroad and road segment. Some evaluations on a real taxi data set from a major city in China have shown that our recommendation scheme works well and has a higher probability of catching vacant taxis than that of our previous approach and other ones. Our scheme further improves the user experience of taxi services.

INDEX TERMS Markov decision process, waiting location sequence, pass rate of vacant taxis, multi-passengers competition, taxi services.

I. INTRODUCTION

Taxi services, with the ubiquitous availability, route flexibility and comfortable travel experience, make a critical contribution to the public transportation system in our cities [1], [2]. Although DiDi service has covered many big cities, there are still few DiDi services in small and medium sized cities for lack of vehicles, and the service is also often closed down and rectified, which makes DiDi service less available. In addition, many middle-aged and elderly people are not accustomed to reserving DiDi service directly from the internet. So, we usually face the difficulty of catching vacant taxis, especially when we are in an unfamiliar place, which will delay our travel plans and undermine our experience of taxi services greatly. Thanks to the development of mobile communication technologies, it is very meaningful to improve the efficiency and user experience of waiting for vacant taxis [3] among the existing methods of improving taxi services.

A. BACKGROUND AND MOTIVATION

At present, ***the random pattern of waiting for vacant taxis*** greatly wastes the valuable time of passengers, because they may not catch vacant taxis after waiting for a long time on

The associate editor coordinating the review of this manuscript and approving it for publication was Justin Zhang.

the roadside. On the other hand, the rapid expansion of our living cities usually makes users experience a longer time of waiting for vacant taxis, feel less reliable with a poorer user experience of taxi services. And for most passengers, where to wait for taxis mainly depends on their experience, which is highly limited to the tolerable time and the walkable area. Inexperienced passengers often spend more time waiting for vacant taxis, which will result in a large time cost and reduce user experience.

As such, it is of paramount importance to design a practical scheme to optimize the process of catching vacant taxis and improve the user experience. Furthermore, the taxis equipped with GPS and mileage recording devices enable us to gather trajectory data of thousands of taxis in a long period of time. So we could analyze and predict when and where more vacant taxis are likely to appear by learning from the history data set [4], [5], and then recommend the waiting locations for passengers.

However, current researches have focused on the one-location recommendation for the current passengers [23]–[26]. They just recommend only one waiting location for the passenger, such as the location nearest to the current passenger, the place with minimum waiting time, and among others. But they usually do not consider the situation of recommendation failure when the passengers catch no taxis within a long time.

Therefore, we propose an approach to recommend several waiting locations for the current waiting passengers. If the passenger is unable to catch a vacant taxi on a certain location, we recommend him to go to the adjacent location to wait; if he is still unable to catch any vacant taxi, we will continue to recommend him with other adjacent locations. We will repeat this process until the termination condition is reached. At last, we will recommend a waiting location sequence for the passengers following this way. In addition, the probability of catching the vacant taxi this way is much higher than the one-location recommendation.

But ***when should the passenger move to the adjacent locations***? We will also tell the passenger of the waiting time on each location, so the passenger can move to the adjacent locations if he waits longer than we recommended waiting time.

B. RELATED WORK

Unlike public transportation, taxis can pick up passengers at any place, therefore, the GPS data of taxi trajectories are often used in taxi-service related researches. Existing researches on taxi trajectories fall into three categories: recommending routes to find next passengers, guiding urban planning and social functions, and recommending passengers waiting for vacant taxis.

As for identifying the best route to find next passengers, the recent research [7] focused on recommending the most profitable seeking route for finding the next passenger. The current location of a taxi driver given, the MNP algorithm they proposed would recommend a sequence of five consecutive road segments for the taxi. Rong *et al.* [8] modeled the passenger seeking process as the Markov Decision Process and found the best move for a vacant taxi to maximize the total revenue in one time slot. Castro *et al.* [9] proposed a traffic density model and gave an accurate forecast for passenger flow and transportation flow. Miao *et al.* [10] proposed a receding horizon control approach for taxi dispatching with real-time sensing data in metropolitan areas.

For the purposes of guiding urban planning and social functions, such as urban flow, land using, or prediction of urban human mobility, the potential use of taxi traces was discussed for urban land-use classification and particularly for recognizing the social function of urban land by using the trace data from 4000 taxis in [11]. Castro *et al.* [12] presented a historical overview of the research work in this field and discussed which areas hold most promise for future research. The relationship between the getting-on/dropping-off characteristics of taxi passengers and the social function of city regions was established, and then the method of measuring social functions of city regions from large-scale taxi behaviors was presented in [13], [14]. Zhang *et al.* [15] mined the semantics of OD flows from raw GPS trace data and proposed a novel way to explore the human mobility and location characteristic. To address the problem that noisy expert behaviors degrade the IRL performance significantly, a robust IRL framework was developed in [16]. Liu *et al.* [17]–[19] proposed a novel, non-density-based approach called mobility-based clustering to perceive the vehicle crowdedness in nearby areas by using their instant mobility, and then they proposed a Gaussian Process Dynamic Congestion Model for analyzing rationality from trajectories in terms of a set of impact factors. A novel approach based on matrix completion (MC) was proposed to recover the successive missing and corrupted data in [20]. For better prediction of cab drivers' future behaviors, a comprehensive study was carried out to reveal how the social propagation affects the drivers, and then the authors proposed a bio-inspired mechanism providing scalar and temporal aggregation of samples in [21]. For the purpose of developing robust traffic management schemes, Abdelghany *et al.* [22] presented a decision support system for proactive-robust traffic network management. The topic of these work serves as indirect supportive knowledge base for this paper.

The third category of work learns knowledge from large-scale taxi data for recommending where to wait for vacant taxis. Until now, there are two categories for recommendations: time-based and distance-based. The time-based strategies use the historical data to predict the possible waiting time when a passenger can catch a vacant taxi somewhere, and then recommend the location for the passenger with the shortest time. Zheng *et al.* [24] proposed their methods based on the observations: the higher proportion of time with vacant taxis parking beside the road and the more vacant taxis leaving a road, the more chances for the passengers to take a taxi quickly. Jing *et al.* [25] used Hadoop to process large amounts of taxi data and they took weather factors into consideration. Qi *et al.* [26] took into account the situation of the queue, i.e. there may have been someone waiting for a taxi there when the passenger arrived at a certain road segment. The distance-based ones usually recommend the hot area nearest to the passenger. Yuan *et al.* [23] proposed an approach to make recommendations for both taxi drivers and passengers based on passengers' mobility patterns and taxi drivers picking-up/dropping-off behaviors learned from the GPS trajectories of taxicabs.

From the above researches, they only focused on one-location recommendation for the current passengers without considering the failure situation where the passengers can't catch any vacant taxi. So, if a passenger waits at a location for an estimated time without catching any vacant taxi, walking to neighborhood location may be a better choice.

C. PROPOSED SCHEME

In our previous work [6], we proposed a Markov Decision Process approach to address the limitations of related work. In this strategy, we recommended a sequence of road segments to passengers for catching vacant taxis. We first recommended the passenger to go to a road segment nearest to him, then we got the next road segment sequence based on the first step. The probability of catching vacant taxis for passengers was much higher than current researches. However, there are also some shortcomings in our previous strategy, such as the low rate of passing vacant taxis on a road segment and the multi-passengers competition for taxis at the same location and time slot, etc..

Based on our previous scheme, we further find that one passenger waiting for taxis at a crossroad will have higher probability of catching vacant taxis, because one crossroad is usually connected to multi-road segments, where more vacant taxis pass by. So, we further propose a Crossroad Network-based Markov Decision Process approach. In the first step, our scheme is to recommend a nearby road segment which can maximize overall probability of the whole waiting process instead of the nearest one. Next, we will recommend a waiting location sequence to current passengers. This sequence includes several crossroads instead of road segments.

The ***road segment*** we are talking about refers to the major urban road in the road network. The ***crossroad*** refers to the area near the crossroad. It is only for the passengers to find more taxis, not the crossroad center, because it is generally not allowed to get on or off the taxi there.

And ***why is our plan divided into two steps***? When passengers leave their houses and wait for the taxis, they often follow the path of the home to the main road instead of going straight to a crossroad and they usually will walk shorter because of the vertical distance between that path to a road segment, so it is in line with the reality and the passengers' habits to first recommend the passengers to a road segment.

We use the pass rate of vacant taxis to evaluate road segments and crossroads, because the higher the pass rate of a vacant taxi is, the more vacant taxis pass by the road segment and crossroad, and the passenger is more likely to catch a vacant taxi. So we need to obtain the pass rate of vacant taxis on each road segment and crossroad in specified time slots by mining from our history taxi mobility.

We also use Non-Homogeneous Poisson Process (NHPP) [27] to simulate the event of vacant taxi passing by, and then derive the waiting time that the passenger is advised to take to wait at recommended locations. Then, we propose a recommendation approach based on Markov Decision Process (MDP) model to select the crossroad sequence with the highest total pass rate for the current waiting passenger.

Further, we extend our scheme to a competitive situation where multi-passengers compete for vacant taxis at the same location within the same time slot. We present the optimal location sequence for different waiting passengers respectively by dynamically updating the pass rate of vacant taxis at each crossroad and road segment.

D. OUR CONTRIBUTIONS

In this paper, we substantially improve the approach in our previous work by generalizing the MDP approach to crossroad networks, extending our scheme to a multi-passengers competitive situation at the same location within the same time slot, and conducting extensive experiments and simulations using real data. Specifically, the major contributions of this paper are summarized as follows:

- We define the pass rate of vacant taxi as the number of vacant taxi passing per unit time, and use NHPP to simulate the pass behavior of a vacant taxi for deriving the passenger's waiting time.
- We propose a Crossroad Network-based MDP (CN-MDP) model to predict the location sequence for catching vacant taxis, and present a multi-passengers competition solution by dynamically updating the pass rate of vacant taxis.
- We conduct intensive case studies and experiments to test the effectiveness and time cost of our proposed CN-MDP scheme.

The rest of the paper is organized as follows. Section II introduces the data set we used and presents analysis settings. Section III states the problem to be solved and presents some formal descriptions. Section IV improves our previous approach and propose a Crossroad Network-based Markov Decision Process approach. Section V presents the experiment results and compares our scheme with other ones. Finally we conclude the paper in section VI.

II. DATA AND MODEL SETTINGS

In this section, we describe the taxi data we used and present analysis settings. Then, we introduce the Markov Decision

TABLE 1. Taxi record detail.

Field Name	Meaning
BUSINESSHIS_ID	Transaction ID (the primary key)
STAMP	The time of the data submitted to the system
UNIT_ID	ID of the current operation taxi
ONTIME	The time of one passenger gets on the taxi
ONLON	The longitude of one passenger gets on the taxi
ONLAT	The latitude of one passenger gets on the taxi
ONANGLE	The angle of the taxi head deviates from the north direction when one passenger gets on the taxi
OFFTIME	The time when one passenger gets off the taxi
OFFLON	The longitude of one passenger gets off the taxi
OFFLAT	The latitude of one passenger gets off the taxi
OFFANGLE	The angle of the taxi head deviates from the north direction when one passenger gets off the taxi
RUNLEN	Current trip mileage
RUNMONEY	Current fare for the trip
RUNTIME	Current operational time

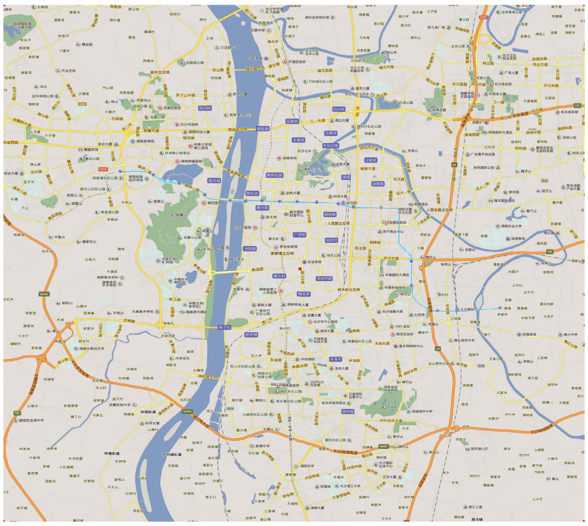


FIGURE 1. **Our study area.**

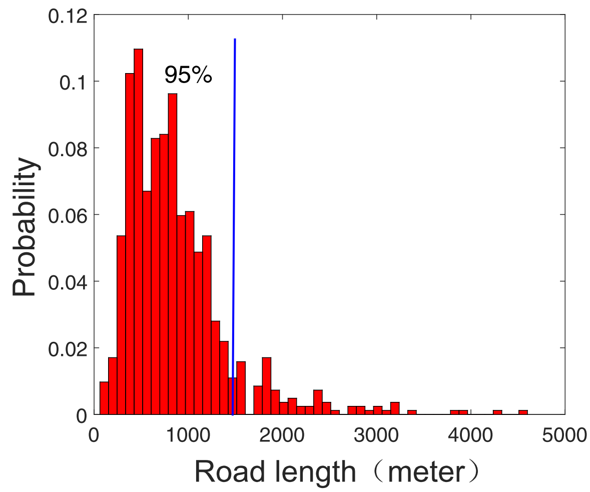


FIGURE 2. **Road's length frequency distribution.**

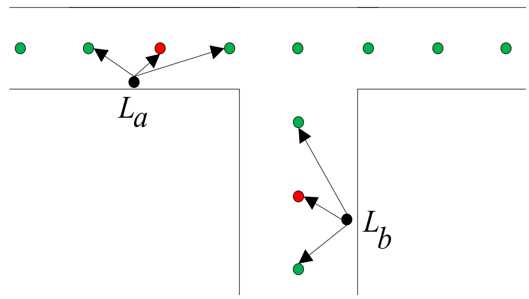


FIGURE 3. **Road matching.**

Process (MDP) and set threshold parameters by actual state constraints.

A. DATA BACKGROUND

The data used in this paper come from the taxi operation data of Changsha, a provincial capital city of China. The data collection period covers 24 hours per day and the number of taxis included in the data exceeds 1400. The raw record table is showed in table 1.

We mainly study the taxi operation data within the scope of Changsha, China, so the GPS records outside this area should be filtered out. We get the research range from 112.474039 to 113.413451 (longitude) and 28.429221 to 28.039691 (latitude) respectively by using Baidu Map [28]. Then, we use a 25.6km × 25.6km bounding box to define the study area. The map of our study area from Baidu Map is shown in Fig. 1.

B. MAP MATCHING AND ANALYSIS SETTINGS

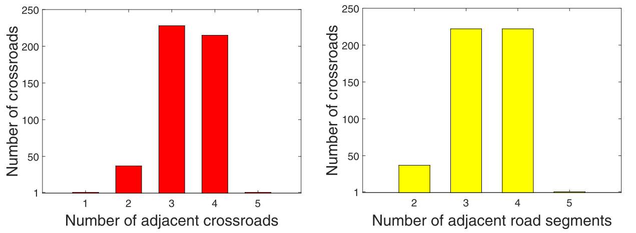
From the historical records of our taxi data set, we only get the latitude and longitude information of the getting-on/dropping-off locations, so we need to match the information to the road network for determining which road segments the locations belong to and restoring the whole trajectories of all taxi trips.

By Baidu Map matching, we get 821 major urban roads and 1633 crossroads on the road network. The road network can be seen as an undirected graph with the road segment as an edge and the crossroad as a vertex. The length of each road segment can be regarded as the weight of the edge. We get the distribution of length of all road segments, shown in Fig. 2, from which we can know that more than 95% of the length of road segments is less than 1500 meters.

In addition, we can also get a series of equally spaced sample points with the interval of 100 meters on each road, and each sample point has a label indicates which road segment it subordinates to, so the getting-on/dropping-off locations can be matched to one of the sample points by using the map matching algorithm based on greedy strategy [29], and then we can know which road segments the locations belong to.

The matching algorithm is described as Fig. 3, where the black points denote the current getting-on/dropping-off locations, the green points describe the sample points along the road segments. For current getting-on/dropping-off locations, the nearest sample points are denoted as the red spots.

In the matching process, we suppose that the distance between getting-on/dropping-off locations and the neighborhood road segments should not exceed 200 meters. Otherwise, these locations are considered to be deviated from the



(a) The adjacent relationship between crossroads and crossroads. (b) The adjacent relationship between crossroads and road segments.

FIGURE 4. Road network distribution.

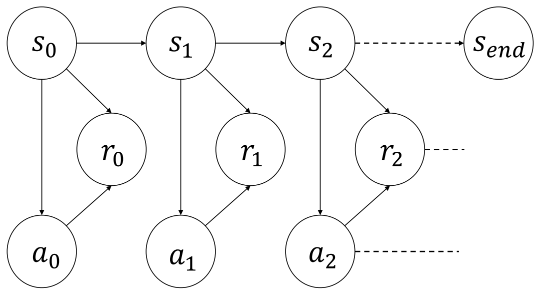


FIGURE 5. MDP state transition process.

main road network, and cannot be mapped into any road segment. So, these locations should be filtered. According to this rule, we can match 89.15% of the dropping-off locations to the corresponding road segments and 92.27% of the getting-on locations to the corresponding ones.

Based on the actual road network, we also analyze the adjacent situation about crossroads. As we see in Fig. 4(a), the horizontal axis is the number of crossroads that each crossroad connects with, and we get the adjacent crossroad number, which is 5 at most and 1 at least. The vertical axis is the number of crossroads that have the corresponding number of neighboring crossroads, which shows most crossroads are connected to 3 or 4 other crossroads. In the similar, we can learn most crossroads are connected to 3 or 4 road segments from Fig. 4(b). So from Fig. 4, we can see that the actual road network is not peculiarly complicated.

C. MARKOV DECISION PROCESS (MDP)

The Markov Decision Process (MDP) is a random decision process with a triple: $M = (S, A, R)$, where three parameters are represented respectively as: state set, action set, and reward set. For the actor, it will change its state after performing an action and it will get a corresponding reward at the same time. Just like Fig. 5, the initial state of an actor is $s_{start} = s_0$ and the action a_0 is selected from A. The actor transfers to the next state s_1 and obtains a reward $r_0(s_0,a_0)$ after finishing the action and this process continues until the end state s_{end} is reached, i.e. the stop condition is satisfied.

So the MDP will result in a sequence of actions called a ***strategy***. But there may be many strategies from s_{start} to s_{end}, which one is better? This is measured by the ***value function*** that consists of the reward of each step. The common form of MDP value function is shown as (1) :

$$\begin{aligned} V(s_0) &= E[\sum_{i=0}^{\infty}(\gamma^i \cdot r_i)|s_{start} = s_0] \\ &= E[r_0 + \gamma \cdot r_1 + \gamma^2 \cdot r_2 + \cdots |s_{start} = s_0] \\ &= r_0(s_0, a_0) + \gamma \cdot V(s_1) \end{aligned} \tag{1}$$

The (1) represents the total reward obtained by a certain strategy with the state s_0 as the initial state. In (1), $\gamma \in [0, 1]$, which determines the net present value of the future rewards, i.e. the greater the γ, the more important the future reward is. Finally, the MDP will recommend the strategy with the biggest total reward to the actor.

Our goal is to get the best waiting location sequence for passengers with the constraints of time and distance, which satisfies well the conditions of MDP model. Because our recommendation is under the random waiting taxi mode, choosing where to wait for vacant taxis is accompanied by the passenger's own decision and each choice will change his location, the process of choosing where to catch a vacant taxi can be considered as a random decision process of a passenger as same as the MDP model. So the (S, A, R) in our problem are as follows:

1) For a passenger, waiting for vacant taxis on a road segment or crossroad is his state, i.e. S is the passenger's waiting location.
2) Changing the waiting location is passenger's action, i.e. A, which will change his state.
3) Obviously, the probability of catching vacant taxis is different at different locations and time slots. so this probability is the reward of the passenger's action, i.e. R.

D. THRESHOLD PARAMETERS SETTING

In addition, the actor will stop at an end state in the MDP model. Similarly, we also have the end state when the thresholds are reached. In real life, passengers actually hope not to go far to catch vacant taxis, so their walking distance must have a threshold, called ***distance threshold***. In the same way, passengers also hope to catch the vacant taxis in the shortest time, the time spent by passengers should have a threshold, called ***time threshold***.

So in our recommendation process, the terminated condition of one recommendation includes: the total distance walked by the passenger is not greater than the distance threshold, and the time spent by the passenger is less than time threshold. In other words, the terminated condition is the condition that our recommendation algorithm ends, our algorithm will stop computing and return the final waiting location sequence when that condition is reached.

Furthermore, the threshold setting starts when the passenger reaches the first road segment, because the passengers are not on the main road before that time and the taxis are only allowed to run on the main road, so passengers will not encounter any vacant taxi before they arrive at the first road segment, which can be ignored.

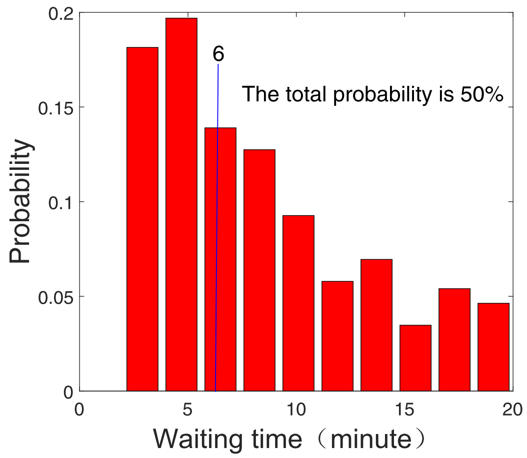


FIGURE 6. Waiting time distribution for 10-11 o'clock.

1) SET DISTANCE THRESHOLD

If the walking distance threshold is too small, the distance between the current passenger's location and the adjacent crossroads may be larger than the threshold value, there may be only one adjacent crossroad that can be recommended or even worse, no one to be recommended. However, if the threshold value is set too large, passengers may have too long a distance to walk. So, how to determine the walking distance threshold?

Our threshold setting begins only when the passenger arrives at the first road segment, and we assume that the passenger arrives at the middle of the road segment under most circumstances. So, the walking distance threshold should be set at least half of the length of the road segment, which ensures that there is at least one adjacent crossroad that can be recommended to the current passenger. It can be seen from Fig. 2, more than 95% of the length of road segments is less than 1500 meters. Obviously, the distance threshold should be set to 750 meters, which ensures there is at least one crossroad to be recommended to the passengers at 95% of the road segments.

2) SET TIME THRESHOLD

The time spent by passengers includes two parts: the time spent on waiting vacant taxis at each location and the time spent on walking to adjacent crossroads, so the threshold setting for time is more complicated.

First we discuss the waiting time threshold. Because the waiting time is different in different time slots, the waiting time threshold should vary with different time slots. We calculate the waiting time of all crossroads in different time slots using the data set (the calculation method will be described later), because the waiting time is too long in some crossroads and passengers will not wait too long, so we select a time that is bigger than 50% of the crossroads' waiting time as the waiting time threshold in different time slots, which ensures the waiting time is less than the threshold on half of the crossroads. We present the frequency distribution of the waiting time, an example is shown as Fig. 6. We can learn the waiting time threshold for 10-11 o'clock should be 6 minutes, because the waiting times on 50% crossroads are less than 6 minutes.

Next we discuss the walking time threshold. Since we suppose the distance threshold to be 750 meters in the above subsection, therefore, passengers will not walk more than 750 meters during the whole process. And the normal walking speed of people is almost 1.5m/s, so passengers will not spend more than 8 minutes walking. Therefore, we set walking time threshold to 8 minutes, and it is the same for different time slots, because the distance threshold is the same with those in different time slots.

To the end, the time threshold in different time slots is the waiting time threshold in the corresponding time slot add the walking time threshold, i.e. 8 minutes.

(***the two threshold parameters vary with the data set***.)

III. PROBLEM STATEMENT

In this section, we present our analysis to evaluate a good strategy. Based on this analysis we formulate the problem as an optimal decision problem.

A. THE GOOD STRATEGY FOR WAITING TAXIS

For a passenger, what's the good waiting strategy? The answer is getting on taxis as soon as possible. While the number of vacant taxis passing varies in different places and different time slots. As we know, there may be many vacant taxis passing by in some places, the passengers can catch vacant taxis quickly. While there are few taxis passing by in other places in the same time slot, the passengers can't get on vacant taxis for a long time.

Therefore, the probability of vacant taxis passing by is the key factor to determining whether one passenger can catch a taxi soon. We call it ***vacant taxi pass rate***. On the other hand, since we suppose only one passenger is waiting the vacant, he can catch a vacant taxi once a taxi passes (the multi-passengers competition can also be converted to the one-passenger situation, which will be explained in detail later). So the probability of passengers catching a vacant taxi equals the vacant taxi pass rate.

So, we can use the vacant taxi pass rate to evaluate the waiting strategies. In general, the strategy that passengers can catch vacant taxis with the higher probability and shorter waiting time is a good one.

B. PROBLEM STATEMENT

Our scheme tries to recommend a waiting location sequence to current passengers for catching vacant taxis with the maximum calculative probability, which can improve passengers' experience. We firstly recommend a road segment (nearby the passenger's location). Next, we recommend a sequence of crossroads to passengers. In order to precisely define the outcome of our proposed scheme, we present the following definitions:

- ***Road Segment***: The road segment refers to one street that starts and ends at intersections but contains no intersection within the street, denoted as r. For each road segment, we assign a unique sample label r_i for it, and $r_i.s$ indicates the start point of r_i, $r_i.e$ indicates the end point of r_i, and $r_i.l$ indicates the length of r_i.
- ***Crossroad***: The crossroad is the intersection of multi-road segments, denoted as c. For each crossroad, it has the unique sample label c_i, which equals the start/end point of r_i above, and the road segment set that connects it $\psi(c_i) = \{r_1, r_2, r_3, \cdots, r_n\}$.
- ***Vacant Taxi Pass Rate***: It refers to the probability that vacant taxis pass by one road segment r_i or crossroad c_i, which is also related to the current time slot t, denoted as $P^t_{r_i}$ or $P^t_{c_i}$.
- ***Waiting Time***: It refers to the time from when passengers arrive at one road segment r_i or crossroad c_i to when they catch vacant taxis, which is also related to the current time slot t, denoted as $T^t_{r_i}$ or $T^t_{c_i}$.
- ***Waiting Route***: It means a sequence of waiting locations we recommend for passengers, including one road segment and the sequence of crossroads: $\pi = \{r_1 \to c_1 \to c_2 \to \cdots \to c_n\}$. We can also get the r_i between c_{i-1} and c_i $(i = 2, 3, \cdots, n)$, and we assume the passenger starts from the middle of the first road segment, so the length of a waiting route could be calculated by adding the length of road segments: $\pi.l = \frac{r_1.l}{2} + \sum_{i=2}^{n} r_i.l$. And the total waiting time could be calculated by accumulating the waiting time of each location in the route: $\pi.T = T^t_{r_1} + \sum_{i=1}^{n} T^t_{c_i}$.

To this end, our problem is formally defined as follows:

Input:

- The passenger's location L.

- The time slot t when the passenger issues a waiting request.

- The historical getting-on/dropping-off taxi data.

- The underlying spatial network.

Output: A waiting route: $\pi = \{r_1 \to c_1 \to c_2 \to \cdots \to c_n\}$ (r_1 is near to L) with the biggest total probability of catching vacant taxis.

Constraints: The $\pi.l$ and $\pi.T$ can't exceed the thresholds we set before, because passengers are unwilling to walk too far or spend too long a time.

IV. A CROSSROAD NETWORK-BASED MARKOV DECISION PROCESS APPROACH

In earlier work, we designed a simplified recommendation strategy by Markov Decision Process [6]. Now, we generalize our model to crossroad networks and model the waiting strategy into a Crossroad Network-Based Markov Decision Process (CN-MDP). Solving this CN-MDP model will give us the best moving strategy for catching vacant taxis at each different location. We further discuss an updating approach of dynamic pass rate to satisfy multi-passenger competition for taxis in the same location and the same time slot.

A. THE STATES AND ACTIONS OF CURRENT PASSENGERS

In our scheme, recommending location sequence for passengers is divided into two steps: in the first step, passengers are advised to wait at one adjacent road segment for suggested time; in the second step, the crossroads sequence is recommended to passengers with the suggested waiting time at each crossroad. Therefore, it is necessary to discuss the road segments and crossroads separately because only one road segment is recommended in the first step, while more than one crossroad will be involved in the second step. Then, the states and actions of the current passengers are defined as follows:

- ***State Set***: $S = \{S_r, S_c\}$. S_r is the state of passengers on the road segment, which includes road segment label and current time slot, we record it as: $S_r = (r_i, TS)$. Because only the first step is related to the road segment, the road segment state has only one, i.e. the initial state of the whole process is $S_{start} = S_r$. S_c is the state of passengers at the crossroad, which includes the crossroad label and current time slot, we record it as: $S_c = (c_i, TS)$.
- ***Action Set***: $A = \{E, S, W, N\}$. This set represents passenger's actions: walking to the east, south, west and north directions, by which they can change the waiting locations, i.e. change the states of passengers.

B. STATE TRANSITION AND OBJECTIVE FUNCTION

Our goal is to recommend a waiting route to passengers with the maximum probability for catching vacant taxis, while how to calculate the probability? In Section II-C, we explain why our problem can be solved by MDP model. So, we design objective functions based on MDP model to calculate the calculative probability. We firstly define the reward of the passenger's actions, which is defined as follows:

- ***Reward Set***: $R = \{R_r, R_c\}$. The set includes the reward of moving to a road segment and crossroad respectively. Since we take the probability of catching vacant taxis as a measure indicator, so, we use the vacant taxi pass rate as the reward i.e., $R_r = P^t_r$, and $R_c = P^t_c$.

In our scheme, we choose the best action by solving value functions, and then design two value functions because of the process divided into two parts. The first value function is $V_c(c)$, which calculates the total return value of the crossroad sequence starting with a crossroad c, and the second value function is $V_r(r)$, which calculates the total return value of the process starting by a road segment r.

In fact, when a passenger arrives at a crossroad c to wait for a vacant taxi, there are two situations: (i) he can catch a vacant taxi immediately, so, the reward is the vacant taxi pass rate; (ii) otherwise, we recommend him to the adjacent crossroad c_{next} to wait, so the reward is the product of the probability of catching no vacant taxis at c and the maximum total return value of subsequent crossroad sequence starting by c_{next}. So the $V_c(c)$ is shown as (2):

$$V_c(c) = R_c + (1 - R_c) \cdot \max\{V_c(c_{next})\} \quad (c_{next} \text{ is near to } c) \qquad (2)$$

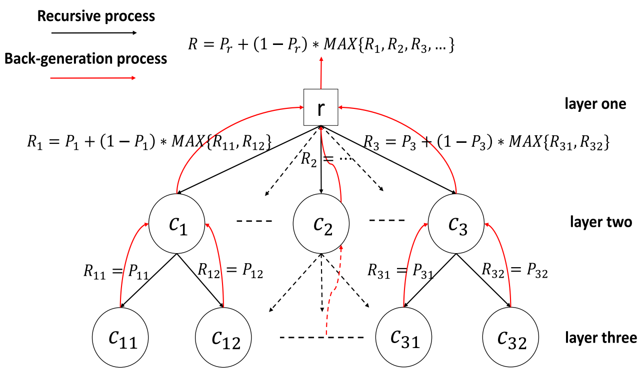


FIGURE 7. State transition of passengers.

In a similar way, the $V_r(r)$ is shown as (3):

$$V_r(r) = R_r + (1 - R_r) \cdot \max\{V_c(c_i)\} \quad (c_i \text{ is near to } r) \tag{3}$$

Next, we use the state transition of the passenger to explain how the two value functions work, shown as Fig. 7.

From Fig. 7, the passenger is recommended to wait at road segment r firstly. Then, our approach finds the adjacent crossroads connected to r: c_1, c_2, c_3, and calculates their return values respectively. In the end, we recommend the passenger to go to the crossroad with the largest return value.

However, the return value of crossroad c_1 is not only related to the reward of this crossroad itself, i.e. the vacant taxi pass rate, but also related to the return value of the following adjacent crossroads. Therefore, under the premise of meeting the distance and time threshold value, we also should consider other crossroads connected to c_1: c_{11}, c_{12}, and continue recursion. When the recursive procedure reaches the third layer that breaks the distance or time threshold, the downward recursion will be stopped, and it is necessary to go back and return the biggest vacant taxi pass rate (i.e. the reward) on the c_{11}, c_{12} to the second layer, then we calculate the total return value of c_1 (c_2, c_3 are also calculated in the similar way) and return it to the first layer. This is the working process of the value function $V_c(c)$. Finally, at the first layer, the total return value of the entire solution is calculated by the value function $V_r(r)$, which will call the $V_c(c)$.

In the whole process of the passenger waiting for a vacant taxi, the current passenger may not necessarily happen to be on a road segment. So our ultimate goal is to find one waiting sequence that gets the largest value of $V_r(r)$ that is the objective function denoted by (4), where L denotes the start location of the current passenger making a request.

$$V_\pi(L) = \max\{V_r(r_i)\} \quad (r_i \text{ is near to } L) \tag{4}$$

C. LEARNING CN-MDP PARAMETERS

From the above analysis, we know that the key to solving the value functions is to get the reward of each road segment and crossroad, i.e. the vacant taxi pass rate on the road segment and crossroad. At the same time, we also need to suggest each appropriate waiting time on the initial road segment and subsequent crossroads.

TABLE 2. Taxi trajectory data.

Field Name	Meaning
StartRoad	The label of road segment that one passenger drops off the taxi
EndRoad	The label of road segment that one passenger gets on the taxi
TimeSlot	The current time slot
Total	The vacant taxis number on one trajectory

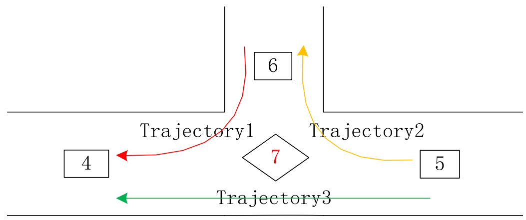


FIGURE 8. An example of taxis' trajectories.

1) VACANT TAXI PASS RATE

The pass rate of vacant taxi is mathematically interpreted as the number of vacant taxi passing by one road segment or crossroad during a unit time. While the pass rate of vacant taxi at each time slot is different, e.g. the number of vacant taxis for 7-8 o'clock is different from that for 15-16 o'clock in the same road segment, because the former is at the morning peak with more taxis. So the vacant taxi pass rate of the road segment and crossroad can be defined as (5):

$$P_r^t = \frac{PassNum_r^t}{|t|}; \quad P_c^t = \frac{PassNum_c^t}{|t|} \tag{5}$$

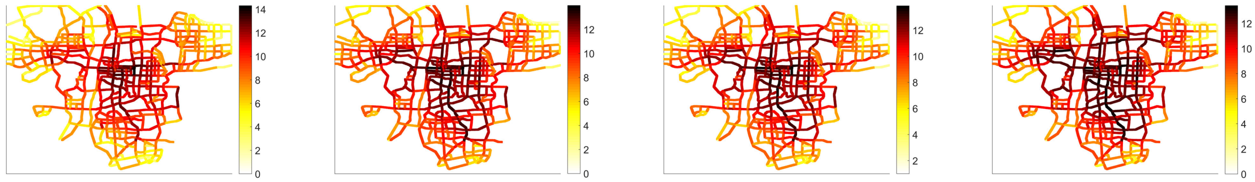
The $PassNum_c^t$ is the number of vacant taxis that pass by one crossroad c within the time slot t, the $|t|$ means the length of time slot and is usually set as 60 minutes in our scheme (***it also varies with the data set***).

From equation (5), we just need to get the $PassNum_c^t$. Then, we calculate the vacant taxi pass rate from the records of table 2 that records the vacant taxis on the trajectory from *StartRoad* to *EndRoad*. A trajectory covers multi-road segments and crossroads, so, we can calculate the number of vacant taxis on road segments and crossroads by the trajectories.

For example, as shown in Fig. 8, there are three trajectories, so the number of vacant taxis on road segment r_6 is the sum number of the vacant taxis on the trajectory 1 and 2, similarly, the number of vacant taxis on crossroad c_7 is the sum number of vacant taxis on the trajectory 1, 2 and 3, which shown as (6).

$$\begin{aligned} PassNum_{r_6}^t &= PassNum_{Tra1}^t + PassNum_{Tra2}^t; \\ PassNum_{c_7}^t &= PassNum_{Tra1}^t \\ &\quad + PassNum_{Tra2}^t + PassNum_{Tra3}^t \end{aligned} \tag{6}$$

We calculate the number of vacant taxi on each road segment and crossroad in each time slot. Fig. 9 shows a hot map of the global distribution of vacant taxi's number passing by each road segment in whole year for four time slots (The color bar uses the exponential representation, 2 i.e. $2^2 = 4$, 4 i.e. $2^4 = 16$, and so on).



(a) The vacant taxi's number (log scale) for 7-8 o'clock. (b) The vacant taxi's number (log scale) for 10-11 o'clock. (c) The vacant taxi's number (log scale) for 15-16 o'clock. (d) The vacant taxi's number (log scale) for 18-19 o'clock.

FIGURE 9. The global hot map of vacant taxi's number in four time slots (best viewed in color).

Obviously, the vacant taxi number of passing by crossroads is larger than that of passing by road segments, so passengers are more likely to catch vacant taxis at crossroads.

2) WAITING TIME

When taxis arrive or leave, passengers get on or drop off, the occurrence of these arriving or leaving events is consistent with the distribution of Non-Homogeneous Poisson Processes (NHPP). Therefore, we can use NHPP to simulate the vacant taxi's passing actions.

We define the parameter of NHPP $\lambda_{(t)} = t \cdot P_c^t$, where P_c^t represents the vacant taxi pass rate of crossroads varying with different time slots. So, the NHPP distribution of the number of events is described below in (7):

$$P\{OccurNum_{(t)} = k\} = \frac{e^{-(t \cdot P_c^t)} \cdot (t \cdot P_c^t)^k}{k!} \tag{7}$$

where $OccurNum_{(t)}$ represents the vacant taxi's passing number during the time slot t. Then, the probability of waiting time T is within t is shown as (8) [30]:

$$\begin{aligned} P\{T \leq t\} &= 1 - P\{TW > t\} \\ &= 1 - P\{OccurNum_{(t)} = 0\} \\ &= 1 - e^{-(t \cdot P_c^t)} \\ &= F_{(t)} \end{aligned} \tag{8}$$

We can finally calculate the mean of t by (9) :

$$\begin{aligned} E(t) &= \int_0^{\infty} t \cdot F'_{(t)} \cdot d_t \\ &= \int_0^{\infty} t \cdot P_c^t \cdot e^{-(t \cdot P_c^t)} \cdot d_t \\ &= \frac{1}{P_c^t} \end{aligned} \tag{9}$$

We can get the waiting time on the crossroad c within time slot t to be $T_c^t = \frac{1}{P_c^t}$. In the similar way, the waiting time on the road segment r within time slot t is $T_r^t = \frac{1}{P_r^t}$.

D. SOLVING THE TWO VALUE FUNCTIONS

From the above analysis, we need to calculate the two value functions: $V_r(r)$ and $V_c(c)$. So we design two algorithms: **W**ait **R**oute and **T**ime (WRT) and **T**otal **P**ass **R**ate (TPR), to solve them respectively through the process of recursive re-generation shown in Fig. 7. And we also have another two auxiliary algorithms: **W**ait **R**oute (WR) and **W**ait **T**ime (WT), to get the final waiting route and waiting time on each location. Next we will introduce these four algorithms in detail.

Algorithm 1 Total Pass Rate (TPR)

Input: Current crossroad c, Current time slot TS, The distance the passenger has walked d, The waiting time the passenger has spent t, Walking distance threshold ΔD, Time threshold ΔT.

Output: Total return value of waiting sequence starting with crossroad c: $V_c(c)$.

1 **if** $d > \Delta D$ *or* $t > \Delta T$ **then**
2   return 0;
3 **end**
4 $max\_rate = 0$;
5 $next\_c = c$;
6 **for** *each* c_i *nearby* c **do**
7     $tem\_rate = TPR(c_i, TS, Distance(c_i, c) + d, t + T_c^{TS})$;
8   **if** $tem\_rate > max\_rate$ **then**
9         $max\_rate = tem\_rate$;
10         $next\_c = c_i$;
11   **end**
12 **end**
13 $DadCrossroad[next\_c] = c$;
14 return $P_c^{TS} + (1 - P_c^{TS}) \cdot max\_rate$;

1) INTRODUCE ALGORITHM TPR

The pseudo-code of algorithm *TPR* is shown as algorithm 1.

In algorithm *TPR*, if the distance the passenger has walked is greater than the distance threshold or the time he spent is more than the time threshold, our algorithm will stop and return to 0, i.e. there is no vacant taxi on c. Otherwise, we select the crossroad with the maximum total return value near c, and set it as the next node of c with an array (*DadCrossroad*). The above process belongs to the lines from 6 to 13. At last, we get the total return value of this crossroad sequence starting by c, i.e. $V_c(c)$.

Under the constraint of thresholds, the number of crossroads in the crossroad sequence usually does not exceed 3, so, we can get the time complexity with $\mathcal{O}(n^3)$, n is the

Algorithm 2 Wait Route and Time (WRT)

```
Input: Passenger's initial location L, Current time slot
       TS, Walking distance threshold ΔD, Waiting
       time threshold ΔT.
Output: Total return value of the entire waiting route:
        V_r(r).
1  road = 0;
2  max_rate = 0;
3  for each r_i nearby L do
4      for each c_i nearby r_i do
5          tem = P_{r_i}^{TS} + (1 − P_{r_i}^{TS}) · TPR(c_i, TS, r_i.length/2, T_{r_i}^{TS});
6          if tem > max_rate then
7              max_rate = tem;
8              road = r_i;
9              CrossList.clear();
10             CrossList = WR(c_i);
11             DadCrossroad.clear();
12         end
13     end
14 end
15 WaitTime = T_{road}^{TS} ∪ WT(CrossList, TS);
16 WaitingRoute = road ∪ CrossList;
17 return max_rate;
```

number of crossroads near c. From Fig. 4(a), obviously, where n is not bigger than 5, so, the time complexity is constant and the spatial complexity is $\mathcal{O}(1)$.

2) INTRODUCE ALGORITHM WRT

The pseudo-code of algorithm *WRT* is shown as algorithm 2.

In algorithm *WRT*, we first select each road segment around the passenger's start location L called r_i. Then we select the sequence of crossroads near r_i by means of the algorithm *TPR* and calculate the $V_r(r_i)$, by the way, we select the biggest one as the total return value of the entire waiting route and the corresponding r_i as *road*. At last, we use a list (*CrossList*) to recode this crossroad sequence derived by the algorithm *WR*. The above process belongs to the lines from 3 to 14. After that, we get the waiting time of the selected crossroad sequence, which is derived by the algorithm *WT*. So the entire waiting route is the selected road segment and *CrossList*, and the list (*WaitTime*) records the corresponding waiting time of them. Finally, we get the total return value of the entire waiting route, i.e. $V_r(r)$.

Similar to the algorithm *TPR*, the time and spatial complexities of algorithm *WRT* are also constant.

3) INTRODUCE TWO AUXILIARY ALGORITHMS

The pseudo-code of algorithm *WR* and *WT* are shown as algorithm 3 and 4.

In algorithm *WR*, we firstly add current crossroad c, which is actually the first crossroad in sequence, to the list. Then, we consider each crossroad c_i in our database to see whether it's the next node of c dealt by the array (*DadCrossroad*).

Algorithm 3 Wait Route (WR)

```
Input: Current crossroad c.
Output: Crossroad sequence CrossList.
i = 1;
CrossList.add(c);
while i <= |CrossSet| do
    if DadCrossroad[c_i] = c then
        CrossList.add(c_i);
        c = c_i;
        i = 1;
    end
    else
        i = i + 1;
    end
end
return CrossList;
```

Algorithm 4 Wait Time (WT)

```
Input: Crossroad sequence CrossList, Current time slot
       TS.
Output: Waiting time of each crossroad in crossroad
        sequence CrossTimeList.
for each c in CrossList do
    CrossTimeList.add(T_c^{TS});
end
return CrossTimeList;
```

If c_i is the next node of c, we add it to the list and set it as the current crossroad and restart, otherwise we consider next crossroad c_{i+1}. At last, we return to the crossroad sequence beginning with c. The time complexity of algorithm *WR* is $\mathcal{O}(|CrossSet|)$ ($|CrossSet|$ is the number of crossroads in database), which is also constant, and the spatial complexity is $\mathcal{O}(1)$.

In algorithm *WT*, we just find the waiting time of each crossroad in the crossroad sequence and add it to a list. The time complexity of algorithm *WT* is $\mathcal{O}(|CrossList|)$ ($|CrossList|$ is the number of crossroad sequence), which is also constant and the spatial complexity is $\mathcal{O}(1)$.

E. MULTI-PASSENGERS COMPETITION

For a single passenger, we recommend a waiting sequence to him for catching a vacant taxi with the highest probability. But what should we do in the case of competitive situation, i.e. there may be more than one passenger making a request in the same time slot at the same location?

The waiting sequence we recommended is specific for the current passenger and the probability is calculated based on the vacant taxi pass rate in the time slot by the historical data. But the vacant taxi pass rate is stable in the same time slot, so in the competitive situation, the same recommended route will be recommended to different passengers with competitive relationship, which obviously does not meet our goal.

Therefore, we should give different recommended routes for different passengers.

In fact, the time of each passenger when he makes the request is in order under the competitive situation, although the time is not much difference. After we give a recommendation to a passenger, the vacant taxi pass rate has changed, so the recommendation to next passenger may be different. So, if we update the vacant taxi pass rate in real time, the multi-passengers competition can be converted to the one-passenger situation. However, the vacant taxi pass rate is determined by the number of passed vacant taxis, so we only need to update the number of passed vacant taxis in real time. Thus, we will reduce the number of vacant taxis by one on all waiting locations of the route after recommending to one passenger, because the passenger may catch a vacant taxi at any location on the recommended route, i.e. he will consume a vacant taxi at an uncertain waiting location.

So considering the competitive situation, our main algorithm is **R**ecommendation for **M**ulti-passengers **C**ompetition (RMC), shown as follows.

Algorithm 5 Recommendation for Multi-Passengers Competition (RMC)

Input: Passenger's location L, Current time slot TS, Walking distance threshold ΔD, Waiting time threshold ΔT.

Output: Waiting route *route*.

$route \leftarrow WRT(L, TS, \Delta D, \Delta T)$;

for *each waiting location l in route* **do**

  $PassNum_l^{TS} \leftarrow (PassNum_l^{TS} - 1)$;

end

return *route*;

We firstly use the algorithm *WRT* to get the waiting route for the current passenger. Then we change the number of passed vacant taxis at each location in that route. Finally, we return the route to current passenger. And we continue to do the same recommendation process for the next passenger. The time and spatial complexities of this algorithm are also constant.

V. EVALUATION

In this section, we first give the verification that our scheme is consistent with the actual best solution through actual data. Then, we compare our scheme with the one we proposed before and among other ones. The results show that our strategy has a greater probability for catching vacant taxis. Finally, we discuss the efficiency of our scheme.

A. EXPERIMENTAL SCHEME

Our taxi data set are collected from February 1st, 2012 to February 1st, 2013. We take the data in the first ten months to calculate the vacant taxi arrival rate on each road segment and each crossroad as well as the corresponding waiting time.

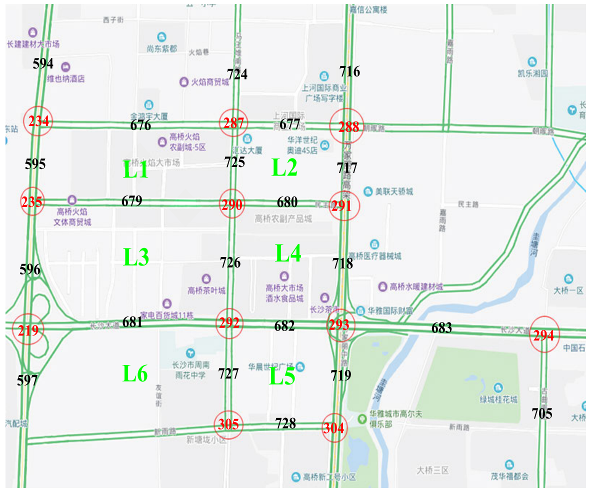


FIGURE 10. Road network used in experiment.

We use the data of last two months as the verification data to verify our results.

First, we randomly generate the locations of 50 passengers respectively and set the corresponding start time (in the remaining two months). Then we use our CN-MDP approach, our previous approach and other ones in related work to recommend for the 50 passengers respectively. Finally, we compare our recommendation result with the actual best route and other ones to verify our CN-MDP approach to be an efficient and good strategy, our previous approach is the second and other ones may be a little bit worse.

The road network used in the experiment is such as Fig. 10. The round frame is the crossroad label, and 50 passenger locations are randomly generated: $L_1, L_2, L_3, \ldots, L_{50}$.

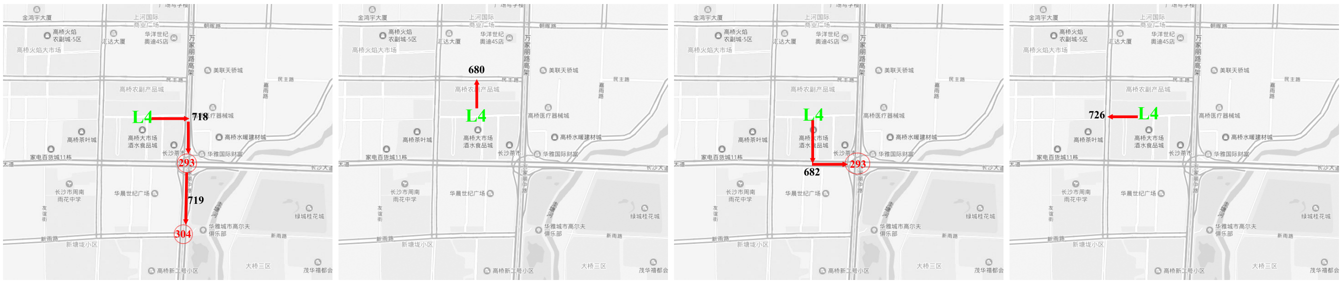
The algorithm is implemented with Java in MyEclipse 8.6. The experiment works on a machine with an Intel(R) Core(TM) i7 2.40GHz CPU and 8GB of main memory.

B. EFFECTIVENESS VERIFICATION FOR CN-MDP

We consider the passenger at the location L_4 as an example to explain the process. We assume that the passenger is ready to catch a vacant taxi at 10:30 am on January 10, 2013. According to the section II-D, the distance threshold is 750 meters and the time threshold is 14 minutes. From the Fig. 10, we can see there are four options for passengers: firstly going to road segment r_{680}, r_{682}, r_{718} or r_{726}.

1) WHEN WAITING ROUTE STARTS BY TAKING ROAD SEGMENT R_{718}

The best recommendation given by our algorithm is in Fig. 11, which starts with road segment r_{718}. In this waiting route, the passengers firstly spend 1 minute arriving at road segment r_{718} and wait for 4 minutes; if they cannot catch a taxi, they should continue to walk to the crossroad c_{293} and wait for 3 minutes; if they still do not catch a taxi, then we suggest they continue to walk to the crossroad c_{304} and wait for 3 minutes, and they stop because of reaching the distance threshold. According to this plan, the probability that



(a) Waiting route starts at r_{718}. (b) Waiting route starts at r_{680}. (c) Waiting route starts at r_{682}. (d) Waiting route starts at r_{726}.

FIGURE 11. Experimental results.

passengers can catch a taxi is 72.16%. Next, we conducted a detailed analysis of the other three options.

2) WHEN WAITING ROUTE STARTS BY TAKING ROAD SEGMENT R_{680}

The waiting route obtained by our algorithm based on historical data is as in Fig. 11(b). The historical data show that the length of this road segment is greater than the threshold of the distance we set, so it is impossible for passengers to reach any crossroad. The total return value of the scheme is the taxi pass rate of r_{680}, i.e. 6.67%.

3) WHEN WAITING ROUTE STARTS BY TAKING ROAD SEGMENT R_{682}

The waiting route obtained by our algorithm based on historical data is as in Fig. 11(c). In this waiting route, passengers need to spend 2 minutes arriving at road segment r_{682} and wait for 7 minutes. If they don't catch any taxi, then they should continue to wait at crossroad c_{293} and wait for 3 minutes, and they stop because of reaching the distance threshold. According to this plan, the probability that passengers can catch a taxi is 64.41%.

4) WHEN WAITING ROUTE STARTS BY TAKING ROAD SEGMENT R_{726}

The waiting route obtained by our algorithm based on historical data is as in Fig. 11(d). The vacant taxi pass rate of this road segment is relatively small according to historical data, so the waiting time is very large, which has exceeded the time threshold we set, and it is impossible for passengers to reach any crossroad either. Therefore, the total return value of the scheme is the taxi pass rate of r_{726}, i.e. 8.18%.

From the above analysis, our algorithm can recommend a waiting route with the highest probability of catching a vacant taxi.

Furthermore, we also query the data for 10-11 o'clock on January 10, 2013 and calculate the vacant taxi pass rate on each road segment and crossroad, basing on which we get the best routes for the 50 passengers to wait for vacant taxis. Under the same constraints of distance threshold and time threshold, our recommended waiting route are almost similar with the best routes in real, which shows our recommendations are in line with the actual situation.

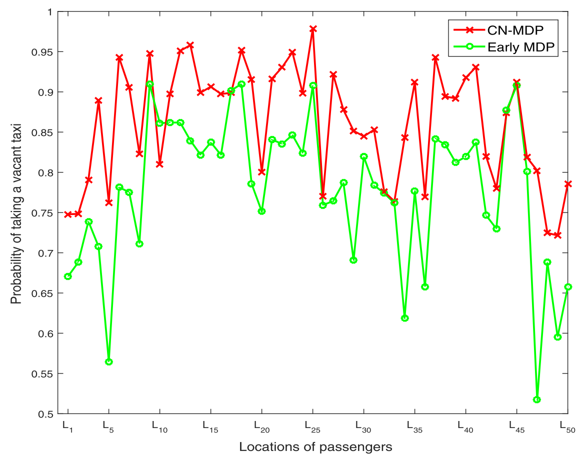


FIGURE 12. Comparing with Early-MDP approach.

C. IMPROVEMENT OF CN-MDP OVER BASELINE METHODS

We first compare our approach with the one we have previously proposed, called *Early-MDP*. In order to unify them, we use the probability of catching a vacant taxi as a comparison criterion. Because we have randomly generated the location of 50 passengers and the corresponding requesting time, we can use these two methods to get the corresponding recommendation strategy under the same conditions. Finally, we calculate the probability of catching the vacant taxis with each recommended strategy, and then compare them.

As shown in Fig. 12, we can see the probability of our algorithm proposed here is much higher than the one we have previously proposed, which shows that our judgment is correct, and the number of vacant taxis passing by the crossroads is greater than that by taking the road segments.

We also compare our algorithm with other algorithms. We mainly compare it with the two most used methods: (i) *Time-based*, which recommends the road segment with shortest waiting time; (ii) *Distance-based*, which recommends the road segment nearest to the passenger. We also use the probability of waiting for a vacant taxi as a comparison

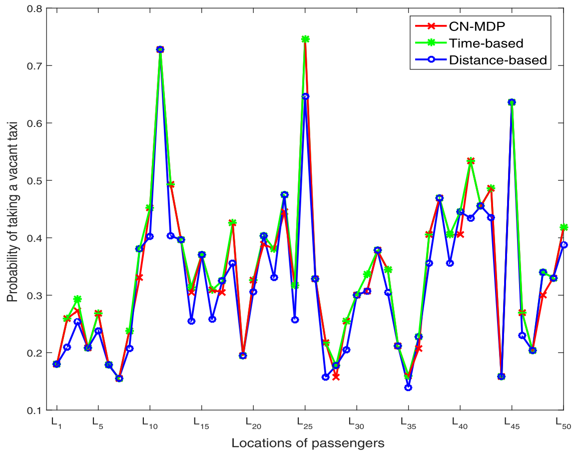


FIGURE 13. **The probability of catching vacant taxis for one-location recommendation.**

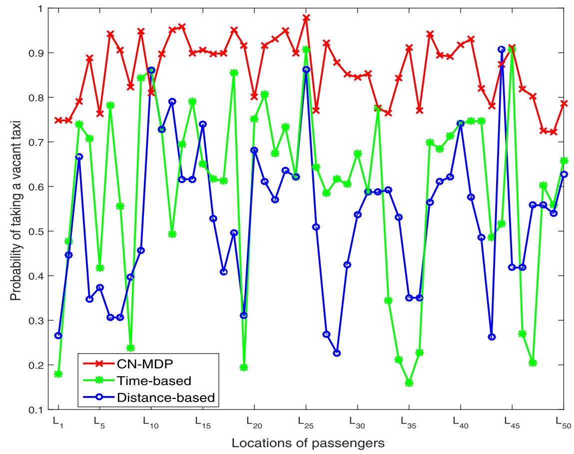


FIGURE 14. **The probability of catching vacant taxis for recommending location sequence.**

criterion just like before. Because our approach recommends a sequence instead of just one waiting place, we compare them from two aspects below.

1) RECOMMEND ONE WAITING LOCATION

Because our approach is divided into two steps, we can use our first step to compare with *Time-based* and *Distance-based*, i.e. one-location recommendation. Then we calculate the probability of catching the vacant taxis with each recommendation and compare them.

As shown in Fig. 13, we can see the probability of our algorithm proposed here is almost the same as other two algorithms. This shows that our first step has already achieved their results.

2) RECOMMEND A WAITING SEQUENCE

Because our total recommendation is a waiting sequence, we can also recommend a waiting sequence using *Time-based* and *Distance-based* respectively under the same conditions, and we calculate the probability of catching vacant taxis with each recommendation, and then compare them.

As shown in Fig. 14, we can see the probability of our algorithm proposed here is much higher than other two algorithms, which shows that passengers can have higher probability to catch a vacant taxi by taking our recommendation than the other two. And our previously proposed approach is also better than the other two by comparing Fig. 12 and Fig. 14.

From the above analysis, our algorithm is indeed better than other algorithms. In fact, our method includes both of the two algorithms, and expands on this basis to achieve better results. Passengers have more chances to catch a vacant taxi according to our recommended route.

D. MULTI-PASSENGERS COMPETITION

However, there are usually multi-passengers waiting for vacant taxis together at the same location and the same time slot, which will cause the multi-passengers to compete

FIGURE 15. **Recommending waiting routes in competitive situation.**

for taxis. How can our scheme adapt to this competition situation? Assuming that three passengers at location $L4$ make requests in the same time slot, we gave three different recommendations according to our algorithm, as shown in the Fig. 15. We recommend the first passenger to wait for vacant taxis by taking route A, the probability that he can catch a taxi is 72.16%. For the second passenger, we suggest him taking the route B, the calculative probability of catching vacant taxis is 70.54%, which is a little smaller than route A. And for the third passenger, the waiting route is route C which is the same as the first recommendation route, but the probability is a little smaller, which is 68.39%, because the first passenger may have caught one vacant taxi before.

We also apply the other three methods to the same multi-passengers competitive situation under the same thresholds in the similar way, and calculate the probability of each passenger who can catch a taxi corresponding to different recommended schemes, as shown in table 3.

From table 3, we can see the probabilities of the above three passengers calculated by our approach are all greater than that of *Early-MDP* approach and the other two baseline ones.

TABLE 3. Recommendation under competitive situation.

Approach	The first passenger	The second passenger	The third passenger
CN-MDP	72.16%	70.54%	68.39%
Early-MDP	60%	59.54%	59%
Time-based	55.81%	54.7%	53.57%
Distance-based	55.81%	54.7%	53.57%

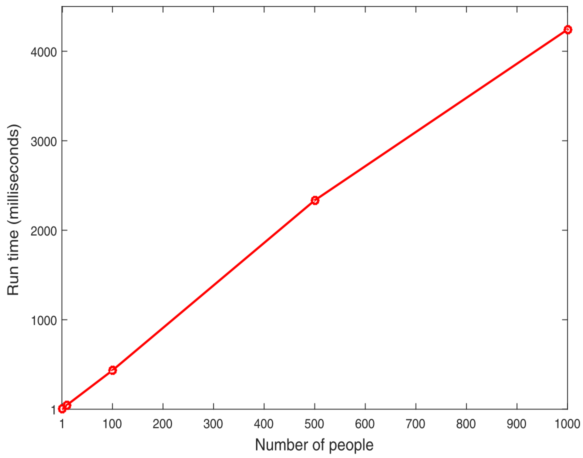


FIGURE 16. Computation time with varying people number.

So, our *CN-MDP* approach performs best under competitive situations of multi-passengers.

E. COMPUTATIONAL EFFICIENCY

Finally we evaluate the computational efficiency of the proposed method. Because our strategy is not only for one or two people, and we also consider the situation of competition, so we must guarantee that our algorithm can give recommendation results in a short time when many people use it.

Fig. 16 shows that the computation time of our algorithm increases linearly with the increase in the number of users, which indicates that our algorithm is very efficient and meets big data requirements. In addition, under the random waiting taxi mode in reality, the number of passengers in a certain area is not much, so our algorithm is reliable in real-time.

VI. CONCLUSION

In this paper, we propose a *CN-MDP* approach to recommend where to wait for taxis by solving two value functions, and then recommend a waiting location sequence for current waiting passengers until the cumulative probability of catching a vacant taxi getting close to 100%. For multi-passengers competition, we update dynamically the pass rate of vacant taxis at each crossroad and road segment, and recommend different waiting routes for different passengers. Some evaluations on a real taxi dataset from a major city in China show that our recommendation scheme works well and has a higher probability of catching vacant taxis than that of our previous approach and other ones, and improves the user experience of taxi services. In a word, our *CN-MDP* scheme is an effective and efficient strategy.

REFERENCES

[1] J. Yuan, Y. Zheng, L. Zhang, X. Xie, and G. Sun, "Where to find my next passenger," in *Proc. 13th Int. Conf. Ubiquitous Comput.*, 2011, pp. 109–118.

[2] Z. Qiu, S. Zhang, W. Zhou, and S. Yu, "Empirical study on taxi's mobility nature in dense urban area," in *Proc. IEEE Conf. Comput. Commun. Workshops*, Apr. 2018, pp. 232–237.

[3] Y. Ge, C. Liu, H. Xiong, and J. Chen, "A taxi business intelligence system," in *Proc. 17th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining (KDD)*, Aug. 2011, pp. 735–738.

[4] M. Veloso, S. Phithakkitnukoon, and C. Bento, "Urban mobility study using taxi traces," in *Proc. Int. Workshop Trajectory Data Mining Anal. (TDMA)*, Sep. 2011, pp. 23–30.

[5] T. Xu, H. Zhu, X. Zhao, Q. Liu, H. Zhong, E. Chen, and H. Xiong, "Taxi driving behavior analysis in latent vehicle-to-vehicle networks: A social influence perspective," in *Proc. 22nd ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining (KDD)*, 2016, pp. 1285–1294.

[6] H. Rong, X. Zhang, Q. Liu, Q. Yang, and J. Gu, "A Mokov decision process approach to optimizing waiting for taxis," in *Proc. IEEE SmartWorld, Ubiquitous Intell. Comput., Adv. Trusted Comput., Scalable Comput. Commun., Cloud Big Data Comput., Internet People Smart City Innov. (SmartWorld/SCALCOM/UIC/ATC/CBDCom/IOP/SCI)*, Oct. 2018, pp. 1346–1351.

[7] M. Qu, H. Zhu, J. Liu, G. Liu, and H. Xiong, "A cost-effective recommender system for taxi drivers," in *Proc. 20th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining (KDD)*, 2014, pp. 45–54.

[8] H. Rong, X. Zhou, C. Yang, Z. Shafiq, and A. Liu, "The rich and the poor: A Markov decision process approach to optimizing taxi driver revenue efficiency," in *Proc. 25th ACM Int. Conf. Inf. Knowl. Manage. (CIKM)*, Oct. 2016, pp. 2329–2334.

[9] P. S. Castro, D. Zhang, and S. Li, "Urban traffic modelling and prediction using large scale taxi GPS traces," in *Proc. Int. Conf. Pervasive Comput.* Berlin, Germany: Springer, Heidelberg, 2012.

[10] F. Miao, S. Han, S. Lin, J. A. Stankovic, D. Zhang, S. Munir, H. Huang, T. He, and G. J. Pappas, "Taxi dispatch with real-time sensing data in metropolitan areas: A receding horizon control approach," *IEEE Trans. Autom. Sci. Eng.*, vol. 13, no. 2, pp. 463–478, Apr. 2016.

[11] G. Pan, G. Qi, Z. Wu, D. Zhang, and S. Li, "Land-use classification using taxi GPS traces," *IEEE Trans. Intell. Transp. Syst.*, vol. 14, no. 1, pp. 113–123, Mar. 2013.

[12] P. S. Castro, D. Zhang, C. Chen, S. Li, and G. Pan, "From taxi GPS traces to social and community dynamics: A survey," *CSURACM Comput. Surv.*, vol. 46, no. 2, pp. 1–34, Nov. 2013.

[13] G. Qi, X. Li, S. Li, G. Pan, Z. Wang, and D. Zhang, "Measuring social functions of city regions from large-scale taxi behaviors," in *Proc. IEEE Int. Conf. Pervas. Comput. Commun. Workshops (PERCOM)*, Mar. 2011, pp. 384–388.

[14] B. Liu, M. Jin, and P. Zeng, "Prioritization of candidate disease genes by combining topological similarity and semantic similarity," *J. Biomed. Informat.*, vol. 57, pp. 1–5, Oct. 2015.

[15] W. Zhang, S. Li, and G. Pan, "Mining the semantics of origin-destination flows using taxi traces," in *Proc. ACM Conf. Ubiquitous Comput.*, Sep. 2012, pp. 943–949.

[16] J. Zheng, S. Liu, and L. M. Ni, "Robust Bayesian inverse reinforcement learning with sparse behavior noise," in *Proc. 28th AAAI Conf. Artif. Intell.*, Jul. 2014, pp. 2198–2205.

[17] S. Liu, Y. Liu, L. M. Ni, J. Fan, and M. Li, "Towards mobility-based clustering," in *Proc. 16th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining (KDD)*, Jul. 2010, pp. 919–928.

[18] S. Liu, Y. Yue, and R. Krishnan, "Adaptive collective routing using Gaussian process dynamic congestion models," in *Proc. 19th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining (KDD)*, Aug. 2013, pp. 704–712.

[19] S. Liu, S. Wang, C. Liu, and R. Krishnan, "Understanding taxi drivers–routing choices from spatial and social traces," *Front. Comput. Sci.*, vol. 9, no. 2, pp. 200–209, Apr. 2015.

[20] K. Xie, X. Ning, X. Wang, D. Xie, J. Cao, G. Xie, and J. Wen, "Recover corrupted data in sensor networks: A matrix completion solution," *IEEE Trans. Mobile Comput.*, vol. 16, no. 5, pp. 1434–1448, May 2017.

[21] A. L. Alfeo, M. G. C. A. Cimino, S. Egidi, B. Lepri, and G. Vaglini, "A stigmergy-based analysis of city hotspots to discover trends and anomalies in urban transportation usage," *IEEE Trans. Intell. Transp. Syst.*, vol. 19, no. 7, pp. 2258–2267, Jul. 2018.

[22] K. Abdelghany, H. Hashemi, and M. E. Khodayar, "A decision support system for proactive-robust traffic network management," *IEEE Trans. Intell. Transp. Syst.*, vol. 20, no. 1, pp. 297–312, Jan. 2019.

[23] N. J. Yuan, Y. Zheng, L. Zhang, and X. Xie, "T-finder: A recommender system for finding passengers and vacant taxis," *IEEE Trans. Knowl. Data Eng.*, vol. 25, no. 10, pp. 2390–2403, Oct. 2013.

[24] X. Zheng, X. Liang, and K. Xu, "Where to wait for a taxi?" in *Proc. ACM SIGKDD Int. Workshop Urban Comput.*, 2012, pp. 149–156.

[25] W. Jing, L. Hu, L. Shu, M. Mukherjee, and T. Hara, "RPR: Recommendation for passengers by roads based on cloud computing and taxis traces data," *Pers. Ubiquitous Comput.*, vol. 20, no. 3, pp. 337–347, Jun. 2016.

[26] G. Qi, G. Pan, S. Li, Z. Wu, D. Zhang, L. Sun, and L. T. Yang, "How long a passenger waits for a vacant taxi—Large-scale taxi trace mining for smart cities," in *Proc. IEEE Int. Conf. Green Computing Commun. IEEE Internet Things IEEE Cyber, Phys. Social Comput.*, Aug. 2013, pp. 1029–1036.

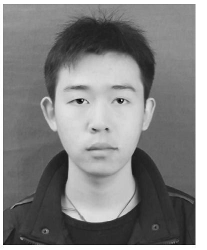
[27] S. Ross, *Simulation*. New York, NY, USA: Academic, 2006.

[28] (2018).*Baidu Map*. [Online]. Available: http://map.baidu.com/

[29] O. Pink and B. Hummel, "A statistical approach to map matching using road network geometry, topology and vehicular motion constraints," in *Proc. 11th Int. IEEE Conf. Intell. Transp. Syst.*, Oct. 2008, pp. 862–867.

[30] S. Ross, *A First Course in Probability*, London, U.K.: Pearson, 2002.

HUIGUI RONG (Member, IEEE) received the Ph.D. degree in information management from Wuhan University, China, in 2010. He was a System Analyst and a Senior Project Manager certified by the Ministry of Industry and Information Technology (MIIT). He was a Visiting Scholar with the Computer Science Department, Michigan State University (MSU), from 2014 to 2015. He is currently a Full Associate Professor of computer science and technology with Hunan University. He has published more than 20 research articles in international conferences and journals, such as the IEEE TBD, ACM CIKM, and the IEEE HPCC. His research interests include data mining and machine learning. He is also a member of CCF.



XUDONG ZHANG is currently a Graduate Student with the College of Computer Science and Electronic Engineering, Hunan University. His main research interests include machine learning, spatiotemporal data mining, and big data analysis. He is also a CCF Member. He received the first-class scholarships in stage of postgraduate.

ZHUO LI received the Ph.D. and M.D. degrees in ophthalmology from Central South University, Changsha, China, in 2007. She is currently the Chief of Clinical Trail Management Office and the Vice Professor with The Second Xiangya Hospital of Central South University, Changsha. She is also the Second Charger of the Program of Chinese National Good Clinical Practice (GCP) Platform of New Medicine. She has published four research articles in international journals and international conferences. Her research interests include AI in medical management data computing and ophthalmology.

ZHAOYANG AI (Member, IEEE) received the Ph.D. degree in linguistics from Hunan University, Changsha, China, in 2016. He is currently working as an Associate Professor with the Institute of Cognitive Control and Biophysics Linguistics, College of Foreign Languages, Hunan University. His researches are focused on cognitive control and lingua-biophysics, and include intelligent translation and interpretation, linguistics, cognitive superconductivity, informatics, smart control, and efficiency studies. He has received the Provincial Social Science Grants, the Provincial Education Science Grant, the Provincial Science Grant, Electrical Engineering, and the State Grid Science Grant. He is also participating in the National Science Fund Project. He is also a member of the IEEE Council of Superconductivity, the IEEE Communications Society, and the IEEE Information Theory Society.