A Cost-Effective Recommender System for Taxi Drivers

Meng Qu
Rutgers Business School
daisymengqu@hotmail.com

Hengshu Zhu
University of Science and Technology of China
zhuhengshu@gmail.com

Junming Liu
Rutgers, the State University of New Jersey
jl1433@scarletmail.rutgers.edu

Guannan Liu
Tsinghua University
guannliu@gmail.com

Hui Xiong[*]
Rutgers Business School
Rutgers University
hxiong@rutgers.edu

ABSTRACT

The GPS technology and new forms of urban geography have changed the paradigm for mobile services. As such, the abundant availability of GPS traces has enabled new ways of doing taxi business. Indeed, recent efforts have been made on developing mobile recommender systems for taxi drivers using Taxi GPS traces. These systems can recommend a sequence of pick-up points for the purpose of maximizing the probability of identifying a customer with the shortest driving distance. However, in the real world, the income of taxi drivers is strongly correlated with the effective driving hours. In other words, it is more critical for taxi drivers to know the actual driving routes to minimize the driving time before finding a customer. To this end, in this paper, we propose to develop a cost-effective recommender system for taxi drivers. The design goal is to maximize their profits when following the recommended routes for finding passengers. Specifically, we first design a net profit objective function for evaluating the potential profits of the driving routes. Then, we develop a graph representation of road networks by mining the historical taxi GPS traces and provide a Brute-Force strategy to generate optimal driving route for recommendation. However, a critical challenge along this line is the high computational cost of the graph based approach. Therefore, we develop a novel recursion strategy based on the special form of the net profit function for searching optimal candidate routes efficiently. Particularly, instead of recommending a sequence of pick-up points and letting the driver decide how to get to those points, our recommender system is capable of providing an entire driving route, and the drivers are able to find a customer for the largest potential profit by following the recommendations. This makes our recommender system more practical and profitable than other existing recommender systems. Finally, we carry out extensive experiments on a real-world data set collected from the San Francisco Bay area and the experimental results clearly validate the effectiveness of the proposed recommender system.

[*]Contact author.

KDD'14, August 24–27, 2014, New York, NY, USA.
Copyright 2014 ACM 978-1-4503-2956-9/14/08:.$15.00.
http://dx.doi.org/10.1145/2623330.2623668.

Categories and Subject Descriptors

H.2.8.d [**Information Technology and Systems**]: Database Applications - Data Mining

General Terms

Algorithms, Experimentation

Keywords

Cost-Effective, Mobile Recommender Systems, Taxi Drivers

1. INTRODUCTION

Recent years have witnessed the rapid development of wireless sensor technologies in mobile environments, such as GPS, Wi-Fi and RFID. The advances of such technologies indicate the possibility to change radically the existing methods of doing taxi business. Indeed, recent efforts have been made on providing personalized mobile services to taxi drivers through the analysis of Taxi GPS traces. In general, there are three existing ways to provide such services. The first way is to focus on the development of the fastest driving route [25, 26, 28, 27, 14], which shows the fastest driving route from the current location to the destination. The second way is to provide a sequence of pick-up points for taxi drivers. The goal is to allow the taxi driver to find a customer within the shortest driving distance [8]. Finally, an alternative service is to strike a balance between the needs of taxi drivers and passengers [24].

Indeed, most of the existing mobile recommender systems for taxi business are focused on extracting energy-efficient transportation patterns from historical location traces and recommending a sequence of potential pick-up points for taxi drivers [25, 26, 8]. However, in the real world, the income of taxi drivers is strongly correlated with the effective driving hours which may not necessarily lead to energy-efficiency. In other words, it is more critical for taxi drivers to know the actual driving routes to minimize the driving time before finding a customer. Taxi drivers usually rent their cabs from taxi companies for a fixed time period. There is a fixed per-hour cost associated with gas usage and the rental fee. The profit of a taxi driver really depends on how much money the driver can make per hour; that is, how effectively the drivers can make use of their driving time.

To that end, in this paper, we propose to develop a cost-effective recommender system for taxi drivers. The design goal is to maximize their profits when following the recommended routes for finding passages. In particular, the proposed system can provide an entire driving route rather than just recommending a sequence of discrete pick-up points and letting the driver decide how to get to those points, and the drivers are able to find a customer with the largest potential profit by following the recommended route. This makes our recommender system more practical and profitable than other existing mobile recommender systems [8].

To achieve the design goal and recommend an entire driving route which allows the taxi drivers to maximize their profits by following the recommended route, there are several factors to be considered. First, it is necessary to know the pick-up probabilities along the route. Second, it should be able to compute the profit that drivers can make after picking up a customer somewhere on the route. Third, the potential driving time on the route should be estimated. Indeed, all these issues can be solved by mining the historical Taxi GPS traces. However, a key challenge is how to combine the impact of all these factors. Indeed, in this paper, we develop a net profit objective function to collectively integrate the impact of the above factors. The net profit objective function can be used for evaluating the potential profit of the driving routes. Then, we develop a graph representation of road networks and provide a Brute-Force strategy to generate optimal driving route for finding passengers. In addition, the search for candidate driving routes is essential a combinatorial search problem. The computational cost is prohibited. Therefore, we further develop a pruning strategy to reduce the search space and improve the computational performances. In particular, we first change the graph representation of road networks to a new structure, namely a recursion tree, based on the special form of the net profit function. Then, we design a novel recursion strategy based on the recursion tree for searching optimal candidate routes in an efficient way.

When recommending the driving routes to the taxi drivers, we also provide a strategy for making a better load balance for the recommendations happening at the same location. Specifically, we exploit a minimum redundant strategy. For each target location, we transform each candidate route in the recommended list associated with this location into a direction vector. Then, we are able to calculate the correlations among this candidate route in terms of their directions. If there are several requests happening at the same location within a short time period, this minimum redundant strategy can provide recommendations in a load balanced way.

Finally, we carry out extensive experiments on a real-world data set collected from the San Francisco Bay area and the experimental results clearly validate both the effectiveness and efficiency of the proposed recommender system.

Overview. The remainder of this paper is organized as follows. Section 2 shows the related works of this paper. In Section 3, we formulate the problem of cost-effective recommendations for taxi drivers and introduce some preliminaries. Section 4 provides a detailed description of our recommender system. In Section 5, we report the experimental results. Finally, Section 6 concludes this work.

2. RELATED WORK

In the literature, many efforts have been devoted to building personalized recommender systems, such as content-based recommendation [13], collaborative filtering based recommendation [17] as well as the hybrid recommendation [15]. Furthermore, some recommender systems [2] also aim to address the information overloaded problem by identifying user interests and providing personalized suggestions. However, those traditional recommender systems [4, 7, 11] are more focused on recommendation of online information, such as online movie, article, book or webpage. In most of the cases, the research data are based on user ratings, which are very different from the data collected in mobile environment.

The development of personalized recommender systems in mobile and pervasive environments is much more challenging than developing recommender systems from traditional domains due to the complexity of spatial data and intrinsic spatio-temporal relationships, the unclear roles of context-aware information [30], and the increasing availability of environment sensing capabilities. Those unique challenges are actually inherit in the mobile data we have. Indeed, recommender systems in the mobile environments have been studied before [1, 3, 5, 6, 12, 19, 20]. For instance, the works in [1, 5] target at the development of mobile tourist guides. Zhu *et al.* proposed a uniform framework of personalized context-aware recommendation for mobile users. The framework can discover users' personal context-aware preferences by mining the context logs of many mobile users. Heijden *et al.* [30] have discussed some technological opportunities associated with mobile recommendation systems [20]. Averjanova *et al.* have developed a map-based mobile recommender system that can provide users with some personalized recommendations [3]. However, the above prior works are mostly based on user ratings or interactions, and corresponding recommender systems are developed for smart mobile devices, such as mobile phones. Indeed, the problem of building mobile recommender systems for taxi business remains pretty much open.

Recently, the abundant availability of Taxi GPS traces has enabled new ways of doing taxi business. Plenty efforts have been made on developing mobile recommender systems for taxi drivers by using Taxi GPS traces. These systems can extract energy-efficient transportation patterns from historical location traces and recommending potential pick-up points for taxi drivers. For example, Ge *et al.* [8] defined a novel problem of mobile sequential recommendation by leveraging the historical GPS data from taxi drivers. By solving this problem, a novel energy-efficient mobile recommender system has been developed. This system can provide a optimal sequence of pick-up points for taxi drivers. Also, Powell *et al.* [16] proposed a grid-based method to suggest the profit locations for taxi drivers by constructing a spatio-temporal profitability map. In addition, Yuan *et al.* [24, 25, 26] have carried out a series of studies on mobile intelligence by leveraging taxi trajectories, such as pick-up points detection based on probabilistic models, and location recommendation for both the taxi drivers and customers. Different from the above studies, in this paper, we propose to develop a novel recommender system that is capable of providing an entire driving route instead of discrete pick-up points, and the drivers are able to find a customer for the largest potential profit by following the recommendations.

3. PROBLEM FORMULATION

In this section, we first introduce some preliminaries, and

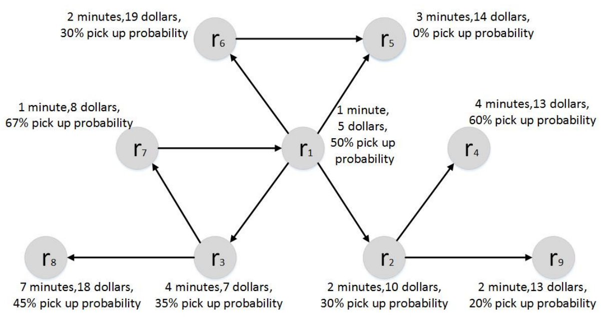


Figure 1: An example of a route segment network.

then formally define the problem of Maximum Net Profit (MNP) recommendation for taxi drivers.

3.1 Preliminaries

Here, we first introduce some basic concepts used throughout this paper.

3.1.1 Road Network Formulation

Definition 1 (Road Segment). *A long street can be separated into several road segments r by its crossroads. Specifically, each segment r is associated with a start point $r.s$ and an end point $r.e$. Moreover, each segment r also has several adjacent segments forming a set $r.next[]$, which satisfies $\forall r_i \in r.next[]$ iff. $r.e = r_i.s$.*

Definition 2 (Route). *A route R is a sequence of connected road segments, i.e., $R = (r_1 \to r_2 \to \cdots \to r_M)$, where $r_{k+1}.s = r_k.e$ $(1 \leq k < M)$. The start point and the end point of a route R can be represented as $R.s = r_1.s$ and $R.e = r_n.e$.*

Definition 3 (Road Segment Network). *The road segment network G can be represented by a graph $G =< V, E >$, where $V = \{r_i\}$ is the node set that consists of all road segments and E is the edge set, which satisfies $\exists e_{ij} \in E$ iff. $r_j \in r_i.next[]$.*

Figure 1 demonstrates an example of the road segment network. In this graph, each node represents for a road segment. Note that, each edge only has one direction. This is because we do not allow taxi drivers to drive back and forth in the same single road segment, which is not recommended in real life and has a high potential to result in traffic jam and accidents. However, taxi drivers can take a loop through three road segments, such as nodes r_1, r_3 and r_7, which can form a loop for drivers.

3.1.2 Calculation of Net Profit

For each segment r, the net profit $g(r)$ consists of two components, namely *potential earning* and *potential cost.* Specifically, we define the potential earnings of segment r as $e(r)$, which can be computed by

$$e(r) = \frac{\sum_{i=1}^{N_r} Fee(i;r)}{N_r} P(r), \tag{1}$$

where N_r is the number of picking-up passengers in segment r during a given time period, $Fee(i;r)$ is the earning from the i-th pick-up passenger and $P(r)$ is the pick-up possibility in segment r, which will be introduced in Section 4. Meanwhile, the potential cost of segment r, i.e., $c(r)$, can be computed by

$$c(r) = (1 - P(r))(L(r) \cdot Gas + T(r) \cdot CompanyFee), \tag{2}$$

where $L(r)$ is the length of segment r, Gas is the price of Gas per unite distance (e.g., per mile), $T(r)$ is the traveling time through segment r and $CompanyFee$ is the opportunity cost per unit time (e.g., per minute). Indeed, $T(r)$ is sensitive to the real-time traffic conditions. For example, a traffic jam will result in a high $T(r)$, and thus bring a high cost of $T(r) \cdot CompanyFee$. In this case, the segment will not recommended by our model. Therefore, the net profit of segment r, i.e., $g(r)$, can be computed by

$$g(r) = e(r) - c(r). \tag{3}$$

Based on the above, we can further define the net profit for each route R. Specifically, given a route $R = (r_1 \to r_2 \to \cdots \to r_M)$ starting from r_1, its total net profit can be computed by

$$G(R, r_1, M) = g(r_1) + \sum_{i=2}^{M} g(r_i) \prod_{j=1}^{i-1} \big(1 - P(r_j)\big). \tag{4}$$

Intuitively, the net profit of route R is the sum of the net profit of road segments $\{r_i\}$ contained in R, which is weighted by the possibility of not picking up any passenger in previous segments (i.e., r_1 to r_{i-1}).

Indeed, with the possibility weights in net profit, the taxi driver will not consider the segments which are far away from her current location because the expected profit there is very low. To be more specific, we can define the average increasing rate of net profit as $\tau = \frac{<G(R,r_i,M+1)> - <G(R,r_i,M)>}{<G(R,r_i,M+1)>}$ to indicate the profit increase when increasing one more road segment in the route. Figure 2 shows the trend of the increasing rate with respect to different numbers of increased road segments and different pick-up possibilities. We can observe that the increasing rate is less than 10% after increasing more than 5 road segments. Indeed, the average pick-up probability of each road segment in our experiments is always less than 0.1, therefore it is possible for us to set an upper bound Λ for route length M in Equation 4. Based on the above definitions, we can formally define the MNP recommendation problem as follows.

Definition 4 (Problem Statement). *Given the current location $LCab \in r$ of a taxi driver, a fixed cruising length M, and a set of route candidates $\mathbb{R}$, where $\forall R \in \mathbb{R}$ satisfies R starts from r. The MNP recommendation problem is to recommend a route $R^* \in \mathbb{R}$, which has the maximum net profit, i.e.,*

$$R^* = \arg\max_{R \in \mathbb{R}} \big\{G(R, r, M)\big\}. \tag{5}$$

Different from other existing recommender systems for taxi drivers, which mainly focus on extracting energy-efficient transportation patterns based on traveling time/length and recommending a sequence of potential pick-up points for taxi drivers [25, 26, 8], the MNP recommendation problem focuses on providing an entire driving route with maximum net profits for a taxi driver. Along this line, there are two major challenges for solving the MNP recommendation problem. First, how to calculate the parameters $g(r)$, $P(r)$ of

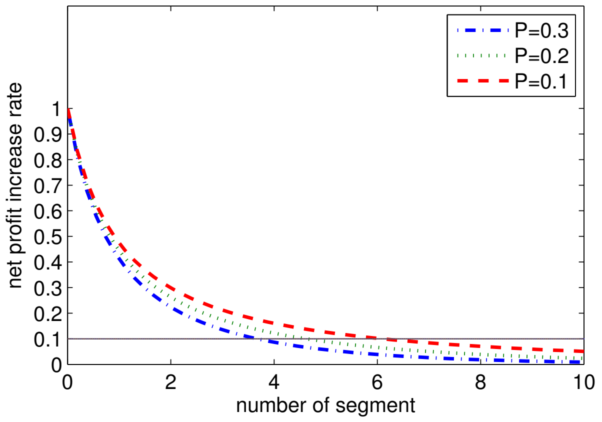


Figure 2: The average increase rate of net profit with respect to different number of increased road segment and the different fixed pick up possibility (i.e., $P(r) = 0.1$, $P(r) = 0.2$, $P(r) = 0.3$).

each segment r from the historical pick-up data. Second, how to efficiently search an optimal route from the complex directed-cyclic route segmentation network. In the following section, we will introduce our solutions for the above two challenges, respectively.

4. MAXIMUM NET PROFIT (MNP) RECOMMENDATION

In this section, we introduce the technical details of our solutions for the MNP recommendation problem.

4.1 Parameter Estimation with Road Buffer

To accurately obtain the taxi driver's current location and the parameters for estimating net profit, i.e., $P(r)$ and $g(r)$, we exploit the *road buffer* estimation for each road segment. Specifically, in geographic information systems, a buffer is a zone of specified distance around the spatial object. The boundary of the buffer is the solid line of equal distance to the edge of the object. Figure 3(a) is an illustration of different buffer operations, such as buffers on a point, three line segments and a polygon [21]. Intuitively, people would like to wait for taxis at road side instead of in the middle of road, and the pick-up points of taxis are always around the road side. Therefore, when calculating for the number of historical pick-up events, we need to build a buffer around each road segment for obtaining the new boundaries of the road. This road buffer usually looks like a rectangle surrounding the road, which is similar with the buffer operation of three line segments. Particularly, the size of the buffer depends on the demands of different real-world problems.

To build the road buffer, we need to define the vertical road and the horizontal road first. To be specific, by using the longitude and the latitude of the starting and the ending points of each road segment, we can calculate the tangent value of this road segment. If the absolute value of the tangent is greater than 1, then we regard the corresponding road as vertical road, otherwise it is a horizontal road. For each vertical road, we keep the longitudes of its starting and ending points and extend the corresponding latitudes to west and east. For each horizontal road, we keep the latitudes of its starting and ending points and extend the corresponding longitudes to north and south. The above buffer operation results in new boundaries formed by four vertex coordinates. For example, Figure 3(b) shows the buffer operations on vertical road and horizontal road.

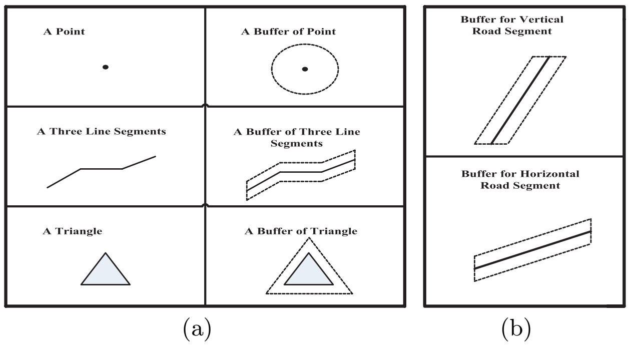

A Point
A Buffer of Point
A Three Line Segments
A Buffer of Three Line Segments
A Triangle
A Buffer of Triangle
Buffer for Vertical Road Segment
Buffer for Horizontal Road Segment
(a)
(b)

Figure 3: (a) Different buffer operations; (b) Buffer operations on vertical and horizontal roads.

Given the historical pick-up data and the road buffers, we are able to calculate the total number of pick-up events in each road segment r, which indicates how frequently a pick-up event can happen when cabs travel across each road segment. Let N_r^0 denote the number of times that taxis are vacant in the buffer of road segment r, and N_r^* denote the number of times that taxis had pick-up events in the buffer of segment r. Thus, the probability of pick-up event for each road segment r, i.e., $P(r)$, can be estimated as

$$P(r) = \frac{N_r^*}{N_r^0 + N_r^*}. \tag{6}$$

From the i-th historical pick-up event in segment r, we can also obtain the earnings $Fee(i; r)$ in Equation 1. Furthermore, the road length $L(r)$ and real-time traveling time $T(r)$ can be estimated from the historical data or some external resources, such as Google Map. Therefore, the net profit of $g(r)$ can be calculated by Equation 3. Particularly, the value $T(r)$, $g(r)$ and $P(r)$ of each road segment r can be pre-stored in corresponding node of the road segment network (e.g., Figure 1).

4.2 MNP Route Recommendation

In this subsection, we introduce how to solve the problem of NMP recommendation by different strategies.

4.2.1 *Brute-Force Recommendation Strategy*

After obtaining the road segment network, we can leverage it for generating route candidates can MNP recommendation. To this end, we first propose a Brute-Force strategy for this task based on the Breadth-First search. Specifically, the recommendation algorithm is shown in Algorithm 1. In this algorithm, we keep a route queue Q for generating a set of route candidates C, and the function $MNP(C)$ in Step 5 is used for finding the optimal route with maximum net profit in candidate set C. However, such Brute-Force method for searching the MNP route is not efficient, since it has to check all possible routes with length M in G.

LEMMA 1. *Given a fixed cruising length M and the road segment network $G = \{V, E\}$, where $|V| = N$, the compu-*

Algorithm 1 Brute-Force based MNP Recommendation

```
Input 1: road segment network $G = \{V, E\}$;
Input 2: the cruising length $M$;
Input 3: taxi driver's current segment $r_1$;
Output: the MNP route $R^*$;
Initialization: A route queue $Q = \{R_0\}$, where $R_0 = \{r_1\}$;
 1: $C = \emptyset$;
 2: //get route from queue $Q$;
 3: $R = Q.del()$;
 4: if ($R = \emptyset$) do
 5:     return $R^* = MNP(C)$;
 6: else if ($|R| == M$) do
 7:     $C = C \cup R$;
 8: else if ($|R| < M$) do
 9:     //$r_k$ is the last road segment in $R$;
10:     for each ($r_i \in V$, $\exists e_{ki} \in E$) do
11:         //add route from queue $Q$;
12:         $Q.add(R \cup \{r_i\})$;
13:     go to Step 3;
```

tational complexity of searching an optimal MNP route by Brute-Force algorithm is $\mathcal{O}(MN^{M-1})$

Proof. Obviously, the total number of route candidates in road segment network G is $\leq N^{M-1}$, and computing the net profit for each route needs M operations. Thus, the complexity of searching optimal MNP route is $\mathcal{O}(MN^{M-1})$ □

Intuitively, the computational complexity of the Brute-Force algorithm is too high to satisfy the needs of real-world applications. There are some algorithm can save the reaching time of freeway travel in real world. To this end, we further propose another recommendation strategy based on the recursive characteristic of the net profit function.

4.2.2 Recursive Recommendation Strategy

By observing the form of the net profit of routes, we can re-write the Equation 4 as follows.

$$G(R, r_1, M) = g(r_1) + \big(1 - P(r_1)\big)G(R - r_1, r_2, M - 1), \quad (7)$$

where $R = (r_1 \to r_2 \to \cdots \to r_M)$. Indeed, the special form of total net profit can be realized by a recursion algorithm. To this end, for each road segment r_1, we can denote all the route candidates starting from r_1 as a *recursion tree* structure. Specifically, the recursion tree of a road segment can be defined as follows.

Definition 5 (Recursion Tree). *The recursion tree Υ_{r_1} of a road segment r_1 is a tree, where each node represents a road segment and the root node is r_1. Moreover, for each node r_i in the recursion tree, it has a children node set that equals to $r_i.next[]$.*

For example, Figure 4 shows an example of the recursion tree of road segment A. In this paper, we propose a method $RTree(r, M)$ for building a M-depth recursion tree Υ_r for r, which is shown in Algorithm 2. Particularly, the tree Υ_r obtained by our algorithm will hold M node sets $\Upsilon_r.level[i]$ $(1 \leq i \leq M)$, which represents the nodes in the i-th level of the tree. With this structure, the MNP recommendation from segment r_1 can be separated into several simpler MNP recommendation tasks recursively. Take Figure 4 as an example, we can develop a bottom-up method to compute the MNP route with length 3, of which the net profit is denoted as $G(A, 3)$. Specifically, according to the definition of net profit, we can obtain $G(A, 3) = g(A) + (1 -$

Algorithm 2 RTree(r,M)

```
Input 1: road segment $r_1$ as root node;
Input 2: the depth $M$ of recursion tree;
Output: a $M$-depth recursion tree $\Upsilon_r$;
Initialization: $Depth = 1$; $\Upsilon_r.level[i] = \emptyset$ $(1 \leq i \leq M)$;
 1: $\Upsilon_r.root = r$; $\Upsilon_r.level[1] = \{r\}$;
 2: if ($Depth \geq M$) do
 3:     return $\Upsilon_r$;
 4: else
 5:     for each ($r_{cur} \in \Upsilon.r.level[Depth]$) do
 6:         $\Upsilon_r.level[Depth+1] \cup = r_{cur}.next[]$;
 7:     $Depth+ = 1$;
 8:     go to Step 2;
```

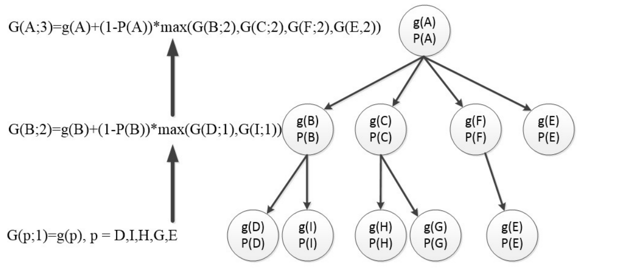

G(A;3)=g(A)+(1-P(A))*max(G(B;2),G(C;2),G(F;2),G(E,2))
g(A) P(A)
G(B;2)=g(B)+(1-P(B))*max(G(D;1),G(I;1))
g(B) P(B)
g(C) P(C)
g(F) P(F)
g(E) P(E)
G(p;1)=g(p), p = D,I,H,G,E
g(D) P(D)
g(I) P(I)
g(H) P(H)
g(G) P(G)
g(E) P(E)

Figure 4: The recursion tree representation of route network. We can calculate the MNP $G(R, A, 3)$ from the leaf nodes of the tree.

$P(A)) \times max\{G(B; 2), G(C; 2), G(F; 2); G(E; 2)\}$, where the net profit of each MNP route with length 2 can also be computed by the profit of their sub-routes. For example, we have $G(B; 2) = g(B) + (1 - P(B)) \times max\{G(D; 1), G(I; 1)\}$, and the profit of each individual segment (i.e., leaf nodes of the tree) can be directly computed by its profit, e.g., $g(D)$. Therefore, given a recursion tree of r, we can obtain the MNP route with length M by recursing $M - 1$ times. Specifically, in this paper we develop a recursion algorithm $rNMP(r, K)$ for MNP recommendation, which is shown in Algorithm 3. By implementing our algorithm with parameters $r = r_1$ and $K = M$, the MNP route starting from road segment r_1 with length M and corresponding MNP value will be obtained.

Algorithm 3 rMNP(r,K)

```
Input 1: recursion tree of $\Upsilon_r$;
Input 2: the depth $M$ of recursion tree;
Output: the MNP value and route stating from $r$;
 1: $Depth = M - K + 1$;
 2: if ($Depth == M$) do
 3:     $Profit = \emptyset$;
 4:     $Route = \emptyset$;
 5:     for each ($r_i \in \Upsilon_r.level[Depth]$) do
 6:         $Profit[i] = g(r)$;
 7:         $Route[i] = r_i$;
 8:     return ($Max(Profit)$,$Max(Route)$);
 9: else
10:     $Profit = \emptyset$;
11:     $Route = \emptyset$;
12:     for each ($r_i \in \Upsilon_r.level[Depth]$) do
13:         $(Profit^*, Route^*) = rMNP(r_i, K-1)$;
14:         $New_route[i] = r_i \cup Route^*$
15:         $Profit[i] = g(r_i) + (1 - p(r_i)) \cdot Profit^*$;
16:     return ($Max(Profit)$,$Max(Route)$);
```

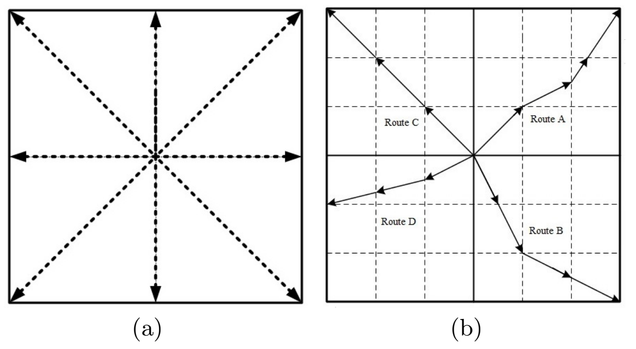


Figure 5: (a) Direction-based clustering; (b) Top-K route Recommendation.

Lemma 2. *Given a M-depth recursion tree Υ, where $\forall r \in \Upsilon$, $|r.next[]| \leq N$, the complexity of searching an optimal MNP route by the recursion method is $\mathcal{O}(N^{M-1})$*

Proof. Assume that the computational cost of finding $G(R, r_1, M)$ is $T(M)$, obviously we have $T(M) \leq NT(M-1)+1$. Moreover $\forall r$ satisfies $|r.next[]| = N$, the computation can be separated into N sub-problems. Particularly, for route with only one segment, we have $T(1) = 1$. Meanwhile, after recursing $M-1$ times, we have $T(M) \leq N^{M-1}T(1) + \frac{N^{M-1}}{N-1}$. Therefore, the computational complexity of searching optimal MNP route by recursing tree is $\mathcal{O}(N^{M-1})$ □

Although the recursion tree can achieve more efficient recommendation than the Brute-Force method, the computational cost increases significantly as M becomes larger. According to the discussion in Section 3, we can set an upper bond Λ for M, since the average increasing rate of the net profit is very low after $M > 5$. Therefore, we set $\Lambda = 5$ in our experiments.

4.3 Top-K Route Recommendation

Based on the above algorithms, our recommender system can recommend an optimal MNP route for a single taxi driver. However, in real life, an ideal recommender system must be capable of recommending multiple taxi drivers in the same area simultaneously. In this section, we address this problem and introduce a minimum redundant strategy for the recommendation process in the real world.

Intuitively, a straightforward recommendation strategy is to recommend the optimal driving route to all available drivers. However, if we recommend the same route to too many drivers at the same time, it will cause an overloaded problem and degrade the performance of the recommender system. The overloaded problem is a classic problem which has been widely studied. For example, the load balancing mechanism distributes requests among web servers in order to minimize the execution time [22, 10]. In our problem, we can treat multiple empty cabs as jobs and multiple optimal drive routes as computers. Instead of solving this overloaded problem by exploiting existing load balancing algorithm, we want to focus on the direction characteristics in the mobile recommender system and exploit a direction-based clustering (DEN) method [29] to distribute the empty cabs by following the top-K optimal drive routes [9, 23].

Before recommending driving routes to taxi drivers, we first rank all the route candidates according to their net profits and obtain the top-K driving routes. After recommending the top ranked route to the first taxi driver, we need to calculate the correlation between this route and all other $K-1$ candidate routes, and then recommend the route with the lowest correlationship [18] to the second driver.

In order to calculate the correlation between those candidate routes, we first partition the space into grids and turn the movement statistics in each grid into a vector which represents the probabilities of moving directions within the grid. Then, we transform the direction information of the taxis' movement into the same data format, and further partition each small grid into 8 direction bins. For example, in Figure 5(a) the angle of each bin has a range of $\pi/4$. Next, we transform each grid into a direction vector $g = (p_1, p_2, p_3, ..., p_8)$, where each p_i is the probability of moving towards direction i within this grid and $p_i = f_i / \sum_{k=1}^{8} f_k$, where f_i is the frequency of moving objects that have passed this grid and has the direction along the direction i.

For instance, as shown in Figure 5(b), we first recommend route A to the first taxi driver, route B,C and D are other candidate routes at the same time and same location. Then we divide the space into small grids and get the direction vectors for each grid. A driving route candidate which has lowest correlation with the previous recommending route is usually the one with a different driving direction in the beginning. Therefore, we only need to analyze the first n grids to decide the driving directions. We combine the direction vectors in n grids together and get a vector with $8n$ elements for each candidate route. For example, the vector for route A is $g(A) = (p_{11}, p_{12}, ...p_{n7}, p_{n8})$. Then, we calculate the correlation of those vectors for each pair of candidate routes. Thus, the correlation between route A and B can be computed by $\rho(A,B) = Cov(g(A), g(B))/\sigma_g(A)\sigma_{(}g(B)$. If route B has the lowest correlation with route A, we will recommend route B for next coming empty cab.

5. EXPERIMENTAL RESULTS

To validate the efficiency and effectiveness of the proposed recommender system, extensive experiments are performed on real world data sets collected in the San Francisco Bay Area in 30 days.

5.1 Experimental Data

Taxi GPS Traces. In the experiments, we use the real-world taxi GPS traces collected by the Exploratorium-the museum of science, art and human perception through the cabspotting project. The mobility traces are the records of the cabs' driving states in consecutive time, with each be represented as a tuple, (*latitude, longitude, fare identifier, time stamp*). By cleaning the dataset, we obtained 89,897 pick-up and drop-off activities in total. Generally, we assume that most drivers would follow the suggested driving route provided by the Google Map, thus we can get the fare related to the specific trip and the fare information can also be used to calculate the profits concerning the trip. The following Figure 6 is an example of one hundred taxi drivers' pick-up points in 30 days in the San Francisco Bay Area, with each red point representing one pick-up activity. Figure 7 is an heat map illustration of pick-up probabilities. Here, different color and area of circles represent different pick-up probabilities. This map shows there are lots of pick-up activities around the Market Street of San Francisco, which is a very busy street with lots of shopping places and

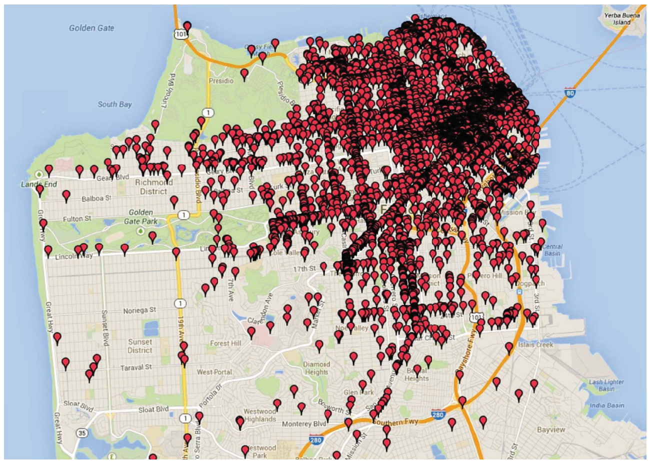


Figure 6: A Demonstration of pick-up points in the dataset.

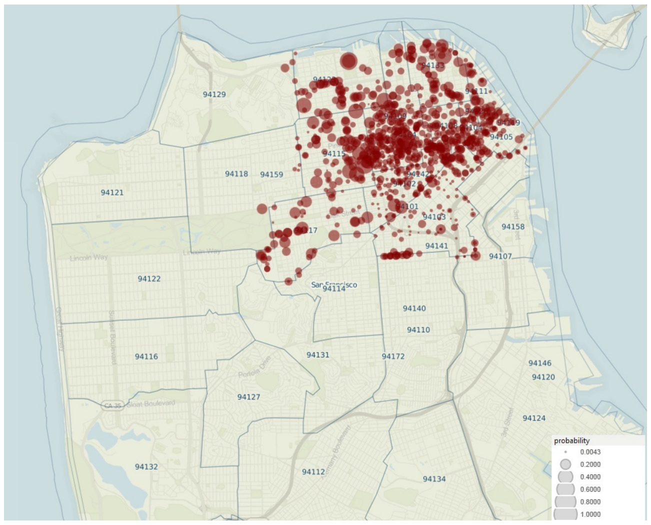
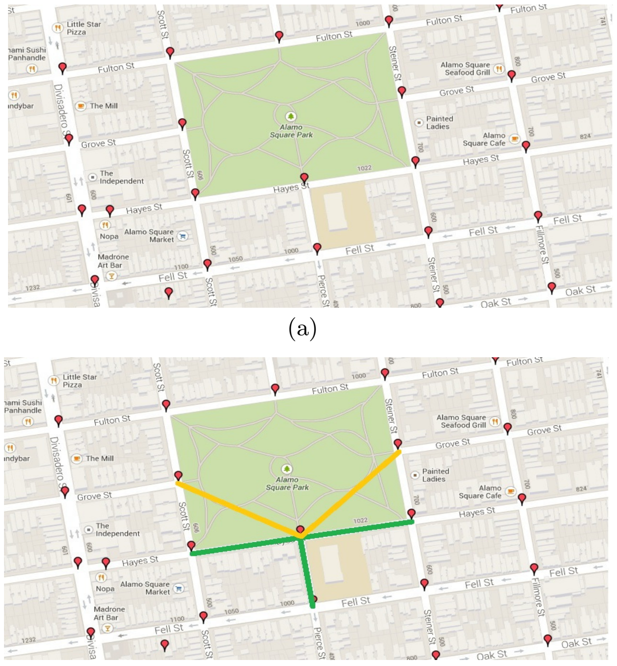


Figure 7: The heat map of pick-up probabilities in the San Francisco bay area.

museums. Other pick-up hot spots including Fisherman's Wharf, Divisadero St, Cathedral Hill and Western Addition.

Road Network Data. Because the quality of existing road networks in San Francisco is not sufficient. We build the road network dataset of San Francisco by using google API. First, we searched for all the street names in San Francisco. Second, we run the google API to find out if there is an intersection between two streets. We keep a record of each intersection point. Figure 8(a) is an illustration of our intersection points. Then, we use each intersection point to search the nearest points in four different directions and connect those 5 points together. Therefore, we can obtain four different connected road segments with starting points and ending points. However, as the yellow line in Figure 8(b), we may accidentally connect two intersections with no road between them. To solve this problem, we calculate the distance of those two intersections by using coordinates and compare it with the driving distance measured by the Google map. If there is a road between those two points, those two distances should be very close to each other. If the distances are not close to each other, it means there is no road between those two intersections and we delete this road segment from the road network dataset.



(a)

(b)

Figure 8: (a) Intersections; (b) Connected Road Segments.

The road network dataset contains 5391 roads in the San Francisco Bay Area, with each consisting of the ID, starting points, ending points and we also calculate the historical pick-up probability and net profit associate with each road segment. For each road, several coordinates of the intermediate points may be recorded and there are also some noise points. After removing the noise points, we selected 2,149 roads with high pick-up probability for our experiment. Then, we can build road buffer with the starting points and the ending points in those road segments.

By matching the pick-up coordinates of the Road Network Dataset with the Taxi Dataset, we are able to get 87,688 valid pick-up activities which can be located in the road segments, therefore the two data sets are combined together with each pick-up point mapped to the constructed road buffer. To implement the proposed algorithm, we also need to calculate the pick-up probability and the net profit for each road segment in those road segments. This has already been presented in Section 4.

Finally, we get the coordinates of the starting and the ending points for each road segment, along with the pick-up probability, the net profit and the average driving time in this road segment. Note that the average driving time is estimated as the distance of each road segment divided by the average driving speed in the San Francisco Bay Area.

5.2 Empirical Studies on Recommendations

Here, we provide two case studies. One case study is on cost effective route recommendations. Another case study is on top-k recommendations.

5.2.1 *A Case Study on Cost-Effective Route Recommendations*

Here, we show two examples of MNP route recommended by our approach and compare it with the suggested route by the Google map. Specifically, in Figure 9 and Figure 10, we plot the optimal driving route suggested by our rec-

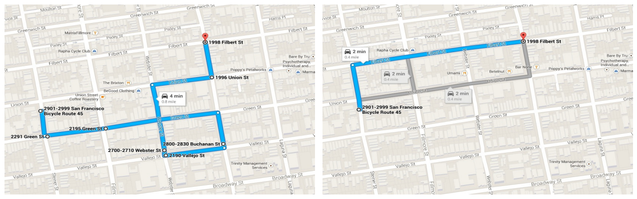


Figure 9: Case Study (a)

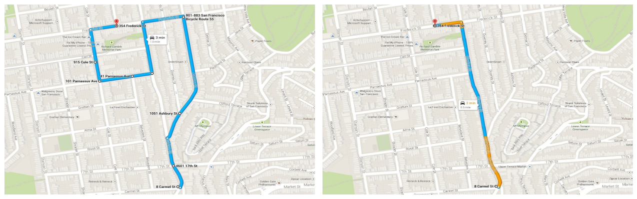


Figure 10: Case Study (b)

ommender system at a randomly selected initial location of the target cab. We also assumed that the driver's expected cruising length is 5, and after every 5 road segments, the system will use the current location as the new starting point for search and restart the recommendation process. In order to do the comparison, we calculate the real driving time of each trip of taxi drivers and restart our recommendation system until the total driving time in those MNP routes is equal to the real driving time of each trip. Then, we connect those MNP routes together and this is the entire driving route that should be recommended to the drivers. In those Figures, the left figures are the driving route recommended by the MNP recommender system and the right figures are the route suggested by the Google Map based on the shortest driving distance. However, this driving route suggested by the Google map cannot maximize taxi drivers' net profit.

Recently, most recommender system can only suggest a sequence of hot spots to taxi drivers. There is no such recommendation system that can suggest an entire driving route. If taxi drivers do not know how to drive to the nearest hot spot, he or she has to follow the driving route provided by the Google map. However, both the pick-up probability and the potential net profit may be very low along those routes. The drivers have a high probability of losing money until they reach the next hot spot. Our recommender system can improve potential net profits for taxi drivers compared to the routes suggested by the Google map.

5.2.2 *A Case Study on Top-K Recommendations*

In Section 4, we introduced a minimum redundant strategy to recommend the Top-K driving routes and solved the overloaded problem. In figure 11, we demonstrate the Top K driving routes starting from the same location, where K equals to 4 in this case. The figure shows that each route has different driving directions and the correlations between those driving distances are very small. Therefore, the minimum redundant strategy can improve the performance of our recommender system.

5.3 Route Recommendation for Inexperienced Taxi Drivers

Given one specific location, our proposed algorithm can recommend several routes with high expected utility for drivers. The algorithm is especially applicable for inexperienced drivers, since they lack of knowledge about the roadmap and the local driving routes that can make profits. To validate the effectiveness of the proposed algorithm, we firstly divide all the drivers into two categories based on their average net profits. The top 10% drivers in the dataset are treated as 'experienced' drivers, while the others are 'inexperienced'. Therefore, the driving routes of the experienced drivers are used as training set and we recommend driving routes for the inexperienced drivers.

We define driver's event $\mathbf{e}$ as a consecutive sequence of 'roam $\rightarrow$ pick up $\rightarrow$ drop off', by extracting the pick-up and drop-off activities of each user, we can reconstruct each event. For each driver, we define the location where the driver starts to search for potential pick-ups as l_0, and after roaming in Δt time, the driver picks up passengers at location l_1 and drive for $\Delta t'$ and drop off at l_2. Let $r_{i,j}$ denotes the road segment between location l_i and l_j, then event e can be represented with $(r_{0,1}, \Delta t, r_{1,2}, \Delta t')$, and the unit time profit of the event can be calculated as $p_e = \frac{pr_{12}}{\Delta t + \Delta t'}$. Thus, the proposed algorithm starts with the location l' which is neareset to l_0, and return a sequence of recommended potential pick-up points and road segments.

The performance of the recommended driving route is measured by the average net profit per unit time p_r, and it is compared with the average unit net profit of the inexperienced drivers, i.e., $p_d = \frac{\sum p_e}{|\mathbf{e}|}$.

The statistical experiment results for recommended driving routes for inexperienced drivers are shown in Table 1, the average net profits per unit time outperforms the real profit of the inexperienced drivers.

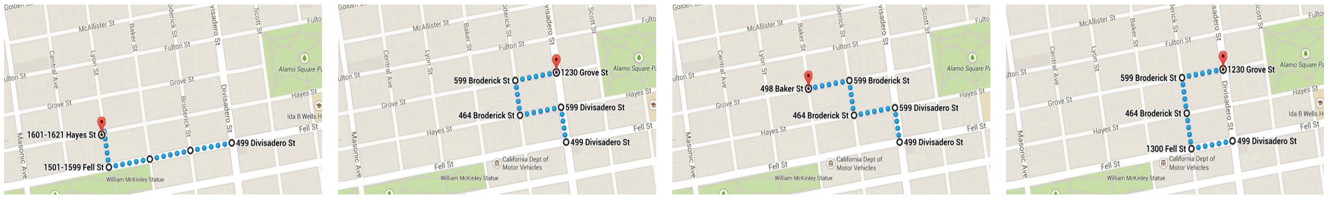
Table 1: Net Profits per Unit Time

	Recommender System	Inexperienced Drivers
Mean	0.038148	0.024162
SD	0.017815	0.018455

We first plot the distribution for the net profit per unit time, i.e., the number of events for specific profit values, as shown in Figure 12. The net profit per unit time of our recommended route is compared with the inexperienced taxi drivers' performance based on statistical histogram. The blue bar of the histogram shows the net profits from our recommendation results and the red bar shows the profits from the inexperienced taxi drivers. As we can see from the figure, the recommendation events mostly positioned on bigger values. This indicates that our recommender systems provide higher profit routes than the real routes by inexperienced drivers.

To further investigate the performance of the recommender system, we also study the difference of net profit per unit time between the recommended routes and the drivers' real routes for each event, i.e. $p_r - p_e$. As shown in Figure 13, the X axis is the difference between the profits of the recommended results and the inexperienced taxi drivers' profits. We can see that most of dot points are positioned to the right of $X = 0$, meaning that the profits of our recommended routes outperform the profits of the routes by the inexperienced drivers.

Then, we evaluated the performance of the Brute-Force recommendation strategy and the performance of the recursive recommendation strategy. This experiment was conducted across 1000 randomly picked starting points. We



(a) (b) (c) (d)

Figure 11: The Top 4 driving routes starting from the same location (longitude: 37.77407074 and latitude:-122.4376221)

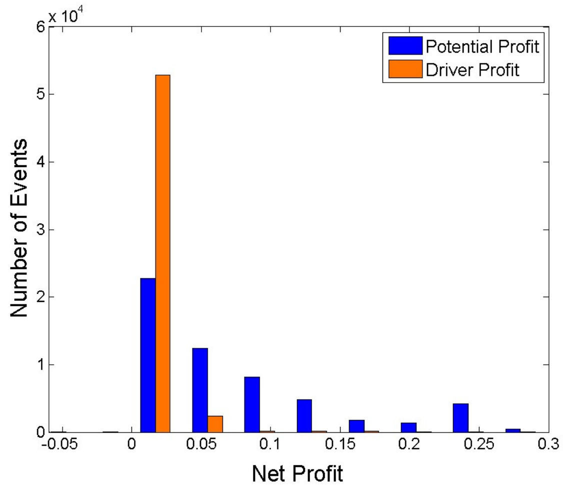


Figure 12: Net Profit Statistics. The blue bar represents the potential profits of our optimized routes and the orange bar represents the profits of taxi drivers' traditional routes ranked below top 10%

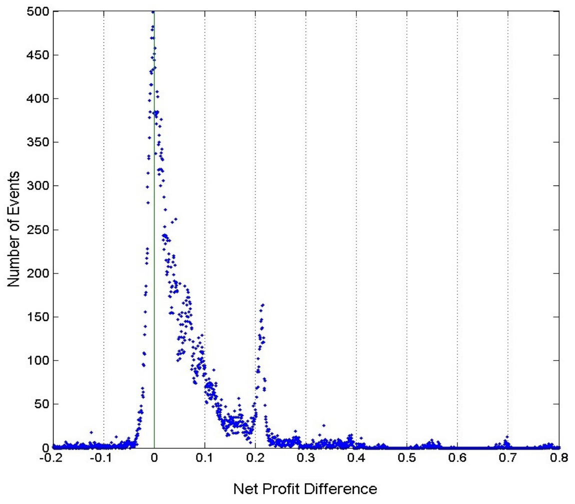


Figure 13: Profit Difference. X axis is Net Profit Difference between our optimized routes and taxi drivers' traditional routes ranked below top 10%. Y axis is the number of events

only compared the running time for five road segments, because the increasing rate of pick-up probability in Equation 4 is less than 10% after increasing more than 5 road segments. As shown in Figure 14, the red line is the running time for the Brute-Force recommendation strategy and the black line is the running time of the recursive Strategy. We can see that the recursive strategy can lead to better efficiency compared to the Brute-Force strategy. Note that all the experiments were conducted on a Windows 7 with Intel(R) Core(TM)i5-3210 CPU and 6.0 GB RAM.

To sum up, the experiments showed that the cost-effective recommender system could help inexperienced taxi drivers find better routes so as to maximize their potential profits. Also, the recursive strategy can help to efficiently identify the recommended optimal routes.

6. CONCLUDING REMARKS

In this paper, we proposed a cost-effective recommender system for taxi drivers to maximize their profits by providing profitable driving routes. To be specific, we first provided a net profit objective function for evaluating the driving routes before finding a customer. Then, we proposed a graph based approach to efficiently generate candidate driving routes for finding passengers. As a result, we can use the net profit objective function to rank each candidate route and make recommendations to taxi drivers in a cost-effective way. An unique perspective of our recommender system is that it can recommend an entire driving route instead of only recommending a sequence of discrete pick-up points. Also, the drivers are able to maximize their profits within the fixed time period by following the recommended driving routes. Finally, the extensive experiments on a real-world data set collected from the San Francisco Bay area clearly validated the effectiveness of the proposed recommender system.

Acknowledgement

This research was partially supported by National Science Foundation (NSF) via grant numbers CCF-1018151 and IIS-1256016. Also, it was supported in part by Natural Science Foundation of China (71329201)

7. REFERENCES

[1] G. Abowd, C. Atkeson, and et al. Cyber-guide: A mobile context-aware tour guide. *Wireless Networks*, 3(5):421–433, 1997.
[2] G. Adomavicius and A. Tuzhilin. Towards the next generation of recommender systems: A survey of the state-of-the art and possible extensions. *TKDE*, 2005.
[3] O. Averjanova, F. Ricci, and Q. N. Nguyen. Map-based interaction with a conversational mobile

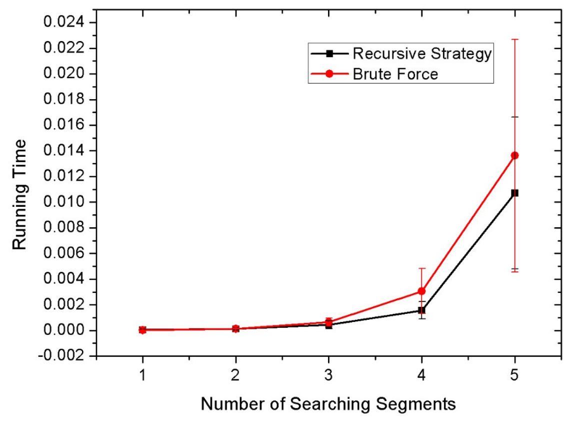


Figure 14: A Comparison of the Running Time. The red line represents the running time of the Brute-Force strategy and the black line represents the running time of the Recursive strategy

recommender system. In *The 2nd Int'l Conf on Mobile Ubiquitous Computing, Systems, Services and Technologies*, 2008.

[4] R. M. Bell and Y. Koren. Scalable collaborative filtering with jointly derived neighborhood interpolation weights. In *Data Mining, 2007. ICDM 2007. Seventh IEEE International Conference on*, pages 43–52. IEEE, 2007.

[5] F. Cena, L. Console, and et al. Integrating heterogeneous adaptation techniques to build a flexible and usable mobile tourist guide. *AI Communications*, 19(4):369–384, 2006.

[6] K. Cheverst, N. Davies, and et al. Developing a context-aware electronic tourist guide: some issues and experiences. In *the SIGCHI Conference on Human Factors in Computing Systems*, pages 17–24, 2000.

[7] M. Deshpande and G. Karypis. Item-based top-n recommendation algorithms. *ACM Transactions on Information Systems (TOIS)*, 22(1):143–177, 2004.

[8] Y. Ge, H. Xiong, A. Tuzhilin, K. Xiao, M. Gruteser, and M. Pazzani. An energy-efficient mobile recommender system. In *Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 899–908. ACM, 2010.

[9] Y. Ge, H. Xiong, Z.-H. Zhou, H. Ozdemir, J. Yu, and K. Lee. Top-eye: Top-k evolving trajectory outlier detection. In *CIKM'10*.

[10] D. Grosu and A. T. Chronopoulos. Algorithmic mechanism design for load balancing in distributed systems. *IEEE TSMC-B*, 34(1):77–84, 2004.

[11] Y. Koren. Factorization meets the neighborhood: a multifaceted collaborative filtering model. In *Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 426–434. ACM, 2008.

[12] B. N. Miller, I. Albert, and et al. Movielens unplugged: Experiences with a recommender system on four mobile devices. In *international conference on Intelligent user interfaces*, 2003.

[13] R. J. Mooney and L. Roy. Content-based book recommendation using learning for text categorization. In *Workshop Recommender Systems: Algorithms and Evaluation*, 1999.

[14] G. Nagy and S. Salhi. Heuristic algorithms for single and multiple depot vehicle routing problems with pickups and deliveries. *European Journal of Operational Research*, 162(1):126–141, 2005.

[15] M. Pazzani. A framework for collaborative, content-based, and demographic filtering. *Artificial Intelligence Review*, 1999.

[16] J. W. Powell, Y. Huang, F. Bastani, and M. Ji. Towards reducing taxicab cruising time using spatio-temporal profitability maps. In *Proceedings of the 12th International Conference on Advances in Spatial and Temporal Databases*, SSTD'11, pages 242–260, Berlin, Heidelberg, 2011. Springer-Verlag.

[17] X. Su and T. M. Khoshgoftaar. A survey of collaborative filtering techniques. *Adv. in Artif. Intell.*, 2009:4:2–4:2, Jan. 2009.

[18] L.-A. Tang, Y. Zheng, X. Xie, J. Yuan, X. Yu, and J. Han. Retrieving k-nearest neighboring trajectories by a set of point locations. In *Advances in Spatial and Temporal Databases*, pages 223–241. Springer, 2011.

[19] A. Tveit. Peer-to-peer based recommendations for mobile commerce. In *the 1st international workshop on Mobile commerce*, 2001.

[20] H. van der Heijden, G. Kotsis, and R. Kronsteiner. Mobile recommendation systems for decision making 'on the go'. In *ICMB*, 2005.

[21] H. Xiong, S. Shekhar, Y. Huang, V. Kumar, X. Ma, and J. S. Yoo. A framework for discovering co-location patterns in data sets with extended spatial objects. In *SDM*. SIAM, 2004.

[22] Z. Xu and R. Huang. Performance study of load balancing algorithms in distributed web server systems. In *TR*, CS213 Univ. of California,Riverside.

[23] J. Yuan, G.-Z. Sun, Y. Tian, G. Chen, and Z. Liu. Selective-nra algorithms for top-k queries. In *Advances in Data and Web Management*, pages 15–26. Springer, 2009.

[24] J. Yuan, Y. Zheng, X. Xie, and G. Sun. T-drive: Enhancing driving directions with taxi drivers' intelligence. *Knowledge and Data Engineering, IEEE Transactions on*, 25(1):220–232, 2013.

[25] J. Yuan, Y. Zheng, C. Zhang, W. Xie, X. Xie, G. Sun, and Y. Huang. T-drive: driving directions based on taxi trajectories. In *Proceedings of the 18th SIGSPATIAL International Conference on Advances in Geographic Information Systems*, pages 99–108. ACM, 2010.

[26] J. Yuan, Y. Zheng, L. Zhang, X. Xie, and G. Sun. Where to find my next passenger. In *Proceedings of the 13th international conference on Ubiquitous computing*, pages 109–118. ACM, 2011.

[27] Y. Zheng, Y. Liu, J. Yuan, and X. Xie. Urban computing with taxicabs. In *Proceedings of the 13th international conference on Ubiquitous computing*, pages 89–98. ACM, 2011.

[28] Y. Zheng, J. Yuan, W. Xie, X. Xie, and G. Sun. Drive smartly as a taxi driver. In *Ubiquitous Intelligence & Computing and 7th International Conference on Autonomic & Trusted Computing (UIC/ATC), 2010 7th International Conference on*, pages 484–486. IEEE, 2010.

[29] W. Zhou, H. Xiong, Y. Ge, J. Yu, H. Ozdemir, and K. C. Lee. Direction clustering for characterizing movement patterns. In *Information Reuse and Integration (IRI), 2010 IEEE International Conference on*, pages 165–170. IEEE, 2010.

[30] H. Zhu, E. Chen, K. Yu, H. Cao, H. Xiong, and J. Tian. Mining personal context-aware preferences for mobile users. In *Proceedings of the IEEE 12th International Conference on Data Mining*, ICDM'12, pages 1212–1217, 2012.