Pick-Up Tree Based Route Recommendation from Taxi Trajectories

Haoran Hu[1], Zhiang Wu[2,⋆], Bo Mao[2], Yi Zhuang[3], Jie Cao[2], and Jingui Pan[1]

[1] State Key Lab. for Novel Software Technology, Nanjing University, Nanjing, China
[2] Jiangsu Provincial Key Laboratory of E-Business,
Nanjing University of Finance and Economics, Nanjing, China
[3] College of Computer and Information Engineering,
Zhejiang Gongshang University, Hangzhou, China
zawuster@gmail.com

Abstract. Recommending suitable routes to taxi drivers for picking up passengers is helpful to raise their incomes and reduce the gasoline consumption. In this paper, a pick-up tree based route recommender system is proposed to minimize the traveling distance without carrying passengers for a given taxis set. Firstly, we apply clustering approach to the GPS trajectory data of a large number of taxis that indicates state variance from "free" to "occupied", and take the centroids as potential pick-up points. Secondly, we propose a heuristic based on skyline computation to construct a pick-up tree in which current position is its root node that connects all centroids. Then, we present a probability model to estimate gasoline consumption of every route. By adopting the estimated gasoline consumption as the weight of every route, the weighted Round-Robin recommendation method for the set of taxis is proposed. Our experimental results on real-world taxi trajectories data set have shown that the proposed recommendation method effectively reduce the driving distance before carrying passengers, especially when the number of cabs becomes large. Meanwhile, the time-cost of our method is also lower than the existing methods.

Keywords: Taxi trajectories, pick-up tree, route recommendation, clustering, skyline.

1 Introduction

The rapid growth of wireless sensors and development of Global Positioning System (GPS) technologies [1] make it increasingly convenient to obtain the time-stamped trajectory data of taxis. In practice, cab drivers wish to be recommended a fastest route to pick up passengers. Such a large number of trajectories provide us unprecedented opportunity to mine useful knowledge and to recommend efficient routes for cab drivers. This recommender system not only helps cab drivers to raise the income, but also decreases the gasoline consumption which is good for environmental protection.

⋆ Corresponding author.

In the literature, a great deal of research has been devoted to mobile recommendations. These studies mainly focus on the following subareas: the mobile tourist recommender systems [2, 3, 4, 5], taxi driving fraud detection systems [6], driving direction recommender systems [7, 8], and routes for carrying passengers recommendation [9], which have greatly advanced the research to a higher level.

The scope of this paper belongs to the field of recommending routes for carrying passengers. The idea of this paper stems from the work in [9] which aims to recommend a travel route for a cab driver in a way such that the potential travel distance before carrying passengers is minimized. In their approach, for the length of suggested driving route $\mathcal{L}$, there are $C_K^{\mathcal{L}}$ candidate routes where K is the number of pick-up points. Then, routes which are not dominated by any other route are recommended to cab drivers. The main task in [9] is to search the best routes among these $C_K^{\mathcal{L}}$ candidate routes. Although a satisfying strategy called *SkyRoute* is proposed [9], it is still time-consuming as the increase of $\mathcal{L}$ and K. In the meanwhile, a multitude of routes are recommended to cab drivers in the equal possibilities without considering the number of passengers and driving distance of the routes, as shown in the left side of the Fig. 1. If we consider the scene that many taxis head for a narrow route with few passengers, conflict for carrying customers or even the traffic jam in that route may happen.

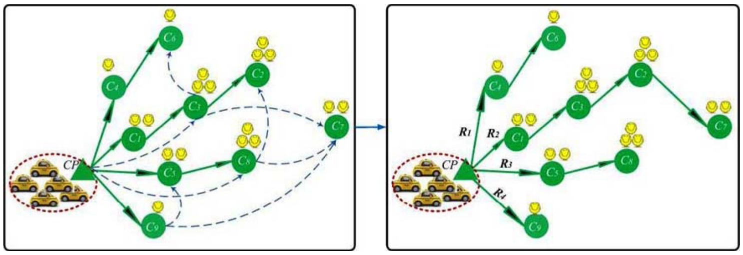


Fig. 1. The illustration for our research motivation

To meet this dilemma, a data structure named *pick-up tree* is presented which takes the current position as its root node and connects all pick-up points is presented. The better routes are recommended in a high probability as shown in thick line in right side of Fig. 1. Moreover, if we assume there are a set of cabs around CP, by adopting our recommendation, most of cabs will cruise along with the better routes and other relatively worse routes will also be covered by less cabs. There remain two problems to solve when we follow this idea:

- An objective function measuring the global profit of the set of taxis by adopting recommendation should be presented. The probabilities for picking up passengers at points should be estimated. Also, the expectation for measuring the weight of routes should be given.
- Finding the optimal value of objective function among all routes is a combinatorial optimization problem which is NP_hard. How to apply a heuristic to this NP_hard problem should be solved.

In this paper, we focus on the recommendation and minimization of the gasoline consumption which is close related to idle driving distance for *a set of cabs* around a position, rather than *one cab*. Historical pick-up points are extracted and clustered, and routes connecting the centroids are taken as recommendation choise. We then propose a heuristic to construct *a pick-up tree* to cover all centroids. The model for estimating oil consumption before carrying passengers of every route is presented. By adopting the oil consumption as the weight of every route, the recommendation method for the set of taxis is proposed.

2 Problem Definition

Let $\mathcal{C}$ be the set of K potential pick-up points $\mathcal{C} = \{C_1, C_2, \cdots, C_K\}$. We then assume there are W target taxis around a position CP, $\mathcal{T} = \{T_1, T_2, \cdots, T_W\}$. The probability that a taxi could carry passengers in the pick-up point C_i is written $P(C_i)$, and the set of mutually independence probability is written $\mathcal{P} = \{P(C_1), P(C_2), \cdots, P(C_K)\}$.

The route recommendation aims to generate W routes for every taxi, and minimize the global gasoline consumption before picking up passengers for all taxis. So, we have the objective function as follows:

Definition 1 (Global Gasoline Consumption Function). *Given W recommended routes $\mathcal{R} = \{\boldsymbol{R_1}, \boldsymbol{R_2}, \cdots, \boldsymbol{R_W}\}$, the global oil consumption function:*

$$O_1 : \min_{\mathcal{R} \in \Omega} \sum_{r=1}^{W} \mathbb{C}(CP, \boldsymbol{R_r}) \tag{1}$$

In the *definition* 1, CP is the current position of taxis, the set of all possible routes is known as Ω, and we try to find a subset of Ω with size of W for minimization. The problem is NP_hard due to the exponential scale of Ω. The oil consumption of a route is measured by the $\mathbb{C}(\bullet)$ function which will be addressed in Section 4. Note that the problem should not only minimize the global gasoline consumption, but also attempt to cover all pick-up points.

3 CabRec Design

In this section, we propose a route recommendation method, named CabRec, for cab drivers. To illustrate it, we first present a heuristic algorithm to generate a pick-up tree. Then, we show how to compute the gasoline consumption for each route. Finally,we describe the weighted Round-Robin recommendation process.

3.1 Pick-Up Tree Generation

The problem is to construct a tree which uses CP as the root and connects all points in $\mathcal{C}$. The straightforward way is to add an edge between any of two nodes, but the derived network will be too complex to process. In our solution, edges between a point and its skyline points are added. The set of skyline points consists of the points that are not dominated by any other point [10]. We should begin with the definition of the point dominance.

Definition 2 (Point Dominance). *Let CP' denote the current node, C_i and C_j are two different pick-up points. These two pick-up points can be described by two dimensions $C_i = (P(C_i), D(C_i, CP'))$ and $C_j = (P(C_j), D(C_j, CP'))$. We say that point C_i dominates point C_j iff one of the cases happens: (1) $P(C_i) = P(C_j)$ and $D(C_i, CP') < D(C_j, CP')$; (2) $P(C_i) > P(C_j)$ and $D(C_i, CP') = D(C_j, CP')$; (3) $P(C_i) > P(C_j)$ and $D(C_i, CP') < D(C_j, CP')$.*

The pseudocode of the pick-up tree generation algorithm is given as follows. Lines 2 and 3 are about initialization. The *FindSkyline* function in line 2 finds skyline points for the current position CP. Then, in line 5, we use a heuristic metric with maximum ratio of pick-up probability to distance to select the next expanding node. The algorithm stops when all points in $\mathcal{C}$ are added to the pick-up tree.

Discussion. If we construct a rectangular coordinate system with CP' as its origin, all remaining pick-up points in $\mathcal{C}$ can be depicted by $C_i = (P(C_i), D(C_i, CP'))$. That there is at least a skyline point for CP'. So, in each *while* loop, at least a point is added to the pick-up tree. The pick-up tree guarantees to include all points in $\mathcal{C}$.

Algorithm 1. Pick-up Tree Generation Algorithm

```
1: procedure CreatePTree($CP, \mathcal{C}$)
2:     $V \leftarrow$ FindSkyline($CP, \mathcal{C}$)
3:     Add edges between $CP$ and its skyline points
4:     while $(\mathcal{C} - V) \neq NULL$ do
5:         $CP' \leftarrow \arg\max_i\{\frac{P(C_i)}{D(C_i, CP')}, i = 1, \cdots, |V|\}$
6:         $V \leftarrow V +$ FindSkyline($CP', \mathcal{C} - V$)
7:         Add edges between $CP'$ and its skyline points
8:     end while
9: end procedure
```

It can be seen clearly that skyline computation encapsulated in the *FindSkyline* function can be done in polynomial time. The straightforward method is to compare each point with all the other points to check whether it can be dominated by some points. If so, remove it, otherwise mark this point as a skyline point. The time complexity of this method is $O(2n^2)$, but with the increase of the number of pick-up points, the performance *FindSkyline* will become the bottleneck. In this article, we use a sort-based skyline computation to implement *FindSkyline*.

Obviously, *FindSkyline* is much more faster than the straightforward way, because comparison is made just on the skyline points and all the points only need to visit once. Also, *Theorem* 1 is presented to guarantee the correctness of the process of *FindSkyline*.

Theorem 1. *After sorting on the data set $\mathcal{D}$, the one with the maximum value will be certainly the skyline point.*

Proof: If the maximum point p_{max} is not a skyline point, it must be dominated by at least one point represented by q. According to the *definition* 1, $P(q) - D(q, CP') > P(p_{max}) - D(p_{max}, CP')$ is surely satisfied. So, p_{max} is not the maximum point and contradiction appears. Proof done.

Algorithm 2. Sort-based Skyline Computation Algorithm

```
1: procedure FindSkyline($CP'$, $\mathcal{D}$)
2:     Sort all points in $\mathcal{D}$ with $P(D_i) - D(D_i, CP')$,get $\mathcal{D}'$
3:     Maintain a set $\mathcal{S}$ for skyline points
4:     Move the first point into $\mathcal{S}$ from $\mathcal{D}'$
5:     while $\mathcal{D}' \neq NULL$ do
6:         Let $p_{max}$ denote the current maximum in $\mathcal{D}'$
7:         Compare $p_{max}$ with all the points in $\mathcal{S}$
8:         If $p_{max}$ is dominated by some points in $\mathcal{S}$,
9:         Just remove $p_{max}$ from $\mathcal{D}'$
10:        Else move $p_{max}$ from $\mathcal{D}'$ to $\mathcal{S}$
11:    end while
12: end procedure
```

3.2 Computational Issues

We assume the oil consumption increases proportionately with the driving distance before carrying passengers. Let k_{max} denote the length of the route $\boldsymbol{R_r} = \{C_1, C_2, \cdots, C_{k_{max}}\}$. Two cases may happen when a taxi selects the route: (1) the taxi carries a customer in one of the k_{max} nodes; (2) the taxi does not carry any customer in that route.

1. Assume the taxi carries a passenger in $C_k (k = 1, 2, \cdots, k_{max})$, the travel distance without customers is as follows:

$$\mathcal{F}_k = \prod_{i=1}^{k-1} (1 - P(C_i)) P(C_k) \sum_{i=1}^{k} D_i \tag{2}$$

   In Eq. (2), D_i is the distance between $C_{(i-1)}$ and C_i, and D_1 is the distance between CP and C_1. The most common way for computing D_i is to use Euclidean distance. However, the earth is roughly a great circle, and the latitude and longitude are defined globally in respect to the earth surface instead of a plane. In this article, we employ Vincenty's formula [11] with the assumption of spherical earth. Let $C_{(i-1)} = (\phi_{(i-1)}, \lambda_{(i-1)})$ and

$C_i = (\phi_i, \lambda_i)$ denote the two points, where ϕ and λ are the latitude and longitude, respectively.

$$D_i = r \cdot \arctan\left(\frac{\sqrt{(\cos\phi_i \sin\triangle\lambda)^2 + (\cos\phi_{(i-1)}\sin\phi_i - \sin\phi_{(i-1)}\cos\phi_i\cos\triangle\lambda)^2}}{\sin\phi_{(i-1)}\sin\phi_i + \cos\phi_{(i-1)}\cos\phi_i\cos\triangle\lambda}\right) \tag{3}$$

In Eq. (3), r is the radius of the Earth and is set to $6372.795km$, and $\triangle\lambda = |\lambda_i - \lambda_{i-1}|$. Eq. (4) measures the average oil consumption in the case of carrying a customer in the route. The smaller $\mathcal{F}(CP, \boldsymbol{R_r})$ is, the more valuable the route will be.

$$\mathcal{F}(CP, \boldsymbol{R_r}) = \sum_{k=1}^{k_{max}} \mathcal{F}_k \tag{4}$$

2. The event that a taxi does not carry a customer in a route may also happen. We have the probability of occurrence as follows:

$$P_\phi(\boldsymbol{R_r}) = \prod_{i=1}^{k_{max}} (1 - P(C_i)) \tag{5}$$

In Eq. (5), $P_\phi(\boldsymbol{R_r})$ decreases with the increase of the length of a route, and the smaller it is, the more valuable the route is.

We then define a cost function to combine above-mentioned two cases:

$$\mathbb{C}(CP, \boldsymbol{R_r}) = \alpha^{P_\phi}\mathcal{F}(CP, \boldsymbol{R_r}) \tag{6}$$

In Eq. (6), an exponential function is employed to magnify P_ϕ. We set $\alpha > 1$ to make the cost monotone increasing with both P_ϕ and $\mathcal{F}(\bullet)$.

3.3 Recommendation Method

Assume the pick-up tree has $N(N \leq K)$ leaf nodes, so there are N possible routes $\mathbf{R} = \{\boldsymbol{R_1}, \boldsymbol{R_2}, \cdots, \boldsymbol{R_N}\}$. In this subsection, we introduce the method for recommending the N routes to W target taxis. We propose to use a standard weight to measure the importance of a route. The cost for each route is obtained by Eq. (6). We then define the weight of each route as follows:

$$\omega(\boldsymbol{R_r}) = \frac{1}{N-1}\frac{\sum_{i=1}^{N}\mathbb{C}(CP, \boldsymbol{R_i}) - \mathbb{C}(CP, \boldsymbol{R_r})}{\sum_{i=1}^{N}\mathbb{C}(CP, \boldsymbol{R_i})} \tag{7}$$

Obviously, weights for all routes are normalized, $\sum_{i=1}^{N}\omega(\boldsymbol{R_i}) = 1$. A Round-Robin method is used to make recommendation for multiple cabs in [9], and k optimal routes are used to generate the circle list. The Round-Robin approach recommends routes in the circle list to the coming empty cabs in turn. If there

Table 1. Description of the 10 centroids

No.	C_0	C_1	C_2	C_3	C_4	C_5	C_6	C_7	C_8	C_9
latitude	37.7901	.7773	.7752	.7472	.7990	.7971	.7641	.7869	.7705	.7800
longitude	-122.4199	.3949	.4169	.4198	.4344	.4043	.4277	.4062	.4657	.4391
Size	3075	1357	1838	1763	2154	3818	3506	4964	1253	2245
Dist.(km)	1.136	2.195	2.243	5.319	2.383	0.466	3.759	0.816	5.711	3.151
$P(C_i)$	0.120	0.082	0.120	0.093	0.131	0.090	0.097	0.097	0.101	0.117

Note: (1) Only the fractional part of latitude and longitude are kept.
(2) Size is the number of pick-up points in this cluster.

Table 2. The cost and weight of recommended routes

No.	Recommended routes	Cost	Weight
1	(CP, C_0)	0.195	0.192
2	(CP, C_4)	0.444	0.182
3	(CP, C_7)	0.114	0.195
4	(CP, C_5, C_1)	0.354	0.185
5	(CP, C_5, C_2, C_6)	1.006	0.158
6	$(CP, C_5, C_2, C_9, C_8, C_3)$	2.731	0.088

are more than k empty cabs, recommendations are repeated from the first route again after the kth empty cab.

In this paper, we firstly generate the circle list according to the weight of each route. For W empty cabs, we add $\lceil \omega(\boldsymbol{R_r}) \cdot W \rceil$ times into the circle list for route $\boldsymbol{R_r}$. Also, we randomly arrange routes in the circle list and employ the Round-Robin method to generate a recommended route for each cab. We benefits the proposed recommendation method from two aspects: (1) each route appears in the circle list at least once, so none pick-up point is ignored; (2) the route with low cost has more opportunities appearing in the circle list, so more cabs will cruise to that route.

Discussion. In section 2, the green recommendation problem is formulated a combinatorial optimization problem which is NP_hard. So, our proposed CabRec is a heuristic in essence. However, the cost estimation for a route and the recommendation method are deterministic. In fact, we apply heuristic during construction of the pick-up tree, and once the pick-up tree is generated, the recommendation list can be produced. This strategy is more efficient The proposed CabRec does not always recommend the best route to cabs, but recommend better routes with high probability.

4 Experimental Results

In this section, we demonstrate the effectiveness of the proposed CabRec. Specifically, we will show: (1) An example of the pick-up tree and the cost and weight of every route; (2) The superior performance of CabRec compared with $\mathcal{LCP}$ on driving distance before carrying passengers and time-cost.

4.1 Experiment Setup

Data Sets. For our experiments, we use a real-world data set containing GPS location traces of approximately 500 taxis collected around 30 days in the San Francisco Bay Area, USA. To distinguish from other works, state variance points are deemed to be the potential pick-up points. The state of the pick-up point is 1(= *occupied*), and the state of its previous point is 0(= $free$). Then, we extract the pick-up points of all cab drivers (536 cabs) on two time periods: *2PM-3PM* and *6PM-7PM*. In total, 25973 points are obtained during *2PM-3PM*, and 6203 points are obtained during *6PM-7PM*. Simple K-means provided by WEKA[1] is used for clustering where"Euclidean Distance" function,"10" seed and "500" max iterations are set.

Methods. Two types of route recommendation methods are implemented for performance comparison. The first one is the proposed CabRec, and another one is $\mathcal{LCP}$ proposed in [9], coded by ourselves in Java. We use $\mathcal{LCP}_x$ to denote the $\mathcal{LCP}$ method of which the length of recommended route is x.

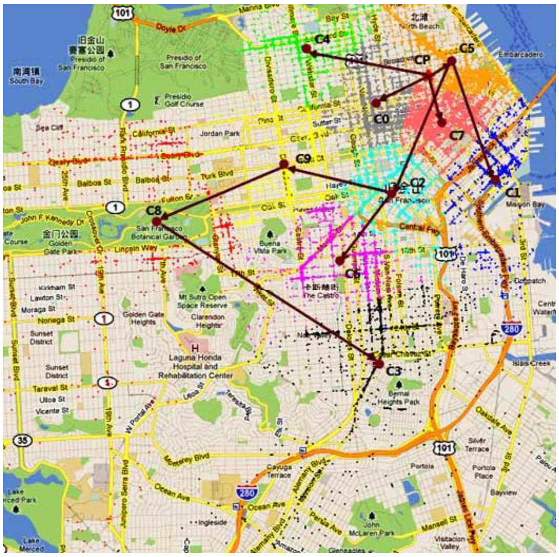


Fig. 2. An example of the pick-up tree

4.2 Results of Pick-Up Tree

In this subsection, we illustrate the output of the pick-up trees with different CP and the number of clusters K. To facilitate the exhibition, we limit the range of map with the latitude in the interval [37.80654, 37.70846] and the longitude in the

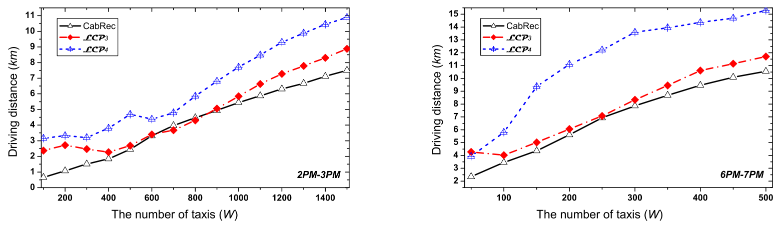
[1] http://www.cs.waikato.ac.nz/ml/weka/

interval [-122.49979, -122.3794]. We then extract all state variance points of 536 cabs during *2PM-3PM* and obtain 25973 potential pick-up points. The current position CP is located at the point (37.7941, -122.4080) which is the China Town in the San Francisco Bay Area. All potential pick-up points are clustered into 10 clusters. The way to estimate $P(C_i)$ of the ith centroid is as follows: finding the adjacent points in the ith cluster for every taxi and computing average difference of time all taxis denoted as $t_{avg}(C_i)$. So, we have $P(C_i) = 1/t_{avg}(C_i)$. Note that two adjacent points implicate the taxi has carried passengers, since the point in each cluster represents state variance from *free* to *occupied*. Table 1 lists the information of the 10 centroids, and the derived pick-up tree is shown in Fig. 2.

In this example, there are 6 routes from CP to the leaf nodes. We set $\alpha = 1.5$ to get the values of cost function by Eq. (6), and then obtain the standard weights of every route by Eq. (7). Table 2 shows the cost and weight of these 6 recommended routes.

4.3 Performance Comparison

Here, we compare the proposed CabRec with $\mathcal{LCP}$ on the driving distance before carrying passengers and time-cost. We design the procedure of our experiment as follows:



(a) 2PM-3PM (b) 6PM-7PM

Fig. 3. A comparison on the average driving distance before carrying passengers

- **Step 1: Pick-Up Points Generation by Clustering.** The state variance points of all taxis during the given time period are clustered into K clusters, and the K centroids are returned as the pick-up points.
- **Step 2: Route Recommendation.** We employ CabRec and $\mathcal{LCP}$ for route recommendation. CabRec utilizes the weighted Round-Robin method to generate recommended routes, while $\mathcal{LCP}$ takes all routes equal and uses the simple Round-Robin method.
- **Step 3: Simulation of Taxis to Pick Up Passengers.** In this step, we assume there are W target taxis around a given current position. Since the time span of the experimental data set is about 30 days, the size of every cluster divided by 30 days is the average number of passengers of each pick-up point every day. After these W target taxis adopt recommended routes

generated by both CabRec and $\mathcal{LCP}$, once a taxi passes a pick-up point having non-zero remaining passengers, the taxi is deemed to be occupied and the number of remaining passenger of that pick-up point should subtract 1.

In the above-mentioned procedure, we are readily to compute the average driving distance of W taxis before carrying passengers. Note that if the taxi does not carry any passenger in the recommended route we extra add $10km$ to its driving distance as penalty. We set $K = 20$ and range W in different scale according to the number of state variance points of different time period. Fig. 3 show the comparison results on two time periods. As can be seen, CabRec works much more better than both $\mathcal{LCP}_3$ and $\mathcal{LCP}_4$, especially when W is large. The reason lies in that since CabRec has estimated the cost of every route and taken it as the weight, more cabs will cruise along with the high-weight routes with the increase of W. In contrast, $\mathcal{LCP}$ takes every route as equal and many taxis head for the routes with few passengers, which reduce the performance dramatically.

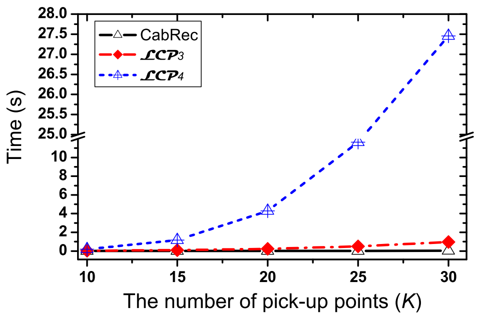


Fig. 4. The comparison on execution time

We also observe the execution time of CabRec and $\mathcal{LCP}$. Fig. 4 shows the execution time with the increase of the number of pick-up points (K centroids). As can be seen, the execution time of $\mathcal{LCP}_4$ goes up sharply. Although the time-cost of $\mathcal{LCP}_3$ is not very high, our CabRec performs even more fast than $\mathcal{LCP}_3$.

To sum up, we have two conclusions from the experiments: (1) The proposed CabRec can successfully decrease the average driving distance of the set of taxis before carrying passengers. (2) The time-cost of our CabRec is also lower than the existing methods.

5 Related Work

With the prevalent of the ubiquitous computing, it is more and more convenient to obtain the information about location, time, trajectory, surrounding, etc. Mobile recommender systems have attracted more attentions. Some research opportunities on mobile recommender systems were discussed in [12]. Quercia et al. have designed a social events recommender system using mobile phone location data [13]. Several kinds of mobile tourist recommender systems have been presented in [2, 3, 4, 5].

In addition, great research efforts have been taken to use GPS trajectories of cabs for analysis. Liu et al.focused on cabdriver operation patterns analysis rather than recommendation based on the large scale cabdrivers' behavior in Shenzhen, China [14]. A mobility-based clustering analysis method is used for identifying hot spots of moving vehicles [15]. Chang et al. have proposed a context-aware prediction model for taxi demand hotspots [16]. Wu et al. investigated service site selection using GIS data [17]. Zheng and Xie et al. have conducted many studies on knowledge discovery from GPS trajectory data in Beijing, China. For instance, the computation method of user similarity based on location history is proposed in [18], friends and locations recommendation method is then studied in [19], and the knowledge extracted from GPS data is also used to solve the congested traffic problems [20, 8]. Chen et al. studied a new problem of searching the k Best Connected Trajectories (k-BCT) by locations from a database, in which context the query is only a small set of locations [21]. Chen et al. also discovered the Most Popular Route(MPR) between two locations by observing the traveling behaviors of many previous users [22]. In this article, we focus on green recommendation for cab drivers, and our experiments utilize the cab traces data set in the San Francisco Bay Area, which is also used by [9, 6].

6 Conclusion and Future Work

Upon taxi trajectories data, this paper proposes a system called CabRec for route recommendation. In CabRec, the state variance points that imply the taxis have carried passengers are clustered, and the centroids are taken as potential pick-up points. Then, a heuristic is employed to construct the pick-up tree which takes the current position as its root node and connect all centroids. A probability model to estimate the weight of every route and the weighted Round-Robin recommendation method for the set of taxis is proposed. Our experimental results on real-world taxi trajectories data set have shown the effectiveness and efficiency of the CabRec.

There are a wealth of research directions that we are currently considering, such as implementing a cab recommender system with dynamic visualization, employing several large-scale data sets, and expanding other recommender applications in the area of intelligent transportation system, and more.

Acknowledgments. This research is supported by National Natural Science Foundation of China (No.61103229, 71072172, and 61003074), Industry Projects in the Jiangsu Science & Technology Pillar Program (No.BE2011198), Jiangsu Provincial Colleges and Universities Outstanding Science & Technology Innovation Team Fund(No.2001013), International Science & Technology Cooperation Program of China (No.2011DFA12910), National Key Technologies R&D

sub Program in 12th five-year-plan (No. SQ2011GX07E03990), Jiangsu Provincial Key Laboratory of Network and Information Security(Southeast University) (No.BM2003201), Natural Science Foundation of Jiangsu Province of China (BK2010373) and Postgraduate Cultivation and Innovation Foundation of Jiangsu Province (CXZZ11_0045).

References

[1] Wang, S., Wu, C.: Application of context-aware and personalized recommendation to implement an adaptive ubiquitous learning system. Expert Systems with Applications (2011)

[2] Abowd, G.D., Atkeson, C.G., Hong, J., Long, S., Kooper, R., Pinkerton, M.: Cyberguide: A mobile context-aware tour guide. Wireless Networks 3(5), 421–433 (1997)

[3] Staab, S., Werthner, H.: Intelligent systems for tourism. IEEE Intelligent Systems 17(6), 53–66 (2002)

[4] Tao, Y., Ding, L., Lin, X., Pei, J.: Distance-based representative skyline. In: Proceedings of the 2009 IEEE International Conference on Data Engineering (ICDE 2009), pp. 892–903 (2009)

[5] Liu, Q., Ge, Y., Li, Z., Chen, E., Xiong, H.: Personalized travel package recommendation. In: IEEE 11th International Conference on Data Mining (ICDM 2011), pp. 407–416 (2011)

[6] Ge, Y., Xiong, H., Liu, C., Zhou, Z.: A taxi driving fraud detection system. In: IEEE 11th International Conference on Data Mining(ICDM 2011), pp. 181–190 (2011)

[7] Yuan, J., Zheng, Y., Zhang, C., Xie, W., Xie, X., Sun, G., Huang, Y.: T-drive: Driving directions based on taxi trajectories. In: Proceedings of the 18th SIGSPATIAL International Conference on Advances in Geographic Information Systems, pp. 99–108 (2010)

[8] Yuan, J., Zheng, Y., Xie, X., Sun, G.: Driving with knowledge from the physical world. In: Proceedings of the 17th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 316–324 (2011)

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

[10] Yuan, Y., Lin, X., Liu, Q., Wang, W., Yu, J., Zhang, Q.: Efficient computation of the skyline cube. In: Proceedings of the 31st International Conference on Very Large Data Bases, pp. 241–252 (2005)

[11] Vincenty, T.: Direct and inverse solutions of geodesics on the ellipsoid with application of nested equations. Survey Review 23(176), 88–93 (1975)

[12] van der Heijden, H., Kotsis, G., Kronsteiner, R.: Mobile recommendation systems for decision making. In: Proceedings of International Conference on Mobile Business (ICMB 2005), pp. 137–143 (2005)

[13] Quercia, D., Lathia, N., Calabrese, F., Lorenzo, G.D., Crowcroft, J.: Recommending social events from mobile phone location data. In: IEEE 10th International Conference on Data Mining (ICDM 2010) (2010)

[14] Liu, L., Andris, C., Ratti, C.: Uncovering cabdrivers' behavior patterns from their digital traces. Computers, Environment and Urban Systems 34(6), 541–548 (2010)

[15] Liu, S., Liu, Y., Ni, L.M., Fan, J., Li, M.: Towards mobility-based clustering. In: Proceedings of the 16th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 919–927 (2010)
[16] Chang, H., Tai, Y., Hsu, J.: Context-aware taxi demand hotspots prediction. International Journal of Business Intelligence and Data Mining 5(1), 3–18 (2010)
[17] Wu, J., Chen, J., Ren, Y.: GIS enabled service site selection: Environmental analysis and beyond. Information Systems Frontier (13), 337–348 (2011)
[18] Li, Q., Zheng, Y., Xie, X., Chen, Y., Liu, W., Ma, W.Y.: Mining user similarity based on location history. In: Proceedings of the 16th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems (2008)
[19] Zheng, Y., Zhang, L., Ma, Z., Xie, X., Ma, W.: Recommending friends and locations based on individual location history. ACM Transactions on the Web (TWEB) 5(1), 5 (2011)
[20] Liu, W., Zheng, Y., Chawla, S., Yuan, J., Xing, X.: Discovering spatio-temporal causal interactions in traffic data streams. In: Proceedings of the 17th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 1010–1018 (2011)
[21] Chen, Z., Shen, H., Zhou, X., Zheng, Y., Xie, X.: Searching trajectories by locations: an efficiency study. In: Proceedings of the 2010 International Conference on Management of Data (SIGMOD 2010), pp. 255–266 (2010)
[22] Chen, Z., Shen, H., Zhou, X.: Discovering popular routes from trajectories. In: Proceedings of the 2009 International Conference on Data Engineering (ICDE 2011), pp. 900–911 (2011)