Available online at www.ijpe-online.com

International Journal of Performability Engineering

Vol. 13, No. 4, July 2017, pp. 530-539
DOI: 10.23940/ijpe.17.04.p20.530539

Research on Destination Prediction for Urban Taxi based on GPS Trajectory

Meng Zhang[a], Yongjian Yang[a,*], Liping Huang[a], Xiaopeng Zhang[b]

[a]*College of Computer Science and Technology, Jilin University, Changchun,China*
[b]*College of Software, Jilin University, Changchun, China*

Abstract

Researching on destination prediction has a particularly important influence on the location-based services' popularization. The traditional destination prediction algorithm is to retrieve the historical trajectory data to find the same trajectory sequences as the query trajectory and then derive the most likely location to be the predicted result. However, due to the limitation of the historical trajectory data, this method has low efficiency and accuracy. Thus, in this paper, we propose the Prediction algorithm based on time (*PBT* algorithm), which considers the influence of the factor of time on destination prediction. Experiments based on real data show that in terms of destination prediction, the *PBT* algorithm not only alleviates the limitation of the historical data in the traditional algorithm to make the results more realistic, but also is more effective.

Keywords: Smart City; Destination Prediction; GPS trajectory; Based on Time Prediction; Data Analysis

(Submitted on February 14, 2017; Revised on May 3, 2017; Accepted on June 15, 2017)

1. Introduction

Intelligent Transportation System (*ITS*) is a set of applications aimed at providing innovative services relating to different modes of transport and traffic management while enabling users to be better informed and make use of transport services in a smarter, more coordinated way. Destination prediction is a popular research direction in *ITS*. It predicts the future geographic location of a person's by analyzing his existing and historical travel information. Thus, in addition to the existing current location or the location of the user manual input that can be used to provide location-based service (*LBS*), a large number of predicting locations can also be used for providing intelligent navigation service according to forecast results that are based on prediction destination advertising. It further expands the coverage of *LBS*. In the study of smart cities, there are many prediction methods. Some of these methods not only consider the users' travel information but also increase the use of other auxiliary information, such as POI. For example, [1] proposed a destination prediction model based on historical data, contextual knowledge, and spatial conceptual maps to greatly reduce the learning cycle and improve the efficiency of prediction; [2] captured individual patterns using a next place prediction algorithm that built different Dynamic Bayesian Network (*DBN*) models based on mobile phone cellular data. An interesting finding in his study is that that planning areas with higher population had higher predictability; and [3] used destination information from Twitter to predict users' destination. Other methods use the historical GPS data of vehicles to complete prediction. For example, [4] proposed a novel model T-DesP that uses GPS data to predict destination; [5] used taxis' GPS data to model taxis' routing behaviors to predict arriving locations of taxis and recommend convenient pick-up spots for passengers; [6] designed a method of predicting taxi

* Corresponding author.
E-mail address: yyj@jlu.edu.cn.

trip time by analyzing the historical trip data; and Chen et al. [7] collected real trajectory data and used the tree structure to represent movement historical patterns, then gradually reduced the tree to match the current trajectory (this method can reached about 80% accuracy in destination prediction). In real life, the places people choose to go are closely related to the date; for example, during the holidays, sites of entertainment are more likely to be visited, while during the weekdays, working places are chosen more often even if the weather is not ideal. The choice of destination is also related to the time interval of a day. Thus, it is very important to take the factor of time into consideration for the travel destination prediction.

In this paper, the taxis' GPS trajectory data and map information are used. First, we partition the region of all GPS trajectories cover into grids of the same size and then number them sequentially. The trajectories can thus be expressed in a series of numbers, as shown in Figure 1, where T^1={1,2,6,10,11}, T^2={1,2,6,10,11}, and T^1 is equal to T^2. In general, destination prediction algorithm retrieves the historical trajectory data to find the trajectory sequences that match the query trajectory and then regards the destination of the retrieved trajectory sequence as the result of destination prediction. For example, in Figure 1, assuming that the historical trajectory data is {T^1, T^2}, the query trajectory is T^3; using the traditional algorithm to find in conformity with the T^3 is T^1 (or T^2), then the destination of T^1 or (T^2)-region 11, as the predicted destination. However, if the query trajectory is T^4, the traditional algorithm cannot predict this. Thus, the "data sparse" problem brings out the low prediction efficiency.

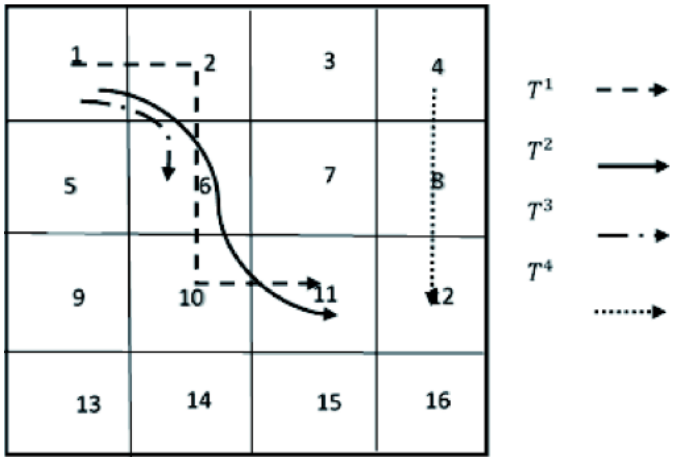


Figure1. Map grids

The *PBT* algorithm considers the influence of time factor on destination prediction and is based on the Sub-Trajectory Synthesis algorithm proposed by Xue et al. [8]. It divides time into weekdays and weekends/holidays, and the time of a day is divided into four periods. Destination prediction can thereby be made for different time periods. We make the following contributions in this paper:

We identify the influencing factors of the trajectory destination prediction, including the day of the week and the time period of a day.

We propose a time feature based on the Markov model for trajectory destination prediction. By establishing the Markov model in different time periods, a time-dependent matrix sequence of state transition probability is built.

We conduct extensive experiments using real taxi trajectory data to investigate the effectiveness of the proposed *PBT* algorithm and compare it to that of the Sub-Trajectory Synthesis algorithm, showing that the prediction results of *PBT* algorithm are more effective.

2. Markov Model and Sub-Trajectory Synthesis Algorithm

2.1. Markov Model

A Markov chain is a process that holds the Markov property and has a countable state space. The formal definition of a Markov chain is that given the present state, the future and the past states are independent. This can be expressed formally as:

$$P_r\{X_{n+1} = x | X_1 = x_1, X_2 = x_2, \ldots, X_n = x_n\} = P_r\{X_{n+1} = x | X_n = x_n\} \quad (1)$$

Where $\{X_1, \ldots X_n\}$ are the states of the Markov process. The Markov model has been widely used. Pang [9] proposed a new Markov model of reliability assurance and failure prediction based on computer network technology in order to ensure the reliable operation of complex products and detect the delitescent faults in earliest time; Wen and Gao [10] forecasted the variation tendency of energy structure in the transport sector by Markov transition matrix (the result of the analyses showed that the proportion of kerosene, natural gas and electricity will increase rapidly in the coming decades, while the other fossil-energy consumption will decrease); and Shi et al. [11] proposed a method of combining user clustering with the Markov chain to predict the next scenic spot a user would browse on a tourism website (the results indicated the proposed method was effective). Thus, the Markov model has a positive impact on the prediction algorithm. A simple Markov model is illustrated in Figure 2.

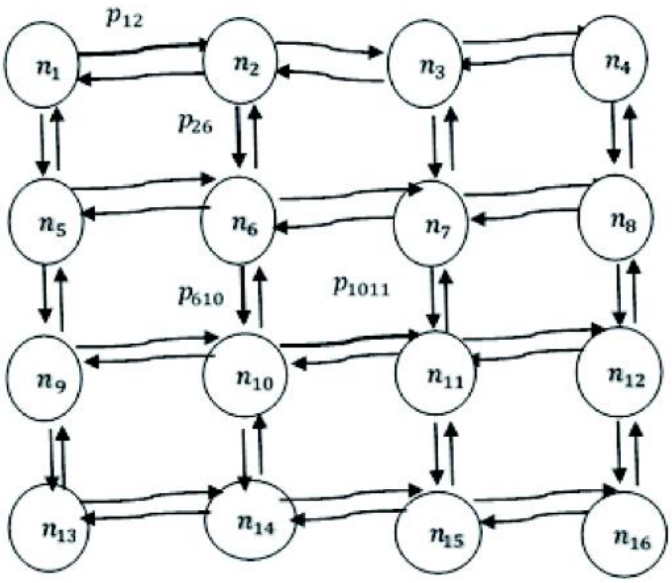


Figure 2. 4*4 Markov model

2.2. Sub-Trajectory Synthesis Algorithm

The Sub-Trajectory Synthesis algorithm's pivotal idea is decomposing the trajectories into sub-trajectories that consist of two adjacent regions, based on the correlation of regions. The sub-trajectories can be combined randomly to form the synthesized trajectories. Therefore, the algorithm increases the size of historical trajectory data and solves the "data sparse" problem faced by traditional algorithms. Its steps are as follows:

- Constructing the Markov Model: A Markov model is constructed by associating the state to each region in the grid map in order to fully leverage the information of historical trajectories. In Figure 2, the constructed Markov model is based on the 4*4 grid map in Figure 1.
- Computing the Transition Matrix: The two-dimensional n^2*n^2 transition matrix $\boldsymbol{M}$, where one dimension corresponds to the region of current state and the other dimension corresponds to the next state, is calculated after the n^2*n^2 Markov model was constructed. The following matrix 16*16 is corresponding to the Markov model in Figure 2.

$$M = \begin{pmatrix} 0 & p_{12} & 0 & 0 & p_{15} & 0 & 0 & \cdots & \cdots & \cdots & \cdots & \cdots & 0 \\ p_{21} & 0 & p_{23} & 0 & 0 & p_{26} & 0 & \cdots & \cdots & \cdots & \cdots & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & \cdots & \cdots & \cdots & \cdots & \cdots & 0 & 0 & p_{1511} & 0 & 0 & p_{1514} & 0 & p_{1516} \\ 0 & \cdots & \cdots & \cdots & \cdots & \cdots & 0 & 0 & 0 & p_{1612} & 0 & 0 & p_{1615} & 0 \end{pmatrix}$$

The elements in the matrix M are the transition probability between adjacent regions:

$$p_{ij} = {|T_{i,j}|}/{|T_i|} \quad (2)$$

In the equation above, $|T_i|$ is the number of the trajectories containing region i, $|T_{i,j}|$ is the number of the trajectories containing the sequence$\{i,j\}$. Supposing there are n regions, p_{ij}must be calculated from 1 to n by (1) then saved in a n^2*n^2 state transition probability matrix $\boldsymbol{M}$. The probability of one region transferred to its nonadjacent region can also be obtained:

$$p_{i\to k} = \sum_{x=L_{i\to k}}^{L_{i\to j}^r} M_{ik}^x \quad (3)$$

$L_{i\to j}$ is the least number of steps from n_i to n_k, and $M^x$is the result of raising M to the x power. Additionally, the following is defined:

$$L_{i\to j}^r = \lceil 1.2 \times L_{i\to j} \rceil \quad (4)$$

- Bayesian Inference Framework for Predicting Destination: the probability of a region n_j being the destination, conditioning on the query trajectory T^q is computed using Bayer's rule as:

$$P(d \in n_j | T^q) = \frac{P(T^q|d\in n_j)P(d\in n_j)}{\sum_{k=1}^{n^2} P(T^q|d\in n_k)P(d\in n_k)} \quad (5)$$

Where the prior probability $\mathrm{P}(\mathrm{d} \in n_j)$can be computed, formally:

$$P(d \in n_j) = \frac{|T_{d\in n_j}|}{|D|} \quad (6)$$

Where $|D|$ is the cardinality of the historical trajectory dataset, $|T_{\mathrm{d}\in n_j}|$is the number of trajectories in D that terminates at region n_j, n is the granularity of the grids. The posterior probability that denotes the probability under the condition of region n_j is the destination, from the starting region $n_o$to the current region n_c via the trajectory T^q, is computed as follows:

$$P(T^q | d \in n_j) = \frac{P(T^q)\cdot p_{c\to j}}{p_{o\to j}} \quad (7)$$

Where the path probability that denotes a user travelling from one region to another via a specific path. The value of the path probability can be obtained by multiplying the transition probabilities between all pairs of regions in this partial path T^q, as follows:

$$P(T^q) = P(T^q_{1,2,..k}) = \prod_{i=1}^{k} p_{i(i+1)} \quad (8)$$

The last step of the algorithm compares the value of (4) for each region. If it has the maximum value, it will be the result of destination prediction.

3. Destination Prediction based on Time

In order to improve the effectiveness of destination prediction, we propose the *PBT* algorithm, which considers the influence of time factor on destination prediction. Figure 3 is the flow chart for the *PBT* algorithm, followed by a detailed description.

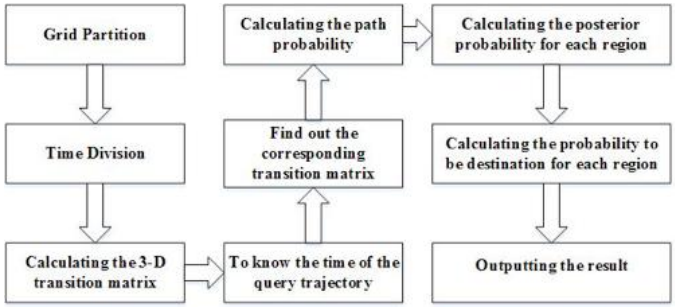


Figure 3. PBT Algorithm Flow Chart

3.1. Grid Partition

According to the range error (*RE*) between the result of destination prediction and the real destination to partition the region of all GPS trajectories cover into grids of same size. The principle of selecting grid granularity is to choose the minimum range error, formally:

$$P(T^q) = P\left(T^q_{1,2,..k}\right) = \prod_{i=1}^{k} p_{i(i+1)} \quad (9)$$

Where floor(x) denotes the biggest integer no greater than x, i and j are the ID of grids, n is grid granularity, and *lat* and *lon* are the length and width of a grid. When we experimented with n=20, 25, 30; the trip completed percentages were 30, 50, 70. The results in Figure 4 demonstrate that each curve is increasing, which shows that with an increase in grid granularity, the range error will also increase, and with an increase in percentage of completed trips, the range error will decrease. Thus, the optimal grid granularity for our training dataset is selected to be 20 according to the global minimum point in Figure 4.

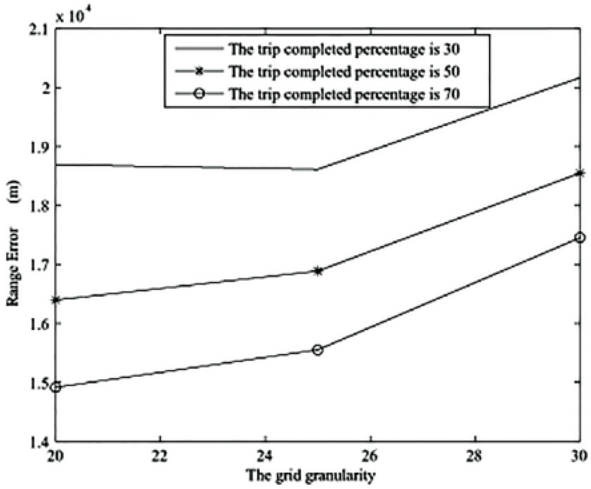
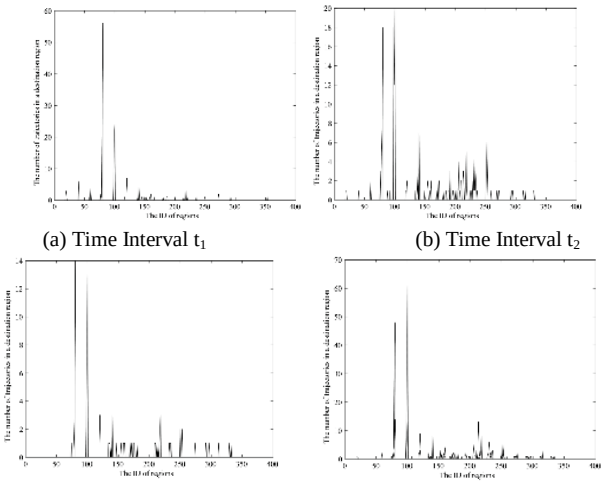


Figure 4. The relation between grid granularity and range error

3.2. Time Period Division

We divided a day into four periods, as follows: t_1- morning peak time, from 6AM to 10AM; t_2 - work time, from 10AM to 5PM; t_3 - evening peak time, from 5PM to 8PM; and t_4 - casual time, from 8PM to 6AM. Figure 5 shows the quantities distribution of the trajectories' destination in all regions on the basic of their starting region is the same. For example, during the time interval t_1 there are some users that choose the regions 180~200 as their destinations, while during the time interval t_3 nobody goes to the same regions. Thus, the temporal information of history trajectories plays an important role in choosing destinations.

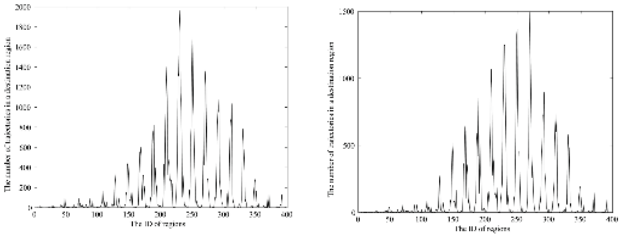


(a) Time Interval t_1 (b) Time Interval t_2

(c) Time Interval t_3 (d) Time Interval t_4

Figure 5. The quantities distribution of the trajectories' destination in all regions

Furthermore, depending on the date, we divided time into weekdays and holidays/weekends. As shown in Figure 6, during the same time interval on weekdays or holidays, users choose the different destinations. For example, during the time interval t_1 on holidays the probability of the regions 150~230 to be chosen as destinations is higher than the regions 300~320, while this is the contrary on weekdays. It also proves that the time factor is related to chosen destination.



(a) Time Interval t_1 on weekday (b) Time Interval t_4 on holiday

Figure 6. The quantities distribution of the trajectories on different dates

3.3. Time Feature Based Markov Model

From section B, it is confirmed that destination prediction is associated with time. Thus, we construct the Markov model according to time. Because the experimental data is from April 2015, we define April 7 to 11 as weekdays and April 5 to 6 as holidays/weekends. Then, we partition 30 days into 120 intervals numbered from 1 to 120. Because users' travel patterns are essentially the same during the same time interval on weekdays or holidays/weekends, we use the "Synthesis time interval," which is a set of intervals satisfying (10), to experiment in order to solve the "data sparse" problem.

$$mod(|t_m - t_n|, 4) = 0 \qquad m, n \in (1,2,3, \dots, 120) \tag{10}$$

Where t_m and t_n represent the time interval with numbers m and n, respectively. Finally, we have obtained eight Markov models and matrixes according to different time intervals.

- Solving the Transition Matrix in Different Time Intervals: under these circumstances, the elements of the transition matrix are p_{ijy}^h , p_{ijy}^w denoting the probability from region i transferred to j in the time interval y on holidays/weekends or weekdays.

$$p_{ijy}^h = {C_{iy}^h}/{C_{ijy}^h} \tag{11}$$

$$p_{ijy}^w = {C_{iy}^w}/{C_{ijy}^w} \tag{12}$$

Where we can use the equation (13), (14) to obtain C_{iy}^h、C_{iy}^w which denote the number of trajectories containing region i in the time interval y on holidays/weekends or weekdays. Similarly, in the corresponding time interval, the number of trajectories containing the sequence { i, j}−C_{ijy}^h、C_{ijy}^w, can be calculated.

$$C_{iy}^h = [X_y^h * p_{iy}^h] \tag{13}$$

$$C_{iy}^w = [X_y^w * p_{iy}^w] \tag{14}$$

In the aforementioned equations, X_y^h and X_y^w are the sum of trajectories in the time interval j on holidays/weekends or weekdays. p_{iy}^h and p_{iy}^w are the average probability of a trajectory containing the region i in the corresponding time interval:

$$p_{iy}^h = \frac{\sum_1^{H_h} p_i^{hy}}{H_h} \tag{15}$$

$$p_{iy}^{w} = \frac{\sum_{1}^{H_w} p_i^{wy}}{H_w} \tag{16}$$

Where H_h and H_w denote the number of holidays/weekends and weekdays. In addition, p_i^{hj} and p_i^{wj} are the probability of a trajectory containing the region i in the same time interval on different dates:

$$p_i^{hy} = \frac{c_{iy}}{c_y} \tag{17}$$

$$p_i^{wy} = \frac{c_{iy}}{c_y} \tag{18}$$

Wherec_{iy} and c_y respectively denote the number of trajectories and the sum of trajectories that contain region i in corresponding time intervals.

- Constructing Three Dimensional Matrix: as shown in Figure 7, the 3-D transition Matrix is constructed, with the abscissa being time. It is corresponding to the Markov model in Figure 2. On the right of Figure 7 is part of a transition matrix M during some one time interval. In this way, the time of the query trajectory T^q can be used to find the corresponding transition matrix, and the destination prediction can be outputted after using (3)-(8).

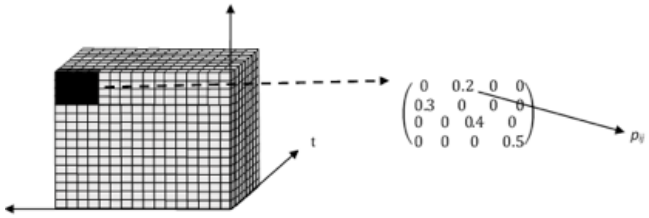


Figure 7. A 3-D transition Matrix

The detailed steps of the *PBT* algorithm are described as follows: first, partition time into time intervals on the basis of (10) and build the Markov model on the different time intervals. Then, construct a 3-D transition matrix after calculating the transition matrices in all time intervals by (11)-(17); at the same time, use the time of the query trajectory to make sure which transition matrix should be used for prediction, while also computing the path probability and the regional posterior probability based on (8), (7). Finally, compare the value of (5) to choose the region with the maximum as the result of destination prediction.

4. Experiment

In this section, we conduct an extensive experimental study to evaluate the performance of our *PBT* algorithm. There are several different parameters to compare and evaluate differences between the *PBT* algorithm and the Sub-Trajectory Synthesis algorithm.

4.1. Dataset

We use real-world large scale trajectory data of taxis in Shanghai during April 2015. It contains 38,000 trajectories in a day. We randomly pick 4,000 trajectories on April 25th (Sat.) and 27th (Mon.) to be the query trajectories, while the remaining trajectories are used as training data. The experiment area is the longitude 121.175~121.775, latitude 30.960~31.410 in Shanghai with 20 * 20 grids.

4.2. Evaluation of Effectiveness

4.2.1. Range Error

Range error reflects the error of the prediction results and the real destination, and it is a suitable parameter to evaluate the effectiveness of the algorithm. We used (9) to compare the range error between the *PBT* algorithm and Sub-Trajectory Synthesis algorithm. As shown in Figure 8, the Range error of the *PBT* algorithm is smaller than that of the Sub-Trajectory Synthesis algorithm under the same conditions.

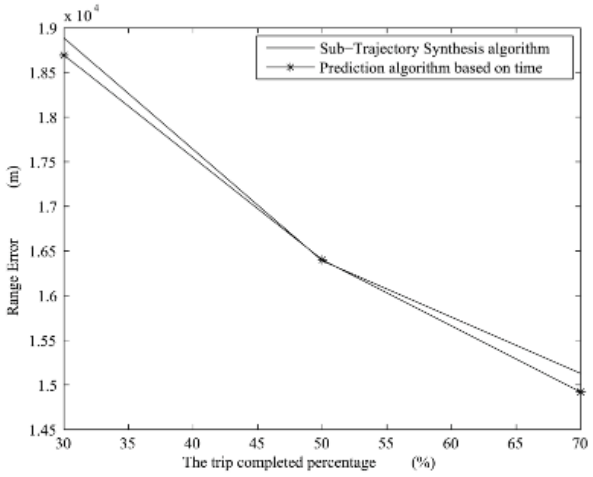


Figure 8. Range Error

4.2.2. Coverage of the Accurate Prediction

If the results of destination prediction are the same as those of the real destination, it is called accurate prediction. In Figure 9, it is proven that the *PBT* algorithm is better than the Sub-Trajectory Synthesis algorithm.

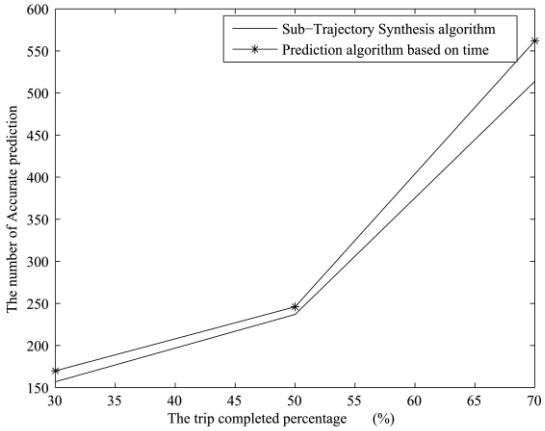


Figure 9. Coverage of the Accurate Prediction

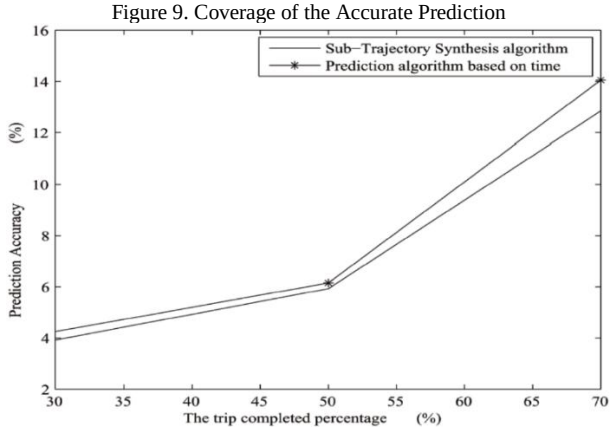


Figure 10. Prediction Accuracy

4.2.3. Accuracy

$$P_{Acc}=F_{accurate}/F_{all} \quad (19)$$

Where $F_{accurate}$ is the number of trajectories that can be predicted accurately, and F_{all} is the sum of the trajectories in the test dataset. The results presented in Figure 10 clearly show that, with the same percentage of completed trips, the *PBT* algorithm has higher accuracy. Additionally, the larger the percentage of completed trips, the higher the prediction accuracy will be.

4.2.4. Coverage

Coverage denotes the number of trajectories that can be predicted. Figure 11 shows that the *PBT* algorithm and Sub-Trajectory Synthesis algorithm consistently predict destinations for almost all trajectories in the experiments, and it illustrates that the *PBT* algorithm successfully solved the "data sparse" problem as well.

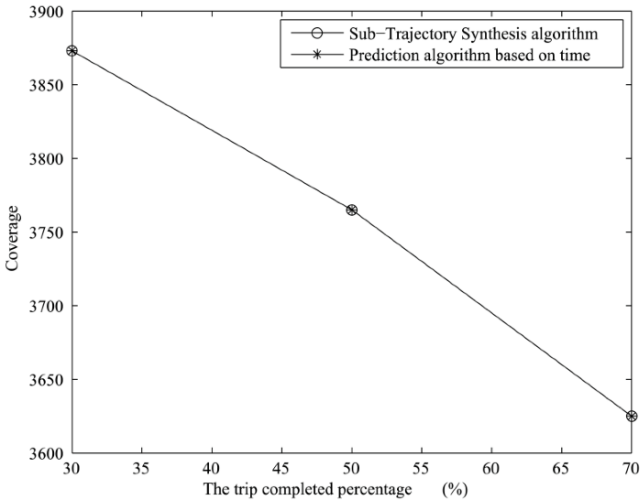


Figure 11. Coverage

5. Conclusions

In this paper, we have proposed the *PBT* algorithm, which considers the influence of time factor on destination prediction and is based on the Sub-Trajectory Synthesis algorithm. This process is formulated based on a three-dimensional transition matrix. Experiments with real GPS data have shown that the *PBT* algorithm predicts destinations for nearly all trajectories, thereby successfully addressing the "data sparse" problem. By considering the time factor, the *PBT* algorithm makes more accurate prediction results than the Sub-Trajectory Synthesis algorithm does. In the best case scenario, the range error of the *PBT* algorithm is less than that of the Sub-Trajectory Synthesis algorithm by over 200 meters. Additionally, the *PBT* algorithm improves accuracy by an average of 1 percent. Thus, it is proven that the *PBT* algorithm is more effective than the Sub-Trajectory Synthesis algorithm.

Acknowledgements

This research has been supported partly by the National Nature Science Foundation of China under Grand no. 61272412 and Jilin Province Science and Technology Development Program under Grant no. 20160204021GX.

References

1. Nadembega, A., Taleb, T., Hafid, A.(2012) "A Destination Prediction Model based on historical data, contextual knowledge and spatial conceptual maps", *IEEE International Conference on Communications*, pp.1416--1420.
2. Dash, M., Koo, K. K., Krishnaswamy, S. P., Jin, Y., Shi-Nash, A.(2016) "Visualize People's Mobility-Both individually and Collectively-Using Mobile Phone Cellular Data", *17th IEEE International Conference on Mobile Data Management*, Vol. 1, pp. 341--344.
3. Shinmura, T., Zhu, D., Ota, J., Fukazawa, Y.(2014) "Destination prediction considering both tweet contents and location transition hitstory", *Seventh International Conference on Mobile Computing and Ubiquitous Networking*, pp. 95--96.
4. Li, X., Li, M., Gong, Y. J., Zhang, X. L., Yin, J.(2016) "T-DesP: Destination Prediction Based on Big Trajectory Data", *IEEE Transactions on Intelligent Transportation Systems*, Vol. 17, pp.2344--2354.
5. Jin, L., Han, M., Liu, G.,Feng, L.(2014) "Detecting Cruising Flagged Taxis' Passenger-Refusal Behaviors Using Traffic Data and Crowdsourcing", *11th Intl Conf on Ubiquitous Intelligence and Computing and 11th Intl Conf on Autonomic and Trusted Computing, and 14th Intl Conf on Scalable Computing and Communications and Its Associated Workshops (UTC-ATC-ScalCom)*, pp. 18--25.
6. Singh, A. D., Wu, W., Xiang, S., Krishnaswamy, S.(2015) "Taxi trip time prediction using similar trips and road network dat", *2015 IEEE International Conference on Big Data*, pp. 2892--2894.

7. Chen, L., Lv, M., Chen, G.(2010) "A system for destination and future route prediction based on trajectory mining", *Pervasive and Mobile Computing*, Vol. 6, pp.657--676.
8. Xue, A. Y., Zhang, R., Zheng, Y., Xie, X., Huang, J., Xu, Z.(2013) "Destination prediction by sub-trajectory synthesis and privacy protection against such prediction", *29th International Conference on Data Engineering (ICDE)*, pp.254-265.
9. Pang, J.(2015) "A new Markov model of reliability assurance and failure prediction using network technology", *4th International Conference on Computer Science and Network Technology*, Vol. 1, pp. 776--780.
10. Wen, L., Gao, Q.(2014) "Research on the Feasibility of the Markov Prediction Model on Energy Consumption". *Journal of Information and Computational Science*, Vol. 11, pp.3149--3155.
11. Shi, Y., Wen, Y., Fan, Z., Miao, Y.(2013) "Predicting the next scenic spot a user will browse on a tourism website based on markov prediction model", *25th International Conference on Tools with Artificial Intelligence*, pp. 195--200.