Discovering Trip Patterns from Incomplete Passenger Trajectories for Inter-zonal Bus Line Planning

Zhaoyang Wang[1,2(✉)], Beihong Jin[1,2], Fusang Zhang[1,2],
Ruiyang Yang[1,2], and Qiang Ji[1,2]

[1] State Key Laboratory of Computer Sciences, Institute of Software,
Chinese Academy of Sciences, Beijing, China
jbh@octaix.iscas.ac.cn
[2] University of Chinese Academy of Sciences, Beijing, China

Abstract. Collecting the trajectories occurring in the city and mining the patterns implied in the trajectories can support the ITS (Intelligent Transportation System) applications and foster the development of smart cities. For improving the operations of inter-zonal buses in the cities, we define a new trip pattern, i.e., frequent bus passenger trip patterns for bus lines (FBPT4BL patterns in short). We utilize the passenger trajectories from bus smart card data and propose a two-phase approach to mine FBPT4BL patterns and then recommend inter-zonal bus lines. We conduct extensive experiments on the real data from the Beijing Public Transport Group. By comparing the experimental results with the actual operation of inter-zonal buses at the Beijing Public Transport Group, we verify the validity of our proposed method.

1 Introduction

With the popularity of smart devices and wireless networks, a large number of digital trajectories in everyday life from different objects (such as persons or vehicles) can be collected. Mining these trajectories gives us the opportunities to discover new knowledge or create new value.

Trip patterns are a kind of frequent sequence pattern. They can be loosely defined as a class of frequently-occurring trajectories that appear in the same spatial positions at the same periods. In real life, trip patterns can be found almost everywhere. For example, a student's behaviors in a university often follow some pattern: he or she often moves in the morning from the dormitory to the classroom, and then gets to the canteen; in the afternoon departs for the playground from the library and then heads for the canteen. Similarly, tourists often follow one of several kinds of travelling routes. As we know, office workers often take the buses of the same line to go to their companies or go home during the weekdays. The vehicles in logistics companies often deliver the goods together. Their trajectories also contain the trip patterns. Further, discovering trip patterns is a constituent part of many applications, which include transportation planning, logistics optimization, friend recommendation, etc.

We note that while mining the trip patterns, the key step is to determine whether a trajectory contains a trip pattern. A variety of criteria can be made, depending on the definition of inclusion relation of the spatial and temporal attributes. However, the inclusion relation on these two dimensions will be completely different in various applications. It leads to the high diversity and complexity in mining trip patterns. Meanwhile, these trajectory data that have been obtained may suffer the problem of incomplete information. For example, the data from bus smart cards which are a specific kind of public transportation card, only recording the bus trajectories, belong to incomplete trajectories. Because we only know the boarding and disembarking times of these trajectories, and do not know the exact times that the passengers pass the intermediate bus stations along the trajectories.

At present, the data from bus smart cards are often used to analyze travel behaviors of individuals or crowds. As far as we know, they have not been used for inter-zonal bus planning and real-time bus scheduling. We think that the data from bus smart cards reflect the public`s travel needs and they are an important basis for bus scheduling. Because of this, this paper attempts to mine a specific trip pattern from the data of bus smart cards and then suggests some inter-zonal bus lines.

The main contribution of this paper is that we propose a solution which utilizes the data of public smart cards for planning inter-zonal buses. In the proposed solution, we first distinguish the commuters by clustering bus passengers. Then, we mine frequent bus passenger trip patterns for bus lines (FBPT4BL patterns in short) and determine some inter-zonal bus lines. We also conduct extensive experiments based on the public smart card data from the Beijing Public Transport Group. By comparing the experimental results with the actual operation of inter-zonal bus at the Beijing Public Transport Group, we verify the validity of our proposed method.

The rest of this paper is organized as follows. First, we review the related work in Sect. 2. Then, we define the problem formally in Sect. 3. Next, we describe the process of mining FBPT4BL patterns from the public smart card data in Sect. 4. In Sect. 5, we give the experimental evaluation with real data as input. Finally, we conclude our paper in Sect. 6.

2   Related Work

We first review the previous methods of mining trip patterns. Next, we investigate the existing work on public transportation card data. Finally, we discuss the ways of optimizing bus lines and public transport scheduling with different data sources.

Trip Pattern Mining. A trip pattern denotes in essence a frequent path. The kernel issue in mining trip patterns is how to organize and search the trajectory information in an effective and efficient way [1, 2]. For example, [1] adopts a sequence instead of a single scalar to describe the frequency of a path so as to avoid the influences of infrequent edges and the number of constituent edges of a path, and construct the footmark graph and the corresponding footmark indexes. Next, [1] proposes a dynamic programming algorithm for path selection and applies an improved Bellman-Ford algorithm for mining frequent paths.

On the other hand, different application requirements derive new variants of trip patterns [3–5]. For example, the T pattern is proposed in [3] for obtaining the classification of persons. Moreover, [3] adopts a density-based clustering method to mine ROIs (Regions-of-Interest) from history trajectories, then mines T patterns in an incremental way from ROI sequences, and finally obtains the classification of persons on the basis of T patterns. For another example, the gathering pattern is proposed in [4] whose goal is to identify various group events, such as celebrations, parades, protests, traffic jams and so on. In [4], objects' trajectories are clustered into several crowds. Next, the Hausdorff distance is used for measuring the distance between crowds, and neighboring crowds comprise the group. And then, the lower bounds for the number of crowds in one group and the number of moving objects appearing in a group are given for mining gathering patterns.

Smart Card Data Analysis. The existing work on smart card data mainly includes the analyses of individual travel behaviors and collective behaviors.

As an example of mining individual behaviors, [6] adopts the bag-of-words model to build a passenger profile by his or her trip information in the smart card data, and employs the hierarchical clustering to get groups of passengers. [6] uses the results of clustering as input of a three-layer back-propagation neural network to generate the predictions. The results show that passengers have their inner mobility patterns and usually hold the same patterns as the ones in the same group. However, for mining the patterns implied in the collective behaviors, the starting point is in general to classify the passengers according to the smart card data. [7] divides passengers into two groups: extreme travelers and non-extreme travelers, and then employs different methods to analyze the mobility and stability of two kinds of travelers. [7] concludes that the stability of extreme traveler's travel pattern cannot last for a long time and the non-extreme shows high regularity. [8] classifies passengers into four types and then conducts the statistical analyses for each passenger type.

Bus Line Planning and Scheduling. Planning bus lines and scheduling buses on the basis of big data is an emerging topic in intelligent transportation systems.

[9] utilizes smart card data and GPS data of buses to calculate the passenger density of a bus service, so as to verify the validity of time arrangements and assess the selections of bus scheduling schemes. [10] focuses on performing the bus scheduling of urban bus lines and proposes an improved multi-objective genetic algorithm and obtains multiple Pareto solutions. In order to meet individual travel needs, [11] proposes a flexible mini-shuttle like transportation system called flexi, to establish a flexible bus scheduling by exploring the taxi trajectories. The authors first adopt a hierarchical clustering algorithm to get the frequent taxi trajectory clustering, named hot line. Then, they construct a directed acyclic graph of hot lines, and mine bus lines and bus schedule strategies depending on the connectivity between hot lines.

Compared with the existing work, our work has a different goal, that is, inter-zonal bus line planning. For this goal, we propose a new variant of the trip pattern named the FBPT4BL pattern, and design a two-phase approach to mining this pattern from the smart card data.

3   Problem Definition

Generally, a trajectory of an object refers to a sequence of time-stamped locations, representing the traces collected by some mobile wireless devices. As far as a bus trip is concerned, the following information can be obtained from bus smart card data, i.e., the bus card ID, the bus ID, the bus line ID, the boarding time of a passenger, the disembarking time of a passenger, the GPS data of the boarding station, and the GPS data of the disembarking station. These data reflect the spatio-temporal characteristics of the boarding and disembarking behaviors. With the aid of bus line details, the station information which a bus passenger passes can be supplemented, but the times of reaching these passing stations are still unknown. So the trajectories from smart card data are incomplete ones.

   For clearly defining the problem to be solved, the concepts below are given at first.

Definition 1 (Bus Trajectory): A bus trajectory is a pair $(T, A) = <(b_0, t_0), b_1, \ldots, (b_n, t_1)>$. Here, T denotes a sequential sequence of bus stations, that is, $T = <b_0, b_1, \ldots, b_n>$, where b_0 stands for the boarding station, b_n is the disembarking station, and the rest are the stations a passenger passes through. However, $A = [t_0, t_1]$, where t_0 and t_1 stand for the boarding and disembarking times, respectively.

Definition 2 (Bus Passenger Trip pattern, BPT pattern): A trip pattern, called BPT pattern, is a pair (P, τ), where $P = <b_0', b_1', \cdots, b_n'>$ denotes a sequential sequence of bus stations which a bus trajectory, in whole or part, passes through, and τ is the time interval that P goes through.

   For deciding whether a trajectory is included in a BPT pattern, we have to define a binary relation between a trajectory and a BPT pattern.

Definition 3 (the inclusion between a bus trajectory and a BPT pattern, $\subseteq$): For any bus trajectory $(T, A) = <(b_0, t_0), b_1, \ldots, (b_m, t_1)>$ and a BPT pattern (P, τ), where $P = <b_0', b_1', \cdots, b_n'>$. We say that (T, A) is included in (P, τ), denoted as $(T, A) \subseteq (P, \tau)$, if and only if the following conditions are satisfied:

1. $m \leq n$ and $t_0 \in \tau$;
2. if $m = n$, then $T = P$;
3. if $m < n$, then $\exists x \geq 0$, $s.t.$ $<b_0, b_1, \ldots, b_m> = <b_x', b_{x+1}', \ldots, b_{x+m}'>$, where $0 \leq x$, $x + m \leq n$.

Definition 4 (Frequent Bus Passenger Trip pattern, FBPT pattern): Given a bus trajectory set D, the minimum support threshold s_{min}, and the time interval $\tau_{interval}$, mining a FBPT pattern is to find a BPT pattern (P, τ) that includes at least s_{min} trajectories and whose τ is within $\tau_{interval}$. That is, for any BPT pattern (P, τ), such that $support_D((P, \tau)) \geq s_{min}$, where $support_D((P, \tau))$ stands for the number of trajectories $(T_k, A_k) \in D$, $k = 1, 2, 3 \cdots$, that satisfy $(T_k, A_k) \subseteq (P, \tau)$ and τ is within $\tau_{interval}$.

   Note that in *Definition* 4, different $support_D((P, \tau))$ will lead to the different definitions of the frequent trip pattern.

   In the paper, we explore bus smart card data to discover the commuters and mine one kind of specific frequent BPT pattern, named FBPT4BL patterns: i.e., frequent BPT patterns for bus lines. Now, we give the definition of mining FBPT4BL patterns.

Definition 5 (Frequent Bus Passenger Trip pattern for Bus Lines, FBPT4BL pattern): Given the bus trajectory set of commuters $D_{commuter}$, the minimum busload threshold between adjacent stations ps_{min}, the minimum trip length r, and the time interval $\tau_{interzonal}$, mining a FBPT4BL pattern is to find a BPT pattern $(P, \tau_{commuter})$ in $\tau_{interzonal}$ whose trip length is longer than r and *score* is higher than ps_{min}. That is, for any BPT pattern $(P, \tau_{commuter})$, such that $Score_{D_{Commuter}}((P, \tau_{commuter})) \geq ps_{min}$ and $length(P) \geq r$, where $length(P)$ is the total length of P, $\tau_{commuter}$ is within $\tau_{interzonal}$, and $Score_{D_{Commuter}}$ stands for the score of $(P, \tau_{commuter})$, which is computed as follows:

$$Score_{D_{Commuter}}((P, \tau_{commuter})) = ITFQ_{D_{Commuter}}(P, \tau_{commuter})/(length(P) - 1) \quad (1)$$

$$\begin{aligned} ITFQ_{D_{Commuter}}((P, \tau_{commuter})) = \{count((T,A))|(T,A) \\ \in D_{commuter} \wedge (T,A) \subseteq (P, \tau_{commuter})\} \end{aligned} \quad (2)$$

where *ITFQ* denotes the busload of $(P, \tau_{commuter})$.

4   Mining Frequent Trip Patterns for Inter-zonal Bus Lines

In this section, we propose an approach to mining commuter-based frequent trip pattern (the MCFTP approach in short) for recommending inter-zonal bus lines. The MCFTP approach consists of two phases: commuter recognition and inter-zonal bus mining. In the first phase, the DBSCAN algorithm is applied for passenger classification. In the second phase, with the support of a forest structure, an incremental data mining algorithm is designed for mining FBPT4BL patterns from the commuter trajectories. As a result, top-k inter-zonal bus lines can be obtained.

4.1   Mining Commuters

For the perspective of bus line planning, inter-zonal bus lines should satisfy the travel needs from commuters. Thus, we employ the DBSCAN algorithm and identify commuters by clustering the trajectories on spatial and temporal features. In spatial aspect, we cluster on boarding and disembarking stations respectively to discover frequent OD (Origin-Destination) pairs and then find frequent trajectories. In temporal aspect, we also use the density clustering algorithm DBSCAN to get the habitual time feature.

Mining Frequent OD Pairs. For mining a passenger's frequent boarding stations, we adopt the DBSCAN algorithm to cluster for each bus station in geographical position. Let $Neib_\varepsilon(i)$ represent the collection of neighbor stations of station i, that is, $Neib_\varepsilon(i) = \{i'|dis(i',i) \leq \varepsilon\}$, where ε is the upper bound on distance and $dis(\cdot)$ is the Euclidean distance. Let *MinPts* be the lower bound on the number of neighbors. The following four steps describe how to cluster.

   Step 1: for a station i in the collection of boarding stations, if i has not been marked (which means that i is not marked as "processed" or "noise"), then check its neighborhood. If $|Neib_\varepsilon(i)| \geq MinPts$, then create a new cluster h, add i into h, and add all its neighbors to the candidate set N.

Step 2: for the neighborhood of station i' that has not been marked in N, if $|Neib_\varepsilon(i')| \geq MinPts$, then add $Neib_\varepsilon(i')$ into N, add i' into h, and mark i' as "processed", else mark i' as "noise".

Step 3: repeat *Step* 2, until all stations in N have been processed.

Step 4: repeat *Step* 1 to *Step* 3, until each station has been marked.

Note that the clustering steps in this paper are a bit different from the ones in the original DBSCAN algorithm. Now, these isolated points in the result of clustering are not noise. Each isolated point indicates that only one bus station is found nearby. Thus, it should form a distinct cluster. We store the clustering attributes of each boarding station in $H_{boarding}$.

For a passenger p, by matching all boarding stations of smart card records with $H_{boarding}$, we can get arrival times for each cluster. Finally, the high frequency clustering is supposed to be frequent boarding stations, and the others are general boarding stations.

While mining frequent disembarking stations, we also employ the DBSCAN algorithm. The specific mining steps are the same as the ones in mining boarding stations. We store all clustering attributes of disembarking stations in $H_{disembarking}$. Then, by matching all disembarking stations in smart card records with $H_{disembarking}$, we can get frequent disembarking stations.

For a passenger p, if the boarding station (O) and the disembarking station (D) of a record belong to different high frequency clustering respectively, then this record is a frequent OD record. If the proportion of the frequent OD records in p's records (denoted as $\rho_{p,OD}$) is greater than 50 % (which is $\rho_{p,OD} \geq 50\%$), then p is frequent in terms of OD.
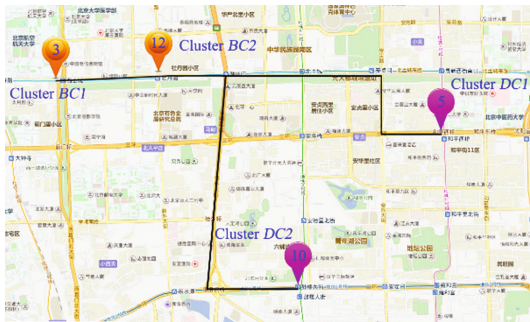
Figure 1 shows the result of mining frequent OD pairs for a passenger. The yellow bubbles represent the results of clustering of frequent boarding stations. And the purple bubbles represent the results of clustering of frequent disembarking stations. The number in each bubble is the frequency of boarding or disembarking for a passenger. If the frequency in a bubble is more than 6, then the station is regarded as a frequent boarding station or disembarking station. Following this rule, clusters *BC2* and *DC2* are frequent stations, while *BC1* and *DC1* are normal ones. So only (*BC2*, *DC2*) is a frequent OD pair. From Fig. 1, we know that the proportion of frequent trajectories is greater than 50 % (10/15 = 67 %), so this passenger is frequent in terms of OD pairs.

Fig. 1. The result of mining frequent OD pairs. (Color figure online)

Mining the Habit of Travel Time. For a passenger p, if he or she takes buses within a certain time interval frequently, this time interval is referred to as the habit of travel time for p. Once again, we adopt the DBSCAN algorithm to cluster the boarding times

of p in the bus smart card records for getting all the habits of travel time of p. In particular, the object i in the DBSCAN algorithm corresponds to the boarding time. The distance between objects corresponds to the time interval expressed in milliseconds. *MinPts* is the lower bound on i's neighbors. The resultant clusters signify frequent times, and isolated points signify noise. If the proportion of p's records with frequent times in all p's records (denoted as $\rho_{p,Habitual}$) is greater than 50 %, then we say that p has habitual time.

Recognizing Commuters. Having the above clustering analysis results, we can further make passenger classification. First, different kinds of passengers are defined as follows:

- OD passengers: the passengers are frequent in terms of OD pairs, but have no habitual time.
- Habitual time passengers: the passengers have habitual times, but are infrequent in terms of OD pairs.
- Commuters: the passengers are both frequent in spatial and temporal aspects.
- Casual passengers: they have none in spatio-temporal features.

Next, for each passenger p, the following rules are given to distinguish commuters:

Rule 1: if both $\rho_{p,OD}$ and $\rho_{p,Habitual}$ are small enough, p is a casual passenger.

Rule 2: if $\rho_{p,Habitual} < 50\%$, $\rho_{p,OD} > \rho_{p,Habitual}$, and $\rho_{p,OD}, \rho_{p,Habitual}$ are not too small, p is an OD passenger.

Rule 3: if $\rho_{p,OD} < 50\%$, $\rho_{p,Habitual} > \rho_{p,OD}$ and $\rho_{p,OD}, \rho_{p,Habitual}$ are not too small, p is a habitual time passenger.

Rule 4: if $\rho_{p,OD} \geq 50\%$ and $\rho_{p,Habitual} \geq 50\%$, p is a commuter.

4.2 Mining FBPT4BL Patterns

For reducing the overheads of trajectory retrieval, we design a forest structure which can organize trajectories with the same boarding station into a tree.

Establishing a Tree Structure. Each tree in the forest is used to store the trajectories with the same boarding station. Specifically, the same boarding station is stored in the root node, the disembarking station and these stations which passengers pass through are stored in the children nodes. Each node stores a triple, i.e., (*item*, *count*, *children*), where *item* stores the name of current station; *count* is the number of commuters' trips whose boarding station is the root node of this tree and the disembarking station is current node; and *children* contains the children of this node which are stored in a hash table.

Creating a Forest. The forest is established by these trees above and stored in a hash table. For a trajectory that has not been visited, if the tree whose root node is the boarding station of the trajectory does not exist, then we create the tree, otherwise we search the tree for the branch that matches with the trajectory. If the matched branch exists, then we increment count, where the count belongs to the node that matches with the disembarking station. Otherwise, we create a new branch and set count of the leaf node to 1.

Executing FBPT4BL Patterns Mining. The mining steps in the form of the pseudo code are shown in Fig. 2.

```
Input: forest forest, trip length bound r
Output: top-k result
Function GetFrequencyPattern(forest)
  Foreach tree in forest
    Foreach tp_c in tree whose count > 0
      If tp_pc exist in TP
        tp_c.ITFQ = tp_pc.ITFQ
        Add c.count of branch from each node o in tp_pc to c
      into tp_c.ITFQ;
      Else
        tp_c.ITFQ = GetPassengerCountP(tp_c);
      End If
      Save tp_c into TP, and count the score of tp_c;
    End Foreach
  End Foreach
  Sort TP by score and return top-k result from TP whose
  length is more than r;
  End Function;
Function GetPassengerCountP(trip)
  If(trip is null)
    Return 0;
  End If
  ITFQ = Sum(count of all the nodes in trip);
  Remove root node in trip;
  ITFQ += GetPassengerCountP(trip);
  Return ITFQ;
End Function;
```

Fig. 2. Mining FBPT4BL patterns

Step 1: traverse the tree in a depth-first order. Let c be the current visiting node. If $c.count > 0$, then mark the branch from root node to c as a BPT pattern tp_c, and store tp_c into a set of BPT pattern TP.

Step 2: let pc be the father node of c. If BPT pattern tp_{pc} exists in TP, then set $tp_c.ITFQ$ to $tp_{pc}.ITFQ$. Then, for each node o in tp_{pc}, find the tree whose root node is o, search for the branch that matches with the sub-trip in tp_c from o to c, and add the *count* of end node in the matched branch to $tp_c.ITFQ$, and then get to *Step* 5. If tp_{pc} is not in TP, then continue *Step* 3.

Step 3: sum all count values in tp_c to $tp_c.ITFQ$.

Step 4: delete the first node of tp_c to get the new pattern tp_c'. If tp_c' is empty, then continue *Step* 5, else match tp_c' against the forest. If the matched branch is found, then turn to *Step* 3. If not, then turn to *Step* 4. Here, the meaning of match is to get a branch whose root node is the first node of tp_c', and each node of the branch is the same as the bus station in tp_c' in order.

Step 5: select BPT patterns in TP whose lengths are larger than r, and compute the commuters' *score* values of these patterns by Eq. 1. Then, sort the selected patterns by

score to get top-k results which are the trips of BPT4BL patterns. By matching these trips to bus lines, the inter-zonal bus lines and bus intervals can be gotten.

4.3   Time Complexity Analysis

Let w be the number of bus stations, g be the number of commuters, and u be the trajectory number. The analysis of the MCFTP approach consists of three parts: commuter mining, forest creation, and FBPT4BL pattern mining.

In the commuter mining part, since the DBSCAN algorithm has the complexity of $O(w \cdot \log w)$, the complexity of mining frequent ODs is $O(2 \, g \cdot u)$, and the complexity of mining habitual time is $O(g \cdot u \log u)$, the total complexity is $O(w \cdot \log w + 2 \, g \cdot u + g \cdot u \log u)$.

In the second part, creating a forest needs to traverse all trajectories, which will make the complexity of this process $O(u \cdot w)$.

In the third part, we use an incremental method to mine FBPT4BL patterns. A new BPT pattern's *ITFQ* is calculated by the existing pattern's *IFTQ* via a traverse from the root node to its father node. From *Step* 2, we see that the complexity is $1 + 2 + 3 + \ldots + d = O(d^2)$, where d is the station number in a branch. Let e be the number of leaf nodes in the forest. The complexity of mining process is $O(d^2 \cdot e)$. Considering that each branch in a tree is a sub-trajectory of passengers and the total station number of one bus line in Beijing is no more than 50, which means that the maximum length of branches cannot be larger than 50, the complexity of mining process is $O(2500 \cdot e)$.

5   Evaluation

In order to show availability of the MCFTP approach, we need to fix following questions:

- How do we set the distance upper bound ε and the minimum number of neighbors *MinPts* in the commuter identification phase?
- Does only commuter data result in better experimental results than all passengers or other types of passengers?
- Does the MCFTP approach is appropriate to a real-time bus scheduling?

To answer the above questions, we first conduct experiments to analyze the sensitivities of ε and *MinPts*. Then we evaluate effectiveness and efficiency of the MCFTP approach. In experiments, we use the real data from Beijing Public Transport Group, including 6,507,837 passengers and 48,427,884 items of bus smart card data from Aug. 3, 2015 to Aug. 9, 2015, as well as 5,622 bus stations and 707 bus lines.

5.1   Sensitivity Analysis of *MinPts* and ε

The application of the DBSCAN algorithm requires two important parameters: the distance upper bound ε and the minimum number of neighbors *MinPts*.

While the DBSCAN algorithm is applied to clustering by the spatial information, ε denotes the upper bound of spatial distance. A small ε may result in a big error since some stations which belong to the same destination may be clustered into different classes, whereas a too large ε also performs poorly since the stations falling into different destinations may be clustered into one class.

In Beijing, the distance between neighbor bus stations ranges from 500 m to 1200 m. Considering that an acceptable walking distance is 1000 m or so, we increase ε from 500 m to 1000 m to evaluate clustering results. However, *MinPts* is used to set a regular cluster density. In the downtown, bus stations within ε should not be very congested. Thus, in experiments, we increase *MinPts* from 1 to 10.

While the DBSCAN algorithm is applied to temporal information, ε denotes the upper bound of the temporal distance. ε is set to 20 min, since in general commuters will arrive at stations at a relatively fixed time points, varying within a very narrow interval. *MinPts* is set to 3. That means a passenger is regarded as frequent in temporal aspect if the passenger takes a bus at same station at least three times one week.
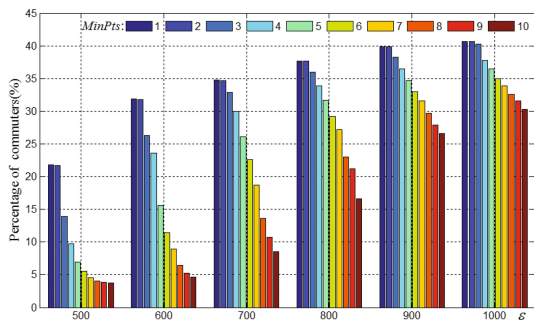

**Fig. 3.** The percentages of commuters.

Figure 3 shows percentages of commuters under different ε and *MinPts* (in spatial clusters). From Fig. 3, we can see that for the same ε, commuter number decreases with the increase of *MinPts*. It is because a large *MinPts* will generate a large number of clusters and those trips which should be in a same cluster will be departed into different clusters. In particular, we find when setting ε to 500 or 600 and *MinPts* to a value greater than 5, the percentages of commuters will be less than 5 %, which cannot be true in Beijing. We deem that the percentages of commuters ranging from 20 % to 35 % is rational, so we set ε to 600 and *MinPts* to 4 for the experiments in Sect. 5.2.

5.2   Effective and Efficient Analysis of MCFTP Approach

We collect the real inter-zonal bus lines in Beijing as the ground truth. So far, about 3,851 inter-zonal bus lines have been operated, covering 282 bus lines.

In experiments, we set r to 15, and then run our MCFTP approach using the smart card data of different days as input. We choose different percentages (denoted as γ, from 1 % to 20 %) of recommended inter-zonal bus lines as the results of our MCFTP approach, and calculate the matching ratio of our results to the ground truth. We also employ the methods in Sect. 4.2 to mine FBPT4BL patterns from all passengers, OD
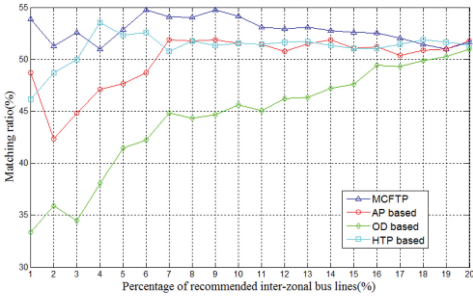
(a) Aug. 4, 2015
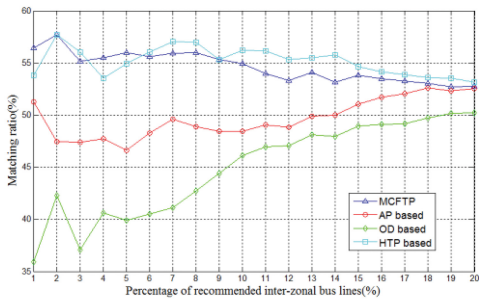
(b) Aug. 5, 2015
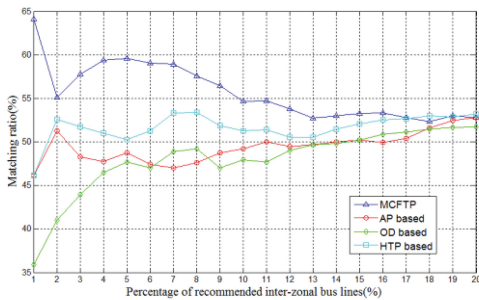
(c) Aug. 7, 2015

Fig. 4. The experiment results of inter-zonal bus recommendation.

passengers and habitual time passengers. The corresponding methods are called the AP based method, the OD based method and the HTP based method, respectively. The results are showed in Fig. 4.

From Fig. 4, we can see that the MCFTP approach performs better than other three methods. The highest matching ratio of our approach is 62 %, and the lowest matching ratio is 51 %. For the data on Aug. 7, our MCFTP approach is 28 % higher than the OD based method, 18 % higher than the HTP based method, and 18 % higher than the AP based method while γ is set to 1 %. On average, the HTP based method is mostly close to our method. The reason behind is that these habitual time passengers have regular boarding habits and inter-zonal bus lines in morning rush hours can greatly benefit these passengers.

We find that the OD based method performs worst in four methods. However, from the above experiments, we know that the sum of all passengers is 6,507,837. Among them, the number of OD passengers is 4,661,267, the number of habitual time passengers is 243,370, and the number of commuters is 1,536,927. OD passenger's number is about 3 times of commuter's number. That means that the inter-zonal bus lines planning should rely on the contributory passenger factor instead of passenger number.

As regard to the time complexity, for the data of one day, we need 65 s to build the forest and 5.852 s to mine inter-zonal bus lines. Our method has the low time complexity, and can be used for scheduling inter-zonal buses in real-time.

6   Conclusion

In the paper, we focus on mining patterns in the trajectories. We propose a two-phase approach to mine FBPT4BL patterns from the bus smart card data and obtain inter-zonal bus lines. The resulting inter-zonal bus lines can be used for evaluating the rationality of existing inter-zonal bus lines or guiding the opening of inter-zonal bus lines.

References

1. Luo, W.M., Tan, H.Y., Chen, L., Lionel, M.N.: Finding time period-based most frequent path in big trajectory data. In: Proceedings of the 2013 ACM SIGMOD International Conference on Management of Data, pp. 713–724 (2013)
2. Lee, A.J.T., Chen, Y.A., Ip, W.C.: Mining frequent trajectory patterns in spatial–temporal databases. Inf. Sci. **179**, 2218–2231 (2009)
3. Fosca, G., Mirco, N., Fabio, P., Dino, P.: Trajectory Pattern Mining. In: KDD 2007 Proceedings of the 13th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 330–339 (2007)
4. Kai, Z., Yu, Z., Nicholas, J.Y., Shuo, S.: On discovery of gathering patterns from trajectories. In: IEEE 29th International Conference on Data Engineering, pp. 242–253 (2013)
5. Li, Z.-H., Bolin, D., Han, J.-W., Roland, K.: Swarm: mining relaxed temporal moving object clusters. J. Proc. VLDB Endowment **3**(1), 723–734 (2010)
6. Dou, M., He, T., Yin, H., Zhou, X., Chen, Z., Luo, B.: Predicting passengers in public transportation using smart card data. In: Sharaf, M.A., Cheema, M.A., Qi, J. (eds.) ADC 2015. LNCS, vol. 9093, pp. 28–40. Springer, Heidelberg (2015)
7. Cui, Z.Y., Long, Y.: Perspectives on stability and mobility of passenger's travel behavior through smart card data. In: UrbComp in Conjunction with ACM SIGKDD (2015)
8. Le, M.K., A.B., Edward C.: Passenger segmentation using smart card data. IEEE Trans. Intell. Transp. Syst., 1537–1548 (2015)
9. Zhang, J., Yu, X., Tian, C., Zhang, F., Tu, L., Xu, C.Z.: Analyzing passenger density for public bus: inference of crowdedness and evaluation of scheduling choices. In: Proceedings of IEEE International Conference on Intelligent Transportation Systems, ITSC (2014)
10. Zuo, X.Q., Chen, C., Tan, W., Zhou, M.C.: Vehicle scheduling of an urban bus line via an improved multiobjective genetic algorithm. IEEE Trans. Intel. Transp. Syst. **16**(2), 1030–1041 (2015)
11. Favyen, B., Huang, Y., Xie, X., Powell, J.W.: A greener transportation mode: flexible routes discovery from GPS trajectory data. In: GIS 2011 Proceedings of the 19th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems, pp. 405–408 (2011)