B-Planner: Planning Bidirectional Night Bus Routes Using Large-Scale Taxi GPS Traces

Chao Chen, Daqing Zhang, Nan Li, and Zhi-Hua Zhou, *Fellow, IEEE*

***Abstract*—Taxi GPS traces can inform us the human mobility patterns in modern cities. Instead of leveraging the costly and inaccurate human surveys about people's mobility, we intend to explore the night bus route planning issue by using taxi GPS traces. Specifically, we propose a two-phase approach for bidirectional night bus route planning. In the first phase, we develop a process to cluster "hot" areas with dense passenger pick up/drop off and then propose effective methods to split big hot areas into clusters and identify a location in each cluster as a candidate bus stop. In the second phase, given the bus route origin, destination, candidate bus stops, and bus operation time constraints, we derive several effective rules to build the bus route graph and prune invalid stops and edges iteratively. Based on this graph, we further develop a bidirectional probability-based spreading algorithm to generate candidate bus routes automatically. We finally select the best bidirectional bus route, which expects the maximum number of passengers under the given conditions and constraints. To validate the effectiveness of the proposed approach, extensive empirical studies are performed on a real-world taxi GPS data set, which contains more than 1.57 million night passenger delivery trips, generated by 7600 taxis in a month.**

***Index Terms*—Bus route planning, human mobility patterns, route graph, taxi GPS traces.**

I. Introduction

BUSES are a popular and economical way for people to travel around the city, and they are generally "greener" than cars and taxis as they help decrease traffic congestion, fuel consumption, carbon dioxide emission, and travel cost [1]. Thus, for sustainable city development, people are encouraged to take public transportation for work, visit, etc. In many cities, the daytime bus transportation systems are usually well designed; however, during late nights, most bus systems are out of service, leaving taxis as the only option for intracity travelling. To provide cost-effective and environment-friendly transport to citizens, many cities start to plan night-through bus routes.

Previously, bus route planning mainly relied on human surveys to understand people's mobility patterns [5], [19]. Although this approach was proved to be workable, the time and cost spent in the survey process were quite substantial. Moreover, such an approach is not able to accommodate the frequent change in the road network and traffic, particularly for cities, which experience rapid development. With the wide deployment of GPS devices and wireless communication in taxis, rich information about taxis, including where and when passengers are picked up or dropped off and which route a taxi takes for a certain trip, can be collected and extracted. Knowing the origin–destination (OD) of each taxi trip provides valuable information to understand passengers' mobility flow in a city at different times of a day, making it possible to accurately plan new night bus routes, which expect the maximum number of passengers along the routes.

In this paper, we intend to explore the bidirectional night bus route design problem leveraging the taxi GPS traces. This problem can be divided into two subproblems: the candidate bus stop identification and the best bidirectional bus route selection. For the first subproblem, we need to identify the candidate bus stops, which are associated with locations having a big number of taxi passenger pick-up and drop-off records (PDRs); the bus stops should be evenly distributed in the "hot" districts to facilitate people's access. After the candidate bus stops are identified, the next step is to select a bidirectional bus route, which connects the bus origin and a sequence of bus stops to the destination, expecting the maximum number of passengers in both directions given a specific bus operation time, frequency, and total travel time. Fortunately, the taxi GPS traces contain quantitative spatial–temporal information about all taxi trips. By mining the taxi GPS data, we can inform where are the hot areas for taxi passengers and how many passengers would potentially travel along a certain route in a specific time duration. Therefore, the bidirectional night bus route design becomes a problem of comparing the number of passengers of all valid bus routes giving certain time constraints.

However, identifying the candidate bus stops from taxi GPS data and enumerating the top-ranked bidirectional bus routes efficiently are not trivial and straightforward. To the best of our knowledge, there is still no work reported on this topic. For example, given the taxi GPS trajectories of night time for a certain time period, let us say that seven dense taxi pick-up/drop-off locations (i.e., C_1-C_7) have been identified as candidate bus

Manuscript received October 22, 2013; revised January 4, 2014; accepted January 6, 2014. Date of publication February 4, 2014; date of current version August 1, 2014. This work was supported in part by Institut Mines-Telecom through the "Futur et ruptures" Program and in part by the National Science Foundation of China under Grant 61333014 and Grant 61105043. The Associate Editor for this paper was F. Zhu.

C. Chen is with the CNRS UMR 5157 SAMOVAR, Institut Mines-Telecom/Telecom SudParis, 91011 Evry, France, and also with the Department of Computer Science, Pierre and Marie Curie University, 75005 Paris, France (e-mail: chao.chen@telecom-sudparis.eu).

D. Zhang is with the CNRS UMR 5157 SAMOVAR, Institut Mines-Telecom/Telecom SudParis, 91011 Evry, France (e-mail: daqing.zhang@telecom-sudparis.eu).

N. Li is with the National Key Laboratory for Novel Software Technology, Nanjing University, Nanjing 210023, China, and also with School of Mathematical Sciences, Soochow University, Suzhou 215006, China (e-mail: lin@lamda.nju.edu.cn).

Z.-H. Zhou is with the National Key Laboratory for Novel Software Technology, Nanjing University, Nanjing 210023, China (e-mail: zhouzh@lamda.nju.edu.cn).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TITS.2014.2298892

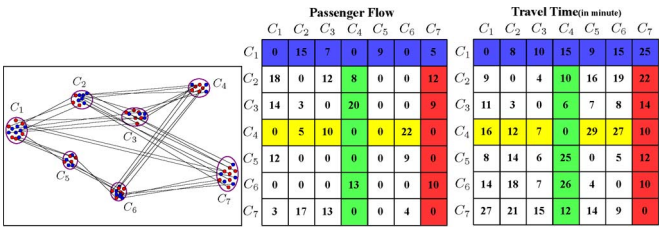

Fig. 1. Illustrative example of the (left) taxi GPS traces, the (middle) passenger flow, and the (right) travel time among bus stops.

stops, as illustrated in Fig. 1, where C_1 and C_7 are the bus origin and destination, respectively; and the corresponding passenger flow and travel time among stops are shown in the right panel in Fig. 1. The objective of bidirectional bus route design is to find a bidirectional bus route ($C_1 \rightarrow C_7$ and $C_7 \rightarrow C_1$) with the maximum number of passengers expected given the bus operation time constraints. Apparently, to design an effective bus route, the following research challenges need to be addressed.

- *Candidate bus stop identification*: The taxi passenger pick-up and drop-off points are distributed in the whole city, with some areas having more PDRs than other areas, but there is no clear guideline about where the bus stops should be put.
- *Tradeoff between the number of passengers and travel time*: To deliver more passengers, the best bus route should go through more bus stops (e.g., go through all stops between C_1 and C_7), but this will take more travel time. Hence, a nontrivial tradeoff is needed.
- *Passenger flow accumulation effect*: Assuming there is no taxi passenger travelling from C_4 to C_7, if we plan the route as $C_1 \rightarrow C_2 \rightarrow C_3 \rightarrow C_7$, then the significant passenger flow in $C_2 \rightarrow C_4$ and $C_3 \rightarrow C_4$ cannot be accommodated. Alternatively, by including C_4 in the route as $C_1 \rightarrow C_2 \rightarrow C_3 \rightarrow C_4 \rightarrow C_7$, these passenger flows can be accommodated with the cost of adding one stop. Therefore, we need to consider this accumulation effect, which tends to lead to a globally better solution.
- *Dynamic passenger flow*: The passenger flows are usually different from time to time; for example, the passenger flow during 23:00–24:00 can be very different from that during 3:00–4:00; thus, we need to consider this dynamics when planning bus routes.
- *Asymmetry of passenger flow and travel time*: It is easy to see that the best route in terms of passenger flow and travel time for one direction (from C_1 to C_7) is probably not the best one for the opposite direction (from C_7 to C_1); we thus need to select the bus route with the maximum accumulated number of passengers in two directions.

In this paper, we address the aforementioned challenges using a two-phase approach, with the process illustrated in Fig. 2. Roughly speaking, in the first phase, we identify candidate bus stops by clustering and splitting hot areas, and then, in the second phase, we propose several strategies to find best bus routes. Specifically, the main contributions of this paper can be summarized as follows.

First, we propose a two-phase approach to tackle the bidirectional night bus route design problem leveraging the taxi GPS traces. To the best of our knowledge, this is the first work on bidirectional night bus route design exploiting the taxi travel speed, time, pick-up and drop-off information in large scale, and real-world taxi GPS traces.

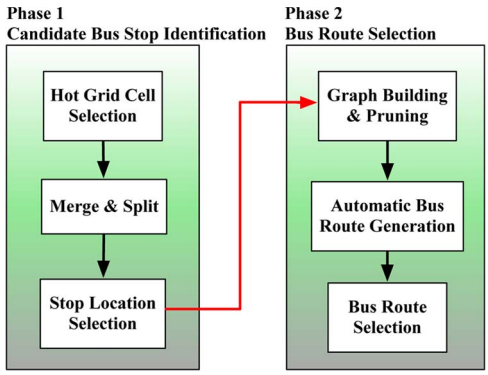

Fig. 2. Two-phase bus route planning framework.

Second, we develop a novel process with effective methods to cluster hot areas with dense passenger pick up/drop off, split big hot areas into walkable size ones, and identify candidate bus stops. Moreover, we study how different thresholds in the merge and split algorithms affect bus stop identification results and final selected bus routes.

Third, we propose rules to build and prune the directed bus route graph. Based on the graph, we propose a new heuristic algorithm, named *bidirectional probability-based spreading* (BPS) algorithm, to select the best bidirectional bus route that can achieve the maximum number of passengers expected in two directions. It is verified that the BPS algorithm outperforms the top-k approach in the selection of the best bus route. We also investigate the impact of different bus stop distances on the final bus routes selection.

Finally, we determine the night bus capacity by computing the maximum number of passenger on buses for the selected bus route at different stops and different bus frequencies. To understand the impact of the new opened bus route on taxi services, we further report the passenger flow change along the bus route before and after the new bus route opened date.

The rest of this paper is organized as follows. In Section II, we first review the related work and show the differences from other work. In Section III, we present the process for candidate bus stop identification, and in Section IV, we elaborate the process for bus route graph building and pruning, automatic bus route generation, and best route selection. Extensive evaluation results are reported in Section V to verify the effectiveness of the proposed approach. Finally, we conclude this paper and chart the future directions in Section VI.

II. Related Work

Here, we briefly review the related work, which can be grouped into two categories. The first category is about mining taxi GPS traces, whereas the second focuses on bus network design other than exploiting taxi GPS traces.

A. Taxi GPS Traces Mining

We can group the existing work about mining taxi GPS traces into three categories: *social dynamics* mining, *traffic*

dynamics mining, and *operational dynamics* mining [9], [45]. Social dynamics is defined as the work studying the collective behavior of a city's population and observing people's movement in the city, which are motivated by diverse needs and influenced by external factors (such as weather and traffic). A deep understanding of social dynamics is essential for the management, design, maintenance, and advancement of a city's infrastructure. Common research problems in this subcategory using taxi GPS traces include the following: where do people go throughout the day [24], [27], what are the "hottest" spots around a city [41], what are the "functions" of these hot spots [29], [33], [42], how strongly connected are different areas of the city [7], [50], etc. Governed by their underlying desires or needs, people will move around the city mainly through the road network. While social dynamics aims to understand people's movement patterns, traffic dynamics studies the resulting flow of the population through the city's road network. Most work in this subcategory aimed at uncovering the root cause of traffic outliers [12], [28], predicting traffic conditions, which are useful for providing real-time traffic forecast [5], [10], [20], and travel time estimation for drivers [6], [43]. Operational dynamics refers to the general study and analysis of taxi drivers' *modus operandi*. The aim is to learn from the taxi drivers' expert knowledge of the city and detect abnormal or effective driving behaviors. The last two subcategories mainly used the ODs of a taxi trip trajectory (i.e., the pick-up and drop-off locations); in the study of operational dynamics, researchers make use of full trajectories, as the routes taken by drivers are of utmost importance. Researchers have mined these trajectories to suggest strategies for quickly finding new passengers/taxis [18], [21], [25], [35], [44], recommend time-dependent navigational routes for reaching a destination quickly [43], plan flexible bus routes [7], and suggest driving routes to achieve dynamic taxi ride sharing [16], [30]. Additionally, new trajectories can be compared against a large collection of historical trajectories to automatically detect abnormal behavior [13], [17], [36], [46]. Before taxi GPS trace mining, data clean and repair may be required since the data can be noisy [48].

Among all the taxi GPS trace mining related papers, the work in the social dynamics subcategory, which addresses "hot spots" and frequent travel OD patterns, is relevant to our work for identifying candidate bus stops and calculating passenger flow among potential bus stops, but there are not many papers, except two [7], [14] using those data for bus route planning. The main goal of [7] is to mine historical taxi GPS trips to suggest a flexible bus route. The work first clusters trips with similar starting time, duration, origin, and destination; it then attempts to identify the route that connects multiple dense taxi trip clusters. The goal is different from ours as it only chooses the route that maximizes the sum of each connected trip cluster. In another word, it does not consider the time constraints and the accumulated effects among connection stops; thus, it would never include the path such as $C_4 \rightarrow C_7$ in Fig. 1 in the planned bus routes, whereas our approach might include the path as long as the route expects the maximum number of accumulated passengers and the total travel time constraint can be met. The research objective of [14] is to find an optimal bus route for a given OD pair in a single direction. Due to the asymmetrical passenger flows of planning route, the one-direction optimal route obtained by [14] is generally not the best one in both directions. However, in real bus route planning, buses usually run on the same route in both directions in order to facilitate passengers' access to the bus service (easy to remember the bus route and bus stops). Therefore, it is ideal to plan the bidirectional bus route that can achieve the maximum number of passengers in both directions. In this paper, we attempt to address the bidirectional bus route planning problem. In addition to conducting bus route planning, this paper also differs from [14] in the following two aspects. First, we provide a mechanism to determine the maximum number of passengers at different bus operation frequencies for estimating the bus capacity for bus planners to save the operation cost. Second, we investigate the effect of adding new bus route on the taxi services through an empirical study.

B. Bus Network Design

Bus network design is an intensively studied area in the urban planning and transportation field [3], [38], [39], [47]. The bus network design is known to be a complex, nonlinear, nonconvex, and multiobjective NP-hard problem [26], [31], [32], [34], [37]. The aim is to determine bus routes and operation frequencies that achieve certain objectives, subject to the constraints and passenger flows. The popular objectives include shortest route, shortest travel time, lowest operation cost, maximum passenger flow, maximum area coverage, and maximum service quality; whereas the constraints include time, capacity, and resources [11], [15], [22], [49].

However, the selection of the objectives should take care of the operator and user requirements, which are often conflicting, leading to design tradeoff rather than an optimal solution. As noted in [32] and [34], early bus network design was mainly based on human survey to get passenger flows and user requirements; it relied heavily on heuristics and intuitive principles developed by a designer's own experience and practice. Recent work on bus network design has also assumed that the passenger flows are given by user survey or population estimation; many complex optimization approaches have been proposed, and among them, the best solving algorithms are based on heuristic procedures [23] to find near-optimal solutions. A detailed review about route network design can be found in [19].

Despite the renewed attention for bus network design, there is still no work addressing the bidirectional night bus route design problem leveraging the taxi passenger OD flow data. Different from existing research, our work aims to find a bidirectional bus route with a fixed frequency, maximizing the number of passengers expected along the route subject to the total travel time constraint. This problem is different from the traditional travelling salesman problem (TSP) [4] in nature, which aims to find the shortest path that visits each given location (node) exactly once. TSP evaluates different routes with exact N locations, which means all candidate stops should be included in the route. Our problem is also different from the shortest path finding problem [40], which intends to get the shortest path for a given OD pair. In our case, we have to consider the accumulated

effect (passenger flows) from all previous stops to the current stop for choosing the bidirectional bus route.

III. Candidate Bus Stop Identification

In the proposed two-phase bus route planning framework, the objective of the phase one is to identify candidate bus stops by exploiting the taxi PDRs. Here, we describe our proposed process for identifying candidate bus stops. As shown in Fig. 2, the whole process consists of three steps.

1) Divide the whole city into small equal-sized grid cells; mark those hot grid cells with high taxi passenger PDRs for further processing.
2) Merge the adjacent hot grid cells to form hot areas; divide each big area into "walkable size" cluster.
3) Choose one grid cell as the candidate bus stop location in each walkable size hot cluster, by assuming that passengers from the same cluster would easily walk to the stop to take bus.

A. Hot Grid Cells and City Partitions

In this work, we first divide the city into equal-sized grid cells, with each cell about 10 m $\times$ 10 m in size. In such a way, the whole city is partitioned into 5000 $\times$ 2500 cells in total. Out of all the grid cells, over 95% of them contain no taxi passenger PDRs as they are either lakes, mountains, buildings, and highways that cannot be reached by taxis or suburb areas that people seldom travel to. Only 0.11% of the grid cells have more than 0.2 PDRs per hour on average if we only count the PDRs in late night. In addition, we name these grid cells as hot ones.

As each grid cell has a maximum of eight neighbors, if we define the connectivity degree (CD) of a hot grid cell as the number of hot neighboring cells, the CD of any grid cell will range from 0 to 8, where the hot grid cell with CD equal to 0 is called the isolated cell. As the city is composed of mixed hot grid cells and common grid cells, both hot and common cells form irregular "hot areas" and "common areas," respectively, as a consequence of same type of cells being adjacent to each other. These hot areas are also called city partitions, as shown in Fig. 3. Apparently, some small partitions (e.g., the green one in Fig. 3) can be very close to some big ones (e.g., the red one in Fig. 3). It would be necessary to consider all the city partitions globally in order to plan the bus stop locations; thus, city partitions close to each other had better merge to form big clusters for better overall bus stop distribution. In the next section, we propose a simple strategy to merge the close partitions into bigger clusters.

B. Cluster Merging and Splitting

We present the cluster merging and splitting approach in Algorithm 1. After obtaining all city partitions, we sort them in a descending order according to the number of PDRs (Line 1). To merge the partitions close to each other iteratively, we propose to use the hottest partition to *absorb* its nearby partitions according to the descending order of PDRs, until no more

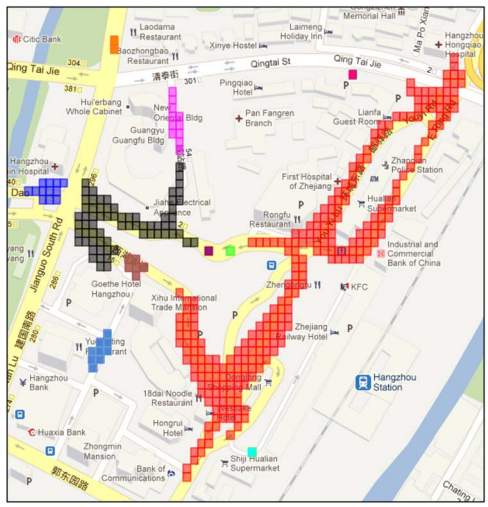

Fig. 3. City partitions near Hangzhou Railway Station.

nearby partitions meet the merging criterion (Line 8). Then, we choose the next hottest partition to repeat the same process until all the partitions are checked (Lines 8–12). The location of each partition is first initialized by computing the weighted average location of all grid cells using

$$\mathrm{loc}(P) = \frac{\sum_{i=1}^{N} (\mathrm{PDR}(g_i) * \mathrm{loc}(g_i))}{\sum_{i=1}^{N} \mathrm{PDR}(g_i)} \tag{1}$$

where $\mathrm{loc}(g_i)$ refers to the longitude/latitude of the member grid cell g_i.

Algorithm 1: Merge Algorithm

Input: List of partitions $\{P_i\}$
Output: List of clusters $\{C_i\}$
1: $P \leftarrow \mathrm{sort}(P)$, $(i = 1, 2, \ldots, n)$ // Sort P according to amount of its PDRs by descending order
2: $i = 1$; // Initialization
3: **while** $P \neq \emptyset$ **do**
4: $C_i = \{P_1\}$;
5: $P = P \backslash \{P_1\}$ // Remove P_1 from P
6: $k = |P|$;
7: **for** $j := 1$ **to** k **do**
8: **if** $\mathrm{dist}(C_i, P_j) < T_1$ **then**
9: $C_i = C_i \cup P_j$ // *absorb* the closer partition
10: $P = P \backslash \{P_j\}$ // Remove P_j from P
11: **end if**
12: **end for**
13: $i = i + 1$;
14: **end while**

After merging one partition, the location of the combined cluster is updated (Line 9), and the absorbed partition is removed from the partition list (Line 10). The dist function refers to the distance between two given partitions. The algorithm will be terminated until no partitions can be merged to a new cluster (Line 3). A main parameter in the merge algorithm is

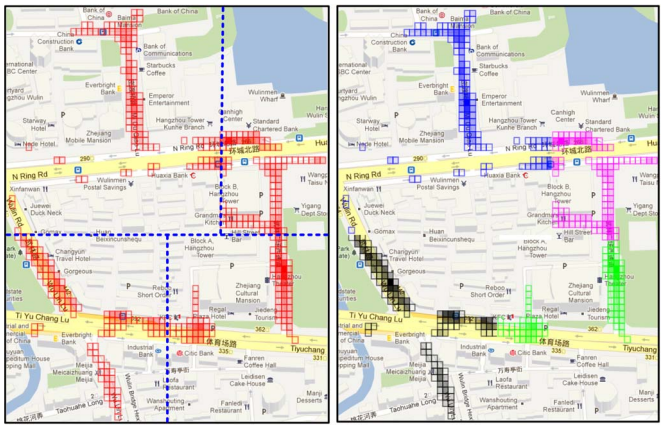

Fig. 4. Illustrative example of splitting. (Left) Big cluster formed via merging. (Right) Big cluster split into four walkable size clusters (in four different colors).

T_1 (Line 8), which controls *how far* a big cluster can absorb its nearby clusters. Intuitively, a bigger T_1 would allow big clusters to absorb more nearby clusters, leading to a fewer number of clusters in total but more big clusters. We will further investigate how T_1 would affect the resulted best route parameters quantitatively in Section V-B2.

In general, the merged clusters can be classified into three groups according to their size (the size of cluster is defined as the minimal rectangle, which covers all the grid cells): 1) with both height and width greater than T_2; 2) with either height or width greater than T_2; and 3) with both height and width less than T_2 (where T_2 is the maximum distance that passengers are willing to walk to reach a bus stop).

As for large clusters (Groups 1 and 2), we adopt a simple strategy to split them. Specifically, for clusters in Group 1, we first split the big cluster into two subclusters, aiming to minimize the difference of PDRs of the resulted clusters both in horizontal and vertical directions, whereas for clusters in Group 2, we only need to split the cluster in one direction. We split the cluster in the horizontal direction if its height is greater than its width; otherwise, we split it in the vertical direction, again with the goal of minimizing the number difference of PDRs of the split subclusters. With one split, one big cluster would produce two smaller subclusters. Thus, a smaller T_2 would need more splitting times and leads to more smaller clusters finally.

Fig. 4 shows an illustrative example of splitting a cluster into four subclusters with the proposed splitting strategy. The initial cluster belongs to Group 1 [see Fig. 4 (left)]; the splitting is first done in the horizontal direction to produce two subclusters with similar PDRs. After the first splitting, two subclusters with width greater than T_2 are generated (T_2 is set to 500 m); thus, both subclusters require further splitting in the vertical direction. The final result with four split subclusters is shown in Fig. 4 (right). We will also study how T_2 would affect the resulted best route parameters in Section V-B2.

C. Candidate Bus Stop Location Selection

After *merging* and *splitting* operations, we obtain a big number of hot clusters with the size smaller than $T_2 \times T_2$, which are scattered in the dynamic districts of the city during late night. The next step is to select a *representative* grid cell in each cluster to serve as the candidate bus stop.

To select this *representative* grid cell, both the CD and the number of PDRs of each cell in the cluster are taken into consideration. While the CD of a grid cell characterizes the accessibility of the cell, the number of PDRs is an indicator of its "hotness." The grid cell having the maximum value defined in (2) in each cluster is selected as the "center" of the cluster, marked as the location of the candidate bus stop

$$\arg\max_i \left[w_1 \times \frac{\mathrm{CD}(i)+1}{9} + w_2 \times \frac{\mathrm{PDR}(i)}{\sum_{i=1}^{n}(\mathrm{PDR}(i))} \right]. \quad (2)$$

We set $w_1 = w_2 = 0.5$ in the evaluation, and in total, we get 579 candidate bus stops in the city by using the taxi GPS data from Hangzhou, China. Note that different weight settings in (2) would only affect locations of the bus stop and would have no impact on the total number of bus stops.

IV. Bus Route Selection

After fixing the candidate bus stops in phase one, the aim of phase two is to find the best bus route for a given OD, expecting to maximize the number of passengers expected under the time constraints in two directions (i.e., O $\rightarrow$ D and D $\rightarrow$ O).

Here, we first approximate the passenger flow and the travel time between any two candidate stops using taxi GPS traces; then, we present the bus route selection method, which contains the following three steps (see Fig. 2): 1) Build the bus route graph and remove invalid nodes and edges iteratively based on certain criteria; 2) automatically generate candidate bus routes with the two proposed heuristic algorithms; 3) select the bus route by comparing the expected number of passengers under the same total travel time constraint.

A. Passenger Flow and Travel Time Estimation

We record the travel demand and time information in two matrices, named passenger flow matrix (FM) and bus travel time matrix (TM). Each element in a matrix refers to the number of passengers or the bus travel time from one stop (ith) to another stop (jth, $i \neq j$). We count the total taxi trips from the ith cluster to the jth cluster as each stop is responsible for its cluster. We set the maximum waiting time for passengers at the stop as 30 min (equal to the bus operation frequency); thus, any pick-up or drop-off events taking place in this time window are counted. We simply assume that the passenger flows among candidate bus stops remain unchanged during each 30-min duration. The final FM is obtained by averaging all flow matrices at different bus frequencies. We also assume that TM remains unchanged across the night time. $\mathrm{tm}(s_i, s_j)$ is the average travel time multiplied by α, which is a constant. We set $\alpha = 1.5$ to consider the speed difference between taxis and buses. For the paths having no taxi trip occurring in history (for instance, nobody travels by taxi due to too short distance), we use $\mathrm{Ddist}(s_i, s_j)/v$ to approximate $\mathrm{tm}(s_i, s_j)$, where $\mathrm{Ddist}(s_i, s_j)$ is the driving distance between s_i and s_j, and v is a constant and is set to 50 km/h.

B. Bus Route Graph Building and Pruning

Selecting the best bus route is a very challenging problem as two conflicting requirements must be met: one requirement is to ensure that the bus route would traverse intermediate stops and finally reach the destination within a limited time; the other one is to maximize the number of passengers accumulated along the route from all previous stops to the destination. For example, if we choose the stop with the heaviest passenger flow from the origin as the first node and then keep choosing the next stop following the heaviest passenger flow principle, then we might neither be able to reach the destination nor achieve the objective of having the maximum number of passengers accumulated along the route. To meet the preceding two requirements and follow the intuitive principles in bus route design, some basic criteria should be set for the building of the bus route graph and the selection of the candidate bus route.

1) Route Graph Building Criteria: Obviously, there would be numerous stop combinations for a given OD pair, and only a small proportion of them meet the first or second requirement. In order to reduce the search space of possible stops and routes, we can build a bus route graph starting from the origin to the destination using heuristic rules. These rules are derived either from one of the preceding two requirements or from the intuitive bus route design principle. For instance, from the shortest travel time perspective, the bus route should extend from the origin toward the direction of destination, which can be further converted into three rules: each new selected stop should be farther from the origin, closer to the destination, and farther from previous stops. From the intuitive bus route design principle, the bus stops should not be too far from each other, and the bus route should not comprise sharp zigzag paths. These can be also translated into two criteria in building the bus route graph. Specifically, given the OD pair (s_1, s_n) and the candidate route $R = \langle s_1, s_2, \ldots, s_n \rangle$, we should follow the following criteria when building the bus route graph with stops (nodes) and directed edges among nodes.

- *Criterion 1: Adequate stop distance*

$$\text{dist}(s_{i+1}, s_i) < \delta \quad (i = 1, 2, \ldots, n-1)$$

where δ is a user-specified parameter. It means the maximum distance between two consecutive stops. We will study the effect of varying δ values on the best route parameters in Section V.

- *Criterion 2: Move forward*

$$x_{\text{new}}(i+1) > x_{\text{new}}(i) \quad (i = 1, 2, \ldots, n-1)$$
$$x_{\text{new}}(i) = x(i)\cos\theta + y(i)\sin\theta$$
$$\theta = \tan^{-1}\frac{y(n)}{x(n)}$$

$(x(i), y(i))$ of s_i is obtained by simply subtracting the longitude and latitude values to that of s_1. x_{new} is the X-axis value of stop in the new coordination, which is with s_1 as the new origin, and from s_1 to s_n as the new direction of the X-axis [see Fig. 5 (left)]. This criterion guarantees that the bus will always move forward along the OD direction.

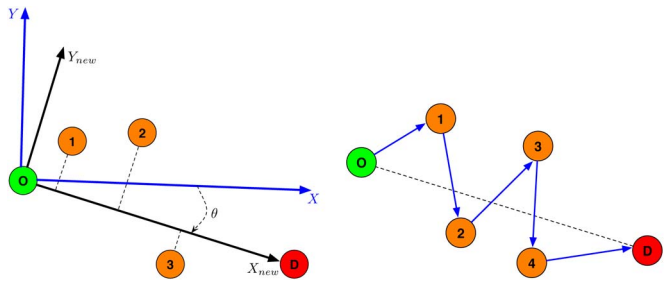

Fig. 5. Demonstration of (left) Criterion 2 and (right) Criterion 5.

- *Criterion 3: Origin-farther*

$$\text{dist}(s_{i+1}, s_1) > \text{dist}(s_i, s_1) \quad (i = 1, 2, \ldots, n-1).$$

This ensures that the bus will move away from the origin s_1 farther in each step.

- *Criterion 4: Destination-closer*

$$\text{dist}(s_{i+1}, s_n) < \text{dist}(s_i, s_n) \quad (i = 1, 2, \ldots, n-1).$$

This ensures that the bus will move closer to the destination s_n in each step.

- *Criterion 5: No zigzag route*

$$\arg\min_{s_j}(\text{dist}(s_{i+1}, s_j)) = s_i \quad (j = 1, 2, \ldots, i).$$

Criterion 5 ensures the smoothness of the route. There would be no sharp zigzag path along the OD direction. The route demonstrated in the right panel in Fig. 5 should not happen, as it violates the *no zigzag route* criterion. We can see $\arg\min_{s_j}(\text{dist}(s_3, s_j)) = s_1 \neq s_2 (j = 1, 2)$ and $\arg\min_{s_j}(\text{dist}(s_4, s_j)) = s_2 \neq s_3 (j = 1, 2, 3)$.

2) Graph Building and Pruning: The aim of graph building is to construct a directed graph with nodes and links given an OD pair, in which the nodes are the stops and edges link the stop to its next possible stops, regardless of passenger flows among them. The goal of graph pruning is to remove invalid edges and nodes according to the proposed criteria.

Graph building: Given the bus route origin and destination, their locations are first used to narrow down the choice of valid candidate stops; only the candidate stops lying between them are under consideration. For each stop within the range, we determine links to its next possible stops according to the proposed Criterion 1 to Criterion 4. The process will terminate when all stops have been checked. Finally, stops having no edges would be excluded.

As *Criterion 5* is related to all stops in one bus route, we use it to prune the route graph after it is built. Fig. 6 (left) shows an illustrated example about a generated bus route directed graph. Note that the graph is built based on the geographical constraints; thus, the edge may have no taxi passenger flow on itself.

Graph pruning: Some nodes and edges can be further pruned because they are not valid for candidate bus route selection. To be specific, nodes without in-coming edges (if not origin) or out-going edges (if not destination) should be deleted as they will not form any valid routes with the bus route OD pair.

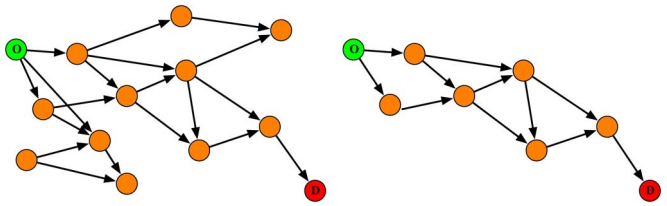

Fig. 6. Bus route directed graph for a given OD. (Left) The route graph is obtained by graph building algorithm and (right) its corresponding graph after applying graph pruning.

We first calculate all the nodes' in-coming and out-going degrees. Afterward, nodes (exclude the given OD) together with related edges would be iteratively deleted from the graph if their in-coming or out-going degree is zero. Finally, a graph with only one zero in-coming degree node (i.e., the given origin) and one zero out-going degree node (i.e., the given destination) would be generated. After graph pruning, all the bus routes starting from the source and following the edges in the graph would eventually reach the destination. Fig. 6 (right) displays the resulted graph after applying pruning to the graph in Fig. 6 (left).

Graph for D → O: An intuitive way of building a route graph for D → O is to run the previous two steps again, with the D as the new origin and O as the new destination. However, *Theorem* 1 below ensures that the route graph from D to O is just the same as that from O to D, with all the edges having opposite directions.

Theorem 1: If $R = \langle s_1, s_2, \ldots, s_n \rangle$ is a candidate bus route for pair (s_1, s_n), then its reversed route $\bar{R} = \langle s_n, s_{n-1}, \ldots, s_1 \rangle$ will be the candidate bus route for the (s_n, s_1) pair.

Proof: To prove $\bar{R}$ is the candidate bus route for the (s_n, s_1) pair, we just need to check whether it meets all the five criteria. It is obvious that $\bar{R}$ meets the first four criteria. For *Criterion 5*, given a particular node $s_i (1 < i < n-1)$ in R, we can derive its two closest nodes as s_{i-1} and s_{i+1}. Thus, $\arg\min_{s_j}(\text{dist}(s_i, s_j)) = s_{i+1} (j = n, n-1, \ldots, i+1)$ will hold. □

C. Automatic Candidate Bus Route Generation

Based on the graph constructed in the previous section, we first propose our probability-based spreading (PBS) algorithm for O → D, followed by the BPS approach, which can select the best bus routes in both directions.

PBS Algorithm: Although we have removed invalid nodes and edges through graph pruning, the problem of enumerating all possible routes from the given source to the destination is proved to be NP hard. Indeed, it is also unnecessary to enumerate all possible routes and compare them all because most of the routes are *dominated* by few others.

Definition 1: We say R_i *dominates* R_j iif: 1) $T(R_i) \leq T(R_j)$; 2) $\text{Num}(R_i) > \text{Num}(R_j)$. The route that is not *dominated* by others in the route set is defined as a *skyline route*, where T and Num are the total travel time and the number of expected delivered passengers, respectively. We compute them based on (3) and (4). The skyline route definition is similar to that in [18], and the rationale behind is that only routes with less travel time but larger number of passengers should be selected. *Skyline detector* [8] will prune the routes, which are dominated by skyline routes in the candidate set. Thus, the comparison can be done among detected skyline routes

$$T = \sum_{i=1}^{n-1} \text{tm}(s_{i+1}, s_i) + (n-2) \times t_0 \tag{3}$$

$$\text{Num} = \sum_{i;j(j>i)}^{n} \text{fm}(s_i, s_j) \tag{4}$$

where t_0 is the average time needed to board at each stop, and we set it to 1.5 min.

Algorithm 2: PBS Algorithm

Input: $G(S, E)$: Single directional graph for the given OD pair
FM: Flow matrix
TM: Travel time matrix
Output: $\mathcal{R}^*$: the set of skyline routes
1: $\mathcal{R} = \emptyset$
2: **Repeat**
3: current$R = s_1$
// starts from the given origin s_1
4: Choose the next stop s_i^* with respect to current$R$ according to (5)
5: $R =$ current$R \cdot s_i^*$
// $\cdot$ operation appends s_i to current$R$
6: **Repeat** Lines 4 and 5 **Until** $s_i^* = s_n$
// ends at the given destination s_n
7: $\mathcal{R} = \mathcal{R} \cup R$
8: Get corresponding skyline routes $\mathcal{R}^*$
9: **Until** $\mathcal{R}^*$ keeps unchanged

The key idea of our proposed PBS algorithm is to randomly select the next stop among the possible candidate stops in each step, where the candidate stops having high accumulated passenger flow with previous stops are given high probability for random selection. We describe the approach in Algorithm 2. The spreading starts from the given source (Line 3). The next stop in the candidate route is chosen based on

$$P\left(s_i^* | \langle s_1, s_2, \ldots, s_j \rangle\right) = \frac{\sum_{m=1}^{j} \text{fm}\left(s_m, s_i^*\right)}{\sum_{i=1}^{|S^*|} \sum_{m=1}^{j} \text{fm}\left(s_m, s_i^*\right)} \tag{5}$$

where $\text{fm}(s_m, s_i^*)$ is the passenger flow from s_m to s_i^*, and S^* contains the next possible stops of s_j (child nodes of s_j in the route graph).

We can see that the selection of the next stop in the candidate route is not only determined by the current stop but also all the previous stops. The output of this algorithm is one candidate bus route, with the number of stops associated with the number of spreading steps. The spreading would be terminated when the given destination is reached (Line 6). For each run, we get either a repeated route or a new route; thus, the candidate route

set $\mathcal{R}$ would increase as the spreading algorithm is activated. Then, a question arises: *how many running times are sufficient to get the best results ?* Based on *Definition* 1 about the skyline routes, we should consider if the skyline route set $\mathcal{R}^*$ remains changed or unchanged.

Theorem 2 below ensures that when the skyline route set remains unchanged with the increase of spreading algorithm runs, then the best route has been discovered.

Theorem 2: $\mathcal{R}_1^*$ and $\mathcal{R}_2^*$ are the detected skyline routes from $\mathcal{R}_1$ and $\mathcal{R}_2$, respectively. If $\mathcal{R}_1 \subseteq \mathcal{R}_2$, then we have, $\forall R_i \in \mathcal{R}_1^*, \exists R_j \in \mathcal{R}_2^*$; $R_i = R_j$ or R_i is dominated by R_j.

In Algorithm 2, we have $\mathcal{R}_{t_1} \subseteq \mathcal{R}_{t_2}$ if the running time $t_1 < t_2$, and the algorithm would be stopped when no better skyline routes are returned with the increase of running times, that is, $\mathcal{R}_{t_1}^* = \mathcal{R}_{t_2}^*$ (Line 9). The computation complexity of the algorithm is $\mathcal{O}(N)$.

Instead of choosing only one stop randomly at each spreading step such as in the PBS algorithm, an intuitive way is to select top-k stops each time, where those k nodes should have the highest accumulated passenger flow with previous stops. In such a way, the first step selects top-k nodes, thus leading to k routes from the origin to those nodes. In the second step, each k node would select another top-k node; thus, the total candidate routes would be k^2. Assuming that n steps are needed to the destination, then the total candidate routes generated would be k^n in the end. Thus, the computation complexity of this algorithm is $\mathcal{O}(k^n)$, which exponentially grows with the spreading step, i.e., n. We use this *top-k spreading* method as the baseline.

BPS Algorithm: In practice, for a particular bus line, buses can run on the same route in both directions. Algorithm 2 can get the best bus route in one direction (e.g., from ZJU to Railway Station); however, it cannot guarantee that the same route in the opposite direction (i.e., from Railway Station to ZJU) would still expect the maximum number of passengers, as the passenger flows in two directions of the route are generally asymmetrical. To get a bus route that has the overall maximum expected number of passengers in both directions, we propose the BPS algorithm, whose basic idea is to run the PBS algorithm in both directions so that we generate one candidate "optimal" route in each direction, and the best route is selected by evaluating all the candidate routes in two directions.

We illustrate the procedure in Algorithm 3. The key idea behind is to run Algorithm 2 in both directions (Lines 3 and 4) and generate one candidate route for each direction at each run (Line 5). The skyline routes are selected based on the total travel time and the expected number of passengers in both directions of each candidate route (Line 6), and the selection process terminates also when no more better skyline routes can be generated (Line 7).

Algorithm 3: BPS Algorithm

Input: $G_{O\to D}(S,E)$: Graph for $O \to D$
$G_{D\to O}(S,E)$: Graph for $D \to O$
FM: Flow matrix
TM: Travel time matrix output:
Output: $\mathcal{R}^*$: the set of skyline routes
1: $\mathcal{R} = \emptyset$
2: **Repeat**
3: Run Lines 2–6 in Algorithm 2 for $G_{O\to D}(S,E)$, and the output is $R_{O\to D}$
4: Run Lines 2–6 in Algorithm 2 for $G_{D\to O}(S,E)$, and the output is $R_{D\to O}$
5: $\mathcal{R} = \mathcal{R} \cup R_{O\to D} \cup R_{D\to O}$
6: Get corresponding skyline routes $\mathcal{R}^*$
7: **Until** $\mathcal{R}^*$ remains unchanged

D. Bus Route Selection

Given the bus operation frequency (once every 30 min), the total travel time constraint, and the taxi passenger flow from 21:30 to 5:30, we obtain the candidate bus routes for a given OD pair using the two different heuristic spreading algorithms, and the skyline route that achieves the maximum expected number of passengers will be selected as the operating route.

With the planned bus route consisting of the selected bus stops, the next step is to find a physical bus route in the real setting, which consists of road segments corresponding to the planned route. The selection of each road segment is done by following the dense and fine trajectories of taxis if they allow buses to operate. Otherwise, similar bus routes near the planned ones can be adopted as a refined solution.

V. Experimental Evaluation

Here, we validate the proposed approach with a large-scale real-world taxi GPS data set, which is generated from 7600 taxis in a large city in China (Hangzhou) in one month, with more than 1.57 million of night passenger-delivering trips. All the experiments are run in MATLAB on an Intel Xeon W3500 personal computer with 12-GB random access memory running Windows 7.

A. Evaluation on Bus Stops

We compare the bus stop results generated with our proposed method with that generated by the popular k-means clustering method. We set $k = 579$, which is the same as our method. We adopt the Eulerian distance as the similarity metric. The centroid of each cluster is selected as the stop. Fig. 7 shows the comparison results. Comparing with the popular k-means approach, our proposed candidate bus stop identification method has two advantages.

1) The centroid of each cluster obtained by k-means is the average location of all its members, and it may fall into nonreachable places like river, as highlighted by the black circles in Fig. 7 (left). In our proposed method, both hotness and connectivity of each grid cell are considered for the bus stop location selection, and the selected bus stops are meaningful and stoppable places.
2) Several identified stops by k-means fall into a small area (highlighted by the blue circle) as the size of clusters obtained by k-means is very different, whereas our

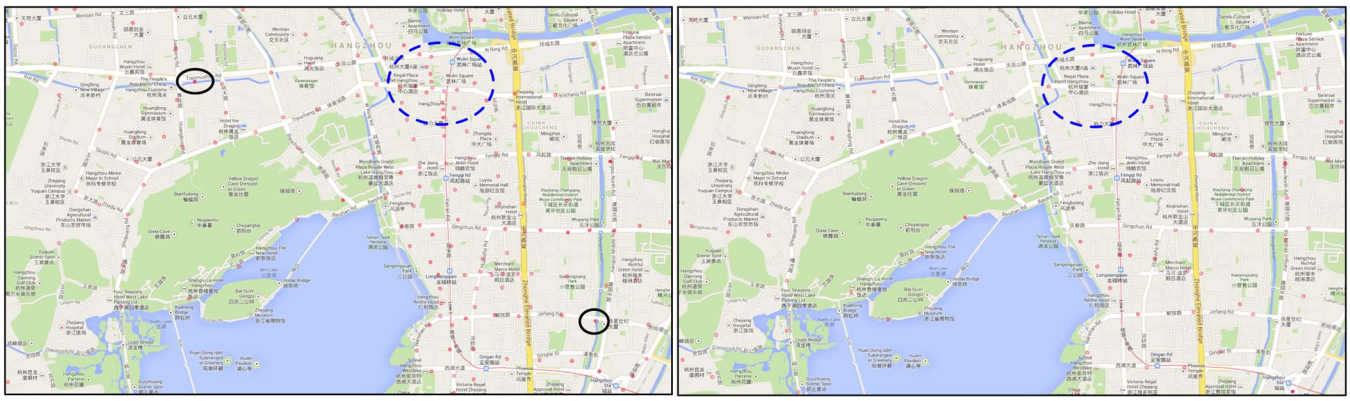

Fig. 7. Comparison results with k-means (best viewed in the digital version). (Left) Results obtained by k-means. (Right) Results obtained by our method.

TABLE I
DETAILED INFORMATION ABOUT STUDIED OD PAIRS

	OD Pairs	Distance (km)	Number of Stops
1	ZJU - Railway	5.70	104
2	Railway - East Railway	5.86	75
3	East Railway - ZJU	8.80	144

proposed method generates candidate bus stops that are evenly distributed in the hot areas, which better meets the common sense design criteria of bus stops.

B. Evaluation on Bus Route Selection Algorithm

We first show the convergence of the proposed algorithm, followed by a parameter sensitivity study. Then, we perform a quantitative statistical analysis of all the candidate routes generated for three given OD pairs. We also give the computed skyline route results. Finally, we validate that our proposed bus route generation approach outperforms the baseline approach. Table I shows the details of three OD pairs for night bus route design experiment, where more than 70 candidate bus stops are in the candidate bus route selection list.

1) Convergence Study: As illustrated in Algorithms 2 and 3, our proposed bus route generation process would be terminated if the resulted skyline routes remain unchanged. We study the similarity of consecutively generated skyline routes from 5000 to 150 000 runs, with a constant interval of 5000 runs. We measure the similarity, i.e., sim, of two sets A and B as follows:

$$\text{sim}(A, B) = \frac{|A \cap B|}{|A \cup B|}. \tag{6}$$

The similarity results of the consecutively generated skyline routes with a 5000-run interval are shown in Fig. 8, and the time cost is put in the diagram as well. In this paper, we can see that sim values gradually reach 1 with the increase of runs for all three OD pairs, meaning that, in all three cases, the best bus route converges to one. In addition, the time cost is almost linearly increased with the number of runs, suggesting that the spreading time cost at each run is almost constant. It is also noted that the three curves for three OD pairs have different slopes; the reason is probably because the bus routes corresponding to different ODs have different lengths and varied

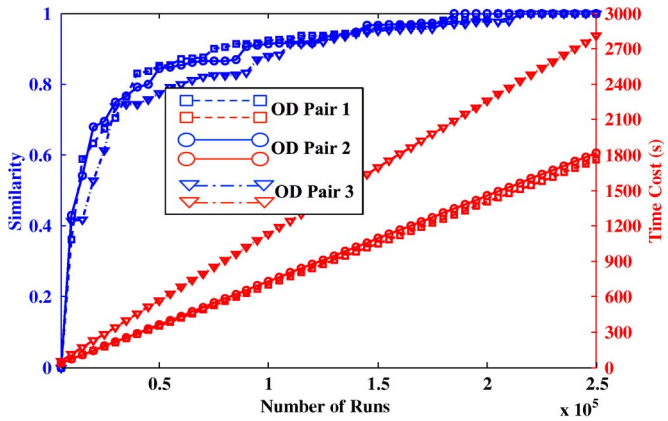

Fig. 8. Convergence study of the proposed BPS algorithm.

numbers of candidate bus stops; thus, the spreading time and the candidate bus stop selection time should be also different.

2) Parameter Sensitivity Study: To better understand the bus stop identification and the bus route selection algorithms, we conduct experiments under different parameter settings to study how they affect the number of expected passengers of selected routes and running time. We examine three parameters in the process, while two of them are in the bus stop identification phase, the remaining one is in the route graph building algorithm.

Varying parameters (T_1 and T_2) for the cluster merge and split algorithms: As discussed in Section III, a bigger T_1 would produce more large clusters, and likewise, a bigger T_2 would also generate more large clusters. Fig. 9(a) and (b) shows the *cumulative distribution function* (cdf) of finally produced clusters in terms of size after cluster merging and splitting under various T_1 ($\in$ [100:50:300] m) and T_2 ($\in$ [400:50:650] m), respectively. We also show the skyline route results under different T_1 and T_2 in Fig. 9(c) and (d). From these results, we can see that choosing the relatively smaller T_1 and larger T_2 will lead to better skyline routes.

Fig. 9(e) and (f) shows the maximum number of expected passengers for the selected bus route and the time cost, respectively, under different T_1 and T_2 combinations. Note that the time cost is the total cost of the candidate bus stop identification phase and the bus route selection phase. We also find that combinations of bigger T_1 and smaller T_2 are not good as they

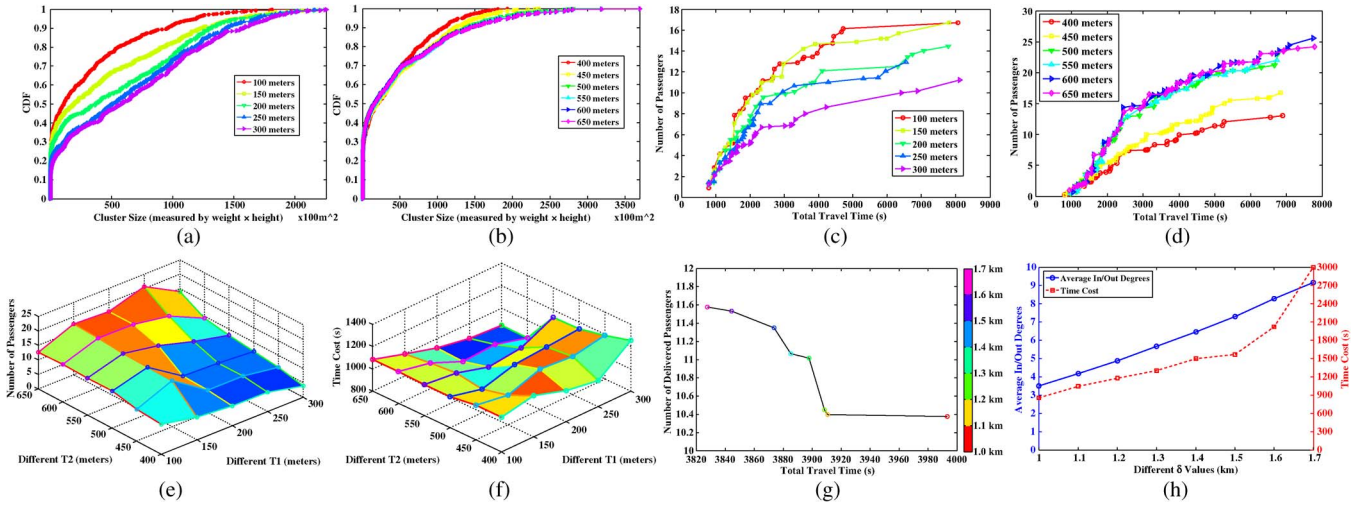

Fig. 9. Results of parameter sensitivity study. (a) CDF results of cluster size under different $T_1 (T_2 = 500$ m). (b) CDF results of cluster size under different $T_2 (T_1 = 150$ m). (c) Skyline route results under different $T_1 (T_2 = 450$ m). (d) Skyline route results under different $T_2 (T_1 = 150$ m). (e) Maximum number of passengers under different T_1 and T_2 combinations. (f) Time cost under different T_1 and T_2 combinations. (g) Selected bus routes at different δ. (h) Route graph complexity and time cost under different δ.

often result in lower number of passengers but higher time cost. Specifically, the minimum number of passengers and the maximum time cost occur at $T_1 = 300$ m and $T_2 = 400$ m. This is probably because, for the candidate bus stop identification phase (i.e., Phase 1), a bigger T_1 would first generate more large clusters in the cluster merging procedure; then, a smaller T_2 would require more spitting operation times during the cluster splitting; finally, a larger number of small-size clusters would be identified; for the bus route selection phase (i.e., Phase 2), the route graph would become more complex with the increase of the number of candidate bus stops, and meanwhile, the number of passengers decreases as the walkable distance is set short. Finally, we choose $T_1 = 150$ m and $T_2 = 500$ m throughout this paper as it expects a larger number of passengers while consuming relatively less time. Additionally, a 500-m distance is an acceptable walk distance for passengers.

Varying the parameter δ for the graph building algorithm: Here, we study the impact of δ selection on the expected number of passengers of the selected bus route and time cost. For a particular stop s_i, a larger δ would lead to more child nodes. Mathematically, we have, $\forall s_i \in S$, $S'_{\delta_1}(s_i) \subseteq S'_{\delta_2}(s_i)$ if $\delta_1 \leq \delta_2$, where $S'(s_i)$ is the child node of s_i in the route graph. In addition, we also have $\mathcal{R}_{\delta_1} \subseteq \mathcal{R}_{\delta_2}$. Therefore, with the increase in δ value, a better route can be obtained. Meanwhile, the route graph would become more complex, resulting in the increase of computation time.

We investigate different δ in the range of [1.0 km, 1.7 km] for OD pair 2, with a constant interval of 0.1 km. Fig. 9(g) shows two metrics of the selected bus route under different δ values. One point on the plane stands for the selected route under a given δ. We can see that the selected route becomes steadily better with the increase in δ (deliver more passengers with less travel time). However, the difference is negligible after $\delta \geq 1.5$. We also show the complexity of the route graph and the time cost under different δ values in Fig. 9(h). The complexity of the graph is simply quantified by the average in-coming/out-going degrees. They are equal to the ratio of the total number of edges

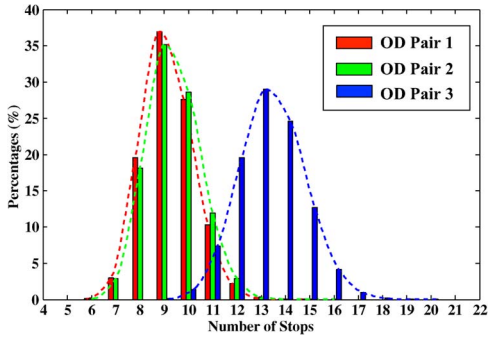

Fig. 10. Number of stops of candidate route stops statistics for three OD pairs.

to the total number of nodes in the route graph. From the figure, we can see that the average in-coming/out-going degrees under 1.7 km are twice more than that under 1.0 km. Furthermore, more computation time is needed when δ increases because the route graph becomes more complex. We set $\delta = 1.5$ km throughout this paper as it leads to good performance with low time cost.

3) Candidate Routes Statistics: Fig. 10 shows the statistical information about the number of stops of candidate routes. Several interesting observations can be obtained.

1) For OD pair 1, routes with 8–10 stops take up over 80% of the cases (both origin and destination are included). Few routes can reach the destination by traversing only four stops or passing more than 11 stops.
2) For OD pair 2, over 60% of the routes contain 9 or 10 stops. Similar to the case of OD pair 1, some routes can reach the destination by passing four stops.
3) For OD pair 3, most of the routes contain 10–18 stops due to the longer OD distance, and almost half of the routes include 13 or 14 stops.
4) The statistical results comply with the intuition that the longer the distance of a given OD pair, the more stops the route would contain.

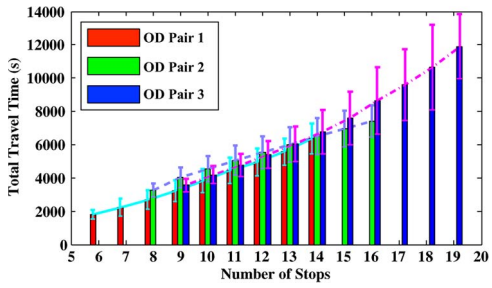

Fig. 11. Relationship between the number of stops and the total travel time statistics for three OD pairs.

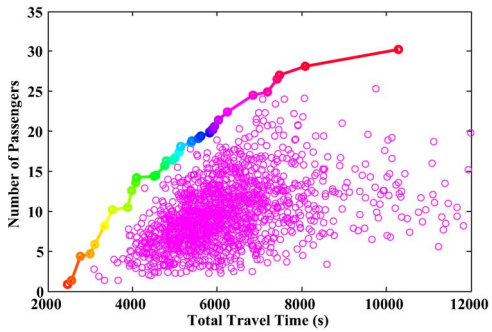

Fig. 12. Detected skyline routes and other candidate routes.

We also provide the statistics of the total travel time of candidate routes having the same number of stops (mean and standard deviation), which is shown in Fig. 11. We can see that, for all three OD pairs, the average total travel time almost linearly increases with the number of stops, suggesting that the total travel time constraint is related to the constraint of the total number of stops.

4) Skyline Routes: We show the skyline routes for OD pair 3 in Fig. 12. Each point in the plane represents a candidate route. The x-axis stands for the total travel time of candidate route, whereas the y-axis represents the expected number of passengers. In Fig. 12, we can see that the skyline routes are connected to form a curve above all the points representing common routes, and over 99% of the routes are dominated by the few skyline routes. Specifically, we get 36 skyline routes across all the travel time frames, out of hundreds of thousands of routes for the case of OD pair 3. Similar phenomena have been observed for the other two cases as well.

5) Comparison With Top-k Spreading Algorithm: In the top-k spreading algorithm, the selection of k is vital to the skyline routes generated and the time needed to generate all the candidate routes. In particular, when $k_1 < k_2$, we have $\mathcal{R}_{k_1} \subseteq \mathcal{R}_{k_2} (k_1 \leq k_2)$. Theorem 2 guarantees that a bigger k would lead to a better set of skyline routes. However, the greater k also results in a significant increase in time cost. We compare the skyline routes generated from the BPS method with that from the top-k spreading method with different k values for the case of OD pair 1, which is shown in Fig. 13. We can see that the BPS approach outperforms the top-k algorithms, even when k is set to 5. Again, similar conclusion can be also drawn for the other two OD pairs.

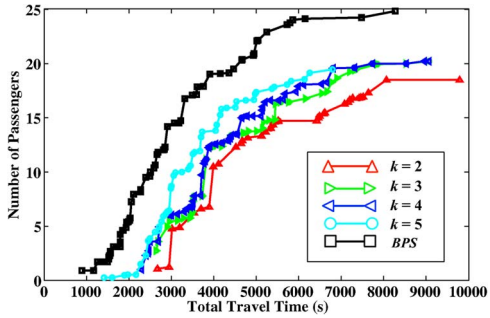

Fig. 13. Comparison results with baseline under different k values.

C. Bidirectional Versus Single-Directional Bus Route

In real life, the bus route obtained by Algorithm 2 may be one of the following: 1) be the skyline route in both directions; 2) be the skyline route in only one direction; 3) not be the skyline route in any direction. It is noteworthy to compare the overall best bidirectional bus route obtained by Algorithm 3 to the best routes in single direction. We have drawn all the selected bus routes on the city digital map in Fig. 14 for OD pair 3. They are different routes, which means that the bidirectional bus route is neither the skyline route in the ZJU $\rightarrow$ East Railway Station direction nor in the East Railway Station $\rightarrow$ ZJU direction. A reasonable explanation is that the passenger flow and the travel time among stops are often asymmetrical, and thus, the bus route that carries the maximum number of passengers under the give time constraints in one direction would probably fail to deliver the same performance in the opposite direction. However, they all have 13 stops in total and share several common stops near the ZJU stop, particularly for the route $R_{O \rightarrow D}$ [see Fig. 14 (left)] and $R_{O \leftrightarrow D}$ [see Fig. 14 (right)]. By further checking, we find that these common stops are popular nightlife centers.

We show the average travel time and the number of expected delivered passengers of these three bus routes in Table II, and note that heavier passenger flow can be found from the East Railway Station to ZJU direction ($R_{D \rightarrow O}$). While $R_{D \rightarrow O}$ takes slightly less time and delivers a larger number of passengers than $R_{O \rightarrow D}$, it carries about 48 more passengers on average per night. $R_{O \leftrightarrow D}$, however, takes the least time, and the average number of delivered passengers lies between $R_{O \rightarrow D}$ and $R_{D \rightarrow O}$.

D. Comparison With Real Routes and Impacts on Taxi Services

As the taxi GPS data set we have was collected from April 2009 to March 2010, we are very interested in knowing if there was any new night bus route created during this year and how the planned bus route generated with our approach compares with the manually created route. Fortunately, we were told that a night bus route was created in February 2010. We could access all the taxi passenger flows before and after the route started date. It is noted that the route is designed by local experts and that the user demands are obtained from expensive human survey. We first draw the newly started night bus route R_3 on

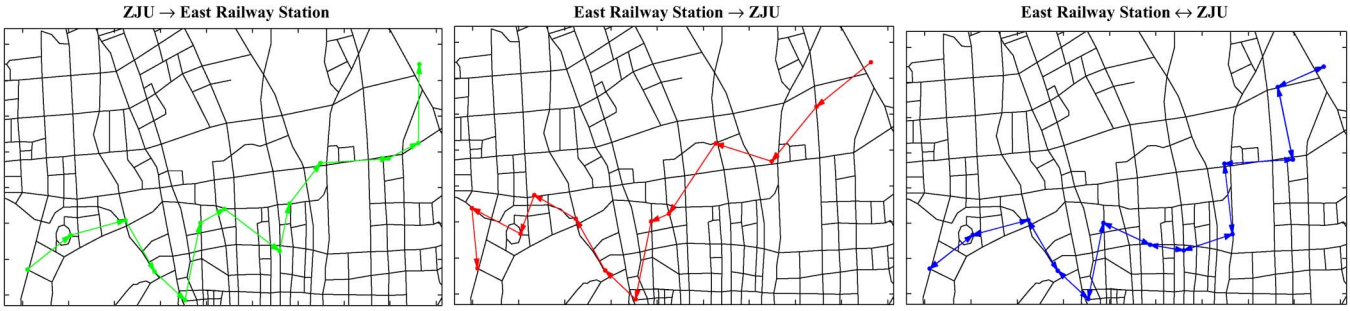

Fig. 14. Comparison results of the selected bus routes in two directions to that in one direction. (Left) $R_{O\to D}$. (Middle) $R_{D\to O}$. (Right) $R_{O\leftrightarrow D}$.

TABLE II
TWO METRICS OF THE SELECTED BUS ROUTES

	Direction	Average Travel Time (in second)	Number of Passengers
$R_{O\to D}$	ZJU→East Railway	5406.7	17.25
$R_{D\to O}$	East Railway→ZJU	5352.2	20.31
$R_{O\leftrightarrow D}$	ZJU↔East Railway	5320.2	18.73

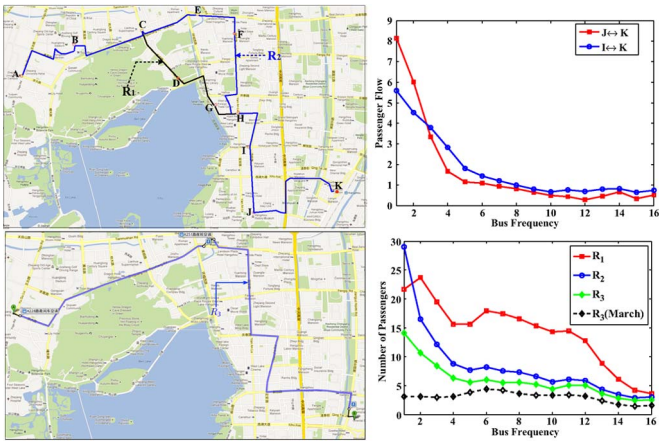

Fig. 15. Results comparison. (Top left) Planned routes. (Top right) Passenger flow comparison of two segments at different frequency. (Bottom left) Opened night bus route. (Bottom right) Number of delivered passengers at different frequency (R_1, R_2, and R_3).

Google map, as shown in Fig. 15 (bottom left), and then, we draw our proposed night bus route R_1 in Fig. 15 (top left). Through comparison, we see that they are quite different. With the newly started route, we decide to take a similar route in our selected candidate bus routes (not the best one), and we find R_2, as shown in Fig. 15 (top left). It is noted that the main difference between R_2 and the newly started route R_3 is that R_2 includes an additional Stop J in the route. By comparing the passenger flow in segment I ↔ K with that in segment J ↔ K at different time slots, it is found that the passenger flow in path J ↔ K is even greater than I ↔ K in the first two time slots, as shown in Fig. 15 (top right). Considering further the accumulation effects, including Stop J in the bus route would significantly increase the expected number of passengers along the route. This is evidenced by Fig. 15 (bottom right). The *accumulated effect* is more remarkable at the first three frequencies. Thus, our candidate bus route R_2 would outperform the newly added bus route R_3, at the cost of adding one more bus stop and more travel time.

TABLE III
TOTAL TRAVEL TIME OF THE BUS ROUTES

Bus Route	Total Travel Time (in second)
R_1	3583.8
R_2	4664.9
R_3	3624.0

We also compare our proposed best route R_1 with the candidate route R_2. The difference between R_1 and R_2 lies in two different paths taken from C to H. While R_2 passes the famous shopping street (Yan'an Road) in Hangzhou $(C \leftrightarrow E \leftrightarrow F \leftrightarrow H)$, R_1 traverses the famous night club areas along the West Lake. If we compare the number of passengers in R_1 and R_2, it is shown in Fig. 15 (bottom right) that the passenger flow of R_2 is heavier than that of R_1 only around 22:00, and it is much lighter soon after 23:00. With the rest of the stops being the same for both R_1 and R_2, there is no doubt about why R_1 has been selected as the best night bus route. If we take a closer look at R_1, R_2, and the newly started route R_3, R_1 takes a much shorter route than R_2 and needs similar travel time as the newly started route R_3 does (see Table III), but R_1 expects much more passengers than R_2 and the newly started route R_3; thus, it is reasonable to conclude that the selected night bus route with our proposed approach is better than the current route-in-service in terms of travel time and expected number of passengers.

It is understood that introducing new public services (i.e., new Metro/bus lines) would affect taxi services in the city [2]. It is interesting to compare the taxi passenger flow change along the new bus route before/after it was opened. We choose the new night bus route (R_3) opened in February 2010 for this study. We prepare taxi GPS data collected in January and March 2010 and calculate the corresponding taxi passenger flow along the new bus route across all bus frequencies, which is shown in Fig. 15 (bottom right). We can see that the number of passengers who travel by taxi along the bus route in March is much smaller but quite stable across all the bus frequencies. This may be interpreted by the fact that, while some passengers might switch to public services, a certain number of passengers still prefer to take taxis at night.

E. Bus Capacity Analysis

After selecting the best bus route for operation, the next important thing is to determine the proper bus capacity to save operation cost. The essence for bus capacity estimation is to determine the maximum number of passengers on the bus

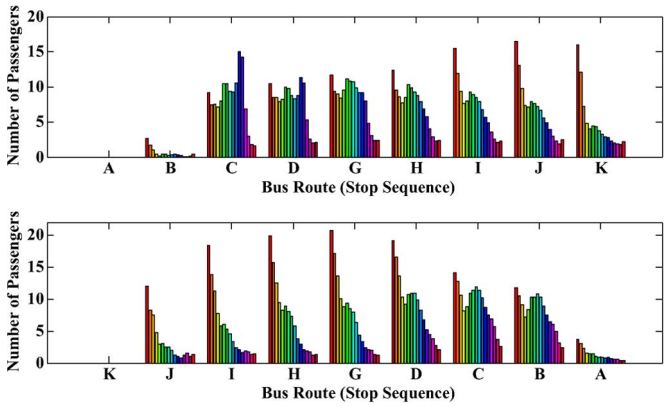

Fig. 16. Number of passengers on the bus before reaching the stop for OD pair 1.

across all the frequencies. For the bus route R_1 of OD pair 1, Fig. 16 shows the number of passengers on the bus across all the frequencies for both directions. As can be seen from the results, choosing buses with 20 seats could well meet the requirements. In addition, we also have the following three observations.

1) More passengers are often expected in both directions for the first operation frequency, except for the eleventh and twelfth frequencies, when the bus runs from C to D.
2) Buses running close to the capacity only last for three stops (from A to K) or four stops (from K to A).
3) Night buses heading toward different directions have quite different passenger flow patterns.

VI. Conclusion and Future Work

In this paper, we have investigated the problem of bidirectional night bus route design by leveraging the taxi GPS traces. This work is motivated by the needs of applying pervasive sensing, communication, and computing technology for sustainable city development. To solve the problem, we propose a two-phase approach for night bus route planning. In the first phase, we develop a process to cluster hot areas with dense passenger pick up/drop off and then propose effective methods to split big hot areas into clusters and identify a location in the cluster as the candidate bus stop. In the second phase, given the bus route origin, destination, candidate bus stops, and bus operation frequency and maximum total travel time, we derive several criteria to build bus route graph and prune the invalid stops and edges iteratively. Based on the graph, we further develop two heuristic algorithms to automatically generate candidate bus routes in both directions, and finally, we select the best route that expects the maximum number of passengers under the given conditions. On a real-world data set, which contains more than 1.57 million passenger delivery trips, we compare our proposed candidate bus stop identification method with the popular k-means clustering method and show that our method can generate more reasonable and meaningful results. We further extensively evaluate our proposed BPS algorithm for automatic bus route generation and validate its effectiveness and its superior performance over the heuristic top-k spreading algorithm. Furthermore, we show that the selected night bus route with our proposed approach is better than a newly started night bus route-in-service in Hangzhou, China.

For this work, we consider the effective design of only one bus route. In the future, we plan to broaden and deepen this work in several directions. First, we attempt to investigate the optimal bus route design with more real-life assumptions. For example, for the bus stop identification, the grid cells in geographical proximity might not be walkable due to physical barriers; for bidirectional bus route selection, one-way routes should be excluded or changed in actual design. Second, we also plan to explore the issue of designing more than one night bus route in an optimal way. Third, we would like to develop practical systems leveraging on taxi GPS traces, enabling a series of pervasive smart transportation services.

Acknowledgment

The authors would like to thank the anonymous reviewers and the editors for their helpful comments and suggestions. The authors also would like to thank S. Li and Z. Wang for providing the taxi GPS data set and L. Sun, P. Castro, T. Atmaca, Z. Wang, and Z. Zhou for their valuable inputs. C. Chen was supported by the China Scholarship Council for his Ph.D. study.

References

[1] *Public Transportation Factbook*, American Public Transportation Association, Washington, DC, USA, 2011, Technical Report.
[2] "Zhejiang Online News," 2013. [Online]. Available: http://biz.zjol.com.cn/05biz/system/2013/01/25/019113078.shtml
[3] C.-N. Anagnostopoulos, I. Anagnostopoulos, V. Loumos, and E. Kayafas, "A license plate-recognition algorithm for intelligent transportation system applications," *IEEE Trans. Intell. Transp. Syst.*, vol. 7, no. 3, pp. 377–392, Sep. 2006.
[4] D. L. Applegate, R. E. Bixby, V. Chvatal, and W. J. Cook, *The Traveling Salesman Problem: A Computational Study*. Princeton, NJ, USA: Princeton Univ. Press, 2007.
[5] J. Aslam, S. Lim, and X. Pan, "City-scale traffic estimation from a roving sensor network," in *Proc. ACM SenSys*, 2012, pp. 141–154.
[6] R. K. Balan, K. X. Nguyen, and L. Jiang, "Real-time trip information service for a large taxi fleet," in *Proc. MobiSys*, 2011, pp. 99–112.
[7] F. Bastani, Y. Huang, X. Xie, and J. W. Powell, "A greener transportation mode: Flexible routes discovery from GPS trajectory data," in *Proc. GIS*, 2011, pp. 405–408.
[8] S. Borzsony, D. Kossmann, and K. Stocker, "The skyline operator," in *Proc. ICDE*, 2001, pp. 421–430.
[9] P. S. Castro, D. Zhang, C. Chen, S. Li, and G. Pan, "From taxi GPS traces to social and community dynamics: A survey," *ACM Comput. Surveys*, vol. 46, no. 2, pp. 17:1–17:34, Jun. 2014.
[10] P. S. Castro, D. Zhang, and S. Li, "Urban traffic modelling and prediction using large scale taxi GPS traces," in *Proc. Pervasive Comput.*, 2012, pp. 57–72.
[11] A. Ceder and N. Wilson, "Bus network design," *Transp. Res. B, Methodol.*, vol. 20, no. 4, pp. 331–344, Aug. 1986.
[12] S. Chawla, Y. Zheng, and J. Hu, "Inferring the root cause in road traffic anomalies," in *Proc. IEEE ICDM*, 2012, pp. 141–150.
[13] C. Chen, D. Zhang, P. Castro, N. Li, L. Sun, S. Li, and Z. Wang, "iBOAT: Isolation-based online anomalous trajectory detection," *IEEE Trans. Intell. Transp. Syst.*, vol. 14, no. 2, pp. 806–818, Jun. 2013.
[14] C. Chen, D. Zhang, Z.-H. Zhou, N. Li, T. Atmaca, and S. Li, "B-planner: Night bus route planning using large-scale taxi GPS traces," in *Proc. IEEE PerCom*, 2013, pp. 225–233.
[15] T. A. Chua, "The planning of urban bus routes and frequencies: A survey," *Transportation*, vol. 12, no. 2, pp. 147–172, Jan. 1984.
[16] P. M. d'Orey, "Empirical evaluation of a dynamic and distributed taxi-sharing system," in *Proc. IEEE Conf. ITS*, 2012, pp. 140–146.
[17] Y. Ge, H. Xiong, C. Liu, and Z.-H. Zhou, "A taxi driving fraud detection system," in *Proc. IEEE ICDM*, 2011, pp. 181–190.

[18] Y. Ge, H. Xiong, A. Tuzhilin, K. Xiao, M. Gruteser, and M. Pazzani, "An energy-efficient mobile recommender system," in *Proc. ACM SIGKDD*, 2010, pp. 899–908.
[19] V. Guihaire and J.-K. Hao, "Transit network design and scheduling: A global review," *Transp. Res. A, Policy Pract.*, vol. 42, no. 10, pp. 1251–1273, Dec. 2008.
[20] A. Hofleitner, R. Herring, P. Abbeel, and A. Bayen, "Learning the dynamics of arterial traffic from probe data using a dynamic Bayesian network," *IEEE Trans. Intell. Transp. Syst.*, vol. 13, no. 4, pp. 1679–1693, Dec. 2012.
[21] H. Hu, Z. Wu, B. Mao, Y. Zhuang, J. Cao, and J. Pan, "Pick-up tree based route recommendation from taxi trajectories," in *Proc. Web-Age Inf. Manage.*, 2012, vol. 7418, pp. 471–483.
[22] S. Jerby and A. Ceder, "Optimal routing design for shuttle bus service," *Transp. Res. Rec., J. Transp. Res. Board*, no. 1971, pp. 14–22, 2006.
[23] S. Kim, S. Shekhar, and M. Min, "Contraflow transportation network reconfiguration for evacuation route planning," *IEEE Trans. Knowl. Data Eng.*, vol. 20, no. 8, pp. 1115–1129, Aug. 2008.
[24] B. Li, D. Zhang, L. Sun, C. Chen, S. Li, G. Qi, and Q. Yang, "Hunting or waiting? Discovering passenger-finding strategies from a large-scale real-world taxi dataset," in *Proc. IEEE PerCom Workshop*, 2011, pp. 63–68.
[25] Q. Li, Z. Zeng, T. Zhang, J. Li, and Z. Wu, "Path-finding through flexible hierarchical road networks: An experiential approach using taxi trajectory data," *Int. J. Appl. Earth Observ. Geoinf.*, vol. 13, no. 1, pp. 110–119, Feb. 2011.
[26] C.-L. Liu, T.-W. Pai, C.-T. Chang, and C.-M. Hsieh, "Path-planning algorithms for public transportation systems," in *Proc. IEEE Conf. ITS*, 2001, pp. 1061–1066.
[27] L. Liu, C. Andris, and C. Ratti, "Uncovering cab drivers' behavior patterns from their digital traces," *Comput., Environ. Urban Syst.*, vol. 34, no. 6, pp. 541–548, Nov. 2010.
[28] W. Liu, Y. Zheng, S. Chawla, J. Yuan, and X. Xing, "Discovering spatio-temporal causal interactions in traffic data streams," in *Proc. ACM SIGKDD*, 2011, pp. 1010–1018.
[29] Y. Liu, F. Wang, Y. Xiao, and S. Gao, "Urban land uses and traffic "source-sink areas": Evidence from GPS-enabled taxi data in Shanghai," *Landscape Urban Planning*, vol. 106, no. 1, pp. 73–87, May 2012.
[30] S. Ma, Y. Zheng, and O. Wolfson, "T-share: A large-scale dynamic taxi ridesharing service," in *Proc. ICDE*, 2013, pp. 410–421.
[31] M. H. Baaj and H. S. Mahmassani, "TRUST: A Lisp program for the analysis of transit route configurations," *Transp. Res. Rec.*, no. 1283, pp. 125–135, 1990.
[32] G. F. Newell, "Some issues relating to the optimal design of bus routes," *Transp. Sci.*, vol. 13, no. 1, pp. 20–35, Feb. 1979.
[33] G. Pan, G. Qi, Z. Wu, D. Zhang, and S. Li, "Land-use classification using taxi GPS traces," *IEEE Trans. Intell. Transp. Syst.*, vol. 14, no. 1, pp. 113–123, Mar. 2013.
[34] S. Pattnaik, S. Mohan, and V. Tom, "Urban bus transit route network design using genetic algorithm," *J. Transp. Eng.*, vol. 124, no. 4, pp. 368–375, Jul. 1998.
[35] J. Powell, Y. Huang, F. Bastani, and M. Ji, "Towards reducing taxicab cruising time using spatio-temporal profitability maps," in *Proc. Adv. Spatial Temporal Databases*, 2011, vol. 6849, pp. 242–260.
[36] L. Sun, D. Zhang, C. Chen, P. S. Castro, S. Li, and Z. Wang, "Real time anomalous trajectory detection and analysis," *Mobile Netw. Appl.*, vol. 18, no. 3, pp. 341–356, Jun. 2013.
[37] W. Szeto and Y. Wu, "A simultaneous bus route design and frequency setting problem for Tin Shui Wai, Hong Kong," *Eur. J. Oper. Res.*, vol. 209, no. 2, pp. 141–155, Mar. 2011.
[38] F.-Y. Wang, "Driving into the future with ITS," *IEEE Intell. Syst.*, vol. 21, no. 3, pp. 94–95, Jan./Feb. 2006.
[39] F.-Y. Wang, "Parallel control and management for intelligent transportation systems: Concepts, architectures, and applications," *IEEE Trans. Intell. Transp. Syst.*, vol. 11, no. 3, pp. 630–638, Sep. 2010.
[40] Z. Wang and J. Crowcroft, "Analysis of shortest-path routing algorithms in a dynamic network environment," *SIGCOMM Comput. Commun. Rev.*, vol. 22, no. 2, pp. 63–71, Apr. 1992.
[41] H. Wen Chang, Y. Chin Tai, and J. Y. Jen Hsu, "Context-aware taxi demand hotspots prediction," *Int. J. Bus. Intell. Data Mining*, vol. 5, no. 1, pp. 3–18, Dec. 2010.
[42] J. Yuan, Y. Zheng, and X. Xie, "Discovering regions of different functions in a city using human mobility and POIs," in *Proc. ACM SIGKDD*, 2012, pp. 186–194.
[43] J. Yuan, Y. Zheng, X. Xie, and G. Sun, "T-drive: Enhancing driving directions with taxi drivers' intelligence," *IEEE Trans. Knowl. Data Eng.*, vol. 25, no. 1, pp. 220–232, Jan. 2013.
[44] N. J. Yuan, Y. Zheng, L. Zhang, and X. Xie, "T-finder: A recommender system for finding passengers and vacant taxis," *IEEE Trans. Knowl. Data Eng.*, vol. 25, no. 10, pp. 2390–2403, Oct. 2013.
[45] D. Zhang, B. Guo, and Z. Yu, "The emergence of social and community intelligence," *Computer*, vol. 44, no. 7, pp. 21–28, Jul. 2011.
[46] D. Zhang, N. Li, Z.-H. Zhou, C. Chen, L. Sun, and S. Li, "iBAT: Detecting anomalous taxi trajectories from GPS traces," in *Proc. UbiComp*, 2011, pp. 99–108.
[47] J. Zhang, F.-Y. Wang, K. Wang, W.-H. Lin, X. Xu, and C. Chen, "Data-driven intelligent transportation systems: A survey," *IEEE Trans. Intell. Transp. Syst.*, vol. 12, no. 4, pp. 1624–1639, Dec. 2011.
[48] Z. Zhang, D. Yang, T. Zhang, Q. He, and X. Lian, "A study on the method for cleaning and repairing the probe vehicle data," *IEEE Trans. Intell. Transp. Syst.*, vol. 14, no. 1, pp. 419–427, Mar. 2013.
[49] F. Zhao and X. Zeng, "Optimization of transit route network, vehicle headways and timetables for large-scale transit networks," *Eur. J. Oper. Res.*, vol. 186, no. 2, pp. 841–855, Apr. 2008.
[50] Y. Zheng, Y. Liu, J. Yuan, and X. Xie, "Urban computing with taxicabs," in *Proc. UbiComp*, 2011, pp. 89–98.

Chao Chen received the B.Sc. and M.Sc. degrees in control science and control engineering from Northwestern Polytechnical University, Xi'an, China, in 2007 and 2010, respectively. He is currently working toward the Ph.D. degree at Pierre and Marie Curie University, Paris, France, and Institut Mines-Telecom/Telecom SudParis, Evry, France.

In 2009, he was a Research Assistant with The Hong Kong Polytechnic University, Kowloon, Hong Kong. His research interests include pervasive computing, social network analysis, and data mining from large-scale taxi data.

Mr. Chen was a corecipient of the Best Paper Runner Up Award at MobiQuitous 2011.

Daqing Zhang received the Ph.D. degree from the University of Rome "La Sapienza," Rome, Italy, and the University of L'Aquila, L'Aquila, Italy in 1996.

He is currently a Professor with Institut Mines-Telecom/Telecom SudParis, Evry, France. He has authored/coauthored over 180 referred journal and conference papers. His research interests include large-scale data mining, urban computing, context-aware computing, and ambient assistive living. All his research have been motivated by practical applications in digital cities, mobile social networks, and elderly care.

Dr. Zhang is the Associate Editor of four journals, including *ACM Transactions on Intelligent Systems and Technology*. He has been a frequently invited speaker in various international events on ubiquitous computing. He was the recipient of the Ten Years CoMoRea Impact Paper Award at IEEE PerCom 2013, the Best Paper Award at IEEE UIC 2012, and the Best Paper Runner Up Award at Mobiquitous 2011.

Nan Li received the M.Sc. degree in computer science in 2008 from Nanjing University, Nanjing, China,where he is currently working toward the Ph.D. degree in the Department of Computer Science and Technology.

Since 2008, he has been a Faculty Member with the School of Mathematical Sciences, Soochow University, Suzhou, China. His research interests are mainly in machine learning, data mining, and ambient intelligence.

Mr. Li was a corecipient of the Grand Prize (Open Category) in the PAKDD 2012 Data Mining Competition. His coauthored paper won the Best Paper Runner Up Award at MobiQuitous 2011. He was awarded with an IBM Ph.D. Fellowship in 2013–2014 and a Baidu Fellowship in 2013–2014.

Zhi-Hua Zhou (S'00–M'01–SM'06–F'13) received the B.Sc. (with the highest honors), M.Sc. (with the highest honors), and Ph.D. (with the highest honors) degrees from Nanjing University, Nanjing, China, in 1996, 1998, and 2000, respectively, all in computer science.

In 2001, he joined, as an Assistant Professor, the Department of Computer Science and Technology, Nanjing University, where he is is currently a Professor and the Director of the Learning And Mining from Data Group. His research interests are mainly in artificial intelligence, machine learning, data mining, pattern recognition, and multimedia information retrieval. In these areas, he has authored/coauthored over 90 papers in leading international journals or conference proceedings. He is a holder of 12 patents.

Mr. Zhou is a Fellow of the International Association of Pattern Recognition and the Institution of Engineering and Technology/Institution of Electrical Engineers and a Distinguished Scientist of the Association for Computing Machinery. He is the Chair of the Machine Learning Technical Committee of the Chinese Association for Artificial Intelligence, the Chair of the Artificial Intelligence and Pattern Recognition Technical Committee of the China Computer Federation, the Vice Chair of the Data Mining Technical Committee of the IEEE Computational Intelligence Society, and the Chair of the IEEE Computer Society Nanjing Chapter. He is the Founder and the Steering Committee Chair of the Asian Conference on Machine Learning (ACML) and a Steering Committee Member of the Pacific-Asia Conference on Knowledge Discovery and Data Mining (PAKDD) and the Pacific Rim International Conferences on Artificial Intelligence (PRICAI). He serves/ed as a General Chair/Cochair of ACML12, ADMA12, PCM13, and PAKDD14; a Program Chair/Cochair of PAKDD07, PRICAI08, ACML09, SDM13, and others; a Workshop Chair/Cochair of KDD12 and ICDM14; a Tutorial Chair/Cochair of KDD13 and CIKM14; and a Program Vice Chair or an Area Chair of various conferences such as ICML, IJCAI, AAAI, ICPR, etc. He is an Executive Editor-in-Chief of the *Frontiers of Computer Science*, an Associate Editor-in-Chief of the *Chinese Science Bulletin*, and an Associate Editor/Editorial Board Member of the ACM *Transactions on Intelligent Systems and Technology*, the IEEE TRANSACTIONS ON NEURAL NETWORKS AND LEARNING SYSTEMS, and many other journals. He served as an Associate Editor of the IEEE TRANSACTIONS ON KNOWLEDGE AND DATA ENGINEERING in 2008–2012 *and Knowledge and Information Systems* in 2003–2008. He has been a recipient of various awards/honors, including the National Science and Technology Award for Young Scholars of China, the Fok Ying Tung Young Professorship First-Grade Award, the Microsoft Young Professorship Award, and eight international journals/conferences paper awards and competition awards.