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 bi-directional 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 as well as 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 Bi-directional Probability based Spreading (*BPS*) algorithm to generate candidate bus routes automatically. We finally select the best bi-directional 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 7,600 taxis in a month.**

***Index Terms*—Taxi GPS Traces; Human Mobility Patterns; Route Graph; Bus Route Planning**

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 intra-city 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, especially 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, 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 time 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 bi-directional night-bus route design problem leveraging the taxi GPS traces. This problem can be divided into two sub-problems: the candidate bus stop identification and the best bi-directional bus route selection. For the first sub-problem, we need to identify the candidate bus stops which are associated with locations having 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 bi-directional 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 bi-directional night-bus route design becomes a problem of comparing the number of passengers of all valid bus routes giving certain time constraints.

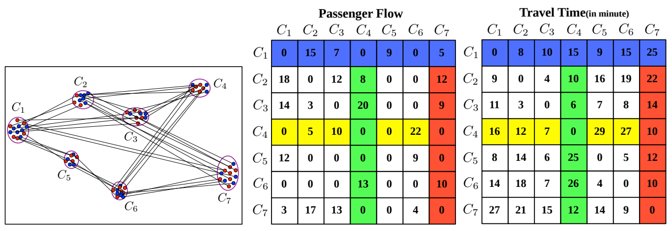
However, identifying the candidate bus stops from taxi GPS data and enumerating the top-ranked bi-directional bus routes efficiently are not trivial and straight-forward. To the best of our knowledge, there is still no work reported on this topic. For example, given the taxi GPS trajectories of night 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 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 of Fig. 1. The objective of bi-directional bus route design is to find a bi-directional bus

Chao Chen is with the CNRS UMR 5157 SAMOVAR, Institut Mines-TELECOM/TELECOM SudParis, Evry 91011, France, and also with the Department of Computer, Universite Pierre et Marie CURIE, 4 place Jussieu 75005 Paris, France (e-mail: chao.chen@telecom-sudparis.eu)

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

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

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


C_1
C_2
C_3
C_4
C_5
C_6
C_7

Passenger Flow

	C_1	C_2	C_3	C_4	C_5	C_6	C_7
C_1	0	15	7	0	9	0	5
C_2	18	0	12	8	0	0	12
C_3	14	3	0	20	0	0	9
C_4	0	5	10	0	0	22	0
C_5	12	0	0	0	0	9	0
C_6	0	0	0	13	0	0	10
C_7	3	17	13	0	0	4	0

Travel Time(in minute)

	C_1	C_2	C_3	C_4	C_5	C_6	C_7
C_1	0	8	10	15	9	15	25
C_2	9	0	4	10	16	19	22
C_3	11	3	0	6	7	8	14
C_4	16	12	7	0	29	27	10
C_5	8	14	6	25	0	5	12
C_6	14	18	7	26	4	0	10
C_7	27	21	15	12	14	9	0

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

route ($C_1 \rightarrow C_7$ and $C_7 \rightarrow C_1$) with 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 pick-up/drop-off records (*PDRs*) than other areas, but there is no clear guideline about where the bus stops should be put.
- ***Trade off 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 to C_7), but this will take more travel time. Hence, a non-trivial trade-off 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 $C_1 \rightarrow C_2 \rightarrow C_3 \rightarrow C_4 \rightarrow C_7$, this 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 maximum accumulated number of passengers in two directions.

In this paper, we address the above-mentioned 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 bi-directional night bus route design problem leveraging the taxi GPS traces. To the best of our knowledge, this is the first work on bi-directional night bus route design exploiting the taxi travel speed, time, pick-up and drop-off information in large-scale, real-world taxi GPS traces.

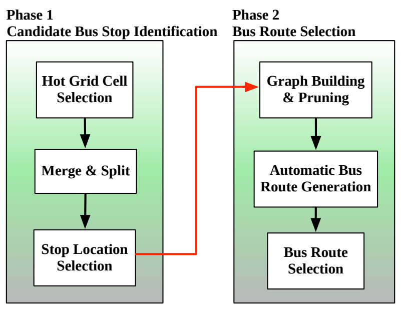


Fig. 2. The two-phase bus route planning framework.

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 Bi-directional Probability based Spreading (*BPS*) algorithm, to select the best bi-directional bus route which 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 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 the 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 the 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, while 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: where do people go throughout the day [24], [27], what are the "hottest" spots around a city [41], what are the "functions" of these hotspots [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. Whilst 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 driver's *modus operandi*. The aim is to learn from taxi drivers' expert knowledge of the city and detect abnormal or effective driving behaviors. The last two sub-categories used mainly the origin-destinations (OD) 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 sub-category which addresses "hotspots" 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 which 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 like $C_4 \rightarrow C_7$ of Fig. 1 in the planned bus routes, while 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 bi-directional bus route which can achieve the maximum number of passengers in both directions. In this paper, we attempt to address the bi-directional bus route planning problem. Besides 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, non-linear, non-convex, multi-objective 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 while the constraints include time, capacity and resources [11], [22], [15], [49].

However, the selection of the objectives should take care of the operator as well as user requirements which are often conflicting, leading to design trade-off rather than an optimal solution. As noted in [32], [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 also assumes 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 bi-directional night-bus route design problem leveraging the taxi passenger OD flow data. Different from existing research, our work aims to find a bi-directional 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 bi-directional 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*. In this section, 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 $10m \times 10m$ in size. In such a way, the whole city is partitioned into 5000×2500 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. And we name these grid cells as "hot" ones.

As each grid cell has maximum 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 equals to 0 is called isolated cell. As the city is composed of mixed hot grid cells and common grid cells, both hot cells and common cells form irregular "hot areas" and "common areas" 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* (Line1). 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 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 Eq. 1.

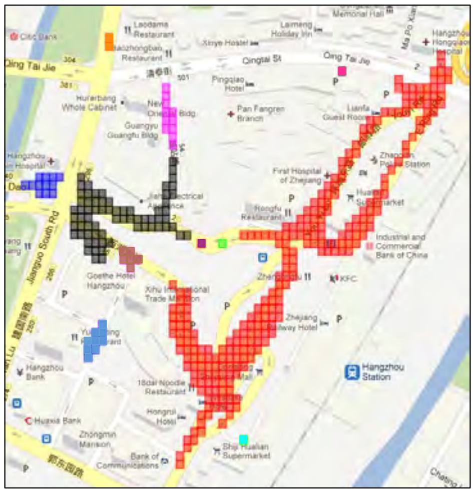


Fig. 3. City partitions near Hangzhou Railway Station.

Algorithm 1 Merge Algorithm

```
Input: List of partitions $\{P_i\}$
Output: List of clusters $\{C_i\}$
 1: $P \leftarrow sort\ (P)$, $(i = 1, 2, \cdots, n)$ // 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 $dist(C_i, P_j) < T_1$ 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
```

$$loc(P) = \frac{\sum_{i=1}^{N}(PDRs(g_i) * loc(g_i))}{\sum_{i=1}^{N} PDRs(g_i)} \quad (1)$$

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

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 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 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

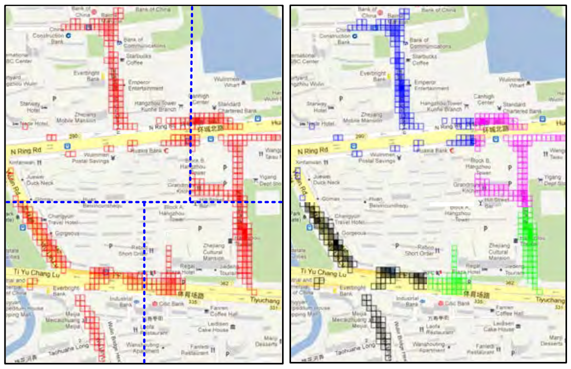


Fig. 4. Illustrative example of splitting. Big cluster formed via merging (left). Big cluster split into 4 walkable size clusters (right, in four different colors).

are willing to walk to reach a bus stop).

As for large clusters (Group 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 sub-clusters, aiming to minimize the difference of *PDRs* of the resulted clusters both in horizonal and vertical directions; while for clusters in Group 2, we only need to split the cluster in one direction. We split the cluster in the horizonal direction if its height is greater than width, otherwise, we split it in the vertical direction, again with the goal of minimizing the number difference of *PDRs* of the split sub-clusters. With one split, one big cluster would produce two smaller sub-clusters. Thus, a smaller T_2 would need more splitting times, and also leads to more smaller clusters finally.

Fig. 4 shows an illustrative example of splitting a cluster into four sub-clusters with the proposed splitting strategy. The initial cluster belongs to Group 1 (Fig. 4 (left)), the splitting is first done in the horizontal direction to produce two sub-clusters with similar *PDRs*. After the first splitting, two sub-clusters with width greater than T_2 are generated (T_2 is set to 500 meters), thus both sub-clusters require a further splitting in the vertical direction. The final result with four split sub-clusters 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 $T_2 \times T_2$, 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 connectivity degree (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 Eq. 2 in each cluster is selected as the "center" of the cluster, marked as the location of the candidate bus stop.

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

We set $w_1 = w_2 = 0.5$ in the evaluation, and totally we get 579 candidate bus stops in the city by using the taxi GPS data from Hangzhou, China. Note that different weight settings in Eq. 2 would only affect locations of the bus stop, and 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→D and D→O).

In this section, 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 (shown in 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 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 matrix, 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 ith cluster to jth cluster as each stop is responsible for its cluster. We set the maximum waiting time for passengers at the stop as 30 minutes (equal to the bus operation frequency), so any pick-up or drop-off events taking place in this time window are counted. We simply assume the passenger flows among candidate bus stops remain unchanged during each 30-minutes duration. The final *FM* is got by averaging all flow matrix at different bus frequencies. We also assume *TM* keeps unchanged across the night time. $tm(s_i, s_j)$ 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 *Ddist*$(s_i,s_j)/v$ to approximate $tm(s_i, s_j)$, where *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 is to ensure that the bus route would traverse intermediate stops and finally reach the destination within a limited time; the other 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

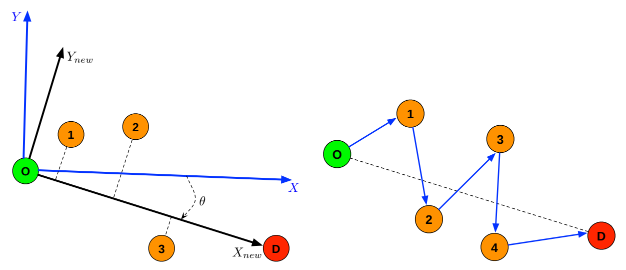


Fig. 5. Demonstration of Criterion 2 (left) and Criterion 5 (right).

achieve the objective of having the maximum number of passengers accumulated along the route. To meet the above 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 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 origin to destination using heuristic rules. These rules are either derived from one of the above two requirements, or from the intuitive bus route design principle. For instance, from the shortest travel time perspective, the bus route should extend from origin towards 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, also the bus route should not comprise sharp zig-zag paths. These can also be translated into two criteria in building the bus route graph. Specifically, given the OD pair (s_1, s_n) and the candidate route $R = \langle s_1, s_2, \cdots, s_n \rangle$, 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*

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

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*

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

$(x(i), y(i))$ of s_i is got by simply subtracting the longitude and latitude value 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 X-axis (see the left panel in Fig. 5). This criterion guarantees the bus will always move forward along the OD direction.

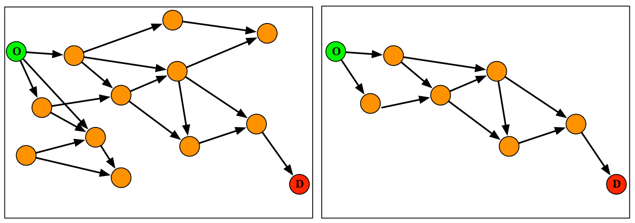


Fig. 6. A bus route directed graph for a given OD. The route graph is got by graph building algorithm (left) and its corresponding graph after applying graph pruning (right).

- *Criterion 3: Origin-farther*

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

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

- *Criterion 4: Destination-closer*

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

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

- *Criterion 5: No zigzag route*

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

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 of Fig. 5 should not happen, as it violates the *no zigzag route* criterion. We can see $\arg\min_{s_j}(dist(s_3, s_j)) = s_1 \neq s_2 \quad (j = 1, 2)$, also $\arg\min_{s_j}(dist(s_4, s_j)) = s_2 \neq s_3 \quad (j = 1, 2, 3)$.

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. While 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 firstly 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*∼*4*. The process will terminate when all stops have been checked. At last, stops having no edges would be excluded.

As *Criterion 5* is related to all stops in one bus route, so 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, so 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.

We first calculate all the nodes' in-coming and out-going degrees. Afterwards 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. At last 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 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* $R = \langle s_1, s_2, \cdots, s_n\rangle$ *is a candidate bus route for pair* (s_1, s_n)*, then its reversed route* $\bar{R} = \langle s_n, s_{n-1}, \cdots, s_1\rangle$ *will be the candidate bus route for* (s_n, s_1) *pair.*

Proof. To prove $\bar{R}$ is the candidate bus route for (s_n, s_1) pair, we just need to check whether it meets all the five criteria. It is obviously 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 are s_{i-1} and s_{i+1}. Thus $\arg\min_{s_j}(dist(s_i, s_j)) = s_{i+1}$ $(j = n, n-1, \cdots, i+1)$ will hold. □

C. Automatic Candidate Bus Route Generation

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

Probability based Spreading Algorithm: Though we have removed invalid nodes and edges through graph pruning, the problem of enumerating all possible routes from given source to destination is proved to be NP hard. Indeed, it is also unnecessary to enumerate all possible routes and compare them all, because most of 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) $Num(R_i) > Num(R_j)$. The route which 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 number of expected delivered passengers. We compute them based on Eqns. 3 and 4. The *skyline route* definition is similar to that in [18], and the rational 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*.

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

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

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

Algorithm 2 Probability based Spreading

```
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:   currentR = $s_1$
     //starts from the given origin $s_1$
4:   Choose the next stop $s_i^*$ with respect to currentR according to Eq. 5
5:   $R$ = currentR$\cdot s_i^*$
     //$\cdot$ operation appends $s_i$ to currentR
6:   Repeat Lines 4~5 Until $s_i^* = s_n$
     //ends at the 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 *probability based spreading 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 Eq. 5.

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

where $fm(s_m, s_i^*)$ 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 the selection of 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 stays 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 like in the *probability based spreading algorithm*, an intuitive way is to select top-k stops each time, where those k nodes should have highest accumulated passenger flow

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: $\mathcal{R}^*$: the set of skyline routes
1: $\mathcal{R} = \emptyset$
2: Repeat
3:    Run Line 2∼6 in Algorithm 2 for $G_{O\to D}(S,E)$, and the output is $R_{O\to D}$
4:    Run Line 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 $\mathcal{R}^*$ keeps unchanged
```

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 nodes would select another top-k nodes, thus the total candidate routes would be k^2. Assume 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 grows exponentially with the spreading step (n). We use this **top-*k* spreading** method as the baseline.

Bi-directional Probability based Spreading (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 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 which has overall maximum expected number of passengers in both directions, we propose the *BPS* algorithm, whose basic idea is to run the *probability based spreading 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 (Line 3∼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 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).

D. Bus Route Selection

Given the bus operation frequency (once every 30 minutes), 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 which 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.

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

V. Experimental Evaluation

In this section, we validate the proposed approach with a large-scale real-world taxi GPS dataset which is generated from 7,600 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 PC with 12-GB RAM 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 got by *k*-means is the average location of all its members, and it may fall into non-reachable 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 is 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 got by *k*-means is very different, while our proposed method generates candidate bus stops that are evenly distributed in the hot areas, which better meets the commonsense design criteria of bus stops.

B. Evaluation on Bus Route Selection Algorithm

We first show the convergence of the proposed algorithm, and 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 Algorithm 2 and 3, our proposed bus route generation process would be terminated if the resulted *skyline routes* keep unchanged. We study the similarity of consecutively generated *skyline routes* from 5,000 to 150,000 runs, with a constant interval of 5,000 runs. We measure the similarity (sim) of two sets A and B as follows:

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

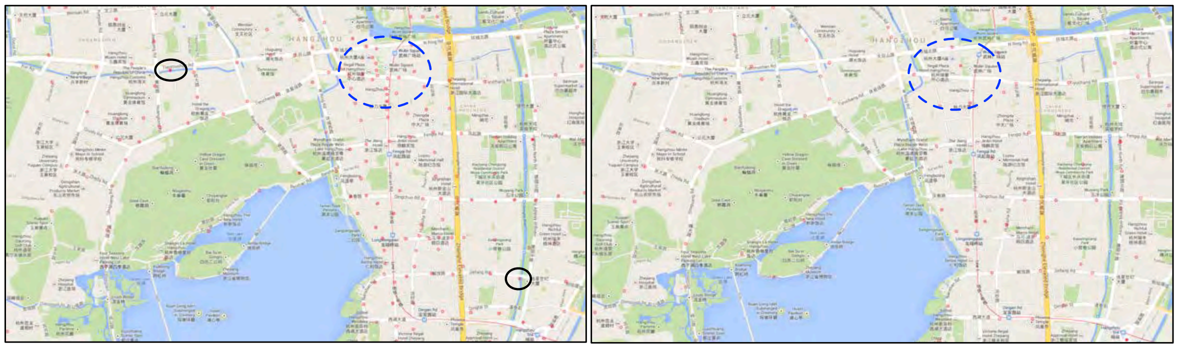


Fig. 7. Comparison results with *k*-means (best viewed in the digital version). Results got by *k*-means (left) and results got by our method (right).

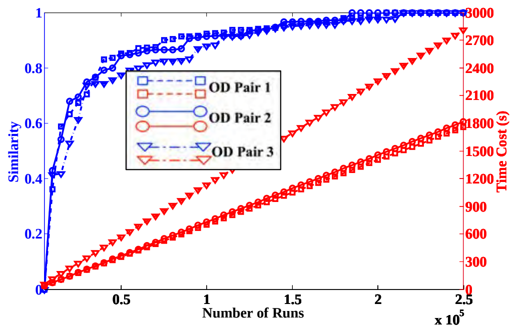


Fig. 8. Convergence study of the proposed *BPS* algorithm.

The similarity results of the consecutively generated skyline routes with a 5,000 run interval are shown in Fig. 8, and the time cost is put in the diagram as well. In this study, 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. Also 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 number of candidate bus stops, thus the spreading time and 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 9(b) show 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] meters) and T_2 ($\in$[400:50:650] meters) respectively. We also show the skyline route results under different T_1 and T_2 in Fig. 9(c) and 9(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 9(f) show 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 often result in lower number of passengers but higher time cost. Specifically, the minimum number of passengers and the maximum time cost occurs 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, at last more 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 the paper as it expects larger number of passengers while consuming relatively less time. Additionally, 500-meter 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, 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. And we also have $\mathcal{R}_{\delta_1} \subseteq \mathcal{R}_{\delta_2}$. Therefore, with the increase of δ value, 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 of 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 graph is simply quantified by the average In-

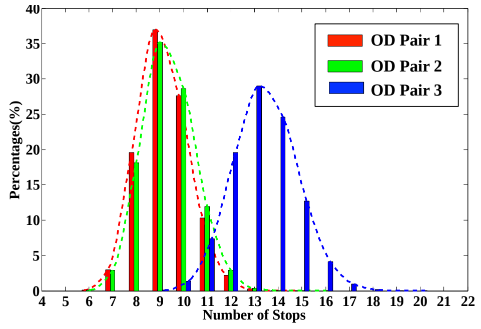


Fig. 10. The number of stops of candidate route stops statistics for 3 OD pairs.

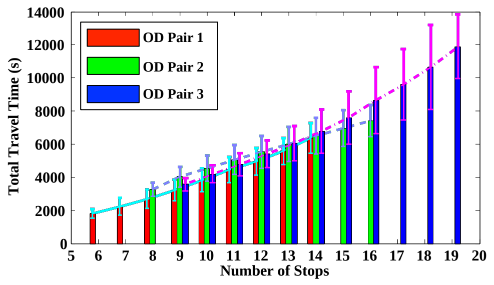


Fig. 11. The relationship between the number of stops and total travel time statistics for 3 OD pairs.

coming/Out-going degrees. They are equal to the ratio of the total number of edges to the total number of nodes in the route graph. From the figure, we can see that the average Incoming/Out-going degrees under 1.7 km is 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 the 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 4 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 4 stops.
3) For OD pair 3, most of the routes contain 10 to 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 distance of a given OD pair, the more stops the route would contain.

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 increases linearly with the number of stops, suggesting 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 the 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, while the y-axis represents the expected number of

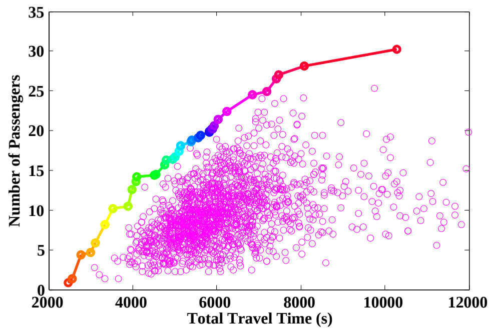


Fig. 12. Detected *skyline routes* and other candidate routes.

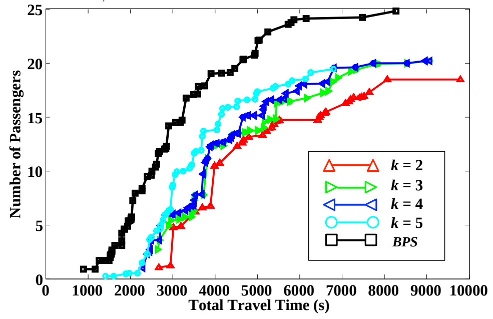


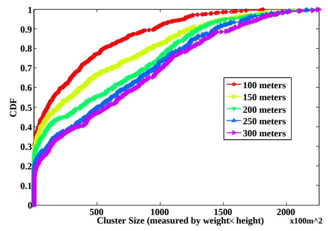
Fig. 13. Comparison results with baseline under different k values.

passengers. From 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 other two cases as well.

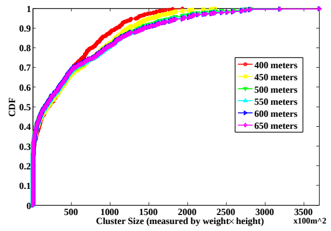
5) Comparison with top-k spreading algorithm: In the top-k spreading algorithm, the selection of k is vital to the skyline routes generated as well as the time needed to generate all the candidate routes. In particular, when $k_1 < k_2$, we have $\mathcal{R}_{k_1} \subseteq \mathcal{R}_{k_2}(k_1 \leq k_2)$. Theorem 2 guarantees that a bigger k would lead to a better set of skyline routes. However, the greater k also results in significant increase of 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.

C. Bidirectional vs Single Directional Bus Route

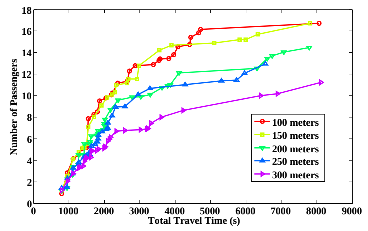
In real life, bus route got by Algorithm 2 may 1) be the *skyline route* in both directions; 2) be the *skyline route* in only one direction; 3) not be the *skyline route* in any direction. It is noteworthy to compare the overall best bidirectional bus route obtained by Algorithm 3 to the best routes in single direction. We have drawn all the seleted bus routes on the city digital map in Fig. 14 for the OD pair 3. They are different routes, which means the bidirectional bus route is neither the *skyline route* in the ZJU→East Railway Station direction, nor in the East Railway Station→ZJU direction. A reasonable explanation is that the passenger flow and the travel time among stops is often asymmetrical, and thus the bus route which carries the maximum number of passengers



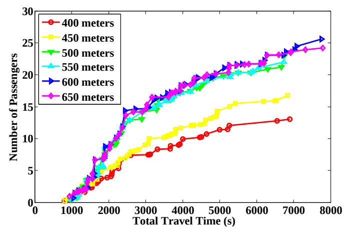
(a) CDF results of cluster size under different T_1 ($T_2 = 500\ m$).



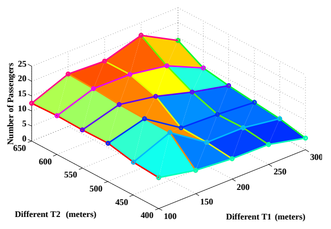
(b) CDF results of cluster size under different T_2 ($T_1 = 150\ m$).



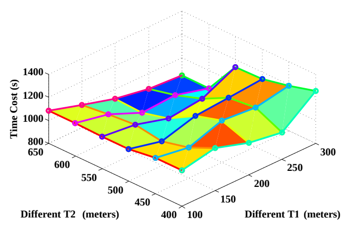
(c) Skyline route results under different T_1 ($T_2 = 450\ m$)



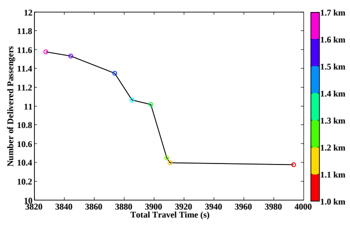
(d) Skyline route results under different T_2 ($T_1 = 150\ m$)



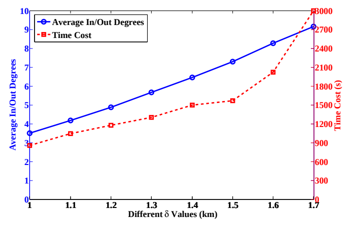
(e) The maximum number of passengers under different T_1 and T_2 combinations.



(f) Time cost under different T_1 and T_2 combinations.



(g) Selected bus routes at different δ.



(h) The route graph complexity and time cost under different δ.

Fig. 9. Results of parameter sensitivity study.

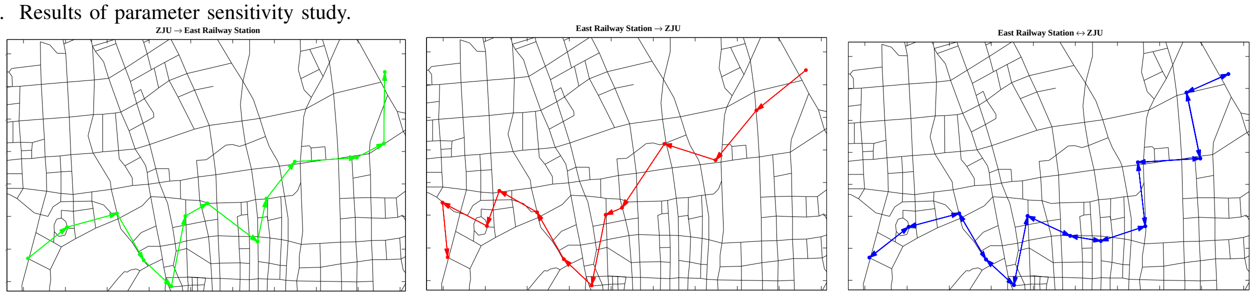

ZJU → East Railway Station
East Railway Station → ZJU
East Railway Station ↔ ZJU
Fig. 14. Comparison results of the selected bus routes in two directions to that in one direction. $R_{O\rightarrow D}$ (left); $R_{D\rightarrow O}$ (middle); $R_{O\leftrightarrow D}$ (right).

TABLE II
TWO METRICS OF THE SELECTED BUS ROUTES.

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

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, especially for the route $R_{O\rightarrow D}$ (left figure in Fig. 14) and $R_{O\leftrightarrow D}$ (right figure in Fig. 14). By further checking, we find that these common stops are popular night life 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 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 dataset 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 the user demands are obtained from expensive human survey. We first draw the newly started night-bus route R_3 on Google map as shown in Fig. 15 (left bottom), then we draw our proposed night-bus route R_1 in Fig. 15 (left top). 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 (left top). 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 (right top). 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.

We also compare our proposed best route R_1 with the

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

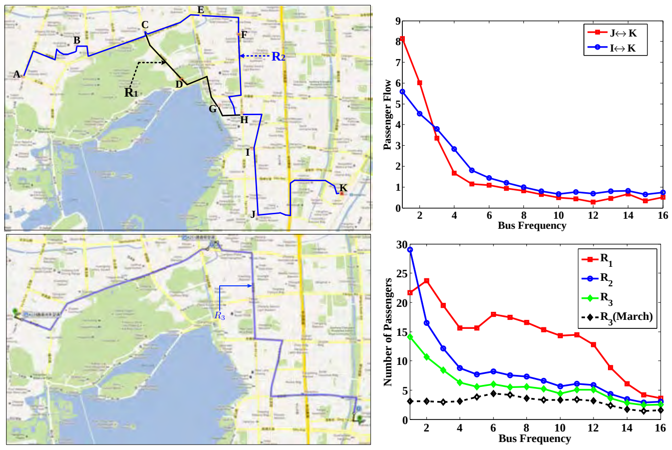


Fig. 15. Results comparison. Planned routes (top left); Passenger flow comparison of two segments at different frequency (top right); Opened night-bus route (bottom left); Number of delivered passengers at different frequency (R_1, R_2 and R_3, bottom right).

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 can be seen from Fig. 15 (right bottom) 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, as R_1 takes a much shorter route than R_2 and needs similar travel time as the newly started route R_3 does (shown in 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 as well as expected number of passengers.

It is understood that introducing of 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 the right bottom subfigure of Fig. 15. 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.

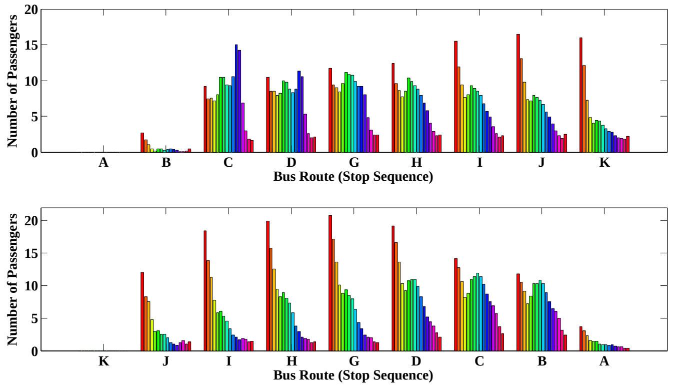


Fig. 16. The number of passengers on the bus before reaching the stop for OD pair 1.

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 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. Besides, we also have the following three observations:

1) More passengers are often expected in both directions for the first operation frequency, except for the 11th and 12th frequencies when the bus runs from C to D.
2) Buses running close to the capacity only last for 3 stops (from A to K) or 4 stops (from K to A).
3) Night buses heading towards different directions have quite different passenger flow patterns.

VI. Conclusion and Future Work

In this paper, we have investigated the problem of bi-directional night-bus route design by leveraging the taxi GPS traces. The 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 cluster as the candidate bus stop. In the second phase, given the bus route origin, destination, candidate bus stops as well as 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 which expects the maximum number of passengers under the given conditions. On a real-world dataset 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 as well as its superior performance over the heuristic top-k spreading algorithm. Further more, we show 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 bi-directional 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.

Acknowledgments

The authors want to thank the anonymous reviewers and the editors for their helpful comments and suggestions. The authors would like to thank Shijian Li and Zonghui Wang for providing the taxi GPS data set, and also thank Lin Sun, Pablo Castro, Tulin Atmaca, Zhu Wang and Zhangbing Zhou for their valuable inputs. Chao Chen was supported by China Scholarship Council (CSC) for his PhD study. This research was partly supported by the Institut Mines-TElECOM through the "Futur et ruptures" Program, and the National Science Foundation of China (61333014, 61105043).

References

[1] Public transportation factbook. Technical report, American Public Transportation Association, 2011.

[2] Zhejiang online news. http://biz.zjol.com.cn/05biz/system/2013/01/25/019113078.shtml, 2013.

[3] C.-N. Anagnostopoulos, I. Anagnostopoulos, V. Loumos, and E. Kayafas. A license plate-recognition algorithm for intelligent transportation system applications. *IEEE Transactions on Intelligent Transportation Systems*, 7(3):377–392, 2006.

[4] D. L. Applegate, R. E. Bixby, V. Chvatal, and W. J. Cook. *The Traveling Salesman Problem: A Computational Study*. Princeton University Press, 2007.

[5] J. Aslam, S. Lim, and X. Pan. City-scale traffic estimation from a roving sensor network. In *Proc. of ACM SenSys*, 2012.

[6] R. K. Balan, K. X. Nguyen, and L. Jiang. Real-time trip information service for a large taxi fleet. In *Proc. of MobiSys*, pages 99–112, 2011.

[7] F. Bastani, Y. Huang, X. Xie, and J. W. Powell. A greener transportation mode: flexible routes discovery from GPS trajectory data. In *Proc. of GIS*, pages 405–408, 2011.

[8] S. Borzsony, D. Kossmann, and K. Stocker. The skyline operator. In *Proc. of ICDE*, pages 421–430, 2001.

[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 Computing Surveys*, 46(2):17:1–17:34, 2013.

[10] P. S. Castro, D. Zhang, and S. Li. Urban traffic modelling and prediction using large scale taxi GPS traces. In *Proc. Pervasive Computing*, pages 57–72, 2012.

[11] A. Ceder and N. Wilson. Bus network design. *Transportation Research Part B: Methodological*, 20(4):331–344, 1986.

[12] S. Chawla, Y. Zheng, and J. Hu. Inferring the root cause in road traffic anomalies. In *Proc. of ICDM*, pages 141–150, 2012.

[13] C. Chen, D. Zhang, P. Castro, N. Li, L. Sun, S. Li, and Z. Wang. iBOAT: Isolation-based online anomalous trajectory detection. *IEEE Transactions on Intelligent Transportation Systems*, 14(2):806–818, 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. of PerCom*, pages 225–233, 2013.

[15] T. A. Chua. The planning of urban bus routes and frequencies: A survey. *Transportation*, 12(2):147–172, 1984.

[16] d'Orey and P. M. Empirical evaluation of a dynamic and distributed taxi-sharing system. In *Proc. of IEEE Conference on ITS*, pages 140–146, 2012.

[17] Y. Ge, H. Xiong, C. Liu, and Z.-H. Zhou. A taxi driving fraud detection system. In *Proc. of ICDM*, pages 181–190, 2011.

[18] Y. Ge, H. Xiong, A. Tuzhilin, K. Xiao, M. Gruteser, and M. Pazzani. An energy-efficient mobile recommender system. In *Proc. of ACM SIGKDD*, pages 899–908, 2010.

[19] V. Guihaire and J.-K. Hao. Transit network design and scheduling: A global review. *Transportation Research Part A: Policy and Practice*, 42(10):1251 – 1273, 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 Transactions on Intelligent Transportation Systems*, (99):1–15, 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 *Web-Age Information Management*, volume 7418, pages 471–483. 2012.

[22] S. Jerby and A. Ceder. Optimal routing design for shuttle bus service. *Transportation Research Record: Journal of the Transportation Research Board*, 1971:14–22, 2006.

[23] S. Kim, S. Shekhar, and M. Min. Contraflow transportation network reconfiguration for evacuation route planning. *IEEE Transactions on Knowledge and Data Engineering*, 20(8):1115–1129, 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. of PerCom Workshop*, pages 63 –68, 2011.

[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. *International Journal of Applied Earth Observation and Geoinformation*, 13(1):110 – 119, 2011.

[26] C.-L. Liu, T.-W. Pai, C.-T. Chang, and C.-M. Hsieh. Path-planning algorithms for public transportation systems. In *Proc. of IEEE Conference on ITS*, pages 1061–1066, 2001.

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

[28] W. Liu, Y. Zheng, S. Chawla, J. Yuan, and X. Xing. Discovering spatio-temporal causal interactions in traffic data streams. In *Proc. of ACM SIGKDD*, pages 1010–1018, 2011.

[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 and Urban Planning*, 106(1):73 – 87, 2012.

[30] S. Ma, Y. Zheng, and O. Wolfson. T-share: A large-scale dynamic taxi rideesharing service. In *Proc. of ICDE*, 2013.

[31] B. M.H. and M. H.S. TRUST: A lisp program for the analysis of transit route configurations. *Transportation Research Record*, 1283:125–135, 1990.

[32] G. F. Newell. Some issues relating to the optimal design of bus routes. *Transportation Science*, 13(1):20–35, 1979.

[33] G. Pan, G. Qi, Z. Wu, D. Zhang, and S. Li. Land-use classification using taxi GPS traces. *IEEE Transactions on Intelligent Transportation Systems*, 14(1):113–123, 2013.

[34] S. Pattnaik, S. Mohan, and V. Tom. Urban bus transit route network design using genetic algorithm. *Journal of Transportation Engineering*, 124(4):368–375, 1998.

[35] J. Powell, Y. Huang, F. Bastani, and M. Ji. Towards reducing taxicab cruising time using spatio-temporal profitability maps. In *Advances in Spatial and Temporal Databases*, volume 6849, pages 242–260. 2011.

[36] L. Sun, D. Zhang, C. Chen, P. S. Castro, S. Li, and Z. Wang. Real time anomalous trajectory detection and analysis. *Mobile Networks and Applications*, 18(3):341–356, 2013.

[37] W. Szeto and Y. Wu. A simultaneous bus route design and frequency setting problem for Tin Shui Wai, Hong Kong. *European Journal of Operational Research*, 209(2):141 – 155, 2011.

[38] F.-Y. Wang. Driving into the future with ITS. *IEEE Intelligent Systems*, 21(3):94–95, 2006.

[39] F.-Y. Wang. Parallel control and management for intelligent transportation systems: Concepts, architectures, and applications. *IEEE Transactions on Intelligent Transportation Systems*, 11(3):630–638, 2010.

[40] Z. Wang and J. Crowcroft. Analysis of shortest-path routing algorithms in a dynamic network environment. *SIGCOMM Comput. Commun. Rev.*, 22(2):63–71, 1992.

[41] H. wen Chang, Y. chin Tai, and J. Y. jen Hsu. Context-aware taxi demand hotspots prediction. *International Journal of Business Intelligence and Data Mining*, 5(1):3–18, 2010.

[42] J. Yuan, Y. Zheng, and X. Xie. Discovering regions of different functions in a city using human mobility and POIs. In *Proc. of ACM SIGKDD*, pages 186–194, 2012.

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

[44] N. J. Yuan, Y. Zheng, L. Zhang, and X. Xie. T-finder: A recommender system for finding passengers and vacant taxis. *IEEE Transactions on Knowledge and Data Engineering*, page to appear, 2013.

[45] D. Zhang, B. Guo, and Z. Yu. The emergence of social and community intelligence. *IEEE Computer*, 44(7):21–28, 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. of UbiComp*, pages 99–108, 2011.

[47] J. Zhang, F.-Y. Wang, K. Wang, W.-H. Lin, X. Xu, and C. Chen. Data-driven intelligent transportation systems: A survey. *IEEE Transactions on Intelligent Transportation Systems*, 12(4):1624–1639, 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 Transactions on Intelligent Transportation Systems*, 14(1):419–427, 2013.

[49] F. Zhao and X. Zeng. Optimization of transit route network, vehicle headways and timetables for large-scale transit networks. *European Journal of Operational Research*, 186(2):841 – 855, 2008.

[50] Y. Zheng, Y. Liu, J. Yuan, and X. Xie. Urban computing with taxicabs. In *Proc. of UbiComp*, pages 89–98, 2011.

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 with Pierre and Marie Curie University, Paris, France, and with Institut Mines-TELECOM/TELECOM SudParis, Evry, France. Mr. Chen was a co-recipient of the Best Paper Runner-Up Award at MobiQuitous 2011.

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

Daqing Zhang is a professor at Institut Mines-TELECOM/TELECOM SudParis, France. He obtained his Ph.D from University of Rome "La Sapienza" and the University of L'Aquila, Italy in 1996. His research interests include large-scale data mining, urban computing, context-aware computing, and ambient assistive living. He has published more than 180 referred journal and conference papers, all his research has been motivated by practical applications in digital cities, mobile social networks and elderly care.

Dr. Zhang is the Associate Editor for four journals including *ACM Transactions on Intelligent Systems and Technology*. He has been a frequent Invited Speaker in various international events on ubiquitous computing. He is the winner 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 MSc degree in computer science from Nanjing University, China, in 2008. At the same year, he became a faculty member in the School of Mathematical Sciences at Soochow University, China. He is currently pursuing the PhD degree in the Department of Computer Science & Technology of Nanjing University. His research interests are mainly in machine learning, data mining and ambient intelligence. He and other LAMDA members won the Grand Prize (Open Category) in the PAKDD 2012 Data Mining Competition. His co-authored paper won the Best Paper Runner-Up Award at MobiQuitous 2011. He has been awarded with the IBM PhD Fellowship (2013-2014) and the Baidu Fellowship (2013-2014).

Zhi-Hua Zhou (S00-M01-SM06-F13) received the BSc, MSc and PhD degrees in computer science from Nanjing University, China, in 1996, 1998 and 2000, respectively, all with the highest honors. He joined the Department of Computer Science & Technology at Nanjing University as an assistant professor in 2001, and is currently professor and Director of the LAMDA group. His research interests are mainly in artificial intelligence, machine learning, data mining, pattern recognition and multimedia information retrieval. In these areas he has published over 90 papers in leading international journals or conference proceedings, and holds 12 patents. He has won various awards/honors including the National Science & Technology Award for Young Scholars of China, the Fok Ying Tung Young Professorship 1st-Grade Award, the Microsoft Young Professorship Award and eight international journals/conferences paper awards and competition awards.

He is an Executive Editor-in-Chief of the *Frontiers of Computer Science*, Associate Editor-in-Chief of the *Chinese Science Bulletin*, Associate Editor or editorial boards member of the *ACM Transactions on Intelligent Systems and Technology*, *IEEE Transactions on Neural Networks and Learning Systems*, and many other journals. He served as Associate Editor for *IEEE Transactions on Knowledge and Data Engineering* (2008-2012) and *Knowledge and Information Systems* (2003-2008). He is the founder and Steering Committee Chair of ACML, and Steering Committee member of PAKDD and PRICAI. He serves/ed as General Chair/Co-chair of ACML12, ADMA12, PCM13, PAKDD14, Program Chair/Co-Chair of PAKDD07, PRICAI08, ACML09, SDM13, etc., Workshop Chair/Co-Chair of KDD12 and ICDM14, Tutorial Chair/Co-Chair of KDD13 and CIKM14, and Program Vice Chair or Area Chair of various conferences such as ICML, IJCAI, AAAI, ICPR, etc. He is the Chair of the Machine Learning Technical Committee of the Chinese Association of Artificial Intelligence, Chair of the Artificial Intelligence & Pattern Recognition Technical Committee of the China Computer Federation, Vice Chair of the Data Mining Technical Committee of IEEE Computational Intelligence Society and the Chair of the IEEE Computer Society Nanjing Chapter. He is an IEEE Fellow, IAPR Fellow, IET/IEE Fellow and ACM Distinguished Scientist.