Rapid Route Comparison Based on GPS Coordinates and Bounding Boxes

Shih-Hao Huang and Chow-Sing Lin
Dept. of Computer Science and Information Engineering
National University of Tainan, Taiwan
Email: mikelin@mail.nutn.edu.tw

***Abstract*—Dynamic real-time carpooling service needs to rapidly pair passengers with drivers if their routes are intersected on their ways to destinations. However, the previous studies comparing two routes on a gridded map take a long time and result in high errors. In this paper, we propose an accurate and rapid approach to compare two routes based on the concept of bounding boxes. In addition to the traditional brute force (BR) approach, we also developed three approaches using bounding boxes to compare two routes, which are 1-BB, n-BB, and t-BB. Our proposed approaches were successfully tested under 7626 pairs of the random generated routes on the eastern United States map. Furthermore, we also compared our approaches with Gjaldbæk's gridding method for route comparing. The experimental results show that our proposed approaches are much faster than BR and Gjaldbæk's approach, which can significantly improve the efficiency of pairing passengers with drivers in real-time carpooling systems.**

***Index Terms*—route comparison, carpooling, bounding box, navigation, global positioning system**

I. Introduction

Carpooling, or known as ride sharing, was successfully tested to have many benefits, such as reducing traffic jam, saving money, and protect environments. To succeed in pairing passengers with drivers in carpooling, we must compare their routes if they are intersected on the ways to destinations. The previous studies of comparing two routes on a gridded map not only take a long time but also result in high errors. A simple example of comparing two routes is shown in Fig. 1.

Many previous studies have been conducted on comparison of two routes based on a gridded map [1]-[4]. The intersected segments in routes can be determined by whether they are in the same grid. Apparently, the size of a grid will affect the accuracy of comparing two routes. A large size of a grid would probably result in high errors on the accuracy of comparing two routes. On the contrast, a small size of a grid would result in the high accuracy of comparing two routes at the cost of computation time.

In general, a route is composed of multiple segments, and every segment consists of many pairs of latitude and longitude coordinates. Comparing every coordinate in two routes by traditional brute force (BR) approach results in much higher accuracy than the approach comparing two routes on a gridded map. However, BR approach has high cost of computation time though it has no errors in the results of comparing two routes. To provide not only accurate but also rapid route matching, in this paper we apply the concept of bounding boxes [5], [6] and propose three route comparison algorithms, 1-BB, n-BB, and t-BB. The experimental results show that they are not only faster than BR but also more accurate than approaches with a gridded map on comparing two routes.

Figure 1. A simple example of comparing two routes and the intersected segment, where Si and Ei (i=1, 2) represent the beginning and the end of a route.

The rest of this paper is organized as follows. In Section II we review related works. In Section III we define a route and categories of intersected segments in two routes. In Section IV we define a bounding box of a route, and presents the algorithm for determining the driving direction in two routes and for comparing two routes. We evaluate the algorithms comparing two routes in Section V and report the results. Finally, we make conclusions and suggest the further research work in Section VI.

II. Related Work

There are many previous studies related to comparison of routes. Gjaldbæk [4] solved the problem of comparing two routes rasterized on a gridded map. In [7], [8] and [9], the map matching problem was discussed. In [10], [11] and [12], the trajectory pattern mining problem was discussed. In [1], [3], and [13], the problem of finding frequent routes was discussed. In [2], the problem of toward mobility-based clustering on a gridded map was discussed.

Manuscript received July 15, 2018; revised November 2, 2018.

doi: 10.18178/jtle.7.1.5-9

He *et al.* [3] mined the frequent route based on a gridded map in a carpooling service. They calculate the visit frequency of the segment in a route passed on a grid field and remove the segment if the frequency is less than a threshold. Cao et al. [1] also mined the frequent route based on a gridded map and substring tree. The disadvantages of the approach based on a gridded map in [1] and [3] are necessary to rasterize routes and match the grid fields. Therefore, the approach based on a gridded map not only requires a lot of memory space to store information of every segment of the routes but also takes a long time to mine the frequent route.

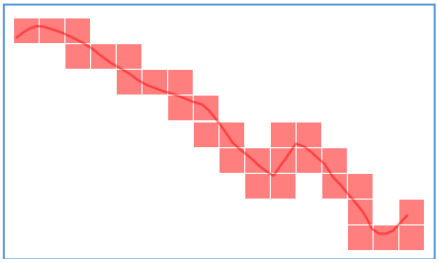

Figure 2. One of the example route and a flattened view of the sequence of grid fields to which it rasterizes with the size of a grid in 75 × 75 m^2. [4]

Gjaldbæk [4] developed an algorithm to solve the problem of comparing two routes on grid map. The algorithm is divided into three stages. The first stage is to rasterize two routes into grid fields shown in Fig. 2. The second stage is to match the grid fields derived from the previous stage. The last stage is to extract the solution from the matching results. In addition, the second stage uses a modified dynamic programming algorithm from approximate string matching. Finally, the intersected route is found by tracing backward from the bottom right to the top left cell in the matrix filled in the second stage. We would compare the efficiency of Gjaldbæk's approach with ours because both of us use the same path definition which is assuming that a path is defined by a sequence of GPS coordinates.

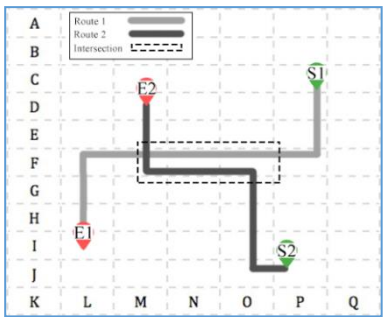

Figure 3. A large size of a grid causes that intersecting route at three grids.

However, the size of a grid would affect the result of comparing two routes. For example, there are two same routes on a map in the different size of a grid in Fig. 3 and Fig. 4. In Fig. 3, a large size of a grid causes that two routes are intersected at (M, F), (N, F), and (O, F) segments. In Fig. 4, a small size of a grid causes that two routes are intersected at only (P, E) segment. Obviously, the result of comparing two routes in Fig. 4 is more accurate than in Fig. 3. Apparently, a large size of a grid would probably result in high errors on the accuracy of comparing two routes. On the contrast, a small size of a grid would result in the high accuracy of comparing two routes at the cost of computation time

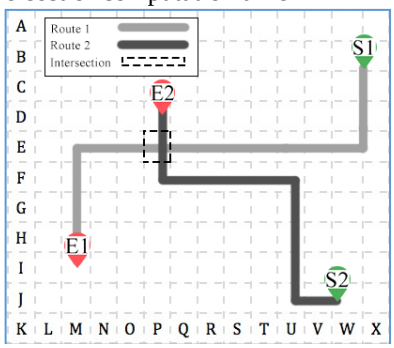

Figure 4. A small size of a grid causes that intersecting route at a grid.

In our proposed the 1-BB, n-BB, and t-BB route matching approaches would not only rapid but also generate no errors in the result of comparing two routes because these approaches use bounding boxes to initially filter unnecessary coordinates in two routes, which can save a lot of computation time, and then use BR approach to accurately compare the remaining coordinates one by one in two routes.

III. Intersection Routes

A route is composed of multiple segments from the beginning to the end, denoted as $\{s_1, \cdots, s_i\}$, where s_i consists of many pairs of latitude and longitude coordinates denoted as $\{p_1, \cdots, p_j\}$. Then, a route can be represented by a linked list, i.e., $r\{s_1 \rightarrow s_2 \rightarrow s_3 \rightarrow \cdots \rightarrow s_i\}$ or $r\{p_1 \rightarrow p_2 \rightarrow p_3 \rightarrow \cdots \rightarrow p_j\}$. We denote a coordinate as (lat, lng), where lat is latitude and lng is longitude. In this paper, A route r is represented by a sequence of coordinates as $r\{p_1 \rightarrow p_2 \rightarrow p_3 \rightarrow \cdots \rightarrow p_j\}$.

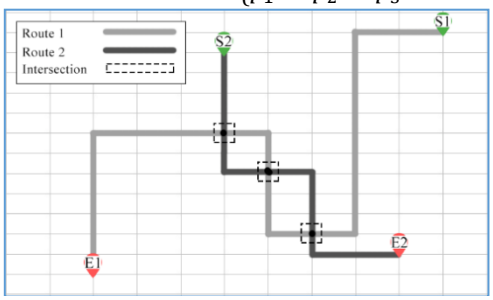

Figure 5. Two routes intersect at multiple coordinates.

After defining a route, we classify the intersected segments of two routes into the following nine categories. The basic three categories of *continuously intersected segments* in two routes are not intersecting, intersecting at a coordinate, or intersecting at a segment. Next, two categories of *discontinuously intersected segments* of two routes are intersecting at many nonadjacent coordinates as shown in Fig. 5 and intersecting at multiple nonadjacent segments as shown in Fig. 6. Finally, we infer that four additional categories of discontinuously

intersected segments of two routes are intersecting at a coordinate and a segment, intersecting at a coordinate and multiple segments, intersecting at multiple coordinates and a segment, and intersecting at multiple coordinates and segments.

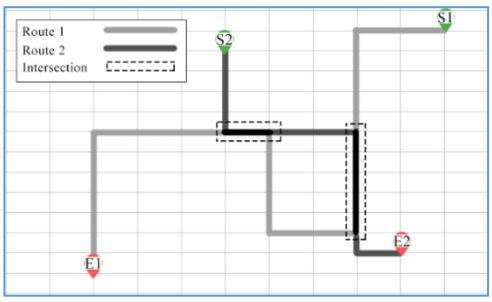

Figure 6. Two routes intersect at multiple segments.

IV. Rapid Route Comparison

To improve the efficiency of comparing two routes, we apply bounding box of a route in advance to eliminate most of the coordinates in a route. Then using brute force approach to compare two routes obtains an absolutely accurate result. Let us define a bounding box and four algorithms for comparing two routes, and determine the driving direction in two routes.

A. Defining a Bounding Box

A bounding box can be constructed from the left-bottom vertex and the right-top vertex as shown in Fig. 7. The left-bottom vertex (V_LB) is a coordinate with the minimum latitude and longitude in a route, and the right-top vertex (V_RT) is a coordinate with the maximum latitude and longitude in a route. Therefore, a bounding box can be defined as,

$$B(r) = \{V_{LB}, V_{RT}\}, \quad (1)$$

where V_{LB} is $\left(Min\{p_j.lat \mid p_j \in r\}, Min\{p_j.lng \mid p_j \in r\}\right)$ and V_{RT} is $\left(Max\{p_j.lat \mid p_j \in r\}, Max\{p_j.lng \mid p_j \in r\}\right)$.

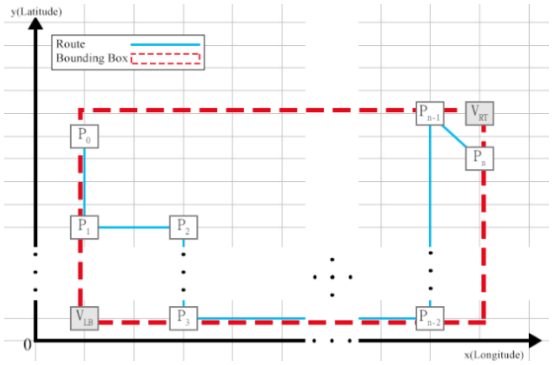

Figure 7. A bounding box of a route.

B. Determining Driving Directions in Two Routes

If the driving directions in the intersected segment of two routes is different, these two routes are not intersected at the segment, i.e., the directions of segments must be same to be truly intersected. In this paper, we propose the following approach to determine the driving direction of an intersected segment in two routes.

Step 1—***Finding two adjacent coordinates in the intersected segment.*** Determining the driving direction of the intersected segment in two routes is to find two arbitrary adjacent coordinates which are in the intersected segment s_i, denoted by p_n and p_m. If only one coordinate is found in the intersected segment, the driving direction of two routes is different.

Step 2—***Using the indexes of two adjacent coordinates in the intersected segment to determine the driving direction.*** A route is a series of pairs of latitude and longitude coordinates, and every coordinate is indexed from the beginning to the end. We denote $p_n.idx_1$ and $p_m.idx_1$ as the index of the p_n and p_m in the r_1 and $p_n.idx_2$ and $p_m.idx_2$ as the index of the p_n and p_m in the r_2. Therefore, determining the driving direction in the intersected segment in two routes can be indicated by the following function as,

$$D(p_n, p_m) = \begin{cases} 1 & ,if\ (p_n.idx_1 - p_m.idx_1) \times (p_n.idx_2 - p_m.idx_2) > 0 \\ 0 & ,if\ (p_n.idx_1 - p_m.idx_1) \times (p_n.idx_2 - p_m.idx_2) < 0, \\ null, & if\ (p_n.idx_1 - p_m.idx_1) \times (p_n.idx_2 - p_m.idx_2) = 0 \end{cases} \quad (2)$$

where $D(p_n, p_m) = 0$ represents that the driving direction is opposite, $D(p_n, p_m) = 1$ represents that the driving direction is same, and $D(p_n, p_m) = null$ represents that two routes are intersected at a coordinate.

C. Algorithms for Comparing Two Routes

1) Brute Force(BR)

Obtaining the intersected segments in two routes is done by comparing every coordinate in the route r_1 with in the route r_2. The intersected segments in two routes are denoted by $\mathcal{R} = r_1 \cap r_2 = \{p_1 \to p_2 \to p_3 \to \cdots \to p_k\}$. Moreover, we could use the comparing results by BR to know the category of the intersected segments because BR sequentially compares every coordinate from the beginning to the end in two routes. The pseudocode for the BR is given in the Algorithm 1.

Algorithm 1: Brute Force, BR

Input: r_1, r_2 are the set of (lat, lng)
Output: $\mathcal{R} = r_1 \cap r_2$
Procedure:
1. For each p_j in r_1 **do**
2. **For** each p_i in r_2 **do**
3. **If** $p_i = p_j$,
4. Add p_i or p_j into set of $\mathcal{R}$
5. Return $\mathcal{R}$

2) One-Time Bounding Boxing(1-BB)

We build two bounding boxes of two routes in 1-BB before using BR to compare two routes. If two bounding boxes are not overlapped, two routes are definitely not intersected. However, two intersected bounding boxes represent that two routes may be intersected. Therefore, we define a membership function as,

$$S_{overlap}(B(r_1), B(r_2)) = \begin{cases} B(r_1) \cap B(r_2), if\ B(r_1) \cap B(r_2) \neq \emptyset \\ \emptyset \quad , otherwise \end{cases}, \quad (3)$$

where $B(r_1)$ is the bounding box of r_1 and $B(r_2)$ is the bounding box of r_2. If $S_{overlap}$ is not an empty set, first remove the pairs of coordinates of each route which are in the range of $S_{overlap}$. Next, coordinates of two routes in the $S_{overlap}$ are compared one by one to find the overlapped route.The pseudocode for the 1-BB is given in Algorithm 2.

Algorithm 2: One-Time Bounding Boxing, 1-BB

Input: r_1, r_2 are the set of (lat, lng)
Output: $\mathcal{R} = r_1 \cap r_2$
Procedure:
1. Build the bounding boxes of the r_1, r_2
2. If $B(r_1) \cap B(r_2) = \emptyset$, **Return** $\emptyset$
3. Else
4. $r_1 = \{r_1 | r_1 \; in \; S_{overlap}\}, r_2 = \{r_2 | r_2 \; in \; S_{overlap}\}$
5. For each p_j in r_1 **do**
6. **For** each p_i in r_2 **do**
7. **If** $p_i = p_j$,
8. Add p_i or p_j into set of $\mathcal{R}$, **Break**
9. Return $\mathcal{R}$

3) N-Times Bounding Boxing Until Equal(n-BB)

As mentioned previously, there are two categories of intersected segments in two routes, which are continuous and discontinuous. In the continuously intersected category, two routes are only intersected at a coordinate or intersected at a segment. Thus, we repeatedly build the bounding boxes of two routes and remove the pairs of coordinates which are not in $S_{overlap}$, in two routes, and then r_1 would be equal to r_2 and r_2 would be equal to $\mathcal{R}$. Finally, $B(r_1)$ would be equal to the $B(r_2)$. In the discontinuously intersected category, two routes are intersected at nonadjacent coordinates or segments. As a result, r_1 would be not equal to r_2 and r_2 would be not equal to $\mathcal{R}$. However, $B(r_1)$ would be equal to the $B(r_2)$ because the two bounding boxes are bounded by the intersected coordinates or segments instead of the non-intersected coordinates or segments in the two routes. With this observation $B(r_1)$ would be equal to $B(r_2)$ and then we could improve the efficiency of 1-BB by repeatedly building the bounding boxes of two routes and removing the pairs of coordinates not in $S_{overlap}$ before comparing two routes. The pseudocode for the n-BB is given in Algorithm 3.

Algorithm 3: N-Times Bounding Boxing Until Equal, n-BB

Input: r_1, r_2 are the set of (lat, lng)
Output: $\mathcal{R} = r_1 \cap r_2$
Procedure:
1. Build the bounding boxes of the r_1, r_2
2. While $(B(r_1) \neq B(r_2))$ **do**
3. **If** $B(r_1) \cap B(r_2) = \emptyset$, **Return** $\emptyset$
4. **Else**
5. $r_1 = \{r_1 | r_1 \; in \; S_{overlap}\}, r_2 = \{r_2 | r_2 \; in \; S_{overlap}\}$
6. Build the bounding boxes of the r_1, r_2
7. For each p_j in r_1 **do**
8. **For** each p_i in r_2 **do**
9. **If** $p_i = p_j$,
10. Add p_i or p_j into set of $\mathcal{R}$, **Break**
11. Return $\mathcal{R}$

4) Threshold-Based Bounding Boxing(t-BB)

This is intuitively true that BR, which would break out of the for-loop if the same coordinate is found, becomes faster when the overlapping ratio of two routes is very high. Because we are unable to know the real overlapping ratio of two routes before comparing two routes, t-BB uses the intersected area of two bounding boxes to estimate the real overlapping ratio of two routes. We define, A_{box}, which is ratio of the overalpped area of two bounding boxes by,

$$A_{box} = \frac{Area(S_{overlap})}{Area(B(r_1) \cup B(r_2))} \times 100\%. \quad (4)$$

The pseudocode for the t-BB is given in Algorithm 4

Algorithm 4: Threshold-Based Bounding Boxing, t-BB

Input: r_1, r_2 are the set of (lat, lng)
Output: $\mathcal{R} = r_1 \cap r_2$
Procedure:
1. Build the bounding boxes of the r_1, r_2
2. If $B(r_1) \cap B(r_2) = \emptyset$, **Return** $\emptyset$
3. If $A_{box} < 90\%$,
4. **While** $(B(r_1) = B(r_2))$ **do**
5. **If** $B(r_1) \cap B(r_2) = \emptyset$, **Return** $\emptyset$
6. **Else**
7. $r_1 = \{r_1 | r_1 \; in \; S_{overlap}\}, r_2 = \{r_2 | r_2 \; in \; S_{overlap}\}$
8. Build the bounding boxes of the r_1, r_2
9. For each p_j in r_1 **do**
10. **For** each p_i in r_2 **do**
11. **If** $p_i = p_j$,
12. Add p_i or p_j into set of $\mathcal{R}$, **Break**
13. Return $\mathcal{R}$

V. Experimental Results

We analyzed the performance of BR, the proposed 1-BB, n-BB, T-BB, and Gjaldbæk's algorithm and implemented on the Google Map. Routes are planned in the driving mode on the eastern United States map. The algorithms are implemented by C language and run on a desktop computer equipped with Intel Xeon E3-1231 v3 3.40 GHz CPU and 32GB RAM, and operating system is Linux Mint 17.2 Cinnamon.

In our experiments, the two randomly selected routes may not be the similar number of coordinates. We arbitrarily selected 124 routes where the number of coordinates is between 24 and 1823 and the average length is 24.25 km on the New York map bounded by VLT(40.89172, -73.90732) and VRB(40.59727, -

73.49945). There were 7626 pairs of the routes compared. In the following figures, Gjaldbæk(s) represents that routes are rasterized with a size of s ×s m^2 grid.

As shown in Fig. 8 and Fig. 9, with the aid of bounding boxes to eliminate unnecessary coordinates, our three approaches, even using bounding boxing once, like 1-BB, are all much faster than BR and Gjaldbæk's approaches whose grid sizes are 35, 75, and 150, respectively. In our experiments, n-BB is as fast as t-BB because the few number of coordinates in two routes take little computation time. Gjaldbæk (75) is much faster than Gjaldbæk (35) because rasterizing a route in a large size of a grid takes little computation time to matching each grid fields than a small one at the cost of positioning accuracy.

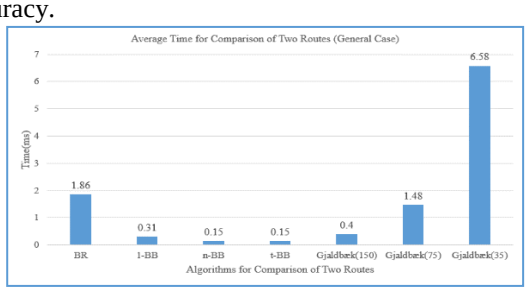

Figure 8. The average time for comparing two routes.

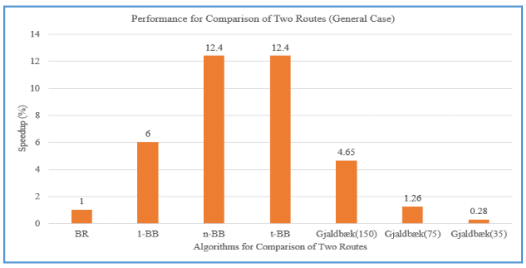

Figure 9. The speedup for comparing two routes.

The experimental results show that our approaches are more accurate and efficient than Gjaldbæk's. Gjaldbæk's approach is obviously expected to have poor performance because it is similar to our approach but it builds too many bounding boxes for rasterizing routes and matches pairs of grid fields in two rasterized routes. In contrast to our proposed approaches, 1-BB, n-BB, and t-BB always build two bounding boxes and remove unnecessary coordinates from two routes iteratively. Thus, our approaches could decrease much computation time for comparing two routes.

VI. Conclusion

In this paper, we specifically described the categories of intersection of two routes and proposed three approaches based on bounding boxes to compare two routes. Moreover, we also implemented the approaches on the Google Map and analyzed their performance compared with Gjaldbæk's approach. The proposed t-BB and n-BB can be also used in other geographic information systems (e.g., Apple Map and Bing Map) and achieve rapid and accurate result of comparison of two routes. Finally, we can rapidly pair passengers with drivers in real-time carpooling systems by t-BB when there are many passengers and drivers to carpool.

Acknowledgment

This work was partially supported by Ministry of Science and Technology, Taiwan (R.O.C) under grand numbers 104-2815-C-024-008-E and 106-2221-E-024-003.

References

[1] H. Cao, N. Mamoulis, and D. W. Cheung, "Mining frequent spatio-temporal sequential patterns," *Proc. IEEE Int. Conf. Data Mining, ICDM*, pp. 82–89, 2005.

[2] L. M. Ni and J. Fan, "Towards mobility-based clustering categories and subject descriptors," *Kdd*, pp. 919–927, 2010.

[3] W. He, K. Hwang, and D. Li, "Intelligent carpool routing for urban ridesharing by mining GPS trajectories," *IEEE Trans. Intell. Transp. Syst.*, vol. 15, no. 5, pp. 2286–2296, 2014.

[4] M. Gjaldbæk, "Practical polyline matching for GPS data," *Tech. Univ. Denmark, DTU, DK-2800 Kgs. Lyngby, Denmark*, pp. 1–24, 2010.

[5] G. Barequet and S. Har-Peled, "Efficiently approximating the minimum-volume bounding box of a point set in three dimensions," *J. Algorithms*, vol. 38, no. 1, pp. 91–109, 2001.

[6] H. Haverkort and F. Van Walderveen, "Locality and bounding-box quality of two-dimensional space-filling curves," *Comput. Geom. Theory Appl.*, vol. 43, no. 2 SPEC. ISS., pp. 131–147, 2010.

[7] S. S. Chawathe, "Segment-Based map matching," in *Proc. IEEE Intell. Veh. Symp.*, 2007, pp. 1190–1197.

[8] J. Huang, C. Liu, and J. Qie, "Developing map matching algorithm for transportation data center," in *Proc. - 2014 9th Int. Conf. P2P, Parallel, Grid, Cloud Internet Comput. 3PGCIC 2014*, 2015, pp. 167–170.

[9] Y. Li, Q. Huang, M. Kerber, L. Zhang, and L. Guibas, "Large-Scale Joint Map Matching of GPS Traces," in *Proc. 21st ACM SIGSPATIAL Int. Conf. Adv. Geogr. Inf. Syst.*, 2013, pp. 214–223.

[10] L. Bermingham, K. Lee, and I. Lee, "Spatio-temporal trajectory region-of-interest mining using delaunay triangulation," 2014.

[11] F. Giannotti, M. Nanni, D. Pedreschi, and F. Pinelli, "Trajectory pattern mining," pp. 330–339, 2007.

[12] N. Mamoulis, H. Cao, G. Kollios, M. Hadjieleftheriou, Y. Tao, and D. W. Cheung, "Mining, indexing, and querying historical spatiotemporal data," in *Proc. ACM SIGKDD Int. Conf. Knowl. Discov. data Min. KDD 04*, 2004, p. 236.

[13] W. He, D. Li, T. Zhang, L. An, M. Guo, and G. Chen, "Mining regular routes from GPS data for ridesharing recommendations," in *Proc. ACM SIGKDD Int. Work. Urban Comput. - UrbComp '12*, 2012, p. 79.