A Novel Algorithm for Detecting Spatial-Temporal Trajectory Outlier

Shenglan Lv
School of Computer Science and Technology
Nanjing Normal University
Nanjing, China
lvshenglan_njnu@163.com

Yifan Zhang
School of Computer Science and Technology
Nanjing Normal University
Nanjing, China

Genlin Ji
School of Computer Science and Technology
Nanjing Normal University
Nanjing, China
glji@njnu.edu.cn

Bin Zhao
School of Computer Science and Technology
Nanjing Normal University
Nanjing, China

***Abstract*—As an important area of spatial-temporal data mining, trajectory outlier detection has already attracted broad attention in recent years. In this paper, we present a novel distance measurement between spatial-temporal sub-trajectories and propose algorithm STOD for detecting outliers in both spatial and temporal dimensions jointly. Each trajectory is divided into line segments at first, and the corresponding minimal boundary boxes are constructed. After combination, the outlier index is computed with our distance measurements including overlapping volume, angle and speed. To improve the efficiency of algorithm STOD, we present algorithm PSTOD for parallel detecting spatial-temporal trajectory outlier, which is implemented using Spark framework. The experiment results on real taxi dataset show that the two algorithms are effective and efficient.**

Keywords*—Spatial-temporal trajectory; Outlier detection; Parallel data mining

I. INTRODUCTION

With the proliferation of GPS-equipped devices and the maturity of wireless communication technology, massive trajectory data collection (e.g. hurricane data, vehicle position data, cell-phone data) from moving objects becomes much easier in the emerging application fields such as traffic management and location-based service. Among the trajectory analyses, outlier detection is viewed as promising and significant, especially in aspects of criminal activities analyses, consumption pattern classification and weather forecast. For example, we could improve the warning accuracy by studying historical typhoon trajectories to discover abnormal moving patterns.

Some main existing algorithms [1-7] are variously performed with different definitions and measurements. As an initial edition, Knorr et al. [2] have presented a distance-based algorithm with extracting trajectory global features such as endpoints and average speed. For enhancement, Jae-Gil et al. [4] have proposed an important partition-and-detect framework and the outlier detection algorithm TRAOD which partitions a trajectory into a set of line segments and detecting outlying sub-trajectories from the database. TOP-EYE [8], another impressive method, continuously computes the outlying score for each trajectory in an accumulating way based on the evolving moving direction and density of trajectories. To achieve the objective of detecting taxi driving frauds or road network changes in modern cities, algorithm iBAT [9] focuses on representing each trajectory as a sequence of symbols with higher efficiency. Based on the MEX framework [10], the persistence of sub-trajectory spatial similarity is measured with new outlier definitions and key properties of trajectory stream. However, these algorithms above can only detect spatial outliers, which mean the temporal outliers are lost. Trajectories with different sampling time always correspond to different motion patterns. Without specific temporal detection, it is easy to lose outliers in the time dimension. Figure 1 shows an example of similar spatial trajectories which actually should be divided into three groups by different sampling instants. TR_3 will be missed if we use previous algorithms, however, TR_3 is an outlier obviously.

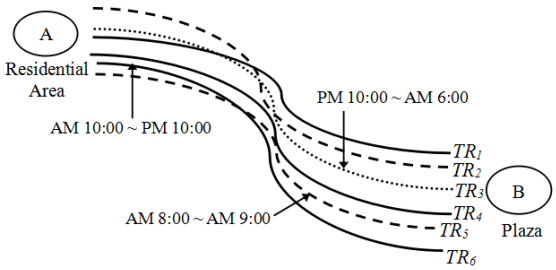


Fig. 1: Example of different spatial-temporal trajectory patterns

There are six taxi trajectories from residential area A to Plaza B which should be considered similar in space in Figure 1. In fact, TR_2 and TR_5 are generated during 8 am to 9 am probably belonging to shop assistants; TR_1, TR_4 and TR_6 are

generated during 10 am to 10 pm probably belonging to customers; TR_3 is generated during 10 pm to 6 am probably belonging to a criminal.

Taking full advantage of trajectories in time dimension can help acquiring potential valuable information and raise accuracy of moving pattern analysis especially in the field of outlier detection. Some related spatial-temporal detecting methods have been proposed and each addresses a certain aspect of abnormality.

Originally, a three-step approach [11] locates spatial outliers first and then discovers the temporal outliers among the checked unusual trajectories. But this algorithm is unable to detect temporal anomalies if they are similar in space. And the computation time cost is too high when facing large-scale data. For improvement, algorithm COD [12] has been presented to perceive congestion events. It is capable of measuring spatial-temporal distance between sub-trajectories jointly. Another algorithm TPRO [13] divides dataset into groups to obtain the top-k most popular routes. This method labels an outlier if it has a great difference with the selected routes. Nevertheless, algorithm COD only uses overlapping volume to measure similarity between sub-trajectories so that other structural features are lost, and the grid in algorithm TPRO is too rough to ensure the accuracy. Besides, those two algorithms above are not available for common moving pattern analyses of cellphone users or vehicles except taxi.

In this paper, we propose a novel spatial-temporal trajectory outlier detection algorithm STOD (Spatial-temporal Trajectory Outlier Detection) which addresses the current issues above. Firstly, the *MBBs* (Minimal Boundary Boxes) for each sub-trajectory are constructed, and then the front and back *MBBs* will be combined if the angle difference is small enough. Secondly, a comprehensive detecting approach is accomplished by considering distance between sub-trajectories in the aspects of spatial-temporal, angle and speed. To promote efficiency, an enhanced spatial-temporal grid index is created. Finally, we evaluate the performance of algorithm STOD and PSTOD through experiments with real taxi GPS data.

The contributions of this paper can be summarized as follows:

1. This paper takes both spatial and temporal attributes into consideration jointly to detect spatial-temporal outliers and presents a novel algorithm for trajectory outlier detection.

2. We put forward a new distance measurement between sub-trajectories with other structural features and enhance the performance of the algorithm by taking full advantage of grid index and Spark.

3. With the experiments on real GPS trace dataset, we evaluate two algorithms in the aspects of efficiency and effectiveness.

The remainder of this paper is organized as follows. The necessary concepts and formulas are defined in Section II. Efficient solutions for spatial-temporal outlier detection are presented in Section III. The experimental observations on real dataset are reported in Section IV. Finally, Section VI concludes this paper.

II. PROBLEM STATEMENT

A. Notations

The notations of this paper are shown in Table 1.

We denote a spatial-temporal trajectory TR_j as $TR_j = \{tr_1, ..., tr_i,...,tr_n\}$, where n is the number of the sub-trajectories and tr_i is formed by the front sampling point p_i and the back sampling point p_{i+1}. There are four attributes including sampling time t, longitude x, latitude y and sampling speed v of p_i, denoted as $p_i(t, x, y, v)$.

TABLE I: Table of notations

Notation	Definition
p_i	Spatial-temporal sampling point
tr_i	Sub-trajectory
TR_j	Spatial-temporal trajectory
MBB_i	Minimal boundary box of tr_i
N	Neighboring threshold
$OV(MBB_i,MBB_j)$	Overlapping volume of *MBB* pairs
$OVMeasure_i$	Overlapping measure of MBB_i
$AngleMeasure_i$	Angle measure of MBB_i
$SpeedMeasure_i$	Speed measure of MBB_i
AI_i	Anomaly index of MBB_i
OI_j	Outlier index of TR_j
$\varepsilon, \delta, \gamma$	Parameters of anomaly index
θ	Angle threshold for combination
E	Threshold for outlier index

According to the statement of algorithm COD, minimal boundary box MBB_i of tr_i can be constructed with the boundary points as follows:

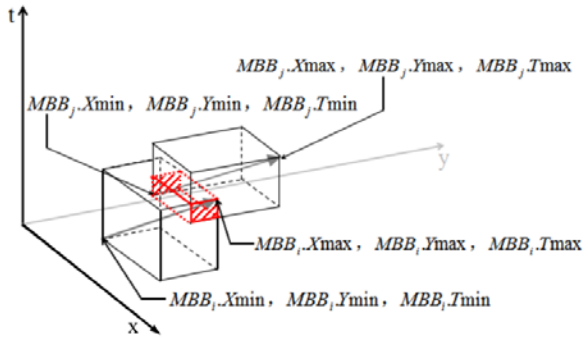
$$\{MBB_i.X\text{min}, MBB_i.X\text{max}, MBB_i.Y\text{min}, MBB_i.Y\text{max}, MBB_i.T\text{min}, MBB_i.T\text{max}\}$$

Each plane of MBB_i is parallel or vertical to the coordinate system, where XY represent the location and T represents time. The body diagonal of MBB_i indicates tr_i. In this paper, there are two extra MBB_i attributes denoted as $MBB_i.angle$ and $MBB_i.speed$. To construct MBB_i, the value of boundary points and attributes are shown as follows:

$$MBB_i.X\text{min} = p_i.x \qquad MBB_i.X\text{max} = p_{i+1}.x$$

$$MBB_i.Ymin = p_i.y \qquad MBB_i.Ymax = p_{i+1}.y$$
$$MBB_i.Tmin = p_i.t \qquad MBB_i.Tmax = p_{i+1}.t$$
$$MBB_i.Angle = \frac{180}{\pi} * arctan\left(\left|\frac{p_{i+1}.y - p_i.y}{p_{i+1}.x - p_i.x}\right|\right)$$
$$MBB_i.Speed = \frac{|p_i p_{i+1}|}{p_{i+1}.t - p_i.t}$$

For every pair of *MBBs*, there are four relationships between them denoted as disjoint, part-overlapping, full-overlapping and equal. *OV* is the volume of the overlapping part. Figure 2 shows an example of part-overlapping *MBBs*.



$$OV = \begin{vmatrix} (MBB_i.Xmax - MBB_j.Xmin) * \\ (MBB_i.Ymax - MBB_j.Ymin) * \\ (MBB_i.Tmax - MBB_j.Tmin) \end{vmatrix}$$

Fig. 2: Illustration of part-overlapping

B. Problem Definitions

Definition 1 (*Segment Combination*) Given a line segment tr_i, the angle difference α between tr_i and $tr_{i+1},...,tr_k$ is calculated in order, where $k \le n$. Combination can be carried out when the calculation meeting following requirements:

1) when $k<n$, if α between tr_i and $tr_{i+1},...,tr_k$ is less than threshold θ but between tr_i and tr_{k+1} is more than θ, then combine $tr_i,...,tr_k$ as tr_i';

2) when $k=n$, if α between tr_i and $tr_{i+1},...$,tr_k is less than threshold θ, then combine $tr_i,...,tr_k$ as tr_i';

To accomplish the combination, the value of boundary points and attributes are recalculated as follows:

$$MBB_i.Xmin = p_i.x \qquad MBB_i.Xmax = p_{k+1}.x$$
$$MBB_i.Ymin = p_i.y \qquad MBB_i.Ymax = p_{k+1}.y$$
$$MBB_i.Tmin = p_i.t \qquad MBB_i.Tmax = p_{k+1}.t$$
$$MBB_i.Angle = \frac{180}{\pi} * arctan\left(\left|\frac{p_{k+1}.y - p_i.y}{p_{k+1}.x - p_i.x}\right|\right)$$
$$MBB_i.Speed = (p_i.v + p_k.v)/2$$

Figure 3 shows the procedure of the combination.

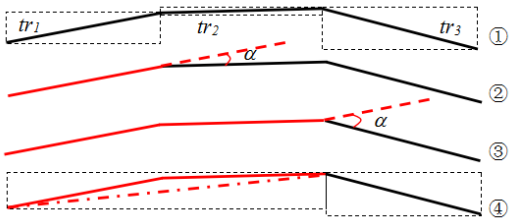


Fig. 3: Example of combination

In step 2, we have $\alpha<\theta$ after computing the angle difference between tr_1 and tr_2. Then we have $\alpha \ge \theta$ after computing the angle difference between tr_1 and tr_3 in step 3. Finally, we combine tr_1 and tr_2 in step 4.

Definition 2 (*Segment Neighbor*) Given a neighboring threshold *N*, segment tr_N is denoted as a *segment neighbor* of tr_i if the distance *Dist* between tr_i and tr_N is less than *N*. The distance *Dist* between line segments can be calculated as

$$Dist(tr_i, tr_N) = \min\left\{|p_i p_N|, |p_i p_{N+1}|, |p_{i+1} p_N|, |p_{i+1} p_{N+1}|\right\} \quad (1)$$

The distance measurements will be calculated between tr_i and its segment neighbors.

Definition 3 (*Overlapping Measure*) The spatial-temporal distance between *MBB* pairs denoted as $OVMeasure_i$ can be calculated as

$$OVMeasure_i = \frac{Volumn(MBB_i)}{1 + \sum OV(MBB_i, MBB_j)} \quad (2)$$

where MBB_i is the current one and MBB_j is the comparative one. The number of sub-trajectories near the MBB_i is inversely proportional to $OVMeasure_i$.

Definition 4 (*Angle Measure*) The angle difference between *MBB* pairs denoted as $AngleMeasure_i$ can be calculated as

$$AngleMeasure_i = \sum |MBB_i.Angle - MBB_j.Angle| \quad (3)$$

Definition 5 (*Speed Measure*) The speed difference between *MBB* pairs denoted as $SpeedMeasure_i$ can be calculated as

$$SpeedMeasure_i = \sum |MBB_i.Speed - MBB_j.Speed| \quad (4)$$

Definition 6 (MBB_i *Anomaly Index*) The Anomaly Index of MBB_i represents the anomaly degree that can be calculated as

$$AI_i = \varepsilon * OVMeasure_i + \delta * AngleMeasure_i + \gamma * SpeedMeasure_i \quad (5)$$

where ε, δ and γ are weight factors between 0 and 1 satisfied $\varepsilon+\delta+\gamma=1$. The possibility of the line segment being abnormal is basically proportional to AI_i.

Definition 7 (*Outlier Index*) The Anomaly Index of TR_j represents the anomaly degree that can be calculated as

$$OI_j = \sum_{tr_i \in TR_j} AI_i \quad (6)$$

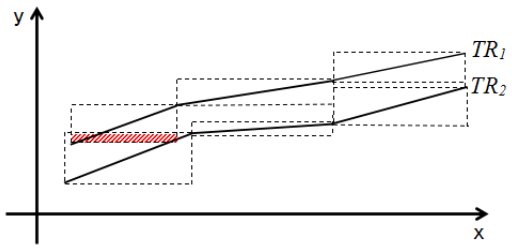
TR_j will be labeled as an outlier if $OI_j \geq E$, where E is the threshold given by a user.

III. ALGORITHMS

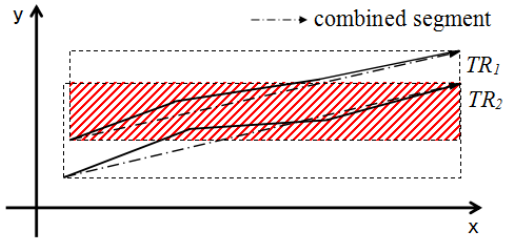
This section introduces the ideas of algorithm STOD and PSTOD. The descriptions of two algorithms are given.

A. Sub-trajectory Combination

The distance measure between spatial-temporal line segments in algorithm COD only contains overlapping volume and has no trajectory preprocessing, which is possible to cause errors.



(a) before sub-trajectory combination



(b) after sub-trajectory combination

Fig. 4: Example of sub-trajectory combination.

In Figure 4, TR_1 and TR_2 are two spatial-temporal trajectories projections in coordinates on X-Y plane. In the similar way of projection, the dotted box and the red shadow area correspond to *MBB* and overlapping volume respectively. We can see that TR_1 and TR_2 are similar according to ordinary standard. If we take the measurement in Figure 4(a), TR_1 and TR_2 are dissimilar because of the small overlapping part. But if we combine the segments satisfied prerequisites as Figure 4(b) shown, it is easy to draw the right conclusion that TR_1 and TR_2 are similar.

Algorithm 1: Combine Sub-trajectories

Input: trajectory dataset T, threshold θ

Output: combined trajectory set T'

01: **for** each trajectory $TR_j \in T$ **do**
02: Divide TR_j into line segments;
03: **for** each segment $tr_i \in TR_j$ **do**
04: Construct MBB_i of tr_i;
05: **end for**
06: **if** α between tr_i and $tr_{i+1},...,tr_k$ meets $\alpha<\theta$ **then**
07: Combine $tr_i,...,tr_k$ as tr_i';
08: $TR_j' \leftarrow TR_j$;
09: **end if**
10: **end for**

To accomplish the sub-trajectory combination, we divide each trajectory into line segments and construct minimal boundary boxes as shown in Algorithm 1. Then we combine sub-trajectories based on Definition 1. After this combination, we have processed trajectories for creating grid index and detecting outliers.

B. Spatial-Temporal Grid Index

In algorithm STOD, angle and speed are considered besides taking the advantage of overlapping volume. Under this circumstance, the result might be not accurate enough because of the shared weight if we compare each segment with all the rest segments in the dataset. Therefore, we only compare each sub-trajectory with its segment neighbors according to Definition 2. To improve the efficiency, an enhanced spatial-temporal grid index is created, which is expanded into three dimensions including time and X-Y coordinate.

Assuming the height and width of a grid is m, the depth is n. The body diagonal can be calculated as

$$BodyDiagonal = \sqrt{m^2 + n^2} \quad (7)$$

Given a cell g_{abc} is located at the a-th row on X-axis, b-th row on Y-axis and c-th row on T-axis. The affect region is the set of cells whose minimum distance with g_{abc} is not greater than body diagonal, and it can be denoted as $AR(g_{abc})=\{g_{ijk} \in G \mid |i-a| \leq 2, |j-b| \leq 2, |k-c| \leq 2$, and $|i-a|+|j-b|+|k-c|<6\}$. Based on that, we will calculate the distance between segments in the affect region additionally to verify that the distance is below the neighboring threshold N. The pseudo code of creating grid index is shown in Algorithm 2.

Algorithm 2: Create Grid Index

Input: trajectory dataset T'

Output: index I

01: *grid=CreateGrid(x, y, t)*; // create $x \times y \times t$ grids on the map
02: **for** each trajectory $TR_j' \in T'$ **do**
03: Divide TR_j' into line segments;
04: **for each** $tr_i' \in TR_j'$ **do**
05: *gridlist=GetGridID (tr_i')*; // get *gridID* that tr_i' passed
06: **for each** *gridID* $\in$ *gridlist* **do**
07: Output (*gridID*, tr_i');
08: **end for**
09: **end for**
10: **end for**

Figure 5 shows an example of spatial-temporal grid index and tr_3 is the current segment, tr_7 and tr_8 are comparative segments. Assuming that the index has been created and then

the region query of tr_3 is executed with the threshold of one grid width. As a result, tr_7 is the segment neighbor of tr_3 and the distance measurement between tr_7 and tr_3 will be calculated. The grids that tr_8 passed are beyond the threshold, and the segments from the same trajectory do not participate in the distance calculation.

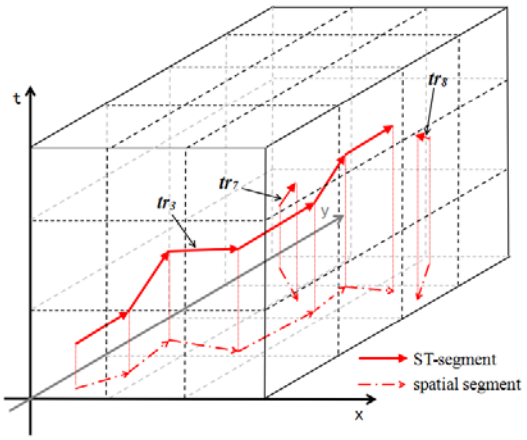


Fig. 5: Example of spatial-temporal grid index.

C. Algorithm STOD

During the detection stage, the anomaly index AI_i of each MBB_i is calculated and accumulated to obtain OI_j. If OI_j of TR_j is larger than the threshold E, TR_j will be labeled as an outlier and output. The pseudo code of algorithm STOD is shown as follows.

Algorithm 3: Algorithm STOD

Input: trajectory dataset T, threshold θ, N, E, parameter ε, δ , γ

Output: outlier set O'

01: Combine Sub-trajectories; // Algorithm 1
02: $T' \leftarrow T$; // T' is the new trajectory dataset after combination
03: Create Grid Index; // Algorithm 2, generate index I
04: **for** each $TR_j' \in T'$ **do**
05: **for** each segment $tr_i' \in TR_j'$ **do**
06: obtain segment neighbors of tr_i' with N and I;
07: **for** each segment neighbor tr_{Nk}
08: Calculate AI_i;
09: // calculate Anomaly Index between tr_i' and tr_{Nk}
10: **end for**
11: Calculate OI_j; // Outlier Index of TR_j'
12: **if** $OI_j > E$ **then**
13: **Output** (ID, TR_j') // TR_j' is an outlier
14: **end if**
15: **end for**
16: **end for**

D. Algorithm PSTOD

The spatial-temporal outliers are detected in the three dimensions. If we only execute algorithm STOD on single computer, it will lead to a vast number of calculations and large time cost. Therefore, we present a parallel algorithm PSTOD based on Spark to improve efficiency.

Spark is a novel parallel computation framework launched by Berkeley in recent years. Aiming at the calculation deficiency of MapReduce in the respects of iterative, interactive and streaming works, Spark develops an extension and presents a brand new concept named RDD which accomplishes data sharing throughout stages. To meet the demands of users, Spark also provides multi-scenario ecological components based on its core API and shows great performance of big data mining.

The general framework of algorithm PSTOD is presented as shown in Figure 6.

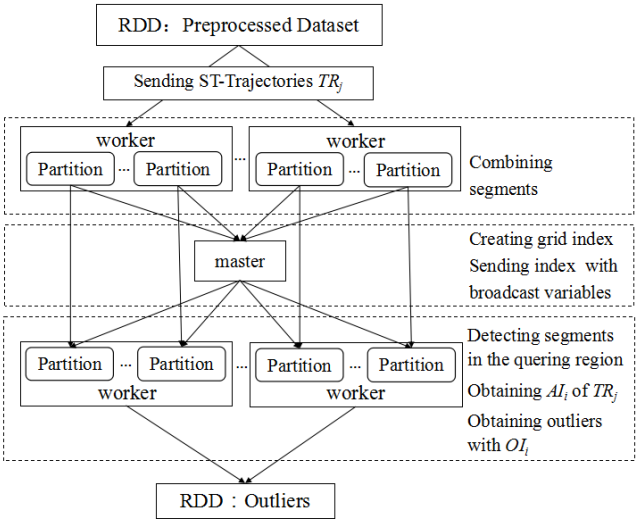


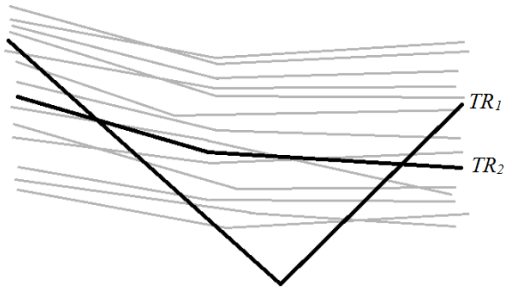
Fig. 6: General framework of algorithm PSTOD

IV. EXPERIMENTS

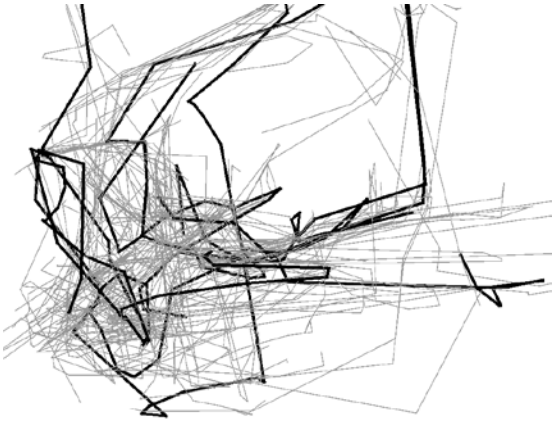
In this section, we give an exhibition of our experiment based on the cloud computing platform with real-world taxi trajectory dataset. The environment of cloud computing in this paper consists of 19 servers (1 master node, 18 slave nodes), and each node has the same configuration. The CPU of each node is Intel(R) Xeon(R) CPU E5620(2.40GHZ) with 64 bit Linux Debian operating system and the version of Spark is Spark-1.4.0.

Being different from the past, the objective aimed to detect spatial-temporal outliers has expanded the calculation to three-dimension. Under this circumstance, the time cost of this problem will be higher than the detection of spatial outliers. To compare efficiency between the serial and parallel algorithm, we pick up different numbers of sub-trajectories corresponding to 115200, 230400, 345600, and 460800 under a real-world dataset which is collected from taxis in San Francisco. Each original trajectory includes 1440 sampling points. To observe performance of algorithm PSTOD, we conduct an experiment with different number of computing nodes under the dataset which contains 3200 trajectories corresponding to 4.6million segments.

Figure 7 shows the visualization of detection results, in which common lines are normal trajectories and bold lines are spatial-temporal outliers. In Figure 7(a), TR_1 is an abnormal trajectory in geographic location and TR_2 is an outlier in temporal because its sampling time we set is at 6 am and other trajectories are at 8 pm. In Figure 7(b), we pick 120 trajectories from the dataset and can easily see the outliers appear if their directions are different from those neighbors or they have very few neighboring trajectories. Although we are not able to judge the temporal outliers if they are normal in spatial with naked eye, the results in Figure 7(a) shows the effectiveness of our method.



(a) Outliers for synthetic data



(b) Outliers for taxis in San Francisco

Fig. 7: Examples of outliers

The comparison between algorithm STOD and parallel algorithm PSTOD is shown as Table 2 and Figure 8.

TABLE II: Execution time of serial and parallel algorithms (sec.)

Number of sub-trajectories	STOD	PSTOD
115200	937	120
230400	3508	144
345600	9019	168
460800	15485	186

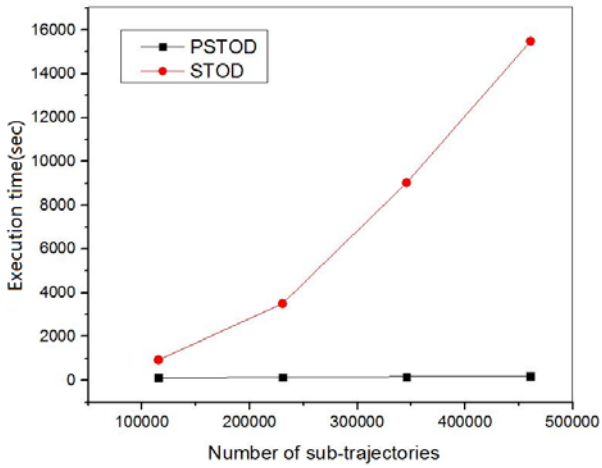


Fig. 8: Execution time of comparison between two algorithms

We can see that the efficiency of algorithm PSTOD is much higher than algorithm STOD in Figure 8. The grid index is created in both algorithms so that the time cost difference is not so great when the amount of trajectories is small. As the dataset grows larger, the executing time of algorithm STOD increases rapidly. Because of the expanding detection, the three-dimensional calculation takes a lot of time and leads to huge cost if the dataset becomes a little larger on a single computer.

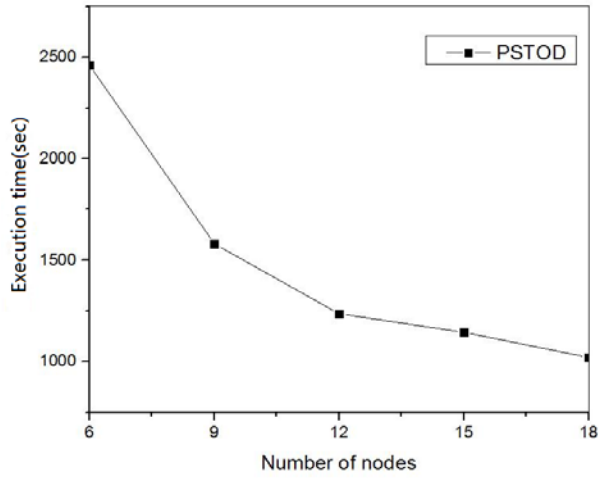


Fig. 9: Execution time of algorithm PSTOD

To demonstrate the performance of algorithm PSTOD, we execute the algorithm with different number of computing nodes as shown in Figure 9. The executing time is descending as the number of nodes increases. When the computing nodes increase to a certain amount, the communication cost will take up the main time, resulting in a slower decline of the total executing time decreases.

V. Conclusion

In this paper, we propose a novel algorithm STOD for detecting spatial-temporal trajectory outliers. Firstly spatial-temporal trajectories are divided into line segments and the

sub-trajectories are combined according to definitions. Secondly, for each line segment, the different measurements are calculated to obtain the anomaly index with an enhanced three-dimensional grid index. A trajectory will be labeled as an outlier if its outlier index is greater than the threshold. To improve the efficiency of spatial-temporal trajectory outlier detection, we present a parallel algorithm PSTOD based on Spark. Finally, we evaluate two algorithms on a real taxi dataset and demonstrate the efficiency and effectiveness. In the future, we can make an effort to improve the precision of distance measurement and broaden the application fields.

Acknowledgments

This work is supported by National Natural Science Foundation of China under Grants No. 41471371 and the Natural Science Foundation of Jiangsu Higher Education Institutions of China under Grant No. 15KJB520022.

References

[1] Breunig M M, Kriegel H P, Ng R T, et al. LOF: identifying density-based local outliers [J]. Acm Sigmod Record, 2000, 29(2):93-104.

[2] E.M.Knorr, R.T.Ng, V.Tucakov. Distance-based outliers: Algorithms and applications [J]. VLDB Journal, 2000, 8: 237-253.

[3] A.Struyf, P.J.Rousseeuw. High-dimensional Computation of the Deepest Location [J]. Computational Statistics and Data Analysis, 2000, 34: 415-426.

[4] J.G.Lee, J.Han, X.Li. Trajectory Outlier Detection: A Partition-and-Detect Framework[C]// Proceedings of the 24th International Conference on Data Engineering, Washington: IEEE Computer Society, 2008: 140-149.

[5] Z Liu, D Pi, J Jiang. Density-based trajectory outlier detection algorithm [J]. Journal of Systems Engineering & Electronics, 2013, 24(2):335-340

[6] G Yuan,S Xia,L Zhang,Y Zhou,C Ji. Trajectory Outlier Detection Algorithm Based on Structural Features [J]. Journal of Computational Information Systems, 2011, 7(11).

[7] Wang Z, Bovik A C, Sheikh H R, et al. Image quality assessment: from error visibility to structural similarity [J]. IEEE Transactions on Image Processing, 2004, 13(4):600 - 612.

[8] Ge Y, Xiong H, Zhou Z H, et al. Top-Eye: top-k evolving trajectory outlier detection[C]// Proceedings of the 19th ACM Conference on Information and Knowledge Management, CIKM 2010, Toronto, Ontario, Canada, October 26-30, 2010. 2010:1733-1736.

[9] Zhang D, Li N, Zhou Z H, et al. iBAT: detecting anomalous taxi trajectories from GPS traces[C]// Proceedings of the 13th international conference on Ubiquitous computing. ACM, 2011:99-108.

[10] Yu Y, Cao L, Rundensteiner E A, et al. Detecting moving object outliers in massive-scale trajectory streams[C]// Acm Sigkdd International Conference on Knowledge Discovery & Data Mining. ACM, 2014:422-431.

[11] Birant D, Kut A. Spatio-Temporal Outlier Detection in Large Databases [J]. Science, 2006, 14(4):291-297.

[12] Ying X, Xu Z, Yin W G. Cluster-Based Congestion Outlier Detection Method on Trajectory Data[C]// FSKD 2009, Sixth International Conference on Fuzzy Systems and Knowledge Discovery, Tianjin, China, 14-16 August 2009, 6 Volumes. 2009:243-247.

[13] Zhu J, Jiang W, Liu A, et al. Time-Dependent Popular Routes Based Trajectory Outlier Detection [M]// Web Information Systems Engineering-WISE 2015. Springer International Publishing, 2015.

[14] Cheng T, Li Z. A Multiscale Approach for Spatio-Temporal Outlier Detection [J]. Transactions in Gis, 2006, 10(2):253–263.

[15] Li Y, Chung W, Bae H Y. A Novel Outlier Detection Method for Spatio-Temporal Trajectory Data [J]. Lecture Notes in Computer Science, 2011, 6935:698-707.

[16] Shen M, Liu D R, Shann S H. Outlier detection from vehicle trajectories to discover roaming events [J]. Information Sciences, 2015, 294:242-254.

[17] Chen C, Zhang D, Castro P S, et al. Real-Time Detection of Anomalous Taxi Trajectories from GPS Traces [M]// Mobile and Ubiquitous Systems: Computing, Networking, and Services. Springer Berlin Heidelberg, 2012:63-74.

[18] Fontes V C, Alencar L A D, Renso C, et al. Discovering Trajectory Outliers between Regions of Interest [J]. 2013.

[19] Gupta M, Gao J, Aggarwal C, et al. Outlier Detection for Temporal Data: A Survey [J]. IEEE Transactions on Knowledge & Data Engineering, 2014, 26(9):1-1.