A Novel Moving Object Trajectories Clustering Approach for Very Large Datasets

Jian Dai*†
*University of Chinese Academy of Sciences, Beijing, China
†National Engineering Center of Fundamental Software,
Institute of Software, Chinese Academy of Sciences, Beijing, China
daijian@nfs.iscas.ac.cn

Abstract—**Witnessing a rapid and continuous diffusion of mobile devices such as on-board GPS navigators, smartphones and tablet computers, we also facing the challenge that how to effectively and efficiently cluster the trajectories derived from these devices. Such an algorithm can benefit a range of applications and services, including intelligent transportation systems, personalized route planner, smart sports and prediction-based social network, etc. Coping with moving object trajectory clustering has long been an important research direction on moving object pattern mining, but still remains two difficulties to deal with. First, how to define the similarity between two trajectories. Second, how to cluster a large number of trajectories efficiently. To define the similarity, a key point is to choose appropriate granularity. If fine-grained policy is adopted, we can expect a more precise result. However, it requires a lot of computations. On the other hand, if coarse-grained strategy is used, we can quickly cluster very large number of trajectories. However, the results are not as good as fine-grained algorithm. Hence, it's a dilemma. To cluster a large number of trajectories, a lot of algorithms are proposed. Some of them are focusing on finding moving objects which move close to each other for a time duration, such as: swarm. Some of them employ the notion of density connection in order to enable the formulation of arbitrary shapes of groups, such as: convoy. In this paper, we aim at find the similarity between the trajectory sets, where each set is generated by a moving object. In this way, we can explore the personalized patterns which are more crucial to a lot of real-world applications as mentioned above. First, we define the similarity between two moving object trajectories. Then, we give an algorithm to cluster trajectory sets. Finally, intensive experiments are conducted and the results prove both the effectiveness and efficiency of our algorithm.**

Keywords-Moving Object Trajectories; clustering; algorithm

I. INTRODUCTION

Technically, a mobile device can report the location coordinates (including longitude, latitude and altitude) in high sampling rates (one record per second for instance). And based on these samples, there are rich techniques to perform effectively clustering. However, with the increasing number of such mobile devices, corresponding trajectory database cannot support these high-sampling-rate observations[1, 6–8]. Considering the situation in Beijing as an example, there are about 80,000 taxis equipped with GPS devices. If each car send its location to the trajectory database per second, the amount of data produced will be up to 1 TB per day. The data volume will bring a heavy burden of storage and communications, and wasted a lot of energy. Therefore, as a matter of fact, a car averagely send its location to the database in every one minutes, and even a few minutes in some districts. Consequently, the low-sampling-rate records in the database bring new challenges to the previous trajectory clustering algorithms.

Trajectory clustering can be applied to many scenarios and applications[2–5]. Basically, we divide it into two categories: neighborhood identification (NI), that is to find moving objects which move close to each other; and activity recognition (AR), that is to extract high-level activity and goal related information from low-level sensor readings. Both of these two categories are trying to discover more semantic information beyond the raw trajectory data. Despite the advantage of previous researches in providing the position and the context information of a mobile user, we are interested in another way clustering. In this paper, we also aim at exploring semantic information from the data. However, there are two differences between our work and previous clustering algorithms. First, the data in our algorithm is in low-sampling-rate. Second, the semantic information is about the correlations between two moving objects. Specifically, through the trajectories, we can say two persons (Peter and Audrey for example) are closer friends than the other two persons (Peter and Carl), because Peter and Audrey usually go to restaurants and cinemas together while Peter and Carl only go to classrooms together.

To fulfill our goal, we need resolve the following two problems. The first one is how to define the similarity between two trajectories, which is a non-trivial problem. The second one is how to cluster trajectory sets derived from moving objects. In our proposal, considering a trajectory is typically made up of a series of coordinate points with time stamps, we design a similarity metric which utilize the data providing by each coordinate point. And in road network constrained environment, to evaluate the similarity between two trajectory sets derived from two moving objects, in turn, becomes a problem of finding maximum common road trajectories. Accordingly, we give an algorithm to cluster the trajectory sets based on threshold. To validate our algorithm, we firstly collect data from mobile phones with several volunteers in our lab for tracking and validating daily routines. Then we implemented and tested our clustering algorithm on real trajectories derived from these volunteers to find the correlations between two or more friends who share their daily routines. Our results show that our clustering algorithm is effective even in low-sampling-rate conditions and efficient even the number of trajectories is very large.

The rest of this paper is structured as follows: in section 2.1 and section 2.2 we define similarity measures not only between two trajectories, but two trajectory sets. In section 2.3 we give the details about our clustering algorithm. In section 3, we conduct and experimental study over real trajectories in order to evaluate our approach. Finally, the conclusions of our research along with ideas for future work are summarized in section 4.

II. Moving Object Trajectories Clustering

In animal tracking and some other scenarios, the objects typically move freely in the water, on the surface of the Earth, or in the air. In such cases, the movement domain is often modeled as two- or three-dimensional Euclidean space. In this paper, we consider the settings where the objects are vehicles, and treat the movement domain as a spatial graph that models a road network. Furthermore, the spatial extent of a moving object is ignored so that the position of an moving object at a given time can be treated as a point[9, 10].

The scenario in this paper is that multiple objects, each equipped with a mobile device with a unique identifier id, report its location at every specific time interval. Each device samples the location of its object and is capable of transmitting its location to a central trajectory database. The update policy we used is to report its location every predefined time interval. For example, devices may take a position sample at a fixed frequency such as every second. A location record has the format (t, x_t, y_t, id), where the sampled location at time t is (x_t, y_t). The trajectory of an moving object is assumed as a polyline which consists of a sequence of samples. And the location of a moving object in-between the two consecutive samples is approximated as $\frac{P_t - P_{t_i}}{t - t_i} = \frac{P_{i+1} - P_i}{t_{i+1} - t_i}$, where P_{i+1} and P_i are the two observations at time t_{i+1} and t_i, respectively.

A. Similarity between Two Trajectories

In the time series database, the similarity of two time series data, where each has *n* value, is given by the raw Euclidean distance between vectors in R^n[14]. Since the moving object trajectory also has the time series data feature, we borrow the core idea from time series database and define the similarity between two trajectories. First, let's review the corresponding definition in time series database. When there are two time series data, $c =< w_1, w_2, \cdots, w_n >$, $c' =< w'_1, w'_2, \cdots, w'_n >$, the distance $D(c, c')$ is defined as follows:

$$D(c, c') = \sqrt{((w_1 - w'_1)^2 + \cdots + (w_n - w'_n)^2)^2} \quad (1)$$

A straightforward extension to the above definition is to treat the two trajectories as query trajectory and the to-be compared trajectory, respectively. Here, let χ be the set of discrete trajectories stored in the database, and each $\chi_i(\chi_i \in \chi)$ is a discrete trajectory, such as $\chi_i =< x_1, x_2, \cdots, x_m >$. The query trajectory χ_q is given as $\chi_q =< x_1, x_2, \cdots, x_n >$. The similarity query can then be defined using $\chi_q\chi$, and the previous defined distance between two time series vectors[11–13].

However, sensor devices often lose the location reports due to various reasons. For example, as illustrated in figure 1(a), a trajectory may lost the report at time slots *F* and *G*. To cope with such cases, we need modify our similarity definition, because the original Euclidean distance can be only used in the case where all trajectory are measured by the same interval, that is $\Delta t = t_{i+1} - t_i (i = 1, \cdots, n-1)$, where t_i is an interval from the time when x_i is measured. Therefore, we define a temporal normalized trajectory similarity measurement for two trajectories, as follows.

Given a trajectory χ defined for time interval $[t_S, t_E]$, and a sampling number m, the temporal normalized discrete trajectory $\chi\Delta t$ is defined as follows:

$$\chi_{\Delta t} =< \chi(t_S), \chi(t_S + \Delta t), \cdots, \chi(t_S + m\Delta t) > \quad (2)$$

where $t_S + m\Delta t = t_E$.Intuitively, this discrete trajectory $\chi_{\Delta t}$ is the re-sampled trajectory per fixed interval Δt from χ. In other words, $\chi_{\Delta t}$ is generated by dividing χ into equal interval Δt. For discrete trajectory χ, we can use the piecewise linear approximation instead of χ. In the case of Figure 1(a), the temporal normalized discrete trajectory (right) is generated from the approximate trajectory (left).

Given two trajectories χ and χ' with the same temporal length (i.e. $L_T(\chi) = L_T(\chi')$) and a natural number m, the spatiotemporal distance (similarity) $D_{TS}(\chi, \chi')$ between χ and χ' is defined as follows:

$$D_{TS}(\chi, \chi') = \frac{1}{m+1} \sqrt{\sum_{i=0}^{m} D(X_{\chi\Delta t}(i), X_{\chi'\Delta t}(i))} \quad (3)$$

where $\Delta t = \frac{L_T(\chi)}{m} = \frac{L_T(\chi')}{m}$.Note that $D_{TS}(\lambda, \lambda')$ can be defined as $D_{TS}(X, X')$. In this definition, the similarity is the Euclidean distance between trajectories represented as m + 1 dimensional vectors, and the interval of each trajectory is normalized. Using this definition, it is possible to find trajectories whose shape is more similar to the query trajectory than can be found using previous methods.

B. Similarity between Two Trajectory Sets

Intuitively, the more common trajectory road segments that two trajectory sets share, the more similar the two trajectory sets. The common trajectory road segments has been extended to a broader extent with a threshold which specifies the distance between two trajectories. As shown in figure 1(b), MO1 and MO2 are clustered into one set. Meanwhile, MO3 and MO4 are clustered together. To identify the similarity between trajectory sets, only using the trajectory similarity is not enough. Therefore, we introduce another similarity measure TRJ-TFIDF which is derived from the original TFIDF. TFIDF(term frequencyCinverse document frequency), is a numerical statistic which are usually applied to reflect how important a word is to a document in a collection or corpus. The TFIDF value increases proportionally to the number of

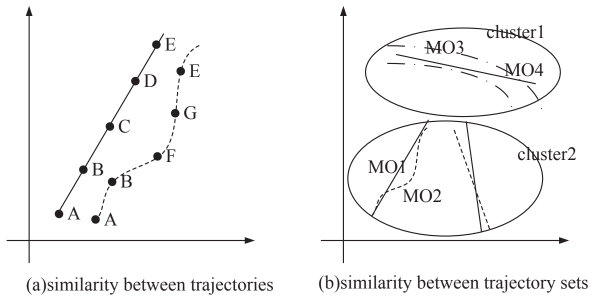

A
B
C
D
E
A
B
F
G
E
(a)similarity between trajectories
cluster1
MO3
MO4
cluster2
MO1
MO2
(b)similarity between trajectory sets

Fig. 1. Similarity between Two Trajectories

times a word appears in the document, but is offset by the frequency of the word in the corpus, which helps to control for the fact that some words are generally more common than others. We employ the idea of TFIDF in our algorithm to compute the similarity between two trajectory sets. The corresponding relationships are that the trajectory set is viewed as a document, specific road segments are treated as the words and some popular road segments are used to control the fact that some segments are more general than others.

Specifically, based on the similarity between two trajectories, we can discover independent **S**imilar **R**oad Segment **S**et (**SRS** for short), where each set includes a collection of similar road segments, and can be used interchangeably. TRJ-TFIDF is the product of two statistics, road segment frequency and inverse trajectory frequency. Various ways for determining the exact values of both statistics exist. In the case of the road segment frequency TRJ-TF(t,d), we adopt the choice that is to simply use the raw frequency of a road segment in a trajectory set, i.e. the number of times that a road segment t occurs in a trajectory set d. Since we denote raw frequency of t as TRJ-F(t,d), the simple TRJ-TF scheme is TRJ-TF(t,d)=TRJ-F(t,d).

The inverse trajectory frequency is a measure of whether the road segment is common or rare across all trajectories. It is obtained by dividing the total number of trajectories by the number of trajectories containing the road segments, and then taking the logarithm of that quotient. Then TFIDF is calculated as TRJ-TFIDF=TRJ-TF$\times$TRJ-IDF.

C. Clustering Algorithm

Clustering of trajectories is perhaps one of the most fundamental operations used in various types of trajectory pattern mining, since the discovery of trajectory patterns typically involves the process of grouping similar positions, trajectories, and objects. Clustering is a fundamental concept in data mining, due to its wide spectrum of applications. Clustering techniques for different types of data are likely to vary significantly with respect to the underlying concepts and the specific techniques employed. In our algorithm, clustering is the process of organizing moving objects into groups so that the members of a group are similar and so that members of distinct groups are dissimilar, according to the definition of similarity we give above.

We use the polyline model to represent a trajectory and polyline set to describe the moving object movements. With these models, our algorithm treats a road segment of a trajectory as a minimum unit when computing the distance between two trajectories and uses a road segment of a road network as a minimum unit(word) when computing the distance between two trajectory set. Aiming at clustering moving objects with more similarity of the trajectory sets into one cluster and separating moving objects with more dissimilarity of the trajectory sets far away, it is essential to design an effective clustering algorithm. In this subsection, we offer details on our algorithm.

Given a road network G, we generate the adjacent matrix GM by computing the road segments and junctions of it, where each road segment is the minimum unit linking two adjacent junctions. The weights of the edges is depicted in GM as the value of the cell to represent the length of corresponding road segment. Each moving object has its own adjacent matrix $GM\text{-}MO$ to depict the trajectory set collected from its historical trajectory data. Similar to the classical K-means approach, we firstly arbitrarily select k centroids as the cluster centers. Then, iteratively computing the distances(according to the similarity measures defined in previous two subsections) and performing the assignment step(equivalent to expectation step) and update step(equivalent to maximization step). Finally, after the converge to the global optimum, output the results. As shown in algorithm 1, our clustering algorithm is an iteratively process. The main function converges when the subroutine TRJ-TFIDF distance computation cannot change the current centroids of our clusters.

III. Experiments

In this section, we present an experimental study in order to evaluate our approach. The experiments were run on a PC with Intel Core Duo at 2.53 GHz, 4 GB RAM and 240 GB hard disk. We implemented the proposed algorithms using C++.

A. Datasets and Configurations

To the best of our knowledge in the trajectory database domain there is no available real datasets and their clusters to be used as the ground truth for benchmarking. To validate our algorithm, we collect data from smart phones with several volunteers in our lab for tracking and validating daily routines. And we manually cluster these trajectory sets with a threshold δ. Then we extract clusters via our algorithms. Finally, we compute the accuracy by the comparison of the manually clustered ones and the extracted ones. We also recorded the running times for various trajectory set sizes to validate the efficiency. The initial dataset consists of the trajectory sets of 25 persons in the area of Beijing between August and September 2012. There are 58,964 position records including the position information, identifiers and the time information. The time interval between two consecutive samples are 5 seconds.

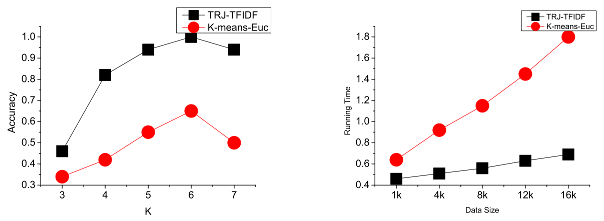
B. Results and Analysis

We implemented a variation of the original K-means algorithm accompanying with Euclidean distance appropriately

Algorithm 1 Clustering Algorithm

```
Input:   trajectory sets:$S_{trj}$,road network:$G$
Output:   clustered moving object sets:$S_{mo}$
    subroutine: TRJ-TFIDF Distance Computation
 1: for each trajectory $trj_i$ in $S_{trj}(MO_i)$ do
 2:     for each trajectory $trj_j$ in $S_{trj}(MO_j)$ do
 3:         $D_{TS}(trj_i, trj_j)$
 4:         if $(\frac{1}{m+1}\sqrt{\sum_{k=0}^{m} D(X_{trj_i\Delta t}(k), X_{trj_j\Delta t}(k))} < \delta)$ then
 5:             $F(rs(i,j), rs \in G) += 1$
 6:         end if
 7:     end for
 8: end for
 9: for each trajectory $trj_i$ in $S_{trj}(MO_i)$ do
10:     for each trajectory $trj_j$ in $S_{trj}(MO_j)$ do
11:         $F(rs(i,j), rs \in G) += 1$
12:     end for
13: end for
14: $TRJ\text{-}TF = \frac{F(rs(i,j),rs\in G)}{maxF(rs(\star,j),rs\in G):rs(\star,j)\in G}$
15: $TRJ\text{-}IDF = \log \frac{|rs\in G|}{|rs\in G:t\in d|}$
16: $TRJ\text{-}TFIDF = TRJ\text{-}TF \times TRJ\text{-}IDF$
    main: Clustering
17: generate clusters according to the $TFIDF$ distance
18: locate new center of the clusters by $\frac{\sum_{i=0}^{k}(loc_{cluster(j)})}{k}$
19: calculate the distances between the trajectory sets and new
    centers
20: re-generate clusters according to the $TFIDF$ distance
```

refined for the comparison with our algorithm. In order to be as fair as possible, the modified K-means is called as K-means-Euc which also uses the same raw trajectory data, along with the distance roughly determined between MBRs. In our first experiment we tested the effect of our parameter k which is the number of clusters. We then recorded the running times of our algorithm and K-means-Euc, respectively. The corresponding results are shown in Figure 2. Note that the accuracy is computed by the proportion of extracted clusters in total real clusters.



(a) Accuracy (b) Running Time

Fig. 2. Experiment Results

Clearly, Figure 2 demonstrates that our algorithm achieves very good results, with a typical accuracy above 80%, while it reaches nearly 100% when the cluster size(k) is set to appropriate value, regardless of the value of other parameters, where our clustering algorithm can capture the inherent characteristics of the trajectories. Moreover, our algorithm outperforms the K-means-Euc always, which proves the effectiveness of the algorithm. On the other hand, when applying the same experimental settings and trajectory size over our algorithm and the K-means-Euc, our algorithm also outperforms the K-means-Euc at the running time. With the increasing trajectory size, both running times of the two algorithms are undoubtedly increasing. Nevertheless, the general observation obtained from our experiments, is that the running time of our algorithm is more stable than the K-means-Euc, which indicates that our algorithm is more suitable to cluster trajectory in large size.

IV. Conclusion

In this paper, we propose a novel moving object trajectory clustering algorithm, motivated by the observation that the moving objects usually moving together are in closer relationship than others. Aiming at this goal, we firstly define the similarity measure between trajectories and the similarity measure between trajectory sets. Then accordingly, based on these two similarity measures, we describe the details of our novel clustering algorithm. Finally, we has proved the effectiveness and the efficacy of our algorithm by intensive experiments.

We plan to adopt some smarter sampling strategies for large trajectory data so as to adaptively follow the inherent distributions of the trajectories. And another clear future work is to develop an index-based version of our clustering algorithm to further improve the performance of our algorithm particularly on large trajectory datasets.

References

[1] D. Pfoser, and C. S. Jensen, Capturing the Uncertainty of Moving-Object Representations. In Proc. of SSD, 1999
[2] L. Chen, M. Tamer ¨zsu, and V. Oria, Robust and Fast Similarity Search for Moving Object Trajectories, In Proc. of SIGMOD, 2005
[3] E. Frentzos, K. Gratsias, Y. Theodoridis. Index-based Most Similar Trajectory Search, In Proc. of ICDE, 2007
[4] F. Giannotti, M. Nanni, D. Pedreschi, F. Pinelli, Trajectory Pattern Mining, In Proc. of SIGKDD, 2007
[5] J.-G. Lee, J. Han, and K.-Y. Whang, Trajectory clustering: a partition-and-group framework. In Proc. of SIGMOD,2007.
[6] N. Pelekis, I. Kopanakis, I. Ntoutsi, G. Marketos, G. Andrienko and Y. Theodoridis. Similarity Search in Trajectory Databases. In Proc. of TIME, 2007
[7] O. Abul, F. Bonchi, M. Nanni, Never Walk Alone: Uncertainty for Anonymity in Moving Objects Databases, In Proc. of ICDE, 2008.
[8] T. Zhang, R. Ramakrishnan, and M. Livny, BIRCH: An Efficient Data Clustering Method for Very Large Databases, In Proc. of SIGMOD, 1996.
[9] H. Qiu, E.R. Hancock 'Graph simplification and matching using commute times', Int. J. Pattern Recognition (PR) 40(10):2874 − 2889 (2007)
[10] A. Robles-Kelly, E.R. Hancock, 'Graph edit distance from spectral seriation', IEEE Trans. Pattern Anal. Mach. Intell. 27 (2005) $365C378$
[11] Y. Yanagisawa, Jun-ichi Akahani, T. Satoh 'Shape-Based Similarity Query for Trajectory of Mobile Objects'. In MDM2003 Conference Proceedings, 63 − 77, 2002
[12] Y.-S. Moon, K.-Y.Whang, andW.-S. Han. 'General match: A subsequence matching method in time-series databases based on generalized windows'.In SIGMOD 2002 Conference Proceedings, $382C393$, 2002
[13] O. Wolfson, B. Xu, S.Ch amberlain, and L.Jian g. Moving objects databases: Issues and solutions.I n Statistical and Scientific Database Management (SSDM98) Conference Proceedings, $111C122$, 1998.
[14] A.P .Si stla, O.W olfson, S.C hamberlain, and S.Da o. Modeling and querying moving objects.In ICDE97 Proceedings, $422C432$, 1997.
[15] L.P. Cordella, P.Foggia, C. Sansone, M. Vento. 'A (Sub)Graph Isomorphism Algorithm for Matching Large Graphs'. IEEE Trans. Pattern Anal. Mach. Intell. (PAMI) 26(10):1367-1372 (2004)