

A Framework for Online Mobility Pattern Discovery from Trajectory Data Streams

Ticiania L. Coelho da Silva*, Karine Zeitouni†, José A. F. de Macêdo*, Marco A. Casanova‡

*Federal University of Ceará, Brazil

Email: ticianalc@ufc.br, jose.macedo@lia.ufc.br

†Université de Versailles-St-Quentin, France

Email: karine.zeitouni@uvsq.fr

‡Department of Informatics - PUC-Rio, Brazil

Email: casanova@inf.puc-rio.br

Abstract—Trajectory pattern mining allows characterizing movement behavior, which leverages new applications and services. Most existing approaches analyse the whole object trajectory rather than the current movement. Besides existing approaches for online pattern discovery are restricted to instantaneous positions. Subsequently, they fail to capture the movement behaviour along time. By continuously tracking moving objects sub-trajectories at each time window, rather than just the last position, it becomes feasible to gain insight on the current behaviour, and potentially detect mobility patterns in real time. This demonstration presents a novel framework for online mobility pattern discovery in sub-trajectory data streams. Key innovations include: (i) Online discovery of mobility patterns and pattern evolution by tracking the sub-trajectories of moving objects; (ii) A novel structure, called *micro-group*, to represent the relationship among moving objects; and (iii) An incremental algorithm to maintain *micro-groups* and to capture their evolution on highly dynamic sub-trajectory data. We present various demonstration scenarios using a real data set.

I. INTRODUCTION

The huge volume of collected trajectories opens new opportunities for discovering the hidden patterns about the mobility behavior of individuals - persons, animals or vehicles - as well as of groups of individuals sharing similar trajectories for a certain time period. Usually, trajectory analysis is done off-line, i.e., by applying data analysis and data mining techniques on historical data [1]. This allows characterizing the past movements of the objects, but not the current mobility patterns. Nowadays, many applications exist that continuously (e.g., every second or every minute) track the trajectory of moving objects. Analyzing these data in real time may bring a real added-value to the understanding of city dynamics by detecting regularities and anomalies in moving objects, which is essential for decision making. Among these patterns, in this paper, we investigate how the trajectories of groups of individuals emerge and evolve, based on a technique of sub-trajectory cluster analysis. Such an investigation may help search for effective traffic re-engineering or dynamically detect events or incidents at city level, for example.

Two important properties of a tracking application are the incremental nature of the data and the fact that such data may reach huge volumes as time goes. Therefore, finding

patterns in such data in (quasi) real time is challenging. Since all the tracked moving objects change their positions over time, movement patterns also evolve in time. Furthermore, new moving objects may start sending their positions, while others may stop.

Several approaches for online mobility pattern discovery using moving objects position have been proposed, but they are restricted to instantaneous positions. Subsequently, they fail to capture the displacement behavior along time. By continuously tracking moving objects sub-trajectories at each time window, rather than just the last position, it becomes possible to gain insight on the current behavior and potentially detect suspicious behavior in real time. Although the majority of mobility patterns approaches (including trajectory clustering, flocks, convoys and swarms [1]) analyze historical (sub)trajectory data, no solution exists to find and maintain these patterns for sub-trajectories in real time.

In this paper, we address two problems: (1) online discovery of mobility patterns and (2) maintaining pattern evolution by tracking the sub-trajectories of moving objects at each time window. Since most objects change their sub-trajectories over time, new moving objects appear as well as others disappear from the system during a time window. We define a new structure, called *micro-group*, to incrementally maintain the relationship among moving objects. To the best of the authors' knowledge, this work is the first to tackle this problem. To solve this problem, we first propose a framework discussed in Section II. Then, we present the demonstration scenarios in Section III.

II. FRAMEWORK

Our framework (see Figure 1), firstly proposed in [2], follows four main steps: (i) collect the trajectory data stream at each time window; (ii) apply a similarity measure, (iii) maintain the *micro-group(s)*; and (iv) discover the mobility patterns.

A *trajectory* is a sequence of locations of a moving object at each time-stamp and is denoted $TR_j = p_1 p_2 \dots p_r \dots p_{len_j}$. Here, p_k ($1 \leq k \leq len_j$) is a point (x_k, y_k, t_k) in a three dimensional space, where (x_k, y_k) indicates the location of

the object at time t_k . The length len_j of the trajectories in a set may not be the same.

First step. Let $i = [t, t + \delta t]$ be the time window observed for a set I_i of moving object sub-trajectories. Let $I_i = \{(o_1, ST_{1,i}), (o_2, ST_{2,i}), \dots, (o_n, ST_{n,i})\}$, where $ST_{j,i}$ is the sub-trajectory of moving object o_j in I_i . Therefore, I_i is the stream at the time window i .

Second step. To measure the similarity between two moving objects sub-trajectories $ST_{k,i}, ST_{j,i}$ at the time window i , we implement the synchronous Euclidean distance [3], which accounts for time, space, and direction. However, our framework is suitable to any distance function for trajectories. Throughout this paper, $dist(ST_{k,i}, ST_{j,i})$ denotes the distance function between sub-trajectories.

From a group of moving object sub-trajectories $S_i = \{(o_1, ST_{1,i}), (o_2, ST_{2,i}), \dots, (o_m, ST_{m,i})\}$ at the time window i , we define the *representative trajectory* as a pair $(o_j, ST_{j,i})$ composed of one moving object and its sub-trajectory which is similar to the major behavior of S_i . To choose $(o_j, ST_{j,i})$ as the representative, our approach checks the number of moving objects that have their sub-trajectories similar to $ST_{j,i}$ and uses a Gaussian kernel function as our *voting function* to estimate the representativeness of $ST_{j,i}$. This "voting" function is also used in [4] and it has been widely used in a variety of applications of pattern recognition.

Definition II.1. Consider a set of moving object sub-trajectories $S_i = \{(o_1, ST_{1,i}), (o_2, ST_{2,i}), \dots, (o_m, ST_{m,i})\}$ at the time window i . Let ρ be a representativeness threshold, σ be a standard deviation (sometimes called the Gaussian width), ϵ be a given distance threshold and τ be a size/density minimum threshold. Then, $(o_j, ST_{j,i})$ is a *representative trajectory* of S_i if and only if:

- 1) $\forall (o_k, ST_{k,i}) \in S_i$,

$$\text{voting}(ST_{j,i}, ST_{k,i}) = e^{-\frac{dist^2(ST_{j,i}, ST_{k,i})}{2\sigma^2}} > \rho$$
- 2) $|N_\epsilon(o_j)| \geq \tau$, where $N_\epsilon(o_j)$ is defined as

$$N_\epsilon(o_j) = \{(o_k, ST_{k,i}) \in S_i | dist(ST_{j,i}, ST_{k,i}) \leq \epsilon\}.$$

The parameter σ shows how fast the "voting" function decreases with the distance. The intuition behind the relationship of "distance" and "voting" function is: If "distance" is close to zero, then "voting" is close to its maximum value. This means that, if $ST_{j,i}$ is very close (in time, space and direction, for example) to $ST_{k,i}$, then $(o_j, ST_{j,i})$ is a candidate to be the representative. Otherwise, if the distance is high, the "voting" function is close to its minimum value, meaning that $ST_{j,i}$ is very far way from $ST_{k,i}$ and, therefore, $ST_{k,i}$ is not be represented by $ST_{j,i}$.

We define a new structure, called *micro-group*, to capture and maintain small and dense groups of moving objects around the representatives at each time window. This adapts to many situations such as families with young children staying together or persons sharing the same trip using a public transport.

Definition II.2. Let a set of moving object sub-trajectory $S_i = \{(o_1, ST_{1,i}), (o_2, ST_{2,i}), \dots, (o_m, ST_{m,i})\}$ at the time

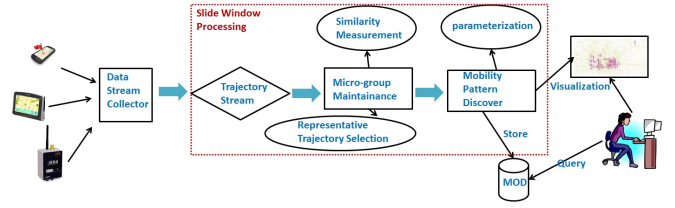


Figure 1. Steps of our proposed framework

window i , O_i be the set of moving objects in S_i , ρ be a representativeness threshold, ϵ be a distance threshold and τ be a size/density minimum threshold. Then, a set of objects g is a *micro-group* if and only if:

- 1) $g \subseteq O_i$
- 2) $\exists o_j \in g$, such that $(o_j, ST_{j,i})$ is a representative trajectory of g w.r.t. ϵ, τ and ρ .

The ρ value can be chosen according to the maximum allowed distance between the representative sub-trajectory and the farthest member of a micro-group. The ϵ and τ thresholds capture the density around the representative trajectory, similarly to DBSCAN.

Third step. We propose an algorithm to incrementally maintain each micro-group (since all moving objects update their sub-trajectories, some moving objects may leave or join a micro-group) and to capture its evolution patterns. At the initialization phase (i.e., a cold-start of the clustering), the representative trajectories are randomly chosen among the core objects (as defined in DBSCAN). Micro-groups are first derived as the objects that vote for these representatives. The most important contribution is the maintenance phase. The intuition behind is: (i) to check, for each micro-group, whether the representative is still valid in the next time window (it survives), otherwise, the micro-group disappears or splits; (ii) to track moving object sub-trajectories that are likely to join the micro-group (e.g., outliers, new objects, and other micro-group objects migrating to another micro-group); (iii) for the remaining objects, a similar initialization process allows creating new micro-groups.

When a micro-group g_i survives, the algorithm checks for each moving object $o_k \in g_i$ if it is still well represented by $(o_j, ST_{j,i})$. If it is not, o_k is deleted from g_i and either it migrates to another micro-group, or it becomes an outlier, or it forms a new micro-group with other outliers. A micro-group g_i splits or disappears when it changes its representative trajectory. If g_i splits, new micro-groups have to be computed using the g_i data. However, if g_i is not dense enough to generate micro-group(s), its moving objects become outliers and g_i disappears.

Fourth step. We use the maintained micro-groups to discover mobility patterns by capturing the evolution of micro-groups over time. Since each micro-group is density based, it is suitable to find sub-trajectory density based clustering (for example, merging micro-groups results in density based sub-trajectory clusters). Furthermore, we are able to online

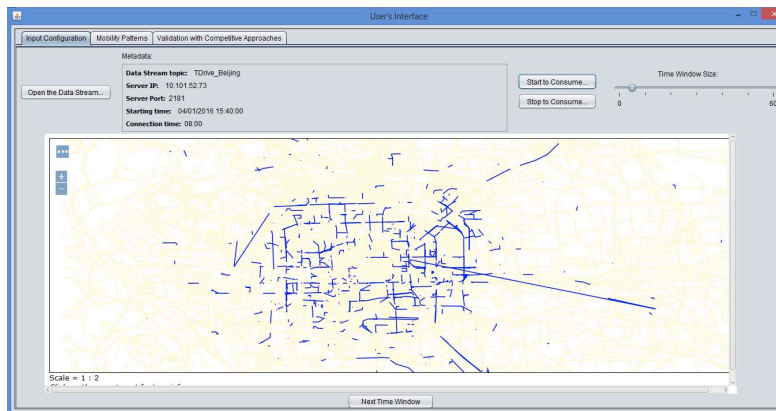


Figure 2. Datastream Control and Input Configuration

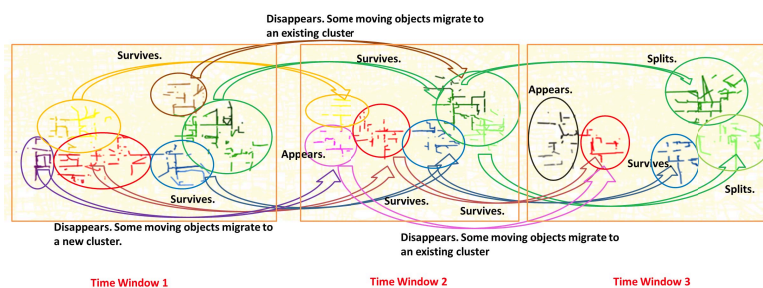


Figure 3. Evolution of the clusters during 3 consecutive time windows

discover patterns such as flocks, convoys, leadership, to name a few. For example, flocks could be derived from micro-groups by performing a light post-processing since it is a subset of the latter. The convoys are also similar to the density based clusters generated by our algorithm. The representative trajectory is the closest notion to leadership. Hence, this information could enrich a Moving Object Database (MOD), allowing new types of queries and visualizations.

III. DEMONSTRATION DETAILS AND SCENARIOS

We cover various scenarios that demonstrate the usefulness of our framework. An interactive interface is the access point for the user to: (i) connect with a server that continuously fires moving object trajectories data stream; (ii) track these moving objects during each time window (whose interval is specified as a parameter); (iii) online discover the mobility patterns in an incremental way; (iv) analyze the performance and quality of the detected mobility patterns; and (v) validate the results against competitive approaches.

Our framework is seeded with real road-network graphs obtained from the OpenStreetMap with the assistance of the GeoServer framework. To examine the quality of our approach, we use a real data set with moving object trajectories. The data set contains a real world GPS data from taxis in Beijing [5]. To consume the streams in real time, we developed an application using the Apache Kafka framework [6] to fire the trajectory data as streams in our framework.

The demonstrated scenarios are as follows.

A. Moving Object (Sub)-trajectories Data Stream

We initially interact with the framework (as showed in Figure 2) by setting the server credentials that fires trajectory data stream. Because our data is dynamic, each moving object sub-trajectory is manipulated in our system with the help of one of these operations: (i) a moving object can *start* sending its sub-trajectory (inserted into the system); (ii) a moving object can *stop* sending its sub-trajectory (deleted from the system); and (iii) all the remaining moving object *update* their sub-trajectories during the time window. At each time window, we aim at tracking these moving objects and incrementally discover mobility patterns (instead of discovering patterns from scratch) using only the current and last time window data.

B. Discover Incremental Mobility Patterns

In this scenario, we show how the user can interact with the framework to run one of the supported mobility patterns. The incremental mobility patterns proposed are: (i) compute *Micro-Group(s)*, (ii) discover *Density Based Sub-trajectory Clusters* and (iii) detect *Trajectory Flocks*. In all cases, our approach maintains the micro-groups and captures their evolution. Our (sub)-trajectory clustering is derived from merging two or more micro-groups at each time window. As we can see in the Figure 3, which shows the clusters (including its evolution)



Figure 4. Running our online algorithm (on the left). Stored results obtained by existing offline approaches (on the right)

and the maintained micro-groups resulted by applying our approach during three consecutive time windows (using TDrive data set). Each cluster and its micro-group(s) are represented by one color. Among the many definitions proposed for the concept of flock pattern, we adopt that introduced in [7], adapted to *trajectory flocks*:

Definition III.1. Let $m \in \mathbb{N}$ be a minimum number of moving objects trajectories, $k \in \mathbb{R}$ be a number of time windows and $r \in \mathbb{R}^+$ be a radius. Given a set of n moving object trajectories in the plane, a *trajectory-flock*(m, k, r), or simply a *flock*, in a time interval I , where the duration of I is at least k time windows, consists of a subset F of the moving objects trajectories with at least m elements and, for every time window within I , there is a circle C of radius r such that C contains all moving objects in F .

In our framework, a flock is a subset of moving objects trajectories (with at least m moving objects) that stay together (according to a radius r) in the same micro-group for more than a time interval threshold (k time windows). The major difference between our flocks and micro-groups is the time interval constraint in the first. At each time window, the framework enables the user to see the micro-groups movement (as well as its evolution) and the mobility patterns discovered (see in the Figure 4, the trajectory clusters detected using our approach in one time window). Techniques as swarm, convoy, herds, gathering, among others, will be supported by our framework in future.

C. Validation Against Competitive Approaches

We are able to run competitive approaches to detect collective patterns among moving objects as flock and trajectory clustering using the approaches that execute from scratch. In this way, the user will be able to execute the longest duration flock algorithm [7], TraClus [8] and the DBSCAN algorithm [9]. However, these approaches are executed offline, since they are not proposed for real time analysis. Furthermore, for each time window we run DBSCAN and TraClus from

scratch giving as input only the sub-trajectories of this time window, and not the "whole" trajectory. We also adapted our competitors to use our distance function, in order to take into account the distance in time, space and direction.

In the validation scenario, our results are compared with the ground truth (DBSCAN and the longest duration flock algorithm) to test the effectiveness (precision and recall) and the efficiency (the running time cost). The user can also visually compare the results from our approach and the ground truth (as seen in Figure 4).

IV. CONCLUSION

In this demonstration, we proposed a framework to discover and track mobility patterns in moving objects trajectory data streams, especially Density Based Sub-trajectory Clusters and Trajectory Flocks. Our experiments validate the proposed algorithms, and allow its comparison with many approaches in the state of the art, from both perspectives of quality and performance.

REFERENCES

- [1] Yu Zheng. Trajectory data mining: an overview. *ACM TIST*, page 29, 2015.
- [2] Ticiano Coelho da Silva, Karine Zeitouni, José de Macêdo, and Marco A Casanova. On-line mobility pattern discovering using trajectory data. In *EDBT*, pages 682–683, 2016.
- [3] Elias Frenzos, Kostas Gratsias, and Yannis Theodoridis. Index-based most similar trajectory search. In *ICDE*, pages 816–825, 2007.
- [4] Costas Panagiotakis, Nikos Pelekis, Ioannis Kopanakis, Emmanuel Ramasso, and Yannis Theodoridis. Segmentation and sampling of moving object trajectories based on representativeness. *TKDE*, pages 1328–1343, 2012.
- [5] Yu Zheng. T-drive trajectory data sample. August 2011.
- [6] Apache Kafka. A high-throughput, distributed messaging system. 2014.
- [7] Joachim Gudmundsson and Marc van Kreveld. Computing longest duration flocks in trajectory data. In *ACM SIGSPATIAL*, pages 35–42, 2006.
- [8] Jae-Gil Lee, Jiawei Han, and Kyu-Young Whang. Trajectory clustering: a partition-and-group framework. In *SIGMOD*, pages 593–604. ACM, 2007.
- [9] Martin Ester, Hans-Peter Kriegel, Jörg S, and Xiaowei Xu. A density-based algorithm for discovering clusters in large spatial databases with noise. In *KDD*, pages 226–231, 1996.