

Clustering Algorithm for Network Constraint Trajectories

Ahmed Kharrat¹, Iulian Sandu Popa¹
Karine Zeitouni¹, Sami Faiz²,

¹ PRiSM Laboratory, University of Versailles
45, avenue des Etats-Unis - 78035 Versailles, France

² LTSIRS, Ecole nationale d'ingénieurs de Tunis
B.P. 37 – 1002 Tunis-Belvédère, Tunisie

Abstract. Spatial data mining is an active topic in spatial databases. This paper proposes a new clustering method for moving object trajectories databases. It applies specifically to trajectories that only lie on a predefined network. The proposed algorithm (NETSCAN) is inspired from the well-known density based algorithms. However, it takes advantage of the network constraint to estimate the object density. Indeed, NETSCAN first computes dense paths in the network based on the moving object count, then, it clusters the sub-trajectories which are similar to the dense paths. The user can adjust the clustering result by setting a density threshold for the dense paths, and a similarity threshold within the clusters. This paper describes the proposed method. An implementation is reported, along with experimental results that show the effectiveness of our approach and the flexibility allowed by the user parameters.

Keywords: Spatial data mining, clustering algorithm, similarity measure, moving objects database, road traffic analysis.

1. INTRODUCTION

Trajectory database management is a relatively new topic of database research, which has emerged due to the profusion of mobile devices and positioning technologies like GPS or recently the RFID (Radio Frequency Identification). Trajectory similarity search forms an important class of

queries in trajectory databases. Beyond querying such complex data, new problems motivate research on the management of moving objects in general and on the spatiotemporal data mining in particular. The clustering of trajectories is part of this research.

We advocate that discovering similar sub-trajectories density based on the network is very useful. There are many examples in real applications. We present hereafter three application scenarios.

1. Knowledge and prediction of the road traffic: Given that the numbers of vehicles increases on the roads, information related to the density on the network becomes very useful for many purposes as navigation, trip planning, etc.
2. Car-sharing: In these last years, the massive use of the private means of transport caused many problems, namely the pollution and also the raising of oil prices. Car-sharing appears as an interesting alternative. Identifying the similar trajectories or even sub-trajectories becomes very useful for such types of applications.
3. Transport planning: At the moment of its creation, each road is planned for certain utilization. Reporting trajectory groups allows assessing the suitability of the road infrastructure with its actual use.

Generally speaking, clustering is a data mining technique extensively used in applications like market research, financial analysis or pattern recognition from images, to name but a few. Several types of clustering algorithms have been proposed among which K-Means (Lloyd, 1981), BIRCH (Zhang et al., 1996), DBSCAN (Ester et al., 1996) and OPTICS (Ankerst et al., 1999). Recent researches on trajectory clustering uses these algorithms while adapting them to the studied domain (Lee et al., 2007), since trajectories are complex objects.

We borrow the idea of density based algorithms such as DBSCAN, and adapt it to trajectories. The key idea behind our approach is that the knowledge of traffic density on the network would allow guiding the clustering of trajectories. We propose a two-step approach. In a first step, we define the similarity between the road segments and use it to group them in dense paths. In a second step, we propose a similarity measure between trajectories, and then we use it to make up the trajectory clusters around the dense paths. As in Lee et al., (2007), the time factor is relaxed in our approach. Nevertheless, we take account of the trajectory orientation. Another feature is that it regroups sub-trajectories rather than the whole of trajectories. Thus, a trajectory can belong to several clusters.

In summary, the contributions of this paper are as follows:

- We propose an innovative and effective method for network constraint trajectory clustering based on the network density.

- We define new similarity functions.
- We implement this framework and conduct an extensive experimentation that validates the method and shows its usefulness.

The rest of this paper is organized as follows. In section 2, we will detail a state of the art on the similarity and the clustering of the trajectories. We will explain our clustering approach in section 3. We will present in section 4 the first phase of the algorithm - named NETSCAN - for the clustering of the road segments. We will describe the second phase of the algorithm afterwards. In section 6 we will present our experimental results. Finally, we will conclude this article in section 7 and will propose some tracks for the pursuit of this research.

2. RELATED WORK

Research on the clustering of moving objects trajectories is closely connected to three topics: trajectories representation, similarity and clustering algorithms. In an orthogonal manner, we also distinguish the following criteria: the aspect of either constraint or free movement of the trajectory, the temporal aspect, the respect of the movement orientation, and finally, the grouping of sub-trajectories or entire trajectories in the clusters. This section describes the main works related to these three topics while situating them in relation with the above criteria.

Concerning the first research topic, many studies have investigated ways that the trajectory of a moving object can be represented. It can be geometric as in Lee et al., (2007) or symbolic as in Hadjieleftheriou et al., (2005). Indeed, if we know in advance the geometry and the topology of the network, we can represent a trajectory by the list of traversed segments, and alternatively, along with the instant to which the object passed from a segment to another, if we respect the temporal aspect. This representation is very precise at the spatial level, but maybe less precise at the temporal level. Nevertheless, it can be sufficient in many cases and especially in our context where the time is relaxed.

Regarding the works related to the similarity of moving objects trajectories, we first mention those in the free moving trajectory context, and then for constrained trajectories. Yanagisawa et al. (2003) focused on the extraction of the individual moving patterns of each object from the trajectories considering both time and location. Their approach uses the shape similarity between lines to retrieve required objects. Shim and Chang (2003) considered the similarity of sub-trajectories and proposed a distance 'K - Warping' algorithm. Lin et al. (2005) focused on the spatial shapes and compared spatial shapes of moving object trajectories by developing algo-

rithms for evaluating OWD (One Way Distance) in both continuous and discrete cases of the trajectories for similarity search. We also find similar approaches in Valachos et al. (2003), Sakurai et al. (2005), and Chen et al. (2005). Valachos et al. (2002) presented an investigation for analysis of spatio-temporal trajectories for moving objects where data contain a great amount of outliers. Therefore, they propose the use of a non metric distance function that is based on the Longest Common Sub Sequences (LCSS) algorithm in conjunction with a Sigmoid Matching function to increase the performance of Euclidean and Time Warping Distance. Zeinalipour-Yazti et al. (2006) introduce a distributed spatiotemporal similarity search based on the LCSS distance measure and propose two new algorithms offering good performances.

All these methods are inappropriate for similarity calculation on road networks since they use the Euclidian distance as a basis rather than the real distance on the road network. It is this last point that motivated the proposition of Hwang et al. (2005) that were the first to propose a similarity measure based on the spatiotemporal distance between two trajectories using the network distance. The algorithm of similar trajectory search consists of two steps: a filtering phase based on the spatial similarity on the road network, and a refinement phase for discovering similar trajectories based on temporal distance. Tiakas et al. (2006) and Chang et al. (2007) also use the same spatiotemporal distance, based on the road network, in their algorithm of similar trajectory search.

Concerning the works on trajectory clustering, we mention the two next ones: Gaffney and Smyth (1999) sustained that the vector based trajectory representation is inadequate in several cases. To surmount this problem, they introduce a model of probabilistic regression mixtures and show how the EM algorithm could be used in trajectory clustering. This approach considers non constraint trajectories. Moreover, it groups similar trajectories as a whole, thus ignoring similar sub-trajectories.

Lee et al. (2007) propose an algorithm named TRACCLUS that groups similar sub-trajectories into a cluster. It consists of two phases: the partitioning of trajectories as line segments and then, the grouping of these according to their similarities. Nevertheless, this work always supposes free moving objects and, moreover, the time is not considered in this work.

As previously mentioned the majority of existing methods for trajectory similarity assumes that objects can move anywhere in the underlying space, and therefore do not support motion constraints. Even works which deal with searching similar trajectories of moving objects in a spatial network, group similar trajectories as a whole. However they could miss common sub-trajectories on one hand. On the other hand, the similarity

measure adopted in these works assume that, to characterize two similar trajectories, it is not necessary to share common road segments, therefore the similarity measure must take into account the closeness of the trajectories. This assumption is not appropriate for a particular type of applications like care-sharing. In order to overcome the inefficiency of the previously described methods we propose a new clustering method that groups network constrained sub-trajectories based on a similarity measure which calculates the rate of inclusion of a trajectory compared to a dense path on the network. Our method is time relaxed.

3. The Clustering Procedure

The clustering consists of creating from a database, groups of similar objects (Han and Kamber 2006). By the clustering procedure, we mean, the steps to be performed ranging from getting a sample of data to processing the results obtained after partitioning the data space into clusters. Typically, the clustering procedures are composed of the following steps: (i) data representation; (ii) defining a similarity criteria, (iii) clustering, (iv) data abstraction (v) quality clustering evaluation.

We propose a two-steps clustering approach. The first phase allows grouping the road sections. Generally, we speak about section clustering. The second phase performs concretely the clustering of the trajectories. Because these kinds of clustering are relative to complex objects, we have to precise for each step; the representation, the similarity and the specific clustering algorithm.

3.1 Data Representation

The network representation is given by the road sections set. Hence, knowing the trajectory set, we compute¹ a transition matrix relative to road network (cf. Fig. 1). This matrix give statistics about the passages across the cross-roads and the turning movements, while reporting the number of moving objects transiting from one section to another connected one. This matrix will be denoted thereafter M and a node $M(i,j)$ will denote the occurrence of moving objects traversing from section S_i to section S_j .

¹ Alternatively, one may use traffic data produced by embedded sensors in the road or cameras.

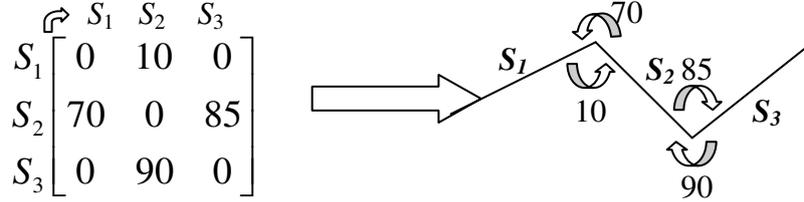


Fig. 1. Transitions Matrix assigned to the road network.

We adopt a symbolic representation of the trajectories (Du Mouza and Rigaux, 2004), (Savary et al., 2004). In this representation, moving objects appear as a sequence of symbols where each one refers to one road section.

$$TR = \langle S_{i1}, \dots, S_{in} \rangle$$

The symbols order shows the movement direction.

3.2 Similarity Measure

The similarity is the base of the clustering operation. We define the similarity at two levels. At the network level, the similarity is computed between two transitions as the difference of their density values. This measure concerns only the consecutive transitions.

$$\text{Sim_segment} (M(i,j) = |M(i,j) - M(j,k)| \tag{3.2.1}$$

At the trajectories level, we define a similarity measure between two trajectories where one is the reference. This measure reflects the resemblance to an object and it is not symmetric. It allows comparing the effective trajectories to a fictive type trajectory. To this end, the similarity is computed as the report between the common length among a trajectory and the reference from one side, and the length of the reference trajectory from another side.

$$\text{Sim_traj} = \text{Lenght} .(\text{common_part}) / \text{Lenght} (\text{ref_traj}) \tag{3.2.2}$$

For the works mentioned in section 2, the similarity relies upon the Euclidian distance and/or shapes (Yanagiswa et al., 2003). This comes from the fact, that they represent the trajectories by their geometry and their forms. Our work adopts quite different criteria because it is situated in the constraint context. It uses the available information relative to the network

density from one side, and the symbolic representation of the trajectories allows obtaining sequence similarities as in Chen et al. (2005), from the other side.

3.3 Clustering

The clustering step corresponds effectively to the grouping phase and aims at deriving a database partitioning as relevant as possible. To achieve this goal, we propose a two-step clustering algorithm that we call NETSCAN. At the first step, it finds the most dense road sections, and merges them to form dense paths on the road network. The second step permits to classify the trajectories of moving objects according these dense paths. The figure 2 reports the main steps of NETSCAN. Our algorithm share the same features with the DBSCAN algorithm and it is based on two steps:

1. Section gathering: find the network paths that are the densest in terms of moving objects transiting on them.
2. Trajectory gathering: for each path, gather the trajectory similar to it.

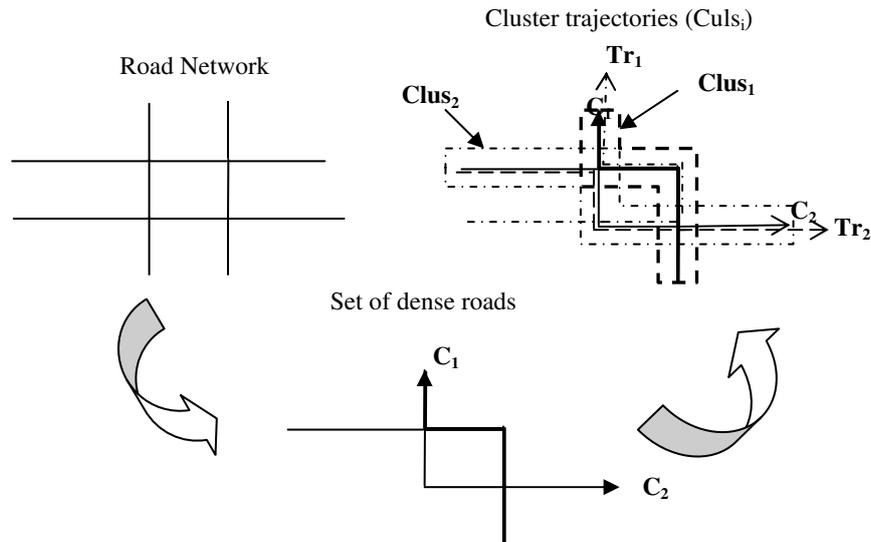


Fig. 2. Trajectory clustering Example by NETSCAN Algorithm

4. Segment Clustering

The first part of the proposed algorithm, NETSCAN-PHASE 1, which is described here, performs the segment clustering.

NETSCAN Algorithm - PHASE 1 /* Dense path discovery*/

Input:

- Set of road segments $S = \{S_1, S_2, \dots, S_{no_segments}\}$
- Transition matrix M .
- Threshold ε -- maximal density difference between neighbour segments.
- Threshold α -- minimal required density for a transition.

Output:

- Ordered dense path set $O = \langle C1, C2, \dots, C_{no_paths} \rangle$.

Algorithm:

1. $O \leftarrow \emptyset$ -- Initialisation
 2. $k \leftarrow 0$
 3. While there exists non marked transitions $M(i,j) \geq \alpha$
 4. $k \leftarrow k+1$
 5. $M(d,f) = \max (M (i, j))$
 6. $C_k \leftarrow \langle S_d, S_f \rangle$ -- generate a new dense path from this transition
 7. Mark the transition $M(d,f)$
 8. While there exists u such as $M(f,u) \geq \alpha$ and u not marked
 -- forward extension
 9. Select $M(f, f_succ)$ such as $|M(d,f) - M(f,u)|$ is minimum
 10. If $|M(d,f) - M(f, f_succ)| \leq \varepsilon$
 11. Insert last (C_k, S_{f_succ})
 12. Mark $M(f, f_succ)$
 13. $d \leftarrow f; f \leftarrow f_succ$ -- extend path
 14. End If
 15. End While
 16. While there exists u such as $M(u,d) \geq \alpha$ and u not marked
 -- backward extension
 17. Select $M(d_prec, d)$ such as $|M(d,f) - M(u,d)|$ is minimum
 18. If $|M(d,f) - M(u,d)| \leq \varepsilon$
 19. Insert first (C_k, S_{d_prec})
 20. Mark $M(d_prec, d)$
 21. $f \leftarrow d; d \leftarrow d_prec$ -- extend path
 22. End If
 23. End While
 24. $O = O \cup C_k$
 25. End While
 26. Return O
-

Fig. 3. NETSCAN Algorithm–Phase 1: clustering segments as dense paths.

This phase is inspired from the density based clustering principle introduced with the DBSCAN algorithm (Ester et al., 1996), while applying it to road segments. It takes as input the set of segments that constitute the road network, the transition matrix (as in section 3.1), a density threshold α and a similarity threshold between the transition densities ε . In this phase, the algorithm firstly finds the dense transitions, i.e. those having maximum value. Afterwards, for each dense transition, it groups the connected segments and transitions that have similar densities, thus creating dense paths. The process begins with the transition having the maximal density. Following, it begins searching the connected transitions in both ways in order to find those with a density ε near to the maximal density. To insure the non reuse of transitions that are included in dense paths, they are marked at the first assignment.

The extension of a dense path is done in both ways if the constraints are verified, i.e., the candidate transition is only marked if it respects the density (α) and similarity (ε) thresholds. The obtained segment clusters correspond to the densest paths in the network. The figure 3 presents the first phase of the NETSCAN algorithm.

The dense paths are represented as a sequence of segments, the same as with the trajectories (section 3.1). Each segment is identified by an associated symbol.

5. Trajectory Clustering

In this section, we present the second phase of the NETSCAN method, which corresponds to the trajectory clustering. This part uses the results obtained in the first phase as presented in the above section. Indeed, the dense paths are considered as natural cluster centres for the MO trajectories.

The trajectory clustering algorithm consists in grouping the trajectories according to their similarity to each dense path generated in phase 1 of NETSCAN. We use the similarity measure as defined in section 3.2. The input of the algorithm consists of the set of the dense paths, the set of trajectories and the threshold σ . For each dense path, it computes the similarity with each trajectory. If the similarity is above the threshold value, then the trajectory is kept in the cluster. More precisely, the common part between the trajectory and the dense path is added to the cluster. The number of returned clusters is equal to the number of dense paths.

NETSCAN Algorithm - PHASE 2

Input:

- Set of paths returned from NETSCAN algorithm - PHASE 1
 $O = \langle C_1, C_2, \dots, C_{no_paths} \rangle$
- Set of MO trajectories $TR = \langle TR_1, TR_2, \dots, TR_{no_trajectories} \rangle$
- Minimal similarity threshold σ

Output:

- Set of clusters $Clus = \{Clus_1, \dots, Clus_{no_paths}\}$

Algorithm:

1. For each path C_i in O
2. For each trajectory TR_j in TR such as TR_j overlaps C_i
3. Compute $Sc = C_i \cap TR_j$ -- set of common segments Sc_k
4. Compute the sum of common segment lengths :

$$L_c = \sum_i length (Sc_i)$$

5. Compute the similarity between C_i and TR_j as:
 $Sim = L_c / length(C_i)$
 6. If $Sim \geq \sigma$
 7. Add Sc to $Clus_i$
 8. End If
 9. End For
 10. Add $Clus_i$ to $Clus$
 11. End For
 12. Return $Clus$
-

Fig. 4. NETSCAN Algorithm – Phase 2: Trajectory Clustering.

6. Experimental Evaluation

In this section, we evaluate the effectiveness of our trajectory clustering algorithm NETSCAN. We describe the experimental data and environment in section 6.1. We discuss the impact of parameter values in sections 6.2 and 6.3. The last section addresses the optimization issue.

6.1 Experimental Setting

NETSCAN algorithm has been implemented on a PC running under Windows XP Professional. The hardware configuration is as follows: a 2.0 GHz AMD Athlon™ 64 Dual Core processor, 1.5 GB main memory, and 80 GB HDD. We use Oracle 10g as data server.

The trajectories may be obtained from several sources, such as Floating Car Data (FCD). However, it is preferable to use for validation and test a public data source. In the context of constraint moving objects, the generator developed by Brinkhoff is the mostly used for benchmarking and test (Brinkhoff, 2002). Based on the road network of San Joaquin bay which consists of 18496 nodes and 24123 edges (i.e. road sections), we apply the generator to produce 2064 trajectories of moving objects on the road network. The figure bellow shows the road map of San Joaquin on the left, and the locations of different moving objects displayed from the generated (virtual) GPS log.

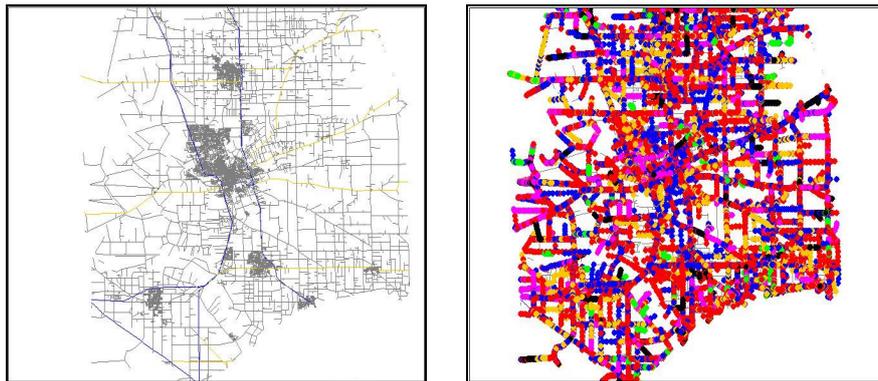


Fig. 5. The road map of San Joaquin.

Based on the above information, it becomes possible to compute the density matrix appropriate to this network. Precisely, for each transition (i,j) in this matrix, we count the occurrences of moving objects traversing it. Figure 6 shows the distribution of the transition densities (i.e. the distribution of the number of moving object going from section i to section j). As shown here, there are 225 transitions that contain 10 moving objects. As one can expect, the number of transitions having a dense traffic (a high transition value, e.g. transition value = 243) is very limited compared to those where it is less dense or either null. Notice that we did not draw the values under 10 for the transition value because the corresponding transi-

tions where too numerous (over 2000), which would render the curve less significant.

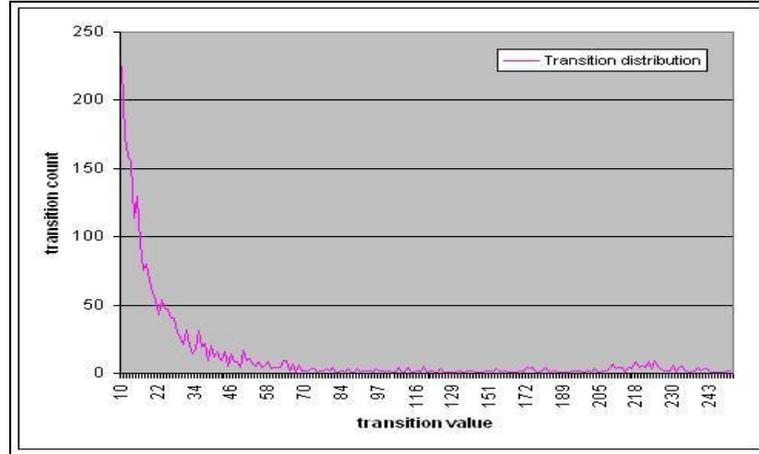


Fig. 6. Transition distribution

6.2 Experimental Results – Phase 1

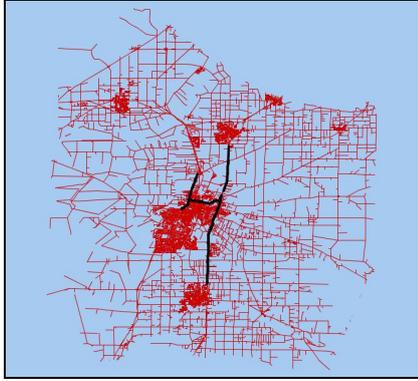
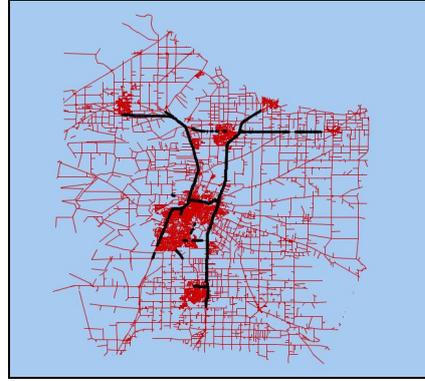
To test the first phase of NETSCAN algorithm, we change the input parameter specifying the density threshold α . We successively test the first phase by setting the following values for α : 10, 50, 100, 200, and 230. We measure the impact of those parameters on the results.

For each test, we evaluate the number of dense transitions, i.e. where the density is above the threshold α , as well as the number of dense paths resulting from Phase 1 of NETSCAN. The experimentation results are summarized in table 1. As an example, for $\alpha = 200$, there are 145 dense transitions, and the algorithm generates 91 dense paths. Figure 7.a visualizes 91 dense paths on the road network of San Joaquin for $\alpha = 200$, while Figure 7.b shows 239 dense paths when $\alpha = 50$.

Actually, the advantages of the phase 1 of NETSCAN is, on one hand the computation of dense paths that facilitate the decision making, and on other hand an initialization of the clusters by aggregating trajectories around the obtained dense paths. This is achieved in the second phase of NETSCAN.

Table 1. Impact of the density threshold

α	#dense_paths	#transitions $>\alpha$	σ	Min (#traj. /cluster)	Mean (#traj. /cluster)	Max (#traj. /cluster)
10	1303	2231 (5,7%)	0.5	11	74,82	486
50	239	416 (1,06 %)	0.7	38	147,84	261
100	143	252 (0.64 %)	0.7	97	205,01	267
200	91	145 (0.37 %)	0.7	209	233,868	267
230	27	37 (0.09 %)	0.7	239	248,92	267

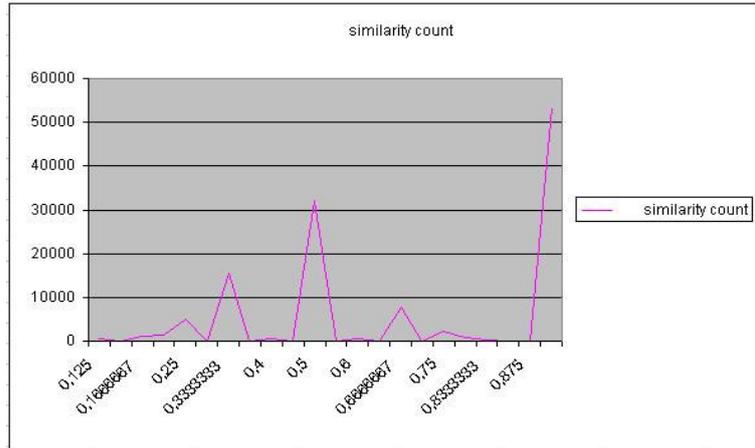
**a-** $\alpha = 200$ **b-** $\alpha = 50$ **Fig. 7.** Mapping Dense Paths

6.3 Experimental Results – Phase 2

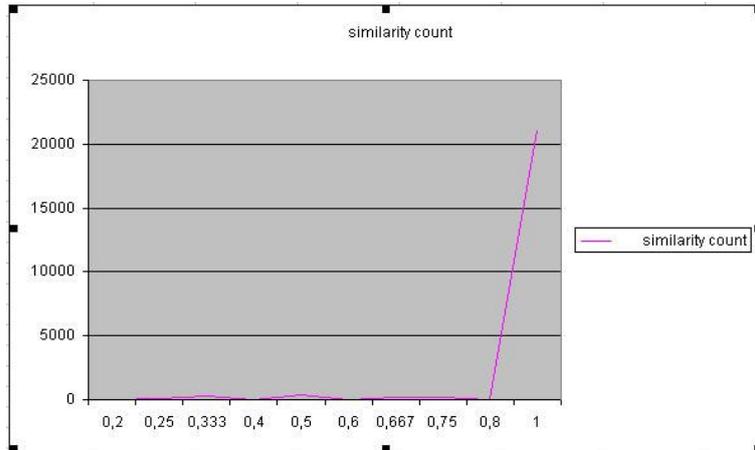
In this test, we evaluate the impact of the similarity threshold σ . When σ is equal to 1, this means that the cluster only groups the trajectories including the whole dense path. The other cases correspond to a degree of similarity. This value mainly impacts the number of trajectories belonging to each cluster. As the similarity threshold increases, the number of trajectory tends to decrease and vice versa. In table 1 above, we measure the minimum, the average, and the maximum number of trajectories by cluster.

In order to tune the similarity threshold σ , we provide the user another distribution curve as in Figure 8 bellow. This last draws the number of trajectories corresponding to each similarity value between trajectories and (intersecting) dense paths. As an example, as shown in Figure 8.a., for $\alpha = 10$,

many trajectories include half to 80% of the dense paths. However, in Figure 8.b. where $\alpha = 200$, most trajectories include the whole dense paths or contain at least 80% of their length, that is no need to choose a similarity threshold under 0.8.



8-a. $\alpha = 10$



8-b. $\alpha = 200$

Fig. 8. Tuning the similarity threshold

6.4 Optimization

This algorithm has been first implemented without any optimization. The algorithm, mainly in Phase 2, performed too slowly (in many hours). In-

deed, the trajectory database was scanned as many times as the number of dense paths. Since symbolic data representation adapts to relational database, it was possible to use conventional query optimisation techniques such as indexing. Doing so, we improved the performances to a quasi real time processing, even in the case of a great volume of data.

Concerning the first phase, the optimisation has concerned the storage costs. In fact, the size of the density matrix is theoretically the square of the number of sections (e.g. 24123^2 here). But this is a very sparse matrix. Therefore, we choose to store it in a table as (i, j, #objects) where only the transitions having non null values (#objects >0) are materialized. This reduces the storage size drastically (39130 transitions in our case).

7. Conclusion

This paper deals with spatiotemporal data mining. More precisely, it addresses the problem of moving object clustering and adapts it to network constrained moving objects. We have proposed a two-step clustering algorithm. The first one focuses on the road sections and allows obtaining the densest paths all over the network. The second step processes the trajectories in order to obtain similar trajectories classes.

The proposed algorithms have been implemented, optimized, and tested using simulated moving objects. The experimental results have allowed a preliminary test in order to validate our approach.

As future work, we aim at applying our approach to real datasets in order to validate it by experts in traffic management. Thereafter, we will study the extension to spatio-temporal clustering. Another promising issue is the combination of spatio-temporal clustering and previous proposals in spatio-temporal OLAP, as proposed by Savary et al. (2004) and Wan and Zeitouni (2006). Finally, we will explore this mining task for moving objects equipped by sensors, such as pollution or temperature sensors. To this end, we may keep our symbolic representation, i.e. by reference to the road segments, and extend it to capture the measure range. This implies the extension of the transition matrix with the temporal dimension and measures. The similarity also should be extended and the algorithms should be adapted and optimized.

Acknowledgements

The authors would like to thank Benjamin Nguyen for his reading and discussions about the ideas of this paper. This work has been partly supported by grants from Région Ile-de-France.

References

- Ankerst M., M. M. Breunig, H.-P. Kriegel and J. Sander (1999), OPTICS: Ordering Points to Identify the Clustering Structure, In Proc. 1999 ACM SIGMOD Int'l Conf. on Management of Data, Philadelphia, Pennsylvania, pp. 46-60.
- Brinkhoff T., A Framework for Generating Network-Based Moving Objects, *GeoInformatica*, Vol. 6, No. 2, Kluwer, 2002, 153-180
- Chang J-W., R. Bista, Y-C. Kim and Y-K Kim (2007) Spatio-temporal Similarity Measure Algorithm for Moving Objects on Spatial Networks. ICCSA 2007, pp.1165-1178.
- Chen L., M.T. Ozsu and V. Oria (2005), Robust and Fast Similarity Search for Moving Object Trajectories. In: ACM SIGMOD, pp. 491-502. ACM Press, New York.
- Du Mouza C. and P. Rigaux. Mobility Patterns. In *Proc. Intl. Workshop on Spatio-temporal Databases (STDBM'04)*.
- Ester M., H.-P. Kriegel, J. Sander and X. Xu (1996) A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise, In Proc. 2nd Int'l Conf. on Knowledge Discovery and Data Mining, Portland, Oregon, pp. 226-231.
- Gaffney S. and P. Smyth, (1999) Trajectory Clustering with Mixtures of Regression Models, In Proc. 5th ACM SIGMOD Int'l Conf. on knowledge Discovery and Data Mining, San Diego, California, pp. 63-72.
- Hadjieleftheriou M., G. Kollios, P. Bakalov, V. Trotras (2005) Complex Spatio-Temporal Pattern Queries. In Proc. of the 31st VLDB Conference.
- Han J. and M. Kamber (2006) *Data Mining: Concepts and Techniques*, 2nd ed., Morgan Kaufmann.
- Hwang J-R., H-Y. Kang and K-J. Li (2005) Spatio-temporal Analysis Between Trajectories on Road Networks. ER Workshops 2005, LNCS 3770, pp. 280-289.
- Lee J-G, J. Han and K-Y. Whang (2007) Trajectory Clustering: A Partition-and-Group Framework. In Proc. SIGMOD'07, Beijing, China.
- Lin B., J. Su (2005) Shapes Based Trajectory Queries for Moving Objects. GIS, pp. 21-30.
- Lloyd S. (1982) Least Squares Quantization in PCM, *IEEE Trans. on Information Theory*, 28(2): 129-137.
- Sakurai Y., M. Yoshikawa and C. Faloutsos (2005) FTW: Fast Similarity Search under the Time Warping Distance. In: PODS, pp. 326-337.
- Savary L., Wan T., Zeitouni K., Spatio-Temporal Data Warehouse Design for Activity Pattern Analysis, DEXA Workshop on Geographic Information Management, September, 2004, Zaragoza, Spain, pp. 814-818.
- Shim C-B and J-W Chang (2003) Similar Sub-Trajectory Retrieval for Moving Objects in Spatiotemporal Databases. In: Proc. of the 7th EECADIS, pp.308-322.
- Tiakas E., A. N. Papadopoulos, A. Nanopoulos and Y. Manolopoulos (2006) Trajectory Similarity Search in Spatial Networks. In : Proc. of the 10th IDEAS, pp. 185-192.

- Vlachos M., D. Gunopulos and G. Kollios (2002) Robust Similarity Measures of Mobile Object Trajectories. In: Proc. of the 13 th Intl. Workshop on DEXA, IEEE Computer Society Press, Los Alamitos pp. 721-728.
- Vlachos M., G. Kollios and D. Gunopulos (2002) Discovering Similar Multidimensional Trajectories. In: Proc. Of the 18th ICDE. IEEE Computer Society Press, Los Alamitos pp. 673-684.
- Wan T. and Zeitouni K., An OLAP System for Network-Constraint Moving Objects, ACM SAC 2007, The 22nd Annual ACM Symposium on Applied Computing, Seoul, Korea, March, 2007, pp. 13-18.
- Yanagisawa Y., J. Akahani, T. Satoh (2003) Shape-Based Similarity Query for Trajectory of Mobile Objects. In : Proc. Of the 4th Intl. Conf. On MDM, pp. 63-77.
- Zeinalipour-Yazti D., S. Song Lin, D. Gunopulos (2006) Distributed Spatio-Temporal Similarity Search. CIKM, pp. 14-23.
- Zhang T., R. Ramakrishnan, and M. Livny (1996) BIRCH: An Efficient Data Clustering Method for Very Large Databases. In Proc. ACM SIGMOD Int'l Conf. on Management of Data, Montreal, Canada, pp. 103-114.