Graph-Based Approaches to Clustering Network-Constrained Trajectory Data

Mohamed K. El Mahrsi[1,2] and Fabrice Rossi[2]

[1] Télécom ParisTech, Département INFRES
46, rue Barrault 75634 Paris CEDEX 13, France
khalil.mahrsi@telecom-paristech.fr
[2] Équipe SAMM EA 4543, Université Paris I Panthéon-Sorbonne
90, rue de Tolbiac 75634 Paris CEDEX 13, France
fabrice.rossi@univ-paris1.fr

Abstract. Clustering trajectory data attracted considerable attention in the last few years. Most of prior work assumed that moving objects can move freely in an euclidean space and did not consider the eventual presence of an underlying road network and its influence on evaluating the similarity between trajectories. In this paper, we present an approach to clustering such network-constrained trajectory data. More precisely we aim at discovering groups of road segments that are often travelled by the same trajectories. To achieve this end, we model the interactions between segments w.r.t. their similarity as a weighted graph to which we apply a community detection algorithm to discover meaningful clusters. We showcase our proposition through experimental results obtained on synthetic datasets.

Keywords: similarity, clustering, moving objects, trajectories, road network, graph.

1 Introduction

Traffic congestion has become a major problem that affects many human activities on a daily basis, resulting in both serious transportation delays and environmental damages. Monitoring the state of the road network is commonly conducted by using dedicated sensors that register the number of vehicles passing by the section where they are installed. The prohibitive cost of deploying and maintaining such sensors limits their deployment to the highways and the road network's main arteries. Subsequently, the collected data portray a partial and incomplete state of the road network, thus complicating data mining tasks that aim at extracting useful knowledge about flow dynamics and the behavior of drivers moving along the network.

An alternative (or complementary) approach to addressing these shortcomings may consist in analyzing GPS logs collected using location-aware devices (e.g. classic GPS, smartphones, PDAs, etc.). These logs can be acquired through probing vehicles, dedicated data acquisition campaigns (using buses, taxis or an enterprise's fleet of vehicles) or even by means of a crowdsourced approach where different individuals willingly contribute by uploading their different commute logs. Therefore, it is perfectly feasible to collect large amounts of trajectory data that can be stored in dedicated databases (known as Moving Object Databases [1]). These data offer a better coverage of the road network and can be, later on, explored using data mining and statistical learning techniques.

Clustering is one of such techniques. Prior work on trajectory data clustering focused mainly on the case where moving objects move freely in a euclidean space [2–5]. By doing so, these approaches did not account for the presence, in the case of car trajectories as well as in other cases, of an underlying network that constrains the movement. The network's constraints, however, do play a paramount role in determining the similarity between the trajectories to be clustered. Moreover, the majority of these approaches relied on the use of density-based clustering which makes them vulnerable to the way the parameters of the clustering algorithm are selected.

In [6], we presented a framework for clustering network-constrained trajectories using a graph-based approach. This framework was directed towards discovering groups of similar trajectories that moved along the same parts of the road network. The hierarchical, non-parametric algorithm that we used in the clustering step made our framework flexible and suitable for exploring the discovered groups of trajectories at various levels of detail: the user can start with a limited number of high-level, coarse clusters and delve (by means of consecutive zooming) in the refinement of the clusters he deems interesting.

The work presented in this paper builds upon the one undertaken in [6]. We extend our framework to the case of road segments as we try to discover relevant groups of segments that are commonly used and explored together by a considerable number of trajectories. Our contributions can be summarized as follows:

- We define a similarity measure that evaluates the resemblance between pairs of road segments based on the trajectories that travelled along both of them;
- We use a graph representation to model interactions between the different road segments. The resulting similarity graph is partitioned using modularity-based community detection in order to discover a hierarchy of nested clusters of road segments;
- We test our proposition on synthetic datasets and showcase how it can be used, in association with the technique we presented in [6], for understanding and characterizing the traffic in the road network.

The rest of this paper is organized as follows. In Section 2, we present the network-constrained trajectories data model and we formalize our segment clustering problem. Our segment clustering approach is described in detail in Section 3. Section 4 discusses the computational complexity of our proposition as well as how the discovered clusters can be interpreted and used, complementarily with the trajectory clusters presented in [6], in order to discover useful knowledge about the flow dynamics and traffic in the road network. Experimental results are presented in Section 5. Related work is discussed in Section 6. Finally, Section 7 concludes the paper.

2 Data Representation and Problem Statement

We opt for the symbolic data representation which is the model of choice adopted for representing network-constrained trajectories in most of prior work [7–10]. In this model, the road network is modeled as a graph, defined as follows.

Definition 1 (Road Network). *The road network is represented as a directed graph $\mathcal{G} = (\mathcal{V}, \mathcal{S})$. The set of vertices $\mathcal{V}$ represents intersections and terminal points of roads whereas the set of directed edges $\mathcal{S}$ represents the road segments interconnecting them. A directed edge $s = (v_i, v_j)$ indicates that a road segment links the two nodes v_i and v_j and that it can be traveled from v_i in the direction of v_j but not the other way around (unless another edge states otherwise).*

Given this graph representation, moving objects (i.e. vehicles) moving along the road network produce trajectories that can be modeled conformably to the following definition.

Definition 2 (Constrained Trajectory). *A constrained trajectory T that travels along the road network $\mathcal{G}$ can be modeled as a sequence of visited segments:*

$$T = \langle id, \{s_1, s_2, ..., s_l\} \rangle$$

id being the identifier of the trajectory, l its length (i.e. number of segments) and $\forall 1 \leq i < l, s_i$ and s_{i+1} are connected segments belonging to $\mathcal{S}$.

In a real-case scenario, trajectories are collected as GPS logs (sequences of latitude and longitude points) on which a map matching technique (e.g. [7, 9]) is applied in order to produce the sequence of traveled segments. The map matching step is out of the scope of this paper. Hence, we suppose that the trajectories are already and correctly map matched to the corresponding road segments.

Finally, we formalize the road segment clustering problem that we study in this paper as follows.

Definition 3 (Road Segment Clustering Problem). *Given a road network represented by a graph $\mathcal{G} = (\mathcal{V}, \mathcal{S})$ and a set of trajectories $\mathcal{T} = \{T_1, T_2, ..., T_n\}$ that traveled along it, road segment clustering aims to partition the set of road segments $\mathcal{S}$ into a set of disjoint clusters $\mathcal{C_S} = \{C_1, C_2, ..., C_K\}$ in such a fashion that:*

- *Segments grouped in the same cluster C_i are visited by a considerable amount of common trajectories (i.e. a trajectory T that visits a segment $s \in C_i$ also visits a fair amount of segments in this same cluster);*
- *Segments belonging to two different clusters C_i and C_j are visited by as few common trajectories as possible (i.e. they are unlikely to be part of a same trajectory).*

3 A Graph-Based Approach to Road Segment Clustering

We now present our solution to the road segment clustering problem introduced in the previous section. First, we define a similarity measure between road segments based on the comparison of the common trajectories that visited them (Section 3.1). Based on this measure, we build a graph depicting the relationships between different road segments (Section 3.2). The graph is then partitioned using a modularity-based community detection algorithm in order to discover a hierarchy of nested segment clusters (Section 3.3).

3.1 Road Segment Similarity

Similarly to the bag-of-words model (where a text is considered as an unordered collection of words), we consider each road segment as a bag-of-trajectories that visited it (i.e. $\forall s \in \mathcal{S}, s \equiv \{T \in \mathcal{T} : s \in T\}$).

In order to compare two road segments s_i and s_j, one can simply observe how often they co-appear in trajectories (i.e. calculate $|\{T \in \mathcal{T} : s_i \in T \wedge s_j \in T\}|$). The larger the number of concomitant appearances of both segments is, the more they are considered similar. However, different trajectories do not hold the same discriminative power when it comes to characterizing the similarity between road segments they visit: a lengthy trajectory that travels along a considerable number of road segments is not very informative when judging the similarity between two segments in particular and, vice versa, short trajectories are highly relevant to the formation of the cluster that contains the segments they visit.

We account for this observation by devising a tfidf-like weighting strategy where the contribution of each trajectory is proportional to its length. The weight $\omega_{T,s}$ assigned to trajectory T while inspecting a road segment s is expressed in formula (1):

$$\omega_{T,s} = \frac{n_{s,T}}{\sum_{T' \in \mathcal{T}} n_{s,T'}} \cdot \log \frac{|\mathcal{S}|}{|s \in \mathcal{S} : s \in T|} \tag{1}$$

The first part in this weight calculates the contribution of T to the segment s by calculating the ratio between the number of appearances $n_{s,T}$ of s in T and the total number of appearances of s in the whole dataset of trajectories $\mathcal{T}$. Since multiple visits of a same road segment are very rare, this part is often equal to $\frac{1}{|\{T \in \mathcal{T} : s \in T\}|}$. The second part evaluates the importance of the trajectory across the whole set of road segments : the more segments a trajectory visits the less important it becomes and vice versa.

We use a cosine similarity to measure the similarity between two road segments e_i and e_j as expressed in formula (2):

$$\text{Similarity}(s_i, s_j) = \frac{\sum_{T \in \mathcal{T}} \omega_{T,s_i} \cdot \omega_{T,s_j}}{\sqrt{\sum_{T \in \mathcal{T}} \omega_{T,s_i}^2} \cdot \sqrt{\sum_{T \in \mathcal{T}} \omega_{T,s_j}^2}} \tag{2}$$

3.2 Road Segment Similarity Graph

We model the similarity relationships between road segments using an undirected, weighted graph $\mathcal{SG}_{\mathcal{S}} = (\mathcal{S}, \mathcal{E}, \mathcal{W})$. Each road segment in $\mathcal{S}$ is mapped to a vertice in $\mathcal{SG}_{\mathcal{S}}$. An edge between a pair of segments s_i and s_j exists if and only if Similarity$(s_i, s_j) > 0$ (i.e. if there is at least one common trajectory that crossed both segments). In which case the similarity is assigned as a weight to that edge. This concept of similarity graph is depicted in Fig. 1.

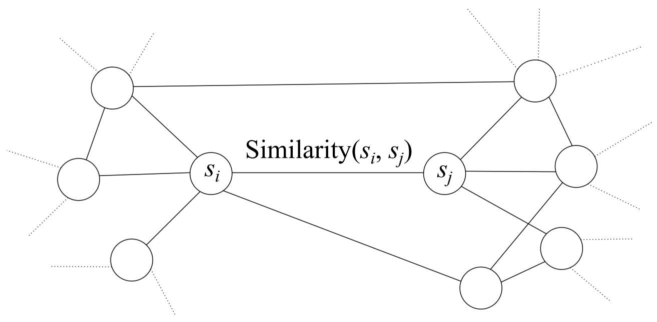


Fig. 1. Excerpt from a segment similarity graph. Vertices represent the studied road segments while weighted edges indicate the presence and strength of the similarity between pairs of segments.

The main advantage of using this graph representation, besides being natural and easy to understand, is that it does not invent an "artificial" similarity between totally incompatible road segments. On the contrary, it emphasizes on the fact that road segments that do not share common trajectories are independent and should, therefore, not be "immediately" grouped in the same cluster since there is no similarity edge linking them.

3.3 Clustering the Similarity Graph

Road networks are complex and contain a considerable amount of segments, resulting, therefore, in a large similarity graph. Moreover, since one common trajectory is sufficient for a similarity edge to exist between a pair of segments, the vertices of the similarity graph tend to have high degrees (although, from our observations, this degree distribution does not follow a proper power law). Modularity-based community-detection algorithms are a popular and widely adopted choice to clustering such graphs [11].

Given a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathcal{W})$, with vertices $\mathcal{V} = \{v_1, v_2, ..., v_n\}$, weighted edges $\mathcal{E}$ such as $\omega_{ij} \geq 0$ and $\omega_{ij} = \omega_{ji}$, and given a partition of the vertices into K clusters (or communities) $C_1, ..., C_K$, the modularity of the partition is expressed according to formula (3):

$$\mathcal{Q} = \frac{1}{2m} \sum_{k=1}^{K} \sum_{i,j \in C_k} \left(\omega_{ij} - \frac{d_i d_j}{2m} \right) \tag{3}$$

$d_i = \sum_{j \neq i} \omega_{ij}$ and $m = \frac{1}{2} \sum_i d_i$. The modularity measures the quality of the clustering by inspecting the arrangement of the edges within the communities of vertices. A high modularity is an indicator that the edges within the communities outnumber (or have higher weights than) those in a similar randomly generated graph (i.e. one that does not present a community structure). Communities discovered using modularity optimization have a structure that is similar to the structure of cliques. In our context of segment clustering, this means that segments grouped together are heavily connected (which is the intended result) and are travelled by a considerable number of shared trajectories.

To cluster the segment similarity graph, we use the implementation of hierarchical modularity-based clustering described in [12]. The pseudo-code is given in Algorithm 1. First, the algorithm retrieves a partition of the vertices with optimal modularity (line 1): the *Partition* procedure start by considering the trivial partition where each vertex is in its own community and merges communities in a greedy fashion (i.e. each time, it merges the two communities that produce the maximum increase of modularity). The merging operation stops when no possible merge can be done without a degradation of the modularity. In which case the *Partition* procedure proceeds to a refinement step where members of different communities are interchanged in an attempt to further improve the modularity of the partition.

Algorithm 1 Hierarchical modularity-based clustering.

```
Input: an undirected, weighted graph $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathcal{W})$
Output: hierarchy of nested clusters of vertices
 1: $C_1^{(1)}, ...C_K^{(1)} \leftarrow Partition(\mathcal{G})$          ▷ initial partition
 2: $K_T \leftarrow K$                                                    ▷ clusters counter
 3: $l \leftarrow 1$                                                      ▷ hierarchy level
 4: repeat
 5:     $l \leftarrow l + 1$
 6:     for all cluster $C \in C_1^{(l-1)}, ..., C_K^{(l-1)}$ do
 7:         extract the sub-graph $\mathcal{G}_C$ of vertices belonging to $C$
 8:         $C_1^C, ...C_k^C \leftarrow Partition(\mathcal{G}_C)$
 9:         if $TestSig(C_1^C, ...C_k^C)$ then
10:             $C_{K_T+1}^{(l)}, ..., C_{K_T+k}^{(l)} \leftarrow C_1^C, ...C_k^C$
11:             $K_T \leftarrow K_T + k$
12:         end if
13:     end for
14: until no significant subdivision of level $l$ can be found
```

Once the initial partition is retrieved, the algorithm proceeds iteratively to construct the hierarchy of communities (lines 4 through 14). For each community at a given level, the sub-graph containing only the vertices of the community and the edges connecting them is isolated (line 7). This subgraph is partitioned separately (by invoking *Partition* as shown in line 8). The *TestSig* evaluates the significance of the found partition (by comparing its modularity to the modularity of partitions obtained on similar randomly generated graphs). If the partition is significant indeed, its communities are considered for partitioning in the next iteration (lines 9-12), otherwise it's rejected and the original community is retained. The iterations stop when none of the communities at level l yield a significant partition (line 14).

Modularity-based graph clustering approaches are very popular and achieve good results in practice [11]. Nevertheless, we do not exclude the use of other graph clustering alternatives (e.g. spectral clustering [13]) if such techniques can yield better results.

4 Discussion

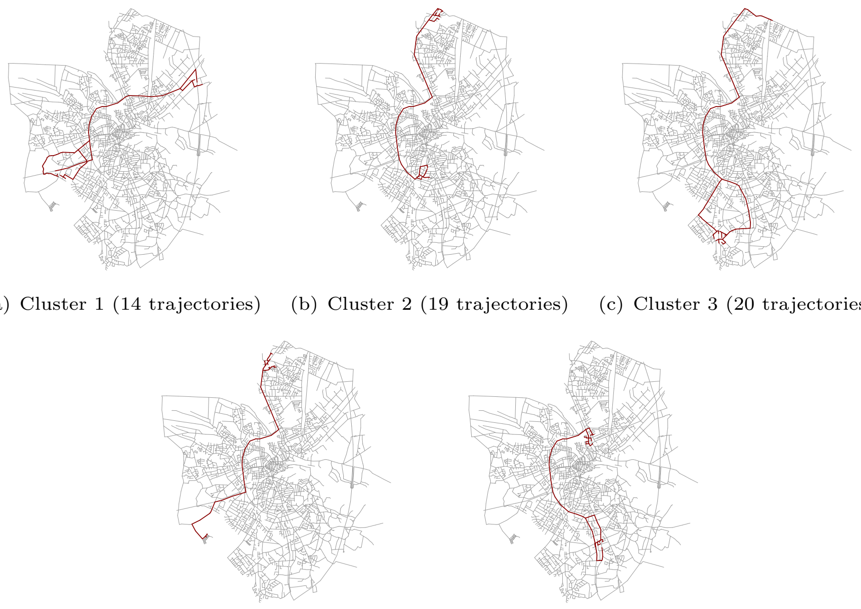
First, we focus on how the produced clusters can be explored and analyzed in order to deduce useful knowledge about the flow dynamics and the drivers' behavior in the road network (Section 4.1). Then, we address the algorithmic complexity of our approach (Section 4.2).

4.1 Cluster Exploration

We illustrate how the trajectory clusters [6] and segment clusters can be explored and used conjointly. For illustration purposes, we use a synthetic dataset containing 85 trajectories that moved along the Oldenburg road network (cf. Section 5 for more details about this network) and visited a total of 485 distinct road segments. We manually partitioned the trajectories into five clusters (depicted in Figure 2) that we consider hereafter as the ground-truth clusters.

Applying the trajectory clustering [6] results in a hierarchy of clusters where the optimal level w.r.t. modularity (i.e. the very first level) contains only three trajectory clusters: the ground-truth clusters 2 and 3 are considered as part of a same cluster (the same occurs with clusters 4 and 5). Nevertheless, all the ground-truth clusters are retrieved correctly (some of them are even refined) in the following levels. The cluster hierarchy is especially suitable for exploring large datasets where a flat clustering can still produce a high number of clusters: the analyst can start with the few, coarse clusters contained in the first hierarchical levels in order to gain a quick grasp of the general tendencies and movement patterns in the road network. He, then, can choose clusters of interest that he can explore, by means of successive zooms, in higher detail. This idea is depicted in Figure 3 which shows a coarse trajectory cluster and its three, more refined subclusters.

Segment clusters are not as easy to grasp and understand as trajectory clusters. Even though it is feasible to try and explore these clusters as stand-alone



(a) Cluster 1 (14 trajectories) (b) Cluster 2 (19 trajectories) (c) Cluster 3 (20 trajectories)

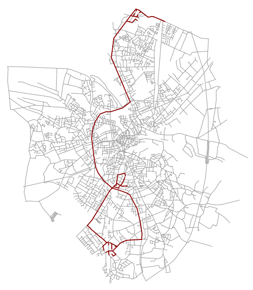
(d) Cluster 4 (20 trajectories) (e) Cluster 5 (12 trajectories)

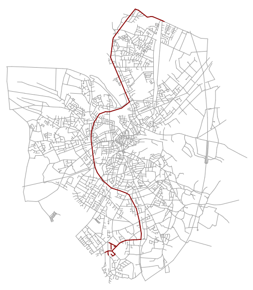
Fig. 2. Ground-truth clusters in the dataset.

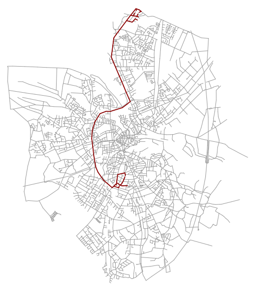
clusters, we recommend involving the trajectory clusters in the process. Cross-comparing both types of clusters can reveal interesting information about flow dynamics and yield a better interpretation of the clusters. For example, a segment cluster can be interpreted based solely on the trajectory groups that interacted with it, thus revealing potential hubs, etc. Fig. 4 shows the crossed matrix of the second level trajectory clusters (reported on the rows) and the second level road segment clusters (on the columns) and gives an idea about the sizes of the clusters and how clusters of one type interact with those of the other type.

The crossed matrix does indeed reveal some interesting patterns and interactions. For instance, the fourth segments clusters is explored exclusively by two trajectory clusters. Visualizing both this segment cluster and its visiting trajectory clusters (Fig. 5) shows that the segment cluster plays the role of a hub for these two groups of trajectories that converge to it from two different areas in order to travel to two different destinations.

Crossing trajectory clusters and segment clusters is flexible and can be done at various levels of the hierarchies of both cluster types. However, it is totally up to the user to decide the relevance of the crossed clusters. The case of the eleventh segment cluster (cf. Fig. 4) illustrates this point: this segment cluster is very interesting since it interacts with six trajectory clusters. However, it is evident that the segment cluster contains a lot of "noise" segments which is expressed by the considerable amount of white space in the six first rows (rep-


(a) Parent cluster (39 trajectories)


(b) Subcluster 1 (12 trajectories)


(c) Subcluster 2 (19 trajectories)

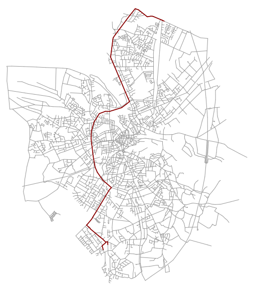

(d) Subcluster 3 (8 trajectories)

Fig. 3. A coarse cluster containing 39 trajectories (a) and its more detailed subclusters (b-d).

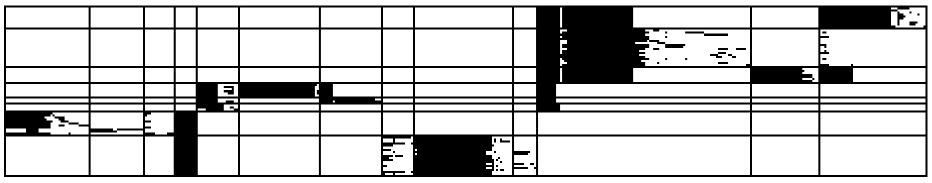


Fig. 4. Crossed matrix of the trajectory clusters (rows) and road segment clusters (columns). Each cell gives an idea about the interaction between the corresponding trajectory and segment clusters: the more black the cell contains the more trajectories in the trajectory cluster cross segments belonging to the segment cluster.

resenting the trajectory clusters) in the column representing the cluster in the crossed matrix. Consequently, drawn conclusions about the interactions between the clusters won't be very reliable. A wiser alternative would be to study the interactions between the, more refined, subclusters of this segment cluster with the six trajectory cluster it interacts with.

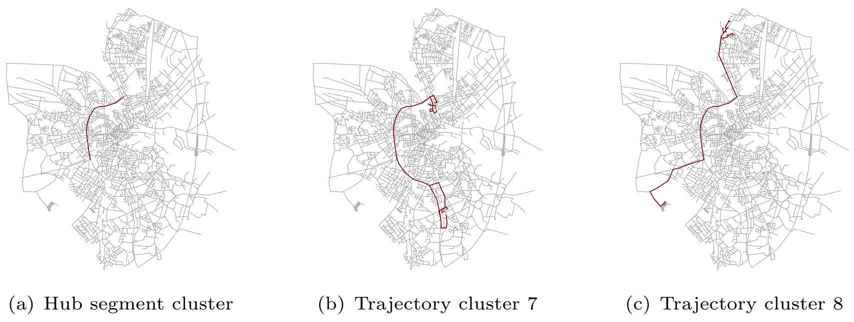

(a) Hub segment cluster (b) Trajectory cluster 7 (c) Trajectory cluster 8

Fig. 5. A segment cluster (a) playing the role of a hub for two different trajectory clusters ((b) and (c)) that borrow it to travel to two separate destinations.

4.2 Algorithmic Complexity

Let n be the number of trajectories in $\mathcal{T}$ and m the number of road segments in $\mathcal{S}$. Road segments can be represented as a matrix M containing m rows (each representing a road segment) and n columns (each corresponding to a trajectory). $m_{i,j}$ corresponds to the weight of the trajectory represented by column j while inspecting the segment represented by row i. Using this vector model representation, comparing two road segments can then be done in $O(n)$ time complexity. Constructing the similarity graph requires $\frac{m(m-1)}{2}$ similarity calculations. Therefore, the cost of constructing the graph is $O(nm^2)$.

The similarity graph contains m vertices (representing the m segments of $\mathcal{S}$) and, at most, $\frac{m(m-1)}{2}$ edges. Therefore, the theoretical (maximal) complexity of the community detection algorithm used in our clustering phase is $O(m^3)$ [11]. However, this complexity is rarely observed in practice where the complexity is somewhere near $O(m^2)$.

The complexity of the approach we presented in [6] can be deduced using the same reasoning: the trajectory similarity graph is constructed in $O(mn^2)$ and is clustered in $O(n^3)$ in theory ($O(n^2)$ in practice).

5 Experimental Results

In this section, we validate the effectiveness of our approach by comparing it to two alternative graph clustering techniques. First, we describe our experimental setting, including the used datasets and the evaluated algorithms in Section 5.1. Then cluster quality results are presented in Section 5.2.

5.1 Experimental Setting

In order to validate our choice of modularity-based clustering, we compare it to two other graph clustering techniques: i. spectral clustering; and ii. label propagation clustering. In spectral clustering [13], eigenvectors are extracted from the graph's Laplacian and are used to conduct a k-means clustering in order to partition the graph's vertices. Label propagation, on the other hand, works by labeling the vertices with unique labels and then updating the labels by majority voting in the neighborhood of the vertex [14].

We compare the performances of the three algorithms on five synthetic datasets (cf. Table 1) produced with the Brinkhoff generator [15] using the Oldenburg road network. The latter is composed of 6105 vertices and about 14070 road segments. Each dataset contains 100 trajectories visiting a various amount of road segments.

Table 1. Characteristics of the five synthetic datasets.

Dataset	Number of segments	Number of edges in the similarity graph
1	2562	79811
2	2394	100270
3	2587	110095
4	2477	87023
5	2348	80659

The performance of each algorithm is evaluated by measuring the quality of the segment partition $\mathcal{C_S}$ it produces according to formula (4):

$$\mathcal{Q}(\mathcal{C_S}) = \sum_{C \in \mathcal{C}} \frac{1}{|C|} \sum_{s_i, s_j \in C} \frac{|\{T \in \mathcal{T} : s_i \in T \wedge s_j \in T\}|}{|\{T \in \mathcal{T} : s_i \in T \vee s_j \in T\}|} \tag{4}$$

$|C|$ is the number of segments in clusters C, $|\{T \in \mathcal{T} : s_i \in T \wedge s_j \in T\}|$ is the number of trajectories both road segments s_i and s_j while $|\{T \in \mathcal{T} : s_i \in T \vee s_j \in T\}|$ is the number of trajectories that travelled along at least one of them.

5.2 Results

Contrary to the spectral clustering algorithm, the modularity-based and label propagation algorithms do not give the user the possibility to configure the number of resulting clusters: the label propagation algorithm produces just one flat partition while the modularity-based algorithm produces a partial hierarchy (i.e. a hierarchy that does not retain all the merging operations).

First, we compare modularity-based clustering and spectral clustering based on the former's optimal number of clusters (i.e. the number of clusters at the hierarchy's first level). The results are depicted in Table 2.

In order to compare the three algorithms at once (cf. Table 3), we proceed as follows. Since label propagation clustering produces only on partition, we

Table 2. Comparison between spectral clustering and modularity-based clustering.

	Clustering quality (clusters)	
Dataset	Spectral	Modularity
1	306.33 (23)	**657.20** (23)
2	254.97 (21)	**524.46** (21)
3	245.64(20)	**561.08** (20)
4	249.89 (22)	**594.75** (22)
5	284.74 (26)	**666.23** (26)

configure the spectral clustering to produce the same number of clusters as this partition. As for modularity-based clustering, we choose the hierarchical level that produces the closest number of clusters to those discovered by label propagation.

Table 3. Cluster qualities achieved by the three algorithms on the five datasets.

	Clustering quality (clusters)		
Dataset	Label prop.	Spectral	Modularity
1	684.19 (68)	678.81 (68)	**1614.40** (67)
2	550.66 (59)	549.70 (59)	**1276.63** (57)
3	606.45 (66)	567.57 (66)	**1516.45** (61)
4	634.63 (68)	637.62 (68)	**1406.38** (57)
5	604.97 (64)	539.27 (64)	**1418.67** (65)

From both Table 2 and Table 3 we can verify that, as expected, the clustering quality increases as the number of clusters increases. Results also show the superiority of modularity-based clustering over label propagation and spectral clustering and suggest that applying the former results in better and more compact clusters of road segments.

We also notice that label propagation results in a large number of clusters. This supports the observation we made in Section 4.1 where we claimed that flat clustering is not suitable for exploring large datasets.

6 Related Work

Approaches to trajectory clustering are mainly adaptations of existing algorithms to the case of trajectories. Existing problem formulations and propositions include flock patterns [3], convoy patterns [5], the TRACLUS partition-and-group framework [4] and the T-OPTICS and TF-OPTICS algorithms [2]. The aforementioned algorithms use euclidean-based similarities and distances and can, therefore, be used only in the case of unconstrained trajectories. Furthermore, the majority of these approaches use density-based algorithms which suffer from two major drawbacks: i. their results are very sensitive to the parameter values; and ii. they assume that trajectories in the same cluster have a rather homogeneous density, which is rarely the case (as discussed in [10]).

Roh et Hwang [10] present a network-aware approach to clustering trajectories where the distance between trajectories in the road network is measured using shortest path calculations. A baseline algorithm, using agglomerative hierarchical clustering, as well as a more efficient algorithm, called NNCluster, are presented for the purpose of regrouping the network constrained trajectories. In [8], the authors describe an approach to discovering "dense paths" or sequences of frequently traveled segments in a road network. This approach resembles our segment-based clustering although they diverge on many key aspects. For instance, the approach in [8] produces flat clusters using a density-based approach (which requires fine tuning) whereas ours produces a hierarchy of nested clusters and does not require parametrization. In [6], we presented our graph-based framework to clustering network-constrained trajectories. The work described in the present paper build upon this framework as it extends it to the case of road segment clustering.

A wide variety of graph clustering algorithms was proposed in the literature, including spectral clustering [13], clustering using label propagation [14], etc. (complete surveys on graph clustering can be found in [16, 11]). Among these propositions, modularity-based community detection algorithms stand out for the good results they yield in practice. We use the hierarchical modularity-based clustering implementation described in [12] (which follows the recommendations in [17]) in our clustering step of our framework in order to detect the presence of clusters among road segments.

7 Conclusion

In this paper, we presented a framework for clustering road segments based on the moving object trajectories that travelled along them. The main novelty of the framework is the use of a graph representation to structure the similarity relationships and interactions between road segments. This framework presents many advantages: i. it does not require parameters, contrary to the majority of existing approaches that are very sensitive to their threshold values; and ii. it also produces a hierarchy of nested clusters promoting exploration at various levels of granularity and detail in situations where a flat clustering approach would have produced a unique level containing a very large number of clusters. Moreover, we showed how segment clusters can be used in conjunction with the trajectory clusters we defined in [6] in order to better understand flow dynamics in the road network.

The framework, however, is not flawless. The community detection algorithm used in the clustering step can be sensitive in presence of noise (i.e. marginal road segments that do not forcefully belong to any cluster) which can degrade the quality of the discovered clusters. Also, the computational cost of the approach and the fact that it requires predisposing of all the data beforehand prohibits it from being used in a streaming context.

In future work, we will focus on alternative graph representations for trajectory data. Mainly, the use of a bipartite graph to represent interactions between trajectories and segments. Such graphs can be partitioned using bi-clustering algorithms in order to simultaneously discover clusters of trajectories and road segments (this is done separately in the present framework) which has the main advantage of automatically crossing both types of clusters based on how they interact, thus relieving the user from this delicate task.

References

1. Giannotti, F., Pedreschi, D., eds.: Mobility, Data Mining and Privacy - Geographic Knowledge Discovery. Springer (2008)
2. Nanni, M., Pedreschi, D.: Time-focused clustering of trajectories of moving objects. J. Intell. Inf. Syst. **27**(3) (2006) 267–289
3. Benkert, M., Gudmundsson, J., Hübner, F., Wolle, T.: Reporting flock patterns. In: ESA'06: Proceedings of the 14th conference on Annual European Symposium, London, UK, Springer-Verlag (2006) 660–671
4. Lee, J.G., Han, J., Whang, K.Y.: Trajectory clustering: a partition-and-group framework. In: SIGMOD '07: Proceedings of the 2007 ACM SIGMOD international conference on Management of data, New York, NY, USA, ACM (2007) 593–604
5. Jeung, H., Shen, H.T., Zhou, X.: Convoy queries in spatio-temporal databases. In: ICDE '08: Proceedings of the 2008 IEEE 24th International Conference on Data Engineering, Washington, DC, USA, IEEE Computer Society (2008) 1457–1459
6. El Mahrsi, M.K., Rossi, F.: Modularity-Based Clustering for Network-Constrained Trajectories. In: Proceedings of the 20-th European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning (ESANN 2012), Bruges, Belgique (April 2012) 471–476
7. Brakatsoulas, S., Pfoser, D., Salas, R., Wenk, C.: On map-matching vehicle tracking data. In: Proceedings of the 31st international conference on Very large data bases. VLDB '05, VLDB Endowment (2005) 853–864
8. Kharrat, A., Popa, I.S., Zeitouni, K., Faiz, S.: Clustering algorithm for network constraint trajectories. In: SDH. Lecture Notes in Geoinformation and Cartography, Springer (2008) 631–647
9. Lou, Y., Zhang, C., Zheng, Y., Xie, X., Wang, W., Huang, Y.: Map-matching for low-sampling-rate gps trajectories. In: Proceedings of the 17th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems. GIS '09, New York, NY, USA, ACM (2009) 352–361
10. Roh, G.P., Hwang, S.w.: Nncluster: An efficient clustering algorithm for road network trajectories. In: Database Systems for Advanced Applications. Volume 5982 of Lecture Notes in Computer Science. Springer Berlin - Heidelberg (2010) 47–61
11. Fortunato, S.: Community detection in graphs. Physics Reports **486**(3-5) (2010) 75–174
12. Rossi, F., Villa-Vialaneix, N.: Représentation d'un grand réseau à partir d'une classification hiérarchique de ses sommets. Journal de la Société Française de Statistique **152**(3) (2011) 34–65

13. Luxburg, U.: A tutorial on spectral clustering. Statistics and Computing **17**(4) (December 2007) 395–416
14. Raghavan, U.N., Albert, R., Kumara, S.: Near linear time algorithm to detect community structures in large-scale networks. Physical Review E **76**(3) (September 2007)
15. Brinkhoff, T.: A framework for generating network-based moving objects. Geoinformatica **6** (June 2002) 153–180
16. Schaeffer, S.E.: Graph clustering. Computer Science Review **1**(1) (2007) 27 – 64
17. Noack, A., Rotta, R.: Multi-level algorithms for modularity clustering. In: Proceedings of the 8th International Symposium on Experimental Algorithms. SEA '09, Berlin, Heidelberg, Springer-Verlag (2009) 257–268