Incremental Clustering of Mobile Objects

Sigal Elnekave, Mark Last, Oded Maimon
Ben-Gurion University *Tel-Aviv University*
elnekave@bgu.ac.il, mlast@bgu.ac.il, maimon@eng.tau.ac.il

Abstract

Moving objects are becoming increasingly attractive to the data mining community due to continuous advances in technologies like GPS, mobile computers, and wireless communication devices. Mining spatio-temporal data can benefit many different functions: marketing team managers for identifying the right customers at the right time, cellular companies for optimizing the resources allocation, web site administrators for data allocation matters, animal migration researchers for understanding migration patterns, and meteorology experts for weather forecasting. In this research we use a compact representation of a mobile trajectory and define a new similarity measure between trajectories. We also propose an incremental clustering algorithm for finding evolving groups of similar mobile objects in spatio-temporal data. The algorithm is evaluated empirically by the quality of object clusters (using Dunn and Rand indexes), memory space efficiency, execution times, and scalability (run time vs. number of objects).

1. Introduction

The discovery of patterns in spatio-temporal data can greatly influence different fields like animal migration analysis, weather forecasting, and consumer habits discovery. Clustering spatio-temporal data can also help in social groups' discovery, which is used in tasks like shared data allocation, targeted advertising, and personalization of content and services. As technology progress, more data is available on the location of moving objects at different times, either by GPS technologies, mobile computer logs, or mobile communication devices. This creates an appropriate basis for developing efficient new methods for mining moving objects.

Previous work on mining spatio-temporal data includes querying data using special indexes for efficient performance, recognizing trajectory patterns (usually represented by a set of equations), and clustering trajectories of moving objects as 'moving clusters'. More extensive research has been done on spatial data and temporal data separately, including periodicity that has been studied only with time series databases. But the field of spatio-temporal mining is relatively young, and requires much more research.

The goal of this work is to incrementally cluster objects' trajectories in order to recognize evolving groups of moving objects. In order to achieve our goal we first generate a spatio-temporal dataset that will be used as a benchmark for testing and evaluating the proposed algorithm. Due to the sensitivity of this kind of data, it will not be possible to obtain a real-world dataset for research purposes. Next, as a preprocessing stage, we define a compact representation of a spatio-temporal trajectory. Then we define a similarity measure between trajectories for discovering object groups according to proximity in time and space. The measure will allow the discovery of groups that are close in time and space. Next we develop a spatio-temporal version of an incremental algorithm for clustering objects according to their trajectories, using the similarity measure defined earlier for discovering similar trajectories. Finally we evaluate the proposed algorithm by conducting experiments on a synthetic dataset. We cluster trajectories by the proposed algorithm with and without incremental learning in order to compare the performance in terms of the clustering quality (Dunn and Rand indexes) and execution times.

2. Related Work

2.1. Spatio-temporal data summarization

Goh and Taniar [4] propose a model for handling the problem of process lag times, which increase with the volume of data as mobile applications usage increases. The proposed model performs minor data analysis and summary at the mobile node level before the raw data is processed by the data mining machine. By the time the mobile object data arrives to the data mining machine, it will be in the form of summary transactions, which reduce the amount of further processing required in order to perform data mining by reducing the number of mined transactions. The cost efficient model lets the mobile nodes summarize their transactions and only then to be sent to the server for data mining.

Performance evaluation shows that this model is significantly more efficient than the traditional model to perform mobile data mining. Yet, in this model the summarization is per mobile node, implying that only times are summarized, but there is no summarization of locations for multiple nodes.

In Anagnostopoulos et al. [1] the authors motivate the need for global distance oriented segmentation techniques. Segmentation leads to simplification of the original objects into smaller, less complex primitives that are better suited for storage and retrieval purposes. Different flavors of distance-based segmentation algorithms are presented that operate at various scales of computational granularities. A variance-based hybrid variation is presented that can provide an excellent compromise between running time and approximation quality.

2.2. Spatio-temporal group patterns mining

The problem of mining group patterns by deriving grouping information of mobile device users based on the spatio-temporal distances among them is approached by Wang, Lim, and Hwang [12] This paper defines a valid segment as a set of consecutive time points [t,t+k] that group users which are not farther than a maximal threshold distance from each other during the time of the interval, and some of the users are farther than this at time t-1. Two algorithms are presented for this mission: AGP (Apriori-like algorithm for mining valid group patterns) and VG-growth (valid group graph data structures, built from a set of valid 2-groups).

Evaluation on a synthetic dataset shows that VG growth outperforms AGP, especially when the minimum weight (used for validating a pattern) becomes smaller. Both algorithms spend most time on searching for valid groups of two elements, which is the main cost of mining here.

The problem of deriving group information on mobile devices based on the trajectory model is approached in Hwang et al. [5]. The trajectory model was chosen for storage space savings and coping with untracked location data. A trajectory T is a set of linear functions, each of which maps from a disjoint time interval to an n-dimensional space. The problem is to find all valid mobile groups given minimal weight and duration thresholds, maximal distance threshold, and trajectory-based dataset.

A valid mobile group exists when the weight of a mobile group exceeds a minimum weight threshold.

In order to find all valid mobile groups under such a model the paper presents two algorithms. Performance evaluation comparing the two algorithms shows that TVG growth (based on VG-growth) is better than TAGP (based on AGP) according to their execution times.

The problem of mining trajectory patterns from a set of imprecise trajectories is studied in Yang J., Hu M. [13]. It uses a novel measure to represent the importance of a trajectory pattern. The authors define the concept of a pattern group to represent the trajectory patterns in a concise manner. Since the Apriori property no longer holds on the trajectory patterns, a new mean-max property is identified for the trajectory patterns, and it is the basis for the developed TrajPattern algorithm that mine trajectory patterns by first finding short patterns and then extending them in a systematic manner.

2.3. Clustering moving objects and trajectories

Li, Han, and Yang [7] study the problem of clustering moving objects. Clustering analysis had been studied successfully before on static data, but without the support of on-the-fly changes. There are solutions that use moving micro-clusters for handling very large datasets. A micro-cluster denotes a group of objects that are not only close to each other at current time, but also likely to move together for a while. In principle, those moving micro-clusters reflect some closely moving objects, based on which the high quality clustering result can be obtained naturally. The paper [7] extends the idea and uses moving micro clusters (MMC), grouping close objects which also move together for a while. It proposes incremental algorithms to keep the moving micro clusters geographically small by identifying split events when the bounding rectangles reach some boundary and by using knowledge about the collisions among the MMC (splitting or unifying MMCs when those events occur). In this way moving micro clusters are dynamically maintained.

In experiments conducted on synthetic data, with K-means as the generic algorithm used in the micro-clustering, MMC showed improvement in running times compared to NC (normal clustering), though with a slight deterioration in performance.

The problem of trajectory clustering is also approached at D'Auria, Nanni, and Pedreschi [2]. They propose clustering trajectory data using density-based clustering, based on distance between trajectories.

The OPTICS system uses reachability distance (from one point to the next) between points and presents a reachability plot showing objects ordered by visit times(X) against their reachability measure (Y), allowing users to see the separation to clusters to decide on a separation threshold. This paper [2] proposes clustering patterns by temporal focusing approach, with the aim of exploiting intrinsic semantics of the temporal dimension to improve the quality of trajectories. Some changes to OPTICS are suggested, by focusing on the most interesting time intervals instead of checking all intervals, the interesting intervals are those with optimal quality of the achieved clusters.

Comparing K-means, three versions of hierarchical agglomerative clustering and the trajectory version of OPTICS on synthetic dataset generated by CENTRE trajectory generator, shows that OPTICS improves purity, with a decrease in completeness.

In Nehme and Rundensteiner [9] The SCUBA algorithm is proposed for efficient processing of large numbers of spatio-temporal queries on moving objects. The authors describe the incremental cluster formation technique that efficiently forms clusters at run-time. Their approach assures longevity and quality of the motion clusters by utilizing two key thresholds, namely distance threshold and speed threshold. SCUBA combines motion clustering with shared execution for query execution optimization. Given a set of moving objects and queries, SCUBA groups them into moving clusters based on common spatio-temporal attributes.

To optimize the join execution, SCUBA performs a two-step join execution process by first pre-filtering a set of moving clusters that could produce potential results in the join-between moving clusters stage and then proceeding with the individual join-within execution on those selected moving clusters. Experiments show that the performance of SCUBA is better than traditional grid-based approach where moving entities are processed individually. SCUBA demonstrates efficient execution of queries on moving objects that have common spatio-temporal attributes, has low cluster maintenance/overhead cost, and naturally facilitates load shedding using motion clusters while optimizing the processing time with minimal degradation in result quality.

2.4. Similarity measures

The problem of finding the most appropriate similarity measure for spatio-temporal data classification is approached at Vlachos, Kollios, and Gunopulos [11], suggesting the use of non metric similarity functions based on the Longest Common Subsequence (LCSS), that give more weight to similar parts of the sequence, and are robust to noise that usually appears in the case of trajectories in two or three dimensional space.

This measure allows stretching sequences over time (different speeds) and translating the sequence in space (similar motion in different space regions). The paper provides approximate algorithms for computing these measures.

Experiments on real datasets (Australian sign language, SEAL, video tracking data) and synthetic datasets compared these new methods to Euclidean and Dynamic Time Warping distance functions (DTW) and showed the superiority of the suggested approach, especially under strong presence of noise.

2.5. Spatio-temporal data generation

The problem of generating spatio-temporal data in order to create available datasets for research targets is approached at Giannotti et al. [3], where the CENTRE system is provided (CEllular Network Trajectories Reconstruction Environment). Their aim is to generate benchmark datasets for cellular devices positioning data, since it is not publicly available for scientific research. A spatial temporal data is represented as {object id, antenna id, time, distance}. The system aims at simulating semantic based movement behaviors from a setting of user parameters, and allows adding user preferences which may influence random distributions or domain semantics like cartography or geographic constraints. CENTRE is composed of 3 modules. The first, Synthetic Trajectories Generation, is based on the Generator for Spatio-Temporal Data (GSTD) and generates objects simulating people movements. The second, Logs Generation, simulates the telephone detection by the network and produces the position log. The third, Approximated Trajectories Reconstruction, performs reconstruction starting from the logs, considering the approximation of the data.

2.6. Incremental maintenance of moving patterns

Ma et al. [8] study a fast incremental updating technique for maintenance of maximal moving pattern. It proposes "PrefixTree+" algorithm that use the last time mining results to improve the mining efficiency. Its novelty is materializing prefix trees, and using the lemma that any moving sequence that is potentially frequent with respect to a database must occur as a frequent moving sequence in at least one of the partitions. In this technique maximal moving sequential patterns are stored in prefix trees, and new moving sequences can be combined with the existing patterns. Performance study has shown that this algorithm is more efficient and scalable than "LM" and "PrefixTree".

2.7. Gaps in related work and open research problems

In the last sections we have reviewed different research problems in the area of mining behavior of massive moving objects.

We saw different batch (non-incremental) algorithms for summarizing spatio-temporal data points like segmentation or summary transactions, but those methods may be expensive in terms of running times, and that might be a problem when adopting those methods for real time applications. The same problem exists in the methods reviewed for spatio-temporal group patterns mining, since finding frequent patterns out of similar sub-sequences requires intensive computation.

Existing methods for incremental spatio-temporal clustering that dynamically maintain moving micro clusters (by identifying split events using knowledge about the collisions among the MMC or based on common spatio-temporal attributes in SCUBA) are also

computationally expensive and not practical for real time applications to massive streams of spatio-temporal data. Other methods, like focusing on interesting times when clustering trajectory data based on distance between trajectories, are less efficient in the generic case when all times are equally significant.

Moreover, the methods we have reviewed creates clusters of objects that always move together during the cluster's duration, while we are interested in clusters of objects that have some identical movement, but with the possibility of different movement patterns at some of the time intervals that the cluster covers.

We design an algorithm for clustering moving objects that better fits real-time application needs in terms of run time and memory space.

The similarity measures we have reviewed are not fitted for spatio-temporal patterns. We define similarity measures to match our trajectory representation definition and to enable finding similarity between spatio-temporal trajectories.

2.8. Originality and Expected Contribution

In this work we detect evolving groups of moving objects using incremental clustering techniques on trajectories data. The result of this work is an algorithm for incrementally discovering mobile object clusters. The clusters discovered by algorithms like MMC [7] and SCUBA [9] are clusters of objects that move together at the same time, while we search for clusters of objects having similar but not necessarily identical moving patterns ,.

We use an incremental clustering method that exploits knowledge from previous clustering results in order to decrease computations. Also, other methods like SCUBA and MMC use incremental clustering for updating moving micro-clusters of several objects rather than updating trajectories of the same object.

Since running times will be reduced, our proposed algorithms will also be more suitable for real-time applications than existing methods.

3. Specific methods

3.1. Generating spatio-temporal data

In the absence of real spatio-temporal datasets, mainly according to privacy issues, synthetic data is generated using the CENTRE system [3] that simulates the behavior of cellular users. CENTRE software is freely available at the URL: http://wwwkdd.isti.cnr.it/CENTRE/

The data is asynchronous. It is not obtained at a constant frequency, since we assume events occur once in a while, and not at constant time intervals.

The proposed algorithm is evaluated on this dataset as described in the experimental results section.

3.2. Representing trajectories

A spatio-temporal trajectory is a series of data-points traversed by a given moving object during a specific period of time. In this work we assume that each object has only one trajectory in a given period of time.

As a part of our suggested preprocessing techniques we represent a trajectory as a list of minimal bounding boxes (MBB). A minimal bounding box (MBB) represents an interval bounded by limits of time and location. It is similar to the Minimum Bounding Rectangles (MBRs) representationused in [1].

An updating method for a MBB will be needed for building a trajectory, and also later for the updating clusters phase. We calculate its bounds as the minimal and maximal value of a data-point in that dimension, and we calculate its property values by aggregating the matching values in the MBB members (e.g., the amount of data-points in each MBB is calculated as the aggregation of data-points in all sub-trajectories of the box interval). The MBB bounds will be calculated as follows:

$$i.x_{\min} = \min(\forall m \in i, m.x_{\min})$$
$$i.x_{\max} = \max(\forall m \in i, m.x_{\max})$$
$$i.y_{\min} = \min(\forall m \in i, m.y_{\min})$$
$$i.y_{\max} = \max(\forall m \in i, m.y_{\max})$$
$$i.t_{\min} = \min(\forall m \in i, m.t_{\min})$$
$$i.t_{\max} = \max(\forall m \in i, m.t_{\max})$$
$$i.p = avg(\forall m \in i, m.state)$$

Where i represents a MBB in the centroid, m represents a member in a box, x and y are spatial coordinates, t is time and p stands for the value of a property variable.

Each MBB records maximal and minimal times, X, Y, and the number of data-points that indicate the presence of the object between minimal and maximal X and Y coordinates, and between the minimal and maximal time of occurrence.

A trajectory for an object O1 is represented as: $\{O1,[t_1,t_2,x_1,x_2,y_1,y_2,N_1],[t_3,t_4,x_3,x_4,y_3,y_4,N_2]..,[t_{n-1},t_n,x_{n-1},x_n,y_{n-1},y_n,N_n]\}$ Where t represents time, x and y represent coordinates, and N represents the amount of data points that belong to this MBB.

Figure 1 describes an object's trajectory for a given day.

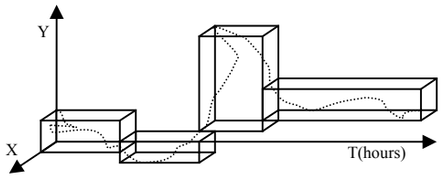

Figure 1. Object's trajectory

Data-points update the MBBs by the order of their occurrence times. Therefore, the minimal time bound of the first MBB is the time of the earliest data-point in the dataset, and the maximal time bound of this MBB will be stretched till the space distance between the maximal and the minimal locations of this MBB is equal to some maximal distance threshold (that will be determined after some initial performance trials, and will be set to some minimal percentage of the maximal value given in the dataset for that dimension). When this threshold is met, a new minimal bounding box will be created with the time of a data-point next in time as its minimal time bound. Two sequential data-points whose times are far by more than a maximal threshold (that will also be determined in the same way) will create a new MBB.

A new incremental algorithm for building an object trajectory by continuously updating its MBBs is described below:

```
Input: a set of data-points (D) describing an object
movement, a threshold of x and y distances within MBB, a
threshold of time duration of an MBB.
Output: new objects' trajectory (T)
Building an object's trajectory:
T.addMBB(D[1])
For i= 2 to D. length              // for each data point
  while( |D[i].X-T.lastMBB.maxX|<XdistThreshold
  and |D[i].Y-T.lastMBB.maxY|<YdistThreshold
  and |D[i].T-T.lastMBB.maxT|<durationThreshold)
       T.lastMBB.addPoint(D[i])  // insert into current MBB
  T.addMBB(D[i])  // if exceeds bounds create a new MBB
```

The algorithm moves along each data point along the dataset (sorted by its timestamp) and inserts the data point into an existing MBB if it is within the bounds accepted as parameters; Otherwise it creates a new MBB. "lastMBB" returns the MBB with the maximal (latest) time bounds in the trajectory, "addMBB" initializes a new MBB in the trajectory with bounds and properties updated by the accepted data-point (in the first creation minimal and maximal are equal to the data-points values) and "addPoint" updates the properties of an MBB accepted as a parameter, and also updates its bounds when needed.

The thresholds accepted as parameters to this algorithm will be determined after some initial tests. We will test some bounds between 0.5 and 2 percents of the maximal value in the dimension, but this is certainly an open question for farther research.

3.3. Defining similarity measures

When each object has one trajectory that defines its pattern, a similarity measure between two objects is actually the similarity between two trajectories.

We define similarity between two trajectories as the sum of the similarities between the trajectories' MBBs, as described in figure 2.

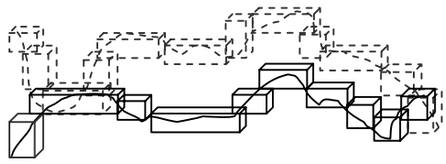

Figure 2. Similarity between two trajectories

We define similarity between two MBBs as a combination of their similarities on X, Y, and time axes, similarity between their amount of data points, and durations: $sim_{MBB} = (sim_X + sim_Y) \cdot sim_T \cdot sim_{Dur} \cdot sim_{\#}$

We have three options for similarity in each of the X,Y and time dimensions according to the overlapping levels; Full overlapping, as described in figure 3.a, gets a similarity value of 1. Partial overlapping, as described in figure 3.b, gets a similarity value that is equal to the length of the overlap divided by the length that the two MBBs extend to. No overlapping, as described in figure 3.c, means that there is no similarity in this dimension.

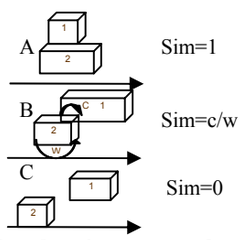

Figure 3. Overlapping stages of two MBBs: a. Full overlapping; b. Partly overlapping; c. No overlapping

We also calculate a similarity of data points amount as the minimal amount of data points (among the two MBBs) divided by the maximal amount. The similarity of the MBBs duration is calculated as the duration of shortest MBB divided by the duration of the longest MBB.

This similarity measure will be stronger as the MBBs are smaller, meaning that the more we summarize the raw

data at the preprocessing stage, the less accuracy we will get in the rest of the phases.

3.4. Object clustering

An object cluster contains objects that have similar trajectories. Objects in the same cluster are frequently (in most of the time intervals) close in space. An object cluster represents a group of moving objects that use similar trajectories but not necessarily move together in the same trajectory all the time.

Since in order to run generic clustering algorithms on the trajectories data, the data needs to be converted into a relational format, we developed a trajectory-fitted version of incremental KMEANS algorithm for clustering objects according to their trajectories by the similarity measures defined earlier. This version handles data within intervals, and with variable amount of attributes. It uses a new similarity measure that was defined in the last section, and a new centroid structure which is a trajectory (a list of unknown amount of MBBs). It also requires a new method for updating the clusters centroids, since instead of averaging centroids' data points, we bound centroid's MBBs.

Each object is represented by an object ID and by a set of its trajectory clusters (as described ahead).

This algorithm receives as input a set of trajectory clusters of the form: {O1,[t_1,t_2,x_1,x_2,y_1,y_2,N_1],[t_3,t_4,x_3,x_4,y_3,y_4,N_2]..,[t_{n-1},t_n,x_{n-1},x_n,y_{n-1},y_n,N_n]}O2,[t_1,t_2,x_1,x_2,y_1,y_2,N_1],[t_3,t_4,x_3,x_4,y_3,y_4,N_2]..,[t_{n-1},t_n,x_{n-1},x_n,y_{n-1},y_n,N_n]}…. and it outputs new object clusters of the form: {[c1,(O1,O2,O9), trajectory centroid of c1], [c1,(O3,O7,O8), trajectory centroid of c2]…)}.

We adapt the incremental approach in order to benefit from the difference between the initial clustering, when no previous data is available, and the next clustering processes, where using previous clusters' centroids can help performing a more efficient incremental clustering process (less updates are needed assuming that most of the constant groups stay relatively stable). With the non-incremental approach, the clustering algorithm is applied repeatedly (e.g., on a daily basis) to all data accumulated so far.

An algorithm for incrementally discovering object clusters during a given period of time includes the initialization of cluster centroids according to the previous cluster results of the same objects, when early clustering results exist.

4. Evaluation Methods

Since we have not found an algorithm in the literature that performs the same task that we perform here (the clusters searched for in the literature are moving clusters with objects that move together, while we search for clusters with objects with many identical movements, but with the possibility of some different movements), we cannot compare our suggested algorithm to another proposed algorithm from the reviewed literature. Instead, the proposed algorithm is tested for correctness and evaluated with with and without incremental learning in order to estimate its efficiency and effectiveness. Experiments were conducted on synthetic datasets that were generated using the CENTRE generator [3].

4.1. Cluster validity estimation

Since clustering is an unsupervised machine learning technique, there is no set of correct answers that can be compared to the results. We can, though, generate data sets by some logic that will help building the correct partition into groups, and then test the clustering results using the Rand index that measures clustering accuracy compared to ground truth. Rand index performs a comparison of all pairs of objects in the data set after clustering. "Agreement" is a pair that is together or not together in both the real and the measured clusters, and "disagreement" is the opposite case. The Rand index is computed as: [10] $R_I = \frac{A}{A+D}$

The Dunn index measures the overall worst-case compactness and separation of a clustering, with higher values being better. [10] $D_I = \frac{D\min}{D\max}$. Where Dmin is the minimum distance between any two objects in different clusters and Dmax is the maximum distance between any two items in the same cluster. The numerator captures the worst-case amount of separation between clusters, while the denominator captures the worst-case compactness of the clusters.

4.2. Clustering efficiency estimation

We measure running times of the clustering algorithms with certain parameter values, in order to compare to running times with other parameter values.

We also theoretically compute the required memory space.

In addition, the algorithms' scalability will be tested by measuring running times of the same clustering algorithm versus different number of objects that are being clustered.

5. Experimental Results

Our experiments were run on a PC Intel Pentium at 1.86GHz with 1GB RAM and 60GB hard disk.

5.1. Simulating Data

For the empirical evaluation of the proposed incremental clustering algorithm, we used synthetic data, produced using the CENTRE generator. We simulated 30 mobile objects during five different week days. These objects belong to 6 groups (The CENTRE generator requires the definition of groups and the objects that belong to each group). The maximal number of shifting between groups was set to 10, and each object was sampled at 200 snapshots (between 0 and 1) in each day.

We clustered the objects five times, one clustering for each day, using two versions of our suggested algorithm; One using a non-incremental method, and the other one using the incremental method.

Both time and distance bounds were set to 5 (no time restriction exists in summarization phase since time is between 0 and 1). We ran experiments with two KMeans iterations amount bounds: 15 and 30, and with the following numbers of clusters (K): 3, 4, 5, 6, 8, 10.

5.2. Results

We compared the non-incremental and incremental versions of our suggested algorithm according to their running times, dunn indexes and rand indexes (the last two are for checking clusters quality as mentioned earlier, when using the rand index is possible since we defined the groups in advance).

The next figures will present only four days clustering results since the first clustering of the five is not incremental and thus will act identical in both versions.

We did not compare memory space since both methods have similar requirements (o(n) for summarization phase, and (n*record-size)o(m)+o(m)+o(k)((2m+k)*record-size)) for clustering phase, when n is records amount and m is summarized records amount).

Results show that the incremental algorithm usually decreases running times. For example, figure 4 presents clustering into 3 clusters with 15 iterations. It demonstrates a decrease in running times in each of the four week days we clustered.

Figure 4. Running times comparison when clustering into 3 clusters with 15 iterations.

The same is true, yet less significant for K = 30, as shown in figure 5.

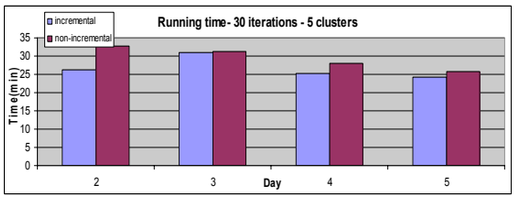

Figure 5. Running times comparison when clustering into 5 clusters with 30 iterations

We can see in figure 6 that the incremental algorithm usually improves cluster quality according to the Dunn index (the Y axis).

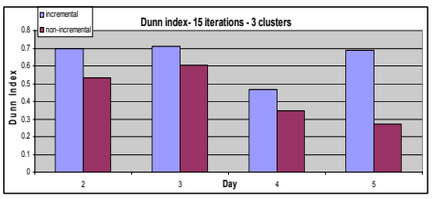

Figure 6. Dunn index comparison when clustering into 3 clusters with 15 iterations

When K = 30, the improvement is similar, as shown in figure 7.

Figure 7. Dunn index comparison when clustering into 3 clusters with 30 iterations

We can see in figures 8 and 9 that the incremental algorithm usually increases cluster quality according to the rand index (the Y axis), both for K = 15 and for K = 30. The improvement is maximal when K is equal to the amount of groups (6 in our case), and is also significant when K is greater than the group amount.

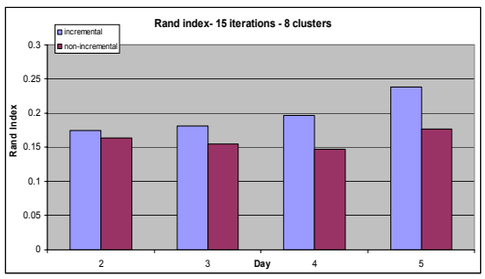

Figure 8. Rand index comparison when clustering into 8 clusters with 15 iterations

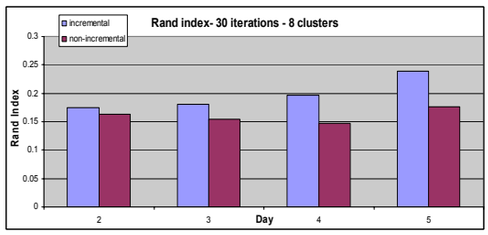

Figure 9. Rand index comparison when clustering into 8 clusters with 30 iterations

6. Conclusions

In this paper, new ways for summarizing spatio-temporal data as trajectories were presented, including a new similarity measure between trajectories and an algorithm for recognizing clusters of similar moving objects by incremental clustering methods.

Our proposed incremental algorithm for clustering moving objects was shown to be a time efficient and quality improving technique for recognizing groups of similar moving objects.

Further work is needed for tuning the parameters of the algorithm, for visualizing the results and for improving the suggested similarity measure. One can also search for improvements when dealing with large amounts of data by pre-clustering each object trajectories into one trajectory-centroid to be used as the summarized trajectory that were the input of our proposed clustering algorithm.

7. Bibliography

[1] Anagnostopoulos A., Vlachos M., Hadjieleftheriou M., Keogh E.,Yu P.s. (2006) "Global Distance-Based Segmentation of Trajectories". *KDD'06*, August 20–23, 2006, Philadelphia, Pennsylvania, USA.

[2] D'Auria M., Nanni M., Pedreschi D. (2006) "Time-focused density-based clustering of trajectories of moving objects" To appear in JIIS Special Issue on *"Mining Spatio-Temporal Data"*.

[3] Giannotti F., Mazzoni A., Puntoni S., Renso C. (2005), "Synthetic Generation of Cellular Network Positioning Data". GIS'05: *Proceedings of the 13th ACM International Workshop on Geographic Information Systems*, p 12-20.

[4] Goh J.Y., Taniar D., (2004). "An Efficient Mobile Data Mining Model". *Lecture notes in computer science*, Springer. vol. 3358, p. 54-58.

[5] Hwang S.Y., Liu Y.H., Chiu J.K., Lim F.P. (2005) "Mining Mobile Group Patterns: A Trajectory-based Approach". *Lecture Notes in Artificial Intelligence* (Subseries of Lecture Notes in Computer Science), v 3518, Advances in Knowledge Discovery and Data Mining: 9th Pacific-Asia Conference, PAKDD 2005. Proceedings, p 713-718

[6] Jain A. K., Murty M. N. and Flynn, P. J.(1999), Data clustering: a review, *ACM Computing Surveys (CSUR)*, v.31 n.3, p.264-323

[7] Li Y., Han J., Yang J. (2004) "Clustering Moving Objects". *KDD-2004 - Proceedings of the Tenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2004, p 617-622

[8] Ma S., Tang S. Yang D., Wang T., Yang C. (2004) "Incremental Maintenance of Discovered Mobile User Maximal Moving Sequential Patterns". *Lecture notes in computer science, Springer*. vol. 2973, p. 824-830

[9] Nehme R. and Rundensteiner E.A. (2006). "SCUBA: Scalable Cluster-Based Algorithm for Evaluating Continuous Spatio-Temporal Queries on Moving Objects", *10th International Conference on Extending Database Technology*, Munich, Germany, March 26th - 30th, 2006

[10] Schenker A., Bunke H., Last M., Kandel A. (2005) "Graph-Theoretic Techniques for Web content Mining", World Scientific, *Series in Machine Perception and Artificial Intelligence*, Vol. 62

[11] Vlachos M., Kollios G., Gunopulos D. (2002) "Discovering Similar Multidimensional Trajectories". *Proceedings - International Conference on Data Engineering*, p 673-684

[12] Wang Y., Lim E.P., Hwang S.Y. (2003) "On Mining Group Patterns of Mobile Users". *Proceedings of the 14th International Conference on Database and Expert Systems Applications*, DEXA, pp. 287-296.

[13] Yang J., Hu M. (2006) "TrajPattern: Mining Sequential Patterns from Imprecise Trajectories pf Mobile Objects". *Lecture notes in computer science*, Springer. vol. 3896, p. 664-681.