**International Academic Institute for Science and Technology**

International Academic Journal of Science and Engineering
Vol. 3, No. 1, 2016, pp. 41-52.
ISSN 2454-3896

**International Academic Journal of Science and Engineering**

www.iaiest.com

# A New Approach to Trajectory Data Clustering

Anahita Ghalami[a], Hassan Naderi[b]

[a]*Master of Science in Computer Software Engineering, Iran University of Science and Technology.*

[b]*Faculty Member of Computer Engineering Department, Iran University of Science and Technology.*

## Abstract

Data analysis of the moving object trajectories and extracting their moving patterns using data mining methods has attracted the interest of researchers in recent years. In many moving objects management programs, analysis of data clustering has turned out to be an essential requirement. Clustering is one of the most significant methods of analysis that groups similar data to produce a summary of the data distribution patterns in a dataset. In this paper, after examining the existing algorithms for clustering the moving object trajectories in an environment where moving objects are not limited in road networks, one of the most prominent algorithms in the field called Traclus was selected. The algorithm rests on the partition-and-group framework. In this algorithm for clustering subtrajectories, only the latitude and longitude points of the trajectory are taken into account. Therefore, this algorithm can only identify dense points without considering the time when these areas have been dense. However, in many cases, it is essential to determine what areas have been dense and at what times. Therefore, with considering the temporal characteristics of the trajectory points, the algorithm was developed and our own algorithm called T-Traclus was proposed. The advantage of this method over existing methods is its capability of detecting dense areas at the same time.

**Keywords:** Spatio-temporal databases, moving objects trajectory, clustering, trajectory clustering.

## Introduction:

Given the rapid developments in GPS and wireless technology, nowadays we are exposed to a wide range of spatial data. We are also able to collect movement data generated by mobile users, cars, buses, planes, animals, etc. GPS, sensors, monitoring visual systems and the Internet are being utilized more than ever to record moving entities. The movement of entities that is recorded case by case can be connected together

in the form of a sequence as a trajectory (L. Bermingham & I. Lee, 2015). Increase in movement data along with the growing number of motor vehicles (annual increase of 3.69 million from 1960 on) (X. Meng et al, 2014) has provoked the need for efficient techniques for the analysis and management of movement data. Thus, given the fact that the datasets of moving objects are massive and complex, effective and efficient visual analysis techniques and data mining algorithms are required to extract useful information and explore the patterns in these immense movement datasets. In fact, the abundance of trajectory data brings about a worthwhile opportunity for the industry and the scientific community to extract previously unknown patterns and lucid patterns of the movement of entities. A variety of scientific disciplines including research on database, transportation analysis, flight control systems and research on migration behavior of animals have manifested a great desire to investigate the movement patterns of moving objects. Similarly, in location-based services, it is of paramount importance to know the places that are tremendously crowded because it can contribute enormously to marketing policies (X. Meng et al., 2014).

According to the investigations conducted so far, based on their movement, spatial-temporal data pertinent to moving objects can be broken down into two categories: the movement trajectory of objects that move on the road network and the movement trajectory of objects that move in a space that is not limited. After a brief look at each of these two areas, clustering of spatial-temporal data related to the movement of the moving objects in a space in which they move freely was chosen as the field work. Hence, we investigated the algorithms available for clustering trajectories in an environment where moving objects are not limited. One of the leading algorithm in this area is Traclus. Due to the significance of concurrently considering the place and time of the movement trajectory of moving objects, in this paper, we will take the spatial and temporal parameters of trajectory into account to cluster them based on the development of Traclus Algorithm. Clustering sub-trajectories with considering the spatial and temporal information allow us to identify areas of high traffic based on time.

In the rest of the paper, the second section provides a review of the literature and the third section offers definitions of concepts related to clustering trajectory data. It is followed by section four that presents the proposed method. The fifth section is devoted to the evaluation of the proposed method. Finally, in the sixth section, conclusions and suggestions for further research are discussed.

## Literature Review

In recent years, plentiful research has been conducted on clustering of moving objects with a myriad of applications. In many cases, the objects move in networks with limited space. In case of clustering moving objects in road networks, we can refer to an integrated framework provided for clustering moving objects in space networks (CMON). CMON framework works on the basis of the concept of clustering block. This framework has been developed based on three criteria resting on distance, density and K-partitioning (X. Meng et al, 2014). NEAT and FlowScan Algorithms have also been presented for clustering the trajectory of moving objects in road networks. Binh Han et al. (2012) have proposed NEAT as a road network aware approach for fast and effective clustering of spatial trajectories of mobile objects travelling in road networks. They argued that when spatial clustering of moving object trajectories aims at road network aware location based applications, density and Euclidean distance are no longer viewed as the effective measures. The reason is that traffic flows in a road network and the flow-based density characterization

have turned out to be critical factors for discerning trajectory clusters of mobile objects travelling in road networks. Their method considers the physical limitations of the road network, the network proximity and the traffic flows among successive road segments to organize trajectories into spatial clusters. FlowScan Algorithm detects the highest traffic density in the road network (Y. Zheng and X. Zhou, 2011). Instead of clustering moving objects, this algorithm clusters traffic density-based routes in road networks.

Many of the present studies focus on clustering moving objects with the assumption of a space of free movement, like the movement of animals and air masses. With focusing on variables that have been repeatedly measured, Genolini et al. presented a new idea concerning the joint evolution of several variables and concluded that a joint study is far more significant (Genolini et al., 2013). In articles written by Genolini et al. (2013) and Ossama et al. (2011), k-means algorithm has been employed to find out the cluster and common trajectories. This method handled the missing values as well as presented a number of quality criteria for detecting better partitions. Chu et al. have utilized FCM (fuzzy c-means) for finding the trajectory of storm movement and comparing it with clustering, normalization techniques of cutting and k-means algorithm (H.J. Chu et al., 2012). Karavasilis, et al. have proposed an algorithm that extracts trajectories using the sequence of images based on clustering model (V. Karavasilis, et al, 2012). This integrated model serves as the regression of trajectories of clusters with variable longitude. Parameters of this model are selected using the expectation-maximization algorithm. Some algorithms available for clustering trajectories focus on geometrical and spatial characteristics of trajectories. Despite many of existing clustering algorithms, Debnath, et al. have presented a new framework for clustering sub-trajectories based on integrating spatial and non-spatial characteristics (M. Debnath et al., 2013). This algorithm integrates methods of network, spatial geometry and string-based approaches. Jiashun and Dechang have offered a new method for clustering trajectories based on partition cluster element (PCE) (C. Jiashun and P. Dechang, 2013). They have presented a new partition method called PCE based on the partition framework and grouping and finally, based on PCE and related definitions, have offered a new algorithm called 'partition clustering element trajectory cluster '(PCETC). This method is comprised of three steps: The trajectory partitioning, clustering and extraction. Accordingly, line segments of similar trajectories have been used to refer to the geometrical properties of space. Clustering process can be carried out by classifying similar line segments to from trajectories. Pao, et al. have analyzed diverse trajectories including mouse movement trajectory, written characters, and animal movement and presented a general approach to user verification based on these trajectories (H. K. Pao, et al., 2012). They have described the behavior trajectory using Markov chain model with a Gaussian distribution.

Lee, et al. proposed Traclus algorithm. They expanded the partition-group framework for clustering trajectories so that trajectories are divided into line segments and then similar lines are placed into same cluster (J. Gill Lee, et al, 2007). The majority of algorithms for clustering trajectories including Traclus need to insert two parameters in entry and are highly sensitive to entry parameters. Incorrect settings of these parameters may lead algorithm to producing false clusters. Owing to this vulnerability, Jiashun has presented a trajectory clustering algorithm, called SPSTC, to protect the sensitivity of parameters (C. Jiashun, 2012). Note that there are other applications of clustering tools, such as K-means, etc., in the literature. For example, k-means clustering has been used by Rastegari and Mobin (2016) for maintenance

decision-making; k-means is used to improve the performance of the evolutionary algorithm by Mobin et al. (2017).

## Statement of the Problem

The present study intends to present an efficient method for clustering moving object trajectories. To do so, we are planning to propose a method for clustering trajectories with considering spatial and temporal parameters concurrently based on Traclus Algorithm.

***Definition 1:*** A trajectory is the path that a moving object follows through space and and is illustrated as a given set of points as a function of time. For instance, it is denoted in the form of P1→P2→…→Pn. Each point has geographical and temporal characteristics: P=(x,y,t) (Y.Zheng and X.Zhou, 2011).

***Definition 2:*** Suppose V is a set of vectors $V = \left\{\overrightarrow{v_1}, \overrightarrow{v_2}, \ldots, \overrightarrow{v_n}\right\}$, Vector toward average, that is $\overrightarrow{V}$ for the set V is defined as Formula (1). Here, |V| is cardinality of V.

$$(1) \quad \overrightarrow{V} = \frac{\overrightarrow{v_1} + \overrightarrow{v_2} + \ldots + \overrightarrow{v_n}}{|V|}$$

Traclus Algorithm consists of two main phases: partitioning and grouping ( J. Gil Lee, et al., 2007). In the first step, partition is being carried out. The purpose of partitioning is to find out trajectory points on which behavior of trajectory changes quite rapidly. These points are called characteristic points. A trajectory is partitioned on the basis of these characteristic points and each partition features two characteristics points. The best partitioning of a trajectory must have two properties: It must be precise and concise. Therefore, it is essential to set a tradeoff between precision and concision. To explore the tradeoff, 'minimum length description principle' (MDL) (P. Grunwald, I. J. Myung and M. Pitt, 2005) has been employed. In the second step, density-based line-segment clustering algorithm, based on DBSCAN, has been presented for clustering trajectories. In this step, density that is defined according to distance measure which is the sum of perpendicular, parallel and angle distance between line segments, is assessed. Line segments are assigned into a group with a line segment that their distance from that line segment is less than Parameter ε. It is carried out by exploiting execute procedure ComputeNε (L). This method works in a way that for each input line segment like L, line segments that are in ε distance are explored and considered as neighbours to L. In the event that the minimum numbers of line segments available in a group are equal to or greater than MinLns parameter, the group in question is viewed as a cluster. Eventually, for each cluster, a representative trajectory is computed. The representative trajectory of a cluster describes the overall movement of the trajectory partitions that belong to the cluster. It can be considered a model for clusters. In this article, a version of the algorithm developed by adding time parameters will be proposed. It is called T-Traclus algorithm. To add the time parameter, it is needed to provide a definition for a time distance of two line segments.

***Definition 3:*** For two line segments with time sharing, the total length of the time difference intervals of two line segments divided by the sum of length of time intervals of two line segments is called the time

distance between two line segments. It is denoted by $d_t$ Obviously, this distance criterion is always a value between 0 (if there is concurrency between two line segments) and 1 (in case there is no time sharing (concurrency) between the two line segments).

## The Proposed Method

Phases 1 and 2 are carried out in a similar way to Traclus Algorithm, but they differ in that in Phase 2, before executing ComputeNε(L) procedure in this algorithm, we have to conduct a pre-processing. Our suggestion here is that before computing the distance between Line Segment L and any other line segment, initially their concurrency is considered and it should be demonstrated whether those two line segments have any concurrency or not. If two line segments have no concurrency, they are not examined in terms of spatial distance. But if Line Segment L has time sharing with another line segment, their time distance is calculated in accordance with the definition provided above. In the event that the time distance between two line segments is grater greater than a threshold, called 'interval threshold', these two line segments cannot be considered as neighbors. However, if their time distance is less than the threshold, the start and end points of each of those line segments must be specified at the beginning and end of the concurrent time interval ($[t_s, t_e]$). Then, the distance between those two line segments is computed according to the new start and end points. To obtain the new start and end points, it is needed to calculate the specifications of points of line segments in $t_s$ and $t_e$. To do so, we can use the formula (2) that was presented in Y.Zheng and X.Zhou (2011).

(2) $x_1 = x_0 + \frac{t_1 - t_0}{t_5 - t_0}.(x_5 - x_0)$ , $y_1 = y_0 + \frac{t_1 - t_0}{t_5 - t_0}.(y_5 - y_0)$

Where ($X_0$, $Y_0$, $T_0$) are start points and ($X_5$, $Y_5$, $T_5$), end points of line segments.

In Figure 1, we have displayed the operation that is needed to be carried out before calculating ComputeNε(L) for inserting the time parameter. After obtaining the new start and end points of line segments, their distance is computed and if their distance is less than the value of the parameter ε, the line segment under examination is added to the list of line segments neighboring Line Segment L.

**Algorithm Check Synchronize**

**Input: (1) startPoint1, (2) endPoint1, (3) startPoint2, (4) endPoint1**
**Output: (1) newStartPoint1, (2) newEndPoint1, (3) newStartPoint2, (4) newEndPoint1, (5) hasConcurrency**
**Algorithm:**
**01: Put the maximum of intervals start time to S;**
**02: Put the minimum of intervals end time to E;**
**03: if ( length of interval (s, e) > 0) then**
**04: convert intervals start and end points to new start and end points;**
**05: hasConcurrency := True;**
**06: else**
**07: hasConcurrency := False;**

**Figure 1: Determining the start and end points of line segments based on the time of the start and end of the concurrent time interval**

After clustering, now it is time to select the representative trajectory. In this step, we act in the same way as the main algorithm. It differs only in that here for representative trajectory points, in addition to calculating geographical longitude and latitude, it is needed to determine the time of each point. The required changes have been made to the aforementioned algorithm as in Figure 2. As it can be observed in algorithm in Figure 2, to specify the time of each point on representative trajectory, first rotating the axes in a way that the x-axis is toward the average vector. In each sweep that is perpendicular to the rotated x-axis, finding the number of intersection points of the sweep line intersecting each of the line segments. In each case that the number of intersection points is greater than or equal to Parameter MinLns and the difference between the current $X'$ (X in rotated axis) and $X'$ of the point previously calculated for the representative trajectory, is greater than the desired value of $\gamma$, we calculate the specifications of each intersection point.

**Algorithm Representive Trajectory Generation In T-Traclus**

**Input: (1) A cluster $C_i$ of line segments, (2) MinLns, (3) A smoothing parameters $\gamma$**
**Output: A representative trajectory $RTR_i$ for $C_i$**
**Algorithm:**
**01: Compute the average direction vector $\vec{V}$;**
**02: Rotate the axis so that the X axis is parallel to $\vec{V}$;**
**03: Let *P* be the set of the starting and ending points**
**of the line segments in$C_i$;**
**/* $X'$–values denotes the coordinate of the $X'$ axis */**
**04: Sort the points in the set *P* by their$X'$–values;**
**05: for each (p ϵ *P*) do**
**/* count $num_p$ using a sweep line */**
**06: Let $num_p$be the number of the line segments**
**that contain the $X'$–value of the point p;**
**07: if ($num_p$ >= MinLns) then**
**08: diff := the difference in $X'$–values between p**
**and its immediately previous point;**
**09: if (diff >= $\gamma$) then**
**10: Compute each sweep point with rotated coordinate;**
**11: Compute each sweep point reverse rotation to**
**calculate it's time and the average of those times as avg_time;**
**12: Compute average of rotated sweep points as $avg'_p$;**
**13: Compute reverse rotation of $avg'_p$to get the $avg_p$;**
**14: Set avg_time as a time of $avg_p$;**
**15: append $avg_p$ to the end of $RTR_i$;**

**Figure 2: The algorithm for calculating the representative trajectory of each cluster through the temporal specifications of points**

The average of specifications of these points is computed as $\boldsymbol{avg'_p}$ point. This point is rotated in the reverse direction so that its specifications in main axes are obtained as Point $\boldsymbol{avg_p}$. To calculate the time of this point, after obtaining each intersection point on rotated axes, the specifications of that point on main axis are attained and considering the fact that we have the specifications of start and end points of each line segment and using Formula 2, the time of that point is computed. After obtaining the time of all intersection points, the average of times are worked out and considered as avg-time. Avg-time is assigned as the time of $\boldsymbol{avg_p}$ point.

**The function of measuring the quality of clustering**

Here, SSE mentioned in J. Han, et al. (2011) is employed as the criterion for measuring the quality of clustering as QMeasure. In SSE used here, to calculate the distance between any two line segments in

clusters, in addition to the total sum of perpendicular, parallel, and angle distances, the time distance between two line segments is added up to the aforementioned three distances.

$$(3)\ QMeasure = \sum_{i=1}^{num_{clus}} \left( \frac{1}{2|C_i|} \sum_{x \in C_i} \sum_{y \in C_i} dist^*(x,y)^2 \right)$$

$$dist^*(\mathrm{x},\mathrm{y}) = w_\perp . d_\perp(\mathrm{x},\mathrm{y}) + w_\parallel . d_\parallel(x,y) + w_\theta\ d_\theta(x,y) + w_t\ d_t(x,y)$$

## Evaluating the Proposed Method

To evaluate the proposed algorithm, we implemented T-Traclus algorithm through Java programming language and JDK 1.6. The operating environment for the implementation was a laptop with Intel(R) Core(TM) 2 Duo CPU T9600 2.8 GHz processor and 4GB memory and its operating system was Win 7 64-bit.

We performed experiments on a dataset to evaluate T-Traclus algorithm. The datasets in question are related to hurricanes of the Atlantic area within 2000 and 2013. This dataset includes 218 trajectories. Each trajectory is specified by a number of points having geographical latitude and longitude as well as time. In total, such data includes 6406 points (Systems, 2015). The reason for choosing this dataset is that it consists of trajectories with freedom of movement and optional shapes; nor do they have movement limitations existing in road networks.

Since in Traclus algorithm, the time parameter is also involved, accordingly the obtained clusters differ from the output of the main algorithm because it is likely that sub-trajectories that are put together in one cluster in the main algorithm do not have the desired concurrency; therefore, in T-Traclus algorithm they are not clustered together. That is why we utilize the quality criterion mentioned in the preceding section to compare Traclus algorithm and T-Traclus algorithm. To perform T-Traclus algorithm as well as calculate QMeasure, the value of Interval-Threshold parameter is set at 0.33. That is, only line segments whose time distance from each other is less than 0.33 can be assumed as neighbours. Initially, each of the two Traclus and T-Traclus algorithms are heuristically run to find the appropriate values for Parameters E and MinLns. Entropy values for variations in E Parameter are illustrated in Figure 3. As it is clear, for Traclus Algorithm, the minimum value of entropy is obtained when ε = 10. For T-Traclus, the minimum value of entropy is obtained when ε = 15.

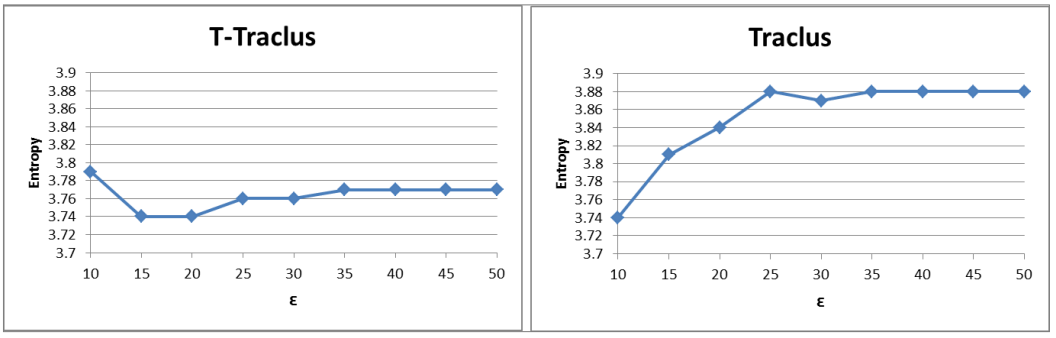

**Figure 3: Entropy for various values of ε for Traclus and T-Traclus Algorithms**

The minimum entropy obtained for T-Traclus Algorithm (4.47) equals that of Traclus Algorithm's. It indicates that the heterogeneity of obtained clusters in T-Traclus Algorithm at the optimum state resembles that of Traclus at optimum state. In Table 1, the parameters obtained for both algorithms and the obtained specifications of their clusters are displayed.

**Table 1: The results of applying Traclus and T-Traclus Algorithms with heuristic parameter ε and MinLns.**

| Traclus | | T-Traclus | |
|---|---|---|---|
| ε=10 , MinLns=3 | | ε=15 , MinLns=2 | |
| Number of clusters: 4 | | Number of clusters: 4 | |
| Cluster ID | Participated Trajectories | Cluster ID | Participated Trajectories |
| 0 | 33, 79, 112, 197, 199 | 0 | 33, 112, 197 |
| 1 | 4, 81, 98, 173 | | |
| QMeasure | 1308.34 | QMeasure | 122.52 |
| Run time | 11 seconds | Run time | 11 seconds |

As it can be observed from the above results, with taking QMeasure criterion into account, the quality of clusters obtained from T-Traclus Algorithm is better than those from Traclus Algorithm because this criterion for clustering conducted by T-Traclus Algorithm has a smaller value.

To graphically represent the trajectories in the experiment as well as the results obtained by their clustering, M-Atlas software was applied. Figure 4 illustrates the result of clustering on dataset, using T-Traclus and Traclus algorithms on this dataset. The red lines represent the trajectories of our dataset and dark lines show representative trajectories of the obtained clusters.

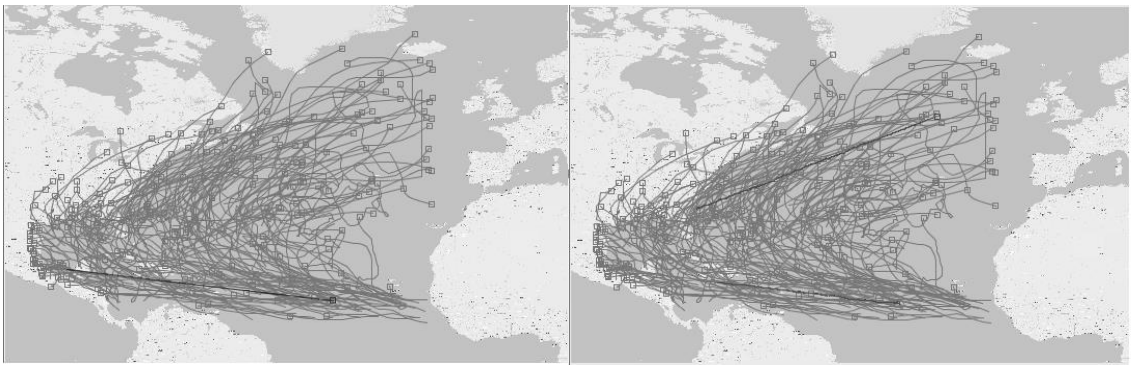

**Figure 4: (a) Clustering through T-Traclus Algorithm, (b) Clustering through Traclus Algorithm**

As it was expected, due to considering the time parameter, clusters obtained in two algorithms are different. In Figure 5, two Traclus and T-Traclus algorithms are compared based on various parameters.

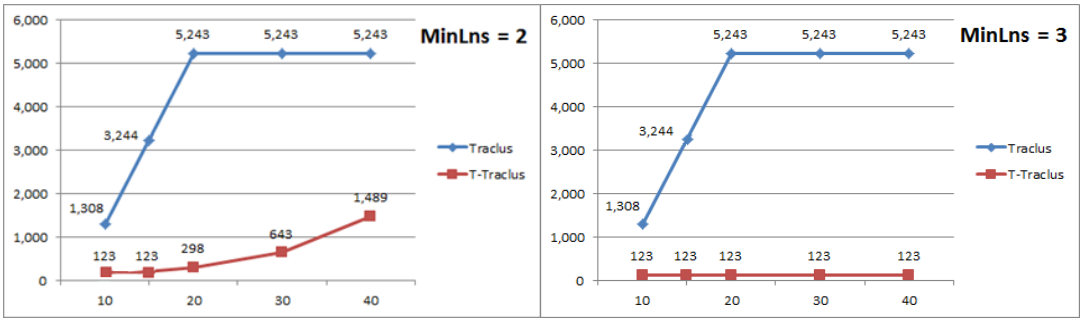


**Figure 5: (a) Comparing QMeasure in Traclus and T-Traclus algorithms through MinLns= 2 and diverse values of ε, (b) Comparing QMeasure in Traclus and T-Traclus algorithms through MinLns= 3 and diverse values of ε**

As the charts above show, the quality of clustering in T-Traclus Algorithm is higher than that in Traclus because this algorithm takes the time distance of sub-trajectories into consideration while Traclus Algorithm does not care for time parameter and for the same reason sub-trajectories that have far time distances can be placed in one cluster. It in turn raises the value of QMeasure and reduces the clustering quality. Another interesting point is that according to the results of the experiments made, our algorithm is not dissimilar from Traclus Algorithm in terms of running time. Moreover, the calculations added for examining time sharing of sub-trajectories has not brought about a marked increase in running time.

## Discussion and Conclusion

In many applications, we need to determine trajectories which are located close to each other in concurrent time intervals. From another perspective, we seek out to figure out what places are crowded and at what times. Hence, we intended to present an algorithm that in addition to detecting the proximity of trajectories with a specific neighborhood radius even for a limited interval, can take the time of trajectories into account in proximity criterion. Therefore, Traclus Algorithm was chosen as the basis for work and time parameter was also added to it; our algorithm called T-Traclus was then presented. Our algorithm is based on the partition-grouping framework and has three main phases: partitioning, grouping and selecting the representative trajectory. We intervened time criterion in grouping phase as well as selecting the representative trajectory phase. Therefore, after attaining sub-trajectories in the first step, when grouping them, only sub-trajectories whose time distance is less than the threshold value in question are likely to be assigned into a group. Thus, we had to define our own criterion for measuring the time interval of two sub-trajectories. Accordingly, the clustering step of sub-trajectories was taken. In selecting the representative trajectory step, the time characterization was considered in computations when the representative trajectory points were determined.

For further research, the performance of the proposed clustering algorithm can be improved when the evolutionary algorithms used as enhancement tools (Mobin et al., 2017; Li et al., 2016). In addition, Traclus Algorithm can be improved in a such way that is less sensitive to parameters. A number of approaches, for instance, optics, have been developed with that purpose in mind as far as points data are concerned. These approaches can be applied to the trajectory data. Moreover, because the criteria available for assessing clustering quality often is compatible with points data, defining a suitable quality measure for comparing algorithms for clustering the trajectory data provides us with a more accurate comparison.

## References:

B. Han, L. Liu and E. Omiecinski. (2012). NEAT: Road Network Aware Trajectory Clustering. Proceedings of the 2012 IEEE 32nd International Conference on Distributed Computing Systems (ICDCS '12), (pp. 142-151). Washington.

C. Jiashun. (2012). A New Trajectory Clustering Algorithm Based on TRACLUS. Computer Science and Network Technology (ICCSNT), 2012 2nd International Conference, (pp. 783-787). Changchun.

C. Jiashun and P. Dechang. (2013). A New Trajectory Clustering Based on Paritition-Cluster-Extration. Computational and Information Sciences (ICCIS '13), (pp. 783-787). Washington.

C. Genolini, J. B. Pingault, T. Driss, S. CôTé, R. E. Tremblay, F. Vitaro, C. Arnaud, and B. Falissard. (2013). A non-parametric algorithm for clustering joint Trajectories. Computer methods and programs in biomedicine, 109, 104-111.

H.J.Chu, C.J.Liau, C.H.Lin and B.S. Su. (2012). Integration of fuzzy cluster analysis and kernel density estimation for tracking typhoon trajectories in the Taiwan region. Expert Systems with Applications, 39, 9451–9457.

H.K.Pao, J.Fadlil and H.Y.Lin. (2012). Trajectory analysis for user verification and recognition. Knowledge-Based Systems, 34, 81-90.

J. Gil Lee, J. Han and K. Y. Whang. (2007). Trajectory clustering: a partition-and-group framework. Proceedings of the 2007 ACM SIGMOD international conference on Management of data (SIGMOD '07), (pp. 593-604).

J.Han, M.kamber and J.pei. (2011). "Data Mining: Concepts and Techniques" (3rd ed.). Academic Press, Morgan Kaufmarm Publishers.

L. Bermingham and I. Lee. (2015). clustering, A general methodology for n-dimensional trajectory. Expert Syst. Appl., 42, 7573-7581.

Li Z., Mobin M., Keyser T., (2016) Multi-objective and Multi-Stage Reliability Growth Planning in Early Product Development Stage, IEEE Transaction on Reliability, 65(2), 769-781.

M. Debnath, P.K. Tripathi and R. Elmasri. (2013). A Novel Approach to Trajectory Analysis Using String Matching and Clustering. IEEE 13th International Conference on Data Mining Workshops, (pp. 986-993). Dallas.

Mobin M., Li Z., Komaki M., (2017), A Multi-Objective Approach for Multi-Stage Reliability Growth Planning by Considering the Timing of New Technologies Introduction, IEEE Transaction on Reliability, DOI: 10.1109/TR.2016.2638124.

O.Ossama, H.M.O.Mokhtar and M.E.El-Sharkawi. (2011). An extended k-means technique for clustering moving objects. Egyptian Informatics Journal, 12, 45-51.

P. Grunwald, I. J. Myung and M. Pitt. (2005). Advances in Minimum Description Length: Theory and Applications. MIT Press.

Rastegari A., Mobin M., (2016) Maintenance Decision Making Supported by Computerized Maintenance Management System, Proceeding of 2016 Annual Reliability and Maintainability Symposium (RAMS 2016), Arizona, USA.

Systems, U. W. (2015, April 28). unisys.com. (Unisys Weather Information Systems) Retrieved 10 01, 2015, from http://weather.unisys.com/hurricane/atlantic

V.Karavasilis, K.Blekas and N.Christophoros. (2012). A novel framework for motion segmentation and tracking by clustering incomplete trajectories. Computer Vision and Image Understanding, 116, 1135-1148.

X. Meng, Z. Ding and J. Xu. (2014). Moving Objects Management Models, Techniques and Applications (2 Ed.). Springer Publishing Company, Incorporated.

Y.Zheng and X.Zhou. (2011). Computing with Spatial Trajectories (1st ed.). Springer Publishing Company, Incorporated.