A Time Parameterized Technique for Clustering Moving Object Trajectories

Hoda M. O. Mokhtar[1] Omnia Ossama[2] Mohamed El-Sharkawi[3]
[1]h.mokhtar@fci-cu.edu.eg [2]omnia@ieee.org [3]melsharkawi@gmail.com
Faculty of Computers and Information, Cairo University
Cairo, Egypt

ABSTRACT

Today portable devices as mobile phones, laptops, personal digital assistants(PDAs), and many other mobile devices are ubiquitous. Along with the rapid advances in positioning and wireless technologies, moving object position information has become easier to acquire. This availability of location information triggered the need for clustering and classifying location information to extract useful knowledge from it and to discover hidden patterns in moving objects' motion behaviors. Many existing algorithms have studied clustering as an analysis technique to find data distribution patterns. In this paper we consider the clustering problem applied to moving object trajectory data. We propose a "time-based" clustering algorithm that adapts the k-means algorithm for trajectory data. We present two techniques: an exact, and an approximate technique. Besides, we present experimental results on both synthesized and real data that show both the performance and accuracy of our proposed techniques.

KEYWORDS

Moving object databases; mining moving object trajectories; clustering moving objects; and similarity search in moving object trajectories.

1.Introduction

Transportation is the driving force of any country. It is important for the economic growth and for improving the mobility and quality of life for citizens. On the other hand, traffic jams and route congestions nowadays affect traffic safety. Consequently, solving the problem of severe traffic congestion has risen to the top of the agenda in many cities because of its negative effects like reducing regional economic health, and increasing pollution. The concern about traffic safety has been growing over years. During 2008 there was 43,017 fatal crashes on USA roadways [3]. In addition to the crash data for 2001-2005 provided by the General Estimates System, an average of 2,079 injuries and an average of 137 fatal rollover crashes were reported [2]. Intelligent Transportation System is an emerging application that is currently being applied and investigated by many countries [4].The adoption of Intelligent Transportation Systems (ITS) is thus an initial step towards increasing road safety. The goal behind ITS is to integrate modern communication and information technologies into existing transportation infrastructure to achieve safer, more secure, and more reliable surface transportation networks. Over the last several years, Federal Motor Carrier Safety Administration has collaborated with the trucking industry to test, evaluate, and encourage the deployment of several safety systems for commercial motor vehicles to

enhance the safety of all roadway users [2]. Analyzing traffic data and drivers' motion behavior is thus a crucial step towards achieving an efficient ITS. Data mining is a key tool that could be used as an analysis technique to support ITS. Data mining is among the recent approaches that is finding increasing acceptance in science and business areas that need to analyze large amounts of data to uncover hidden patterns and trends. Today, with the fast advances in wireless and positioning technologies, we are facing a flood of location information. Data mining or knowledge discovery is basically the process of analyzing data from different perspectives and summarizing it into useful information [12]. This information could then be converted into knowledge about historical patterns or future trends. Hence, employing data mining techniques in ITS enables traffic related decision makers to answer queries like: ***Q1****:Retrieve the highly congested squares in the last month from 2:00pm to7:00pm*", ***Q2****:Retrieve the freeways with the largest number of accidents in the last 2 years"*. Those queries along with many others can be evaluated once vehicle trajectory data is analyzed over time. In addition, clustering trajectories can also be employed in various location based services for example m-commerce. Identifying the motion pattern of mobile users and the areas that they visit is a crucial requirement in m-commerce; sending the right advertisement, to the right customer, at the right time is the key m-commerce dilemma.

Clustering moving object trajectory data is thus an appealing research direction to fulfill the needs of many applications. In general, clustering is defined as the division of data into groups of similar objects. Each group, called cluster, consists of objects that are similar between themselves and dissimilar to objects of other groups [6]. From a practical perspective clustering plays an outstanding role in data mining applications such as scientific data exploration, information retrieval and text mining, spatial database applications, web analysis, CRM, marketing, medical diagnostics, computational biology, and many others. Since data mining deals with large databases, this imposes additional severe computational requirements on clustering analysis, thus many new algorithms have recently emerged that meet these requirements and were successfully applied to real-life data mining problems. Clustering algorithms are classified into different categories [6]; *hierarchical methods* like agglomerative algorithms and divisive algorithms; *partitioning methods* like relocation algorithms, probabilistic clustering, k-medoids methods, k-means methods, and density-based algorithms; *grid-based methods*; *methods based on co-occurrence of categorical data; constraint-based clustering*; along with clustering algorithms used in machine learning like *artificial neural networks and evolutionary methods; scalable clustering algorithms*; and finally, algorithms for high dimensional data like *sub-space clustering, projection techniques, and co-clustering techniques.*

In this paper we employ the famous k-means clustering algorithm that was first proposed by Stuart Lloyd and James MacQueen [21, 26], later several variants of the algorithm were developed [10, 18],etc. We employ the traditional (basic) k-means algorithm to develop two new algorithms that are adapted for trajectory data. The key difference between trajectory data and regular data are the temporal and spatial characteristics of trajectories. Our proposed algorithms are tailored to capture both the temporal and spatial aspects. The key feature of the algorithms is the way they utilize the *time dimension* in the clustering process. Simply, we cluster the objects based on the time intervals of different trajectory's motions. Hence, if an object spans different time intervals, it will eventually belong to different clusters. Finally, each object will be associated with a linked list that shows the clusters visited by the object trajectory throughout its life time. The paper proposes two approaches, an exact approach and an approximate approach. The main difference between the two approaches lies in their query performance. Given a query object, the exact method computes the actual clusters visited by the object throughout its life time. On the other hand, the approximate method exactly computes some of the actual visited

clusters and based on those computed clusters along with the clusters generated from the remaining data set, it *"predicts"* the future motion pattern of the query object.
The rest of this paper is structured as follows: Section 2 presents a literature survey of key related work. Section 3 defines the problem and presents our technical definitions. Section 4 presents our proposed clustering algorithms. Section 5 shows our experimental results. Finally, section 6 concludes the paper and proposes directions for future work.

2.Related Work

With the extensive existence of moving objects and the fast development in positioning systems, new challenges are imposed on database systems. Managing, storing, and querying moving objects were the key research focus in many work including [24, 32, 27, 23]. Recently, applying data mining techniques to moving object databases became a crucial direction for many applications. There is a deep relationship between clustering techniques and many other disciplines [19]. Clustering in data mining was brought to life by the developments in information retrieval and text mining [30], spatial database applications, for example, GIS or astronomical data [34], web applications [13], and DNA analysis in computational biology [5].
The generic definition of clustering is usually redefined depending on the type of data to be clustered and the clustering objective. Different and scalable clustering algorithms have been proposed in [29, 31]. In [28], the authors propose a clustering technique that uses a distance function which combines both position and velocity differences, they employ the k-center clustering algorithm [11] for histogram construction. In [8], the authors represent a trajectory as a list of (MBB) minimal bounding boxes. MBB represents an interval bounded by limits of time and location. They use k-means algorithm for clustering trajectories. Nevertheless, they evaluate their work based on a very small synthetic data of 10 mobile objects making their evaluation less realistic. Another interesting clustering technique is clustering according to network distance which is more realistic [20], in this paper the authors proposed variants of partitioning, density based, and hierarchical methods that apply dominant clustering paradigms on the network-based clustering problem. They defined the problem of clustering objects based on the network distance, and proposed algorithms for three different clustering paradigms, i.e., k-medoids for k-partitioning, E-link for density-based, and single-link for hierarchical clustering. Their algorithms are designed to avoid unnecessary distance computations and redundant accesses of the network components through exploiting the properties of the network. In [35], the authors study the problem of clustering moving objects using an efficient technique that employs the idea of Micro Moving Clusters (MMC) to catch interesting spatio-temporal regularities of moving object. MMC were designed to denote a group of similar objects both at current time and at near future time. Each MMC maintains a bounding box for the moving objects contained in it whose size grows over time. Trajectory clustering problem was also investigated in [22]. In this work, a clustering technique called temporal focusing was introduced, the aim of the paper was to exploit the intrinsic semantics of the temporal dimension to improve the quality of trajectory clustering. In the paper the authors propose an adaptation of the density-based clustering algorithm to trajectory data based on a simple notion of distance between trajectories. In addition in [25], the authors used clustering moving objects to optimize the continuous spatio-temporal query execution. The moving cluster is represented by a circle in their algorithms. In [16], authors proposed a unified framework for Clustering Moving Objects in spatial Networks (CMON). In this framework they divided the clustering process into the continuous maintenance of cluster blocks (CBs) and the periodical construction of clusters with different criteria based on CBs. A CB groups a set of

objects on a road segment in close proximity to each other at present and in the near future. In [17], authors worked on discovering moving clusters from historical trajectories of objects. Important events like collision among micro-clusters are also identified. The authors studied the historical trajectories of moving objects, and proposed algorithms that discover moving clusters. A moving cluster is defined as a sequence of spatial clusters that appear in consecutive snapshots of the object movements so that consecutive spatial clusters share a large number of common objects.

In this paper we propose a clustering technique that is close to the partition and group framework proposed in [15] which partitions a trajectory into a set of line segments, and then, groups similar line segments together into a cluster. The main contributions of our proposed work are:

- Our proposed clustering approaches are able to discover *common sub-trajectories* from a trajectory database. This property makes the approaches more accurate than techniques that cluster based on the whole trajectory behavior.
- The algorithms proposed use the idea of recapping to create a list of clusters that are visited by different trajectory's segments.
- We developed an approximate clustering algorithm that uses the visited cluster list to predict the future motion pattern of the trajectory.
- We experimentally show that our approximate clustering algorithm provides better running time along with acceptable error margin.

In the remainder of the paper we present our technical contributions and our proposed algorithms.

3.Problem Definition

Moving objects are defined as objects that change their location and/or shape over time [27, 33]. In general, moving objects are classified as moving points and moving regions. Moving points are those where the location is continuously changing over time but not the shape; like cars, buses, trains, plans, mobile phone users, etc. On the other hand, moving regions are those characterized by their shape changing over time; like hurricanes, forest fires, etc. In this paper we focus on *moving points*. The motion of a moving point is usually expressed by its *trajectory* that represents the path that the object follows during its motion over time. A moving object trajectory can be modeled in many different ways depending on the underlying space, i.e. Euclidean space or cellular space [14]. The basic difference between cellular space and Euclidean space is the fact that the cellular space is not based on any geometric representation; the location of a moving object is referenced by a cell identifier where the moving object is located. Thus it ignores the spatial features of the moving objects which are very essential in many applications. Nevertheless, it may be useful in some applications such as sensor network applications where each cell is a coverage area of a sensor. In [14] the authors represented the trajectory of a moving object by the sequence of cells that it visited. In this paper we consider the Euclidean space. We consider a trajectory to be a piece-wise linear function of time. Each linear piece of the trajectory is denoted as a *motion*. Each motion is then associated with a time interval over which the trajectory *segment* is defined, this in turn constitutes the basic building block of a trajectory. In this work, we focus on 2-dimensional moving points, we model time *T*, our third dimension, as a densely ordered domain of *time instants*.

Let t_1, $t_2 \in T$, $t_1 \leq t_2$, and "-∞", "∞" be two special symbols. A *time interval* can be expressed as one of the following forms that denotes a subset of *T*:

- $[t_1, t_2]$, which denotes the set $\{x \mid t_1 \leq x \leq t_2\}$,
- $(-\infty, t_1]$, which denotes the set $\{x \mid x \leq t_1\}$, or
- $[t_1, \infty)$, which denotes the set $\{x \mid t_1 \leq x\}$.

In the remainder of the paper we consider a trajectory as a sequence of line segments.

Definition 3.1 *A* motion *is a pair* (*A,B*) *that represents the linear function At + B, where t is a time instant. A* segment *is an association of a motion and a valid time interval represented as a triple* (*A,B,V*)*, where* (*A,B*) *are the motion vector parameters, and V is a valid time interval from T such that* $\forall t, t \in T$.
A moving object trajectory *is a finite sequence of segments such that:*

- *(time continuity) the union of time intervals of all segments is equal to the life time of a trajectory which is a single time interval, thus,* $\bigcup_{i=1}^{k} V_i$ *is a single time interval [9], and*
- *(motion continuity) let* $s_1 = (A_1, B_1, V_1)$ *and* $s_2 = (A_2, B_2, V_2)$ *be 2 consecutive segments. Then,* $V_1 \cap V_2 = [\tau, \tau]$*, and,* $A_1\tau + B_1 = A_2\tau + B_2$*. (i.e. the 2 segments intersect at* τ*).*

Let $\boldsymbol{T}$ *be the set of all trajectories.*
We assume the existence of an infinite set *O* of *object identifiers* (oids).

Definition 3.2 *A* moving object database *(or MOD) is a pair* (*O, T*) *where O is a finite subset of* $\mathcal{O}$*, and T is a mapping from O to* $\mathcal{T}$*.*

In this paper we focus on clustering moving object trajectories. The main target of the paper is to construct efficient algorithms to cluster trajectories in MOD. We address an important issue in clustering moving object, namely, how can we use the semantics of the temporal dimension of moving objects to achieve efficient trajectory clustering. The importance of this issue stems from the fact that moving objects might show very different behavior if we consider their entire life time although they may become very similar if we consider only a restricted sub-interval of their life time. This observation is common in monitoring the motion pattern of cars along road networks [22].
In order to achieve our target, we first need to define our notion of moving object trajectories similarity. Our similarity measure is based on the Euclidean distance between the locations of the objects at each time instant. The main assumption in our similarity measure is that objects are defined over the same time period. In general, Euclidean distance is the earliest and basic approach for fast similarity search between two trajectories through viewing them as vectors and using Euclidean distance between them to calculate the similarity distance. In the remainder of the paper, we employ the k-means clustering algorithm to group similar trajectory segments together into the same cluster. The main difference between traditional k-means and the proposed k-means is that the proposed algorithm is tailored for trajectories rather than points. Thus, given a set of 2-dimensional trajectories, we aim at grouping different trajectory segments into different clusters such that within-cluster distance is minimized. We first initialize each cluster centroid with a segment of a trajectory in MOD. This centroid is then updated as new segments are added to the cluster.

Definition 3.3 Given a cluster *C* with centroid (segment) *c* = (*X, Y,W*), where *X, Y* are motion vector parameters and *W* a time interval, and a trajectory segment *s* =(*M,N,P*).
The distance between *C* and *s* is defined as:

$$D(C,s) = \frac{\int_P (d(Xt+Y, Mt+N))}{P} dt \qquad (1)$$

Where $d(Xt+Y, Mt+N)$ is the Euclidean distance between *c, s* at time instant $t \in P$.

Using Equation (1), we determine the membership of different segments to different clusters during their life time. Similar segments are grouped together into the same cluster. Each cluster has a time interval associated with it that covers the life time of the segments in the cluster.

Definition 3.4 Given a cluster *C*, a *cluster time interval* $C|_t$ is the union of the time intervals of trajectory segments in *C*. After applying our clustering algorithm, we generate a linked list associated with each trajectory that shows the clusters visited by the object's trajectory throughout its life time.

Definition 3.5 Let *n* be a natural number, a *cluster linked list of a trajectory* is an ordered list $(C_1, C_2, \dots, C_n)$ such that:

- $\forall\ 1 \le i \le n$, C_i is a cluster,
- If $C|_t$ is a cluster with cluster interval *t*, and *s* = (*A,B, V*) is a trajectory segment. Then, $s \epsilon C_i$ iff $V \subseteq t\ \wedge D(C_i, s) \le D(C_j, s)$, $\forall 1 \le j \le n, i \ne j$.

The paper thus focuses on the problem of efficient clustering of similar trajectories that could match only during sub-interval(s) of their duration. The following section presents the clustering algorithm in details.

4.Clustering Technique

Modern geographic information systems do not only have to handle static information but also dynamically moving objects. Clustering algorithms for these moving objects provide new and helpful information for many applications, e.g. traffic jam detection and prediction. Clustering is a crucial and fundamental approach in data mining [12]. The role of data mining techniques such as clustering is to analyze already existing information in order to extract further knowledge and reveal behavioral patterns and trends that are useful in the decision making process. Similarity search techniques are usually classified based on the analysis objective. In this section we present our clustering algorithm Tk-means (Trajectory k-means). The algorithm relies on the basic concepts of the k-means clustering algorithm [21]. The k-means clustering algorithm is one of the simplest unsupervised learning algorithms that solves the well known clustering problem.

Our proposed Tk-means algorithm aims to partition a number of objects' trajectories into *k* clusters such that each segment of the object's trajectory finally belongs to the cluster with the nearest mean (e.g. clusters are represented by a mean value which is the Euclidean centroid of the segments in it). Therefore, as traditional k-means classifies a given data set into a certain number of clusters (assume k clusters) fixed a priori, the Tk-means algorithm extends this idea to create clusters that group similar trajectories' segments. The main feature of the Tk-means algorithm is that a trajectory can belong to different clusters during different time intervals (i.e. different segments can belong to different clusters).

The algorithm proceeds as follows: Given a cluster $C|_t$ with centroid *c* that is defined during the cluster time interval $t = [t_1, t_2]$ and a query trajectory τ with a trajectory segment *s = (x,y,v)* such that $v \subseteq t$. Our goal is to insert the segments of τ into the most similar cluster. For each cluster, we first compute the Euclidean distance *D*(*C, s*) using Equation (1). Then, we insert the query trajectory segment into the most similar clusters (i.e. smallest *D*). The same procedure is repeated for the remaining segments constituting τ. Finally, we obtain a cluster linked list of τ as defined in Definition 3.5 that simply represents the clusters that show similar motion pattern to the query trajectory's segments during their time intervals.
Tk-means algorithm thus proceeds as shown in Figure 1.

Algorithm:Tk-means

Input: C_{List}: a list of k-clusters, τ a query trajectory
Output: A 2-dimensional matrix L=[τ, cluster linked list]
/* Let n be the number of segments in τ */
for $(1 \leqslant i \leqslant n)$ **do**
    Segment = $\tau.segment[i]$;
    C_{id}=Compute_Similarity(Segment,C_{List});
    $L.Add(\tau, C_{id})$
end
return L;

Figure 1: Tk-means Algorithm

The algorithm first calls Compute_Simlairity procedure that simply computes the Euclidean distance between the query trajectory segment *s* and the cluster *C*, given that the segment's time interval is contained in the cluster time interval. Let *CList* be the list of existing clusters. The procedure proceeds as shown in Figure 2.

Procedure:`Compute_Simlairity`(s, C_{List})

```
Output: C_id: identifier of cluster with maximum similarity score
min_dist = ∞;
for (1 ⩽ i ⩽ C_List.Length) do
    if (s.interval ⊆ C_List[i].interval) then
        Dist=Compute_Euclidean_Distance(s, C_List[i]);
        if Dist ⩽ min_dist then
            min_dist = Dist;
            C_id = i;
        end
    end
end
return C_id;
```

Figure 2: Similarity distance computation

The Compute_Euclidean_Distance step in Figure 2 simply applies the Euclidean distance function shown in Equation 1. We initially apply the algorithm in Figure 1 to all trajectories in MOD as a pre-processing step for building our clusters. So initially the k-clusters are empty and as we examine the different segments we update the centroid of the clusters to keep within cluster distance minimized. Then, we query the generated clusters to find similar trajectories.
Inspired by the fact that time is an important factor in measuring the performance of querying moving objects. We propose two approaches for evaluating the best match for a query trajectory. The first approach is an exact approach that compares query trajectory and trajectories in pre-generated clusters to choose the most similar one through testing all trajectory segments of both trajectories. The second approach is an approximate approach that compares only part of query trajectory against the pre-processed clusters and then predicts the future clusters that the query trajectory is expected to visit based on the motion pattern of similar trajectories. The exact algorithm proceeds as shown in Figure 1. The approximate is shown in Figure 3.

The approximate technique works as follows:
Let *f* be an approximation factor that represents the fraction of query trajectory segments that will be exactly clustered (for example, f =1/2 means that half the segments will be exactly clustered and half will be approximated). The algorithm proceeds as follows. Given a query trajectory τ, an approximation factor *f*, and a MOD D. We first cluster the first *f* segments of τ using the exact Tk-means algorithm in Figure 1. Then, we compare the resulting cluster linked list with the cluster linked list of the trajectories in MOD. We select the trajectory that shows the maximum number of matches between the 2 cluster linked lists for their first *f* segments. We finally use the cluster linked list of the resulting most similar trajectory as an estimate for the cluster linked list of the query trajectory for the remaining unexamined segments.
The details of the algorithm are shown in Figure 3.
The approximate algorithm uses an input parameter value that represents an approximation factor *f* that defines the number of segments that will be used to predict the future motion pattern of the query object. Based on this variable, the algorithm proceeds by computing the Euclidean distance

between the specified number of segments of the query trajectory against overlapping clusters. The most similar trajectory whose linked list of visited clusters shows maximum number of matches with query trajectory is then selected as the most similar trajectory. We finally use the remaining part of the retrieved trajectory's linked list as an expectation for the future motion pattern of query trajectory during the remaining time. Although a percentage of error is expected, the experiments presented in the following section show that the accuracy is still acceptable and the query performance is improved.

Algorithm:Approximate Approach

Input: τ:Query Trajectory, f: approximation factor, D: a MOD, C_{List}: list of k-clusters
Output: M: Most similar cluster linked list
for $(1 \leqslant i \leqslant f)$ **do**
  $\tau'.segment[i] = \tau.segment[i]$;
end
/* Let L be the cluster linked list of τ' */
L= Tk-means(C_{List}, τ');
count =0;
for $(1 \leqslant j \leqslant len(\text{MOD}))$ **do**
  new_count = len(L ∩ MOD[j].cluster linked list);
  if $(new\_count > count)$ **then**
    count = new_count;
    o_{id} − j;
  end
end
return MOD$[o_{id}]$.*cluster linked list*;

Figure 3: Approximate Approach

Example 4.1 Consider the motion of 4 buses represented by the trajectories of objects T_1, T_2, T_3, T_4 shown in Figure 4 Using the exact Tk-means algorithms, with number of clusters = 4, denoted as C_1, C_2, C_3, C_4, we get the clustering result as in Figure 5.
Assume a query trajectory Q as shown in Figure 4, using the approximation approach with an approximation factor f = 2. We first execute the exact approach on 50% of the trajectory segments, we first get, $Q.s_1, Q.s_2$ in C_1, and $Q.s_3$ in C_2. Next, we apply the approximation approach to predict the remaining clusters visited by Q by choosing the closest trajectory's linked list to complete Q list of clusters, we get T_2 as a candidate. Finally, we obtain the complete approximate cluster list for Q as:C_1, C_2, C_3.

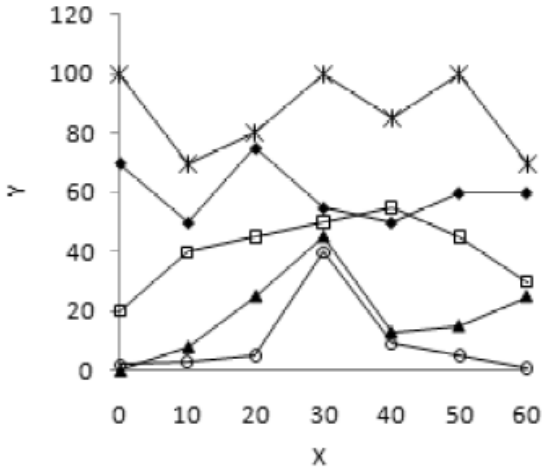


Figure 4: Four Buses Trajectories

Cluster[Time Interval]	Segments in clusters							
$C_1[t_1,t_2[$	$T_1.S_1$	$T_1.S_2$	$T_2.S_1$	$T_2.S_2$				
$C_2[t_3,t_4[$	$T_2.S_3$	$T_2.S_4$	$T_3.S_3$	$T_3.S_4$				
$C_3[t_3,t_6[$	$T_1.S_3$	$T_1.S_4$	$T_3.S_5$	$T_3.S_6$	$T_4.S_3$	$T_4.S_4$		
$C_4[t_1,t_6[$	$T_1.S_5$	$T_1.S_6$	$T_3.S_1$	$T_3.S_2$	$T_4.S_1$	$T_4.S_2$	$T_4.S_5$	$T_4.S_6$

Figure 5: Tk-means Clustering Results

5.Experimental Evaluation

In this section we examine both the performance and accuracy of our proposed similarity measures. We conducted our experiments on both real and synthetic data sets. We used a real data set [1] that represents the movement of 1100 moving object in London during a one month period (July 2007) with a 10 sec sampling rate. This data set consists of 1100 trajectories and 94098 segments. For the performance experiment, we conducted our experiment on a larger synthetic data set generated by the Brinkoff generator that is commonly used for generating realistic moving objects [7]. Using the generator, we simulated two dimensional trajectories of vehicles on the road network in the city of San Francisco. In our experiments we evaluated our clustering techniques along two important metrics namely, performance and accuracy. We measure the performance in terms of query processing time. Our experiments were conducted using Windows

Vista with processor Intl Core2 Duo, 2.0 GHz CPU, 2MB L_2 cache,4GB DDR2, and 1000GB hard disk.

Our first experiment considers the exact Tk-means algorithm, the experiment tests the effect of changing the number of clusters k on query performance (i.e. running time). We conducted this experiment on both real and synthetic data set of 100000 trajectories. We vary k from 10 to 50. Figure 6 shows the increase in the running time of the algorithm with increasing number of clusters. The reason behind this observation is due to the increase in number of comparisons that need to be done to choose the right cluster(s) for each trajectory. Thus increasing the number of possible clusters increases the number of possible clusters for each trajectory segment, hence, more computations need to be performed to determine the most similar cluster.

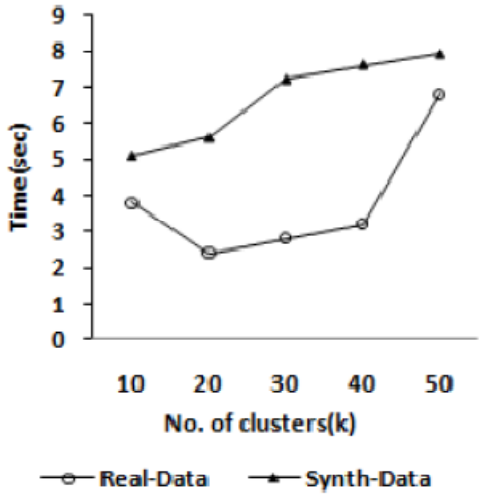


Figure 6: Effect of number of clusters on running time

The second experiment measures the query performance of the exact Tk-means algorithm for different trajectory lengths. We conducted this experiment on both real and synthetic data set. Figure 7 shows that as number of segments per trajectory increases, query running time increases as well. This result is logical consequence for increasing trajectory length, because more segments need to be examined and clustered and this in turn increases query time. Thus, as trajectory length increases, more computations per trajectory are required and the overall query processing time increases as well.

The next two experiments were conducted on synthetic data set. The third experiment examines the effect of changing the approximation factor f on the query accuracy. We change the approximation factor so that the exact to actual segments range from 1/5 to 1/2. We measure the effect of this change on the percentage of error in the predicted cluster linked list. Figure 8 shows that decreasing the approximation factor leads to an increase in percentage of error. This result is due to the fact that decreasing approximation factor f on the query accuracy. We change the approximation factor so that the exact to actual segments range from 1/5 to 1/2. We measure the effect of this change on the percentage of error in the predicted cluster linked list. Figure 8 shows

that decreasing the approximation factor leads to an increase in percentage of error. This result is due to the fact that decreasing approximation factor means decreasing number of segments exactly examined and clustered to help in choosing the most similar trajectory, thus percentage of error increases. Therefore, as the number of precisely clustered segments increases, a better predication can be achieved with less error. This follows from the fact that the remaining number of segments that need to be approximated decreases.

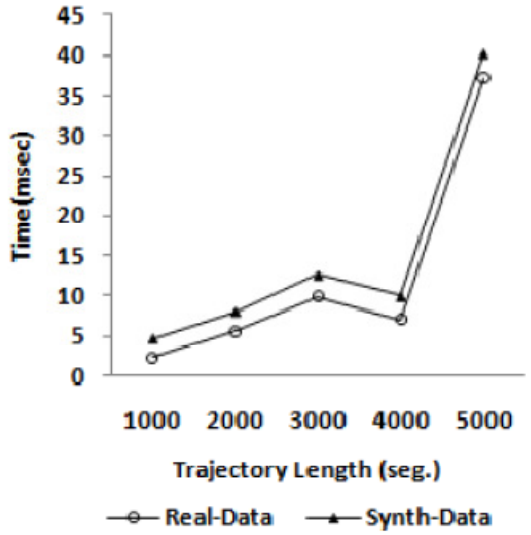


Figure 7: Exact Tk-means performance

The next experiment measures the accuracy of the approximate Tk-means algorithm. The experiment measures the percentage of error between the cluster linked lists of query trajectory resulting from applying the exact and the approximate algorithms. Figure 9 shows the results of the experiment. In this experiment we separately selected from MOD a group of trajectories of different length (from 1000 to 5000 segments), then applied Tk-means algorithm to generate k clusters, and generated the cluster linked lists visited by different trajectory segments in MOD as a preprocessing phase. Then we used the approximate algorithm to select the most similar trajectory to the query trajectory by examining the in first half if the segments (i.e $f = 1/2$). Finally, we match the two cluster linked lists to calculate the percentage of error. Figure 9 shows that the percentage of error decrease versus the increase in trajectory length, the intuition behind this result is that when trajectory length increases, search space increases and hence the chances for matches between two cluster linked lists increases as well. Thus, there is a better chance to find a similar cluster list (i.e. a similar trajectory).

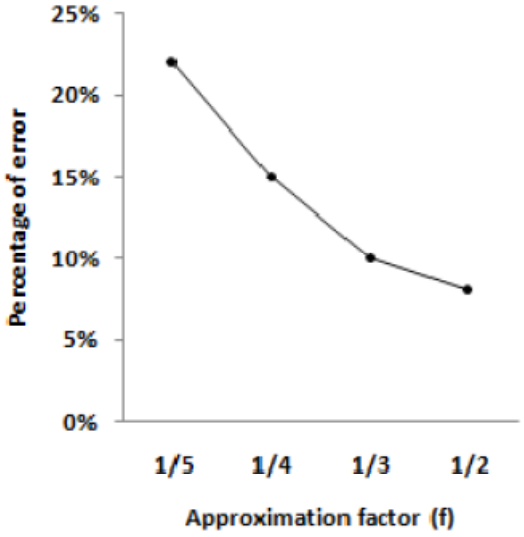


Figure 8: Effect of changing approximation factor on query accuracy

The last experiment compares the performance of the approximate algorithm to the exact algorithm. We measure the performance through comparing their query running time. We apply the approximate Tk-means algorithm with approximation factor $f = 1/2$ calculate the running time. The experiments show that query performance is highly improved (up to 70%). Figure 10 shows the increase of query time versus different trajectory length, and the performance gain achieved by applying the approximate technique on both real and synthetic data set. The performance improvement is expected to increase as we decrease the approximation factor (i.e. less segments need to be precisely clustered), this is due to the fact that less computation time will be required to cluster the segments.

From the above experiments we can conclude that in applications where retrieving *near* similar trajectory is sufficient, using the approximate technique is a reasonable compromise. On the other hand, when accuracy is crucial, using the exact algorithm will be the optimal choice. Thus, for example in intelligent transportation systems context, we are more interested in the number of objects occurring in a spatial region during a certain time interval to propose traffic alternatives. And since we are not interested in the specific object information (object-id) using the approximate approach with its acceptable tolerance would be a reasonable approach. Using the approximate algorithm we can obtain a fair estimate of the number of moving objects (vehicles) that followed the same path and consequently visited (or will visit) the same regions. Whereas, in other applications as detecting the exact objects that were involved in an accident, identifying the exactly similar trajectories along with their object ids enforces the use of the exact algorithm. Nevertheless, in either case, using the proposed clustering algorithms out performs exhaustive trajectory search.

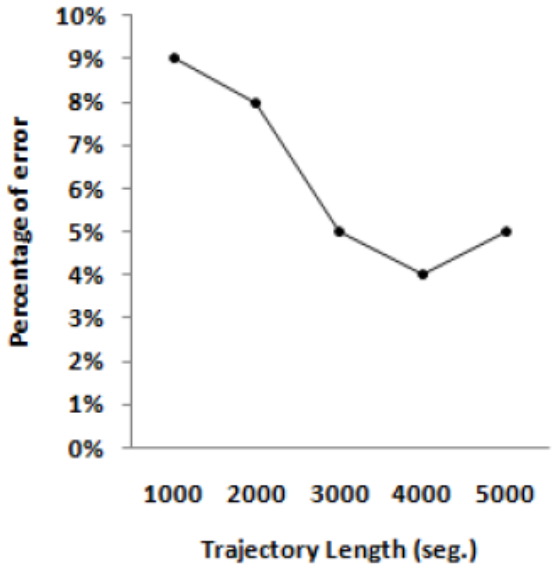
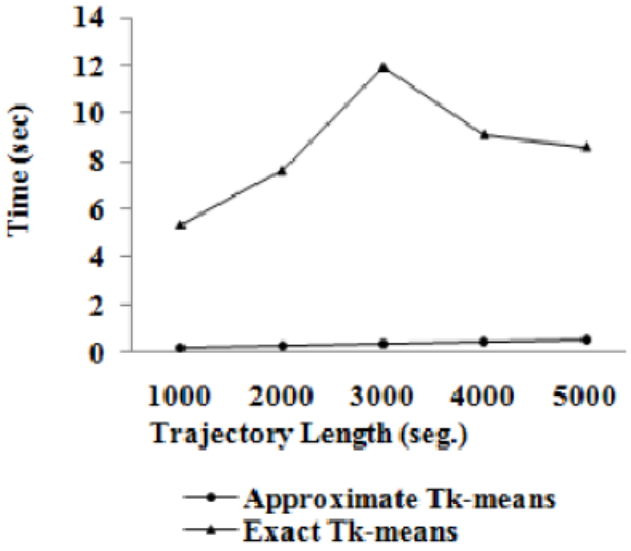


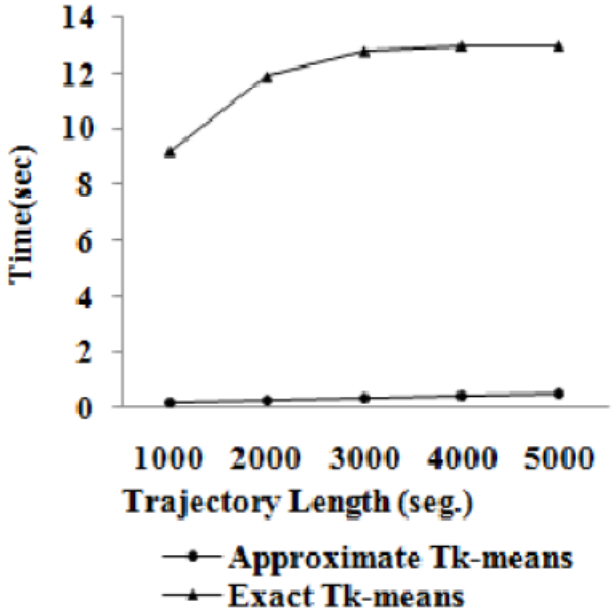
Figure 9: Approximate Tk-means accuracy experiment

6.Conclusions and Future Work

In this paper we present new clustering techniques that are based on traditional k-means algorithm. Basically, we propose a *time-based* clustering algorithm that tailors the k-means algorithm for trajectory data. We present two approaches: an exact and an approximate technique. Our experiments show that approximate technique enhances query performance up to 70% with a reasonable percentage of error. For future work, we aim to examine the performance of the proposed algorithms on larger data sets where index structures need to be considered. We also plan to examine the effect of different distance measures on our similarity search. We also think that other data mining techniques can be also integrated with moving objects databases for efficient discovery of motion patterns. Besides, for future work we aim to extend the work by employing data warehousing techniques for efficient decision making based on historical location information.



(a) Real data set



(b) Synthetic data set

Figure 10: Query processing time

References

[1] http://www.ecourier.co.uk, February 2010.
[2] Annual number of vehicle rollover crashes by rsc systems, 20012005.
http://www.itsbenefits.its.dot.gov/its/benecost.nsf/SummID/B2009-00607, April 2010.
[3] Fatal crashes in usa, 2008. http://www-fars.nhtsa.dot.gov/Crashes/
CrashesLocation.aspx, February 2010.
[4] Intelligent transportation systems society of canada.
http://www.itscanada.ca/english/aboutits.htm, February 2010.
[5] Amir Ben-Dor and Zohar Yakhini. Clustering gene expression patterns. In *RECOMB '99: Proceedings of the third annual international conference on Computational molecular biology*, pages 33--42, New York, NY, USA, 1999.
[6] Pavel Berkhin. Survey of clustering data mining techniques. Technical report, Accrue Software, San Jose, CA, 2002.
[7] T. Brinkhoff. Generating traffic data. *IEEE Computer Society Technical Committee on Data Engineering*, pages 19--25, 2003.
[8] Sigal Elnekave, Mark Last, and Oded Maimon. Measuring similarity between trajectories of mobile objects. In *Applied Pattern Recognition*, pages 101--128, 2008.
[9] Allen James F. Maintaining knowledge about temporal intervals. *Communications of the ACM*, 26(11):832--843, 1983.
[10] Gereon Frahling and Christian Sohler. A fast k-means implementation using coresets. *Proceedings of the twenty-second annual symposium on Computational geometry (SoCG)*, 2006.
[11] Gonzalez. Clustering to minimize the maximum intercluster distance. *Theoretical Computer Science*, 38:293--306, 1985.
[12] Jiawei Han and Micheline Kamber. *Data Mining: Concepts and Techniques*. Morgan Kaufmann, 1st edition, September 2000.
[13] Jeffrey Heer, Ed H. Chi, and H. Chi. Identification of web user traffic composition using multi-modal clustering and information scent. In *Proc. of the Workshop on Web Mining, SIAM Conference on Data Mining*, pages 51--58, 2000.
[14] Kang Hye-Young, Joon-Seok, and Ki-Joune. Similarity measures for trajectory of moving objects in cellular space. In *SAC '09: Proceedings of the 2009 ACM symposium on Applied Computing*, pages 1325--1330, New York, USA, 2009.
[15] Lee Jae-Gil, Han Jiawei, and Whang Kyu-Young. Trajectory clustering: a partition-and-group framework. In *SIGMOD '07: Proceedings of the 2007 ACM SIGMOD international conference on Management of data*, pages 593--604, New York, USA, 2007.
[16] Chen Jidong, Lai Caifeng, Meng Xiaofeng, Xu Jianliang, and Hu Haibo. Clustering moving objects in spatial networks. In *DASFAA'07: Proceedings of the 12th international conference on Database systems for advanced applications*, pages 611--623, Berlin, Heidelberg, 2007.
[17] Panos Kalnis, Nikos Mamoulis, and Spiridon Bakiras. On discovering moving clusters in spatio-temporal data. In *SSTD*, pages 364—381, 2005.
[18] Tapas Kanungo, Nathan S. Netanyahu, and Angela Y. Wu. An efficient k-means clustering algorithm: Analysis and implementation. *IEEE TRANSACTIONS ON PATTERN ANALYSIS AND MACHINE INTELLIGENCE*, 24(7):881--892, 2002.
[19] Massart Desire L. and Kaufman Leonard. *The interpretation of analytical chemical data by the use of cluster analysis,* Wiley, New York, 1983.

[20] Yiu Man Lung and Mamoulis Nikos. Clustering objects on a spatial network. In *SIGMOD '04: Proceedings of the 2004 ACM SIGMOD international conference on Management of data*, pages 443--454, New York, NY, USA, 2004.

[21] J. B. Macqueen. Some methods of classification and analysis of multivariate observations. *Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability*, pages 281--297, 1967.

[22] Dino Pedreschi Margherita D'Auria, Mirco Nanni. Prediction time-focused density-based clustering of trajectories of moving objects. In *SIGMOD'04: Proceedings of the 2004 ACM SIGMOD international conference on Management of data*, New York, NY, USA, 2005.

[23] Hoda Mokhtar and Jianwen Su. A query language for moving object trajectories. In *SSDBM: International Conference on Scientific and Statistical Database Management*, pages 173--182, Santa Barbara, California, June 27-29 2005.

[24] Hoda M. O. Mokhtar and Jianwen Su. Universal trajectory queries for moving object databases. In *IEEE Int. Conference on Mobile Data Management (MDM'04)*, pages 133--144, Berkeley, California, January 19 - 22 2004.

[25] Rimma V. Nehme and Elke A. Rundensteiner. Scuba: Scalable cluster-based algorithm for evaluating continuous spatio-temporal queries on moving objects. In *EDBT*, pages 1001--1019, 2006.

[26] Lloyd Stuart P. Least square quantization in PCM. *IEEE Transactions on Information Theory*, 28(2):129--137, 1982.

[27] Sistla A. Prasad, Wolfson Ouri, Chamberlain Sam, and Dao Son. Modeling and querying moving objects. In *ICDE '97: Proceedings of the Thirteenth International Conference on Data Engineering*, pages 422--432, Washington, DC, USA, 1997.

[28] Zhang Qing and Lin Xuemin. Clustering moving objects for spatio-temporal selectivity estimation. In *ADC '04: Proceedings of the 15th Australasian database conference*, pages 123--130, Darlinghurst, Australia, Australia, 2004.

[29] Iwerks Glenn S., Samet Hanan, and Smith Ken. Continuous k-nearest neighbor queries for continuously moving points with updates. In *VLDB:Proceedings of the 29th international conference on Very large data bases*, pages 512—523, 2003.

[30] Michael Steinbach, George Karypis, and Vipin Kumar. A comparison of document clustering techniques. In Marko Grobelnik, Dunja Mladenic, and Natasa Milic-Frayling, editors, *KDD-2000 Workshop on Text Mining*, pages 109--111, Boston, MA, 2000.

[31] LiuWenting, Wang Zhijian, and Feng Jun. Continuous clustering of moving objects in spatial networks. In *KES: Proceedings of the 12th international conference on Knowledge-Based Intelligent Information and Engineering Systems, Part II*, pages 543--550, Berlin, Heidelberg, 2008.

[32] O.Wolfson, S. Chamberlain, S. Dao, and L. Jiang. Location management in moving objects databases. In *Proc. the Second International Workshop on Satellite-Based Information Services (WOSBIS'97)*, Budapest, Hungary, October 1997.

[33] Ouri Wolfson, Bo Xu, Sam Chamberlain, and Liqin Jiang. Moving objects databases: Issues and solutions. In *Statistical and Scientific Database Management*, pages 111--122, 1998.

[34] Xiaowei Xu, Martin Ester, Hans-Peter Kriegel, and JÄorg Sander. A distribution-based clustering algorithm for mining in large spatial databases. In *ICDE '98: Proceedings of the Fourteenth International Conference on Data Engineering*, pages 324--331, Washington, DC, USA, 1998.

[35] Li Yifan, Han Jiawei, and Yang Jiong. Clustering moving objects. In *KDD'04: Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 617--622, New York, NY, USA, 2004.