Spatio-Temporal Similarity of Network Constrained Moving Object Trajectories

Many a person has held close, throughout their entire lives, two friends that always remained strange to one another, because one of them attracted by virtue of similarity, the other by difference

Emil Ludwing (1881-1948)

3.1 Introduction

Trajectory database management has emerged due to the profusion of mobile devices and positioning technologies like GPS or recently the RFID (Radio Frequency Identification). Intelligent Transportation Systems [133 Wolfson 2010], have been developed allowing better monitoring and control of traffic in order to optimize traffic flow. A Trajectory Database (TD) consists of objects whose location changes over time (e.g. moving humans or vehicles). The concept of TD has become increasingly important and has posed great challenges to the data mining community [41 Giannotti 2007] due the integration of wireless communications and positioning technologies. The support of efficient trajectory similarity techniques is indisputably very important for the quality of data analysis tasks in TD. Moving objects commonly display repeated patterns of movement and the understanding of such pattern is often highly useful for planning, control and other such management activities.

The challenge [65 Tiakas 2008] is to express trajectory similarity by respecting network constraints, which is also a strong motivation for the following real and practical applications in field of transportation network and security Informatics.
i. By identifying similar trajectories, effective data mining techniques (e.g., clustering) can be applied to discover useful patterns. For example, a dense cluster is an indication of traffic jam situation which necessitates expansion of road in future enforcing for using alternative routes/restricted entry for some category during peak hours.

ii. Trajectory similarity can also help in several road network applications such as, routing applications which support historical trajectories, logistic applications, city emergency handling, drive guiding systems, flow analysis, etc. In such applications, efficient indexing and query processing techniques are required.

iii. Knowledge and prediction of the road traffic: Given that the number of vehicles increases on the roads, information related to the density on the network becomes very useful for many purposes such as navigation and trip planning.

iv. Car-sharing: One of the best ways of oil conservation is vehicle-sharing which also appears as a solution to traffic congestion in busy roads. Car-sharing appears as an interesting alternative due to environmental factors and for the security of the persons traveling since a quiet lot of problems in lonely travel could be avoided. Identifying the similar trajectories or even sub-trajectories becomes very useful for such types of applications.

v. Transport planning: At the moment of its creation, each road is planned for certain utilization. Reporting trajectory groups allows assessing the suitability of the road infrastructure useful for taxi or heavy vehicles services in major cities.

vi. As dark web is the concept of terrorist web site activities, trajectory can be treated as the path formed by series of web clicks. Trajectory similarity of moving objects resembles path similarity of user click-streams in the area of web usage mining. By analyzing the URL path of each user, it will be able to determine paths that are very similar, and therefore effective caching strategies can be applied.
In particular trajectory similarity problem has specific applications in the area of security informatics as discussed below, which has been identified as an additional extension of the concept discussed in section 2.5, Chapter 2, which deals with the concepts of automatic security alarming schemes\[106\]Sajimon2008. The proposal suggests in giving security alarm trigger to vehicles entering into an emergency (critical or classified) area. There is no direction further to these vehicles about how to go further or to deviate to which direction or to stop the movement at all. To give better suggestion to the triggered vehicles the system has to know the traffic patterns of the alternate routes. Finding similar trajectories in these routes at specified time period will give traffic patterns and the system can automatically suggest the possible way to go further.

3.2. Related Works

During the last decade there has been a lot of research work in the literature regarding trajectory similarity search. Most of the existing approaches so far are mainly inspired by the time series analysis domain and propose generic similarity metrics for 1D data \[49\]Agrawal1993, \[50\]Korn1997, \[51\]Chan1999. Other approaches deal with some basic trajectory features \[52\]Vlachos2002, \[53\]Vlachos2002, \[54\]Chen2004] such as the different sampling rates on trajectories, the different speeds of moving objects, the possible outliers that might be introduced due to an anomaly in data collection procedure, the different scaling factors, the different trajectory lengths, etc. The common characteristic of previous works is that they are interested in the movement shape of the trajectories, which are usually considered as 2D or 3D time series data. In other words, what is important in measuring the similarity between two trajectories is just the sequences of the sampled positions. This means that the temporal dimension is ignored, leaving the time recordings out of the knowledge discovery process. In real world applications trajectories are represented by finite sequences of time-referenced locations\[155\]Yoon2008. Such sequences may result from time-based (e.g. every 30 seconds), change-based (e.g. when the location of an entity deviates from the previous one by a given threshold), location-based (e.g. when a moving object is close to a sensor), event-based
recording (e.g. when a user requests for localization), or even various combinations of these basic approaches [56Andrienko2007]. A different perspective is required therefore, capable of coping with real world application scenarios. In addition to the above, Trajectory Database introduce temporal issues related with derived parameters of motion, such as speed and direction[156Pelekis2007].

Regarding the works related to the similarity of moving objects trajectories considering temporal dimension, we first mention those in the free moving trajectory context and then for constrained trajectories. In [57Yanagisawa2003] the authors, focused on the extraction of the individual moving patterns of each object from the trajectories considering both time and location. Their approach uses the shape similarity between lines to retrieve required objects. The methods on the similarity of sub-trajectories[58Shim2003] proposed a distance 'K-Warping' algorithm. We also find similar approaches in [59Sakurai2005], and [55Chen2005] presenting an investigation for analysis of spatio-temporal trajectories for moving objects where data contain a great amount of outliers. Therefore, they propose the use of a non metric distance function that is based on the Longest Common Sub Sequences (LCSS) algorithm in conjunction with a Sigmoid Matching function to increase the performance of Euclidean and Time Warping Distance. There is a proposal[60Zeinalipour2005] on distributed spatio-temporal similarity search based on the LCSS distance measure and propose two new algorithms offering good performance.

All these methods are inappropriate for similarity calculation on road networks since they use the Euclidian distance as a basis rather than the real distance on the road network. Therefore, it is more ideal to find the distance between two points on road networks defined along road sectors using network distance than Euclidean distance as it represents the real-world situation.

This point has motivated the proposition of [61Hwang2005, 62Hwang2006] that were the first to propose a similarity measure based on the spatio-temporal distance
between two trajectories using the network distance. The algorithm of similar trajectory search consists of two steps: a filtering phase based on the spatial similarity on the road network, and a refinement phase for discovering similar trajectories based on temporal distance. The work of Hwang(2005) seems to be the first research work studying trajectory similarity on networks where the authors propose a simple similarity measure based on Points Of Interest (POI) and Time Of Interest (TOI). They retrieve similar trajectories on road network space and not in Euclidean space. The above work does not give the percentage of the similarity measure when some trajectories do not pass through POI’s and the research work in this chapter has made an attempt to address this issue. In [130 Leticia2008, 157 Leticia2009] the authors study how moving object data analysis can benefit from replacing raw trajectory by a sequence of stops and moves. Here Place of Interest has been taken as a stop of the object’s trajectory. In between stops, a trajectory has moves. There are research works [63 Tiakas2006, 64 Chang2007] that uses spatio-temporal distance, based on the road network, in their algorithm of similar trajectory search. One of the recent works in trajectory similarity problem for network constrained objects can be found in [65 Tiakas2009] which introduces new similarity measures that can be employed to express similarity between two trajectories that do not necessarily share any common sub-path. They define new similarity measures based on spatial and temporal characteristics of trajectories, such that the notion of similarity in space and time is well expressed and moreover they satisfy the metric properties. In addition, it demonstrates the similarity range queries in trajectories that are efficiently supported by utilizing metric-based access methods, such as M-trees[45 Caetano2000].

3.3 A note on Similarity Search

The basic task of a similarity search [49 Agrawal1993] is to find objects in the database which are similar to a query object. This section will start with a discussion on the different aspects of similarity search. The first important aspect is the concept of similarity itself. A formal concept of similarity is a necessary basis for any application in this field. In the literature, two concepts of similarity have
been applied successfully which are the feature vector approach \cite{Korn1998} and the concept of distance-based similarity \cite{Ciaccia1998}. The following sub sections briefly discuss these two concepts.

### 3.3.1 The Feature Vector Approach

In this very common approach, a domain expert chooses a set of single-valued object features that describe an object from that application domain. These features form a so-called feature vector space and objects are represented as points in this space. This is done by creating a feature vector for each object using the feature values of the specific object. Then, the similarity or dissimilarity of two objects is defined based on their distance in the feature space. The feature vector approach for similarity, whose idea is illustrated in Figure 3.1, has been successfully applied in several application domains like medical imaging \cite{Korn1998} and protein similarity \cite{Bobbie2008}.

Several measures are available to determine the distance between two points in the feature space like Manhattan distance and Euclidean distance \cite{Hui2008}. Most often it is a variant of the Lp-norms \cite{lei2004}, which are explained below.

The length of a vector $x = (x_1, x_2, \ldots, x_n)$ is usually given by the Euclidian norm.

$$\|x\|_2 = \left( x_1^2 + x_2^2 + \cdots + x_n^2 \right)^{1/2}$$

\[Figure 3.1\] Similarity based on the feature vector approach.
But this is by no means the only way of defining length. If \( p \) is a real number, \( p \geq 1 \), define the \( l_p \) norm of \( x \) by

\[
\|x\|_p = \left( |x_1|^p + |x_2|^p + \cdots + |x_n|^p \right)^{1/p}
\]

Now the distance between two vectors \( x \) and \( y \) is defined by the function

\[
d_p(x, y) = \left( \sum_{i=1}^{n} |x_i - y_i|^p \right)^{1/p}
\]

The above distance measure is a metric and is known as \( L_p \) distance.

**Definition 3.1:** (\( L_p \)-norms) Let there be two vectors \( x = (x_1, \ldots, x_n) \), \( x \in \mathbb{R}^n \), and \( y = (y_1, \ldots, y_n) \), \( y \in \mathbb{R}^n \). The \( L_p \)-norms between \( x \) and \( y \) are defined as:

\[
L_p(x, y) = \left( \sum_{i=1}^{n} |x_i - y_i|^p \right)^{1/p}
\]

For \( p = 1 \) and \( p = 2 \) the \( L_p \)-norms are the well-known Manhattan distance and the Euclidean distance, respectively. Most often, the Euclidean distance is used in similarity search applications based on the feature vector approach.

### 3.3.2 Distance-Based Similarity

The distance-based similarity model \(^{154}\)Ciaccia1998 is a generalization of the feature vector model discussed above. Here a distance measure for the data objects themselves is defined, instead of transforming the data objects into a feature space and then measuring the distance of the objects in the feature space. Therefore no feature extraction and no choice of features is necessary in finding the similarity. Also, it can take all object properties into account. The concept of distance-based similarity is illustrated in Figure 3.2.

Since the complete object has to be managed in this measure, it has higher complexity though as a whole the measure is having increased flexibility. Therefore the computational complexity of the similarity measure has to be chosen carefully.
to ensure efficiency. While this very high flexibility may be useful in certain special applications, it usually makes sense to impose some restrictions on the similarity distance measure to ensure that efficient query processing is possible. These restrictions can be summarized by demanding the measure to satisfy the four metric properties [13Chavez2001] as shown below:

i. Positivity Property: \( \forall x, y : d_{sim}(x, y) \geq 0 \)

ii. Definiteness Property: \( \forall x, y : d_{sim}(x, y) = 0 \), for \( x = y \)

iii. Symmetry Property: \( \forall x, y : d_{sim}(x, y) = d_{sim}(y, x) \)

iv. Triangular inequality Property: \( \forall x, y, z : d_{sim}(x, z) \leq d_{sim}(x, y) + d_{sim}(y, z) \)

The positivity and definiteness property of the similarity distance reflect the idea that a low distance means high similarity and therefore, identical objects should be assigned the lowest possible similarity distance. The concept that the objects are mutually similar is expressed by the symmetry Property. The triangular inequality property ensures that no object can be very similar to two very dissimilar objects at the same time. The requirement of these metric properties on a similarity distance has the advantage that efficient access methods and search algorithms can be applied. The similarity measure in this chapter is based on feature vector space as the moving object is considered as point object and it’s shape or extend is not at all important.
3.4 Efficient Similarity Search

The huge size of modern database and the complexity of the similarity searching are making efficiency as an important issue for any similarity search applications[163Bohm2002]. In this section, we will present two techniques to speed up the query processing in similarity search applications. The two techniques, the use of index structures, and the use of a multi-step query processing architecture, are not meant to be mutually exclusive. Instead, they can both be applied in parallel or at different stages of the query processing. The similarity methods proposed in this thesis are based on multi-step query processing architecture.

3.4.1 Index Structures

The use of standard index structures like ISAM, Hash tables, B-tree etc are being used to improve query processing efficiency in database systems [161Codd199,162Elmasri2000]. Numerous other index like structures like R-tree, have been proposed for many different data types and applications. The two type of index structures which are important for similarity search in structured data are index structures for high-dimensional vector spaces [163Bohm2002] and for metric spaces. The first category is being used when the similarity feature is based on vector space approach, and the second is useful when the distance based similarity approach is followed. But, for the distance based similarity model, where the similarity measure is often complex, speeding up the query processing is essential. An overview over existing approaches for indexing metric spaces is given in [43Chavez 2001] and its variants M-tree [44Paolo1997], Slim-tree [45Caetano2000] are specifically designed to allow dynamic updates.

3.4.2 Multi-step Query Processing

In similarity search applications, the complexity of the similarity distance measure is often a problem for efficient query processing. Hence to reduce the number of distance calculations the reduction of the search space is necessary. One way to exclude unnecessary rows from scanning, which reduces the number of necessary
similarity distance calculations is to employ a multi-step query processing architecture \cite{Papadopoulos1999,Seidl1998}. As shown in Figure 3.3, a multi-step query processing is performed in two or more steps. The filter step returns a number of candidate objects from the database which actually eliminate all unwanted objects based on a similarity measure. For those candidate objects, the exact similarity distance is then determined in the refinement step and the objects fulfilling the query predicate are reported as the result. To reduce the overall search time, the filter step has to fulfill certain constraints. First, it is essential that the filter predicate is considerably easier to evaluate than the exact similarity measure. Second, a substantial part of the database objects must be filtered out in the first step. Obviously, it depends on the complexity of the similarity measure used for filtering.

![Figure 3.3. Schema of a multi-step query processing architecture](image)

As we try to eliminate unwanted objects in the filter step to achieve completeness, caution should be taken to ensure that eligible candidates are not filtered out during the process. Thus completeness in this context means that no false drops occur during the filter step. Available similarity search algorithms guarantee completeness, if the distance functions in the filter step fulfill the lower-bounding property.

Definition 3.1 (lower-bounding property) : For any two objects p and q, a lower-bounding distance function $d_{lb}(p, q)$ in the filter step has to return a value that is not larger than the exact distance between p and q,

$$\forall p, q : d_{lb}(p, q) \leq d_e(p, q).$$

Such a lower-bounding distance function will safely filter out, all database objects which have a filter distance larger than the current query range.

A multi-step query processing methodology is used for range queries \cite{Agrawal1993} and for k-nearest-neighbor search \cite{Korn1998,Seidl1998} in
the sense that the minimal number of exact distance calculations is performed
during query processing.

This thesis has employed the multi-step query processing approach in order to
ensure efficient similarity search process.

3.5 Requirements for Similarity Measures

In the preceding sections, we discuss several aspects of similarity search
applications. The following are a few requirements which a similarity measure for
structured data should fulfill [16]Bohm2002].

i. The similarity measure should be adaptable to the needs of specific
applications and to the needs of the users.

ii. It is necessary to provide an explanation of the similarity distance value
between two data objects, to allow the user a purposeful and easy
adaptation of the parameters of the similarity distance measure.

iii. The final requirement is concerned with the efficiency of the query
processing in similarity search applications. First, the measure should be
of moderate time complexity, since it has to be evaluated often,
especially in today’s large and fast growing databases.

3.6 Trajectory Similarity in Spatial Networks

The common characteristic of the aforementioned approaches and research works is
that objects are allowed to move freely in 2D or 3D space, without any motion
restrictions. However, in a large number of applications, objects are allowed to
move only on pre-defined paths of an underlying network, resulting in constraint
motion [1]Guting2005]. For example, vehicles in a city can only move on road
segments. In such a case, the Euclidean distance between two moving objects does
not reflect their real distance. Objects moving in a spatial network follow specific
paths determined by the graph topology and therefore arbitrary motion is
prohibited. This means that two trajectories which are similar regarding the Euclidean distance may be dissimilar when the network distance is considered. The majority of existing methods for trajectory similarity assume that objects can move anywhere in the underlying space and therefore do not support motion constraints. Most moving objects are in road network space rather than in Euclidean space.

3.6.1 Significance of Distance Computation

In answering moving object based queries like querying nearest neighbors, analyzing the history information of moving objects the shortest path between moving objects is very much important. As an example, let us analyze the queries “where was the nearest ambulance to the place of the accident at that time” and “find all vehicles whose distances are within 2 kilometers to the gas-station between 9:00am and 12:00am on last Monday”. In order to answer these queries, one has to analyze the position and time details of the moving objects trajectories, as well as the underlying network information to compute network distances. The distance computation is the most important step in processing this kind of query. The efficiency of the distance function has direct impact over these queries.

Euclidian Distance and Network Distance

Distances in transportation research can be measured using geographic information systems (GIS) in three forms: Euclidean distance [159Hui2008], Network distance [61Hwang2005] and Manhattan distance [63Tiakas2006]. Manhattan distance is not commonly used in transportation research since it is generally meaningful only on a grid system, which holds strictly in few urban contexts. Euclidean distance is the aerial distance measured between origin and destination" as the crow flies". The network distance, which is the distance between origin and destination as shortest path measured along a transportation network, it is more realistic distance computation in a road network [66Miller2001]. These two kind of distance measures are illustrated in Figure 3.4.
The Euclidian distance computation is conventional and straightforward while the network based distance computation is much more complex, which involves the process of searching the shortest path after calculating distance between locations. Consider the figure 3.5, where there are four roads $r_1,r_2,r_3,r_4$ and three points $p_1, p_2$, and $p_3$ in the network. The Euclidean distance value between $p_2$ and $p_3$ is the length of the line directly connecting them. However, the network distance is the length of the shortest path along the route from $p_2$ to $p_3$ (we assume the network is bidirectional which means that there is no single route). So, the network distance and the Euclidean distance between $p_2$ and $p_3$ are different.
3.7 Finding Similar Trajectories on Road Networks

Most of the trajectory similarity methods consider only spatial similarity in measuring the similarity between moving object trajectories. For example, let us assume that the number of common locations visited by the trajectories is considered to be a spatial measure in finding similarity. Then if two trajectories pass through the same points at different time intervals on road networks, we understand by spatio-temporal intuition that they are not similar to each other. However, if we consider only spatial measure, the two trajectories are similar to each other. To address such issues we define spatial and temporal similarity based on road networks.

**Definition 3.1: Trajectory definition based on binary coded location.**

The dimension reduction of trajectory data using binary encoded location discussed in chapter 2 is used in defining a trajectory as follows

Let $T$ be a set of trajectories in a spatial network, in which each trajectory is represented as

$$T = ( (b_1,t_1),(b_2,t_2),(b_3,t_3), \ldots, (b_n,t_n) ),$$

where $n$ is the trajectory description length, $b_i$ denotes a location in binary string and $t_i$ is the time instance (expressed in time units, e.g. seconds) that the moving object reached node $b_i$, and $t_1 < t_i < t_m$, for each $1 < i < m$. It is assumed that moving from a node to another comes at a non-zero cost, since at least a small amount of time will be required for the transition.

It is difficult to search directly for similar trajectories from a number of trajectories on road networks as discussed in section 3.3.2, since the number of trajectories is huge. We follow multi-step query processing concept with a filtering step followed by refinement. Filtering is the process of elimination of unwanted trajectories and refinement is the process of tuning up the filtered candidates based on an accuracy threshold. We use spatial filtering followed by temporal refinement or temporal filtering followed by spatial refinement. We are interested in the movement of
objects through selected locations as Points Of Interest (POI) or selected times as Time Of Interest (TOI).

After performing the filtering step based on the spatial or temporal similarity, we need a refinement step which either based on temporal distance or spatial distance in order to search for similar trajectories. Here since our objective is for spatio-temporal similarity search, we propose the two step process for finding similar trajectories using a combined spatio-temporal measure which will be based on both POI and TOI. Thus we identify the following similarity types related to road network environment.

(i) Finding similar trajectories based on spatial filtering and temporal distance
(ii) Finding similar trajectories based on temporal filtering and spatial distance
(iii) Finding similar trajectories based on spatio-Temporal filtering and spatio-Temporal distance

3.7.1 Finding similar trajectories based on spatial filtering and temporal distance
Spatial filtering is considered by spatial similarity, which is based on Point of Interest (POI). POI set may be a set of junction points in a road network or location of some specific importance, and is usually decided by the user. This will be useful in finding out the movements of objects through known locations of interest, which may be terrorist locations, points of emergency, list of strategically important locations or famous tourist spots. Figure 3.6 shows the multi step process of finding similar trajectories by using the filtering step and the refinement step.

![Figure 3.6](image-url)
Definition 3.2. Spatial Similarity between Trajectories on road network space.
Suppose that \( P \) is a set of POI’s on a given road network. Then spatial similarity between two trajectories \( TR_a \) and \( TR_b \) is defined as

\[
\text{Sim}_{\text{POI}}(TR_a; TR_b; P) = \begin{cases} 
1 & \text{if } \forall p \in P, p \text{ is on } TR_a \text{ and } TR_b \\
0 & \text{otherwise}
\end{cases}
\]

Illustration: Figure 3.7 shows an illustrative example of the above definition on spatial similarity between two trajectories \( TR_a \) and \( TR_b \).

![Diagram of Spatial Similarity based on Point of Interest (POI)](image)

For a given POI set Temporal Distance can be defined at a point is defined follows:

Definition 3.3 Temporal Distance between Trajectories at a point.
Suppose that \( p \in P \), and \( P \) is the set of POI. Then the temporal distance between two trajectories \( TR_a \) and \( TR_b \) is

\[
\text{dist}_T(TR_a, TR_b, p) = |t(TR_a, p) - t(TR_b, p)|
\]

If neither \( TR_a \) nor \( TR_b \) pass through \( p \), the temporal distance is considered as infinity.

Illustration: Figure 3.8 shows an illustrative example of the above definition on temporal distance between two trajectories at a point in POI.
If we consider \( t(\text{TR}; p_i) \) as the time the \( i \)th POI, was passed each trajectory, TR, is plotted as a point \( t(\text{TR}) = (t(\text{TR}, p_1), t(\text{TR}, p_2), \ldots, t(\text{TR}, p_k)) \) in a \( k \)-dimensional space where \( k \) is the number of POIs. Then the temporal distance between two trajectories for a POI set is defined as the \( L_p \) distance of this \( k \)-dimensional space defined as follows:

**Definition 3.4** Temporal Distance between Trajectories for a POI set.

Suppose that \( P \) is a set of POI and \( \text{TR}_A \) and \( \text{TR}_B \) are two trajectories. Then the temporal distance between \( \text{TR}_A \) and \( \text{TR}_B \) is

\[
\text{dist}_T(\text{TR}_A, \text{TR}_B, P) = L_p(\text{TR}_A, \text{TR}_B, P) = \left( \sum_{i=1}^{k} |p_i(\text{TR}_A) - p_i(\text{TR}_B)|^p \right)^{1/p}
\]

**Illustration:** Figure 3.9 shows an illustrative example of the above definition on temporal distance between two trajectories at POI set. Note that when \( p = 2 \), the above distance become the Euclidian measure (Refer Definition 3.1).

Using above two definitions, the algorithm for searching similar trajectories is developed as a two stage process, filtering using spatial similarity (definition 3.2)
followed by refinement using temporal distance (definition 3.4). In the first stage, the algorithm proposed in [62 Hwang2006] has the following drawbacks.

(i) The similarity in space with definition (1 = similar, 0 = dissimilar) does not take into account any notion of similarity percentage or similarity range. Therefore, we cannot determine how similar two trajectories are in space.

(ii) The spatial similarity of two trajectories is based only on the fact that they share common points, and not into the general network space. Therefore many similarities have been excluded. For example, trajectories that have parallel edges with only a city block distance and no common points are considered completely dissimilar.

The second issue in finding structural similarity between locations [136 Sajimon2012] is addressed in chapter 4. The first issue is addressed [104 Sajimon2010] as a modification to the algorithm (Hwang2006) obtaining the algorithm 3.1.
**Algorithm 3.1.** Searching for Similar Trajectories moving through Points Of Interest based on Spatial Filtering and refinement using Temporal distance. Input: Input trajectories TRIN, spatial threshold ρ, temporal threshold δ, Query trajectory trQ, POI set P; Output: similar trajectories TROUT

```
Begin
   TRCandidate = φ
   TROUT = φ
   n = number of Points in P
   For each tr in TRIN,
      tr.k = 0
      For each p in P
         If p is on tr and in trQ then
            tr.k = tr.k + 1
      End For
      If (tr.k/n) > ρ then
         TRCandidate = TRCandidate U{tr}
      End For
   For each tr ∈ TRCandidate
      If distT (trQ, tr, P) < δ then
         TROUT = TROUT U{tr}
      End For
   return TROUT
End
```

A threshold is introduced to spatial similarity measure to determine how similar the trajectories with reference to how many points it pass through and then this measure is used to filter the trajectories based on spatial distance. So all the trajectories which satisfy this spatial threshold will be considered in the next stage of refinement. The binary encoded location data will have the advantage of primary filtering based on administrative district code or road code when the POI’s are within that constrained area.

Here k is a member variable stored along with each trajectory tr which will contain the percentage of similarity in matching with the number of points in P. This measure could be used later to cluster the similar trajectory to see how each trajectory is distant from the query trajectory. The spatial threshold ρ is an empirical value and has been fixed as per the accuracy requirement in spatial filtering. As the
proposed measure finds percentage of common points visited by both the trajectories and since our experiments go for considering 10000 POI’s we take the value of threshold \( \rho \) as 0.9 as we consider that a trajectory will be in the filter set even if it does not go through 10% or less points in POI.

3.7.2 Finding similar trajectories based on temporal filtering and spatial distance

In terms of practical application, the meaning of distance between two time intervals can rarely be found. However, it is interested to discuss the time intervals, TOI (Time of interest) as important characteristics in learning the moving behavior of moving objects on road networks. If trajectories pass the same points at the same TOI on road networks, they are similar to each other. Therefore, temporal similarity is defined based on TOI. For example, the heaviest traffic time intervals on a specific road network can be TOI. The trajectories are filtered using this definition. A similar consideration is given for temporal similarity measure \(^6^2\)Hwang2006 which will consider two trajectories temporally similar with respect to a given set of TOI, only when both the trajectories alive in all the time points mentioned. This is modified (as shown in algorithm 3.2) by introducing a threshold \( \delta \) to determine how similar the trajectories with reference to how many time points both the trajectories pass through. The temporal threshold \( \delta \) is an empirical value which has been fixed as per the accuracy requirement in temporal filtering as already discussed for spatial threshold in section 3.7.1. If two trajectories pass through the same TOI, they are considered similar by the following definition:

**Definition 3.5** Temporal Similarity between Trajectories on Road Networks

Suppose that \( T \) is a set of TOIs on a given road networks. Then, temporal similarity between two trajectories \( TR_A \) and \( TR_B \) is defined as

\[
\text{Sim}_{T_{\text{TOI}}} (TR_A, TR_B, T) = \begin{cases} 
1 & \text{if } \forall t \in T, t \in [t(t(TR_A)), t(t(TR_A))] \land t \in [t(t(TR_B)), t(t(TR_B))] \\
0 & \text{otherwise}
\end{cases}
\]
As an illustrative example for temporal similarity consider figure 3.10. If $T = [10:20, 10:28]$ Since both $TR_a$ and $TR_b$ are alive during $T$ we say both trajectories are temporarily similar at $T$.

If $T = [10:00, 10:30]$ then $TR_a$ and $TR_b$ are not temporarily similar, since $TR_b$ is not alive at the period $[10.00, 10.05)$.

**Definition 3.6 Spatial Distance between Trajectories**

Suppose that $t \in T$, and $T$ is the set of TOIs. Then the spatial distance between two trajectories $TR_A$ and $TR_B$ is defined as

$$dist_s(TRA, TRB, T) = \sum dist_s(p(TRA, t_i), p(TRB, t_i)),$$

where summation will go for number of common time points in both trajectories. Here $dist_s$ is the shortest road distance between two trajectories at time $t_i$, which is obtained by Dijkstra’s algorithm[164].

As an illustration take Figure 3.11. Let us take $TOI = \{10:00, 10:20, 10:30\}$,

Spatial distance between $TR_a$ and $TR_b = \text{Sum of the shortest network distance between } TR_a \text{ and } TR_b \text{ at each of the time points in TOI}$

$$= \text{Min}(2+1+6+2+4) + 3 + 1 + \text{Min}(1+6+2+4) + 3 + 3 + 1 + 4 + 3$$

$$= \text{Min}(10, 8) + 8 + 11 = 29$$
Algorithm 3.2 Objects moving through Times of Interest - Searching based on Temporal Filtering and refinement on Spatial Distance

**Input**: Input trajectory set $TR_{IN}$, threshold-s $\rho$, threshold-t $\delta$, query trajectory $tr_Q$, TOI set $T$, Output: Similar trajectory set $TR_{OUT}$

**Begin**

$TR_{Candidate} = \emptyset$

$TR_{OUT} = \emptyset$

$nt$ = number of Time Points in $T$

**For each** $t_r$ in $TR_{IN}$

$t_r.s = 0$

**For each** $t$ in $T$

**If** $t$ is a time point in $t_r$ and $tr_Q$ *then*

$t_r.s = t_r.s + 1$

**End For**

**If** $(t_r.s/nt) > \delta$ *then*

$TR_{Candidate} = TR_{Candidate} \cup \{ t_r \}$

**End if**

**For each** $t_r$ $\in$ $TR_{Candidate}$

**If** $dist_S(tr_Q, t_r, T) < \rho$ *then*

$TR_{OUT} = TR_{OUT} \cup \{ t_r \}$

**End if**

**End For**

**return** $TR_{OUT}$

**End**
Though the above method has theoretical relevance it has very little practical application as usually initial filtering is based on location or spatial features. The disadvantage of this method is that many trajectories are selected from trajectory data set by temporal filtering. For example, if the time interval of a query trajectory is much shorter than the total time interval for all moving objects, most trajectories are selected from trajectory data. Therefore a combined measure of both spatial and temporal similarity will be more accurate and useful in terms of practical application which is discussed in next section.

3.7.3 Finding similar trajectories based on spatio-temporal filtering and spatio-temporal distance refinement

In earlier two methods we follow two stage process with initial filtering and then refinement. In this method (as shown in algorithm 3.3), spatial and temporal similarity together is being used in the filtering step. Afterwards, we refine similar trajectories using spatial and temporal distance based on POI and TOI.

Advantages of threshold $\rho_1$, $\delta_1$ are that we can see the degree of similarity if the trajectories does not passes through all points in POI and all time Points in TOI. This measure could be used later for trajectory clustering purpose.

In all the above algorithms the advantages with binary encoding is that an initial filtering can be done by using binary code component of the district or road from the location data. Thus a large number of trajectories could be pruned out at the initial stage itself when the query requirement is restricted to a district or road.

3.8 Experimental Evaluation

All experiments have been conducted on an Intel Core 2 Duo machine running Windows XP, with 2 GB of RAM, and a 320 GB SATA2-16 MB hard disk. Two real data sets have taken for experimentation purposes, namely a small data set of fleet of trucks available in [16Theodoridis2007] trajectory data set and large data set INFATI[17Jensen2008] as discussed in Chapter 1.
Algorithm 3.3 Similarity Searching based on Spatio-Temporal Filter and Spatio-Temporal Distance Refinement

Input: Input trajectory set $TR_{IN}$, spatial threshold - $\rho_1, \rho_2$; temporal threshold - $\delta_1, \delta_2$; query trajectory $tr_Q$, POI set $P$, TOI set $T$.

Output: Similar trajectory set $TR_{OUT}$

\[
\begin{align*}
&\text{Begin} \\
&T_{\text{Candidate}} = \emptyset \\
&TR_{\text{OUT}} = \emptyset \\
&np = \text{number of Locations in POI} \\
&nt = \text{number of Time Points in TOI} \\
&\text{For each } t_r \text{ in } TR_{IN}, \\
&\quad \text{tr.k} = 0; \\
&\quad \text{For each } p \text{ in } P \\
&\quad \quad \text{If } p \text{ is on } t_r \text{ and in } tr_Q \text{ then} \\
&\quad \quad \quad t_r.k = t_r.k + 1 \\
&\quad \text{End for} \\
&\quad t_r.s = 0 \\
&\quad \text{For each } t \text{ in } T \\
&\quad \quad \text{If } t \text{ is a time point in } t_r \text{ and } tr_Q \text{ then} \\
&\quad \quad \quad t_r.s = t_r.s + 1 \\
&\quad \text{End For} \\
&\quad \text{If } (t_r.k/np) > \rho_1 \text{ and } (t_r.s/nt) > \delta_1 \text{ then} \\
&\quad \quad T_{\text{Candidate}} = T_{\text{Candidate}} \cup \{t_r\} \\
&\text{End for} \\
&\text{For each } t_r \in T_{\text{Candidate}} \\
&\quad \text{If } (\text{dist}_T(tr_Q,t_r,P) < \rho_2) \text{ and } (\text{dist}_s(tr_Q,t_r,T) < \delta_2) \text{ then} \\
&\quad \text{TR}_{\text{OUT}} = TR_{\text{OUT}} \cup \{t_r\} \\
&\text{End For} \\
&\text{return } TR_{\text{OUT}} \\
&\text{End}
\end{align*}
\]

Comparison of Similarity Search Time

The average search time in query processing for the three methods [104Sajimon2010] discussed above are compared with those discussed in [62Hwang2006] for different set of POI’s using INFATI data set. As shown in Figures 3.12, 3.13 and 3.14 the experimental evaluation done in [62Hwang2006] for TOI based method involves so many trajectories including meaningless trajectories at the filtering stage as the time interval of a query trajectory is smaller than the total life span for all moving objects. But in the above experiments one could
pruned out large number of unwanted trajectories when POI or TOI has taken in a specific district or road using the concept of binary encoding. Though as shown in Figure 3.12 the search time in first two methods are little higher than that in [62Hwang2006], for higher number of POI’s it is better than Hawang method as shown in Fig 3.13 and Fig 3.14. As each method could do a preliminary filtering based on administrative district or road, the average search time in all the three proposed methods are less compared to the earlier implementation. Also the additional thresholds used in the filtering step provide the possibility of clustering the trajectories for future data mining applications.

In all the following graphs Algorithm 3.1, Algorithm 3.2 and Algorithm 3.3 are respectively referred as Algorithm 1, Algorithm 2 and Algorithm 3 for convenience.

![Comparison of Search Time](image)

Figure 3.12. Evaluation of search time performance for 3000 POI’s

**Evaluation of Accuracy of Algorithms**

In this experiment, we focus on evaluating the accuracy of the algorithms, i.e., the amount of false positives and false negatives. A false negative(FN), is the error of not finding a pattern that does exist in the data. A false positive(FP), is the error of
finding a “pattern” that does not exist in the data. A True Positive (TP) is the case of finding a pattern that does exist in the data and a True Negative (TN) is the case not finding a pattern which does not exist in the data.
Sensitivity and specificity are statistical measures of the performance of a binary classification test, also known in statistics as classification function. Sensitivity (also called the true positive rate, or the recall rate in some fields) measures the proportion of actual positives which are correctly identified as such (e.g. the percentage of sick people who are correctly identified as having the condition). Specificity measures the proportion of negatives which are correctly identified as such (e.g. the percentage of healthy people who are correctly identified as not having the condition, sometimes called the true negative rate). These two measures are closely related to the concepts of type I and type II errors in Testing of Hypothesis. A perfect predictor would be described as 100% sensitive (i.e. predicting all people from the sick group as sick) and 100% specific (i.e. not predicting anyone from the healthy group as sick); however, theoretically any predictor will possess a minimum error bound known as the Bayes error rate.

Below formulae were used to calculate sensitivity, specificity and accuracy.

\[
\text{Sensitivity} = \frac{TP}{TP + FN} \\
\text{Specificity} = \frac{TN}{TN + FP} \\
\text{Accuracy} = \frac{(TP + TN)}{(TP + FP + TN + FN)}
\]

Table 3.1 shows sensitivity, specificity and accuracy for different similarity algorithms in our proposal. Figure 3.15 shows the graphical representation of these measures.

<table>
<thead>
<tr>
<th>Similarity Algorithms</th>
<th>Sensitivity</th>
<th>Specificity</th>
<th>Accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td>Algorithm 3.1</td>
<td>82.24%</td>
<td>74.82%</td>
<td>79.11%</td>
</tr>
<tr>
<td>Algorithm 3.2</td>
<td>72.22%</td>
<td>73.16%</td>
<td>78.45%</td>
</tr>
<tr>
<td>Algorithm 3.3</td>
<td>85.75%</td>
<td>75.63%</td>
<td>80.1%</td>
</tr>
</tbody>
</table>

Table 3.1 Sensitivity, Specificity and Accuracy
The experiment shows that Algorithm 3.3 is having better performance in identifying true positives, true negatives and accuracy, comparing with other two algorithms. The following analysis technique has been used to evaluate the overall performance of these algorithms.

**ROC Space Analysis**

A Receiver Operating Characteristic (ROC) space has been used to evaluate the overall accuracy and predictive value of the algorithms discussed above. It is a standard approach to evaluate sensitivity and specificity of diagnostic procedures. It is defined by False Positive Rate and True Positive Rate which shows relative trade-off between true positive and false positive.

\[
\text{True Positive Rate} = \frac{TP}{TP + FN} \\
\text{False Positive Rate} = \frac{FP}{FP + TN}
\]

Table 3.2 shows True Positive Rate and False Positive Rate for above three Algorithms 3.1, 3.2 and 3.3. Figure 3.16 shows True Positive Rate and False Positive Rate for these algorithms in graphical format. The best possible prediction model will be at coordinate (0, 1) in graph on Figure. 3.16. This will represent...
100% True Positive Rate and no False Positive Rate which will be ideal case. In the conducted experiment none of the method could reach this ideal rate but algorithm 3.3 which takes the spatio-temporal measure is more nearer to the ideal point.

<table>
<thead>
<tr>
<th>Similarity Algorithms</th>
<th>True Positive Rate</th>
<th>False Positive Rate</th>
</tr>
</thead>
<tbody>
<tr>
<td>Algorithm 3.1</td>
<td>0.8525</td>
<td>0.2433</td>
</tr>
<tr>
<td>Algorithm 3.2</td>
<td>0.8236</td>
<td>0.2512</td>
</tr>
<tr>
<td>Algorithm 3.3</td>
<td>0.8768</td>
<td>0.2353</td>
</tr>
</tbody>
</table>

Table 3.2 True Positive Rate and False Positive Rate

The experimental results shows that the revised algorithms proposed in finding spatio-temporal similarity with binary encoding scheme outperform the existing algorithm in terms of similarity search time, and shows better performance in evaluating accuracy in ROC space analysis.

3.9 Conclusion
The similarity problem in trajectory database for moving objects on road networks has many applications in security informatics area like traffic security,
identification of traffic congestion and re-routing etc. It has applications in other areas like logistics, supply chain management and geo-marketing. An earlier work on dimension reduction has been identified as a baseline concept, and has demonstrated the relevance of binary encoding scheme of location data to have advantages over existing trajectory similarity algorithms for network constrained moving objects. These algorithms have modified in finding percentage of similarity which could be used as a measure for trajectory clustering applications. As a continuation work, one can plan to use these measures in different types of clustering algorithms that will have lot of applications in security related area like segregation of objects with specific moving characteristics, emergency area clusters and dark web link analysis. Some attempt has been made by the researcher as detailed in the next two Chapters, which address sequence similarity and trajectory clustering.