공간 네트워크에서 이동객체 궤적을 위한 공간 유사도 측정방법의 설계

라빈드라 비스타^{O} 장재우
전북대학교 컴퓨터공학과
{rabindra^{O}, jwchang}@dblab.chonbuk.ac.kr

Design of Spatial Similarity Measure for Moving Object Trajectories in Spatial Network

Rabindra Bista^{O}, Jae-Woo Chang
Computer Engineering Department, Chonbuk National University

Abstract

Similarity search in moving object trajectories is an active area of research. In this paper, we introduce a new concept of measure that computes spatial distance (similarity) between two trajectories of moving objects on road networks. In addition, we propose an algorithm that generates a sequence of matching edge pairs for two trajectories that are to be compared and computes spatial distance between them which is non Euclidian in nature. With an example, we explain how our algorithm works to show spatial similarity between trajectories of moving objects in spatial network.

1. Introduction

Similarity search in moving object trajectories has attracted a lot of research work recently. Spatial similarity between trajectories can be defined as spatial distance between two trajectories. A data trajectory R is said to be spatially similar to query Q when the spatial distance between them is lesser than the spatial distance between the query and other trajectories. For this, first, we divide the data trajectory as well as the query trajectory into edges (road segments between two intersections). For each edge of the query Q, matching edge pair is selected from the data trajectory R on the basis of spatial closeness to each other. After this, a sequence of matching edge pairs is generated. Then, spatial similarity of query Q to data trajectory R is computed by averaging the spatial distances of the matching edge pairs.

The concept of spatial similarity search in moving object trajectories is an important technique useful for analyzing the behavior of moving objects on road networks. Hence, there is a need to define a measure to determine similarity among the trajectories of moving objects. Similarity measurement allows for the retrieval of similar trajectories and eventual discovery of the knowledge from patterns of the similar trajectories. For example, we can make marketing strategies by analyzing the trajectories of cars on a road.

In this paper, we introduce a new concept of measure for similarity search, called spatial similarity. For this, we define distance definitions and exploit benefits of road networks to measure spatial distance (non Euclidian distance) between two edges of the road. In addition, we propose an algorithm to calculate spatial similarity between trajectories.

The remainder is organized as follows. We briefly overview related work in Section 2. Our method is presented with an example in Section 3. Section 4 concludes the paper with future works.

2. Related Work

The problem of discovering similar trajectories received increasing attention in recent years and this motivated the development of various algorithms and tools. The focus in this area lies primarily on the development of distance functions for similarity measurement.

Some important studies of search methods for similar trajectories are found in [1] and [2]. A method for finding the most similar trajectory to a given query trajectory within a database using the longest common subsequence model was proposed in [2]. Since this method is based on Euclidean space, it cannot address to search similar trajectories of moving objects on road networks. In [4], a method was defined for measuring the spatial similarity between trajectories based on shape. The advantage of this definition is that spatio-temporal aspect is taken into account but since this method assumes Euclidean space, it is difficult to apply it to road networks. In [3], a similar method was proposed but has the same problem of Euclidean distance as [1]. A comparison of spatial shapes of moving object trajectories was proposed in [5]. The main advantage of this approach is to evaluate one-way distance (OWD) function in both continuous and discrete cases of moving object trajectories. But the problem with this approach to address spatial similarity between trajectories in road networks is that it is based on Euclidian distance.

3. Spatial Similarity Measure

As discussed in section 2, although there have been much researches in the field of finding similar trajectories of moving objects, there exists little research on searching similar trajectories of moving objects like trajectories of cars on road networks. Two main aspects of showing similarity between two trajectories of moving object on road networks are how to compare them and how to calculate spatial distance between them. To show spatial similarity between two trajectories, we describe a method to generate a sequence of matching edge pairs of two trajectories and define a measure to calculate spatial distance between them.

3.1. Some Definitions Related to Spatial Similarity:

Let us consider that there are two trajectories, one of which is a query trajectory Q and another one is data trajectory R, each of them containing a set of edges, Q = {q1, q2, q3,⋯,qi} and R = {r1, r2, r3, ⋯,rj}.

Definition1: Spatial distance between two edges (qi, rj) from two different trajectories (Q, R) respectively, on road networks can be defined as the average distance of the two network distances obtained from the start and end nodes of one edge to the start and end nodes of another edge respectively.

$$D_{edge}(q_i, r_j) = \frac{1}{2}\left\{\begin{matrix} D_n\left(q_i.startNode, r_j.startNode\right) \\ +D_n\left(q_i.endNode, r_j.endNode\right) \end{matrix}\right\}$$

Definition2: Let us consider there exist qi and rj as a matching edge pair and *Dist* is Dedge i.e. spatial distance between edges of the matching edge pair (qi, rj).Then, a set of matching edge pairs for trajectories Q and R can be defined as

$$MatchingEdgePair(Q,R) = \{q_i, r_j, Dist\}$$

Definition3: Spatial Similarity (SS) between two trajectories Q and R can be defined as an average numerical value obtained from the summation of spatial network distances of all matching edge pairs:

$$Spatial\ Similarity(SS) = \frac{1}{m}\left[\sum_{k=1}^{m} MatchingEdgePair[k].Dist\right]$$

where, k= number of matching edge pairs

3.2. Spatial Similarity Algorithm

Our spatial similarity algorithm shows spatial co-relationship between two trajectories of moving objects on road networks. For this, it generates a sequence of matching edge pairs with spatial distance between two edges of the each edge pair and computes spatial similarity of the trajectories.

A. Spatial Similarity Algorithm to measure spatial distance between a query Q and a data trajectory R:

```
SpatialSimilarity(Q,R)
 {
   MatchingEdgePair= {ϕ};
   SpatialSimilarity= 0;
   getMatchingEdge(Q, R, MatchingEdgePair);
    for every ith pair in MatchingEdgePair
      SpatialSimilarity += MatchingEdgePair[i].Dist;
      SpatialSimilarity = SpatialSimilarity/I;
 }
```

B. Generating a sequence of matching edge pairs from Q and R:

```
getMatchingEdge( Q, R, Medge )
 {  Medge = {ϕ };
    for each ith edge of Q
      findMatchEdge( Qi, R, Medge );
    for ith unselected edge of R
      findMatchEdge( Ri, Q, Medge );
 }
```

C. Searching matching edge pairs

```
findMatchEdge(Qi, R, Medge)
   {
    minDistS=minDistE= ∞ ;
    for each jth edge of R
    {
     if (Dist(qi.StartN, rj.StartN) < minDistS)
      { minDistS= Dist(qi.StartN, rj.StartN);
       minStartN= j;}
     if (Dist(qi.EndN, rj.EndN) < minDistE)
       { minDistE= Dist(qi.EndN, rj.EndN)
       minEndN= j;}
     if (minDistS==minDistE = = 0)
      if ( qi.eID == ri.eID)    // eID = edge ID
       { Medge = Medge U {qi, rj, 0}; return;}
12      else Medge = Medge U < qi, ri,
                    1/3(l.qi + l.ri)/2 >;
    }
13      if ((Dedge(qi, RminStartN) - Dedge(qi,
                              RminEndN) > 0 )
14          Medge= Medge U< qi, RminEndN,
                    Dedge(qi, RminEndN) >;
15      else Medge= Medge U<qi, RminStartN,
                    Dedge(qi, RminStartN)> ;
  }
```

Fig.1. Spatial Similarity Search Algorithm

Now, we explain spatial similarity algorithm in brief. Lines 1-2 of algoritm A are initialization part. Line 3 invokes getMatchingEdge(Q, R, Medge) function where, Medge is a sequence matching edge pairs of query Q and data trajectory R. This function returns a sequence of matching edge pairs from Q and R. Lines 4-6 calculate spatial similarity i.e. spatial distance from Q to R. Algorithm B receives a sequence of matching edge pairs from algorithm C and passes it to algorithm A. For this, Algorithm B invokes findMatchEdge(Qi, R, Medge) function to find matching edge pairs for all query edges from R (lines 2 and 3) and invokes findMatchEdge(Ri, Q, Medge) function to find matching edge pairs for all unselected edges of R from Q (lines 4 and 5). Algorithm C is for searching the matching edge pairs for Q and R. In line 1, minDistS (minimum distance between edge of query star point and edge of data trajectory start point) and minDistE (minimum distance between edge of query end point and edge of data trajectory end point) are initialized. For loop runs between lines 2-12 and searches every edges of data trajectory to find matching edge pair for each edge of query trajectory and the searching process is opposite to it if there exist unselected edges of R. Lines 3-8 find minimum distances from start point of one query edge to start points of all data edges and from end point of the query edge to the end points of all data edges one by one. Among all of the edge pairs of query edge qi with an edge of the data trajectory, an edge rj is selected as the matching edge pair for the query edge qi which has minimum spatial distance from it. Lines 10 and 11 return the matching edge pair rj for qi where distance between them is zero. That means both of them have same start point and end point and represent a single segment of a road network. Line 12 returns the matching edge rj which has the same starting and ending points as qi has but both of them represent different segments of a road network. Lines 13-15 return the matching edge pair (qi and rj) with minimal spatial distance. For this, qi start point to rj start point and qi end point to rj end point distances are compared to each other and the spatial distance for the matching pair qi and rj is taken one which is lower than another spatial distance value.

In summary, algorithm C searches a sequence of matching edge pairs for query Q and data trajectory R. It computes minimum spatial distance between edges of each (qi,rj) pair and sends it to algorithm B. Algorithm B which acts as a mediator between Algorithms A and C. It receives a sequence supplied by Algorithm C and passes it to Algorithm A. Finally, algorithm A takes the sequence of the matching edge pairs and computes the spatial similarity for query Q and data trajectory R.

Furthermore, in order to calculate spatial distance between two edges, the algorithm computes the shortest spatial distance considering how they are connected to each other on road networks. For this, a road network materialization file is used that consists of spatial distance between every two intersections of the road.

3.3. Example

In this example, we show how a sequence of matching edge pairs is generated and how spatial similarity between two trajectories is calculated. We assume that there are three trajectories, one is query trajectory Q and other two are data trajectories R and S as shown in figure2. The direction of all trajectories is from left to right.

In figure2, we can see that query trajectory Q consists of edges Q = {q1, q2, q3, q4, q5, q6, q7}. Similarly, data trajectories R consists of R = {r1, r2, r3, r4, r5, r6, r7} and S consists of S = {s1, s2, s3, s4, s5, s6, s7} edges. Each edge of the network has provided with spatial distance. Now, we are generating a sequence of matching edge pairs for each pair of a query trajectory and a data trajectory (i. e. QR and QS) using spatial similarity algorithm.

Let us start with QR pair. First, the algorithm finds the matching edge for q1. For this, the algorithm calculates spatial distance from the start and end points of q1 to start and end points of each edge of R, respectively. Second, it selects an edge which has minimum distance from q1. Since r1 and q1 have same start and end points and as both of them represent same edge of the network, the distance from q1 to r1 is zero. So, r1 is selected as matching edge pair of q1 and Medge = {(q1, r1, 0)}. Next, edge pair is searched for q2. Both q2 and r2 have same start point. Therefore, r2 is considered as the matching edge pair of q2 and distance is given by the average of starting points and end points distances i.

e. Dedge = (0+390)/2=195. Medge is updated as Medge= {(q1, r1, 0), (q2, r2, 195)}. After this, edge pair for q3 is found. Since R has no such edge which is either start point or end point common to points of q1, the matching edge for q3 is selected from a set of edges of R which has minimum edge distance from q3. r3 is selected as edge pair of q3 because Dedge(q3, r3)=425 which is the lowest value among the spatial distance values of other combinations of qi and rj. Thus, Medge = {(q1, r1, 0), (q2, r2, 195), (q3, r3, 425)}. Now, the turn is to find edge pair of q4. Like q3, pair edge of q4 is calculated by finding the edge in R, which is located at the shortest spatial distance from q4. r4 is in the shortest distance from q4 and Medge = {(q1, r1, 0), (q2, r2, 195), (q3, r3, 425), (q4, r4, 435)}. For q5, matching pair is r5 because both have same end point and lower spatial distance than other. So, Medge = {(q1, r1, 0), (q2, r2, 195), (q3, r3, 425), (q4, r4, 435), (q5, r5, 205)}. As a penultimate edge pair, r6 is selected for q6 because both q6 and r6 has same start point, end point and length and Medge = {(q1, r1, 0), (q2, r2, 195), (q3, r3, 425), (q4, r4, 435), (q5, r5, 205), (q6, r6, 0)}. Finally, r7 is selected for q7 as both of them have same start point and located at the shortest distance. Now, Medge is updated for the final time and contains a sequence of seven matching edge pairs, Medge = {(q1, r1, 0), (q2, r2, 195), (q3, r3, 425), (q4, r4, 435), (q5, r5, 205), (q6, r6, 0), (q7, r7, 205)}.

Similarly, we can generate a sequence of matching edge pairs for Q and S. First, the algorithm finds matching edge of q1. Although, q1 and s1 have same start point and q1 and s3 have same end point, s3 is selected for matching edge pair of q1 because s3 (=175) is spatially closer to q1 than s1 (=180). The pair, q1 and s3 are stored in Medge' and Medge' = {(q1, s3, 175)}. Second, the edge pair is searched for q2. Since s4 has the same start point, end point and same edge ID as the q2, the distance from q2 to s4 is zero. s4 is selected as matching edge pair of q2 and Medge' = {(q1, s3, 175), (q2, s4, 0)}. Third, s5 is selected as the edge pair of q3 because both of them have same start point and are spatially closer. So, Medge' = {(q1, s3, 175), (q2, s4, 0), (q3, s5, 218)}. Fourth, s5 is selected as an edge pair of q4 because s5 and q4 are spatially closer to each other than others. Thus, Medge' = {(q1, s3, 175), (q2, s4, 0), (q3, s5, 218), (q4, s5, 230)}. Fifth, q5 and s5 are selected as the next matching edge pair because both of them have same end point and are close to each other. Medge' = {(q1, s3, 175), (q2, s4, 0), (q3, s5, 218), (q4, s5, 230), (q5, s5, 215)}. Sixth, s6 is selected for q6 as a matching edge pair because both q6 and s6 has same start point, end point and edge ID. So, Medge' = {(q1, s3, 175), (q2, s4, 0), (q3, s5, 218), (q4, s5, 230), (q5, s5, 215), (q6, s6, 0)}. After this, matching edge pair is searched for q7. q7 and s7 have same start and end points and they are taken as a matching edge pair. Although, both q7 and s7 have same start and end points, their paths are different from one another representing different segments of road networks. The spatial distance between them is given by 1/3(210+230)/2 = 1/3 *220 = 73 and Medge' = {(q1, s3, 175), (q2, s4, 0), (q3, s5, 218), (q4, s5, 230), (q5, s5, 215), (q6, s6, 0), (q7, s7, 73)}.

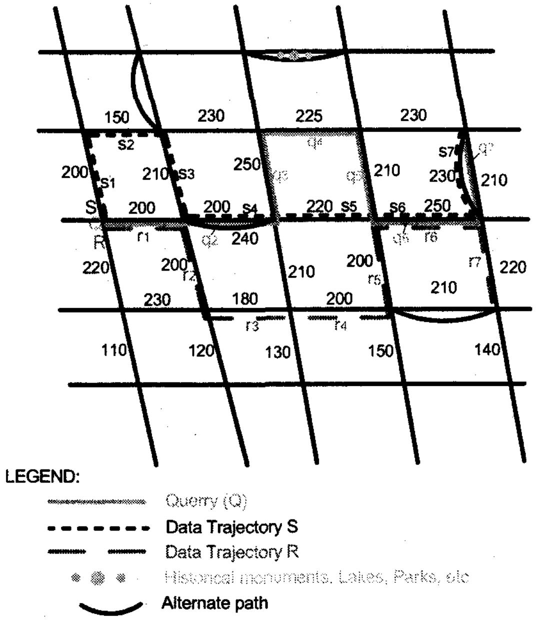

Fig.2. A Road network showing two data trajectories R and S and one query trajectory Q

Until now, there is no matching edge for s1 and s2. These edges are known as unselected edges. For each unselected edge, the algorithm searches matching pair from the set of query edges. For this, the algorithm finds matching pair of unselected edge of the data trajectory from the set of query edges by selecting the spatially closest edge. In the case of edge s1, q1 is selected as matching edge pair because s1 and q1 have same start point and are spatially close to each other. The spatial distance between s1 and q1 is 180 and Medge' = {(q1, s3, 175), (q2, s4, 0), (q3, s5, 218), (q4, s5, 230), (q5, s5, 215), (q6, s6, 0), (q7, s7, 73), (s1, q1, 180)}. At last, s2 and q1 are selected for matching edge pair because q1 is spatially closer to s2 than other query edges. The edge distance between s2 and q1 is 205. Finally, Medge' = {(q1, s3, 175), (q2, s4, 0), (q3, s5, 218), (q4, s5, 230), (q5, s5, 215), (q6, s6, 0), (q7, s7, 73), (s1, q1, 180), (s2, q1, 205)}.

After generating a sequence of matching edge pairs, the algorithm computes spatial similarity of a data trajectory with query Q. The spatial similarity for Q and R is calculated as SS= (0+ 195+ 425+ 435+ 205+ 0+ 205)/7 = 209. Similarly, spatial similarity for Q and S is SS' = (175+ 0+ 218+ 230+ 215+0+ 73+ 180+ 205)/9=144.

In this example, S is spatially more similar to Q than R because spatial similarity of Q with S is 144, which is smaller than spatial similarity of Q with R (=209).

4. Conclusions and Future Works

In this paper, we propose a new measure for computing spatial similarity between two trajectories of moving objects on road networks. Given a query object Q and data trajectory T, our proposed measure returns the spatial similarity between Q and T. The spatial similarity between two trajectories represents the spatial network distance between them. We present an example to explain how our algorithm works to compute the spatial similarity between two trajectories.

As future work, we will implement the proposed algorithm to evaluate spatial similarity measure.

ACKNOWLEDGEMENTS:
This work was supported by the Korean Research Foundation Grant funded by the Korean Government (MOEHRD) (KRF-2005-211-D00 322).

5. References

[1] Michail Vlachos, George Kollios, and Dimitrios Gunopulos. Discovering Similar Multidimensional Trajectories. In *Proceedings of the Eighteenth International Conference on Data Engineering*, pages 673{684. IEEE Computer Society, 2002.

[2] Michail Vlachos, Dimitrios Gunopulos, and George Kollios. Robust Similarity Measures for Mobile Object Trajectories. In *Proceedings of the Thirteenth International Workshop on Database and Expert Systems Applications*, pages 721{728.IEEE Computer Society, 2002}.

[3] Choon-Bo Shim and Jae-Woo Chang. Similar Sub-Trajectory Retrieval for Moving Objects in Spatio-temporal Databases. In *Proceedings of The Seventh East European Conference on Advances In Databases and Informations Systems*, pages 08{322. Springer-Verlag, 2003.

[4] Yutaka Yanagisawa, Jun ichi Akahani, and Tetsuji Satoh. Shape-Based Similarity Query for Trajectory of Mobile Objects. In *Proceedings of the Fourth International Conference on Mobile Data Management*, pages 63{77. Springer-Verlag, 2003.

[5] Bin Lin and Jianwen Su, Shapes Based Trajectory Queries for Moving Objects. *GIS' 05, November 4-5, 2005*, Bremen, Germany.