EDS: A Segment-based Distance Measure for Sub-trajectory Similarity Search

Min Xie *
The Hong Kong University of Science and Technology
mxieaa@ust.hk

ABSTRACT

In this paper, we study a *sub-trajectory similarity search* problem which returns for a query trajectory some trajectories from the trajectory database each of which contains a *sub-trajectory* similar to the query trajectory. We show the insufficiency of the *distance measures* that are originally designed for *trajectory similarity search* where each trajectory *as a whole* is compared with the query trajectory, and thus we introduce a new *segment-based* distance measure called *EDS* (Edit Distance on Segment) for sub-trajectory similarity search. We conducted experiments on a real data set showing the superiority of our EDS distance measure.

Categories and Subject Descriptors

H.2.8 [**Database Management**]: Database Applications—*spatial databases and GIS*

Keywords

Sub-trajectory; Similarity search

1. INTRODUCTION

Due to the popularity of location-tracking devices trajectory data is ubiquitous nowadays. *Trajectory similarity search* which returns for a query trajectory some trajectories from the trajectory database each of which *as a whole* is similar to the query trajectory is a popular query on trajectory data [1–3, 5, 6]. In some cases, a trajectory that is *not similar* to the query trajectory *as a whole* might contain a *sub-trajectory* (a portion of the trajectory) that is *similar* to the query trajectory, and finding the similar sub-trajectories is useful in many applications. To illustrate, consider the following scenario.

Suppose that we as a taxi company received a complaint from a passenger that one of our taxi drivers drove along an unnecessarily long route for a higher charge. Apart from checking whether this fooling behavior happened, we also want to check whether some other taxi drivers have cheated passengers by driving along a similar route. Using the traditional trajectory similarity search techniques might give us the conclusion that no trajectories are similar to the query trajectory. This, however, is usually not the real case since we might have some taxi trajectories which contain *sub-trajectories* that are similar to the query trajectory and thus the corresponding drivers are bad citizens and should be figured out.

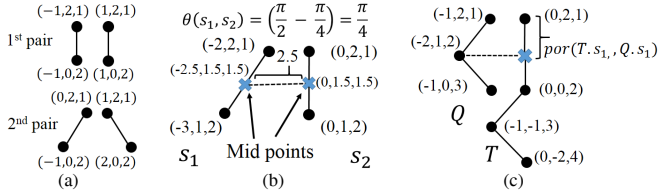


Figure 1: (a) Point-based distance and segment-based distance (b) Segment transformation cost (c) Computational example

For the sub-trajectory search problem, one may think of the approach that computes the distance between *each possible sub-trajectory* of a trajectory and the query trajectory by using one of the existing trajectory distance measures [1–3, 5, 6], which, however, has an expensive computation cost since a trajectory consisting of n sampled points simply has $O(n^2)$ possible sub-trajectories. In this paper, we design a distance measure called *EDS* (Edit Distance on Segment) which avoids the costly enumeration of all possible sub-trajectories for a trajectory. Instead, EDS only enumerates all possible *suffixes* of the trajectory (note that a trajectory consisting of n sampled points has $O(n)$ possible suffixes only). Another related work is [4] which defines a distance measure based on two segments, instead of two trajectories (or sub-trajectories).

Besides, the existing trajectory distance measures [1–3, 5, 6] are all *point-based* which means that only the information about the sampled points of the trajectory is utilized which sometimes is not sufficient enough. To illustrate, consider Figure 1(a) where we have four trajectories each of which has two points and each point is associated with a triplet (x, y, t) indicating that the point has its location at (x, y) and its timestamp of t. According to these existing distance measures, the distance between the first pair of two trajectories (the two at the top) is the same as the distance between the second pair of two trajectories (the two at the bottom), which, makes little sense since obviously the two trajectories in the second pair are more dissimilar than the two in the first pair. Our EDS distance measure is more robust in these cases, since it is *segment-based*, meaning that the information of the segments, including not only the information of points but also the other information such as direction and length, is utilized for measuring the similarity.

Contribution. We identify the problem of sub-trajectory similarity search, design a new distance measure EDS for the problem, and conducted experiments which verified our new measure EDS.

2. DISTANCE MEASURE: EDS

Let T be a trajectory of n segments, $(s_1, s_2, ..., s_n)$. Each segment s_i $(1 \leq i \leq n)$ has a start point denoted by $s_i.p_1$ and an end point denoted by $s_i.p_2$. Besides, we denote by $T.s_i$ the i^{th} segment

*The author would like to thank Cheng Long and Raymond Chi-Wing Wong for their helpful suggestions for improving the presentation in this paper.

SIGMOD/PODS'14, Jun 22-27 2014, Snowbird, UT, USA
ACM 978-1-4503-2376-5/14/06.
http://dx.doi.org/10.1145/2588555.2612665.

of T. Given two points p_1 and p_2, $d(p_1, p_2)$ denotes the Euclidean distance between p_1 and p_2.

The major idea of our distance measure EDS is to define the distance between two trajectories to be the minimum cost of a series of *segment-wise* transformations each of which changes one segment to the other. Before we introduce EDS, we define the cost of a segment-wise transformation, i.e., the cost of changing a segment to another one. The intuition we use is that given two segments s and s', we can transform s to s' by *displacing*, *stretching* and *rotating* s properly. Thus, we capture the cost of the segment-wise transformation by using the cost of each of these three operations.

Definition 1. Given two segments s and s', the cost of the segment-wise transformation between s and s', denoted by $cost(s, s')$, is defined to be

$$w_{dis} \cdot c_{dis}(s, s') + w_{str} \cdot c_{str}(s, s') + w_{rot} \cdot c_{rot}(s, s') \quad (1)$$

where w_{dis}, w_{str} and w_{rot} are three weights, $c_{dis}(s, s') = d(mid(s), mid(s'))/d_{max}$ is the displacing cost ($mid(s)$ is the mid-point of s and d_{max} is a large distance such as the maximum distance between two points in the trajectory database), $c_{str} = 1 - \min\{|s|, |s'|\}/max\{|s|, |s'|\}$ is the stretching cost ($|e| = d(s.q_1, s.p_2)$), and $c_{rot} = \theta(s, s')/\pi$ is the rotating cost ($\theta(s, s')$ is the angle between the vector from $s.p_1$ to $s.p_2$ and that from $s'.p_1$ to $s'.p_2$).

To illustrate, consider Figure 1(b). Suppose that d_{max} is 3.16 (the maximum distance between two points). $c_{dis}(s_1, s_2) = d(mid(s_1), mid(s_2))/d_{max} = 2.5/d_{max} = 0.791$, $c_{str}(s_1, s_2) = 1 - \frac{|s_2|}{|s_1|} = 1 - \frac{1}{\sqrt{2}} = 0.293$ and $c_{rot}(s_1, s_2) = \theta(s_1, s_2)/\pi = (\frac{\pi}{2} - \frac{\pi}{4})/\pi = 0.25$. Then, $cost(s_1, s_2) = 0.791 + 0.293 + 0.25 = 1.334$ if all weights are 1.

With the segment-wise transformation cost defined, we explain our distance measure EDS as follows. Let T be a trajectory and Q be the query trajectory. We denote the distance between T and Q under the EDS distance measure by $EDS(T, Q)$. We transform between T and Q with two operators, namely *insertion* and *replacement*. We start from $T.s_1$ on T and $Q.s_1$ on Q. We have three options.

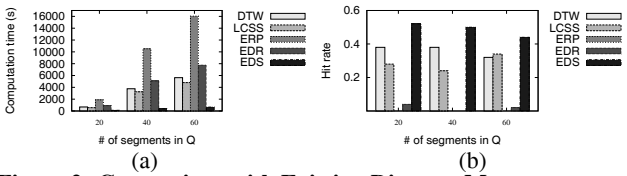
- We *insert* $T.s_1$ into $Q.s_1$ which means that $T.s_1$ is transformed to the *portion* of $Q.s_1$ that begins from the start point of $Q.s_1$ and stops at the point along $Q.s_1$ which is nearest to the end point of $T.s_1$. We denote this portion of $Q.s_1$ by $por(Q.s_1, T.s_1)$. In this case, $EDS(T, Q)$ corresponds to $cost(T.s_1, por(Q.s_1, T.s_1))$ plus the EDS distance between T excluding $T.s_1$ and Q excluding $por(Q.s_1, T.s_1)$ (which is defined recursively).
- We *insert* $Q.s_1$ into $T.s_1$ which has an analogous meaning as the first option and we define $EDS(T, Q)$ accordingly.
- We *replace* $T.s_1$ with $Q.s_1$ which means that $T.s_1$ is transformed to $Q.s_1$. Then, $EDS(T, Q)$ is $cost(T.s_1, Q.s_1)$ plus the EDS distance between T excluding $T.s_1$ and Q excluding $Q.s_1$ (which is defined recursively).

Then, the minimum one among these three costs corresponds to $EDS(T, Q)$. Formally, we have $EDS(T, Q) =$

$$\begin{cases} 0 & \text{if } |Q| = 0 \\ \infty & \text{if } |T| = 0 \text{ and } |Q| \neq 0 \\ min\{ & otherwise \\ cost(T.s_1, por(Q.s_1, T.s_1)) + EDS(T\backslash T.s_1, Q\backslash por(Q.s_1, T.s_1)) \\ cost(por(T.s_1, Q.s_1), Q.s_1) + EDS(T\backslash por(T.s_1, Q.s_1), Q\backslash Q.s_1), \\ cost(T.s_1, Q.s_1) + EDS(T\backslash T.s_1, Q\backslash Q.s_1)\} \end{cases}$$

where $T\backslash s$ denotes the resulting trajectory of T by excluding s.(2)

Compared with the existing distance measures [1–3, 5, 6], our EDS distance measure has two advantages. First, in order to do sub-trajectory similarity search (i.e., finding the most similar sub-trajectory to the query trajectory), with EDS, it is enough to enumerate all possible *suffixes* of the trajectory (which has order $O(n)$) while with one of the existing distance measures, it has to enumerate all possible *sub-trajectories* of the trajectory (which has order $O(n^2)$). This is because our definition of EDS already traverses all possible *prefixes* (See the case of "$|Q| = 0$"). Second, EDS is a *segment-based* distance measure which by its nature is better than a *point-based* distance measure (adopted by existing distance measures) since segments capture more information than points.



(a) (b)

Figure 2: Comparison with Existing Distance Measurements

In Figure 1(c), we compute the EDS distance between trajectory Q and T. To obtain the best sub-trajectory in T, we compute the EDS distance between Q and all the suffixes of T, namely $T_1 = T[1, 3]$, $T_2 = T[2, 3]$ and $T_3 = T[3, 3]$ where $T[i, 3]$ is the trajectory from segment s_i to s_3. When computing $EDS(T_1, Q)$, we first insert $Q.s_1$ to $T.s_1$ by creating $por(T.s_1, Q.s_1)$ and compute $cost(por(T.s_1, Q.s_1), Q.s_1)$. $Q.s_2$ is then replaced with the remaining half of $T.s_1$ (i.e. $T.s_1\backslash por(T.s_1, Q.s_1)$). After these two transformations, there are no remaining segments in Q and therefore all the remaining segments in T are skipped. Similarly, we compute $EDS(T_2, Q)$ and $EDS(T_3, Q)$. Finally, $EDS(T_2, Q)$ turns out to be the smallest cost, corresponding to the final answer.

3. EXPERIMENTS AND CONCLUSION

We conducted our experiments on a benchmark dataset "Athens trucks" (http://www.chorochronos.org/?q=node/5) which contains 1100 trajectories of various lengths. Our query trajectory Q was generated by first randomly selecting a trajectory from the data set, second randomly sampling a sub-trajectory of the selected trajectory, and third slightly adjusting the sampled sub-trajectory. We say that an answer is returned correctly by a query Q if the sampled sub-trajectory is returned. We vary the number of the segments in Q for our experiments and use two measures, namely "computation time" and "hit rate". The "computation time" means the time cost of computing the distance measure and the "hit rate" means the percentage of the queries that return answers correctly. The results are shown in Figure 2. According to Figure 2(a), the computation cost based on our EDS distance measure is significantly smaller than that based on other distance measures (the efficiency gap is about 10 times faster). According to Figure 2(b), the hit rate based on our EDS distance measure is higher than that based on other distance measures (the effectiveness gap is about 1.5 to 10 times better).

In conclusion, we propose a new and better measure EDS for sub-trajectory similarity search. An interesting direction is to design some indexing techniques for EDS distance measure.

4. REFERENCES

[1] L. Chen and R. Ng. On the marriage of lp-norms and edit distance. In *VLDB*, 2004.

[2] L. Chen, M. T. Özsu, and V. Oria. Robust and fast similarity search for moving object trajectories. In *SIGMOD, pages 491-502*, 2005.

[3] E. Frentzos, K. Gratsias, and Y. Theodoridis. Index-based most similar trajectory search. In *ICDE, pages 816-825*, 2007.

[4] J. G. Lee, J. Han, , and K. Y. Whang. Trajectory clustering: a partition-and-group framework. In *SIGMOD*, 2007.

[5] M. Vlachos, G.kollios, and D. Gunopulos. Discovering similar multidimensional trajectories. In *ICDE*, 2002.

[6] B.K. Yi, H.V. Jagadish, and C. Faloutsos. Efficient retrieval for similar time sequences under time warping. In *ICDE, pages 201-208*, 1998.