Similarity Search on Uncertain Spatio-Temporal Data

Johannes Niedermayer[1], Andreas Züfle[1], Tobias Emrich[1], Matthias Renz[1]
Nikos Mamoulis[2], Lei Chen[3], Hans-Peter Kriegel[1]
[1]Institute for Informatics, Ludwig-Maximilians-Universität München
{niedermayer,zuefle,emrich,renz,kriegel}@dbs.ifi.lmu.de
[2]University of Hong Kong, nikos@cs.hku.hk
[3]Hong Kong University of Science and Technology, leichen@cse.ust.hk

Abstract: In this work, we address the problem of similarity search in a database of uncertain spatio-temporal objects. Each object is defined by a set of observations ((time,location)-tuples) and a Markov chain which describes the objects uncertain motion in space and time. To model similarity - which is an important building block for many applications such as identifying frequent motion patterns or trajectory clustering - we employ the well-known Longest Common Subsequence (LCSS) measure, which becomes a distribution on uncertain spatio-temporal data (ULCSS). We show how the aligned version (without time shifting) of the ULCSS can be exactly computed in PTIME, which is also verified by extensive experiments.

1 Introduction

Similarity search on trajectory data has an increasing number of applications, especially after the widespread availability of location data, such as GPS tracks. Exemplarily, data analysis tasks such as identifying frequent motion patterns or trajectory clustering require finding objects that moved in a similar way or followed a certain motion pattern. A number of similarity measures have been proposed for trajectory data. One of these is the Longest Common Subsequence (LCSS). The LCSS between two trajectories (i.e., moving objects) can be interpreted as the maximum amount of time the two objects were located at the same position. However, most of the previous work on similarity search in trajectory databases assumes the data to be certain or deterministic, which is not the case in many real applications. For example, even though we can get snapshots of the positions of a mobile object through RFID technology, the trajectory data is incomplete and uncertain: because the locations of the object between two consecutive RFID readers are unknown, they have to be derived from the observations which introduces uncertainty. Therefore, it is essential to develop new techniques to find similar trajectories on uncertain data. In this paper we study how the LCSS can be extended to apply to uncertain trajectories (ULCSS). Since in our scenario the exact motion of an object is unknown, we can model the ULCSS as a distribution of all possible LCSS results. The LCSS over uncertain data has many applications. For example, it can be used to evaluate the spread of flu or other diseases. Suppose that an object was diagnozed with a serious communicable disease (*source object*). To curtail such diseases, a facebook app could identify all individuals possibly been infected by the source object. Let us assume that enough virus cells are transmitted between two individuals if the two persons share the same location for at least k points in time. To identify individuals that might have been infected, the app could run an ULCSS query to find all objects having a large enough probability of being at the same location as the source object for at least k points in time, warning the affected persons.

2 Problem Definition

A spatio-temporal database $\mathcal{D}$ stores triples (*oid*, *location*, *time*), where $oid \in \{o_1, ..., o_{|\mathcal{D}|}\}$ is a unique object identifier, $location \in \mathcal{S}$ is a spatial position and $time \in \mathcal{T}$ is a point in time. Semantically, each such triple corresponds to an *observation* that object o_i has been seen at some location at some time. In $\mathcal{D}$, an object can be described by a function $tr_{o_i} : \mathcal{T} \rightarrow \mathcal{S}$ that maps each point in time to a location in space; this function is called *trajectory*. In this work, we assume a discrete time domain $\mathcal{T} = \{0, \ldots, n\}$. Thus, a trajectory becomes a sequence, i.e., a function on a discrete and ordinal scaled domain. Furthermore, we assume a discrete state space of possible locations (*states*) $\mathcal{S} = \{s_1, ..., s_{|\mathcal{S}|}\} \subset \mathbb{R}^d$.

Uncertain Trajectory Model. Since we consider uncertainty, a trajectory may not be modeled by a simple single path but rather by a (possibly large) set of paths, i.e., a set of possible worlds. In particular, let $\mathcal{D} = \{o_1, ..., o_{|\mathcal{D}|}\}$ be a database containing $|\mathcal{D}|$ uncertain moving objects. For each object $o \in \mathcal{D}$ we store a set of observations $\Theta^o = \{< t_1^o, \theta_1^o >, < t_2^o, \theta_2^o >, \ldots, < t_{|\Theta^o|}^o, \theta_{|\Theta^o|}^o >\}$ where $t_i^o \in \mathcal{T}$ denotes the time and $\theta_i^o \in \mathcal{S}$ the location of observation Θ_i^o. W.l.o.g. let $t_1^o < t_2^o < \ldots < t_{|\Theta^o|}^o$. According to [EKM+12], we can interpret the location of an uncertain spatio-temporal object $o \in \mathcal{D}$ at time t as a realization of a random variable $o(t)$. Given a time interval $[t_0, t_1]$, the set of corresponding uncertain locations becomes an uncertain trajectory. A formal definition of uncertain trajectories can be found in [EKM+12].

This technique allows us to assess the probability of a possible trajectory (i.e., a realization of all random variables). In this work we follow the approaches from [EKM+12] and employ the first-order Markov Chain model as a specific instance of a stochastic process. Markov Chains have been employed for modelling human movement in [MJS11]; furthermore, according to [EKM+12] they enable query processing of a class of queries in polynomial time while following possible worlds semantics. A Markov chain T is a matrix containing the conditional *transition probabilities* $T_{ij}^o(t) := P(o(t+1) = s_j | o(t) = s_i)$ of o from state s_i to state s_j at a given time t. Let $\vec{s}^o(t) = (p_1, \ldots, p_{|\mathcal{S}|})^T$ be the distribution vector of an object o at time t, where $\vec{s}_i^o(t) = P(o(t) = s_i)$. The distribution vector $\vec{s}^o(t+1)$ can be inferred from $\vec{s}^o(t)$ as follows: $\vec{s}^o(t+1) = T^o(t)^T \cdot \vec{s}^o(t)$

Similarity between Uncertain Trajectories. Given two database objects o_1 and o_2, our goal is to assess the similarity between these two uncertain objects by employing the LCSS [VGK02]. Let A and B be two trajectories of moving objects with size n and m respectively, where $A = (a_1, \ldots, a_n)$ and $B = (b_1, \ldots, b_m)$. Let $Head(A) := (a_1, \ldots, a_{n-1})$. Given an integer δ and a real number ϵ, the Longest Common Subsequence is defined as follows: $LCSS_{\delta,\epsilon}(A, B) :=$

$$\begin{cases} 0 \text{ if } A = \emptyset \text{ or } B = \emptyset, \\ 1 + LCSS_{\delta,\epsilon}(Head(A), Head(B)) \quad \text{if } dist(a_n - b_m) < \epsilon \text{ and } |n - m| \leq \delta \\ max(LCSS_{\delta,\epsilon}(Head(A), B), LCSS_{\delta,\epsilon}(A, Head(B))) \quad otherwise \end{cases}$$

Parameter ϵ constraints the spatial distance between two locations in order to match in space; in many applications, objects rarely visit the same locations, but being "close enough" is equivalent to meeting. Parameter δ controls how far in time we can expand in order to match a given point from one trajectory to a point in another trajectory. In this work, we aim at working towards efficiently computing $LCSS_{\delta,\epsilon}(o_1, o_2)$ for two uncertain trajectories o_1 and o_2. According to possible world semantics, the result of $LCSS_{\delta,\epsilon}(o_1, o_2)$ is not a single scalar, but rather a probability density function on $\mathbb{N}$, mapping each possible outcome $k \leq min(length(o_1), length(o_2))$ to a probability $P(LCSS_{\delta,\epsilon}(o_1, o_2) = k)$.

Definition 1 (ULCSS) *Let o_1 and o_2 be two uncertain trajectories. The Uncertain Longest Common Subsequence (ULCSS) between o_1 and o_2 is a random variable, defined by the following probability density function:*

$$ULCSS_{\delta,\epsilon}(o_1, o_2) : \mathcal{D} \times \mathcal{D} \to (\mathbb{N} \to [0, 1] \in \mathbb{R})$$

$$ULCSS_{\delta,\epsilon}(o_1, o_2) := pdf(x \in \mathbb{N}) = P(LCSS_{\delta,\epsilon}(o_1, o_2) = x)$$

However, calculating the exact distribution and the expected value of the length of the LCSS between two random sequences can currently only be achieved by employing exponential algorithms [FL08]. Therefore, we study the special case of ULCSS, where $\delta = 0$; for this case, we propose a PTIME algorithm for its exact computation:

Definition 2 (UALCSS) *Let o_1, o_2 be two uncertain trajectories. The Uncertain Aligned Longest Common Subsequence is defined by $UALCSS_{\epsilon}(o_1, o_2) = ULCSS_{0,\epsilon}(o_1, o_2)$.*

3 Related Work

The stochastic model used in this paper is taken from [EKM$^+$12], where window queries in an uncertain setting have been addressed. Existing approaches for measuring trajectory similarity mainly adapt to trajectories in a certain setting. [VGK02] relaxed this assumption by taking the effect of noise into account. Uncertain trajectory similarity has been investigated in the context of trajectory clustering by [PKK$^+$09]. However this work addresses *position* (e.g. GPS) uncertainty instead of *motion* uncertainty, where neither position nor motion are known at a given point in time, but might have been known at some time in the past. Recently, [LG12] addressed the problem of computing the windowed LCSS on strings. However, the different characters in a word are drawn independently in this context, whereas the state at time t depends on the previous state in our problem setting. Statistical work such as [AW85] provides statistical approximations of the length of the (unaligned) longest common subsequence on the Markov model, e.g. under the assumption that the underlying Markov chains are aperiodic and irreducible and if the length of the underlying sequence is large. Another common application area of Markov models is in the area of bioinformatics where gene sequences have to be matched (e.g.[HK96]). When employing hidden Markov models, viterbi-like algorithms and extensions that can handle insertions and deletions (c.p. e.g. [AV98]) are usually employed for computing the maximum likelihood, and not a distribution. Besides their advantage of estimating the *edit distance* between sequences, these approaches can only be used to match sequences to Markov chains but not two Markov chains. There further exists an exponential approach for calculating the exact distribution and the expected value of the length of the LCSS between two random sequences [FL08]. This algorithm neither considers observations, nor the amount of time shifting δ. Furthermore, the size of its transition matrix depends on the length of the time interval for which the LCSS has to be computed.

4 UALCSS Computation

Overview. While it remains unsolved how to compute the exact ULCSS in the general case, in this section we show how to exactly compute the UALCSS (ULCSS for the special case of $\delta = 0$) between two uncertain spatio-temporal objects, which represents the pdf over all possible lengths of the LCSS between the two evaluated objects with a polynomial time algorithm. The UALCSS is relevant to many spatial applications, like the infection application mentioned in the introduction; virus particles in a droplet infection can only be spread through space, but not through time.

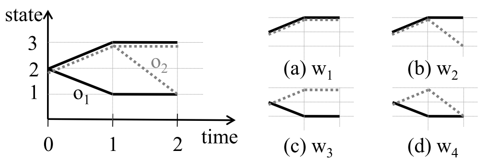

Figure 1: Possible worlds $\{w_{1-4}\}$ of two uncertain objects.

Figure 1 (left) shows the uncertain trajectory of objects o_1 (represented by the solid line) and o_2 (represented by the dotted line). From these, we derive four possible worlds as illustrated in Figure 1 (right). We can see that in w_1 the (certain) LCSS equals 3, in worlds w_2 and w_4 LCSS=2, while in w_3 LCSS=1. If we assume, in this example, that for each object each alternative trajectory has a probability of 0.5, we get a probability vector [0,0.25,0.5,0.25] for the UALCSS, where the kth element in the list denotes k hits between two paths. Clearly, such an approach of enumerating all possible worlds, and aggregating their probabilities is not a viable option, since in general, the number of possible trajectories of an uncertain trajectory is exponential in the length of the uncertain trajectory.

Algorithm 1 UALCSS(o_1, o_2, t_{max})

```
1:  $M^0 = \vec{s}^{o_2}(0) \cdot \vec{s}^{o_1}(0)^T$
2:  $M^1_{i \neq j} = 0$
3:  $M^1_{ii} = M^0_{ii}$
4:  $M^0_{ii} = 0$
5:  for $t = 1; t \leq t_{max}; t++$ do
6:      for $k = t; k \geq 0; k--$ do
7:          $M^k = T^{o_1}(t-1)^T \cdot M^k \cdot T^{o_2}(t-1)$
8:          $M^{k+1}_{ii} = M^{k+1}_{ii} + M^k_{ii}$
9:          $M^k_{ii} = 0$
10:     end for
11:     if $\exists t^{o_1}_i : t^{o_1}_i = t \vee \exists t^{o_2}_j : t^{o_2}_j = t$ then
12:         reweight($\{M^k\}$, $\theta^{o_1}_i$, $\theta^{o_2}_j$)
13:     end if
14: end for
15: $p = Array[t_{max} + 1]$
16: for $t = 0; t \leq t_{max}; t++$ do
17:     $p_k = |M^k|_{L_1}$
18: end for
19: return $p$
```

Algorithm. Algorithm 1 is a pseudocode of the UALCSS algorithm. For computing UALCSS, we have to take certain dependencies of the two objects into account, i.e., the relative position of o_1 to o_2 at time $t = 0$ (w.l.o.g. we assume that the first observations of o_1 and o_2 are at time $t = t^{o_1}_1 = t^{o_2}_1 = 0$) will affect the length of the UALCSS at a later time $t > 0$. For this reason, we have to take the conditional probabilities of object o_1 being in state s_i when o_2 is in state s_j into account: At the initial time $t = 0$ we assume both object locations to be independent, and therefore we can write $P(o_1(0) = s_i \wedge o_2(0) = s_j) = P(o_1(0) = s_i) \cdot P(o_2(0) = s_j)$. Let $M(t)$ be a probability matrix with $M_{ij}(t) = P(o_1(t) = s_i \wedge o_2(t) = s_j)$, denoting that the corresponding objects are within state s_i and s_j at time t. The matrix $M^0(0)$ can be easily computed as follows (the superscript 0 denotes the numbers of hits gained so far): $M^0(0) = \vec{s}^{o_2}(0) \cdot \vec{s}^{o_1}(0)^T$. This is the case because we have $M^0_{ij}(0) = \vec{s}^{o_1}(0)_i \cdot \vec{s}^{o_2}(0)_j = P(o_1(0) = s_i) \cdot P(o_2(0) = s_j)$. The elements $M^0_{ii}(0)$ denote the probabilities that both uncertain objects are located within the same state i at time 0, increasing the longest common subsequence by 1, such that these *possible worlds* have to be marked. This can be simply achieved by moving them into a second matrix $M^1(0)$, where $M^1_{ii}(0) = M^0_{ii}(0)$ and $M^1_{ij}(0) = 0$ for $i \neq j$. Besides, the shifted elements have to be deleted from $M^0(0)$ by performing $M^0_{ii}(0) = 0$. Now both matrices contain possible worlds, split by their number of hits.

After initialization, this method can be applied in a similar manner to compute the equivalence classes of possible worlds within each time $t \neq 0$, which is achieved by updating all state matrices $M^k(t-1)$. As a first step, the states of o_1 and o_2 in $M^k(t-1)$ have to be transitioned. Given a state vector $\vec{s}^{o_i}(t-1)$, this transition is usually performed by multiplying $\vec{s}^{o_i}(t-1)$ with its corresponding, pre-determined, transition matrix $T^{o_i}(t-1)$, i.e., $\vec{s}^{o_i}(t) = T^{o_i}(t-1)^T \cdot \vec{s}^{o_i}(t-1)$. However, in our scenario, we do not have a single state vector, but a state matrix $M^k(t-1)$, containing conditional probabilities of both objects. The elements in this matrix have to be transitioned according to both transition matrices $T^{o_1}(t-1)$ and $T^{o_2}(t-1)$. It can be proven that $M^k(t) = T^{o_1}(t-1)^T \cdot M^k(t-1) \cdot T^{o_2}(t-1)$.

After performing the transition, the matrix element $M^k_{ij}(t)$ again contains the conditional probabilities at time t that o_1 is in state j while o_2 is in state i. After transitioning, the hits are extracted from the matrix, by shifting the diagonal elements to $M^{k+1}_{ii}(t)$ and removing them from $M^k_{ii}(t)$. Therefore each of the t transitions leads to at most one additional matrix; thus, the total space complexity of this algorithm is at most $O(|S|^2 \cdot t)$ and the runtime complexity is $O(\Delta t^2 \cdot |\mathcal{S}|^3)$, with Δt beeing the number of considered timesteps. In practice, these costs are much lower since vectors $\vec{s}^{o_i}(t)$ and $T^{o_i}(t)$ are both sparse and we can save space and computations by employing sparse matrix operations on compressed representations. After having completed t transitions, we can derive the probability distribution for the relative frequency of worlds that had a given number k of hits: $P(|\{x \in \mathcal{T} | o_1(x) = o_2(x)\}| = k) = \sum_{\forall i,j} M^k_{ij}(t)$ Incorporating further observations (function *reweight()* in Algorithm 1) can be achieved as follows. Let us first assume that o_1 was observed at state s. Then all columns $j \neq s$ in M^k have to be set to 0 and all matrices have to be reweighted such that $\sum_{x \in t} M^k = 1$. Accordingly, if o_2 has been observed at a given state, the corresponding rows have to be set to zero. Furthermore, the algorithm can be easily adapted for $\epsilon > 0$. In this case, not only diagonal elements from M^k have to be shifted, but also further matrix elements that correspond to locations with a distance $\leq \epsilon$ to a given location of an uncertain object.

5 Experiments

The experiments are based on a discrete state space in the two-dimensional Euclidean space, consisting of n states. Each of these states is drawn uniformly from the $[0,1]^2$ space. Afterwards, a graph was created from these states by connecting points with an Euclidean distance smaller than $r = \sqrt{\frac{b}{n * \pi}}$, with b being the average number of neighbours of a state, i.e. the branching factor. The graph's edge weights, i.e. the transition probabilities were assigned indirectly proportionally to the distance of a state to its neighbour, assuming that it is more probable that during a transition an object moves to a closer state than to

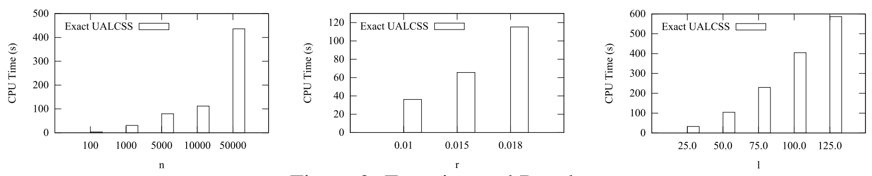

Figure 2: Experimental Results

a state further away. Based on the resulting transition matrix, a random trajectory was drawn to construct (certain) observations of an uncertain object, and every i-th point from this trajectory was used as an observation of the uncertain object. In the evaluation, we varied the number of states n from 100 to 50K (default 10000), the length l from 25 to 125 (default 50), and the range r from 0.01 to 0.075 (default $\sqrt{\frac{b}{n*\pi}}$). The interval between two observations is 10 timestamps. In the first experiment we aimed at varying the worlds *size* (n), keeping the graph's branching factor constant, while increasing r (the next experiment) can be interpreted as increasing the *resolution* of a world, i.e. the branching factor. As shown in Figure 2 (left), with increasing n, matrix operations become more costly such that the performance of the UALCSS drops. Varying the range of connectivity r (Figure 2 (center)) clearly shows a negative impact on the performance of the UALCSS algorithm. With a higher connectivity, the filling degree of transition matrices increases, such that more states can be reached in a shorter amount of time. Increasing the length of the time interval for which the UALCSS has to be computed (Figure 2 (right)) increases the number of iterations such that more matrix multiplications have to be performed. Note that the number of matrix multiplications for this algorithm is $O(t^2)$.

To compute the general ULCSS, we plan to investigate sampling techniques. The main problem for sampling approaches is to incorporate observations.

References

[AV98] J. C. Amengual and E. Vidal. Efficient Error-Correcting Viterbi Parsing. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20:1109–1116, 1998.

[AW85] Richard Arratia and Michael S. Waterman. An Erdös-Rényi Law with Shifts, 1985.

[EKM$^+$12] T. Emrich, H.-P. Kriegel, N. Mamoulis, M. Renz, and A. Züfle. Querying Uncertain Spatio-Temporal Data. In *Proc. ICDE*, 2012.

[FL08] J. C. Fu and W. Y. Wendy Lou. Distribution of the length of the longest commmon subsequence of two multi-state biological sequences. *Journal of Statistical Planning and Inference*, 138:3605–3615, 2008.

[HK96] Richard Hughey and Anders Krogh. Hidden Markov models for sequence analysis: extension and analysis of the basic method, 1996.

[LG12] Zheng Li and Tingjian Ge. Online windowed subsequence matching over probabilistic sequences. In *Proc. SIGMOD*, pages 277–288, 2012.

[MJS11] Arezu Moghadam, Tony Jebara, and Henning Schulzrinne. A markov routing algorithm for mobile DTNs based on spatio-temporal modeling of human movement data. In *Proc. WSIM*, 2011.

[PKK$^+$09] Nikos Pelekis, Ioannis Kopanakis, Evangelos E. Kotsifakos, Elias Frentzos, and Yannis Theodoridis. Clustering Trajectories of Moving Objects in an Uncertain World. In *Proc. ICDM*, pages 417–427, 2009.

[VGK02] Michail Vlachos, Dimitrios Gunopulos, and George Kollios. Discovering Similar Multidimensional Trajectories. In *Proc. ICDE*, pages 673–684, 2002.