Similarity of Uncertain Trajectories

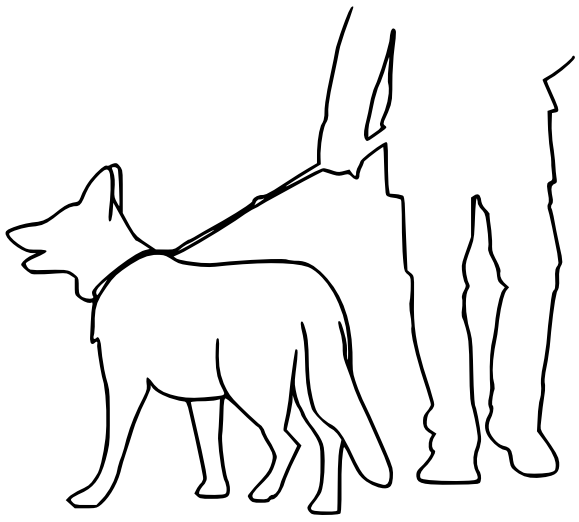
Popov, Aleksandr

Award date:
2019

Similarity of Uncertain Trajectories

Master thesis

Aleksandr Popov



Supervisors / Committee:
dr. Kevin Buchin
dr. Marcel Roeloffzen
dr. Maarten Löffler

Thanks for the permission to use the titlepage drawing to Mariana Souza.

Abstract

A trajectory is a sequence of time-stamped locations, each of which is represented as a point in two or three dimensions. An uncertain trajectory replaces each point, according to some uncertainty model, with a probability distribution, a disk, a set of points, etc. This is used to express uncertainty about the true location of an object and handle measurement imprecision transparently.

Fréchet distance is a commonly used trajectory similarity metric, often introduced as the smallest length of a leash needed for a leashed dog and its owner to walk along their two respective trajectories without going backwards.

In this thesis, we discuss similarity of uncertain trajectories, with the focus on region-based uncertainty models (disks, line segments, sets of points) and discrete and continuous Fréchet distance.

We show that finding the upper bound on both the discrete and continuous Fréchet distance on uncertain trajectories is an NP-complete problem in many settings. We also study expected discrete and continuous Fréchet distance, assuming uniform distribution of points within the respective regions, and show that finding it in many settings is #P-hard.

Finally, we discuss the setting with time bands, where we enforce temporal similarity of the two trajectories, thus decreasing the complexity, and give the polynomial-time algorithms to find upper bound and expected value of discrete and continuous Fréchet distance in this setting for trajectories with uncertainty model of sets of points.

Contents

1

Introduction

Trajectory data is very common. It arises in all kinds of settings: tracking animals, investigating movement of particles, or analysing someone's workout all involve trajectory data. Most commonly trajectory data is represented as a sequence of time-stamped locations in two or three dimensions with associated data (see Figure 1.1 (left)). It is not surprising that many methods for analysing trajectory data have been developed. Moreover, these methods help address many different analysis goals, some of the most common ones being quantifying the similarity of two trajectories, clustering trajectories, finding a trajectory representative of a group, simplifying a trajectory, or segmenting a trajectory based on some useful criteria.

These analysis goals can be quite nuanced depending on the application area. For instance, if one wants to compare trajectories of two racing drivers, time has to be taken into account to say anything meaningful, as the speed has a large impact on the trajectory of the car. In other settings, however,

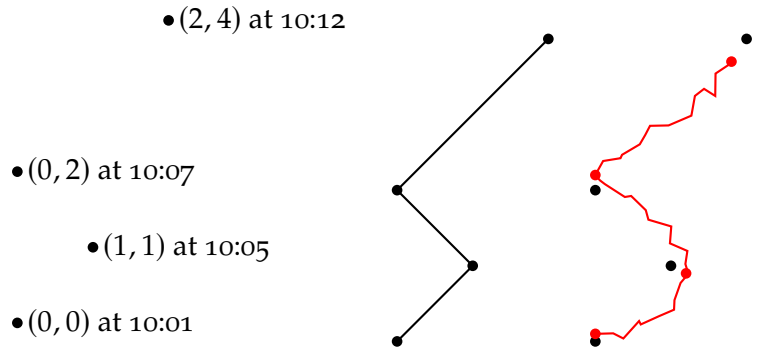


FIGURE 1.1: Left: Trajectory data. Centre: Polygonal curve on the data. Right: Uncertain trajectory and measured data.

time may not be of interest—consider the trajectories of people walking through a check-in hall at the airport. Most likely, speed and relative timing are not so relevant here, but rather the distribution of people through the various parts of the hall is of interest. If many trajectories are involved, one may get a better overview by performing clustering and selecting some trajectories as representatives for the clusters for further analysis. If one instead considers a workout tracker, segmentation is important—one would want to separate cycling, running, swimming, and taking a bus into separate parts and process them differently. So, in a variety of scenarios, trajectory analysis is called for.

In many of these real-life scenarios, however, data actually comes from GPS trackers or base radio station triangulation, so the position recorded for each data point has some inherent uncertainty. For example, in Figure 1.1 (right), the recorded (black) data points are all off the actual (red) trajectory due to measurement imprecision. On the scale of this example, imprecision might turn out to be significant. Coupled with the fact that the sampling rate may be somewhat low and that the position of an object between the measurements is inherently uncertain, the entire trajectory of an object may be difficult to pinpoint, and hence tricky to analyse.

However, most existing approaches to handling trajectories tend to ignore data imprecision and other sources of uncertainty, simply treating trajectories as polygonal curves, or even just as a discrete set of points. See Figure 1.1 (centre) for an example of a polygonal curve on data points; note that the curve is very different from the actual trajectory shown in red in Figure 1.1 (right). If there is little uncertainty relative to the scale of analysis, then this approach may still yield passable results; however, when the imprecision is significant, any analysis might only be meaningful if uncertainty is taken into account in any further calculations in a transparent manner.

Therefore, either new methods should be developed, or the existing ones should be adapted, to treat uncertain trajectories. Doing this in the context of similarity of trajectories is the focus of this thesis.

In particular, in this thesis we focus on a specific similarity measure developed for precise trajectories and, taking several simple models for measurement uncertainty, study how the uncertainty propagates to the similarity measure, establishing lower bounds, upper bounds, and expected values and either giving the algorithms to solve the problems or show corresponding hardness results.

To discuss the contributions of the thesis in more detail, we first need to establish how the similarity between trajectories can be measured (Sec-

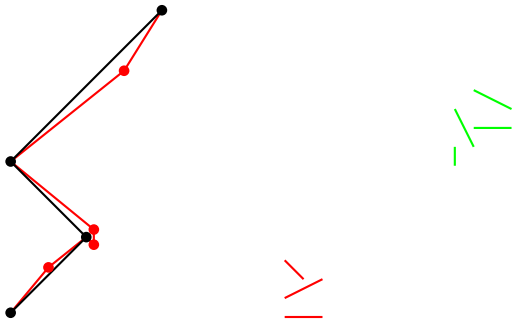


FIGURE 1.2: Left: Black curve is the simplification of the red curve. Note that they are similar. Right: Trajectories clustered in red and green clusters. Note that each trajectory is similar to others in its cluster.

tion 1.1), and how to take uncertainty into account in geometric algorithms in general (Section 1.2) and in algorithms for trajectory similarity in particular (Section 1.3).

1.1 Similarity of Trajectories

Various flavours of trajectory analysis show up in many different application areas. Most of the analysis goals fall into a few groups, with the benefit that one can develop the algorithms without tying them to a specific application area. As mentioned previously, the most common tasks are similarity, simplification, segmentation, and clustering.

Simplification of a trajectory is the process of reducing the complexity of the trajectory without losing specific information. There are several algorithms for curve simplification that do not take time into account, that is, they reduce a curve to a *similar* curve by omitting some points from the curve. See Figure 1.2 for an example: the black trajectory is a simplification of the red one, as it uses a subset of its vertices and is simpler; note that the two trajectories are similar, or the simplification would be of low quality.

A simple algorithm that works well in practice is the Ramer–Douglas–Peucker algorithm [18, 34]. Another commonly used algorithm is due to Imai and Iri [27]. It is a more involved algorithm with guarantees on the quality of the simplification. Note that the concept of *similarity* plays an important role here, since we need to quantify how similar the simplified trajectory is to the original. The two algorithms so far use *Hausdorff distance* for the error measure. The algorithm by Agarwal et al. [2] can use the *Fréchet distance* instead; the difference will be explained further, but usually it is a more natural measure to use for trajectories. It also provides an

approximation rather than an optimal solution, but the trade-off is that it does so much faster than the Imai–Iri algorithm. The latter can also be adapted to use Fréchet distance.

Recently there has been work by Bringmann and Chaudhury [9] providing an algorithm for a different version of simplification problem that runs asymptotically as fast as the Imai–Iri algorithm; they also give a conditional lower bound showing that at least in high dimensions Imai–Iri and their new algorithm are optimal.

All of these algorithms only do curve simplification; there are also approaches that aim to take time into account, so e.g. information about the speed changes is not lost [26], as well as other extensions of the basic algorithms, but they are not the focus of this overview.

Segmentation of a trajectory is the process of identifying subtrajectories that fit some criteria of interest, such as bounds on speed, direction, etc. For monotone criteria a greedy approach can be applied; for more complicated criteria or combinations thereof one can apply the algorithm based on the so-called start–stop diagram due to Aronov et al. [7]. The basic idea is that, based on the criteria, one can build a diagram with *free space* that is open to traverse and blocked cells, and then construct a specific trajectory through that diagram to find the segmentation. More recently, Alewijnse et al. [4] have suggested a general criterion-based segmentation framework that runs in $O(n \log n)$ time as opposed to $\Theta(n^2)$ in previous work. Alewijnse et al. [5] also study segmentation (and classification) of trajectories incorporating uncertainty in the movement between the measured points on the trajectory.

Clustering of multiple trajectories is the process of assigning a label to each trajectory, thus grouping them according to some *similarity metric*. See the example in Figure 1.2: two clusters of trajectories are clearly visible, and red trajectories are more similar to each other than to green trajectories.

Commonly used clustering algorithms include variants of k-means clustering that accept arbitrary distance metrics and DBSCAN [21]; the most interesting part of applying them to the trajectories is the choice of the underlying metric that should show how *similar* two trajectories are. One could also use (k, ℓ)-centre clustering, where the algorithm needs to find k cluster centre trajectories of complexity at most ℓ, minimising some distance metric between each clustered trajectory and its cluster centre [12].

One may notice that most tasks described so far internally use the concept of *similarity* of two trajectories. The similarity is quantified by some *measure* of similarity; for some applications, such as clustering, we would want to use a *metric* rather than any measure, so we need the triangle inequality to hold.

Similarity measures Many different measures of similarity of trajectories have been proposed. One measure that has been used quite often is *Hausdorff distance;* however, it is by no means specific to trajectories, as it completely ignores the sequence and instead views each trajectory as a set of points, so it does not provide intuitive results most of the time.

Perhaps the most commonly used measure is *dynamic time warping (DTW)* [30]. Its notable feature is the fact that it aligns the trajectories monotonously in time—if one were to imagine two objects moving along the trajectories and consider the link between them, DTW imposes the restriction that both objects have to traverse their respective trajectories from start to end, and that neither of them may at any point go backwards or jump over parts of the trajectory. However, it is sometimes inconvenient that DTW is not a metric, i.e. the triangle inequality does not hold.

Longest Common Subsequence (LCSS) adapted to trajectories traverses the two trajectories monotonously and considers points equivalent if they are within a certain distance from each other; the resulting value then is the maximum number of points that can be considered equivalent [36]. It can be a useful measure, as it is robust to outliers, since it ignores some points by design; however, it is again a non-metric.

Variants of *edit distance,* e.g. edit distance on real sequence, have been proposed, with the idea that the current point on a trajectory may be skipped, adding one edit; or it may be matched to the current point on the other trajectory, adding one edit if they are too far, or no edits if they are close enough [16]. It is thus quite similar to LCSS, and has the same benefits and issues.

Fréchet distance is very similar to DTW, except that it is a bottleneck measure, using the maximum instead of the sum of values; a crucial benefit of this measure is it being a metric, in addition to the benefit of it being a fairly natural fit to the trajectory similarity problem [6, 24]. A variant of this metric exists that only considers the vertices of the trajectory, called *discrete Fréchet distance;* it is easier to compute and is an upper bound on Fréchet distance, but it may yield misleading results if the sampling rate of the trajectories does not let the points align well [20]. Many variants of the Fréchet distance exist, including Fréchet distance with shortcuts [19] (where vertices may be skipped to make the measure more robust to outliers) and weak Fréchet distance [6] (where one can also go back along a trajectory, but cannot jump). Since the Fréchet distance appears to be the best fit, its extensions will be used as similarity measures in this thesis. Further details on Fréchet distance and discrete Fréchet distance, as well as the standard algorithms to find them, are provided in Section 2.1.

Time bands In the context of some of the preceding measures, especially DTW and Fréchet distance, it often makes sense to impose stricter requirements on the temporal alignment of trajectories. That can be achieved by using a *time band*, that is, by disallowing aligning points that are too far apart in time. The constrained measures are thus forced to only consider alignments that satisfy temporal limitations, ensuring that trajectories that are very similar, but are misaligned in time, will be perceived as dissimilar.

The most common type of a time band is the *Sakoe–Chiba band*, which simply restricts the aligning of point k on one trajectory to points $k \pm w$ on the other trajectory, for all k and for some fixed w [35]. Another commonly used time band is the *Itakura parallelogram;* as the name suggests, the time band resembles a parallelogram in shape, being narrow at the start and end of the trajectories and the widest in the middle [28]. Finally, a generalised version has been proposed specifically for use with Fréchet distance, *constrained free-space diagrams*, where the diagram is specified by defining the boolean constraints and storing the relevant attributes with the trajectories [10]. A time band applied on top of regular Fréchet distance can be seen as a specific instance of constraints. We use the Sakoe–Chiba band later in this thesis.

1.2 Uncertainty in Computational Geometry

Measurements are never exact, resulting in inherent uncertainty in measurement data. However, algorithms often assume precise input, not treating measurement uncertainty explicitly and assuming that it does not have an important effect on the result.

Treating uncertainty transparently is a rather new and different approach. Doing so requires establishing an *uncertainty model.* Consider, for now, some problem on a static set of points. If a point is uncertain, it can be often modelled as a region of possible point locations, or as a finite set of such locations, with any location inside the region being equally likely, and the locations outside the region impossible. Another approach would be to assign a probability distribution over the space under consideration to each uncertain point; there is some work by Löffler and Phillips [33] where the authors compute a distribution on the output of a shape fitting problem, given the uncertain points presented as probability distributions as input. While this approach provides more information, it is often difficult to deal with, so the region-based model is used more often. Based on that model with each imprecise point a disk of fixed radius, Cai and Keil [15] compute the visibility inside a simple polygon. In turn, Knauer et al. [29] study the

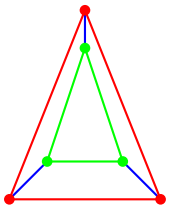


FIGURE 1.3: Lower and upper bound on the convex hull area on imprecise points modelled as line segments.

directed Hausdorff distance between imprecise point sets. More recently, Löffler [32, Chapter 3] provides an extensive overview of new algorithms and hardness results for lower and upper bounds on the output of smallest bounding box, smallest enclosing circle, and minimum width strip problems on imprecise points given as disks or squares. He also provides algorithms dealing with lower and upper bounds on convex hull area on imprecise points represented as disks, squares, and line segments [32, Chapter 4, 5], as well as bounds on diameter of a point set [32, Chapter 6] and bounds on the perimeter of a polygon [32, Chapter 11]. The approach of finding *lower bounds* and *upper bounds* on the output occurs quite often in general in the problems on uncertain points with region-based uncertainty. See the example for the realisations of the lower and upper bounds on convex hull area in Figure 1.3.

There are also some methods that use uncertainty models in the context of moving points. The model of imprecision here is that we can query the location of an object precisely, but it then moves, querying takes time, and we can (usually) only query one object at a time, so at any given moment we do not have a current precise location for most objects. Evans et al. [22] study the querying strategies in this model that would minimise uncertainty. When considering trajectories in this context, one could use the *Brownian bridge model*. The idea of that model is that the uncertainty can be represented as Brownian motion between a fixed starting point and a fixed end point. The model has been used to detect specific movement patterns by Buchin et al. [13]. There is also work using that model (or similar movement models) on segmentation and classification of trajectories [5].

1.3 Similarity of Uncertain Trajectories

There is surprisingly little work on similarity of trajectories with some uncertainty model. Ahn et al. [3] take the region-based imprecision model with each uncertain point modelled by a disk of a certain radius and find

the lower bound discrete Fréchet distance using a dynamic programming approach. They also present approximation algorithms both for the lower and upper bound discrete Fréchet distance under some extra assumptions. They stipulate that finding upper bound Fréchet distance is hard; it is confirmed by Fan and Zhu [23] for the case of thin rectangles as imprecision model and is further explored in this thesis.

Buchin and Sijben [14] approach the problem of computing discrete Fréchet distance on uncertain trajectories with points defined by probability distributions. As discussed in Section 1.1, many similarity measures for trajectories are based on *aligning* the trajectories according to some optimisation goal. In most previous work, when given the distributions of uncertain points, the aim is to come up with the distribution of the output of interest; in this case, it would be the distribution of the Fréchet distance. Buchin and Sijben propose a different approach: they give the distribution of the output conditioned on the *alignment* of trajectories, so for each alignment they can compute the distribution of the discrete Fréchet distance, and so their end goal is to find the alignment that has the largest probability to realise a certain discrete Fréchet distance. In a way, this approach swaps the choice of the alignment and the computation of the outcome for that alignment—this is different from what we focus on in this thesis.

1.4 Contributions

Setting In this thesis, we focus on Fréchet distance and discrete Fréchet distance as similarity measures, since they are metrics and provide natural results for trajectory similarity.

We choose the region-based imprecision model and make a distinction between *indecisive* points, where each point is a set of equiprobable real locations, and *imprecise* points, where each point is a closed region of equiprobable real locations. The latter can be of any shape, but we focus on disks and line segments. We do not consider any uncertain movement models between the measurements, instead assuming linear interpolation between real locations. These choices provide a convenient framework; in the future it would be, of course, interesting to also consider uncertainty for the movement between measurements.

Results We consider upper bound Fréchet distance, lower bound Fréchet distance, and *expected* Fréchet distance between trajectories. Note that

		indecisive	imprecise	
			disks	line segments
discrete FD	UB	NP-complete	NP-complete	NP-complete
	Exp	#P-hard	—	#P-hard
FD	UB	NP-complete	NP-complete	NP-complete
	Exp	#P-hard	—	—

Table 1.1: Summary of hardness results in this thesis.

the expected distance can be defined, as we essentially assume uniform distribution for the locations of uncertain points over the respective regions.

First of all, finding lower bound discrete Fréchet distance for imprecise points has been tackled by Ahn et al. [3]. Their approach can also be extended to indecisive points in a straightforward way.

In other areas, we get the following results:

1. NP-completeness results. As discussed before, there is very little work on uncertain trajectories. In the line of work by Ahn et al. [3] and Fan and Zhu [23], we show NP-completeness for upper bound on discrete and continuous Fréchet distance in several settings. See Table 1.1 for details.

2. #P-hardness results.[1] There is also very little work on *expected value* of any function on uncertain points. This seems to be a natural extension of previous work tackling lower and upper bounds. However, this turns out to be rather difficult: we show #P-hardness for expected value of discrete and continuous Fréchet distance in several settings. See Table 1.1 for details.

3. Algorithms for discrete and continuous Fréchet distance with Sakoe–Chiba time bands. Previous sets of results suggest that there is little room for positive algorithmic results in this setting. However, it turns out that in the setting with time bands the problems become feasible, and the limitations imposed by time bands may sometimes prove to be useful. In particular, we give algorithms to find, given *indecisive* trajectories, upper bound discrete and continuous Fréchet distance, as well as expected value of discrete and continuous Fréchet distance.

[1] Hardness class #P is a class of hard counting problems. For example, SAT ('Is there a satisfying assignment to a boolean formula?') is an NP-complete problem, whereas #SAT ('How many satisfying assignments to a boolean formula are there?') is a #P-complete problem.

Outline The rest of this thesis is organised as follows. In Chapter 2 we present some preliminaries regarding the Fréchet distance and discrete Fréchet distance, as well as uncertainty models and notation in use. In Chapter 3 we proceed to formally state and prove the hardness results mentioned above. In Chapter 4 we give the algorithms for upper bound and expected distance when using the time bands. Finally, we conclude the thesis and discuss future work in Chapter 5.

2

Preliminaries

It is necessary to provide some background information and introduce the notation before we proceed. In this chapter, we give the definition of Fréchet distance and discrete Fréchet distance and recall the standard algorithms to compute these metrics on precise (not uncertain) trajectories, introduced by Alt and Godau, and Eiter and Mannila, respectively [6, 20, 24]. Then we formally introduce the uncertainty model for imprecise and indecisive points and give an overview of the notation used in the rest of the thesis. The notation used is inspired by several works on Fréchet distance and imprecise point sets [3, 8, 29].

2.1 Fréchet Distance

Denote $[n] \equiv \{1, 2, \ldots, n\}$. Suppose we operate in d-dimensional Euclidean space. Consider a *sequence* of points in this space $P = \langle p_1, p_2, \ldots, p_n \rangle$. A *polygonal curve* P is defined by these points by linearly interpolating between the successive points and can be seen as a continuous function: $P(i + \alpha) = (1 - \alpha)p_i + \alpha p_{i+1}$ for $i \in [n - 1]$ and $\alpha \in [0, 1]$. The *length* of such a curve is the number of its vertices, $|P| = n$. We denote the *concatenation* of two polygonal curves P and Q of lengths n and m by $P \parallel Q$; the new curve follows P, then has a segment that connects $P(n)$ to $Q(1)$, and then follows Q. Similarly, $p \parallel q$ for any two points p and q denotes the line segment from p to q. We can generalise this notation:

$$\Big\|_{i \in [n]} p_i \equiv p_1 \parallel p_2 \parallel \cdots \parallel p_n \equiv P\,.$$

A *trajectory* is a polygonal curve with timestamps associated with each vertex of the curve.[1] Therefore, whenever timestamps are not relevant, trajectories and polygonal curves behave the same, and so we use the terms interchangeably. We denote a *subtrajectory* from vertex i to j of curve P as $P[i:j] \equiv p_i \parallel p_{i+1} \parallel \cdots \parallel p_j$.

Metrics definitions *Fréchet distance* is a metric on trajectories that is often intuitively introduced with the following example. Suppose a man is walking his dog, and they go along their respective trajectories, from start to end. Each of them must complete the entire trajectory, without jumping over bits and without going backwards (but they may stop). The dog is on a leash; the Fréchet distance is the smallest leash length necessary for them to be able to walk along their trajectories fully.

Discrete Fréchet distance is occasionally introduced using a similar analogy with leaping frogs, but the mechanics of it seems rather confusing; the difference with the regular Fréchet distance is that now we only consider the vertices on both trajectories and disregard the way the dog and its owner move between vertices.

The idea and the difference between the two versions can perhaps be best explained graphically: see Figure 2.1 and its caption for more intuition.

We shall now formally define the continuous and discrete Fréchet distance. Let Φ_n be the set of all continuous non-decreasing functions $\phi : [0,1] \to [1,n]$, and let Ψ_n be the set of all non-decreasing functions $\psi : [0,1] \to [n]$, with the additional restriction that functions in Ψ_n may not skip over any values in $[n]$. Another restriction is that $\phi(0) = \psi(0) = 1$ and $\phi(1) = \psi(1) = n$. We call pairs of such functions (ϕ_1, ϕ_2) and (ψ_1, ψ_2) *couplings* between trajectories. We denote the *discrete Fréchet distance* between precise trajectories A and B of lengths n and m, respectively, by $d_{\mathrm{dF}}(A,B)$ and define it as follows:

$$d_{\mathrm{dF}}(A,B) = \inf_{\psi_1 \in \Psi_n, \psi_2 \in \Psi_m} \max_{t \in [0,1]} \|A(\psi_1(t)) - B(\psi_2(t))\|,$$

where $\|\cdot\|$ denotes the Euclidean distance.

Similarly, we define *(continuous) Fréchet distance* between precise trajectories A and B of lengths n and m, respectively, as

$$d_{\mathrm{F}}(A,B) = \inf_{\phi_1 \in \Phi_n, \phi_2 \in \Phi_m} \max_{t \in [0,1]} \|A(\phi_1(t)) - B(\phi_2(t))\|.$$

[1] In general, it need not be a polygonal curve; however, as we assume linear interpolation between successive vertices in this thesis, it is easier to not make such distinction.

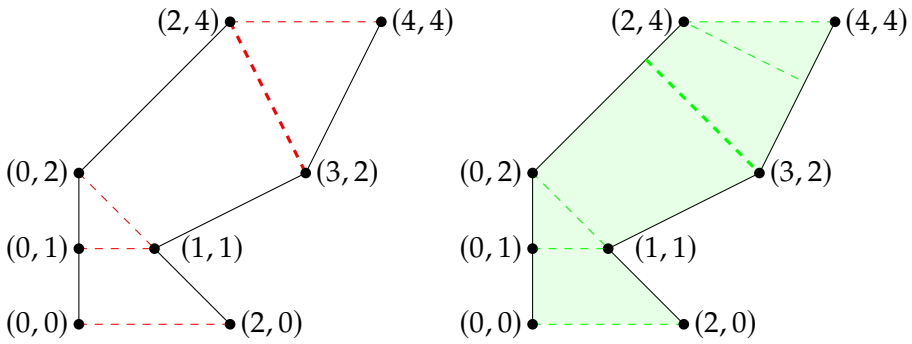


FIGURE 2.1: Left: Discrete Fréchet distance is computed when only considering the vertices. An optimal coupling is shown in dashed red lines, and the largest of them is marked thick and defines the resulting distance. Right: Fréchet distance is computed when considering entire trajectories. Some parts of an optimal coupling are shown with dashed lines; between them the coupling can be interpolated linearly. The line defining the result is marked thick; note that it does not connect two vertices. Observe that discrete Fréchet distance is always an upper bound for Fréchet distance.

So, Fréchet distance is the bottleneck distance for the best reparameterisation on the two trajectories, restricted to only moving forward on each trajectory.

When considering the discrete Fréchet distance, it is sometimes inconvenient (and usually unintuitive) to use the notation for couplings introduced earlier, even though it makes the definitions for the regular Fréchet distance and discrete Fréchet distance notably similar. In that case, we can consider a coupling on trajectories A and B of lengths n and m, respectively, to be a sequence of $r > 0$ elements from $[n] \times [m]$, $c = \langle (a_1, b_1), \ldots, (a_r, b_r) \rangle$, where a *valid* coupling also satisfies $(a_1, b_1) = (1, 1)$, $(a_r, b_r) = (n, m)$, and, for any $i \in [r-1]$,

$$(a_{i+1}, b_{i+1}) = \begin{cases} (a_i + 1, b_i)\,, \\ (a_i, b_i + 1)\,, \\ (a_i + 1, b_i + 1)\,. \end{cases}$$

Denote the set of all valid couplings on trajectories of lengths n and m as C and denote the length of a sequence $r = |c|$. Then we can give an equivalent definition of discrete Fréchet distance as

$$d_{\mathrm{dF}}(A, B) = \inf_{c \in C} \max_{s \in [|c|]} \|A(a_s) - B(b_s)\|\,,$$

where $c_s = (a_s, b_s)$ for all $s \in [|c|]$. This definition is convenient when one does not want to reason directly about pairs of functions with a continuous domain that have the same role as the sequence c in this definition. In

4	$\sqrt{10}$	$\sqrt{5}$	2
$2\sqrt{2}$	$\sqrt{2}$	3	$2\sqrt{5}$
$\sqrt{5}$	1	$\sqrt{10}$	5
2	$\sqrt{2}$	$\sqrt{13}$	$4\sqrt{2}$

4	$\sqrt{10}$	$\sqrt{5}$	$\sqrt{5}$
$2\sqrt{2}$	2	3	$2\sqrt{5}$
$\sqrt{5}$	2	$\sqrt{10}$	5
2	2	$\sqrt{13}$	$4\sqrt{2}$

Table 2.1: Left: Distance matrix on vertices for example of Figure 2.1. Right: Dynamic program for discrete Fréchet distance, filled from bottom left corner. Rows correspond to points from the left trajectory, columns—to points from the right trajectory. Optimal path is marked in grey.

words, a coupling is a relation on the vertices of two trajectories, where we never move to a previous vertex, couple all vertices, couple the first vertices to each other, and couple the last vertices to each other. We then find the bottleneck distance for each coupling and pick the distance stemming from the best coupling, so we get the bottleneck distance of the best coupling.

Discrete Fréchet distance optimisation algorithm This latter definition of discrete Fréchet distance is instructive to arrive to the dynamic programming algorithm by Eiter and Mannila, which is deduced in a standard manner from the following recursion:

$$\begin{aligned} d_{\mathrm{dF}}(A[1:i+1], B[1:j+1]) = \max(&\|A(i+1) - B(j+1)\|, \\ &\min(d_{\mathrm{dF}}(A[1:i], B[1:j]), \\ &\quad d_{\mathrm{dF}}(A[1:i+1], B[1:j]), \\ &\quad d_{\mathrm{dF}}(A[1:i], B[1:j+1]))) . \end{aligned}$$

In words, the discrete Fréchet distance is the maximum of the distance of the newly added element in the coupling and the value that was considered best previously. Due to the coupling restrictions, there are only three possible subproblems that we need to consider, and we may choose the best of them, thus obtaining the recursion above. It is straightforward to turn it into a dynamic program.

Table 2.1 gives the distance matrix and the computation of the discrete Fréchet distance for the example of Figure 2.1. Each cell of the table on the right shows the value of the discrete Fréchet distance so far; the final result can be read out from the top right corner of the table, and the coupling that yields this result can be read from the sequence of grey cells. Notice that the table shows the same coupling as Figure 2.1.

So, there is a recursion that yields a dynamic programming solution to discrete Fréchet distance on precise trajectories. Given two trajectories of length n and m in d dimensions, this solution takes $\Theta(dmn)$ time to run.

4	$\sqrt{10}$	$\sqrt{5}$	2
$2\sqrt{2}$	$\sqrt{2}$	3	$2\sqrt{5}$
$\sqrt{5}$	1	$\sqrt{10}$	5
2	$\sqrt{2}$	$\sqrt{13}$	$4\sqrt{2}$

False	False	True	True
False	True	False	False
True	True	False	False
True	True	False	False

TABLE 2.2: Left: Distance matrix on vertices for example of Figure 2.1. Right: Dynamic program for decision version of discrete Fréchet distance with threshold $\varepsilon = 2.5$.

More recently, Agarwal et al. [1] presented an algorithm that computes discrete Fréchet distance in time $O\left(\frac{mn \log\log n}{\log n}\right)$ in two dimensions, for $m \leq n$. However, it is rather complex and does not help the intuition about the problems discussed in this thesis, so we will not go into further detail.

Discrete Fréchet distance decision algorithm Note that here we were solving the *optimisation* problem—finding the actual value of discrete Fréchet distance. Instead, one could pose a *decision* problem—check whether the value of discrete Fréchet distance is below a certain threshold. Obviously, we can solve it with the same algorithm, except now we can store the values of True and False. Table 2.2 shows the same distance matrix and the dynamic program for the decision problem for the threshold of $\varepsilon = 2.5$. Naturally, the decision version here still takes time $\Theta(dmn)$. The decision version of the algorithm by Agarwal et al. [1] mentioned earlier runs in time $O\left(\frac{mn \log\log n}{\log^2 n}\right)$ for $m \leq n$.

Fréchet distance decision algorithm One can use a similar approach to solve the decision version of the Fréchet distance problem, except now we have free and blocked areas within each cell of the table rather than simply having a boolean value in each cell. The resulting table is called a *free-space diagram.* On polygonal curves, each cell becomes an intersection of an ellipse with the cell, with the inside of the ellipse being free. The answer to the problem is True if and only if there is a monotone path from the bottom left corner to the top right corner of the free-space diagram. A free-space diagram for the example of the two polygonal curves of Figure 2.1 is shown in Figure 2.2.

Algorithmically this can be checked by keeping the open intervals on the edges of the cells, i.e. the white segments on cell borders shown in Figure 2.2. The algorithm then runs in time $\Theta(dmn)$. For further details the reader is invited to consult the work by Alt and Godau [6] or previous work on the same topic [24].

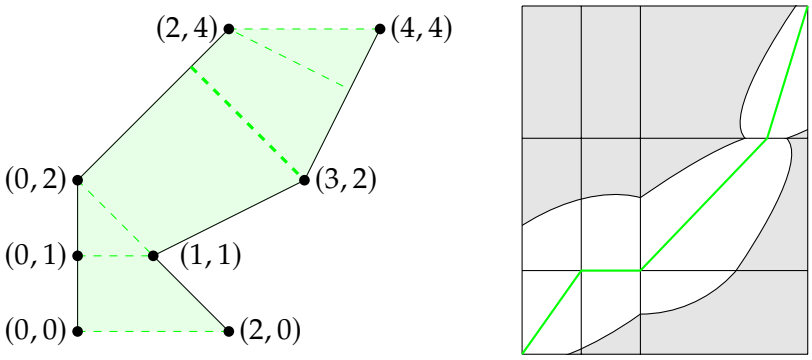


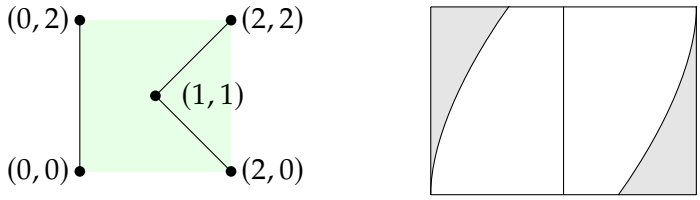
Figure 2.2: Left: Visualisation of Fréchet distance. Right: Free-space diagram for threshold $\varepsilon = 2.15$. One can draw a monotonous path from the lower left corner to the upper right corner of the diagram, so the Fréchet distance between trajectories is below the threshold.

Fréchet distance optimisation algorithm In principle, to solve the optimisation problem we need to find the value of ε where the solution to the decision problem turns from `True` to `False`. So, we need to find the threshold where the monotone path can no longer be drawn through the free-space diagram. Therefore, we need to consider the intervals we store; the solution will occur when the relative configuration of the intervals changes, so the path can no longer be drawn. The events when this occurs are:

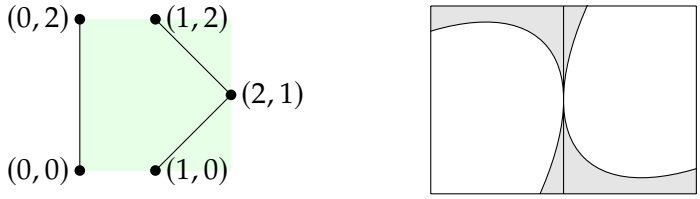
1. For lower ε the lower left and top right points are not in free space;
2. For lower ε there is no more vertical or horizontal passage between some two neighbouring cells;
3. For lower ε the monotone path cannot be drawn (vertically or horizontally) through two non-adjacent cells.

There can be one event of the first type, $O(mn)$ events of the second type, and $O(m^2n + mn^2)$ events of the third type. The events are illustrated in Figure 2.3. This way, we only need to consider a finite set of events, each of which corresponds to a certain value of the threshold ε that we can find in $\Theta(d)$ time. These values are called *critical values.*

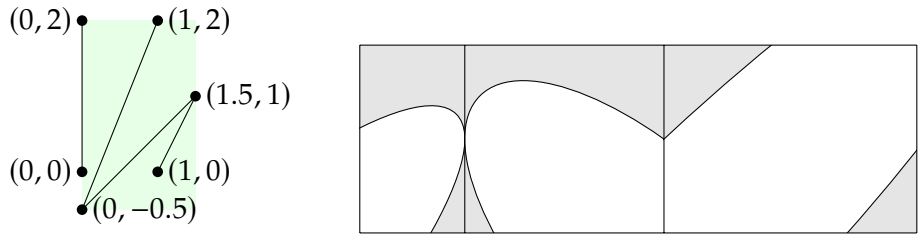
Having identified critical values, one can use binary search on the sorted critical values, executing the decision algorithm every time, to find the critical value that is the solution to the optimisation problem. Sorting the critical values takes $O((m^2n + mn^2)\log mn)$ time; doing the binary search takes $O(dmn \log mn)$ time, so the algorithm finds the optimal value in time $O((m^2n + mn^2)\log mn)$ [24], assuming $d \in O(\log mn)$.



(a) Critical event 1, for $\varepsilon = 2$. First and last point are just in free space.



(b) Critical event 2, for $\varepsilon = 2$. A passage just opens between neighbouring cells in free space.



(c) Critical event 3, for $\varepsilon = 1.5$. A horizontal passage just opens between non-neighbouring cells in free space.

FIGURE 2.3: Types of critical events, illustrated with trajectories and the corresponding free-space diagrams. In all three cases, for any lower ε there is no monotone path from the lower left corner to the upper right corner, and there is no reparameterisation of the trajectories yielding the distance between every pair of points of at most ε.

This can be improved by using Cole's variant of parametric search instead of binary search, obtaining the running time of $O(dmn \log mn)$ [6]. However, we do not discuss this technique further.

Buchin et al. [11] show that the running time might be slightly improved to $O(n^2\sqrt{\log n}(\log\log n)^{1.5})$ in the expected case on two trajectories of length n on a pointer machine, and to $O(n^2(\log\log n)^2)$ on a word RAM, assuming d is constant in both cases. However, it has been shown that no strongly subquadratic algorithms for Fréchet distance are likely to exist [8], so it is assumed that the bound cannot be considerably improved.

2.2 Uncertainty Model

Again, suppose we operate in d-dimensional Euclidean space. There are different uncertainty models; in general, uncertain points can be represented as probability distributions, but we take a simpler approach in this thesis. An *uncertain* point is commonly represented as a compact region $H \subset \mathbb{R}^d$. Usually, it is a finite set of points, a disk, a rectangle, or a line segment. The intuition is that only one point from this region represents the true location of the point; however, we do not know which one. A *realisation* of such a point h is one of the points from the region, chosen according to some probability distribution $\mathbb{P}$; we denote this as $h \in_{\mathbb{P}} H$.

An *indecisive* point is a special case of an uncertain point: it is a set of points $H = \{h_1, \ldots, h_k\}$, where each point $h_i \in \mathbb{R}^d$ for $i \in [k]$. Similarly, an *imprecise* point is a compact connected region $H \subset \mathbb{R}^d$. We will usually use disks as such closed regions, although we could also use rectangles, line segments, etc. We denote the disk with the centre at $p \in \mathbb{R}^d$ and radius $r \geq 0$ as $D(p, r)$. We denote the line segment between points p_1 and p_2 by $S(p_1, p_2)$. The probability distribution for indecisive points is discrete; the one for imprecise points is usually continuous. Note that a precise point is a special case of an indecisive point (set of size one) and an imprecise point (disk of radius zero).

2.3 Notation

Consider a sequence of uncertain points $\mathcal{H} = \langle H_1, \ldots, H_n \rangle$, referred to as an *uncertain trajectory* or an *uncertain curve.* A *realisation* of such uncertain trajectory is a polygonal curve $P = \langle p_1, \ldots, p_n \rangle$, each point of which is a realisation of the corresponding uncertain point; so, for all $i \in [n]$, we need $p_i \in_{\mathbb{P}} H_i$ for some fixed distribution $\mathbb{P}$. We denote the fact that P is a

realisation of $\mathcal{H}$ under distribution $\mathbb{P}$ by $P \Subset_{\mathbb{P}} \mathcal{H}$. If the distribution is clear from context, we sometimes omit it and simply write $P \Subset \mathcal{H}$.

Extending the notation to uncertain trajectories H and V, we consider the upper bound and the lower bound on the discrete Fréchet distance under different possible realisations, defined as follows:

$$d_{\mathrm{dF}}^{\max}(\mathcal{H}, \mathcal{V}) = \max_{A \Subset \mathcal{H}, B \Subset \mathcal{V}} d_{\mathrm{dF}}(A, B)\,,$$
$$d_{\mathrm{dF}}^{\min}(\mathcal{H}, \mathcal{V}) = \min_{A \Subset \mathcal{H}, B \Subset \mathcal{V}} d_{\mathrm{dF}}(A, B)\,.$$

Here we do not consider any specific distribution $\mathbb{P}$; the only important point is that no realisation should be completely excluded by $\mathbb{P}$.

Extending our notation for Fréchet distance to the uncertain trajectories, we can define

$$d_{\mathrm{F}}^{\max}(\mathcal{H}, \mathcal{V}) = \max_{A \Subset \mathcal{H}, B \Subset \mathcal{V}} d_{\mathrm{F}}(A, B)\,,$$
$$d_{\mathrm{F}}^{\min}(\mathcal{H}, \mathcal{V}) = \min_{A \Subset \mathcal{H}, B \Subset \mathcal{V}} d_{\mathrm{F}}(A, B)\,.$$

Finally, one may ask what will be the expected value of the Fréchet distance if the realisation is picked randomly according to $\mathbb{P}$, independently for each point on the trajectories. We call this the *expected Fréchet distance:*

$$d_{\mathrm{F}}^{\mathbb{E}(\mathbb{P})}(\mathcal{H}, \mathcal{V}) = \mathbb{E}_{A \Subset_{\mathbb{P}} \mathcal{H}, B \Subset_{\mathbb{P}} \mathcal{V}}[d_{\mathrm{F}}(A, B)]\,,$$
$$d_{\mathrm{dF}}^{\mathbb{E}(\mathbb{P})}(\mathcal{H}, \mathcal{V}) = \mathbb{E}_{A \Subset_{\mathbb{P}} \mathcal{H}, B \Subset_{\mathbb{P}} \mathcal{V}}[d_{\mathrm{dF}}(A, B)]\,.$$

As before, if the distribution is clear from the context, we might just write $d_{\mathrm{F}}^{\mathbb{E}}$ and $d_{\mathrm{dF}}^{\mathbb{E}}$ for the expected continuous and discrete Fréchet distance, respectively.

3

Hardness Results

In this chapter we discuss the hardness results for upper bound and expected value of continuous and discrete Fréchet distance in different settings, as defined in Chapter 2.

As mentioned previously, we do not consider lower bounds: Ahn et al. [3] have already covered the case of discrete Fréchet distance on imprecise points modelled as disks, and it is easy to cover the case with indecisive points. To do that, one simply should apply the same algorithm, however, instead of keeping track of intersections of disks, one needs to keep track of subsets of point sets.

As we show in this chapter, with a reasonably simple construction we can show that upper bound and expected case are hard. This means that we need to find other ways forward, and we discuss one such approach in Chapter 4.

3.1 NP-Completeness of Upper Bound Fréchet Distance

We present proofs of NP-hardness of determining $d_{\mathrm{F}}^{\max}$ and $d_{\mathrm{dF}}^{\max}$ exactly in several models by showing reductions from CNF-SAT (Satisfiability). In particular, we show that the problem is NP-hard in two-dimensional Euclidean space for both the indecisive and imprecise setting both for discrete and for continuous Fréchet distance.

We will construct two trajectories, one of them uncertain; one trajectory will correspond to the variables occurring in a CNF-SAT formula, and the other one will correspond to the clauses. Recall that when computing any variant of Fréchet distance, we pick an *alignment* or *coupling*. Because of

the way we construct trajectories, only some couplings can be optimal, and thus only those couplings need to be considered. In our construction any subtrajectory corresponding to a clause of the formula may be chosen to align with the variable trajectory, so we would traverse both at the same time linearly and map the rest of the formula trajectory to a point. If a clause is satisfied, then the Fréchet distance resulting from such coupling will be large. If some clause is not satisfied, we can pick that clause to align the subtrajectory with the variable trajectory, and the Fréchet distance will be low.

Therefore, if for all clauses the corresponding coupling yields a large Fréchet distance and no other couplings are possible, then the overall Fréchet distance will be large. On the other hand, in that case all clauses are satisfied, and so the formula is true for the given variable assignment. So, we get a large Fréchet distance if and only if a formula is satisfiable. The different realisations of the uncertain trajectories will then correspond to different assignments.

More formally, suppose we are given a CNF-SAT *formula* C with

$$C = \bigwedge_{i\in[n]} C_i\,, \qquad C_i = \bigvee_{j\in J\subseteq[m]} x_j \vee \bigvee_{k\in K\subseteq[m]\setminus J} \neg x_k \quad \text{for all } i \in [n].$$

Here n and m is the number of clauses and variables, respectively, and x_j for any $j \in [m]$ is a *boolean variable.* Such a variable may be assigned 'true' or 'false'; an *assignment* is a function $a : \{x_1, \ldots, x_m\} \to \{\mathsf{True}, \mathsf{False}\}$ that assigns a value to each variable, $a(x_j) = \mathsf{True}$ or $a(x_j) = \mathsf{False}$ for any $j \in [m]$. We denote by $C[a]$ the result of substituting $x_j \mapsto a(x_j)$ in C for all $j \in [m]$. To restate the problem,

> CNF-SAT
> Input: A CNF-SAT formula C.
> Output: 'Yes' if there is a variable assignment a such that $C[a] = \mathsf{True}$, 'No' otherwise.

As shown in the Cook–Levin theorem, any problem in NP can be reduced in polynomial time to CNF-SAT [17, 31].

Our goal is to demonstrate that by solving some problem of our choosing we obtain a solution to CNF-SAT in polynomial time. We define the following decision problem in the domain of interest:

> Upper Bound Discrete Fréchet
> Input: Two uncertain trajectories $\mathcal{H}$ and $\mathcal{V}$ and a threshold $t \in \mathbb{R}$.
> Output: 'Yes' if $d_{\mathrm{dF}}^{\max}(\mathcal{H}, \mathcal{V}) > t$, 'No' otherwise.

We can define a similar problem for the continuous Fréchet distance:

> Upper Bound Continuous Fréchet
> Input: Two uncertain trajectories $\mathcal{H}$ and $\mathcal{V}$ and a threshold $t \in \mathbb{R}$.
> Output: 'Yes' if $d_{\mathrm{F}}^{\max}(\mathcal{H}, \mathcal{V}) > t$, 'No' otherwise.

In the remainder of this section, we will show that it is possible, given an instance of CNF-SAT problem, to construct an instance of Upper Bound Discrete Fréchet and Upper Bound Continuous Fréchet and in both cases to use the solutions to those problems to construct a solution to CNF-SAT, and such construction can be done in polynomial time.

Construction for Discrete Fréchet Distance on Indecisive Points

Suppose we are given a CNF-SAT formula C with n clauses, denoted as $C_1, \ldots, C_n$, and m variables, $x_1, \ldots, x_m$. We will construct two indecisive trajectories in two-dimensional space. The gadgets are shown in Figure 3.1.

We define a *variable gadget* VG_j for $j \in [m]$ as a single indecisive point and a synchronisation point:

$$\mathrm{VG}_j = \{(0, 0.5 + \varepsilon), (0, -0.5 - \varepsilon)\} \parallel (2, 0)\,.$$

Here one can set $\varepsilon \in \mathbb{R}$ arbitrarily, as long as $0 < \varepsilon < 0.25$. We define an *assignment gadget* $\mathrm{AG}_{i,j}$ for each $i \in [n]$ and $j \in [m]$ as two precise points, where we refer to the first one as an assignment point and to the second one as a synchronisation point:

$$\mathrm{AG}_{i,j} = \begin{cases} (0, -0.5) \parallel (1, 0) & \text{if } x_j \text{ is a literal of } C_i, \\ (0, 0.5) \parallel (1, 0) & \text{if } \neg x_j \text{ is a literal of } C_i, \\ (0, 0) \parallel (1, 0) & \text{otherwise.} \end{cases}$$

We show a useful relation between the gadgets. To do so, we introduce the *one-to-one coupling* as a valid coupling $c = \langle (a_1, b_1), \ldots, (a_r, b_r) \rangle$, where the condition is restricted to $(a_{s+1}, b_{s+1}) = (a_s + 1, b_s + 1)$ for all $s \in [r - 1]$. Necessarily, such a coupling can only exist for trajectories of equal length.

Lemma 1. *Suppose we are given a clause C_i and a variable x_j that both occur in the CNF-SAT formula C, and we restrict the set of valid couplings C to only contain*

one-to-one couplings. We only get the discrete Fréchet distance equal to $1 + \varepsilon$ *if the realisation of* VG_j *we pick corresponds to the assignment of* x_j *that ensures the clause* C_i *is satisfied; otherwise, the discrete Fréchet distance is* 1*. In other words, if we consider* $A \Subset \mathrm{VG}_j$ *that corresponds to setting* $a(x_j)$ *to some values, then*

$$d_{\mathrm{dF}}(A, \mathrm{AG}_{i,j}) = \begin{cases} 1 + \varepsilon & \text{iff } C_i[a] = \mathit{True}, \\ 1 & \text{otherwise.} \end{cases}$$

Proof. First of all, observe that, as we only consider one-to-one couplings, the second points of both gadgets must be coupled; the distance between them is $\|(2,0) - (1,0)\| = 1$, thus the discrete Fréchet distance between the trajectories must be at least 1.

Now consider the possible realisations of VG_j. Say, we pick the realisation $(0, 0.5 + \varepsilon) \parallel (2, 0)$, which corresponds to assigning $a(x_j) = \mathsf{True}$. If x_j is a literal of C_i, so $C_i[a] = \mathsf{True}$, then by construction we know that $\mathrm{AG}_{i,j}$ is $(0, -0.5) \parallel (1, 0)$. Since we consider only the one-to-one couplings, we must couple the first points together, yielding the distance $\|(0, 0.5+\varepsilon)-(0, -0.5)\| = 1+\varepsilon > 1$, so the discrete Fréchet distance in this case is $1+\varepsilon$, and indeed we picked the assignment that ensures that C_i is satisfied. If instead $\neg x_j$ is a literal of C_i, so $C_i[a] = \mathsf{False}$, then we know that $\mathrm{AG}_{i,j}$ is $(0, 0.5) \parallel (1, 0)$, and it is easy to see that, as $\|(0, 0.5 + \varepsilon) - (0, 0.5)\| = \varepsilon < 1$ we get the discrete Fréchet distance of 1, and that we picked an assignment that does not ensure that C_i is satisfied.

A symmetric argument can be applied when we consider the realisation $(0, -0.5 - \varepsilon) \parallel (2, 0)$ for VG_j: if $\neg x_j$ is a literal of C_i, then we get the discrete Fréchet distance of $1 + \varepsilon$ and we picked an assignment that surely satisfies C_i.

Finally, consider the case when $\mathrm{AG}_{i,j} = (0, 0) \parallel (1, 0)$. This implies that assigning a value to x_j has no effect on C_i, i.e. a literal involving x_j does not occur in C_i, so neither assignment (and neither realisation of VG_j) would ensure that C_i is satisfied. Also observe that $\|(0, 0.5 + \varepsilon) - (0, 0)\| = \|(0, -0.5 - \varepsilon) - (0, 0)\| = 0.5 + \varepsilon < 1$, so both realisations yield the discrete Fréchet distance of 1.

So, we can conclude that we get the distance $1 + \varepsilon$ if and only if the partial assignment of a value to x_j ensures that C_i is satisfied; otherwise, we get the distance 1. ∎

Now we can construct a *variable clause gadget* and an *assignment clause gadget:*

$$\mathrm{VCG} = (-2, 0) \parallel \Big\Vert_{j \in [m]} \mathrm{VG}_j\,, \qquad \mathrm{ACG}_i = (-1, 0) \parallel \Big\Vert_{j \in [m]} \mathrm{AG}_{i,j}\,.$$

It is crucial now that we show the following:

Lemma 2. *Given a CNF-SAT formula C containing some clause C_i and m variables $x_1, \ldots, x_m$, consider trajectories $\pi_1 \parallel \mathrm{VCG} \parallel \pi_1'$ and $\pi_2 \parallel \mathrm{ACG}_i \parallel \pi_2'$ for arbitrary $\pi_1, \pi_1', \pi_2, \pi_2'$ with $|\pi_1| = k$ and $|\pi_2| = l$. If some optimal coupling between $\pi_1 \parallel \mathrm{VCG} \parallel \pi_1'$ and $\pi_2 \parallel \mathrm{ACG}_i \parallel \pi_2'$ for any realisation of* VCG *has a pair $(k+1, l+1)$, then there is an optimal coupling that has pairs $(k+s, l+s)$ for all $s \in [2m+1]$, i.e. there is an optimal coupling that is one-to-one on the gadgets for any realisation of* VCG*.*

Proof. Observe that both gadgets have exactly $2m+1$ points. Suppose the optimal coupling Opt has a pair $(k+1, l+1)$, so it matches the first points of VCG and ACG_i. If Opt is already one-to-one for all $s \in [2m+1]$, there is nothing to be done. Suppose now that it is one-to-one until some $1 \geq r < 2m+1$, so it has pairs $(k+s, l+s)$ for all $s \in [r]$, but it does not have a pair $(k+(r+1), l+(r+1))$. Then one of the following cases occurs.

- $r = 2q+2$ is even; then we know that the point $(2,0)$ in VG_{q+1} is not coupled with the point $(1,0)$ in $\mathrm{AG}_{i,q+1}$, but the preceding indecisive point is coupled with the assignment point. Then either $(2,0)$ is coupled to an assignment point, with the distance at least 2, or $(1,0)$ is coupled to an indecisive point, yielding the distance of $\sqrt{1+(0.5+\varepsilon)^2} > 1$. If we eliminate that pair and instead couple $(2,0)$ to $(1,0)$, we will still have a valid coupling and obtain the distance of 1 on this pair; thus, the new coupling is not worse that the original one, and so it is also an optimal coupling that is one-to-one for all $s \in [r+1]$.

- $r = 2q+1$ is odd; then we know that the indecisive point in VG_{q+1} is not coupled with the assignment point in $\mathrm{AG}_{i,q+1}$, but the preceding $(2,0)$ and $(1,0)$ (or $(-2,0)$ and $(-1,0)$) are coupled. Then either Opt has a pair of the indecisive point and $(1,0)$, or it has a pair of the assignment point and $(2,0)$. (The cases for $(-1,0)$ and $(-2,0)$ are symmetrical.) In either case, we want to eliminate that pair from the coupling and instead add the pair of the indecisive point and the assignment point, yielding a valid coupling that is one-to-one for all $s \in [r+1]$. To complete the proof for this case, we need to show that such coupling is optimal.

  Consider the first possible coupling. The distance between the indecisive point and $(1,0)$ is $\sqrt{1+(0.5+\varepsilon)^2}$, whereas the distance between

the indecisive and the assignment point is ε, $0.5 + \varepsilon$, or $1 + \varepsilon$. As $\varepsilon < 0.25$, note that

$$\begin{aligned} 0.25 + \varepsilon &> 2\varepsilon \\ 1 + 0.25 + \varepsilon + \varepsilon^2 &> 1 + 2\varepsilon + \varepsilon^2 \\ 1 + (0.5 + \varepsilon)^2 &> (1 + \varepsilon)^2 \\ \sqrt{1 + (0.5 + \varepsilon)^2} &> 1 + \varepsilon \,, \end{aligned}$$

so our change to the optimal coupling will replace a pair with a pair with lower distance, so the new coupling is at least as good as the original one, and thus optimal.

Now consider the second coupling. The distance between the assignment point and $(2, 0)$ is at least 2, and $2 > 1 + \varepsilon > 0.5 + \varepsilon > \varepsilon$, so again our change yields an optimal coupling.

By induction, we conclude that the statement of the lemma holds. ∎

We can now use the two previous results to show the following.

Lemma 3. *Given a CNF-SAT formula C containing some clause C_i and m variables $x_1, \ldots, x_m$, construct trajectories $\pi_1 \parallel \mathrm{VCG} \parallel \pi_1'$ and $\pi_2 \parallel \mathrm{ACG}_i \parallel \pi_2'$ for arbitrary $\pi_1, \pi_1', \pi_2, \pi_2'$ with $|\pi_1| = k$ and $|\pi_2| = l$. If some optimal coupling between $\pi_1 \parallel \mathrm{VCG} \parallel \pi_1'$ and $\pi_2 \parallel \mathrm{ACG}_i \parallel \pi_2'$ for any realisation of VCG has a pair $(k + 1, l + 1)$ and $d_{\mathrm{dF}}(\pi_1, \pi_2) \leq 1$ and $d_{\mathrm{dF}}(\pi_1', \pi_2') \leq 1$, then the discrete Fréchet distance between the trajectories is $1 + \varepsilon$ for realisations of VCG that correspond to satisfying assignments for C_i, and 1 for realisations that do not. In other words, if $A \Subset \mathrm{VCG}$ corresponding to assignment a and we only consider the restricted couplings, then*

$$d_{\mathrm{dF}}(\pi_1 \parallel A \parallel \pi_1', \pi_2 \parallel \mathrm{ACG}_i \parallel \pi_2') = \begin{cases} 1 + \varepsilon & \text{iff } C_i[a] = \mathit{True}, \\ 1 & \text{otherwise.} \end{cases}$$

Proof. First of all, since some optimal coupling between $\pi_1 \parallel \mathrm{VCG} \parallel \pi_1'$ and $\pi_2 \parallel \mathrm{ACG}_i \parallel \pi_2'$ for any realisation of VCG has a pair $(k + 1, l + 1)$, we can use Lemma 2 to find an optimal coupling Opt that is one-to-one on the subtrajectories corresponding to the gadgets. That means that we can, essentially, split the trajectories, if we consider only such restricted couplings:

$$\begin{aligned} &d_{\mathrm{dF}}(\pi_1 \parallel A \parallel \pi_1', \pi_2 \parallel \mathrm{ACG}_i \parallel \pi_2') \\ &= \max(d_{\mathrm{dF}}(\pi_1, \pi_2), d_{\mathrm{dF}}(A, \mathrm{ACG}_i), d_{\mathrm{dF}}(\pi_1', \pi_2')) \\ &= \max(1, d_{\mathrm{dF}}(A, \mathrm{ACG}_i)) \,, \end{aligned}$$

where the last equality follows from the fact that $d_{\mathrm{dF}}(A, \mathrm{ACG}_i) \geq 1$, since the first points are in a coupling and have the distance 1, and from the assumption that $d_{\mathrm{dF}}(\pi_1, \pi_2) \leq 1$ and $d_{\mathrm{dF}}(\pi_1', \pi_2') \leq 1$. Note that here we do not restrict the coupling on π_1, π_2 and π_1', π_2' in any way.

To obtain the end result, we need to consider the distance between A and ACG_i under a one-to-one coupling. Using Lemma 1, it is easy to see that, if we have $a(x_j) = \mathsf{True}$ for some variable x_j and x_j is a literal in C_i, then $C_i[a] = \mathsf{True}$, and $d_{\mathrm{dF}}(A, \mathrm{ACG}_i) = 1 + \varepsilon$; similarly, if $a(x_j) = \mathsf{False}$ for some variable x_j and $\neg x_j$ is a literal in C_i, then $C_i[a] = \mathsf{True}$, and $d_{\mathrm{dF}}(A, \mathrm{ACG}_i) = 1 + \varepsilon$. If there is no such x_j, then $C_i[a] = \mathsf{False}$ and $d_{\mathrm{dF}}(A, \mathrm{ACG}_i) = 1$. We can thus conclude that the lemma holds. ∎

We can now define the trajectories of interest and give the final reduction. Define the *variable trajectory* and the *clause trajectory* as follows:

$$\mathrm{VT} = (0,0) \parallel \mathrm{VCG} \parallel (0,0)\,, \qquad \mathrm{CT} = \Big\Vert_{i \in [n]} \mathrm{ACG}_i\,.$$

So, the trajectory VT contains a gadget per variable in the formula C, starting and ending in $(0,0)$, and the trajectory CT contains a gadget per clause in the formula C, where each such gadget considers the role of each variable in the clause. We show the following result.

Lemma 4. *Given a CNF-SAT formula C with n clauses and m variables, construct the trajectories* VT *and* CT *as defined above and consider a realisation* $(0,0) \parallel A \parallel (0,0)$ *of trajectory* VT*, corresponding to some assignment a. Then, under no restrictions on the couplings except those imposed by the definition,*

$$d_{\mathrm{dF}}((0,0) \parallel A \parallel (0,0), \mathrm{CT}) = \begin{cases} 1+\varepsilon & \text{iff } C[a] = \mathit{True}, \\ 1 & \text{iff } C[a] = \mathit{False}. \end{cases}$$

In other words, the discrete Fréchet distance is $1 + \varepsilon$ *if and only if the realisation corresponds to a satisfying assignment, and is* 1 *otherwise.*

Proof. We can show this by proving that the premises of Lemma 3 are satisfied.

First of all, note that all the points of CT are within distance 1 from $(0,0)$. Furthermore, note that we can always give a coupling with the distance at most $1 + \varepsilon$: couple $(0,0)$ to $(-1,0)$ from ACG_1, then walk along realisation of VCG and ACG_1 in a one-to-one coupling, and then couple the remaining points in CT to $(0,0)$. As all the points of CT are within distance 1 from $(0,0)$ and as this is otherwise the construction of Lemma 3, this coupling

yields the discrete Fréchet distance of at most $1 + \varepsilon$ for any realisation of VT. Therefore, any coupling that has pairs further away than $1 + \varepsilon$ cannot be optimal. Observe that the only point within that distance from $(-2, 0)$ is $(-1, 0)$. Therefore, we only need to consider couplings that couple the first point of realisation of VCG with the first point of some ACG_i as possibly optimal. Thus, for each of the n couplings we get, we can apply Lemma 3. There are two cases to consider.

- There is some gadget ACG_i with the distance 1 to A under the one-to-one coupling. Then we can choose that gadget to align with A and couple all the other points in CT to $(0, 0)$ at the beginning or at the end of VT as suitable. As all the points of CT are within distance 1 from $(0, 0)$, this coupling will yield distance 1; as lower distance is impossible, this coupling is optimal, so then $d_{\mathrm{dF}}((0, 0) \parallel A \parallel (0, 0), \mathrm{CT}) = 1$. Observe that by our construction this situation corresponds to the case when $C_i[a] = \mathsf{False}$, by Lemma 3, and so indeed $C[a] = \mathsf{False}$.

- The distance between any gadget ACG_i and A under the one-to-one coupling is $1 + \varepsilon$. Then, no matter which gadget we choose to align with A, we will get the distance of $1 + \varepsilon$, so in this case $d_{\mathrm{dF}}((0, 0) \parallel A \parallel (0, 0), \mathrm{CT}) = 1 + \varepsilon$. Note that, by our construction, this means that $C_i[a] = \mathsf{True}$ for all $i \in [n]$; therefore, indeed $C[a] = \mathsf{True}$.

As we have covered all the possible cases, we conclude that the lemma holds. ■

We are now prepared to state the main result of this section.

Theorem 5. *The problem Upper Bound Discrete Fréchet for indecisive trajectories is NP-complete.*

Proof. First of all, observe that, if two realisations of length n and m are given as a certificate for a 'Yes'-instance of the problem, then one can verify the solution by computing discrete Fréchet distance between the realisations and checking that it is indeed larger than some threshold t. The computation can be done in time $\Theta(mn)$, using the algorithm proposed by Eiter and Mannila [20]. Therefore, the problem is in NP.

Now suppose we are given an instance of CNF-SAT, i.e. a CNF-SAT formula C with n clauses and m variables. We construct the trajectories VT and CT, as described previously, and get an instance of Upper Bound Discrete Fréchet on trajectories VT and CT and threshold $t = 1$. If the

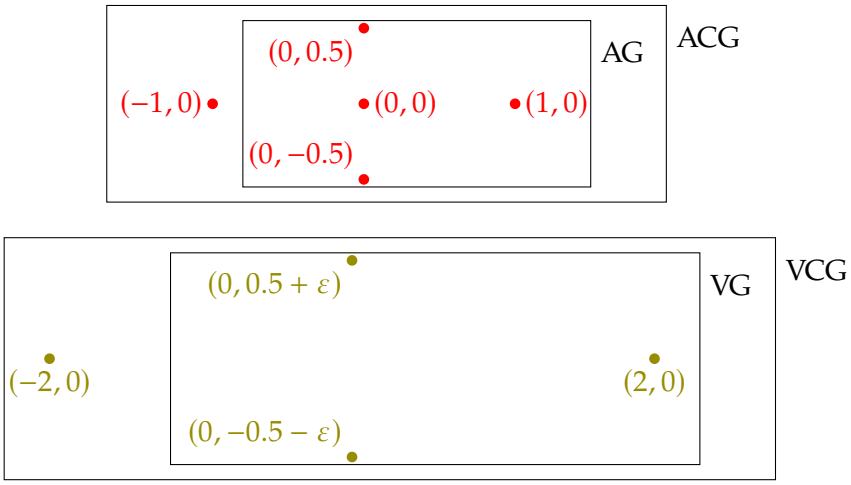


FIGURE 3.1: Illustration of gadgets used in the basic construction.

answer is 'Yes', then we also output 'Yes' as an answer to CNF-SAT; otherwise, we output 'No'.

Using Lemma 4, we can see that, if there is some assignment a such that $C[a] = \mathsf{True}$, then for the corresponding realisation the discrete Fréchet distance is $1 + \varepsilon$; the other way around, if for some realisation we get the distance $1 + \varepsilon$, then by our construction all the clauses are satisfied and $C[a] = \mathsf{True}$; and so $d_{\mathrm{dF}}^{\max}(\mathrm{VT}, \mathrm{CT}) = 1 + \varepsilon$. On the other hand, if there is no such assignment a, then for any assignment a there is some C_i with $C_i[a] = \mathsf{False}$, yielding $C[a] = \mathsf{False}$, and also for any realisation of VT there is some gadget ACG_i that yields the discrete Fréchet distance of 1; and so $d_{\mathrm{dF}}^{\max}(\mathrm{VT}, \mathrm{CT}) = 1$. Therefore, the formula C is satisfiable if an only if $d_{\mathrm{dF}}^{\max}(\mathrm{VT}, \mathrm{CT}) > 1$, and so our answer to the CNF-SAT instance is correct.

Furthermore, observe that the trajectories have $2m+2$ and $2mn+n$ points, respectively, and so the instance of Upper Bound Discrete Fréchet that gives the answer to CNF-SAT can be constructed in polynomial time. Thus, we conclude that Upper Bound Discrete Fréchet for indecisive trajectories is NP-hard; combining it with the first part of the proof shows that it is NP-complete. ■

Construction for Continuous Fréchet Distance on Indecisive Points

We use exactly the same construction as for the discrete Fréchet distance. To do the same proof, we need to present arguments for the continuous case

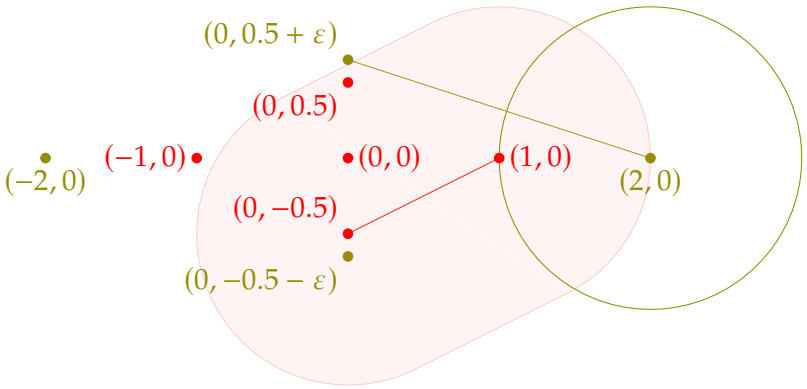


Figure 3.2: Construction for $\varepsilon = 0.15$. Shaded red area is the points within distance 1 from the segment $(0, -0.5) \parallel (1, 0)$. Observe that $(0, 0.5 + \varepsilon)$ is outside that region, and that $(1, 0)$ is the only red point within distance 1 from $(2, 0)$.

that lead up to an alternative to Lemma 4. For the arguments to work, we need to further restrict the range of ε to be $[0.12, 0.25)$.

Consider the construction drawn in Figure 3.2. The key points here are that $(0, 0.5 + \varepsilon)$ is far from any point on the clause trajectory, and that $(2, 0)$ is only close enough to $(1, 0)$. We can present a lemma similar to Lemma 1.

Lemma 6. *Given a clause C_i and a variable x_j that both occur in the CNF-SAT formula C, we only get the Fréchet distance equal to $(1 + \varepsilon) \cdot \frac{2}{\sqrt{5}}$ if the realisation of VG_j we pick corresponds to the assignment of x_j that ensures the clause C_i is satisfied; otherwise, the Fréchet distance is 1. In other words, if we consider $A \Subset \mathrm{VG}_j$ that corresponds to setting $a(x_j)$ to some values, then*

$$d_{\mathrm{F}}(A, \mathrm{AG}_{i,j}) = \begin{cases} (1 + \varepsilon) \cdot \frac{2}{\sqrt{5}} & \text{iff } C_i[a] = \mathit{True}, \\ 1 & \text{otherwise.} \end{cases}$$

Proof. Consider the possible realisations of VG_j. Suppose we pick the realisation $(0, 0.5 + \varepsilon) \parallel (2, 0)$, which corresponds to assigning $a(x_j) = \mathsf{True}$. If x_j is a literal in C_i, so $C_i = \mathsf{True}$, then by construction we know that $\mathrm{AG}_{i,j}$ is $(0, -0.5) \parallel (1, 0)$. As noted in Figure 3.2, the distance between $(0, 0.5 + \varepsilon)$ and any point on $(0, -0.5) \parallel (1, 0)$ is larger than 1. To be more specific, the distance between the point (x, y) and the line defined by $(x_1, y_1) \parallel (x_2, y_2)$ can be determined using a standard formula as

$$d = \frac{|x(y_2 - y_1) - y(x_2 - x_1) + x_2 y_1 - x_1 y_2|}{\sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}}.$$

In our case, we get

$$d = \frac{|0 - (0.5 + \varepsilon) \cdot (1 - 0) - 1 \cdot 0.5 - 0|}{\sqrt{(1-0)^2 + (0 + 0.5)^2}} = \frac{2 \cdot (1 + \varepsilon)}{\sqrt{5}}.$$

As the point $(0, 0.5 + \varepsilon)$ must be coupled to some point on $\mathrm{AG}_{i,j}$, the Fréchet distance we get in this case cannot be smaller than d. Furthermore, it is easy to see that the point $(0, 0.5 + \varepsilon)$ is the furthest point from $\mathrm{AG}_{i,j}$, thus we get that Fréchet distance is exactly d.

On the other hand, if $\neg x_j$ is a literal in C_i, then by construction we know that $\mathrm{AG}_{i,j}$ is $(0, 0.5) \parallel (1, 0)$. As noted in Figure 3.2, the distance between $(2, 0)$ and any point on $(0, 0.5) \parallel (1, 0)$ is at least 1, with the smallest distance achieved at $(1, 0)$. It is clear that this is the furthest pair of points on the two gadgets in this case; thus, we get the Fréchet distance of 1.

A symmetric argument can be applied when we consider the realisation $(0, -0.5 - \varepsilon) \parallel (2, 0)$ for VG_j: if $\neg x_j$ is a literal in C_i, then we get the Fréchet distance of d and we picked an assignment that satisfies C_i; and in the other case, we get that $C_i[a]$ is not necessarily satisfied and the Fréchet distance of 1.

Finally, consider the case when $\mathrm{AG}_{i,j} = (0, 0) \parallel (1, 0)$. Again, this implies that assigning a value to x_j has no effect on C_i, so neither assignment (and neither realisation of VG_j) would ensure that C_i is satisfied. Also observe that both realisations give rise to trajectories that are entirely within distance 1 of $(0, 0) \parallel (1, 0)$, yielding the Fréchet distance of 1. ∎

We can now naturally get a lemma similar to Lemma 3.

Lemma 7. *Given a CNF-SAT formula C containing some clause C_i and m variables $x_1, \ldots, x_m$, construct trajectories $\pi_1 \parallel \mathrm{VCG} \parallel \pi_1'$ and $\pi_2 \parallel \mathrm{ACG}_i \parallel \pi_2'$ for arbitrary $\pi_1, \pi_1', \pi_2, \pi_2'$ with $|\pi_1| = k$ and $|\pi_2| = l$. If some optimal coupling ϕ_1, ϕ_2 between $\pi_1 \parallel \mathrm{VCG} \parallel \pi_1'$ and $\pi_2 \parallel \mathrm{ACG}_i \parallel \pi_2'$ for any realisation of VCG has some value t such that $\phi_1(t) = k + 1$ and $\phi_2(t) = l + 1$ and $d_{\mathrm{F}}(\pi_1, \pi_2) \leq 1$ and $d_{\mathrm{F}}(\pi_1', \pi_2') \leq 1$, then the Fréchet distance between the trajectories is $(1 + \varepsilon) \cdot \frac{2}{\sqrt{5}}$ for realisations of VCG that correspond to satisfying assignments for C_i, and 1 for realisations that do not. In other words, if $A \Subset \mathrm{VCG}$ corresponds to assignment a and we only consider the restricted couplings, then*

$$d_{\mathrm{F}}(\pi_1 \parallel A \parallel \pi_1', \pi_2 \parallel \mathrm{ACG}_i \parallel \pi_2') = \begin{cases} (1 + \varepsilon) \cdot \frac{2}{\sqrt{5}} & \text{iff } C_i[a] = \textsf{True}, \\ 1 & \text{otherwise.} \end{cases}$$

Proof. First of all, observe that, as we traverse VCG, we need to couple $(2, 0)$ to $(1, 0)$ to obtain an optimal coupling. Therefore, essentially, the

traversal can be split into m parts, each of which corresponds to traversing VG_j and $\mathrm{AG}_{i,j}$ at the same time for all $j \in [m]$. We can thus use Lemma 6 to note that, if some variable x_j is assigned a value that makes clause C_i satisfied, then the Fréchet distance becomes $(1+\varepsilon)\cdot\frac{2}{\sqrt{5}}$; if that is not the case for any variables, then we can traverse the entire trajectory, as well as π_1 and π_1' by linearly interpolating our position between the vertices of the trajectories and otherwise using the coupling of the discrete case, while staying within distance 1 of the other trajectory, yielding the Fréchet distance 1. The distance also cannot be smaller than 1 due to coupling of $(2,0)$ and $(1,0)$. ■

While this proof is a bit less formal than that of Lemma 3, its validity should be sufficiently clear from geometric considerations described earlier in this section.

Now we can provide a lemma that mirrors Lemma 4.

Lemma 8. *Given a CNF-SAT formula C with n clauses and m variables, construct the trajectories* VT *and* CT *as defined above and consider a realisation $(0,0) \parallel A \parallel (0,0)$ of trajectory* VT*, corresponding to some assignment a. Then*

$$d_{\mathrm{F}}((0,0) \parallel A \parallel (0,0), \mathrm{CT}) = \begin{cases} (1+\varepsilon)\cdot\frac{2}{\sqrt{5}} & \text{iff } C[a] = \textit{True}, \\ 1 & \text{iff } C[a] = \textit{False}. \end{cases}$$

In other words, the Fréchet distance is $(1+\varepsilon)\cdot\frac{2}{\sqrt{5}}$ if and only if the realisation A corresponds to a satisfying assignment, and is 1 otherwise.

Proof. First of all, observe that any point of CT is within distance 1 of $(0,0)$; furthermore, when starting to traverse A, we must couple $(-2,0)$ to $(-1,0)$ in an optimal coupling. Thus, the premise of Lemma 7 is satisfied, and, using reasoning similar to that of Lemma 4, we observe that an optimal coupling chooses one of the clauses to traverse in parallel with the variable trajectory, and so if there is a clause that is not satisfied, then we get the Fréchet distance of 1, and if all of them are satisfied, then all of then yield the Fréchet distance of $(1+\varepsilon)\cdot\frac{2}{\sqrt{5}}$. Thus, we conclude that the lemma holds. ■

Finally, we can show the main result.

Theorem 9. *The problem Upper Bound Continuous Fréchet for indecisive trajectories is NP-complete.*

Proof. First of all, observe that, if two realisations of length n and m are given as a certificate for a 'Yes'-instance of the problem, then one can verify

the solution by checking that the Fréchet distance between the realisations is indeed larger than some threshold t. The computation can be done in time $\Theta(mn)$, using the algorithm proposed by Alt and Godau [6, 24]. Therefore, the problem is in NP.

Now suppose we are given an instance of CNF-SAT, i.e. a CNF-SAT formula C with n clauses and m variables. We construct the trajectories VT and CT, as described previously, and get an instance of Upper Bound Continuous Fréchet on trajectories VT and CT and threshold $t = 1$. If the answer is 'Yes', then we also output 'Yes' as an answer to CNF-SAT; otherwise, we output 'No'.

Using Lemma 8, we can see that, if there is some assignment a such that $C[a] = \mathsf{True}$, then for the corresponding realisation the Fréchet distance is $(1 + \varepsilon) \cdot \frac{2}{\sqrt{5}}$; the other way around, if for some realisation we get the distance $(1 + \varepsilon) \cdot \frac{2}{\sqrt{5}}$, then by our construction all the clauses are satisfied and $C[a] = \mathsf{True}$; and so $d_{\mathrm{F}}^{\max}(\mathrm{VT}, \mathrm{CT}) = (1 + \varepsilon) \cdot \frac{2}{\sqrt{5}}$. On the other hand, if there is no such assignment a, then for any assignment a there is some C_i with $C_i[a] = \mathsf{False}$, yielding $C[a] = \mathsf{False}$, and also for any realisation of VT there is some gadget ACG_i that yields the Fréchet distance of 1; and so $d_{\mathrm{F}}^{\max}(\mathrm{VT}, \mathrm{CT}) = 1$. Therefore, the formula C is satisfiable if an only if $d_{\mathrm{F}}^{\max}(\mathrm{VT}, \mathrm{CT}) > 1$, and so our answer to the CNF-SAT instance is correct.

Furthermore, as before, the instance of Upper Bound Discrete Fréchet that gives the answer to CNF-SAT can be constructed in polynomial time. Thus, we conclude that Upper Bound Continuous Fréchet for indecisive trajectories is NP-hard; combining it with the first part of the proof shows that it is NP-complete. ■

Construction for Discrete Fréchet Distance on Imprecise Points

Here we consider imprecise points modelled as disks and as line segments; the results and their proofs turn out to be very similar.

Disks We use a construction very similar to that of the indecisive points, except now we change the gadget containing a non-degenerate indecisive point so that it contains a non-degenerate imprecise point, for all $j \in [m]$:

$$\mathrm{VG}_j = D((0, 0), 0.5 + \varepsilon) \parallel (2, 0)\,.$$

Essentially, the two original indecisive points are now located on the diameter of the disk.

We can reuse the proof leading up to Theorem 5, if we can show the following:

Lemma 10. *Suppose* $d_{\mathrm{dF}}^{\max}(\mathrm{VT}, \mathrm{CT}) = \nu$*. If one considers all realisations* A *of* VT *that yield* $d_{\mathrm{dF}}(A, \mathrm{CT}) = \nu$*, then among them there will always be a realisation that only places the imprecise point realisations at either* $(0, 0.5 + \varepsilon)$ *or* $(0, -0.5 - \varepsilon)$*.*

Proof. First of all, note that the points $(2, 0)$ and $(1, 0)$ are still in the trajectories in the same quality as before, so they must be coupled, and hence the lowest discrete Fréchet distance achievable with any realisation is 1.

Now consider a realisation of an imprecise point. Suppose that all the clause assignment points for that imprecise point are placed at $(0, -0.5)$. Then geometrically it is obvious that the distance is maximised by placing the realisation at $(0, 0.5 + \varepsilon)$; if there is a realisation that achieves the best possible value ν without doing this, then we can move this point and still get ν.

Suppose that some clause assignment points are at $(0, -0.5)$ and some at $(0, 0.5)$. As the realisation comes from the disk of radius $0.5 + \varepsilon$, there is no realisation that is further than 1 away from both assignment points; therefore, to maximise the distance we have to choose one of the two locations, and then the previous case applies.

So, it is clear that, from an arbitrary optimal realisation, moving to the (correct) indecisive point realisation will still yield an optimal realisation for the maximum discrete Fréchet distance, thus the statement of the lemma holds. ■

Line segments Again, we use a very similar construction, except now we change the gadget to be, for all $j \in [m]$:

$$\mathrm{VG}_j = S((0, -0.5 - \varepsilon), (0, 0.5 + \varepsilon)) \parallel (2, 0)\,.$$

Again, the two original indecisive points are now located on the ends of the segment; moreover, the segment is a strict subset of the disk.

We can state a similar lemma.

Lemma 11. *Suppose* $d_{\mathrm{dF}}^{\max}(\mathrm{VT}, \mathrm{CT}) = \nu$*. If one considers all realisations* A *of* VT *that yield* $d_{\mathrm{dF}}(A, \mathrm{CT}) = \nu$*, then among them there will always be a realisation that only places the imprecise point realisations at either* $(0, 0.5 + \varepsilon)$ *or* $(0, -0.5 - \varepsilon)$*.*

Proof. Since the line segments include these points and are subsets of the disks, the statement of Lemma 10 immediately yields this result. ■

So, now we can state the following for both models:

Theorem 12. *The problem* Upper Bound Discrete Fréchet *for imprecise trajectories modelled as line segments or as disks is* NP*-complete.*

Proof. As shown in the proof of Theorem 5, the problem is in NP for any uncertain trajectories.

Furthermore, as we have shown in Lemma 10 and Lemma 11, for the same CNF-SAT formula the upper bound discrete Fréchet distance on indecisive and imprecise points is equal for our construction. So, trivially, Upper Bound Discrete Fréchet is NP-hard for imprecise trajectories. Therefore, it is NP-complete. ■

Construction for Continuous Fréchet Distance on Imprecise Points

We use exactly the same construction as in the previous section. The argument here follows the previous ones very closely, so we can immediately state the following theorem.

Theorem 13. *The problem Upper Bound Continuous Fréchet for imprecise trajectories modelled as line segments or as disks is NP-complete.*

Proof. Note that we can apply exactly the same argument as the one in Lemma 10 and Lemma 11 to reduce this problem to the one on indecisive points. Then, we can apply the same argument as in the proof of Theorem 12 to conclude that the problem is NP-hard.

We have shown in Theorem 9 that the problem is in NP for all uncertain trajectories; thus, we conclude that it is NP-complete. ■

3.2 #P-Hardness of Expected Fréchet Distance

Here we present proofs that computing $d_{\mathrm{dF}}^{\mathbb{E}(\mathbb{U})}$ and $d_{\mathrm{F}}^{\mathbb{E}(\mathbb{U})}$ on either indecisive or imprecise trajectories is #P-hard. To that aim, we show reductions from #CNF-SAT. This is a counting version of the CNF-SAT problem formulated previously, and it can be stated as follows:

> #CNF-SAT
> Input: A CNF-SAT formula C.
> Output: The number of distinct variable assignments a such that $C[a] = \mathsf{True}$.

The problems in the domain of interest that we consider can be formulated as follows:

> Expected Discrete Fréchet
> Input: Two uncertain trajectories $\mathcal{H}$ and $\mathcal{V}$.
> Output: $d_{\mathrm{dF}}^{\mathbb{E}(\mathbb{U})}(\mathcal{H}, \mathcal{V})$.

We can define a similar problem for the continuous Fréchet distance:

> Expected Continuous Fréchet
> Input: Two uncertain trajectories $\mathcal{H}$ and $\mathcal{V}$.
> Output: $d_{\mathrm{F}}^{\mathbb{E}(\mathbb{U})}(\mathcal{H}, \mathcal{V})$.

In the remainder of this section, we will show that it is possible, given an instance of #CNF-SAT problem, to construct an instance of Expected Discrete Fréchet and Expected Continuous Fréchet and in both cases to use the solutions to those problems to construct a solution to #CNF-SAT, and such construction can be done in polynomial time.

Construction on Indecisive Points

For the settings involving indecisive points we can use exactly the same construction that was described in the proofs of NP-hardness of the upper bound Fréchet distance.

An attentive reader might have noticed that in the proofs we established the relation between the particular outcomes for the discrete Fréchet distance on specific realisations and the question whether these realisations correspond to satisfying assignments. So, the number of satisfying assignments is equal to the number of realisations yielding the discrete Fréchet distance of $1 + \varepsilon$; such a reduction is called a *parsimonious reduction.* In principle, if we had a trivial counting version of the problem, we could immediately apply a well-known result that states that, if a problem is NP-complete and its NP-hardness has been established through a chain of parsimonious reductions, then the associated counting problem is #P-complete, presented e.g. by Goldreich [25, Theorem 6.18]. The problem of finding the expected value is not a counting problem, but it intuitively is similar to one. Indeed, the proof turns out to be quite simple and follows in a straightforward manner from the fact that the reduction is parsimonious and from the definition of expected value on discrete distributions.

Theorem 14. *The problem Expected Discrete Fréchet for indecisive trajectories is #P-hard.*

Proof. Suppose we are given an instance of the #CNF-SAT problem, i.e. a CNF-SAT formula C with n clauses and m variables. Denote the (unknown)

number of satisfying assignments of C by N. We can use the same reduction as the one suggested in the proof of Theorem 5 to construct indecisive trajectories VT and CT. We then get an instance of Expected Discrete Fréchet on indecisive trajectories. Assuming we solve it and get $d_{\mathrm{dF}}^{\mathbb{E}(\mathbb{U})}(\mathrm{VT}, \mathrm{CT}) = \mu$, we can now compute N as follows:

$$N = (\mu - 1) \cdot \frac{2^m}{\varepsilon}.$$

N is then the output for the instance of #CNF-SAT that we were given. As shown in the proof of Theorem 5, construction of the trajectories can be done in polynomial time; clearly, so can be the computation of N, hence the reduction takes polynomial time.

We now still need to show that the result we obtain is correct. Note that the construction is parsimonious, in the sense described earlier. For a distinct assignment, there is exactly one realisation of the trajectory VT. Furthermore, as we choose the realisation of each indecisive point uniformly and independently, all the realisations have equal probability of 2^{-m}. Furthermore, there are N satisfying assignments; and each of the corresponding realisations yields the discrete Fréchet distance of $1 + \varepsilon$. In the remaining $2^m - N$ cases, the distance is 1. Using the definition of expected value, we can derive

$$\begin{aligned}
\mu &= d_{\mathrm{dF}}^{\mathbb{E}(\mathbb{U})}(\mathrm{VT}, \mathrm{CT}) \\
&= N \cdot 2^{-m} \cdot (1 + \varepsilon) + (2^m - N) \cdot 2^{-m} \cdot 1 \\
&= 2^{-m} \cdot (N + N \cdot \varepsilon + 2^m - N) \\
&= 1 + \frac{N \cdot \varepsilon}{2^m}.
\end{aligned}$$

Then it is easy to see that indeed

$$N = (\mu - 1) \cdot \frac{2^m}{\varepsilon}.$$

So, we get the correct number of satisfying assignments, if we know the expected value under uniform distribution. Therefore, Expected Discrete Fréchet for indecisive trajectories is #P-hard. ■

Observe that, for the continuous Fréchet distance, the reduction is very similar: it is again parsimonious, and the values of the continuous Fréchet distance are only slightly different. Therefore, we can easily state the following.

Theorem 15. *The problem Expected Continuous Fréchet for indecisive trajectories is #P-hard.*

Proof. We can use almost the same reduction as in Theorem 14, so, given an instance of #CNF-SAT (CNF-SAT formula C with n clauses and m variables), we construct the two trajectories, solve Expected Continuous Fréchet to obtain the value of μ, and compute

$$N = 2^m \cdot (\mu - 1) \cdot \frac{\sqrt{5}}{2(1+\varepsilon) - \sqrt{5}}$$

as the output for #CNF-SAT.

To show that the output is correct, note that

$$\begin{aligned}\mu &= 2^{-m} \cdot N \cdot \frac{2}{\sqrt{5}} \cdot (1+\varepsilon) + 2^{-m} \cdot (2^m - N) \cdot 1 \\ &= 1 + 2^{-m} \cdot N \cdot \left(\frac{2}{\sqrt{5}}(1+\varepsilon) - 1 \right),\end{aligned}$$

so we can express N as

$$N = 2^m \cdot (\mu - 1) \cdot \frac{\sqrt{5}}{2(1+\varepsilon) - \sqrt{5}}.$$

Again, the reduction is correct and can be done in polynomial time, so Expected Continuous Fréchet for indecisive trajectories is #P-hard. ∎

Construction on Imprecise Points

Here we only treat the imprecise points modelled as line segments; in principle, we believe that for disks a similar result should hold, but the specifics of the reduction do not allow for a clean computation of N given the expected value in the current construction or its obvious modifications.

We use the same construction as we did when proving Theorem 12, with a minor addition: we define a new gadget, which will force the resulting distance to be predictable. For every $j \in [m]$, we can define

$$\begin{aligned}\mathrm{FG}_j = (-1,0) \parallel \Big\Vert_{k \in [j-1]} ((0,0) \parallel (1,0)) \\ \parallel (0,0.5) \parallel (0,-0.5) \parallel (1,0) \\ \parallel \Big\Vert_{k \in [m] \setminus [j]} ((0,0) \parallel (1,0)).\end{aligned}$$

So, we define a clause gadget that ignores all the variables except for x_j and then features both 'true' and 'false' for x_j. We then define the clause trajectory as

$$\mathrm{CT} = \Big\Vert_{i\in[n]} \mathrm{ACG}_i \,\|\, \Big\Vert_{j\in[m]} \mathrm{FG}_j\,.$$

The idea here is quite simple: we can now choose to align one of FG clauses with the variable trajectory. Note that, as before, due to the synchronisation points we can never get the Fréchet distance below 1. If one of the realisations x_j of the segments falls into the interval $[(0,-0.5),(0,0.5)]$, then it will be not further away than 1 from both the corresponding points on FG_j; all the other points, being in the middle at $(0,0)$, are guaranteed to be at most $0.5+\varepsilon<1$ away from their coupled point; so, the one-to-one coupling will yield the discrete Fréchet distance of 1, thus the optimal discrete Fréchet distance in this case is 1.

Therefore, we only need to consider the situations when all the realisations happen to fall in either the interval $((0,0.5),(0,0.5+\varepsilon)]$ or $[(0,-0.5-\varepsilon),(0,-0.5))$. We will treat the first interval as `True` and the second interval as `False`. Denote the number of satisfying assignments by N. So, overall, to find the expression for the expected discrete Fréchet distance, we need to consider three cases:

- At least one realisation of m variables falls within the y-interval $[-0.5,0.5]$. Note that the realisation on each segment is uniform and independent of other segments. So, the probability of this case happening is

$$\begin{aligned}
&\Pr[\text{at least one realisation from } [-0.5,0.5]]\\
&= 1-\Pr[\text{all realisations outside } [-0.5,0.5]]\\
&= 1-\prod_{j\in[m]}\Pr[\text{realisation of } x_j \text{ outside } [-0.5,0.5]]\\
&= 1-\prod_{j\in[m]}\frac{2\varepsilon}{1+2\varepsilon}\\
&= 1-\left(\frac{2\varepsilon}{1+2\varepsilon}\right)^m.
\end{aligned}$$

  Note that in each such case we get the discrete Fréchet distance of 1, as discussed before.

- All realisations fall outside the y-interval $[-0.5,0.5]$, and they correspond to a non-satisfying assignment. Each specific non-satisfying

assignment corresponds to picking values on the specific interval, either $((0, 0.5), (0, 0.5 + \varepsilon)]$ or $[(0, -0.5 - \varepsilon), (0, -0.5))$, depending on whether we need True or False. So, the probability of getting a particular assignment is

$$\Pr[\text{specific assignment}] = \prod_{j \in [m]} \frac{\varepsilon}{1 + 2\varepsilon} = \left(\frac{\varepsilon}{1 + 2\varepsilon}\right)^m .$$

Overall there are $2^m - N$ such assignments, and each of them contributes the value of discrete Fréchet distance of 1, just as before—we can choose a non-satisfied clause and align the corresponding subtrajectories, yielding the discrete Fréchet distance of 1.

- All realisations fall outside the y-interval $[-0.5, 0.5]$, and they correspond to a satisfying assignment. Again, the probability of getting a particular assignment is $\left(\frac{\varepsilon}{1+2\varepsilon}\right)^m$, and there are N such assignments. Now they contribute values distinct from 1; still, the optimum is contributed by one of the new clauses, and then it will be defined by the realisation closest to $(0, 0)$. This is shown in the following lemma.

  Lemma 16. *Consider some realisation $A \Subset \mathrm{VT}$ where each value can be interpreted either as* True *or* False *and the corresponding assignment satisfies the formula. Pick j such that the subtrajectory of A realising VG_j contains the point closest to $(0, 0)$, at location $(0, 0.5 + \varepsilon')$ or $(0, -0.5 - \varepsilon')$ for some $\varepsilon' > 0$. Then the optimal coupling establishes a matching between A and FG_j, and the discrete Fréchet distance is $d_{\mathrm{dF}}(A, \mathrm{CT}) = 1 + \varepsilon'$.*

  Proof. First of all, note that we still have to couple the synchronisation points and we cannot have discrete Fréchet distance below 1. So, we only need to consider the couplings of A with the gadgets of CT.

  Note that if we align FG_j with A, we get discrete Fréchet distance of $1 + \varepsilon'$.

  Recall that we consider only satisfying assignments, so, if we consider an arbitrary subtrajectory ACG_i, then there is some variable x_j that satisfies the corresponding clause, and so the realisation of that variable is $1 + \varepsilon''$ away from the corresponding assignment point. Therefore, such a coupling will yield the discrete Fréchet distance of $1 + \varepsilon'' \geq 1 + \varepsilon'$.

  Finally, it is easy to see that choosing some FG_k with $k \neq j$ will also yield some distance $1 + \varepsilon'' \geq 1 + \varepsilon'$.

  So, the statement of the lemma holds. ∎

So, here we need to find $\mathbb{E}[\min_{j\in[m]}(1+\varepsilon'_j)]$ with ε'_j sampled uniformly from $(0, \varepsilon]$; we can rephrase this to $1+\varepsilon\cdot\mathbb{E}[\min_{j\in[m]} u_j]$ with u_j sampled uniformly from $(0, 1]$. It is a standard result that the minimum now is geometrically distributed, so we get

$$\mathbb{E}[\min_{j\in[m]} u_j] = \frac{1}{1+m}\,,$$

and hence the expected contribution is

$$1 + \frac{\varepsilon}{1+m}\,.$$

We can bring the three cases together to find

$$\begin{aligned} d_{\mathrm{dF}}^{\mathbb{E}}(\mathrm{VT},\mathrm{CT}) &= 1\cdot\left(1-\left(\frac{2\varepsilon}{1+2\varepsilon}\right)^m\right) \\ &\quad + 1\cdot(2^m-N)\cdot\left(\frac{\varepsilon}{1+2\varepsilon}\right)^m \\ &\quad + \left(1+\frac{\varepsilon}{1+m}\right)\cdot N\cdot\left(\frac{\varepsilon}{1+2\varepsilon}\right)^m \\ &= 1-\left(\frac{2\varepsilon}{1+2\varepsilon}\right)^m + 2^m\cdot\left(\frac{\varepsilon}{1+2\varepsilon}\right)^m \\ &\quad + \frac{N\varepsilon}{1+m}\cdot\left(\frac{\varepsilon}{1+2\varepsilon}\right)^m \\ &= 1 + N\cdot\frac{\varepsilon^{m+1}}{(1+m)\cdot(1+2\varepsilon)^m}\,. \end{aligned}$$

So, if we were to compute $d_{\mathrm{dF}}^{\mathbb{E}}(\mathrm{VT},\mathrm{CT}) = \mu$, then clearly the number of satisfying assignments is

$$N = (\mu-1)\cdot\frac{(1+m)\cdot(1+2\varepsilon)^m}{\varepsilon^{m+1}}\,.$$

This is easy to compute in polynomial time, and our construction can still be done in polynomial time; hence, we can state the following.

Theorem 17. *The problem Expected Discrete Fréchet for imprecise trajectories modelled as line segments is #P-hard.*

Proof. Follows directly from the previous considerations. ∎

4

Algorithms with Time Bands

As we have discussed before, *time bands* are an interesting concept that allows one to enforce temporal synchronisation of trajectories when considering their similarity. As we have also seen in the previous chapter, general computation of upper bound and expected discrete and continuous Fréchet distance is infeasible even under rather simple uncertainty models. However, as it turns out, computing these values when enforcing a time band is indeed feasible, as this chapter shall explain. In all of this chapter we only discuss *indecisive* points.

4.1 Upper Bound Discrete Fréchet Distance: Precise and Indecisive

First of all, let us discuss a rather simple setting. Suppose we are given a trajectory $\mathcal{H} = \langle P_1, \ldots, P_n \rangle$ of n indecisive points, each of them having ℓ options, so for all $i \in [n]$ we have $P_i = \{p_i^1, \ldots, p_i^\ell\}$. Consider also the second trajectory $V = \langle q_1, \ldots, q_n \rangle$ of n precise points. We would like to answer the following decision problem: *'If we restrict the couplings to Sakoe–Chiba band of width w, is it true that $d_{\mathrm{dF}}^{\max}(\mathcal{H}, V) \leq \varepsilon$ for some given threshold $\varepsilon > 0$?'*

So, we want to solve the decision problem for the upper bound discrete Fréchet distance on a pair of precise and indecisive trajectories. In general, as we have seen, the problem is NP-complete. However, because we are limited to the width w, we can actually solve the problem.

Consider the dynamic programming table that we use to compute discrete Fréchet distance on precise trajectories, as introduced in Chapter 2. Recall that we discussed the decision version, where we simply need to

propagate the values of True and False. In principle, the current setting is not that different.

Suppose we position $\mathcal{H}$ to go horizontally along the table, and V to go vertically. Consider an arbitrary column in the table and suppose that we fix the realisation of $\mathcal{H}$ up to the previous column. Then we can simply consider the new column ℓ times, each time picking a different realisation for the new point on $\mathcal{H}$, and computing the resulting reachability. As we do this for the entire column at once, we can ensure consistency of our choice of realisation. This procedure will, essentially, give us a set of binary reachability vectors for the new column, each vector corresponding to a realisation. The *reachability vector* thus is a boolean vector that, for the cell (i, j) of the table, states if for a particular realisation A of $\mathcal{H}[1 : i]$ the discrete Fréchet distance between A and $V[1 : j]$ is below some threshold ε.

However, an important observation is that we do not need to distinguish between the realisations that give the same reachability vector: once we start filling out the next column, all we care about is the existence of some realisation leading to that particular reachability vector. So, we can indeed just keep a *set* of binary vectors corresponding to reachability in the column.

Note that this procedure was suggested for a specific realisation up to the current point, and so for a fixed reachability vector in the previous column. However, we can also simply repeat this process for each previous reachability vector, only keeping the unique results. As all the realisation choices happen along $\mathcal{H}$, by treating the table column-by-column we make sure that we do not have issues with inconsistent choices. Therefore, repeating this procedure n times, we will fill out the last column of the table. At that point, if any vector has False in the top right cell, then there is some realisation $A \Subset_{\mathbb{P}} \mathcal{H}$ such that $d_{\mathrm{dF}}(A, V) > \varepsilon$, and hence $d_{\mathrm{dF}}^{\max}(\mathcal{H}, V) > \varepsilon$.

Algorithm Now that we have provided some intuition, let us formally define the algorithm. First of all, note that width of time band w means that we consider $2w + 1$ couplings for each point, symmetrically w steps away from the diagonal. Similarly to the regular discrete Fréchet distance, we use two tables, distance matrix D and reachability matrix R. We interpret $D_{i,k} = \langle D_{i,k,i-w}, \ldots, D_{i,k,i+w} \rangle$. The pseudocode is shown in Algorithm 1.

Some details might be confusing at first. First of all, we initialise the distance matrix D in a straightforward way. Whenever convenient, we will treat each D_i and R_i as a set of boolean vectors. The reachability of the first column is special: we can only start from the bottom left cell, i.e. from $i = 1$, which is why we use the vector $R0$ for reachability. However, we can then go

Algorithm 1 Finding time-banded upper bound discrete Fréchet distance on an indecisive and a precise trajectory.

1: **function** TBDFDIndPr($\mathcal{H}, V, w, \varepsilon$)
▷ Input constraint: $|\mathcal{H}| = |V| = n$ and $0 \leq w < n$
2: Initialise matrix D of size $n \times \ell \times (2w + 1)$
3: **for all** $i \in [n]$ **do**
4: **for all** $k \in [\ell]$ **do**
5: **for all** $j \in \{\max(1, i - w), \ldots, \min(n, i + w)\}$ **do**
6: $D_{i,k,j} \leftarrow [d(p_i^k, q_j) \leq \varepsilon?]$
7: Initialise matrix R of size $n \times 2^{2w+1} \times 2w + 1$
8: $R0 \leftarrow \langle r_1 = \mathsf{True}, r_2 = \mathsf{False}, r_3 = \mathsf{False}, \ldots, r_{w+1} = \mathsf{False} \rangle$
9: **for all** $k \in [\ell]$ **do**
10: $R_{1,k} \leftarrow$ Propagate($R0, D_{1,k}, 1, w, n$)
11: **for all** $i \in [n] \setminus \{1\}$ **do**
12: **for all** $A \in R_{i-1}$ **do** ▷ For each reachability vector
13: $B \leftarrow A \vee (A << 1)$
14: **for all** $k \in [\ell]$ **do**
15: $C \leftarrow$ Propagate($B, D_{i,k}, i, w, n$)
16: Add C to set R_i
17: $r \leftarrow \mathsf{True}$
18: **for all** $A \in R_n$ **do**
19: $r \leftarrow r \wedge A_n$
20: **return** r
21: **function** Propagate(A, B, i, w, n)
▷ Propagate the reachability upwards in a column
22: $C \leftarrow A \wedge B$
23: $r \leftarrow \mathsf{False}$
24: **for all** $j \in \{\max(1, i - w), \ldots, \min(n, i + w)$ **do**
25: **if** $B_j \wedge C_j$ **then**
26: $r \leftarrow \mathsf{True}$
27: **else if** $B_j \wedge \neg C_j \wedge r$ **then**
28: $C_j \leftarrow \mathsf{True}$
29: **else if** $\neg B_j$ **then**
30: $r \leftarrow \mathsf{False}$
31: **return** C

upwards, if allowed by the distance matrix, that is, add more precise points and couple them to the current point—this is done by function Propagate.

Then we fill out R column-by-column. We take the reachability of the previous column and note that any cell can be reached either with the horizontal step or with the diagonal step, yielding the disjunction of the vectors A shifted by 1 and A, respectively. Then we need to consider various extensions of the trajectory $\mathcal{H}$ with one of the ℓ realisations of the current point: the distance matrix should allow the specific coupling, and we should make sure to allow for upwards movement within the vector, both done by a call to Propagate. Now we just remember the newly computed vector; note that we treat R_i as a set, so we only add distinct vectors.

Finally, we check if there is a realisation that may yield `False` in the last cell; if so, then this will be the realisation chosen by the upper bound operation, and so the answer is `False`. Otherwise, the answer is `True`.

Correctness We use the following loop invariant to show correctness of the algorithm.

Lemma 18. *Consider column i. Every reachability vector of this column corresponds to some realisation of $\mathcal{H}[1:i]$ and the discrete Fréchet distance between that realisation and $V[1:\min(n, i+w)]$; and the other way around, every realisation corresponds to some reachability vector.*

Proof. The statement is trivial for the first column: we consider all ℓ possible realisations of $\mathcal{H}(1)$ and compute reachability of cells $(1,1)$ to $(1,1+w)$ in a straightforward way, so clearly, for every realisation there is a reachability vector, and every reachability vector corresponds to a realisation.

Now suppose the statement holds for column i. As follows from the recurrence establishing discrete Fréchet distance in Section 2.1, the reachability of column $i+1$ only depends on the distance matrix for column $i+1$ and the reachability of column i. We consider every possible extension of $\mathcal{H}[1:i]$ to $\mathcal{H}[1:i+1]$, as for every reachability vector of column i we consider all ℓ options for the distance matrix for column $i+1$. Thus, we only consider valid realisations for column $i+1$, and we consider all of them from the point of view of reachability.

So, the statement of the lemma holds. ∎

Since the lemma holds, the reachability of the top right cell corresponds to the solution to the decision problem for each possible realisation of $\mathcal{H}$. So, if at least one reachability vector for column n has `False` as the last value, then there is some realisation A of $\mathcal{H}$ that ensures that $d_{\mathrm{dF}}(A,V) > \varepsilon$, and

so $d_{\mathrm{dF}}^{\max}(\mathcal{H}, V) > \varepsilon$, correctly yielding the answer `False`; otherwise, we get `True`, as we should.

Running time First of all, populating the distance matrix takes time $\Theta(\ell n w)$. A call to Propagate takes $\Theta(w)$ time, so initialisation of first column of reachability matrix takes $\Theta(\ell w)$ time. Note that, at any further point, we may have at most 2^{2w+1} distinct reachability vectors; for each of them, we get ℓ calls to Propagate, taking $\Theta(4^w \ell w)$ time per column, so over all columns we need $\Theta(4^w \ell w n)$ time. If we assume that adding an element to the set takes amortised constant time, then the previous value dominates. Finally, the check at the end takes $\Theta(4^w)$ time. So, overall the algorithm runs in time $\Theta(4^w \ell n w)$. This agrees with our hardness result: for a small fixed-width time band, we get the reasonable running time of $\Theta(\ell n)$, whereas if we set $w = n - 1$ to compute the unrestricted distance, it will take $\Theta(4^n \ell n^2)$—clearly, exponential time.

Improvements An attentive reader might have noticed that we do not actually even need to store all the reachability vectors. The upper bound chooses the vectors that are most `False`, in a way—so, whenever one vector dominates another in terms of `False`-values, we only keep the dominating one. This improvement brings down the number of possible vectors to $\Theta(4^w \cdot w^{-0.5})$, thus improving the running time of the algorithm by a factor of $\sqrt{w}$.

Summarising our discussion so far, we arrive at the following conclusion.

Theorem 19. *Problem Upper Bound Discrete Fréchet restricted to Sakoe–Chiba time band of width w on trajectories of length n of indecisive points with ℓ options can be solved in time $\Theta(4^w \ell n \sqrt{w})$ in the worst case.*

Proof. Immediately follows from the discussion above. ∎

4.2 Upper Bound Discrete Fréchet Distance: Indecisive

Now we extend our result of the previous section to the setting where both trajectories are indecisive, so $\mathcal{H}$ is defined as before, and instead of V we have $\mathcal{V} = \langle V_1, \ldots, V_n \rangle$, with, for each $j \in [n]$, $V_j = \{q_j^1, \ldots, q_j^\ell\}$.

The setting is actually not too different. Imagine that we were to pick a specific realisation for trajectory $\mathcal{V}$. Then we could apply the algorithm of the previous section immediately. Of course, we cannot run it separately for

every single realisation; instead, we note that the part of the realisation that matters when filling out column i is the points from $i - w$ to $i + w$, since any previous or further points are outside the time band. So, we can fix these $2w + 1$ points and compute the column as in the previous algorithm. We can do so for each possible combination on these $2w + 1$ points. Continuing in this manner, we get an algorithm very similar to the previous section. The pseudocode is given in Algorithm 2. We denote the distance column vector taken for combination of realisations number s on $\mathcal{V}$, for column i and realisation number k of the last point on $\mathcal{H}$, by $D_{i,k}[s]$.

Correctness The correctness of the algorithm follows from the correctness of the one in the previous section, combined with the fact that our search for all realisations of interest on $\mathcal{V}$ ensures consistency. Let us show this in a more explicit manner.

Lemma 20. *Any reachability vector we store in column i corresponds to some realisation of the subtrajectories $\mathcal{H}[1 : i]$ and $\mathcal{V}[1 : \min(i + w, n)]$, and every such realisation has the resulting reachability vector stored in column i.*

Proof. First of all, consider the statement for column 1. Clearly, we consider all possible realisations of both subtrajectories, so the statement holds.

Now, as we move from column i to column $i + 1$, we fix the realisation of points $i - w + 1$ to $i + w + 1$ on trajectory $\mathcal{V}$ and consider all the vectors stemming from the possible values of point $i - w$; as in Lemma 18, we cover all realisations of trajectory $\mathcal{H}$.

As for trajectory $\mathcal{V}$, note that we, again, only need the reachability from the previous column and the distance matrix from the current column, so the points before $i - w + 1$ do not play a role for the consistency between the two, and thus they can be ignored.

So, we only get reachability vectors corresponding to valid realisations, and we do not miss any, as required. ∎

This means that in the very last cell we get correct reachability options corresponding to realisations; if at least one of them is `False`, then clearly the result should be false.

Running time The running time is quite a bit worse; it is easy to see, due to the changes, that overall it now takes $\Theta(4^w n w \ell^{2w})$. For small constant w and ℓ, we get $\Theta(n)$; for $w = n - 1$, we get $\Theta(4^n n^2 \ell^{2n})$—again, exponential time in n. As in the previous algorithm, we could store the boolean vectors more efficiently, thus reducing the running time by a factor of $\sqrt{w}$.

Algorithm 2 Finding time-banded upper bound discrete Fréchet distance on two indecisive trajectories.

1: **function** TBDFDIndInd($\mathcal{H}, \mathcal{V}, w, \varepsilon$)
▷ Input constraint: $|\mathcal{H}| = |\mathcal{V}| = n$ and $0 \leq w < n$
2: Initialise matrix D of size $n \times \ell \times (2w+1) \times \ell$
3: **for all** $i \in [n]$ **do**
4: **for all** $k \in [\ell]$ **do**
5: **for all** $j \in \{\max(1, i-w), \ldots, \min(n, i+w)\}$ **do**
6: **for all** $s \in [\ell]$ **do**
7: $D_{i,k,j,s} \leftarrow [d(p_i^k, q_j^s) \leq \varepsilon ?]$
8: Initialise matrix R of size $n \times \ell^{2w+1} \times 2^{2w+1} \times 2w+1$
9: $R0 \leftarrow \langle r_1 = \mathsf{True}, r_2 = \mathsf{False}, r_3 = \mathsf{False}, \ldots, r_{w+1} = \mathsf{False} \rangle$
10: **for all** $s \in [\ell^{w+1}]$ **do**
11: **for all** $k \in [\ell]$ **do**
12: $R_{1,s,k} \leftarrow$ Propagate($R0, D_{1,k}[s], 1, w, n$)
13: **for all** $i \in [n] \setminus \{1\}$ **do**
14: **for all** $s \in [\ell^{2w+1}]$ **do** ▷ Or fewer in edge cases
15: **for all** $A \in R_{i-1}[s]$ **do** ▷ For each reachability vector with fixed realisation
16: $B \leftarrow A \vee (A << 1)$
17: **for all** $k \in [\ell]$ **do**
18: $C \leftarrow$ Propagate($B, D_{i,k}[s], i, w, n$)
19: Add C to set $R_i[s]$
20: $r \leftarrow \mathsf{True}$
21: **for all** $A \in R_n$ **do**
22: **for all** $s \in [\ell^{2w+1}]$ **do**
23: $r \leftarrow r \wedge A_n[s]$
24: **return** r
25: **function** Propagate(A, B, i, w, n)
▷ Propagate the reachability upwards in a column
26: $C \leftarrow A \wedge B$
27: $r \leftarrow \mathsf{False}$
28: **for all** $j \in \{\max(1, i-w), \ldots, \min(n, i+w)$ **do**
29: **if** $B_j \wedge C_j$ **then**
30: $r \leftarrow \mathsf{True}$
31: **else if** $B_j \wedge \neg C_j \wedge r$ **then**
32: $C_j \leftarrow \mathsf{True}$
33: **else if** $\neg B_j$ **then**
34: $r \leftarrow \mathsf{False}$
35: **return** C

We summarise the results as follows.

Theorem 21. *Suppose we are given two indecisive trajectories of length n with ℓ options per indecisive point. Then we can compute upper bound discrete Fréchet distance restricted to Sakoe–Chiba band of width w using algorithm TBDFDIndInd in time $\Theta(4^w n\sqrt{w}\ell^{2w})$.*

Proof. Follows immediately from previous discussion and Lemma 20. ■

4.3 Expected Discrete Fréchet Distance

Consider first the setting of one precise and one indecisive trajectory. Note that we store the reachability vectors in a set; instead, we could store a counter with each reachability vector, so that every time we get an element that is already stored, we increment the counter. Notice that we cannot use the improvement that would allow us to discard some vectors, as that would eschew the count, and we are not interested in the worst possible result now. We can implement a similar mechanism in the setting of two indecisive trajectories. Moreover, we can clearly propagate the count through the algorithm and in the end find the counts associated with answers True and False to the decision problem.

So, if we store the count of realisations that give us a certain reachability vector, we essentially obtain, for some value of ε,

$$\mathbb{P}(d_{\mathrm{dF}}(A, B) > \varepsilon) \quad \text{when } A, B \Subset_{\mathbb{U}} \mathcal{H}, \mathcal{V}.$$

For any realisation, there is a specific value of ε—a *critical value*—that acts as a threshold between the answers True and False for that realisation, since if we fix the realisation we just compute the regular discrete Fréchet distance. Note that that threshold must be a distance between some two points on different curves. In the case of a precise and an indecisive trajectory, there are $\ell n(2w + 1)$ such distances with the time band of width w; in the case of two indecisive trajectories, there are $\ell^2 n(2w + 1)$ such distances. Therefore, if we run our algorithm for each of these critical values and record the counts of True and False for each threshold, we will obtain the complete cumulative distribution function $\mathbb{P}(d_{\mathrm{dF}}(A, B) > \varepsilon)$ for $A, B \Subset_{\mathbb{P}} \mathcal{H}, \mathcal{V}$.

Then we can simply find, under the time band restriction,

$$d_{\mathrm{dF}}^{\mathbb{E}(\mathbb{U})}(\mathcal{H}, \mathcal{V}) = \int_0^\infty \mathbb{P}_{A,B \Subset_{\mathbb{P}} \mathcal{H},\mathcal{V}}(d_{\mathrm{dF}}(A, B) > \varepsilon)\, \mathrm{d}\varepsilon\,.$$

For any realisation the answer may change from True to False only at one of the critical values. So, the distribution of True and False only changes

at a finite set of critical values and is constant between them; therefore, $\mathbb{P}(d_{\mathrm{dF}}(A, B) > \varepsilon)$ is a step function. Hence, finding the integral of interest amounts to multiplying the value of $\mathbb{P}(d_{\mathrm{dF}}(A, B) > \varepsilon)$ by the distance between two successive values of ε that match, and summing all the results, i.e. to finding the area under the step function by summing up areas of rectangles that make it up.

So, clearly, under the time band restriction, we can run one of our algorithms either $\ell n(2w + 1)$ or $\ell^2 n(2w + 1)$ times to obtain the expected discrete Fréchet distance. We show the details in Algorithm 3 for the two settings. We summarise this result as follows.

Theorem 22. *Suppose we are given an indecisive trajectory $\mathcal{H}$ and a precise trajectory V of length n with ℓ options per indecisive point and want to find the expected discrete Fréchet distance when constrained to Sakoe–Chiba band of width w. Then we can run* ExpTBDFDIndPr$(\mathcal{H}, V, w)$ *to obtain the result in time $\Theta(4^w \ell^2 n^2 w^2)$ in the worst case.*

Proof. First of all, note that from the discussion above it immediately follows that the algorithm is correct. In the worst case, every ε that we have to add to E will be distinct, so we have $\ell n(2w + 1)$ insertions, taking in total $\Theta(\ell nw \log \ell nw)$ time. Then, we run CntTBDFDIndPr once per value in E, and its running time is the same as that of TBDFDIndPr, so here we take time $\Theta(\ell nw \cdot 4^w \ell nw)$ in the worst case, as claimed. ∎

We can formalise the result similarly for the other setting.

Theorem 23. *Suppose we are given two indecisive trajectories $\mathcal{H}$ and $\mathcal{V}$ of length n with ℓ options per indecisive point and want to find the expected discrete Fréchet distance when constrained to Sakoe–Chiba band of width w. Then we can run* ExpTBDFDIndInd$(\mathcal{H}, \mathcal{V}, w)$ *to obtain the result in time $\Theta(4^w n^2 w^2 \ell^{2w})$ in the worst case.*

Proof. Again, note that from the discussion above it immediately follows that the algorithm is correct. In the worst case, we have $\ell^2 nw$ insertions, taking in total $\Theta(\ell^2 nw \log \ell nw)$ time. Then, we run CntTBDFDIndInd once per value in E, and its running time is the same as that of TBDFDIndInd, so here we take time $\Theta(\ell^2 nw \cdot 4^w nw \ell^{2w})$ in the worst case, as claimed. ∎

4.4 Upper Bound Continuous Fréchet Distance

In principle, one could adapt the algorithms for the upper bound discrete Fréchet distance to the case when either both trajectories are indecisive or

Algorithm 3 Finding time-banded expected discrete Fréchet distance on an indecisive and a precise trajectory and two indecisive trajectories.

function ExpTBDFDIndPr($\mathcal{H}, V, w$)
▷ Input constraint: $|\mathcal{H}| = |V| = n$ and $0 \leq w < n$
  Initialise sorted set E
  for all $i \in [n]$ **do**
    for all $k \in [\ell]$ **do**
      for all $j \in \{\max(1, i-w), \ldots, \min(n, i+w)\}$ **do**
        Add $d(p_i^k, q_j)$ to sorted set E
  $s \leftarrow E[1]$
  for $i \leftarrow 1$ **to** $l(E) - 1$ **do**
    $\varepsilon \leftarrow E[i], \varepsilon' \leftarrow E[i+1]$
    $p \leftarrow$ CntTBDFDIndPr($\mathcal{H}, V, w, \varepsilon$)
    $s \leftarrow s + (1-p) \cdot (\varepsilon' - \varepsilon)$
  return s
function ExpTBDFDIndInd($\mathcal{H}, \mathcal{V}, w$)
▷ Input constraint: $|\mathcal{H}| = |\mathcal{V}| = n$ and $0 \leq w < n$
  Initialise sorted set E
  for all $i \in [n]$ **do**
    for all $k \in [\ell]$ **do**
      for all $j \in \{\max(1, i-w), \ldots, \min(n, i+w)\}$ **do**
        for all $s \in [\ell]$ **do**
          Add $d(p_i^k, q_j^s)$ to sorted set E
  $s \leftarrow E[1]$
  for $i \leftarrow 1$ **to** $l(E) - 1$ **do**
    $\varepsilon \leftarrow E[i], \varepsilon' \leftarrow E[i+1]$
    $p \leftarrow$ CntTBDFDIndInd($\mathcal{H}, \mathcal{V}, w, \varepsilon$)
    $s \leftarrow s + (1-p) \cdot (\varepsilon' - \varepsilon)$
  return s
function CntTBDFDIndPr($\mathcal{H}, V, w, \varepsilon$)
▷ Like TBDFDIndPr, but returns the fraction of count of True over False for the final cell.
function CntTBDFDIndInd($\mathcal{H}, \mathcal{V}, w, \varepsilon$)
▷ Like TBDFDIndInd, but returns the fraction of count of True over False for the final cell.

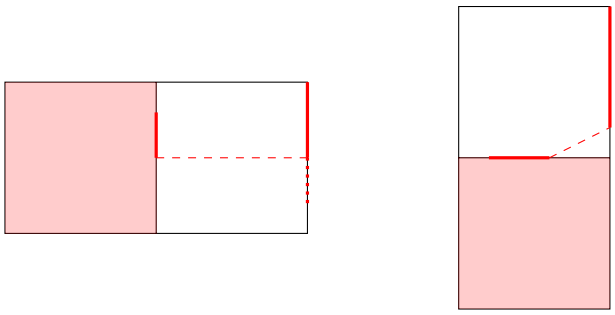


FIGURE 4.1: Reachability adjustments. Left: Although the dotted interval is free according to the distance matrix, only the solid interval is reachable from the cell on the left with a monotone path, assuming entire cell on the left is free. Right: The entire interval that is marked as free according to the distance matrix is reachable with a monotone path from the cell below, assuming the cell below is free.

one is precise and one is indecisive, and we are interested in the decision problem for Fréchet distance and not discrete Fréchet distance. Since we are going column-by-column, we would need to store the reachability intervals on the vertical border of each cell.

It is simpler to see how this would work in the setting of a precise and an indecisive trajectory: each column now is a column of a free-space diagram, and we only need to store the intervals on the right side of the column. As we progress to the next column, we need to consider all the options from the previous column, so we need to run the same algorithm, except we store and process vectors of free-space intervals instead of `True` and `False`. One other distinction is that we do not consider diagonal steps—for Fréchet distance doing so would not make any sense, as the path is continuous, and the diagonal step is not distinguishable from a horizontal step followed by a vertical step, if such situation occurs.

In particular, we now take the intervals stored in the distance matrix and compute reachability based on the previous column: if a cell can be reached horizontally from the previous cell, then the lower bound of the interval in this cell may need to go up, since we can only use monotone paths. PROPAGATE will now take the intervals that correspond to the distance matrix and the precomputed reachability and make the following adjustment: if a cell is reachable from below, then the entire interval on the right is actually reachable. See Figure 4.1 for an example of both cases.

Other than that, the algorithm is exactly the same; clearly, we can make the same adjustments to the algorithm handling two indecisive trajectories.

Notice that we now do not have at most 2^{2w+1} vectors per column, since

we store intervals instead of boolean values, and they can be more varied. However, the number of values is still limited: for any cell, the upper value of the interval is defined by the distance matrix, so there can be at most ℓ or ℓ^2 values for the two settings. The lower value of the interval is defined by the distance matrix or by one of the cells from the same row; these may have at most ℓ or ℓ^2 values each, and there are at most $2w$ of them, so per cell we can have at most $\Theta(\ell w)$ or $\Theta(\ell^2 w)$ lower interval values and $\Theta(\ell)$ or $\Theta(\ell^2)$ upper interval values, instead of just two possible values in the discrete case. The running time changes accordingly, replacing 4^w with $(\ell w)^{2w}$, but, importantly, we still have linear dependency on n, so the running time is polynomial for fixed w and ℓ.

4.5 Expected Continuous Fréchet Distance

We can, of course, again store the associated counts with the vectors of intervals in the algorithm. As we look at the final cell, we can sum up the counts associated with the cases where the upper right corner of this cell is reachable, and so we can find the proportion of `True` to `False` for a particular threshold ε.

Again, we can find critical values; this time they follow in line with those discussed in Section 2.1. The number of the critical values will be different, however: case 1, where we look at the start and end points, now yields $\Theta(\ell^2)$ events; case 2, where we look at two neighbouring cells, so at the distance between a segment and a point, yields $\Theta(\ell^3 nw)$ events; and case 3, where we look at the distance between a segment and two points, yields $\Theta(\ell^4 nw^2)$ events.

Otherwise, we can run Algorithm 3 on the new critical values, calling instead the counting version for the continuous Fréchet distance. This way we can compute the expected Fréchet distance restricted to a Sakoe–Chiba band in time polynomial in n for fixed w and ℓ, as desired.

5

Conclusion

Trajectory data is ubiquitous, and uncertainty is an important aspect to it, and yet there is very little work on uncertain trajectories in general and their similarity in particular.

In this thesis, we approach the topic of similarity of uncertain trajectories, focusing on simple region-based models for measurement uncertainty and Fréchet distance as the similarity metric.

This thesis generalises previous work in two areas:

- Previously, there has been work that would cover lower and upper bounds in various settings in point sets. We aim to provide the same wide spectrum of approaches on uncertain trajectories.
- Previously, there has been very little work on expected value of certain outputs on uncertain points. We study expected discrete Fréchet distance and Fréchet distance on uncertain trajectories.

Both of these extensions make positive algorithmic results difficult, as showcased in Chapter 3 with the proofs of NP-completeness for the upper bound setting and #P-hardness for the expected value setting.

However, there are some restricted versions of the same problems that are feasible to compute—we show this with the example of algorithms using time bands. Time bands are a reasonable restriction that naturally arises in the context of trajectory analysis, when temporal synchronisation of the two trajectories is desired.

This thesis shows that the problems related to upper bound and expected value quickly get difficult to compute in general, even in rather simple uncertainty settings. However, bringing in realistic assumptions on the input and the desired outcome helps to get polynomial-time algorithms.

This work provides a promising starting point for a line of future research on uncertain trajectories. There are some open problems related to similarity of uncertain trajectories. In particular, lower bound continuous Fréchet distance on uncertain trajectories has not been proven difficult or solved. It could also be interesting to look at the problems discussed in this thesis without assuming that $\ell \in \Theta(1)$ for the indecisive point setting—for instance, if we obtain the indecisive points by sampling some unknown probability distribution, we might be interested in optimising the running time of algorithms in terms of ℓ.

There are also many other possible uncertainty models, both for measurement uncertainty considered in this thesis and for movement uncertainty, like the Brownian bridge model; these approaches may be better suited to realistically uncertain trajectories, although the problems in such models may be even more difficult to compute than in the model treated in this thesis, so more realistic assumptions about the data would be needed.

In that direction, one could also consider moving towards results that are practically relevant for application areas—there one would need fast algorithms that give useful results, while operating under reasonable restrictions on the data.

Finally, we could consider other analysis goals on uncertain trajectories, like simplification, clustering, or segmentation. Overall, there are many possible directions for future work on uncertain trajectories, taking this thesis as a starting point.

Bibliography

[1] Pankaj K. Agarwal, Rinat Ben Avraham, Haim Kaplan and Micha Sharir. 'Computing the Discrete Fréchet Distance in Subquadratic Time'. In: *SIAM Journal on Computing* 43.2 (2014), pp. 429–449. ISSN: 0097-5397. DOI: 10.1137/130920526. arXiv: 1204.5333v1 [cs.CG].

[2] Pankaj K. Agarwal, Sariel Har-Peled, Nabil H. Mustafa and Yusu Wang. 'Near-Linear Time Approximation Algorithms for Curve Simplification'. In: *Algorithmica* 42.3 (July 2005), pp. 203–219. ISSN: 1432-0541. DOI: 10.1007/s00453-005-1165-y.

[3] Hee-Kap Ahn, Christian Knauer, Marc Scherfenberg, Lena Schlipf and Antoine Vigneron. 'Computing the Discrete Fréchet Distance with Imprecise Input'. In: *International Journal of Computational Geometry & Applications* 22.01 (2012), pp. 27–44. DOI: 10.1142/S0218195912600023.

[4] Sander P. A. Alewijnse, Kevin Buchin, Maike Buchin, Andrea Kölzsch, Helmut Kruckenberg and Michel A. Westenberg. 'A Framework for Trajectory Segmentation by Stable Criteria'. In: *Proceedings of the 22nd ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems*. SIGSPATIAL '14. New York, NY, USA: ACM, 2014, pp. 351–360. ISBN: 9781450331319. DOI: 10.1145/2666310.2666415.

[5] Sander P. A. Alewijnse, Kevin Buchin, Maike Buchin, Stef Sijben and Michel A. Westenberg. 'Model-Based Segmentation and Classification of Trajectories'. In: *Algorithmica* 80.8 (Aug. 2018), pp. 2422–2452. ISSN: 1432-0541. DOI: 10.1007/s00453-017-0329-x.

[6] Helmut Alt and Michael Godau. 'Computing the Fréchet Distance Between Two Polygonal Curves'. In: *International Journal of Computa-*

tional Geometry and Applications 5.1 (1995), pp. 75–91. DOI: 10.1142/S0218195995000064.

[7] Boris Aronov, Anne Driemel, Marc van Kreveld, Maarten Löffler and Frank Staals. 'Segmentation of Trajectories on Non-Monotone Criteria'. In: *ACM Transactions on Algorithms* 12.2 (Dec. 2015), 26:1–26:28. ISSN: 1549-6325. DOI: 10.1145/2660772.

[8] Karl Bringmann. 'Why Walking the Dog Takes Time. Fréchet Distance Has No Strongly Subquadratic Algorithms Unless SETH Fails'. In: *2014 IEEE 55th Annual Symposium on Foundations of Computer Science*. Piscataway, NJ, USA: IEEE, 1st Aug. 2014, pp. 661–670. DOI: 10.1109/FOCS.2014.76. arXiv: 1404.1448v2 [cs.CG].

[9] Karl Bringmann and Bhaskar Ray Chaudhury. 'Polyline Simplification has Cubic Complexity'. In: *35th International Symposium on Computational Geometry (SoCG 2019)*. Vol. 129. Leibniz International Proceedings in Informatics (LIPIcs). Dagstuhl, Germany: Schloss Dagstuhl–Leibniz-Zentrum für Informatik, 2019, 18:1–18:16. ISBN: 9783959771047. DOI: 10.4230/LIPIcs.SoCG.2019.18.

[10] Kevin Buchin, Maike Buchin and Joachim Gudmundsson. 'Constrained Free Space Diagrams. A Tool for Trajectory Analysis'. In: *International Journal of Geographical Information Science* 24.7 (July 2010), pp. 1101–1125. ISSN: 1365-8816. DOI: 10.1080/13658810903569598.

[11] Kevin Buchin, Maike Buchin, Wouter Meulemans and Wolfgang Mulzer. 'Four Soviets Walk the Dog. Improved Bounds for Computing the Fréchet Distance'. In: *Discrete & Computational Geometry* 58.1 (2017), pp. 180–216. ISSN: 1432-0444. DOI: 10.1007/s00454-017-9878-7.

[12] Kevin Buchin, Anne Driemel, Joachim Gudmundsson, Michael Horton, Irina Kostitsyna, Maarten Löffler and Martijn Struijs. 'Approximating (k, ℓ)-center Clustering for Curves'. In: *Proceedings of the Thirtieth Annual ACM–SIAM Symposium on Discrete Algorithms*. Philadelphia, PA, USA: SIAM, 2019, pp. 2922–2938. ISBN: 9781611975482. DOI: 10.1137/1.9781611975482.181.

[13] Kevin Buchin, Stef Sijben, T. Jean Marie Arseneau and Erik P. Willems. 'Detecting Movement Patterns Using Brownian Bridges'. In: *Proceedings of the 20th International Conference on Advances in Geographic Information Systems*. SIGSPATIAL '12. New York, NY, USA: ACM, 2012, pp. 119–128. ISBN: 9781450316910. DOI: 10.1145/2424321.2424338.

[14] Maike Buchin and Stef Sijben. *Discrete Fréchet Distance for Uncertain Points*. Presented at EuroCG 2016, Lugano, Switzerland. 2016. URL: http://www.eurocg2016.usi.ch/sites/default/files/paper_72.pdf (visited on 10/07/2019).

[15] Leizhen Cai and Mark Keil. 'Computing Visibility Information in an Inaccurate Simple Polygon'. In: *International Journal of Computational Geometry & Applications* 7 (Dec. 1997), pp. 515–538. DOI: 10.1142/S0218195997000326.

[16] Lei Chen, M. Tamer Özsu and Vincent Oria. 'Robust and Fast Similarity Search for Moving Object Trajectories'. In: *Proceedings of the 2005 ACM SIGMOD International Conference on Management of Data*. SIGMOD '05. New York, NY, USA: ACM, 2005, pp. 491–502. ISBN: 9781595930606. DOI: 10.1145/1066157.1066213.

[17] Stephen Arthur Cook. 'The Complexity of Theorem-Proving Procedures'. In: *Proceedings of the Third Annual ACM Symposium on Theory of Computing*. STOC '71. Menlo Park, CA, USA: ACM, 1971, pp. 151–158. DOI: 10.1145/800157.805047.

[18] David H. Douglas and Thomas K. Peucker. 'Algorithms for the Reduction of the Number of Points Required to Represent a Digitized Line or its Caricature'. In: *Cartographica. The International Journal for Geographic Information and Geovisualization* 10.2 (1973), pp. 112–122. DOI: 10.3138/FM57-6770-U75U-7727.

[19] Anne Driemel and Sariel Har-Peled. 'Jaywalking Your Dog. Computing the Fréchet Distance with Shortcuts'. In: *SIAM Journal on Computing* 42.5 (22nd Oct. 2018), pp. 1830–1866. ISSN: 0097-5397. DOI: 10.1137/120865112. arXiv: 1107.1720v4 [cs.CG].

[20] Thomas Eiter and Heikki Mannila. *Computing Discrete Fréchet Distance*. Tech. rep. CD-TR 94/64. Technishe Universität Wien, 25th Apr. 1994. URL: http://www.kr.tuwien.ac.at/staff/eiter/et-archive/cdtr9464.pdf (visited on 23/04/2019).

[21] Martin Ester, Hans-Peter Kriegel, Jörg Sander and Xiaowei Xu. 'A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise'. In: *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining (KDD-96)*. Menlo Park, CA, USA: AAAI Press, 1996, pp. 226–231. ISBN: 9781577350040. URL: https://www.aaai.org/Papers/KDD/1996/KDD96-037.pdf (visited on 10/09/2019).

[22] William Evans, David Kirkpatrick, Maarten Löffler and Frank Staals. 'Competitive Query Strategies for Minimising the Ply of the Potential Locations of Moving Points'. In: *Proceedings of the Twenty-Ninth Annual Symposium on Computational Geometry*. SoCG '13. New York, NY, USA: ACM, 2013, pp. 155–164. ISBN: 9781450320313. DOI: 10.1145/2462356.2462395.

[23] Chenglin Fan and Binhai Zhu. *Complexity and Algorithms for the Discrete Fréchet Distance Upper Bound with Imprecise Input*. 5th Feb. 2018. arXiv: 1509.02576v2 [cs.CG].

[24] Michael Godau. 'A Natural Metric for Curves. Computing the Distance for Polygonal Chains and Approximation Algorithms'. In: *STACS 91. Proceedings of 8th Annual Symposium on Theoretical Aspects of Computer Science*. LNCS 480. Berlin, Germany: Springer Berlin Heidelberg, 1991, pp. 127–136. ISBN: 9783540470021. DOI: 10.1007/BFb0020793.

[25] Oded Goldreich. *Computational Complexity. A Conceptual Perspective*. Cambridge, England: Cambridge University Press, 2008. Chap. 6, pp. 202–205. ISBN: 9780521884730. DOI: 10.1017/CBO9780511804106.

[26] Joachim Gudmundsson, Jyrki Katajainen, Damian Merrick, Cahya Ong and Thomas Wolle. 'Compressing Spatio-Temporal Trajectories'. In: *Computational Geometry* 42.9 (Nov. 2009), pp. 825–841. ISSN: 0925-7721. DOI: 10.1016/j.comgeo.2009.02.002.

[27] Hiroshi Imai and Masao Iri. 'Computational-Geometric Methods for Polygonal Approximations of a Curve'. In: *Computer Vision, Graphics, and Image Processing* 36.1 (1986), pp. 31–41. ISSN: 0734-189X. DOI: 10.1016/S0734-189X(86)80027-5.

[28] Fumitada Itakura. 'Minimum Prediction Residual Principle Applied to Speech Recognition'. In: *IEEE Transactions on Acoustics, Speech, and Signal Processing* 23.1 (Feb. 1975), pp. 67–72. ISSN: 0096-3518. DOI: 10.1109/TASSP.1975.1162641.

[29] Christian Knauer, Maarten Löffler, Marc Scherfenberg and Thomas Wolle. 'The Directed Hausdorff Distance Between Imprecise Point Sets'. In: *Theoretical Computer Science* 412.32 (2011), pp. 4173–4186. ISSN: 0304-3975. DOI: 10.1016/j.tcs.2011.01.039.

[30] Joseph B. Kruskal and Mark Liberman. 'The Symmetric Time-Warping Problem. From Continuous to Discrete'. In: *Time Warps, String Edits, and Macromolecules. The Theory and Practice of Sequence Comparison*. Ed. by David Sankoff and Joseph B. Kruskal. Reading, MA, USA: Addison–Wesley, 1983, pp. 125–161. ISBN: 9781575862170.

[31] Leonid Levin. 'Универсальные задачи перебора (Universal Enumeration Problems)'. Trans. from the Russian by Boris Trakhtenbrot. In: *Проблемы передачи информации (Problems of Information Transmission)* 9.3 (7th June 1972), pp. 265–266.

[32] Maarten Löffler. 'Data Imprecision in Computational Geometry'. PhD thesis. Universiteit Utrecht, 19th Oct. 2009. ISBN: 9789088911217. URL: https://dspace.library.uu.nl/bitstream/handle/1874/36022/loffler.pdf (visited on 15/06/2019).

[33] Maarten Löffler and Jeff M. Phillips. 'Shape Fitting on Point Sets with Probability Distributions. ESA 2009'. In: *Algorithms*. LNCS 5757. Berlin, Germany: Springer Berlin Heidelberg, 2009, pp. 313–324. ISBN: 9783642041280. DOI: 10.1007/978-3-642-04128-0_29. arXiv: 0812.2967v1 [cs.CG].

[34] Urs Ramer. 'An Iterative Procedure for the Polygonal Approximation of Plane Curves'. In: *Computer Graphics and Image Processing* 1.3 (1972), pp. 244–256. ISSN: 0146-664X. DOI: 10.1016/S0146-664X(72)80017-0.

[35] Hiroaki Sakoe and Seibi Chiba. 'Dynamic Programming Algorithm Optimization for Spoken Word Recognition'. In: *IEEE Transactions on Acoustics, Speech, and Signal Processing* 26.1 (Feb. 1978), pp. 43–49. ISSN: 0096-3518. DOI: 10.1109/TASSP.1978.1163055.

[36] Michail Vlachos, George Kollios and Dimitrios Gunopulos. 'Discovering Similar Multidimensional Trajectories'. In: *Proceedings 18th International Conference on Data Engineering*. Piscataway, NJ, USA: IEEE, 2002, pp. 673–684. ISBN: 9780769515311. DOI: 10.1109/ICDE.2002.994784.