Fréchet Edit Distance

Emily Fox
Department of Computer Science, University of Texas at Dallas, TX, USA

Amir Nayyeri
School of Electrical Engineering and Computer Science,
Oregon State University, Corvallis, OR, USA

Jonathan James Perry
Department of Computer Science, University of Texas at Dallas, TX, USA

Benjamin Raichel
Department of Computer Science, University of Texas at Dallas, TX, USA

Abstract

We define and investigate the Fréchet edit distance problem. Given two polygonal curves π and σ and a threshhold value $\delta > 0$, we seek the minimum number of edits to σ such that the Fréchet distance between the edited σ and π is at most δ. For the edit operations we consider three cases, namely, deletion of vertices, insertion of vertices, or both. For this basic problem we consider a number of variants. Specifically, we provide polynomial time algorithms for both discrete and continuous Fréchet edit distance variants, as well as hardness results for weak Fréchet edit distance variants.

2012 ACM Subject Classification Theory of computation → Computational geometry

Keywords and phrases Fréchet distance, Edit distance, Hardness

Digital Object Identifier 10.4230/LIPIcs.SoCG.2024.58

Related Version *Full Version*: `https://arxiv.org/abs/2403.12878`

Funding *Emily Fox*: Work on this paper was partially supported by NSF CAREER Award 1942597 and CCF Award 2311179.
Amir Nayyeri: Work on this paper was partially supported by NSF Awards CCF 2311180 and CCF 1941086.
Jonathan James Perry: Work on this paper was partially supported by NSF CAREER Award 1750780 and CCF Award 2311179.
Benjamin Raichel: Work on this paper was partially supported by NSF CAREER Award 1750780 and CCF Award 2311179.

1 Introduction

1.1 Motivation

We consider the general problem of shape matching between polygonal curves. In a standard formulation of this problem, one is given two sequences of points embedded in a common ambient space like $\mathbb{R}^d$ with d a constant. Depending on the specific application, these inputs may be interpreted directly as the discrete point sequences they are or as the vertices of continuous curves obtained by interpolating between contiguous sequence points.

The computational geometry community has strongly promoted the use of the Fréchet distance to handle determining similarity of the curves and matching corresponding portions. The *continuous* Fréchet distance is often presented using the walks of a person and their dog along the curves; both entities move at any positive variable speed from the beginning to the end of their respective curve, and one seeks the smallest length of a leash needed to keep them connected during their walks. For the *discrete* variant, the person and dog are replaced by a leashed pair of frogs that iteratively hop between contiguous vertices, either individually or at the same time, and the length of the leash is only considered during the moments between hops. Some prior works have also considered the *weak* variants of continuous and discrete Fréchet, where the entities are allowed to move *backwards* at times to keep their leashes short. Beyond its theoretical interest, the Fréchet distance has seen use in mapping and map construction [2, 10], handwriting recognition [23], and protein alignment [20].

We naturally consider two curves to be similar if their Fréchet distance does not exceed some predetermined threshold value δ. This notion of similarity allows for a single choice of δ that can be used regardless of the curves' length, and it is resilient to differing sampling rates (as long as the sequences are sufficiently dense in the case of discrete Fréchet). Unfortunately, this intuitively satisfying notion of similarity has some severe issues once we start applying it to noisy real world data such as GPS traces from individuals' phones or vehicles. In particular, nearly all variants of the Fréchet distance are extremely sensitive to outliers. Adding even a single point to one of the input curves can increase their distance by an arbitrarily high amount if that point lies far away from the other curve, and this issue is present regardless of how many points are present in the curves' input sequences. Similarly, a sparsely sampled continuous curve can change dramatically if even a single vertex is ignored.

1.2 The Fréchet Edit Distance

Multiple modifications of and even alternatives to the Fréchet distance have been proposed to address the issue described above, and we review the most relevant of these alternatives in Section 1.4. At the end of the day, though, we want to keep that our final notion of similarity is based on the standard definitions of Fréchet distance as it remains the best tool we have for working with continuous and densely sampled discrete curves.

We take inspiration from the string edit distance (Levenstein distance). Viewing the input curves' point sequences as a pair of strings, we ask for the minimum number of edits (point deletions and/or insertions) needed to bring one curve within Fréchet distance δ of the other. Intuitively, the fewer edits needed, the more likely it is that the input curves really do represent two instances of the same or at least very similar trajectories through the ambient space. Depending on which of the above variants of the Fréchet distance we use and which edit operations we allow, we obtain one of several specific similarity measures between the curves. However, we refer to any of these combinations under the general term *Fréchet edit distance*. We give the formal definitions and notation for these measures in Section 2.

1.3 Our Results

We describe polynomial time algorithms and NP-hardness results for nearly every variant of Fréchet edit distance proposed above. Let m and n denote the number of points in the two input sequences, with n denoting the number of points in the sequence that can be edited.

1. We describe polynomial time algorithms for certain cases of Fréchet edit distance using the **strong continuous Fréchet distance**. When limited to deletions, in any $\mathbb{R}^d$ we can compute the Fréchet edit distance in $O(mn^3)$ time. If only k deletions are needed, our algorithm can be made to run in $O(k^2mn)$ time. Further, we can also handle the case when we allow deletions on *either* input curve, and the corresponding running times respectively become $O(m^3n^3)$ or $O(k^4mn)$. In the plane, $\mathbb{R}^2$, for insertions only we describe algorithms with times $O(nm^5)$, or $O(nm^3(k^2 + m\log^2 m))$ when limiting to k insertions. When we allow both deletions and insertions these times become $O((m+n)^3nm^3)$ and $O(knm^3(k^2 + m\log^2 m))$.

All of our algorithms for strong continuous Fréchet distance include an embedding of the curve(s) being edited into a *DAG complex* [19], a geometrically embedded directed acyclic graph that represents the different routes one can take through a curve *and its optionally edited portions*. For deletions, we include every direct vertex-to-vertex segment in the complex. Insertions require substantially more care, because it is not clear ahead of time where one should place the new vertices or where the new subcurves they determine will map to. In fact, a newly inserted subcurve may map to a portion on the curve not being edited that starts or ends on the interior of a segment. Despite this challenge, we can argue that one can restrict attention to a bounded set of canonical subcurves, and these subcurves can be computed with the aid of a result from [18] who describe how to compute minimum vertex curves lying within small Fréchet distance to another curve, via the computation of so-called minimum link stabbers.

2. For the **strong discrete Fréchet distance** with edits limited to deletions, we describe an $O(mn)$ time algorithm for any pair of curves in $\mathbb{R}^d$. This result cannot be improved upon by any polynomial factor without violating a conditional lower bound known for the discrete Fréchet distance itself [5, 6]. For insertions (with or without deletions as well), the running time becomes $O(m^2 + mn)$. These algorithms use relatively straightforward dynamic programming recurrences, although we do some non-trivial precomputation to compute a small set of positions in which to insert new points.
3. We show that the variant with **weak discrete Fréchet distance** is NP-hard even for curves in $\mathbb{R}^1$ when attempting to minimize the number of deletions, minimize the number of insertions, and minimize the number of either kind of edit. In fact, even determining if *any* number of deletions leads to small weak discrete Fréchet distance is NP-hard. These results can be extended to **weak continuous Fréchet distance** after moving to the plane $\mathbb{R}^2$. All of our hardness results are shown by a reduction from 3SAT using a similar argument to that used in [7] for the hardness of finding a minimum weak discrete Fréchet distance realization for uncertain curves in $\mathbb{R}^1$.

In addition to deletions and insertions, our results can be extended in a straightforward manner to include *substitutions* as a third possible edit operations for the Fréchet edit distance. We defer the details to the future journal version of the paper.

1.4 Related and Improved Upon Prior Work

As far as we are aware, we are the first to consider this particular measure of similarity in full generality, although there is past work that comes close. The most relevant large body of work concerns the *shortcut Fréchet distance* between curves where one asks for the minimum Fréchet distance possible after replacing disjoint subcurves with line segment shortcuts [14, 8, 13, 4, 15]. For continuous curves, one can either allow the shortcuts to go between any two (interpolated) points on the curve, or restrict the shortcuts to be between vertices of the curve. This vertex-to-vertex shortcut restriction is similar to the deletion only version of Fréchet edit distance, except deletion of multiple contiguous points counts as a single shortcut operation. (By default we assume the shortcut problem is defined without a bound on the number of shortcuts allowed, though the bounded version has also been considered before, and prominently so in [13].)

Most relevant to the current work is a known $O(n^3 \log n)$ time algorithm for deciding if the continuous Fréchet distance with vertex-to-vertex shortcuts on one of two n-vertex curves is at most a given value δ. This algorithm is restricted to curves in $\mathbb{R}^2$ [8]. A slight modification to our first algorithm improves the running time to $O(n^3)$ and works for curves in any $\mathbb{R}^d$. We note that the equivalent shortcut problem for the discrete Fréchet distance has a known *linear* time solution in the plane [4]. (Recall our discrete deletion only edit distance algorithm minimizes the number of point deletions, and thus its running time cannot see a substantial improvement without violating conditional lower bounds [5, 6].) Surprisingly, the shortcut problem becomes NP-hard when shortcuts are allowed between any two points on the continuous curve [8]. Further developing the hardness picture, our Section 6 result for any number of deletions implies even vertex-to-vertex shortcutting is NP-hard if we switch from the strong to the weak Fréchet distance, with the interpretation that the curve must be shortcut before the traversal (i.e. one cannot shortcut a subset of vertices and then later go back to a vertex in the subset, which is automatically not possible in the strong version).

Buchin and Plätz [9] proposed an alternative to the above problem where one seeks the minimum Fréchet distance possible between discrete or continuous curves after removing up to k vertices on one or both curves. By wrapping them in a binary search, their algorithms can be used to solve the deletion only strong Fréchet distance versions of our problem. Our algorithms are faster than theirs by at least a $\log n$ factor in every case except allowing deletions from two continuous curves where their algorithm uses one fewer factor of k.

Leaving behind the Fréchet distance allows one to consider other distance measures that are best defined over the discrete input sequences as opposed to their interpolated curves [1, 16, 17, 12, 11, 21, 22, 24, 25]. Of particular note is the so-called *geometric edit distance* where one attempts to edit one sequence to look *exactly* like the other one, assigning smaller costs for substitutions between nearby points [1, 16, 17]. As opposed to the above measures for discrete sequences, our use of Fréchet distance allows us to work with continuous interpolations of the input sequences. Even when considering the discrete Fréchet distance, we avoid the issue of two nearly identical but offset curves from having a high distance just because they contain a large number of input points. If an input resembling two such curves results in a high Fréchet edit distance, it must be because there is a large number of *outlier* points that need to be cleaned up before similarity is evident.

2 Preliminaries

Throughout, given points $p, q \in \mathbb{R}^d$, $||p - q||$ denotes their Euclidean distance. Moreover, given two (closed) sets $P, Q \subseteq \mathbb{R}^d$, $||P - Q|| = \min_{p \in P, q \in Q} ||p - q||$ denotes their distance, where for a single point $x \in \mathbb{R}^d$ we write $||x - P|| = ||\{x\} - P||$. $B(x, r)$ will be used to denote the closed ball of radius r centered at x. We use angled brackets to denote an ordered list $\langle x_1, \ldots, x_n \rangle$, and use $L_1 \circ L_2$ to denote the concatenation of ordered lists L_1 and L_2, where for a single item x we sometimes write $x \circ L = \langle x \rangle \circ L$.

Fréchet Distance. A *polygonal curve* of length m is a sequence of m points $\pi = \langle \pi_1, \ldots, \pi_m \rangle$ where $\pi_i \in \mathbb{R}^d$ for all i. Such a sequence induces a continuous mapping from $[1, m]$ to $\mathbb{R}^d$, which we also denote by π, such that for any integer $1 \leq i < m$, the restriction of π to the interval $[i, i+1]$ is defined by $\pi(i + \alpha) = (1 - \alpha)\pi_i + \alpha\pi_{i+1}$ for any $\alpha \in [0, 1]$, i.e. a straight line segment. We will view π as both a discrete point sequence and a continuous function interchangeably, and when it is clear from the context, we also may use π to denote the image $\pi([1, m])$. We use $\pi[i, j]$, for $i \leq j$, to denote the restriction of π to the interval $[i, j]$. Given a curve $\pi = \langle \pi_1, \ldots, \pi_m \rangle$, we write $|\pi| = m$ to denote its size.

A reparameterization for a curve π of length m is a continuous non-decreasing bijection $f : [0, 1] \to [1, m]$ such that $f(0) = 1, f(1) = m$. Given reparameterizations f, g of an m length curve π and an n length curve σ, respectively, the *width* between f and g is defined as

$width_{f,g}(\pi,\sigma) = \max_{\alpha\in[0,1]} ||\pi(f(\alpha)) - \sigma(g(\alpha))||$. The (standard, i.e. continuous and strong) *Fréchet distance* between π and σ is then $\mathsf{d}_{\mathcal{F}}(\pi,\sigma) = \inf_{f,g} width_{f,g}(\pi,\sigma)$ where f, g range over all possible reparameterizations of π and σ.

The *discrete Fréchet distance* is similar to the above defined Fréchet distance, except that we do not traverse the edges but rather discontinuously jump to adjacent vertices. Specifically, define a monotone correspondence as a sequence of index pairs $\langle(i_1, j_1), \ldots, (i_k, j_k)\rangle$ such that $(i_1, j_1) = (1, 1)$, $(i_k, j_k) = (m, n)$, for any $1 \le z \le k$ we have $1 \le i_z \le m$ and $1 \le j_z \le n$, and for any $1 \le z < k$ we have $(i_{z+1}, j_{z+1}) \in \{(i_z + 1, j_z), (i_z, j_z + 1), (i_z + 1, j_z + 1)\}$. Let C denote the set of all monotone correspondences, then the discrete Fréchet distance is $\mathsf{d}_{\mathcal{DF}}(\pi,\sigma) = \inf_{c\in C} \max_{(i,j)\in c} ||\pi_i - \sigma_j||$.

Both the Fréchet distance and the discrete Fréchet distance have a corresponding *weak* variant, which is defined analogously except that one is allowed to backtrack on the curves. Specifically, the *weak Fréchet distance*, denoted $\mathsf{d}^{\mathrm{w}}_{\mathcal{F}}(\pi,\sigma)$, is defined similarly to the standard Fréchet distance above, except that when defining the width f and g are no longer required to be non-decreasing bijections, but are still required to be continuous and have $f(0) = 1, g(0) = 1$ and $f(1) = m, g(1) = n$. Similarly, the *weak discrete Fréchet distance*, denoted $\mathsf{d}^{\mathrm{w}}_{\mathcal{DF}}(\pi,\sigma)$, is defined similarly to the discrete Fréchet distance above, except that we no longer require the correspondence to be monotone. Specifically, a (non-monotone) correspondence is a sequence of index pairs $\langle(i_1, j_1), \ldots, (i_k, j_k)\rangle$ such that $(i_1, j_1) = (1, 1)$, $(i_k, j_k) = (m, n)$, for any $1 \le z \le k$ we have $1 \le i_z \le m$ and $1 \le j_z \le n$, and for any $1 \le z < k$ we have $(i_{z+1}, j_{z+1}) \in \{(i_z \pm 1, j_z), (i_z, j_z \pm 1), (i_z \pm 1, j_z \pm 1)\}$.

Free Space. To compute the Fréchet distance one normally looks at the so called *free space*. For the continuous case, the δ free space between curves π and σ, of sizes m and n respectively, is defined as

$$F_\delta = \{(\alpha, \beta) \in [1, m] \times [1, n] \ \mid\ ||\pi(\alpha) - \sigma(\beta)|| \le \delta\}.$$

Alt and Godau [3] observed that any x, y monotone path in the δ free space from $(1, 1)$ to (m, n) corresponds to a pair of reparameterizations f, g of π, σ such that $width_{f,g}(\pi,\sigma) \le \delta$. The converse also holds and hence $d_{\mathcal{F}}(\pi,\sigma) \le \delta$ if and only if such a monotone path exists. Thus in order to determine if $d_{\mathcal{F}}(\pi,\sigma) \le \delta$, we define the reachable free space,

$$R_\delta = \{(\alpha, \beta) \in F_\delta \ \mid\ \text{there exists an } x, y \text{ monotone path from } (1, 1) \text{ to } (\alpha, \beta)\}.$$

Hence $d_{\mathcal{F}}(\pi,\sigma) \le \delta$ if and only if $(m, n) \in R_\delta$.

For the case of the discrete Fréchet distance, the free space can still be considered, and is simply described by an $m \times n$ grid graph. Specifically, the vertices are all pairs (i, j) such that $1 \le i \le m$ and $1 \le j \le n$, and for any vertex (i, j) we create the directed edges $(i, j) \to (i + 1, j)$, $(i, j) \to (i, j + 1)$, and $(i, j) \to (i + 1, j + 1)$ (whenever the corresponding destination vertex exists). A vertex (i, j) is then called *free* if $||\pi_i - \sigma_j|| \le \delta$. Analogous to the continuous case, we then have that $\mathsf{d}_{\mathcal{DF}}(\pi,\sigma) \le \delta$ if and only if there is a path in this directed graph from $(1, 1)$ to (m, n) which only uses free vertices.

For the weak discrete Fréchet distance the free space is described by the undirected graph on the same set of vertices, where vertex (i, j) and vertex (i', j') are adjacent if and only if $|i - i'| \le 1$ and $|j - j'| \le 1$. Again, $\mathsf{d}^{\mathrm{w}}_{\mathcal{DF}}(\pi,\sigma) \le \delta$ if and only if there is a path in this undirected graph from $(1, 1)$ to (m, n) which only uses free vertices. Analogously, the free space for the weak continuous Fréchet distance is the same as that for the strong continuous Fréchet distance, but now we no longer require the path through the free space be monotone.

Fréchet Edit Distance. Given a curve $\sigma = \langle \sigma_1, \dots, \sigma_n \rangle$, a *deletion* of the vertex σ_i produces the $n-1$ vertex curve $\sigma' = \langle \sigma_1, \dots, \sigma_{i-1}, \sigma_{i+1}, \dots, \sigma_n \rangle$. Conversely the *insertion* of a vertex p into σ at position i produces the $n+1$ vertex curve $\sigma' = \langle \sigma_1, \dots, \sigma_{i-1}, \mathsf{p}, \sigma_i, \dots, \sigma_n \rangle$. Both deletions and insertions are referred to as *edits*. Let $\delta > 0$ be a fixed threshold distance. Then given polygonal curves $\pi = \langle \pi_1, \dots, \pi_m \rangle$ and $\sigma = \langle \sigma_1, \dots, \sigma_n \rangle$ define the δ-threshold *Fréchet edit distance* from σ to π as the minimum number of edits to σ, producing a new curve σ', such that $\mathsf{d}_{\mathcal{F}}(\pi, \sigma') \leq \delta$. As $\delta > 0$ is some fixed value, and the term "Fréchet" is implicit, throughout we refer to this more simply as the *edit distance* from σ to π, and we denote it as $\mathsf{ed}_{\mathcal{F}}(\pi, \sigma)$. We analogously define the weak edit distance, denoted $\mathsf{ed}^{\mathrm{w}}_{\mathcal{F}}(\pi, \sigma)$, the discrete edit distance, denoted $\mathsf{ed}_{\mathcal{DF}}(\pi, \sigma)$, and the weak discrete edit distance, denoted $\mathsf{ed}^{\mathrm{w}}_{\mathcal{DF}}(\pi, \sigma)$, by replacing the condition $\mathsf{d}_{\mathcal{F}}(\pi, \sigma') \leq \delta$ with $\mathsf{d}^{\mathrm{w}}_{\mathcal{F}}(\pi, \sigma') \leq \delta$, $\mathsf{d}_{\mathcal{DF}}(\pi, \sigma') \leq \delta$, and $\mathsf{d}^{\mathrm{w}}_{\mathcal{DF}}(\pi, \sigma') \leq \delta$, respectively.

For any one of these variants we may consider the case when only deletions or only insertions are allowed. In this case we prepend D for deletion only, or I for insertion only. (For example, $\mathsf{ed}_{\mathcal{F}}(\pi, \sigma)$ becomes $\mathsf{Ded}_{\mathcal{F}}(\pi, \sigma)$ or $\mathsf{Ied}_{\mathcal{F}}(\pi, \sigma)$.) Note that by considering only deletions or only insertions, there may be no valid edit sequence, in which case we define the edit distance as ∞. Conversely, if we allow both deletions and insertions, there is always a solution of cost $m + n$ by deleting all vertices of σ and inserting all vertices of π.

3 DAG Complexes

[19] define the following generalization of a curve. Consider a directed acyclic graph (DAG) with vertices in $\mathbb{R}^d$, where a directed edge $\mathsf{p} \to \mathsf{q}$ is realized by the directed segment pq. We refer to such an embedded graph as being a *DAG complex*, denoted $\mathcal{C}$, with embedded vertices $V(\mathcal{C})$ (i.e. points) and embedded edges $E(\mathcal{C})$ (i.e. line segments). We assume the underlying graph is weakly connected and thus write $|\mathcal{C}| = |E(\mathcal{C})|$. Note also that a DAG complex is allowed to have crossing edges or overlapping vertices (i.e. it is not necessarily an embedding in $\mathbb{R}^d$). Call a polygonal curve $\pi = \langle \pi_1, \dots, \pi_k \rangle$ *compliant* with $\mathcal{C}$ if $\pi_i \in V(\mathcal{C})$ for all i and $\pi_i \pi_{i+1} \in E(\mathcal{C})$ for all $1 \leq i < k$. (Note this implies π traverses each edge in the direction compliant with its orientation from the DAG.) [19] considered the following.

▶ **Problem 1.** *Given two DAG complexes $\mathcal{C}_1$ and $\mathcal{C}_2$, start vertices $s_1 \in V(\mathcal{C}_1), s_2 \in V(\mathcal{C}_2)$, end vertices $t_1 \in V(\mathcal{C}_1), t_2 \in V(\mathcal{C}_2)$, and a value δ, determine if there exists two polygonal curves π_1, π_2, such that:*
(a) *π_i is compliant with $\mathcal{C}_i$ for $i = 1, 2$.*
(b) *π_i starts at s_i and ends at t_i in $\mathcal{C}_i$, for $i = 1, 2$.*
(c) *$\mathsf{d}_{\mathcal{F}}(\pi_1, \pi_2) \leq \delta$.*

[19] solve Problem 1 in $O(|\mathcal{C}_1||\mathcal{C}_2|)$ time by considering the free space of the product complex of $\mathcal{C}_1$ and $\mathcal{C}_2$. This is analogous to the standard procedure used for the Fréchet distance between curves. In the full version, we describe this standard procedure for curves and then how the proceedure extends to this more general product complex. This in turn allows us to remark how the procedure from [19] can easily be extended to the more general setting where we allow multiple starting and ending points, resulting in the following theorem.

▶ **Theorem 2.** *Given two DAG complexes $\mathcal{C}_1$ and $\mathcal{C}_2$, starting vertices $S_1 \subseteq V(\mathcal{C}_1)$ and $S_2 \subseteq V(\mathcal{C}_2)$, target vertices $T_1 \subseteq V(\mathcal{C}_1)$ and $T_2 \subseteq V(\mathcal{C}_2)$, and a value δ, then in $O(|\mathcal{C}_1||\mathcal{C}_2|)$ time one can determine the set of all pairs $t_1 \in T_1$ and $t_2 \in T_2$, such that there are curves π_1 and π_2 such that*
(a) *π_i is compliant with $\mathcal{C}_i$ for $i = 1, 2$.*
(b) *π_i starts at some $s_i \in S_i$ and ends at t_i, for $i = 1, 2$.*
(c) *$\mathsf{d}_{\mathcal{F}}(\pi_1, \pi_2) \leq \delta$.*

4 Continuous Fréchet Distance

We give algorithms to compute $\mathsf{Ded}_{\mathcal{F}}(\pi,\sigma)$, $\mathsf{Ied}_{\mathcal{F}}(\pi,\sigma)$, and $\mathsf{ed}_{\mathcal{F}}(\pi,\sigma)$. The high level approach in each case is to convert π and σ into DAG complexes and apply Theorem 2.

Recall that in the Fréchet edit distance problems, we are only editing σ, not π. As remarked above, π is itself a DAG complex, and using this complex directly represents that π is not modified. Thus in the following the task is to model edits to σ with an appropriate DAG complex. (For $\mathsf{Ded}_{\mathcal{F}}(\pi,\sigma)$ we will remark that creating such DAG complexes for both π and σ allows modelling the problem where deletion is allowed on either curve.)

4.1 Deletion Only

Given a curve $\sigma = \langle \sigma_1,\ldots,\sigma_n\rangle$, consider the DAG complex produced by adding all possible forward edges to σ, namely all directed edges $\sigma_i\sigma_j$ for all $1 \leq i < j \leq n$. We will refer to this as the *complete DAG complex* induced by σ. Observe that any curve that is compliant with the complete DAG complex is defined precisely by the subsequence of vertices from σ it contains. Thus the set of curves that are compliant with the complete DAG complex is in one to one correspondence with the set of subsequences of σ. Conversely, any curve obtained by deleting a subset of vertices from σ, is defined by the subsequence of σ that remains. Thus one concludes that the set of all curves that are compliant with the complete DAG complex of σ are in one to one correspondence with the set of curves obtainable from σ by deletions.

The above tells us that the complete DAG complex encodes all possible curves produced by deletion, however, it needs to be further modified to also encode the cost of these deletions. To account for this cost we make k additional copies of σ, where k is some bound on the number of allowed deletions (which may be as large as n). Intuitively, the copy number of a given vertex encodes the number of deletions made so far. So let $\sigma^\ell = \langle \sigma^\ell_1,\ldots,\sigma^\ell_n\rangle$ denote the ℓth copy. Then to construct the DAG complex, for all $0 \leq \ell \leq k$ and all $i < j$ such that $\ell + (j-(i+1)) \leq k$, we add the directed edge $\sigma^\ell_i\sigma^{\ell+(j-(i+1))}_j$. Such edges are added since if we wish to delete all vertices between σ_i and σ_j (and hence use the edge $\sigma_i\sigma_j$) then we pay for these $(j-(i+1))$ deletions by advancing from the copy ℓ to copy $\ell+(j-(i+1))$ of σ. Call the resulting complex the *complete weighted DAG complex* of σ.

Now given $\pi = \langle \pi_1,\ldots,\pi_m\rangle$ and $\sigma = \langle \sigma_1,\ldots,\sigma_n\rangle$, our goal is to decide if $\mathsf{Ded}_{\mathcal{F}}(\pi,\sigma) \leq k$. As discussed above, the directed edges of π immediately define a DAG complex, and thus we refer to this complex simply as π. On the other hand, for σ we construct the complete weighted DAG complex for σ, denoted $\mathcal{C}_\sigma$. Now for π we must start at π_1 and end at π_m, however, for σ the optimal solution may delete some prefix of vertices $\sigma_1,\ldots,\sigma_i$, which would correspond to starting at vertex σ^i_{i+1} in $\mathcal{C}_\sigma$. Thus the set S_σ of starting vertices consists of all vertices σ^i_{i+1}. Similarly, the optimal solution may delete some suffix of vertices from σ. To handle this case, however, we simply consider all possible ending vertices, namely $T_\sigma = V(\mathcal{C}_\sigma)$. Then we call Theorem 2, which in $O(k^2mn)$ time (since $|\pi| = O(m)$ and $|\mathcal{C}_\sigma| = O(k^2n)$) computes the set of all pairs in $\pi_m \times V(\mathcal{C}_\sigma)$ such that there are compliant paths from allowable starting vertices whose Fréchet distance is $\leq \delta$. If no such pair exists then $\mathsf{Ded}_{\mathcal{F}}(\pi,\sigma) > k$. Otherwise, let (π_m,σ^α_i) be one of the computed ending pairs. Then reaching this pair corresponds to deleting α vertices before σ_i, plus deleting all $n-i$ vertices after σ_i. Thus for each such pair (π_m,σ^α_i) we check if $\alpha + (n-i) \leq k$, and if this holds for some pair then $\mathsf{Ded}_{\mathcal{F}}(\pi,\sigma) \leq k$, and otherwise $\mathsf{Ded}_{\mathcal{F}}(\pi,\sigma) > k$.

Before stating our summarizing theorem, we observe several easy extensions. First, if deletions are allowed on both curves, then the same procedure works where instead of using π as one of the DAG complexes, we use the complete weighted DAG complex $\mathcal{C}_\pi$, yielding

$O(k^4mn)$ time in total. Alternatively, again only allow deletions on σ, but consider the problem of computing $\mathsf{Ded}_{\mathcal{F}}(\pi, \sigma)$, rather than determine if $\mathsf{Ded}_{\mathcal{F}}(\pi, \sigma) \leq k$ for some k. In this case, the same procedure works by setting $k = n$ (as one cannot delete more vertices than the curve contains), and then finding the pair (π_m, σ_i^α) of allowable end vertices minimizing $\alpha + (n - i)$, resulting in an $O(mn^3)$ running time. Finally, applying this same idea to computing $\mathsf{Ded}_{\mathcal{F}}(\pi, \sigma)$ when deletions are allowed on both curves gives an $O(m^3n^3)$ time, as there can be at most n deletions on σ and at most m on π.

▶ **Theorem 3.** *Given curves $\pi = \langle \pi_1, \ldots, \pi_m \rangle$ and $\sigma = \langle \sigma_1, \ldots, \sigma_n \rangle$, a threshold δ, and an integer parameter $k > 0$, in $O(k^2mn)$ time one can determine if $\mathsf{Ded}_{\mathcal{F}}(\pi, \sigma) \leq k$.*

If deletions are allowed on both π and σ, then in $O(k^4mn)$ time one can determine if $\mathsf{Ded}_{\mathcal{F}}(\pi, \sigma) \leq k$. Finally, one can compute $\mathsf{Ded}_{\mathcal{F}}(\pi, \sigma)$ in $O(mn^3)$ time if deletions are only allowed on σ, and in $O(m^3n^3)$ time if deletions are allowed on both curves.

A slight variation of the above algorithm can be used to solve the vertex restricted shortcut Fréchet distance problem as described in [14, 8], improving the result in [8] for $\mathbb{R}^2$ by a $\log n$ factor while also extending it to $\mathbb{R}^d$. Details appear in the full version.

▶ **Corollary 4.** *Given a threshold δ, a fixed curve $\pi = \langle \pi_1, \ldots, \pi_m \rangle$, and a curve $\sigma = \langle \sigma_1, \ldots, \sigma_n \rangle$ which allows shortcuts, then in $O(mn^2)$ time one can determine if the vertex restricted shortcut Fréchet distance is $\leq \delta$.*

4.2 Insertions

Applying the above approach for insertions only or both deletions and insertions is considerably more difficult. We sketch the main argument here, and give the full details in the full version. For this section, we assume both π and σ are in $\mathbb{R}^2$, since we use the results in $\mathbb{R}^2$ from [18]. For simplicity, we will first assume there are no insertions before σ_1 nor after σ_n.

Observe that if it is beneficial to insert a subcurve between two consecutive vertices of σ, then this subcurve should be a minimum vertex curve with Fréchet distance δ to some portion of π. Unfortunately, the portion of π that we are matching to may not begin and end on vertices of π. Regardless, it suffices to consider a bounded number of canonical starting and ending location pairs.

▶ **Definition 5.** *Given a curve $\pi = \langle \pi_1, \ldots, \pi_m \rangle$, a value δ, and points s and t such that $||s - \pi_1|| \leq \delta$ and $||t - \pi_m|| \leq \delta$, let $\mathsf{mv}_\delta(s, t, \pi)$ denote the curve $\sigma = \langle \sigma_1, \ldots, \sigma_n \rangle$ with the minimum number of vertices such that $\mathsf{d}_{\mathcal{F}}(\pi, s \circ \sigma \circ t) \leq \delta$.*

For an ordered segment q_1q_2 and a point p such that $B(p, \delta) \cap q_1q_2 \neq \emptyset$, let $\mathsf{enter}_\delta(p, q_1q_2)$ denote the point in $B(p, \delta) \cap q_1q_2$ closest to q_1, and similarly let $\mathsf{leave}_\delta(p, q_1q_2)$ denote the point in $B(p, \delta) \cap q_1q_2$ closest to q_2. Finally, given a curve $\pi = \langle \pi_1, \ldots, \pi_m \rangle$ where $m > 2$, and points s and t such that $||s - \pi_1\pi_2|| \leq \delta$ and $||t - \pi_{m-1}\pi_m|| \leq \delta$, define $\mathsf{clip}_\delta(s, t, \pi) = \langle \mathsf{leave}_\delta(s, \pi_1\pi_2), \pi_2, \ldots, \pi_{m-1}, \mathsf{enter}_\delta(t, \pi_{m-1}\pi_m) \rangle$

Let $\mathsf{mv}_\delta(\pi)$ be the analogue of $\mathsf{mv}_\delta(s, t, \pi)$ from Definition 5, except where we require $\mathsf{d}_{\mathcal{F}}(\pi, \sigma) \leq \delta$ instead of $\mathsf{d}_{\mathcal{F}}(\pi, s \circ \sigma \circ t) \leq \delta$, i.e. the starting points s and t are not specified. [18] compute $\mathsf{mv}_\delta(\pi)$ in their Theorem 14. The full version reduces $\mathsf{mv}_\delta(s, t, \pi)$ to $\mathsf{mv}_\delta(\pi)$.

▶ **Theorem 6** ([18])**.** *Given $\pi = \langle \pi_1, \ldots, \pi_m \rangle$, a value δ, and points s and t such that $||s - \pi_1|| \leq \delta$ and $||t - \pi_m|| \leq \delta$, then $\mathsf{mv}_\delta(s, t, \pi)$ can be computed in $O(m^2 \log^2 m)$ time.*

For insertions only, instead of adding directed edges between pairs of vertices of copies of σ, we compute and insert copies of the following $O(nm^2)$ *canonical subcurves.*

$$\mathsf{CS}(\pi,\sigma) = \left\{ \mathsf{mv}_\delta(\sigma_i,\sigma_{i+1},\mathsf{clip}_\delta(\sigma_i,\sigma_{i+1},\pi[\alpha,\beta])) \;\middle|\; \begin{array}{l} i < n,\ \alpha < \beta - 1 \le m-1, \\ i,\alpha,\beta \in \mathbb{Z}^+,\ ||\sigma_i - \pi_\alpha\pi_{\alpha+1}|| \le \delta, \\ ||\sigma_{i+1} - \pi_{\beta-1}\pi_\beta|| \le \delta \end{array} \right\}$$

Computing $\mathsf{CS}(\pi,\sigma)$ takes $O(nm^4\log^2 m)$ time. The complex for π has size $O(m)$, and the insertion weighted complex constructed for σ has size $O(k^2nm^2)$. Thus the total time to construct the complexes and find nearby curves within them is $O(nm^3(k^2+m\log^2 m))$. We can argue that $\mathsf{Ied}_{\mathcal{F}}(\pi,\sigma) = O(m)$. Therefore, $O(nm^3(k^2+m\log^2 m)) = O(nm^5)$. Deletions and insertions together are handled similarly by extending $\mathsf{CS}(\pi,\sigma)$ to be defined over all pairs on σ. See the full version for details.

▶ **Theorem 7.** *Given curves $\pi = \langle\pi_1,\ldots,\pi_m\rangle$ and $\sigma = \langle\sigma_1,\ldots,\sigma_n\rangle$ in $\mathbb{R}^2$, a threshold δ, and an integer $k > 0$, in $O(nm^3(k^2+m\log^2 m))$ time one can determine if $\mathsf{Ied}_{\mathcal{F}}(\pi,\sigma) \le k$. Moreover, one can compute $\mathsf{Ied}_{\mathcal{F}}(\pi,\sigma)$ in $O(nm^5)$ time.*

▶ **Theorem 8.** *Given curves $\pi = \langle\pi_1,\ldots,\pi_m\rangle$ and $\sigma = \langle\sigma_1,\ldots,\sigma_n\rangle$ in $\mathbb{R}^2$, a threshold δ, and an integer $k > 0$, in $O(knm^3(k^2+m\log^2 m))$ time one can determine if $\mathsf{ed}_{\mathcal{F}}(\pi,\sigma) \le k$. Moreover, one can compute $\mathsf{ed}_{\mathcal{F}}(\pi,\sigma)$ in $O((m+n)^3nm^3)$ time.*

5 Discrete Fréchet Distance

We now discuss the discrete analogs $\mathsf{Ded}_{\mathcal{DF}}(\pi,\sigma)$, $\mathsf{Ied}_{\mathcal{DF}}(\pi,\sigma)$, and $\mathsf{ed}_{\mathcal{DF}}(\pi,\sigma)$ of the problems in the previous section. The extra structure afforded by considering discrete point sequences allows us to more directly apply standard dynamic programming techniques and achieve faster running times for all three problems and in any constant dimension.

5.1 Deletion Only

The deletion only variant $\mathsf{Ded}_{\mathcal{DF}}(\pi,\sigma)$ serves as an easy warm up. Let $\mathsf{DedDP}(i,j) := \mathsf{Ded}_{\mathcal{DF}}(\pi[1,i],\sigma[1,j])$ (with $i = 0$ and $j = 0$ denoting empty prefixes, and $\mathsf{DedDP}(0,0) = 0$). Suppose there is a set of deletions changing $\sigma[1,j]$ into a curve σ' such that $\mathsf{d}_{\mathcal{DF}}(\pi[1,i],\sigma') \le \delta$.

If $i \ge 1$, then we must have $j \ge 1$ as well. Suppose further that $||\sigma_j - \pi_i|| \le \delta$. Now, any monotone correspondence between $\pi[1,i]$ and σ' already includes or can be extended to include the pair (π_i,σ_j) without increasing the maximum distance of a pair beyond δ. Therefore, we may assume σ' ends with σ_j. As in the normal dynamic programming solution for the discrete Fréchet distance, we may further assume the rest of the correspondence matches all of curves $\pi[1,i]$ and σ' except for the last point of one or both of them.

If $i = 0$ and $j \ge 1$ then clearly σ_j must be deleted as there is no vertex of π to match it to. Similarly, if $i,j \ge 1$ and $||\sigma_j - \pi_i|| > \delta$, then again σ_j must be deleted as all monotone correspondences between $\pi[1,i]$ and σ' end with a pair containing the last point of both.

From the above discussion, we conclude

$$\mathsf{DedDP}(i,j) = \begin{cases} 0 & \text{if } i = 0 \text{ and } j = 0 \\ \infty & \text{if } i \geq 1 \text{ and } j = 0 \\ 1 + \mathsf{DedDP}(i, j-1) & \text{if } (i = 0 \text{ and } j \geq 1) \\ & \quad \text{or } (i, j \geq 1 \text{ and } ||\sigma_j - \pi_i|| > \delta) \\ \min \begin{Bmatrix} \mathsf{DedDP}(i, j-1)\,, \\ \mathsf{DedDP}(i-1, j)\,, \\ \mathsf{DedDP}(i-1, j-1) \end{Bmatrix} & \text{otherwise} \end{cases} .$$

$\mathsf{Ded}_{\mathcal{DF}}(\pi, \sigma) = \mathsf{DedDP}(m, n)$ can be computed easily in $O(mn)$ time using this recurrence.

▶ **Theorem 9.** *Given curves $\pi = \langle \pi_1, \ldots, \pi_m \rangle$ and $\sigma = \langle \sigma_1, \ldots, \sigma_n \rangle$ in $\mathbb{R}^d$ and a threshold δ, one can compute $\mathsf{Ded}_{\mathcal{DF}}(\pi, \sigma)$ in $O(mn)$ time.*

5.2 Insertions

We now consider the insertion only case $\mathsf{Ied}_{\mathcal{DF}}(\pi, \sigma)$. Let $\mathsf{IedDP}(i, j) := \mathsf{Ied}_{\mathcal{DF}}(\pi[1, i], \sigma[1, j])$. As before, assume there is a set of insertions changing $\sigma[1, j]$ to σ' where $\mathsf{d}_{\mathcal{DF}}(\pi[1, i], \sigma') \leq \delta$.

Suppose σ' ends with σ_j, implying $||\sigma_j - \pi_i|| \leq \delta$. (It is important to note for later that if $||\sigma_j - \pi_i|| \leq \delta$ it does not imply σ' ends with σ_j.) We get the three standard cases for computing the discrete Fréchet distance as before.

Now suppose σ' does not end with σ_j and instead ends with a newly inserted point. Let x denote this final point of σ'. There exists some $k \in \{1, \ldots, i\}$ such that the monotone correspondence with maximum distance at most δ between σ' and $\pi[1, i]$ ends with pairs between points of $\langle \pi_k, \ldots, \pi_i \rangle$ and x. These points of $\langle \pi_k, \ldots, \pi_i \rangle$ all live in $B(x, \delta)$, the ball of radius δ centered at x. Accordingly, let $\mu(i)$ denote the smallest $t \in \{1, \ldots, i\}$ such that the radius of the minimum enclosing ball of $\langle \pi_t, \ldots, \pi_i \rangle$ is at most δ. We may assume x is the center of the ball defining $\mu(i)$ and that $\mu(i) \leq k \leq i$. We have the following recurrence.

$$\mathsf{IedDP}(i,j) = \begin{cases} 0 & \text{if } i = 0 \text{ and } j = 0 \\ \infty & \text{if } i = 0 \text{ and } j \geq 1 \\ 1 + \min_{\mu(i) \leq k \leq i} \mathsf{IedDP}(k-1, j) & \text{if } (i \geq 1 \text{ and } j = 0) \\ & \quad \text{or } (i, j \geq 1 \text{ and } ||\sigma_j - \pi_i|| > \delta) \\ \min \begin{Bmatrix} \mathsf{IedDP}(i, j-1)\,, \\ \mathsf{IedDP}(i-1, j)\,, \\ \mathsf{IedDP}(i-1, j-1)\,, \\ 1 + \min\limits_{\mu(i) \leq k \leq i} \mathsf{IedDP}(k-1, j) \end{Bmatrix} & \text{otherwise} \end{cases} .$$

After $O(m^2)$ preprocessing time and through careful use of simple data structures, we are able to solve all the relevant subproblems in $O(mn)$ time. A similar recurrence and dynamic programming strategy works for $\mathsf{ed}_{\mathcal{DF}}(\pi, \sigma)$. See the full version for details.

▶ **Theorem 10.** *Given curves $\pi = \langle \pi_1, \ldots, \pi_m \rangle$ and $\sigma = \langle \sigma_1, \ldots, \sigma_n \rangle$ in $\mathbb{R}^d$ for constant d and a threshold δ, one can compute $\mathsf{Ied}_{\mathcal{DF}}(\pi, \sigma)$ in $O(m^2 + mn)$ time.*

▶ **Theorem 11.** *Given curves $\pi = \langle \pi_1, \ldots, \pi_m \rangle$ and $\sigma = \langle \sigma_1, \ldots, \sigma_n \rangle$ in $\mathbb{R}^d$ for constant d and a threshold δ, one can compute $\mathsf{ed}_{\mathcal{DF}}(\pi, \sigma)$ in $O(m^2 + mn)$ time.*

6 Hardness

In this section we prove that a number of variants of the weak edit Fréchet distance are NP-hard. For these variants we will first focus on the discrete Fréchet distance case, showing NP-hardness even when the curves are restricted to points in $\mathbb{R}^1$. Afterwards we show how the NP-hardness proofs easily extend to the continuous case for curves in $\mathbb{R}^2$. All the NP-hardness proofs will be by a reduction from 3SAT, inspired by the reduction in [7].

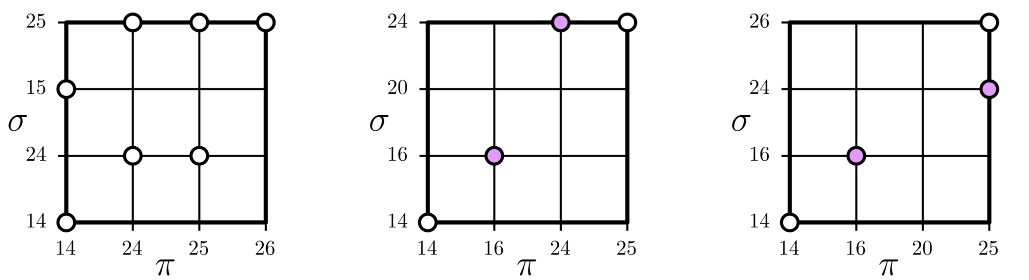
For this section, let π and σ be polygonal curves in $\mathbb{R}^1$ unless otherwise stated, and let $\delta = 1$ be the given threshold with no loss to generality. Since π and σ are curves in $\mathbb{R}^1$, we directly label column i (resp. row j) of the free space with π_i (resp. σ_j). When modifications are restricted to one curve, they will be on σ, which then becomes σ'. We also define an arbitrary 3SAT instance as I, with c clauses and v variables.

6.1 Abstract Framework

Paths and Gaps

Recall from Section 2 that for $\mathsf{d}^{\mathrm{w}}_{\mathcal{DF}}(\pi,\sigma)$ the free space is an $m \times n$ grid graph, where vertex (i,j) and vertex (i',j') are adjacent if and only if $|i-i'| \leq 1$ and $|j-j'| \leq 1$. Then determining if $\mathsf{d}^{\mathrm{w}}_{\mathcal{DF}}(\pi,\sigma) \leq 1$ is equivalent to determining if a path exists from $(1,1)$ to (m,n) in the free space graph which only uses free vertices, namely vertices (i,j) such that $|\pi_i - \sigma_j| \leq 1$. See Figure 1a, for an example when such a path exists.

Consider the highlighted pair of free vertices in Figure 1b. While their horizontal distance is 1, their vertical distance is 2, which we will refer to as a vertical *gap* as it prevents a path through these vertices. Observe, however, that a deletion of the third vertex from σ (i.e. the third row) removes this gap, creating a path from the lower left corner to the upper right corner, and thus $\mathsf{Ded}^{\mathrm{w}}_{\mathcal{DF}}(\pi,\sigma) = 1$. Conversely, observe that if we were only allowed insertions on σ, then there is no way to bridge this vertical gap. Now consider the highlighted pair of free vertices in Figure 1c, where now instead there is a horizontal gap. If we are only allowed deletions on σ then there is no way to bridge this gap (though deletions on π would bridge the gap). However, if we allow insertions on σ, then inserting a value of 20 at the third row would create a path between these two vertices, showing that $\mathsf{Ied}^{\mathrm{w}}_{\mathcal{DF}}(\pi,\sigma) = 1$. Thus in summary, deletion could be used to bridge a vertical gap but not a horizontal one, and insertion could be used to bridge a horizontal gap but not a vertical one.



(a) The Weak Fréchet distance is already 1.

(b) Deletion can close the vertical gap.

(c) Insertion can close the horizontal gap.

Figure 1 Three free spaces diagrams with free spaces represented by circles, and edits permitted on σ only. The values along the axes are the curve coordinates in $\mathbb{R}^1$.

Consider the example in Figure 2, where there are two vertical gaps, and suppose we are considering the deletion only problem. Now the first vertical gap can be removed by deleting $\sigma_3 = 16$. However, doing this creates a horizontal gap at $\pi_8 = 16$, where the other vertical gap was, and this horizontal gap cannot be bridged by deletion(s). Similarly, if we start by trying to close the second vertical gap with deletion of row $\sigma_2 = 14$ then we get an insurmountable horizontal gap at $\pi_2 = 14$. We thus refer to such a pair of vertical gaps as *opposing*. Ultimately, our goal is to use the decision to create a path by bridging a gap as deciding to set a literal in the given 3SAT instance to True. Intuitively, by creating such opposing gaps we can make setting a literal to True correspond to setting instances of the negated literal to False.

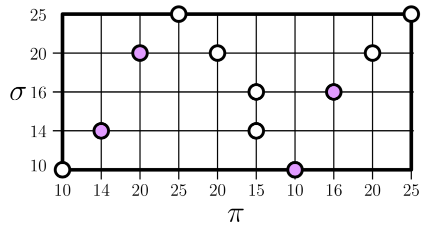


Figure 2 Opposing vertical gaps, where bridging one with a row deletion creates a horizontal gap at the other.

Reduction Framework

We now describe the abstract structure, shown in Figure 3, which we use to represent any 3SAT instance as an instance of weak discrete edit Fréchet distance. First, we make a rectangular free space gadget for each clause, which are then placed in series. Within a given clause gadget, the rows can intuitively be partitioned into three layers, and each layer can be partitioned into three sections of columns. Thus overall the clause gadget consists of 9 logical (roughly square) regions, where, as shown in Figure 3, each region consists of an orange diagonal path of free vertices, which we simply refer to as a *diagonal*. Now for the top and bottom layers, their three diagonals will be unobstructed and connect to each other to allow traversal through these regions. The middle layer will also consist of three diagonals, however, we create gaps on these diagonals to encode the given clause. Namely, the three diagonals will correspond to the three literals of the clause, and choosing to bridge a gap on one of these diagonals will correspond to setting that literal to True.

As mentioned above, the clause gadgets are placed in series. Observe that we enter the first clause gadget at its bottom left corner, and exit at its top right corner. Thus in order to have the second clause gadget start where the first clause gadget ends, we invert the second clause gadget so that it must be traversed from its upper left corner to its lower right corner. In general, the odd clause gadgets must be traversed up and to the right, and the even ones down and to the right. (If there are an even number of clauses, we can insert one more gadget at the end that allows traversing from the lower left to the upper right.) Furthermore, when going from an odd to an even gadget, there will be a column inbetween with only a single free space at the top row (resp. bottom row when going from even to odd), to ensure this is the only point of connection between the gadgets.

Let $L := \langle 15, 25, \ldots 10v + 5\rangle$ be an ordered sequence of values, and let L^R denote L in reverse order. An ascending diagonal path is realized by setting portions of π and σ to both L or both L^R. Similarly, a descending diagonal path is created by setting a portion of π to L (resp. L^R) and a portion of σ to L^R (resp. L).

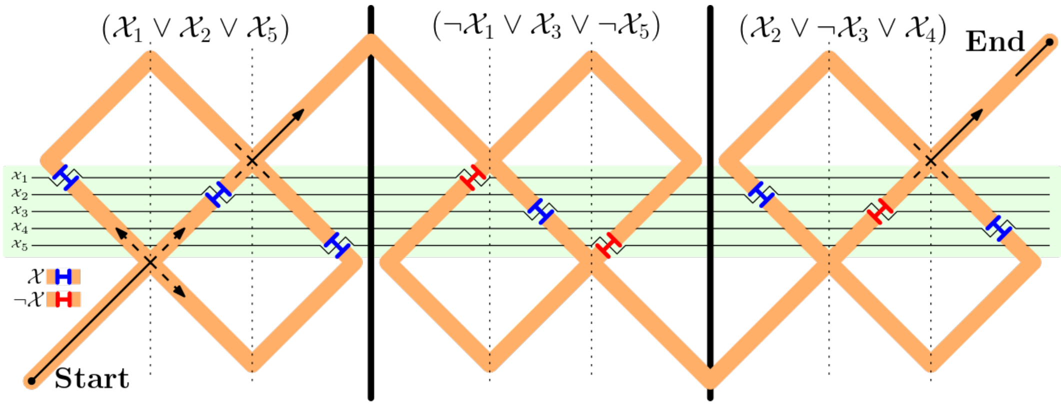


Figure 3 Abstract free space structure. This example is satisfied by setting $\mathcal{X}_2, \neg\mathcal{X}_5 =$ True.

Consider a clause gadget, which consists of 9 diagonals, 3 in each layer, alternating between ascending and descending, creating a zigzag pattern. Within a layer, when two diagonals meet we insert a value in between them such that they are "glued" together by a column which locally has no free vertices except at the one location where the diagonals come together. If the end of one diagonal (correspondingly the beginning of the next one) is the value $10v+5$, then this can be achieved by placing the value $10(v+1)$ between the diagonals, as it is larger than any value in L. Similarly if the diagonal ends at the value 15 then we insert the value 10 before the next diagonal. These inserted values will also similarly act to glue the layers of the clause gadget together. Let π^i denote the portion of π corresponding to the ith clause. Then the *basic clause gadget*, shown in Figure 4, is defined by setting

$$\pi^i = \sigma = \langle 0, 10\rangle \circ L \circ \langle 10(v+1)\rangle \circ L^R \circ \langle 10\rangle \circ L \circ \langle 10(v+1), 10(v+2)\rangle.$$

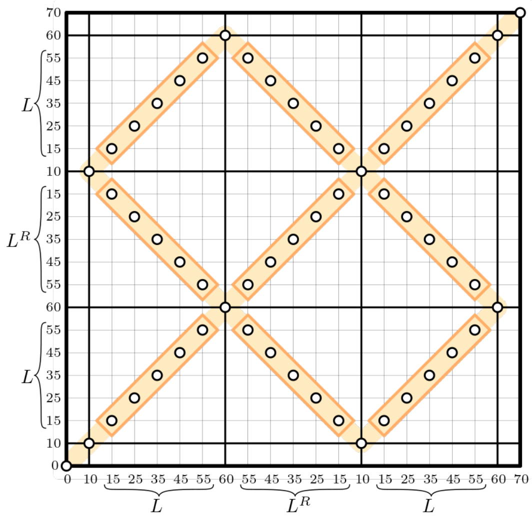


Figure 4 Basic clause gadget, consisting of 9 (highlighted) diagonals made by pairs of L's and L^R's which have been glued together such that the free-space has 3 paths.

Observe that we appended the value 0 at the beginning and $10(v+2)$ at the end of each curve. This serves to glue the successive clause gadgets together at single free vertices, in the same way we glued diagonals within a clause gadget together. Again, the values 0 and

$10(v+2)$ achieve this by respectively being smaller or larger than any value used internally in the clause gadget. Note that the above values used to create the basic clause gadget do not create any gaps in the variable layer. Depending on the edit operation allowed, we modify the construction to create the appropriate gaps. The details for these modifications are given in the full version. We summarize the full suite of results in the following two theorems.

▶ **Theorem 12.** *Given a value δ and curves π and σ in $\mathbb{R}^1$, determining if the weak discrete Fréchet distance between the curves can be made less than or equal to δ by any of the following processes is NP-hard:*

(a) *deleting any number of points from σ;*
(b) *deleting up to k points from π, σ, or both;*
(c) *inserting up to k points into σ; and*
(d) *performing up to k deletions or insertions from/into σ.*

Further, determining if the weak discrete vertex-restricted shortcut Fréchet distance is less than or equal to δ is NP-hard.

▶ **Theorem 13.** *Given a value δ and curves π and σ in $\mathbb{R}^2$, determining if the weak continuous Fréchet distance between the curves can be made less than or equal to δ by any of the following processes is NP-hard:*

(a) *deleting any number of points from σ;*
(b) *deleting up to k points from π, σ, or both;*
(c) *inserting up to k points into σ; and*
(d) *performing up to k deletions or insertions from/into σ.*

Further, determining if the weak continuous vertex-restricted shortcut Fréchet distance is less than or equal to δ is NP-hard.

References

1. Pankaj K Agarwal, Kyle Fox, Jiangwei Pan, and Rex Ying. Approximating dynamic time warping and edit distance for a pair of point sequences. In *Proceedings of the 32nd International Symposium on Computational Geometry*, pages 6:1–6:16, 2016.
2. Mahmuda Ahmed, Sophia Karagiorgou, Dieter Pfoser, and Carola Wenk. *Map Construction Algorithms*. Springer, 2015.
3. Helmut Alt and Michael Godau. Computing the Fréchet distance between two polygonal curves. *Int. J. Comput. Geometry Appl.*, 5:75–91, 1995.
4. Rinat Ben Avraham, Omrit Filtser, Haim Kaplan, Matthew J. Katz, and Micha Sharir. The discrete and semicontinuous Fréchet distance with shortcuts via approximate distance counting and selection. *ACM Trans. Algorithms*, 11(4):29:1–29:29, 2015.
5. Karl Bringmann. Why walking the dog takes time: Fréchet distance has no strongly subquadratic algorithms unless SETH fails. In *Proceedings of the IEEE 55th Annual Symposium on Foundations of Computer Science*, pages 661–670, 2014.
6. Karl Bringmann and Wolfgang Mulzer. Approximability of the discrete Fréchet distance. *JoCG*, 7(2):46–76, 2016.
7. Kevin Buchin, Maarten Löffler, Tim Ophelders, Aleksandr Popov, Jérôme Urhausen, and Kevin Verbeek. Computing the Fréchet distance between uncertain curves in one dimension. *Comput. Geom.*, 109:101923, 2023. `doi:10.1016/j.comgeo.2022.101923`.
8. Maike Buchin, Anne Driemel, and Bettina Speckmann. Computing the Fréchet distance with shortcuts is np-hard. In Siu-Wing Cheng and Olivier Devillers, editors, *30th Annual Symposium on Computational Geometry*, page 367. ACM, 2014. `doi:10.1145/2582112.2582144`.
9. Maike Buchin and Lukas Plätz. The k-outlier fréchet distance, 2022. `arXiv:2202.12824`.

10 Daniel Chen, Anne Driemel, Leonidas J. Guibas, Andy Nguyen, and Carola Wenk. Approximate map matching with respect to the Fréchet distance. In *Proc. 13th Meeting on Algorithm Engineering and Experiments*, pages 75–83, 2011.

11 Lei Chen and Raymond Ng. On the marriage of Lp-norms and edit distance. In *Proceedings of the 30th International Conference on Very Large Databases*, pages 792–803, 2004.

12 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*, pages 491–502, 2005.

13 Jacobus Conradi and Anne Driemel. On Computing the k-Shortcut Fréchet Distance. In *Proc. 49th Intern. Colloquium Automata, Languages, Programming*, pages 46:1–46:20, 2022.

14 Anne Driemel and Sariel Har-Peled. Jaywalking your dog: Computing the Fréchet distance with shortcuts. *SIAM J. Comput.*, 42(5):1830–1866, 2013. doi:10.1137/120865112.

15 Omrit Filtser and Matthew J. Katz. Algorithms for the discrete Fréchet distance under translation. *J. Comput. Geom.*, 11(1):156–175, 2020.

16 Kyle Fox and Xinyi Li. Approximating the geometric edit distance. *Algorithmica*, 84(9):2395–2413, 2022.

17 Omer Gold and Micha Sharir. Dynamic time warping and geometric edit distance: Breaking the quadratic barrier. *ACM Transactions on Algorithms*, 14(4):50, 2018.

18 Leonidas J. Guibas, John Hershberger, Joseph S. B. Mitchell, and Jack Snoeyink. Approximating polygons and subdivisions with minimum link paths. *Int. J. Comput. Geom. Appl.*, 3(4):383–415, 1993. doi:10.1142/S0218195993000257.

19 Sariel Har-Peled and Benjamin Raichel. The Fréchet distance revisited and extended. *ACM Trans. Algorithms*, 10(1):3:1–3:22, 2014. doi:10.1145/2532646.

20 Minghui Jiang, Ying Xu, and Binhai Zhu. Protein structure-structure alignment with discrete Fréchet distance. *J. Bioinformatics and Computational Biology*, 6(1):51–64, 2008.

21 Pierre-François Marteau. Time warp edit distance with stiffness adjustment for time series matching. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 31(2):306–318, 2009.

22 Swaminathan Sankararaman, Pankaj K Agarwal, Thomas Mølhave, Jiangwei Pan, and Arnold P Boedihardjo. Model-driven matching and segmentation of trajectories. In *Proceedings of the 21st ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems*, pages 234–243, 2013.

23 E. Sriraghavendra, Karthik K., and Chiranjib Bhattacharyya. Fréchet distance based approach for searching online handwritten documents. In *Proc. 9th Intern. Conf. Document Analysis and Recognition*, pages 461–465, 2007.

24 Aleksandar Stojmirovic and Yi-kuo Yu. Geometric aspects of biological sequence comparison. *Journal of Computational Biology*, 16(4):579–611, 2009.

25 Xiaoyue Wang, Abdullah Mueen, Hui Ding, Goce Trajcevski, Peter Scheuermann, and Eamonn Keogh. Experimental comparison of representation methods and distance measures for time series data. *Data Mining and Knowledge Discovery*, 26(2):275–309, 2013.