APPROXIMABILITY OF THE DISCRETE FRÉCHET DISTANCE*

Karl Bringmann,† *Wolfgang Mulzer*‡

ABSTRACT. The Fréchet distance is a popular and widespread distance measure for point sequences and for curves. About two years ago, Agarwal *et al.* [SIAM J. Comput. 2014] presented a new (mildly) subquadratic algorithm for the discrete version of the problem. This spawned a flurry of activity that has led to several new algorithms and lower bounds.

In this paper, we study the approximability of the discrete Fréchet distance. Building on a recent result by Bringmann [FOCS 2014], we present a new conditional lower bound showing that strongly subquadratic algorithms for the discrete Fréchet distance are unlikely to exist, even in the *one-dimensional* case and even if the solution may be approximated up to a factor of 1.399.

This raises the question of how well we can approximate the Fréchet distance (of two given d-dimensional point sequences of length n) in strongly subquadratic time. Previously, no general results were known. We present the first such algorithm by analysing the approximation ratio of a simple, linear-time greedy algorithm to be $2^{\Theta(n)}$. Moreover, we design an α-approximation algorithm that runs in time $O(n \log n + n^2/\alpha)$, for any $\alpha \in [1, n]$. Hence, an n^{ε}-approximation of the Fréchet distance can be computed in strongly subquadratic time, for any $\varepsilon > 0$.

1 Introduction

Let P and Q be two polygonal curves with n vertices each. The *Fréchet distance* provides a meaningful way to define a distance between P and Q that overcomes some of the shortcomings of the classic Hausdorff distance [6]. Since its introduction to the computational geometry community by Alt and Godau [6], the concept of Fréchet distance has proven extremely useful and has found numerous applications (see, e.g., [4, 6–10] and the references therein).

The Fréchet distance has two classic variants: *continuous* and *discrete* [6, 12]. In this paper, we focus on the discrete variant. In this case, the Fréchet distance between two sequences P and Q of n points in d dimensions is defined as follows: imagine two frogs traversing the sequences P and Q, respectively. In each time step, a frog can jump to the next vertex along its sequence, or it can stay where it is. The discrete Fréchet distance is the minimal length of a leash required to connect the two frogs while they traverse the two sequences from start to finish, see Figure 1.

*KB supported by an ETH Zurich Postdoctoral Fellowship. WM supported in part by DFG Grants MU 3501/1 and MU 3501/2.

†*Institute of Theoretical Computer Science, ETH Zurich, Switzerland*, karlb@inf.ethz.ch

‡*Institut für Informatik, Freie Universität Berlin, Germany*, mulzer@inf.fu-berlin.de

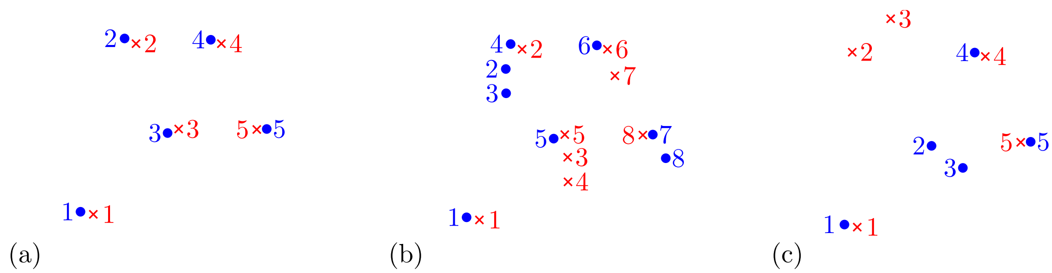

Figure 1: Examples of the discrete Fréchet distance: (a) and (b) show two sequences with small Fréchet distance; (c) shows two sequences with large Fréchet distance.

The original algorithm for the continuous Fréchet distance by Alt and Godau has running time $O(n^2 \log n)$ [6]; while the algorithm for the discrete Fréchet distance by Eiter and Mannila needs time $O(n^2)$ [12]. These algorithms have remained the state of the art until very recently: in 2013, Agarwal *et al.* [4] presented a slightly subquadratic algorithm for the discrete Fréchet distance. Building on their work, Buchin *et al.* [9] managed to find a slightly improved algorithm for the continuous Fréchet distance a year later. At the time, Buchin *et al.* thought that their result provides evidence that computing the Fréchet distance may not be 3SUM-hard [13], as had previously been conjectured by Alt [5]. Even though Grønlund and Pettie [15] showed recently that 3SUM has subquadratic decision trees, casting new doubt on the connection between 3SUM and the Fréchet distance, the conclusions of Buchin et al motivated Bringmann [7] to look for other reasons for the apparent difficulty of the Fréchet distance.

He found an explanation in the *Strong Exponential Time Hypothesis* (SETH) [16,17], which roughly speaking asserts that satisfiability cannot be decided in time[1] $O^*((2-\varepsilon)^n)$ for any $\varepsilon > 0$ (see Section 2 for details). Since exhaustive search takes time $O^*(2^n)$ and since the fastest known algorithms are only slightly faster than that, SETH is a reasonable assumption that formalizes a barrier for our algorithmic techniques. It has been shown that SETH can be used to prove conditional lower bounds even for polynomial time problems [1, 2, 18, 20]. In this line of research, Bringmann [7] showed, among other things, that there are no strongly subquadratic algorithms for the Fréchet distance unless SETH fails. Here, *strongly subquadratic* means any running time of the form $O(n^{2-\varepsilon})$, for constant $\varepsilon > 0$. Bringmann's lower bound works for two-dimensional curves and both classic variants of the Fréchet distance. Thus, it is unlikely that the algorithms by Agarwal *et al.* and Buchin *et al.* can be improved significantly, unless a major algorithmic breakthrough occurs.

1.1 Our Contributions

We focus on the discrete Fréchet distance. Our main results are as follows.

Conditional Lower Bound. We strengthen the result of Bringmann [7] by showing that even in the one-dimensional case computing the Fréchet distance remains hard. More

[1] The notation $O^*(\cdot)$ hides polynomial factors in the number of variables n and the number of clauses m.

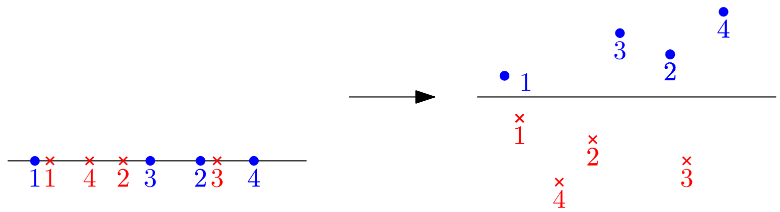

Figure 2: Lifting a one-dimensional discrete Fréchet instance into two dimensions.

precisely, we show that any 1.399-approximation algorithm in strongly subquadratic time for the one-dimensional discrete Fréchet distance violates the Strong Exponential Time Hypothesis. Previously, Bringmann [7] had shown that no strongly subquadratic algorithm approximates the two-dimensional Fréchet distance by a factor of 1.001, unless SETH fails.

One can embed any one-dimensional sequence into the two-dimensional plane by fixing some $\varepsilon > 0$ and by setting the y-coordinate of the i-th point of the sequence to $i \cdot \varepsilon$ (or $-i \cdot \varepsilon$). For sufficiently small ε, this embedding roughly preserves the Fréchet distance, see Figure 2. Thus, unless SETH fails, there is also no strongly subquadratic 1.399-approximation for the discrete Fréchet distance on (1) two-dimensional curves without self-intersections, and (2) two-dimensional x-monotone curves (also called *time-series*). These interesting special cases had been open.

Approximation: Greedy Algorithm. A simple greedy algorithm for the discrete Fréchet distance goes as follows: in every step, make the move that minimizes the current distance, where a "move" is a step in either one sequence or in both of them. This algorithm has a straightforward linear time implementation. We analyze the approximation ratio of the greedy algorithm, and we show that, given two sequences of n points in d dimensions, the maximal distance attained by the greedy algorithm is a $2^{\Theta(n)}$-approximation for their discrete Fréchet distance. We emphasize that this approximation ratio is *bounded*, depending only on n, but not the coordinates of the vertices. This is surprising, since so far no bounded approximation algorithm that runs in strongly subquadratic time was known at all. Moreover, although an approximation ratio of $2^{\Theta(n)}$ is huge, the greedy algorithm is the best *linear time* approximation algorithm that we could come up with. We also show how to extend this algorithm to the continuous case.

Approximation: Improved Algorithm. For the case that slightly more than linear time is acceptable, we provide a much better approximation algorithm: given two sequences P and Q of n points in d dimensions, we show how to find an α-approximation of the discrete Fréchet distance between P and Q in time $O(n \log n + n^2/\alpha)$, for any $1 \leq \alpha \leq n$. In particular, this yields an $n/\log n$-approximation in time $O(n \log n)$, and an n^{ε}-approximation in strongly subquadratic time for any $\varepsilon > 0$. We leave it open whether these approximation ratios can be improved.

2 Preliminaries and Definitions

We begin with some background and basic definitons.

2.1 Discrete Fréchet Distance

Since we focus on the discrete Fréchet distance, we will sometimes omit the term "discrete". Let $P = \langle p_1, \ldots, p_n \rangle$ and $Q = \langle q_1, \ldots, q_n \rangle$ be two sequences of n points in d dimensions. A *traversal* β of P and Q is a sequence of pairs $(p, q) \in P \times Q$ such that (i) the traversal β begins with the pair (p_1, q_1) and ends with the pair (p_n, q_n); and (ii) the pair $(p_i, q_j) \in \beta$ can be followed only by one of (p_{i+1}, q_j), (p_i, q_{j+1}), or (p_{i+1}, q_{j+1}). We call β *parallel* if it only makes steps of the third kind, i.e., if β advances in both P and Q in each step. We define the *distance* of the traversal β as $\delta(\beta) := \max_{(p,q)\in\beta} d(p, q)$, where $d(.,.)$ denotes the Euclidean distance. The *discrete Fréchet distance* of P and Q is now defined as $\delta_{\text{dF}}(P, Q) := \min_\beta \delta(\beta)$, where β ranges over all traversals of P and Q.

We review a simple $O(n^2 \log n)$ time algorithm to compute $\delta_{\text{dF}}(P, Q)$ that is the starting point of our second approximation algorithm. First, we describe a *decision procedure* that, given a value γ, decides whether $\delta_{\text{dF}}(P, Q) \leq \gamma$. For this, we define the *free-space matrix* F. This is a Boolean $n \times n$ matrix such that for $i, j = 1, \ldots, n$, we set $F_{ij} = 1$ if $d(p_i, q_j) \leq \gamma$, and $F_{ij} = 0$, otherwise. Then $\delta_{\text{dF}}(P, Q) \leq \gamma$ if and only if F *allows a monotone traversal from* $(1, 1)$ *to* (n, n), i.e., if we can go from entry F_{11} to F_{nn} while only going down, to the right, or diagonally, and while only using 1-entries. This is captured by the *reach matrix* R, which is again an $n \times n$ Boolean matrix. We set $R_{11} = F_{11}$, and for $i, j = 1, \ldots, n$, $(i, j) \neq (1, 1)$, we set $R_{ij} = 1$ if $F_{ij} = 1$ and either one of $R_{(i-1)j}$, $R_{i(j-1)}$, or $R_{(i-1)(j-1)}$ equals 1 (we define any entry of the form $R_{(-1)j}$ or $R_{i(-1)}$ to be 0). Otherwise, we set $R_{ij} = 0$. From these definitions, it is straightforward to compute F and R in total time $O(n^2)$. Furthermore, by construction we have $\delta_{\text{dF}}(P, Q) \leq \gamma$ if and only if $R_{nn} = 1$; see Figure 3.

With this decision procedure at hand, we can use binary search to compute $\delta_{\text{dF}}(P, Q)$ in total time $O(n^2 \log n)$ by observing that the optimum must be achieved for one of the n^2 distances $d(p_i, q_j)$, for $i, j = 1, \ldots, n$. Through a more direct use of dynamic programming, the running time can be reduced to $O(n^2)$ [12].

We call an algorithm an *α-approximation* for the Fréchet distance if, given point sequences P and Q, it returns a number between $\delta_{\text{dF}}(P, Q)$ and $\alpha\, \delta_{\text{dF}}(P, Q)$.

2.2 Hardness Assumptions

Strong Exponential Time Hypothesis (SETH). As is well-known, the *k-SAT problem* is as follows: given a CNF-formula Φ over Boolean variables $x_1, \ldots, x_n$ with clause width k, decide whether there is an assignment of $x_1, \ldots, x_n$ that satisfies Φ. Of course, k-SAT is NP-hard, and it is conjectured that no subexponential algorithm for the problem exists [14]. The Strong Exponential Time Hypothesis (SETH) goes one step further and basically states that the exhaustive search running time of $O^*(2^n)$ cannot be improved to

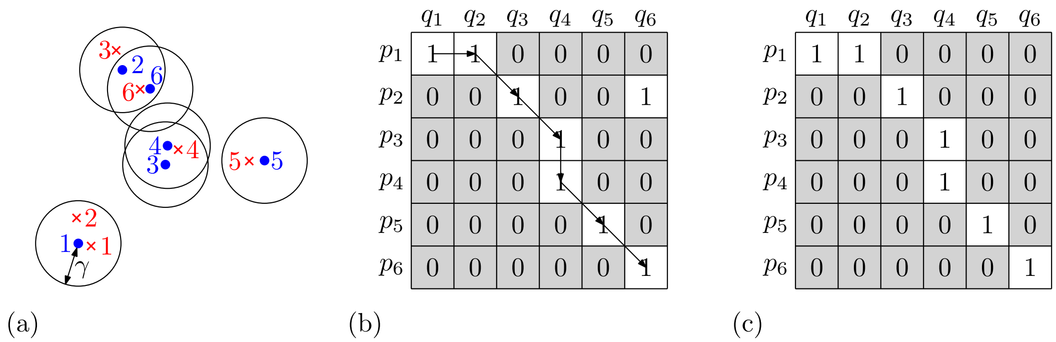

Figure 3: Decision procedure for the discrete Fréchet distance: (a) two point sequences P (disks) and Q (crosses); (b) the associated free-space matrix; (c) the resulting reach matrix.

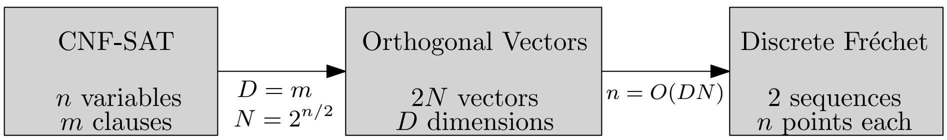

Figure 4: The structure of the reductions and the associated parameters.

$O^*(1.99^n)$ [16, 17].

Conjecture 2.1 (SETH)**.** *For no $\varepsilon > 0$, k-SAT has an $O(2^{(1-\varepsilon)n})$ algorithm for all $k \geq 3$.*

The fastest known algorithms for k-SAT take time $O(2^{(1-c/k)n})$ for some constant $c > 0$ [19]. Thus, SETH is reasonable and, due to lack of progress in the last decades, can be considered unlikely to fail. It is by now a standard assumption for conditional lower bounds.

Orthogonal Vectors (OV). Many reductions involving SETH proceed through the *Orthogonal Vectors problem* (OV), which is defined as follows: given two sequences $u_1, \ldots, u_N$ $v_1, \ldots, v_N \in \{0,1\}^D$ of N vectors in D dimensions, decide whether there are $i, j \in \{1, \ldots, N\}$ with $u_i \perp v_j$, i.e., with $(u_i)_k \cdot (v_j)_k = 0$, for $k = 1, \ldots, D$. We denote by $(u_i)_k$ the k-th coordinate of the i-th vector. This problem has a trivial $O(DN^2)$ algorithm. The fastest known algorithm runs in time $N^{2-1/O(\log(D/\log N))}$ [3], which is only slightly subquadratic for $D \gg \log N$. It is known that OV has no strongly subquadratic time algorithms unless SETH fails [21]; we present a proof for completeness; see Figure 4 for the structure of the reductions in this paper.

Lemma 2.2. *If there exists an $\varepsilon > 0$ such that OV has an algorithm with running time $D^{O(1)} \cdot N^{2-\varepsilon}$, then SETH fails.*

Proof. Let Φ be a k-SAT formula Φ with n variables $x_1, \ldots, x_n$ and m clauses $C_1, \ldots, C_m$. We construct an instance for OV with $N = 2^{n/2}$ and $D = m$. Without loss of generality, we assume that n is even. Denote by $\phi_1, \ldots, \phi_N$ all possible truth assignments to the first $n/2$ variables $x_1, \ldots, x_{n/2}$. For each such assignment ϕ_i, we construct a vector u_i

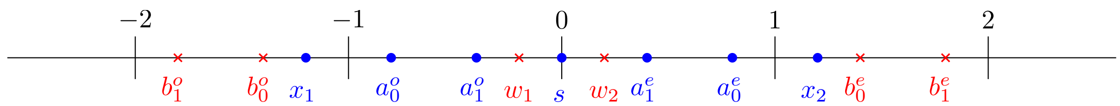

Figure 5: The point set $\mathcal{P}$ constructed in the conditional lower bound.

such that $(u_i)_l = 0$ if ϕ_i satisfies at least one literal in C_l, and $(u_i)_l = 1$, otherwise, for $l = 1, \ldots, D$. Similarly, we enumerate all truth assignments $\psi_1, \ldots, \psi_N$ for the remaining variables $x_{n/2+1}, \ldots, x_n$, and for each ψ_j we construct a vector v_j where $(v_j)_l = 0$ if ψ_j satisfies at least one literal in C_l, and $(v_j)_l = 1$, otherwise, for $l = 1, \ldots, D$. Then, $(u_i)_l \cdot (v_j)_l = 0$ if and only if one of ϕ_i and ψ_j satisfies the clause C_j. Thus, we have $u_i \perp v_j$ if and only if (ϕ_i, ψ_j) constitutes a satisfying assignment for the formula Φ. The vectors can be constructed in time $O(DN)$.

It follows that any algorithm for OV with running time $D^{O(1)} \cdot N^{2-\varepsilon}$ gives an algorithm for k-SAT with running time $m^{O(1)} 2^{(1-\varepsilon/2)n}$. Since $m \leq (2n)^k = 2^{o(n)}$, this contradicts SETH. □

We call a problem Π *OV-hard* if there is a reduction that transforms an instance I of OV with parameters N, D, to an equivalent instance I' of Π of size $n \leq D^{O(1)}N$, in time $D^{O(1)}N^{2-\varepsilon}$, for some $\varepsilon > 0$. A strongly subquadratic algorithm (i.e., with running time $O(n^{2-\varepsilon'})$ for some $\varepsilon' > 0$) for Π would then yield an algorithm for OV with running time $D^{O(1)}N^{2-\min\{\varepsilon,\varepsilon'\}}$. Thus, by Lemma 2.2, if an OV-hard problem has a strongly subquadratic time algorithm, then SETH fails. Most known SETH-based lower bounds for polynomial time problems are actually OV-hardness results; our lower bound in the next section is no exception. Note that OV-hardness is potentially stronger than a SETH-based lower bound, since it may be that SETH fails, while OV still has no strongly subquadratic algorithms.

3 Hardness of Approximation in One Dimension

We prove OV-hardness of the discrete Fréchet distance on one-dimensional curves. By Lemma 2.2, this also yields a SETH-based lower bound.

Let $u_1, \ldots, u_N, v_1, \ldots, v_N \in \{0,1\}^D$ be an instance of the Orthogonal Vectors problem. Without loss of generality, we assume that D is even (if not, we duplicate a coordinate). We show how to construct two sequences P and Q of $O(DN)$ points in $\mathbb{R}$ in time $O(DN)$ such that there are $i, j \in \{1, \ldots, N\}$ with $u_i \perp v_j$ if and only if $\delta_{\mathrm{dF}}(P, Q) \leq 1$. Our sequences P and Q consist of elements from the following set $\mathcal{P}$ of 13 points; see Figure 5.

- $a_0^o = -0.8$, $a_1^o = -0.4$, $a_1^e = 0.4$, $a_0^e = 0.8$.
- $b_1^o = -1.8$, $b_0^o = -1.4$, $b_0^e = 1.4$, $b_1^e = 1.8$.
- $s = 0$, $x_1 = -1.2$, $x_2 = 1.2$
- $w_1 = -0.2$, $w_2 = 0.2$.

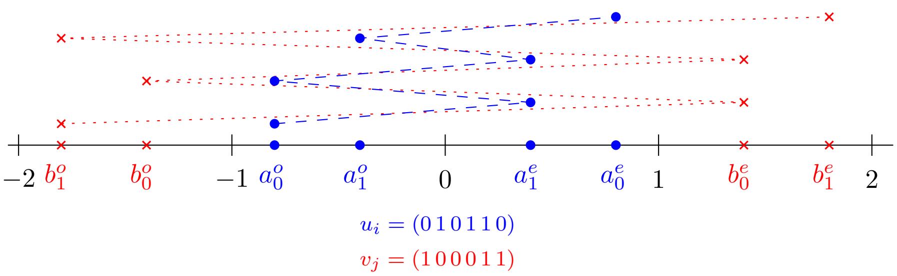

Figure 6: The vector gadgets A_i (disks) and B_j (crosses) for the vectors $u_i = (0,1,0,1,1,0)$ and $v_j = (1,0,0,0,1,1)$. The optimal traversal goes through A_i and B_j in parallel. As A_i and B_j are not orthogonal, the distance in the fifth position is 1.4.

We first construct *vector gadgets*. For each u_i, $i \in \{1,\dots,N\}$, we define a sequence A_i of D points from $\mathcal{P}$ as follows: for $k = 1,\dots,D$ let $p \in \{o,e\}$ be the parity of k (odd or even). Then, the k-th point of A_i is $a^p_{(u_i)_k}$. Similarly, for each v_j, we define a sequence B_j of D points from $\mathcal{P}$. For B_j, we use the points b^p_* instead of a^p_*. The next claim characterizes how the vector gadgets encode orthogonality, see Figure 6.

Claim 3.1. *Fix $i, j \in \{1,\dots,N\}$ and let β be a traversal of (A_i, B_j). We have: (i) if β is not the parallel traversal, then $\delta(\beta) \geq 1.8$; (ii) if β is the parallel traversal and $u_i \perp v_j$, then $\delta(\beta) \leq 1$; and (iii) if β is the parallel traversal and $u_i \not\perp v_j$, then $\delta(\beta) \geq 1.4$.*

Proof. First, suppose that β is not a parallel traversal. Consider the first time when β makes a move on one sequence but not the other. Then, the current points on A_i and B_j lie on different sides of s, which forces $\delta(\beta) \geq \min\{d(a_1^o, b_0^e), d(a_1^e, b_0^o)\} = 1.8$.

Next, suppose that $u_i \perp v_j$. Then, the parallel traversal β of A_i and B_j has $\delta(\beta) \leq 1$. Indeed, for each coordinate $k \in \{1,\dots,D\}$, at least one of $(u_i)_k$ and $(v_j)_k$ is 0. Thus, the k-th point of A_i and the k-th point of B_j lie on the same side of s, and at least one of them is in $\{a_0^o, a_0^e, b_0^o, b_0^e\}$. It follows that the distance between the k-th points in β is at most 1, for $k = 1,\dots,D$.

Finally, suppose that $(u_i)_k = (v_j)_k = 1$ for some k. Let β be the parallel traversal of A_i and B_j, and consider the time when β reaches the k-th points of A_i and B_j. These are either $\{a_1^o, b_1^o\}$ or $\{a_1^e, b_1^e\}$, so $\delta(\beta) = \min\{d(a_1^o, b_1^o), d(a_1^e, b_1^o)\} \geq 1.4$. □

Let W be the sequence of $D(N-1)$ points that alternates between a_0^o and a_0^e, starting with a_0^o (recall that D is even). We set

$$P = W \circ x_1 \circ \Big(\bigcirc_{i=1}^N \; s \circ A_i \Big) \circ s \circ x_2 \circ W$$

and

$$Q = \bigcirc_{j=1}^N w_1 \circ B_j \circ w_2,$$

where $\circ$ denotes the concatenation of sequences, see Figure 7 for an example. The idea is to implement an *or-gadget*. If there is a pair of orthogonal vectors, then P and Q should be

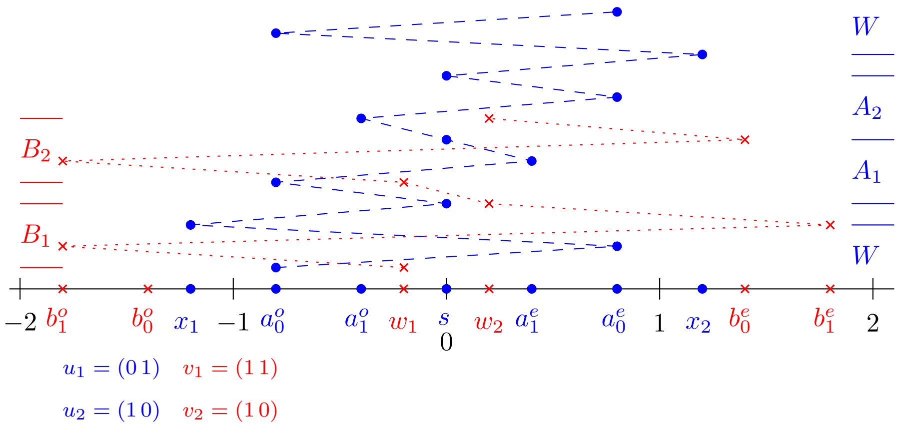

Figure 7: An example reduction for the vectors $u_1 = (0, 1)$, $u_2 = (1, 0)$, $v_1 = (1, 1)$, and $v_2 = (1, 0)$. The vectors u_1 and v_2 are orthogonal.

able to reach the corresponding vector gadgets and traverse them simultaneously. If there is no such pair, it should not be possible to "cheat". The purpose of the sequences W and the points w_1 and w_2 is to provide a buffer so that one sequence can wait while the other sequence catches up. The purpose of the points x_1, x_2, and s is to synchronize the traversal so that no cheating can occur. The next two claims make this precise. First, we show completeness.

Claim 3.2. *If there are $i, j \in \{1, \ldots, N\}$ with $u_i \perp v_j$, then $\delta_{\mathrm{dF}}(P, Q) \leq 1$.*

Proof. Fix $i, j \in \{1, \ldots, N\}$ with $u_i \perp v_j$. We traverse P and Q as follows (see Figure 8 for an example):

1. P goes through $D(N - j)$ points of W; Q stays at w_1.

2. For $k = 1, \ldots, j - 1$, we perform a parallel traversal of B_k and the next portion of W starting with a_0^o and the first point on B_k. When the traversal reaches a_0^e and the last point of B_k, P stays at a_0^e while Q goes to w_2 and w_1. If $k < j - 1$, the traversal continues with a_0^o on P and the first point of B_{k+1} on Q. If $k = j - 1$, we go to Step 3.

3. P proceeds to x_1 and walks until the point s before A_i, Q stays at w_1 before B_j.

4. P and Q go in parallel through A_i and B_j, until the pair (s, w_2) after A_i and B_j.

5. P continues to x_2 while Q stays at w_2.

6. For $k = j+1, \ldots, N$, P goes to the next a_0^o on W while Q goes to w_1. We then perform a simultaneous traversal of B_k and the next portion of W. When the traversal reaches a_0^e and the last point of B_k, P stays at a_0^e while Q continues to w_2. If $k < N$, the traversal continues with the next iteration, otherwise we go to Step 7.

7. P finishes the traversal of W, while Q stays at w_2.

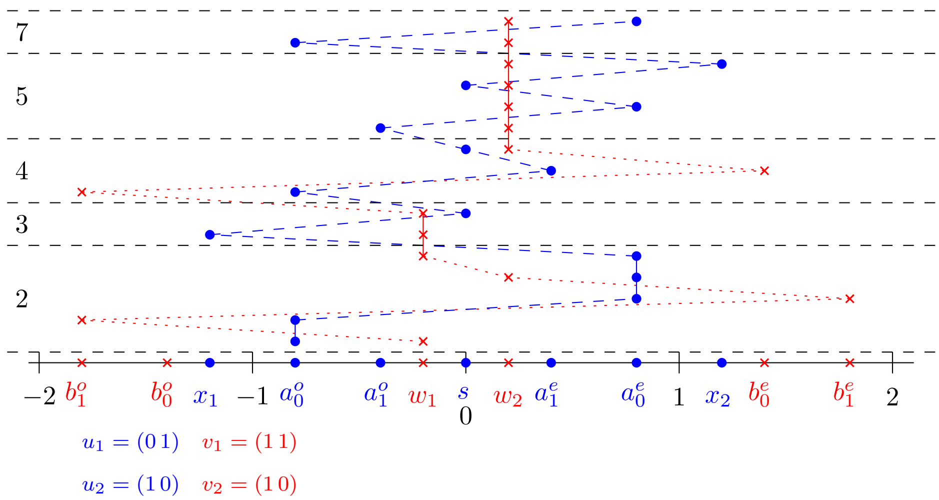

Figure 8: A traversal for the example from Figure 7 with distance 1. The numbers on the left correspond to the steps in the proof of Claim 3.2.

We use the notation max-$d(S,T) := \max_{s\in S,t\in T} d(s,t)$, and max-$d(s,T) :=$ max-$d(\{s\},T)$, max-$d(S,t) :=$ max-$d(S,\{t\})$. The traversal maintains a maximum distance of 1: for Step 1, this is implied by max-$d(\{a_0^o, a_0^e\}, w_1) = 1$. For Step 2, it follows from D being even and from

$$\text{max-}d(a_0^o, \{b_1^o, b_0^o\}) = \text{max-}d(a_0^e, \{b_1^e, b_0^e, w_1, w_2\}) = 1.$$

For Step 3, it is because max-$d(\{x_1, a_0^o, a_1^o, s, a_1^e, a_0^e\}, w_1) = 1$. For Step 4, we use Claim 3.1 and $d(s, w_2) = 0.2$. In Step 5, it follows from max-$d(\{a_0^o, a_1^o, s, a_1^e, a_0^e, x_2\}, w_2) = 1$. In Step 6, we again use that D is even and that

$$\text{max-}d(a_0^o, \{b_1^o, b_0^o, w_1\}) = \text{max-}d(a_0^e, \{b_1^e, b_0^e, w_2\}) = 1.$$

Step 7 uses max-$d(\{a_0^o, a_0^e\}, w_2) = 1$. □

The second claim establishes the soundness of the construction.

Claim 3.3. *If there are no $i, j \in \{1, \ldots, N\}$ with $u_i \perp v_j$, then $\delta_{\mathrm{dF}}(P,Q) \geq 1.4$.*

Proof. Let β be a traversal of (P,Q). Consider the time when β reaches x_1 on P. If Q is not at either w_1 or at a point from $B^o = \{b_0^o, b_1^o\}$, then $\delta(\beta) \geq 1.4$, and we are done. Next, suppose that the current position is in $\{x_1\} \times B^o$. In the next step, β must advance P to s or Q to $\{b_0^e, b_1^e\}$ (or both).[2] In each case, we get $\delta(\beta) \geq 1.4$. From now on, suppose we reach x_1 in position (x_1, w_1). After that, P must advance to s, because advancing Q to B^o would take us to a position in $\{x_1\} \times B^o$, implying $\delta(\beta) \geq 1.4$ as we saw above.

Now consider the next step when Q leaves w_1. Then Q must go to a point from B^o. At this time, P must be at a point from $A^o = \{a_0^o, a_1^o\}$, or we would get $\delta(\beta) \geq 1.4$ (note that P has already passed the point x_1). This point on P belongs to a vector gadget

[2] Recall that we assumed D to be even.

A_i or to the final gadget W (again because P is already past x_1). In the latter case, we have $\delta(\beta) \geq 1.4$, because in order to reach the final W, P must have gone through x_2 and $d(x_2, w_1) = 1.4$. Thus, P is at a point in A^o in a vector gadget A_i, and Q is at the starting point (from B^o) of a vector gadget B_j.

Now β must alternate in parallel in P and Q among both sides of s, or again $\delta(\beta) \geq 1.4$, see Claim 3.1. Furthermore, if P does not start in the first point of A_i, then eventually P has to go to s while Q has to go to a point in B^o or stay in $\{b_0^e, b_1^e\}$, giving $\delta(\beta) \geq 1.4$. Thus, we may assume that β simultaneously reached the starting points of A_i and B_j and traverses A_i and B_j in parallel. By assumption, the vectors u_i, v_j are not orthogonal, so Claim 3.1 gives $\delta(\beta) \geq 1.4$. □

Theorem 3.4. *Fix $\alpha \in [1, 1.4)$. Computing an α-approximation of the discrete Fréchet distance in one dimension is OV-hard. In particular, the discrete Fréchet distance in one dimension has no strongly subquadratic α-approximation unless SETH fails.*

Proof. We use Claims 3.2 and 3.3 and the fact that P and Q can be computed in time $O(DN)$ from $u_1, \dots, u_N, v_1, \dots, v_N$: any $O(n^{2-\varepsilon})$ time α-approximation for the discrete Fréchet distance would yield an OV algorithm with runing time $D^{O(1)} N^{2-\varepsilon}$, which by Lemma 2.2 contradicts SETH. □

Remark 3.5. *The proofs of Claims 3.2 and 3.3 yield a system of linear inequalities that constrain the points in $\mathcal{P}$. Using this system, one can see that the inapproximability factor* 1.4 *in Theorem 3.4 is best possible for our current proof.*

4 Approximation Quality of the Greedy Algorithm

In this section we study the following greedy algorithm. Let $P = \langle p_1, \dots, p_n \rangle$ and $Q = \langle q_1, \dots, q_n \rangle$ be two sequences of n points in $\mathbb{R}^d$. We construct a greedy traversal $\beta_{\text{greedy}} = \beta_{\text{greedy}}(P, Q)$ as follows: We begin at (p_1, q_1). If the current position is (p_i, q_j), there are at most three possible successor configurations: (p_{i+1}, q_j), (p_i, q_{j+1}), and (p_{i+1}, q_{j+1}) (or fewer, if we have already reached the last point from P or Q). Among these, we pick the pair $(p_{i'}, q_{j'})$ that minimizes the distance $d(p_{i'}, q_{j'})$. We stop when we reach (p_n, q_n). We denote the largest distance taken by the greedy traversal by $\delta_{\text{greedy}}(P, Q) := \delta(\beta_{\text{greedy}}(P, Q))$.

Theorem 4.1. *Let P and Q be two sequences of n points in $\mathbb{R}^d$. Then, $\delta_{\text{dF}}(P, Q) \leq \delta_{\text{greedy}}(P, Q) \leq 2^{O(n)} \delta_{\text{dF}}(P, Q)$. Both inequalities are tight, i.e., there are polygonal curves P, Q with $\delta_{\text{greedy}}(P, Q) = \delta_{\text{dF}}(P, Q) > 0$ and $\delta_{\text{greedy}}(P, Q) = 2^{\Omega(n)} \delta_{\text{dF}}(P, Q) > 0$, respectively.*

The inequality $\delta_{\text{dF}}(P, Q) \leq \delta_{\text{greedy}}(P, Q)$ follows directly from the definition, since the traversal $\beta_{\text{greedy}}(P, Q)$ is a candidate for an optimal traversal. Furthermore, one can check that if P and Q are increasing one-dimensional sequences, then the greedy traversal is optimal (this is similar to the merge step in mergesort). Thus, there are examples where $\delta_{\text{greedy}}(P, Q) = \delta_{\text{dF}}(P, Q)$. It remains to show the upper bound $\delta_{\text{greedy}}(P, Q) \leq 2^{O(n)} \delta_{\text{dF}}(P, Q)$ and to provide an example where this inequality is tight. This is done in the next two sections.

4.1 Upper Bound

We call a pair $p_i p_{i+1}$ of consecutive points on P an *edge* of P, for $i = 1, \dots, n-1$, and similarly for Q. Let m be the total number of edges of P and Q, and let $\ell_1 \le \ell_2 \le \cdots \le \ell_m$ be the sorted sequence of the edge lengths. We pick $k^* \in \{0, \dots, m\}$ minimum such that

$$4\,\delta_{\mathrm{dF}}(P,Q) + 2\sum_{i=1}^{k^*} \ell_i < \ell_{k^*+1},$$

where we set $\ell_{m+1} = \infty$. We define δ^* as the left hand side, $\delta^* := 4\,\delta_{\mathrm{dF}}(P,Q) + 2\sum_{i=1}^{k^*} \ell_i$.

Lemma 4.2. *We have (i) $\delta^* \ge 4\delta_{\mathrm{dF}}(P,Q)$; (ii) $\sum_{i=1}^{k^*} \ell_i \le \delta^*/2 - 2\,\delta_{\mathrm{dF}}(P,Q)$; (iii) there is no edge with length in $(\delta^*/2 - 2\delta_{\mathrm{dF}}(P,Q), \delta^*)$; and (iv) $\delta^* \le 3^{k^*} 4\delta_{\mathrm{dF}}(P,Q)$.*

Proof. Properties (i) and (ii) follow by definition. Property (iii) holds since for $i = 1, \dots, k^*$, we have $\ell_i \le \delta^*/2 - 2\delta_{\mathrm{dF}}(P,Q)$, by (ii), and for $i = k^*+1, \dots, m$, we have $\ell_i \ge \delta^*$, by definition. It remains to prove (iv): for $k = 0, \dots, k^*$, we set $\delta_k = 4\,\delta_{\mathrm{dF}}(P,Q) + 2\sum_{i=1}^{k} \ell_i$, and we prove by induction that $\delta_k \le 3^k\, 4\delta_{\mathrm{dF}}(P,Q)$. For $k = 0$, this is immediate. Now suppose we know that $\delta_{k-1} \le 3^{k-1}\, 4\delta_{\mathrm{dF}}(P,Q)$, for some $k \in \{1, \dots, k^*\}$. Then, $k \le k^*$ implies $\ell_k \le \delta_{k-1}$, so $\delta_k = \delta_{k-1} + 2\ell_k \le 3\delta_{k-1} \le 3^k 4\delta_{\mathrm{dF}}(P,Q)$, as desired. Now (iv) follows from $\delta^* = \delta_{k^*}$. □

We call an edge *long* if it has length at least δ^*, and *short* otherwise. In other words, the short edges have lengths $\ell_1, \dots, \ell_{k^*}$, and the long edges have lengths $\ell_{k^*+1}, \dots, \ell_m$. Let β be an optimal traversal of P and Q, i.e., $\delta(\beta) = \delta_{\mathrm{dF}}(P,Q)$.

Lemma 4.3. *The sequences P and Q have the same number of long edges. Furthermore, if $p_{i_1}p_{i_1+1}, \dots, p_{i_k}p_{i_k+1}$ and $q_{j_1}q_{j_1+1}, \dots, q_{j_k}q_{j_k+1}$ are the long edges of P and of Q, for $1 \le i_1 < \cdots < i_k < n$ and $1 \le j_1 < \cdots < j_k < n$, then both β and β_{greedy} contain the steps $(p_{i_1}, q_{j_1}) \to (p_{i_1+1}, q_{j_1+1}), \dots, (p_{i_k}, q_{j_k}) \to (p_{i_k+1}, q_{j_k+1})$.*

Proof. First, we show that for every long edge $p_i p_{i+1}$ of P, the optimal traversal β contains the step $(p_i, q_j) \to (p_{i+1}, q_{j+1})$, where q_j, q_{j+1} is a long edge of Q. Consider the step of β from p_i to p_{i+1}. This step has to be of the form $(p_i, q_j) \to (p_{i+1}, q_{j+1})$ for some $q_j \in Q$: since $\max\{d(p_i, q_j), d(p_{i+1}, q_j)\} \ge d(p_i, p_{i+1})/2 \ge \delta^*/2 \ge 2\delta_{\mathrm{dF}}(P,Q)$, by Lemma 4.2(i), staying in q_j would result in $\delta(\beta) \ge 2\delta_{\mathrm{dF}}(P,Q)$. Now, since $\max\{d(p_i, q_j), d(p_{i+1}, q_{j+1})\} \le \delta(\beta) = \delta_{\mathrm{dF}}(P,Q)$, the triangle inequality gives $d(q_j, q_{j+1}) \ge d(p_i, p_{i+1}) - 2\,\delta_{\mathrm{dF}}(P,Q) \ge \delta^* - 2\,\delta_{\mathrm{dF}}(P,Q)$. Lemma 4.2(iii) now implies $d(q_j, q_{j+1}) \ge \delta^*$, so the edge $q_j q_{j+1}$ is long.

Thus, β traverses every long edge of P in parallel with a long edge of Q. A symmetric argument shows that β traverses every long edge of Q in parallel with a long edge of P. Since β is monotone, it follows that P and Q have the same number of long edges, and that β traverses them in parallel in their order of occurrence along P and Q.

It remains to show that the greedy traversal β_{greedy} traverses the long edges of P and Q in parallel. Set $i_0 = j_0 = 0$. We will prove for $a \in \{0, \dots, k-1\}$ that if β_{greedy} contains the position (p_{i_a+1}, q_{j_a+1}), then it also contains the step $(p_{i_{a+1}}, q_{j_{a+1}}) \to (p_{i_{a+1}+1}, q_{j_{a+1}+1})$

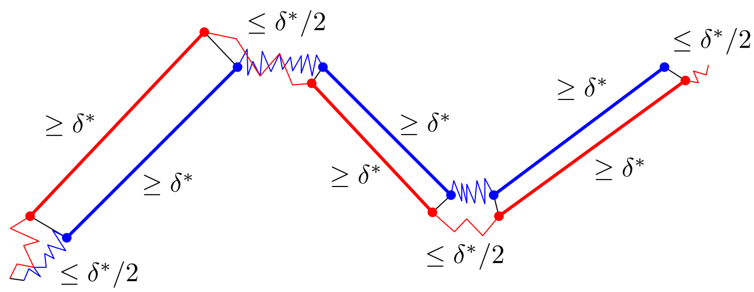

Figure 9: The long edges are matched by the greedy and any optimal traversal. The distance at the endpoints of the long edges is at most $\delta_{\mathrm{dF}}(P,Q)$. The short edges cannot increase the Fréchet distance beyond δ^*.

and hence the position $(p_{i_{a+1}+1}, q_{j_{a+1}+1})$. The claim on β_{greedy} then follows by induction on a, since β_{greedy} contains the position (p_1, q_1) by definition. Thus, fix $a \in \{0, \dots, k-1\}$ and suppose that β_{greedy} contains (p_{i_a+1}, q_{j_a+1}). We need to show that β_{greedy} also contains the step $(p_{i_{a+1}}, q_{j_{a+1}}) \to (p_{i_{a+1}+1}, q_{j_{a+1}+1})$. For better readability, we write i for i_a, j for j_a, i' for i_{a+1}, and j' for j_{a+1}. Consider the first position of β_{greedy} when β_{greedy} reaches either $p_{i'}$ or $q_{j'}$. Without loss of generality, this position is of the from $(p_{i'}, q_l)$, for some $l \in \{j+1, \dots, j'\}$. Then, $d(p_{i'}, q_l) \le \delta^*/2 - \delta_{\mathrm{dF}}(P,Q)$, since we saw that $d(p_{i'}, q_{j'}) \le \delta(\beta) = \delta_{\mathrm{dF}}(P,Q)$ and since the remaining edges between q_l and $q_{j'}$ are short and thus have total length at most $\delta^*/2 - 2\,\delta_{\mathrm{dF}}(P,Q)$, by Lemma 4.2(ii). The triangle inequality now gives $d(p_{i'+1}, q_l) \ge d(p_{i'}, p_{i'+1}) - d(p_{i'}, q_l) \ge \delta^*/2 + \delta_{\mathrm{dF}}(P,Q)$. If $l < j'$, the same argument applied to q_{l+1} shows that $d(p_{i'}, q_{l+1}) \le \delta^*/2 - \delta_{\mathrm{dF}}(P,Q)$ and thus $d(p_{i'+1}, q_{l+1}) \ge \delta^*/2 + \delta_{\mathrm{dF}}(P,Q)$. Thus, β_{greedy} moves to $(p_{i'}, q_{l+1})$. If $l = j'$, then β_{greedy} takes the step $(p_{i'}, q_{j'}) \to (p_{i'+1}, q_{j'+1})$, as $d(p_{i'+1}, q_{j'+1}) \le \delta(\beta) = \delta_{\mathrm{dF}}(P,Q)$, but $d(p_{i'}, q_{j'+1}), d(p_{i'+1}, q_{j'}) \ge \delta^* - \delta_{\mathrm{dF}}(P,Q) \ge 3\,\delta_{\mathrm{dF}}(P,Q)$, by Lemma 4.2(i). □

Finally, we can show the desired upper bound on the greedy algorithm; see Figure 9.

Lemma 4.4. *We have $\delta_{\text{greedy}}(P,Q) \le \delta^*/2$.*

Proof. By Lemma 4.3, P and Q have the same number of long edges. Let $p_{i_1}p_{i_1+1}, \dots, p_{i_k}p_{i_k+1}$ and $q_{j_1}q_{j_1+1}, \dots, q_{j_k}, q_{j_k+1}$ be the long edges of P and of Q, where $1 \le i_1 < \dots < i_k < n$ and $1 \le j_1 < \dots < j_k < n$. By Lemma 4.3, β_{greedy} contains the positions (p_{i_a}, q_{j_a}) and (p_{i_a+1}, q_{j_a+1}) for $a = 1, \dots, k$, and $d(p_{i_a}, q_{j_a}), d(p_{i_a+1}, q_{i_a+1}) \le \delta_{\mathrm{dF}}(P,Q)$ for $a = 1, \dots, k$. Thus, setting $i_0 = j_0 = 0$ and $i_{k+1} = j_{k+1} = n$, we can focus on the subtraversals $\beta_a = (p_{i_a+1}, q_{i_a+1}), \dots, (p_{i_{a+1}}, q_{i_{a+1}})$ of β_{greedy}, for $a = 0, \dots, k$. Now, since all edges traversed in β_a are short, and since $d(p_{i_a+1}, q_{i_a+1}) \le \delta_{\mathrm{dF}}(P,Q)$, we have $\delta(\beta_a) \le \delta_{\mathrm{dF}}(P,Q) + \delta^*/2 - 2\,\delta_{\mathrm{dF}}(P,Q) \le \delta^*/2$ by Lemma 4.2(iii) and the triangle inequality. Thus, $\delta(\beta_{\text{greedy}}) \le \max\{\delta_{\mathrm{dF}}(P,Q), \delta(\beta_1), \dots, \delta(\beta_k)\} \le \delta^*/2$, as desired. □

Lemmas 4.2(iv) and 4.4 prove the desired inequality $\delta_{\text{greedy}}(P,Q) \le 2^{O(n)}\delta_{\mathrm{dF}}(P,Q)$, since $k^* \le m = 2n - 2$.

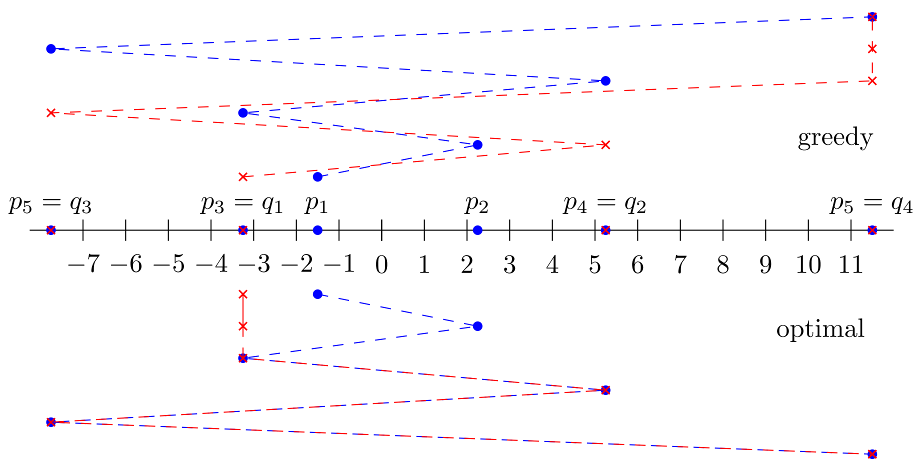

Figure 10: The greedy algorithm traverses P and Q in parallel, increasing the distance by a constant factor in each step. The optimal algorithm delays the traversal of Q for two steps, giving a perfect match for the remainder.

4.2 Tight Example for the Upper Bound

Fix $1 < \alpha < 2$. Consider the sequence $P = \langle p_1, \dots, p_n \rangle$ with $p_i := (-\alpha)^i$ and the sequence $Q = \langle q_1, \dots, q_{n-2} \rangle$ with $q_i := (-\alpha)^{i+2}$. We show the following:

1. The greedy traversal $\beta_{\text{greedy}}(P, Q)$ makes $n-2$ simultaneous steps in P and Q followed by 2 single steps in P. This results in a maximal distance of $\delta_{\text{greedy}}(P, Q) = \alpha^n + \alpha^{n-1}$.

2. The traversal which makes 2 single steps in P followed by $n - 2$ simultaneous steps in both P and Q has distance $\alpha^3 + \alpha^2$.

Together, this shows that $\delta_{\text{greedy}}(P, Q)/\delta_{\text{dF}}(P, Q) = \Omega(\alpha^n) = 2^{\Omega(n)}$, proving that the inequality $\delta_{\text{greedy}}(P, Q) \leq 2^{O(n)}\delta_{\text{dF}}(P, Q)$ is tight, see Figure 10.

To see (1), assume that we are at position (p_i, q_i). Moving to (p_i, q_{i+1}) would result in a distance of $d(p_i, q_{i+1}) = \alpha^{i+3} + \alpha^i$. Similarly, the other possible moves to (p_{i+1}, q_i) and to (p_{i+1}, q_{i+1}) would result in distances $\alpha^{i+2} + \alpha^{i+1}$, and $\alpha^{i+3} - \alpha^{i+1}$, respectively. It can be checked that for all $\alpha > 1$ we have $\alpha^{i+3} + \alpha^i > \alpha^{i+2} + \alpha^{i+1}$. Moreover, for all $\alpha < 2$ we have $\alpha^{i+2} + \alpha^{i+1} > \alpha^{i+3} - \alpha^{i+1}$. Thus, the greedy algorithm makes the move to (p_{i+1}, q_{i+1}). Using induction, this shows that the greedy traversal starts with $n - 2$ simultaneous moves in P and Q. In the end, the greedy algorithm has to take two single moves in P. Thus, the greedy traversal contains the pair (p_{n-1}, q_{n-2}), which is in distance $d(p_{n-1}, q_{n-2}) = \alpha^n + \alpha^{n-1} = 2^{\Omega(n)}$.

To see (2), note that the traversal which makes 2 single steps in P followed by $n-2$ simultaneous moves in P and Q starts with (p_1, q_1) and (p_2, q_1) followed by (p_i, q_{i-2}) for $i = 2, \dots, n$. Note that $d(p_1, q_1) = \alpha^3 - \alpha$, $d(p_2, q_1) = \alpha^3 + \alpha^2$, and $p_i = q_{i-2}$, so that the remaining distances are 0. Thus, we have $\delta_{\text{dF}}(P, Q) \leq \alpha^3 + \alpha^2 = O(1)$.

5 Improved Approximation Algorithm

Let $P = \langle p_1, \ldots, p_n \rangle$ and $Q = \langle q_1, \ldots, q_n \rangle$ be two sequences of n points in $\mathbb{R}^d$, where d is constant. Let $1 \le \alpha \le n$. We show how to find a value δ^* with $\delta_{\mathrm{dF}}(P,Q) \le \delta^* \le \alpha\delta_{\mathrm{dF}}(P,Q)$ in time $O(n \log n + n^2/\alpha)$. For simplicity, we will assume that all points on P and Q are pairwise distinct. This can be achieved by an infinitesimal perturbation of the point set.

5.1 Decision Algorithm

We begin by describing an approximate decision procedure. For this, we prove the following theorem.

Theorem 5.1. *Let P and Q be two sequences of n points in $\mathbb{R}^d$, and let $1 \le \alpha \le n$. Suppose that the points of P and Q have been sorted along each coordinate axis. There exists a decision algorithm with running time $O(n^2/\alpha)$ and the following properties: if $\delta_{\mathrm{dF}}(P,Q) \le 1$, the algorithm returns YES; if $\delta_{\mathrm{dF}}(P,Q) \ge \alpha$, the algorithm returns NO; if $\delta_{\mathrm{dF}}(P,Q) \in (1,\alpha)$, the algorithm may return either YES or NO. The running time depends exponentially on d.*

Consider the regular d-dimensional grid with diameter 1 (all cells are axis-parallel cubes with side length $1/\sqrt{d}$). The distance between two grid cells C and D, $d(C,D)$, is defined as the smallest distance between a point in C and a point in D. The distance between a point x and a grid cell C, $d(x,C)$, is the distance between x and the closest point in C. For a point $x \in \mathbb{R}^d$, we write B_x for the closed unit ball with center x and C_x for the grid cell that contains x (since we are interested in approximation algorithms, we may assume that all points of $P \cup Q$ lie strictly inside the cells). We compute for each point $r \in P \cup Q$ the grid cell C_r that contains it. We also record for each nonempty grid cell C the number of points from Q contained in C. This can be done in total linear time as follows: we scan the points from $P \cup Q$ in x_1-order, and we group the points according to the grid intervals that contain them. Then we split the lists that represent the x_2-,…, x_d-order correspondingly, and we recurse on each group to determine the grouping for the remaining coordinate axes. Each iteration takes linear time, and there are d iterations, resulting in a total time of $O(n)$. In the following, we will also need to know for each non-empty cell the neighborhood of all cells that have a certain constant distance from it. These neighborhoods can be found in linear time by modifying the above procedure as follows: before performing the grouping, we make $O(1)$ copies of each point $r \in P \cup Q$ that we translate suitably to hit all neighboring cells for r. By using appropriate cross-pointers, we can then identify the neighbors of each non-empty cell in total linear time. Afterwards, we perform a clean-up step, so that only the original points remain.

A grid cell C is *full* if $|C \cap Q| \ge 5n/\alpha$. Let $\mathcal{F}$ be the set of full grid cells. Clearly, $|\mathcal{F}| \le \alpha/5$. We say that two full cells $C, D \in \mathcal{F}$ are *adjacent* if $d(C,D) \le 4$. This defines a graph H on $\mathcal{F}$ of constant degree. Using the neighborhood finding procedure from above, we can determine H and its connected components $L_1, \ldots, L_k$ in time $O(n+\alpha)$. For $C \in \mathcal{F}$, the *label* L_C of C is the connected component of H containing C, see Figure 11.

For each $q \in Q$, we search for a full cell $C \in \mathcal{F}$ with $d(q,C) \le 2$. If such a cell exists, we label q with $L_q = L_C$; otherwise, we set $L_q = \perp$. Similarly, for each $p \in P$, we

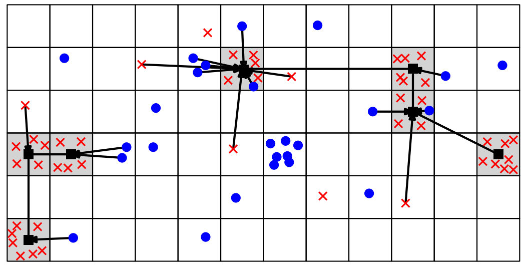

Figure 11: The full cells are shown grey. The graph H has two connected components. The labels of the vertices are indicated by arrows. The remaining vertices are unlabeled.

search a full cell $C \in \mathcal{F}$ with $d(p, C) \leq 1$. In case of success, we set $L_p = L_C$; otherwise, we set $L_p = \perp$. Using the neighborhood finding procedure from above, this takes linear time. Let $P' = \{p \in P \mid L_p \neq \perp\}$ and $Q' = \{q \in Q \mid L_q \neq \perp\}$. The labeling has the following properties.

Lemma 5.2. *We have*

1. *for every $r \in P \cup Q$, the label L_r is uniquely determined;*
2. *for every $x, y \in P' \cup Q'$ with $L_x = L_y$, we have $d(x, y) \leq \alpha$;*
3. *if $p \in P'$ and $q \in B_p \cap Q$, then $L_p = L_q$; and*
4. *if $p \in P \setminus P'$, there are $O(n/\alpha)$ points $q \in Q$ with $d(p, C_q) \leq 1$. Hence, $|B_p \cap Q| = O(n/\alpha)$.*

Proof. Let $r \in P \cup Q$ and suppose there are $C, D \in \mathcal{F}$ with $d(r, C) \leq 2$ and $d(r, D) \leq 2$. Then $d(C, D) \leq d(C, r) + d(r, D) \leq 4$, so C and D are adjacent in H. It follows that $L_C = L_D$ and that L_r is determined uniquely.

Fix $x, y \in P' \cup Q'$ with $L_x = L_y$. By construction, there are $C, D \in \mathcal{F}$ with $d(x, C) \leq 2$, $d(y, D) \leq 2$ and $L_C = L_D$. This means that C and D are in the same component of H. Therefore, C and D are connected by a sequence of adjacent cells in $\mathcal{F}$. We have $|\mathcal{F}| \leq \alpha/5$, any two adjacent cells have distance at most 4, and each cell has diameter 1. Thus, the triangle inequality gives $d(x, y) \leq 2 + 4(|\mathcal{F}| - 1) + |\mathcal{F}| + 2 \leq \alpha$.

Let $p \in P'$ and $q \in B_p \cap Q$. Take $C \in \mathcal{F}$ with $d(p, C) \leq 1$. By the triangle inequality, $d(q, C) \leq d(q, p) + d(p, C) \leq 2$, so $L_q = L_p = L_C$.

Take $p \in P$ and suppose there is a grid cell C with $|C \cap Q| > 5n/\alpha$ and $d(p, C) \leq 1$. Then $C \in \mathcal{F}$, so $L_p \neq \perp$, which means that $p \in P'$. The contrapositive gives (4). □

Lemma 5.2 enables us to design an efficient approximation algorithm. For this, we define the *approximate free-space matrix* F. This is an $n \times n$ matrix with entries from $\{0, 1\}$. For $i, j \in \{1, \ldots, n\}$, we set $F_{ij} = 1$ if either (i) $p_i \in P'$ and $L_{p_i} = L_{q_j}$; or (ii) $p_i \in P \setminus P'$ and $d(p_i, q_j) \leq 1$. Otherwise, we set $F_{ij} = 0$. The matrix F is approximate in the following sense:

Lemma 5.3. *If $\delta_{\mathrm{dF}}(P,Q) \le 1$, then F allows a monotone traversal from $(1,1)$ to (n,n). Conversely, if F has a monotone traversal from $(1,1)$ to (n,n), then $\delta_{\mathrm{dF}}(P,Q) \le \alpha$.*

Proof. Suppose that $\delta_{\mathrm{dF}}(P,Q) \le 1$. Then there is a monotone traversal β of (P,Q) with $\delta(\beta) \le 1$. By Lemma 5.2(3), β is also a traversal of F.

Now let β be a monotone traversal of F. By Lemma 5.2(2), we have $\delta(\beta) \le \alpha$, as desired. □

Additionally, we define the *approximate reach matrix* R, which is an $n \times n$ matrix with entries from $\{0,1\}$. We set $R_{ij} = 1$ if F allows a monotone traversal from $(1,1)$ to (i,j), and $R_{ij} = 0$, otherwise. By Lemma 5.3, R_{nn} is an α-approximate indicator for $\delta_{\mathrm{dF}} \le 1$. We describe how to compute the rows of R successively in total time $O(n^2/\alpha)$.

First, we perform the following preprocessing steps: we break Q into *intervals*, where an interval is a maximal consecutive subsequence of points $q \in Q$ with the same label $L_q \neq \perp$. For each point in an interval, we store pointers to the first and the last point of the interval. This takes linear time. Furthermore, for each $p_i \in P \setminus P'$, we compute a sparse representation T_i of the corresponding row of F, i.e., a sorted list of all the column indices j for which $F_{ij} = 1$. This can be done in $O(n^2/\alpha)$ time as follows: in the preprocessing phase, we have determined for input point the grid cell that contains it. By a single scan through Q, we can thus obtain for each non-empty grid cell the ordered subsequence of points from Q contained in it. For each $p_i \in P \setminus P'$, we inspect all grid cells with distance at most 1 from p_i (this neighborhood was found during preprocessing). By the proof of Lemma 5.2(4), the total number of points from Q in these grid cells is $O(n/\alpha)$, so we can find the sparse representation T_i in $O(n/\alpha)$ time by filtering and merging these lists.

Now we successively compute a sparse representation for each row i of R, i.e., a sorted list I_i of disjoint intervals $[a,b] \in I_i$ such that for $j = 1, \dots, n$, we have $R_{ij} = 1$ if and only if there is an interval $[a,b] \in I_i$ with $j \in [a,b]$. We initialize I_1 as follows: if $F_{11} = 0$, we set $I_1 = \emptyset$ and abort. Otherwise, if $p_1 \in P'$, then I_1 is initialized with the interval of q_1 (since $F_{11} = 1$, we have $L_{p_1} = L_{q_1}$ by Lemma 5.2(3)). If $p_1 \in P \setminus P'$, we determine the maximum b such that $F_{1j} = 1$ for all $j = 1, \dots, b$, and we initialize I_1 with the *singleton* intervals $[j,j]$ for $j = 1, \dots, b$. This can be done in time $O(n/\alpha)$, irrespective of whether p_i lies in P' or not.

Now suppose we already have the interval list I_i for some row i, and we want to compute the interval list I_{i+1} for the next row. We consider two cases.

Case 1: $p_{i+1} \in P'$. If $L_{p_{i+1}} = L_{p_i}$, we simply set $I_{i+1} = I_i$. Otherwise, we go through the intervals $[a,b] \in I_i$ in order. For each interval $[a,b]$, we check whether the label of q_b or the label of q_{b+1} equals the label of p_{i+1}. If so, we add the maximal interval $[b',c]$ to I_{i+1} with $b' = b$ or $b' = b+1$ and $L_{p_{i+1}} = L_{q_j}$ for all $j = b', \dots, c$. With the information from the preprocessing phase, this takes $O(1)$ time per interval. The resulting set of intervals may not be disjoint (if $p_i \in P \setminus P'$), but any two overlapping intervals have the same endpoint. Also, intervals with the same endpoint appear consecutively in I_{i+1}. We next perform a clean-up pass through I_{i+1}: we partition the intervals into conscutive groups with the same endpoint, and in each group, we only keep the largest interval. All this takes time $O(|I_i| + |I_{i+1}|)$.

Case 2: $p_{i+1} \in P \setminus P'$. In this case, we have a sparse representation T_{i+1} of the corresponding row in F at our disposal. We simultaneously traverse I_i and T_{i+1} to compute I_{i+1} as follows: for each $j \in \{1, \dots, n\}$ with $F_{(i+1)j} = 1$, if I_i has an interval containing $j-1$ or j or if $[j-1, j-1] \in I_{i+1}$, we add the singleton $[j, j]$ to I_{i+1}. This takes total time $O(|I_i| + |I_{i+1}| + n/\alpha)$.

The next lemma shows that the interval representation remains sparse throughout the execution of the algorithm, and that the intervals I_i indeed represent the approximate reach matrix R.

Lemma 5.4. *We have $|I_i| = O(n/\alpha)$ for $i = 1, \dots, n$. Furthermore, the intervals in I_i correspond exactly to the 1-entries in the approximate reach matrix R.*

Proof. First, we prove that $|I_i| = O(n/\alpha)$ for $i = 1, \dots, n$. This is done by induction on i. We begin with $i = 1$. If $p_1 \in P'$, then $|I_1| = 1$. If $p_1 \in P \setminus P'$, then Lemma 5.2(4) shows that the first row of F contains at most $O(n/\alpha)$ 1-entries, so $|I_1| = O(n/\alpha)$. Next, suppose that we know by induction that $|I_i| = O(n/\alpha)$. We must argue that $|I_{i+1}| = O(n/\alpha)$. If $p_{i+1} \in P \setminus P'$, then the $(i+1)$-th row of F contains $O(n/\alpha)$ 1-entries by Lemma 5.2(4), and $|I_{i+1}| = O(n/\alpha)$ follows directly by construction. If $p_{i+1} \in P'$ and $L_{p_{i+1}} = L_{p_i}$, then $I_{i+1} = I_i$, and the claim follows by induction. Finally, if $p_{i+1} \in P'$ and $L_{p_{i+1}} \neq L_{p_i}$, then by construction, every interval in I_i gives rise to at most one new interval in I_{i+1}. Thus, by induction, $|I_{i+1}| \leq |I_i| = O(n/\alpha)$.

Second, we prove that I_i represents the i-th row of R, for $i = 1, \dots, n$. Again, the proof is by induction. For $i = 1$, the claim holds by construction, because the first row of R consists of the initial segment of 1s in F. Next, suppose we know that I_i represents the i-th row of R. We must argue that I_{i+1} represents the $(i+1)$th row of R. If $p_{i+1} \in P \setminus P'$, this follows directly by construction, because the algorithm explicitly checks the conditions for each possible 1-entry of R ($R_{(i+1)j}$ can only be 1 if $F_{(i+1)j} = 1$). If $p_{i+1} \in P'$ and $L_{p_{i+1}} = L_{p_i}$, then the $(i+1)$-th row of F is identical to the i-th row of F, and the same holds for R: there can be no new monotone paths, and all old monotone paths can be extended by one step along Q. Finally, consider the case $p_{i+1} \in P'$ and $L_{p_{i+1}} \neq L_{p_i}$. If $p_i \in P \setminus P'$, then every interval in I_i is a singleton $[b, b]$, from which a monotone path could potentially reach $(i+1, b)$ and $(i+1, b+1)$, and from there walk to the right. We explicitly check both of these possibilities. If $p_i \in P'$, then for every interval $[a, b] \in I_i$ and for all $j \in [a, b]$ we have $L_{q_j} = L_{p_i} \neq L_{p_{i+1}}$. Thus, the only possible move is to $(i+1, b+1)$, and from there walk to the right, which is what we check. □

The first part of Lemma 5.4 implies that the total running time is $O(n^2/\alpha)$, since each row is processed in time $O(n/\alpha)$. By Lemma 5.3 and the second part of Lemma 5.4, if I_n has an interval containing n then $\delta_{\mathrm{dF}}(P, Q) \leq \alpha$, and if $\delta_{\mathrm{dF}}(P, Q) \leq 1$ then n appears in I_n. Since the intervals in I_n are sorted, this condition can be checked in $O(1)$ time. Theorem 5.1 follows.

5.2 Optimization Procedure

We now leverage Theorem 5.1 to an optimization procedure.

Theorem 5.5. *Let P and Q be two sequences of n points in $\mathbb{R}^d$, and let $1 \leq \alpha \leq n$. There is an algorithm with running time $O(n^2 \log n/\alpha)$ that computes a number δ^* with $\delta_{\mathrm{dF}}(P,Q) \leq \delta^* \leq \alpha\delta_{\mathrm{dF}}(P,Q)$. The running time depends exponentially on d.*

Proof. If $\alpha \leq 5$, we compute $\delta_{\mathrm{dF}}(P,Q)$ directly in $O(n^2)$ time. Otherwise, we set $\alpha' = \alpha/5$. We sort the points of $P \cup Q$ according to the coordinate axes, and we compute a (1/3)-well-separated pair decomposition $\mathcal{P} = \{(S_1,T_1),\ldots,(S_k,T_k)\}$ for $P \cup Q$ in time $O(n \log n)$ [11]. Recall the properties of a well-separated pair decomposition: (i) for all pairs $(S,T) \in \mathcal{P}$, we have $S,T \subseteq P \cup Q$, $S \cap T = \emptyset$, and $\max\{\mathrm{diam}(S), \mathrm{diam}(T)\} \leq d(S,T)/3$ (here, $\mathrm{diam}(S)$ denotes the maximum distance between any two points in S); (ii) the number of pairs is $k = O(n)$; and (iii) for every distinct $q, r \in P \cup Q$, there is exactly one pair $(S,T) \in \mathcal{P}$ with $q \in S$ and $r \in T$, or vice versa.

For each pair $(S_i,T_i) \in \mathcal{P}$, we pick arbitrary $s \in S_i$ and $t \in T_i$, and set $\delta_i = 3d(s,t)$. After sorting, we can assume that $\delta_1 \leq \ldots \leq \delta_k$. We call δ_i a *YES-entry* if the algorithm from Theorem 5.1 on input α' and the point sets P an Q scaled by a factor of δ_i returns YES; otherwise, we call δ_i a *NO-entry*. First, we test whether δ_1 is a YES-entry. If so, we return $\delta^* = \alpha'\delta_1$. If δ_1 is a NO-entry, we perform a binary search on $\delta_1,\ldots,\delta_k$: we set $l = 1$ and $r = k$. Below, we will prove that δ_k must be a YES-entry. We set $m = \lceil (l+r)/2 \rceil$. If δ_m is a NO-entry, we set $l = m$, otherwise, we set $r = m$. We repeat this until $r = l + 1$. In the end, we return $\delta^* = \alpha'\delta_r$. The total running time is $O(n \log n + n^2 \log n/\alpha)$. Our procedure works exactly like binary search, but we presented it in detail in order to emphasize that $\delta_1,\ldots,\delta_k$ is not necessarily monotone: NO-entries and YES-entries may alternate.

We now argue correctness. The algorithm finds a YES-entry δ_r such that either $r = 1$ or δ_{r-1} is a NO-entry. By Theorem 5.1, any δ_i is a NO-entry if $\delta_i \leq \delta_{\mathrm{dF}}(P,Q)/\alpha'$. Thus, we certainly have $\delta^* = \alpha'\delta_r > \delta_{\mathrm{dF}}(P,Q)$. Now take a traversal β with $\delta(\beta) = \delta_{\mathrm{dF}}(P,Q)$, and let $(p,q) \in P \times Q$ be a position in β that has $d(p,q) = \delta(\beta)$. There is a pair $(S_{r^*},T_{r^*}) \in \mathcal{P}$ with $p \in S_{r^*}$ and $q \in T_{r^*}$, or vice versa. Let $s \in S_{r^*}$ and $t \in T_{r^*}$ be the points we used to define δ_{r^*}. Then

$$d(s,t) \geq d(p,q) - \mathrm{diam}(S_{r^*}) - \mathrm{diam}(T_{r^*}) \geq d(p,q) - 2d(S_{r^*},T_{r^*})/3 \geq d(p,q)/3,$$

and

$$d(s,t) \leq d(p,q) + \mathrm{diam}(S_{r^*}) + \mathrm{diam}(T_{r^*}) \leq d(p,q) + 2d(S_{r^*},T_{r^*})/3 \leq 5d(p,q)/3,$$

so $\delta_{r^*} = 3d(s,t) \in [\delta(\beta), 5\delta(\beta)]$. Since by Theorem 5.1 any δ_i is a YES-entry if $\delta_i \geq \delta_{\mathrm{dF}}(P,Q)$, all δ_i with $i \geq r^*$ are YES-entries (in particular, δ_k is a YES-entry). Thus, $\delta^* \leq \alpha'\delta_{r^*} \leq 5\alpha'\delta_{\mathrm{dF}}(P,Q) \leq \alpha\delta_{\mathrm{dF}}(P,Q)$. □

The running time of Theorem 5.5 can be improved as follows.

Theorem 5.6. *Let P and Q be two sequences of n points in $\mathbb{R}^d$, and let $1 \leq \alpha \leq n$. There is an algorithm with running time $O(n \log n + n^2/\alpha)$ that computes a number δ^* with $\delta_{\mathrm{dF}}(P,Q) \leq \delta^* \leq \alpha\delta_{\mathrm{dF}}(P,Q)$. The running time depends exponentially on d.*

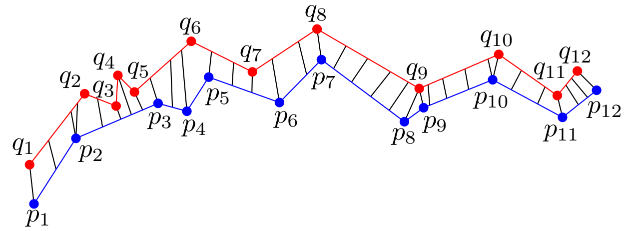

Figure 12: Two polygonal chains and a traversal for them, indicated by black segments between matched points.

Proof. If $\alpha \le 4$, we can compute $\delta_{\mathrm{dF}}(P,Q)$ exactly. Otherwise, we use Theorem 5.5 to compute a number δ' with $\delta_{\mathrm{dF}}(P,Q) \le \delta' \le n \cdot \delta_{\mathrm{dF}}(P,Q)$, or, equivalently, $\delta_{\mathrm{dF}}(P,Q) \in [\delta'/n, \delta']$. This takes time $O(n \log n)$. Set $i^* = \lceil \log(n/\alpha) \rceil + 1$ and for $i = 1, \dots, i^*$ let $\alpha_i = n/2^{i+1}$. Also, set $a_1 = \delta'/n$ and $b_1 = \delta'$.

We iteratively obtain better estimates for $\delta_{\mathrm{dF}}(P,Q)$ by repeating the following for $i = 1, \dots, i^* - 1$. As an invariant, at the beginning of iteration i, we have $\delta_{\mathrm{dF}}(P,Q) \in [a_i, b_i]$ with $b_i/a_i = 4\alpha_i$. We use the algorithm from Theorem 5.1 with inputs α_i and P and Q scaled by a factor $2a_i$ (since $\alpha_i \ge \alpha_{i^*-1} = n/2^{\lceil \log(n/\alpha) \rceil + 1} \ge \alpha/4$, the algorithm can be applied). If the answer is YES, it follows that $\delta_{\mathrm{dF}}(P,Q) \le \alpha_i 2a_i = b_i/2$, so we set $a_{i+1} = a_i$ and $b_{i+1} = b_i/2$. If the answer is NO, then $\delta_{\mathrm{dF}}(P,Q) \ge 2a_i$, so we set $a_{i+1} = 2a_i$ and $b_{i+1} = b_i$. This needs time $O(n^2/\alpha_i)$ and maintains the invariant.

In the end, we return b_{i^*}. The invariant guarantees $\delta_{\mathrm{dF}}(P,Q) \in [a_{i^*}, b_{i^*}]$ and $b_{i^*}/a_{i^*} = 4\alpha_{i^*} \le \alpha$, as desired. The total running time is proportional to

$$n \log n + \sum_{i=1}^{i^*-1} n^2/\alpha_i = n \log n + \sum_{i=1}^{i^*-1} n 2^{i+1} \le n \log n + n 2^{i^*+1} = O(n \log n + n^2/\alpha). \qquad \square$$

6 The Continuous Greedy Algorithm

In this section, we extend the greedy algorithm from Section 4 to continuous curves. Let us briefly review the relevant definitions. In this section only, we denote by $P, Q : [1,n] \to \mathbb{R}^d$ two d-dimensional polygonal chains with n vertices. We assume that P and Q are parametrized in such a way that if we set $p_i = P(i)$ and $q_i = Q(i)$, for $i = 1, \dots, n$, then $P(i+\lambda) = (1-\lambda)p_i + \lambda p_{i+1}$ and $Q(i+\lambda) = (1-\lambda)q_i + \lambda q_{i+1}$, for $i = 1, \dots, n-1$, and $\lambda \in [0,1]$. We call $p_1, \dots, p_n$ and $q_1, \dots, q_n$ the *vertices* of P and Q. A *traversal* of P and Q is a pair $\beta = (\varphi, \psi)$ of continuous, monotone, surjective functions $\varphi, \psi : [1,n] \to [1,n]$. The *continuous Fréchet distance* between P and Q, $\delta_{\mathrm{F}}(P,Q)$, is defined as

$$\delta_{\mathrm{F}}(P,Q) = \inf_{(\varphi,\psi) \in \Phi} \max_{s \in [1,n]} d(P(\varphi(s)), Q(\psi(s))),$$

where Φ is the set of all traversals of P and Q, see Figure 12. The results of Alt and Godau imply that there always exists a traversal that achieves $\delta_{\mathrm{F}}(P,Q)$ [6], but since this is not immediately obvious, we use the infimum in the definition.

The greedy algorithm. The greedy algorithm is analogous to the discrete case: we iteratively build a traversal for P and Q. In each step, we have an *intermediate position* $(p, q) \in P \times Q$, where at least one of p and q is a vertex. If $p = p_n$ or $q = q_n$, we follow the other curve until the end. Otherwise, let p' and q' be the vertices on P and Q strictly after p and q. We find the point q^* on qq' closest to p' and the point p^* on pp' closest to q'. If $d(p', q^*) \leq d(p^*, q')$, we uniformly walk to (p', q^*), otherwise we walk to (p^*, q'). We repeat until we reach the endpoints (p_n, q_n). Since we always advance to a new vertex, the process terminates after at most $2n$ steps. Let $\beta_{\text{greedy}} = (\varphi_{\text{g}}, \psi_{\text{g}})$ be the resulting *greedy* traversal, and set

$$\delta_{\text{greedy}} = \max_{s \in [1,n]} d(P(\varphi_{\text{g}}(s)), Q(\psi_{\text{g}}(s))).$$

Furthermore, let $\beta = (\varphi, \psi)$ be an *optimal* traversal with

$$\delta_{\text{F}}(P, Q) = \max_{s \in [1,n]} d(P(\varphi(s)), Q(\psi(s))).$$

As mentioned above, the results by Alt and Godau imply that β exists [6].

Definitions and first properties. For brevity, we will write δ_{F} for $\delta_{\text{F}}(P, Q)$. Similar to Section 4.1, we let $\ell_1 \leq \ell_2 \leq \cdots \leq \ell_m$ be the sorted sequence of edge lengths, and we pick $k^* \in \{0, \ldots, m\}$ minimum with

$$A\Big(\delta_{\text{F}} + \sum_{i=1}^{k^*} \ell_i\Big) \leq \ell_{k^*+1},$$

where $\ell_{m+1} = \infty$ and A is an appropriate large constant. We set

$$\delta^* = A\Big(\delta_{\text{F}} + \sum_{i=1}^{k^*} \ell_i\Big).$$

The following lemma is analogous to Lemma 4.2.

Lemma 6.1. *We have (i)* $\delta_{\text{F}} \leq (1/A)\delta^*$*; (ii)* $\sum_{i=1}^{k^*} \ell_i \leq (1/A)\delta^*$*; and (iii)* $\delta^* \leq (A+1)^{k^*} A\delta_{\text{F}}$.

Proof. Properties (i) and (ii) follow by definition. It remains to prove (iii): for $k = 0, \ldots, k^*$, we set $\delta_k = A\big(\delta_{\text{dF}}(P, Q) + \sum_{i=1}^{k} \ell_i\big)$, and we prove by induction that $\delta_k \leq (A+1)^k A\delta_{\text{dF}}(P, Q)$. For $k = 0$, this is immediate. Now suppose we have $\delta_{k-1} \leq (A+1)^{k-1} A\delta_{\text{dF}}(P, Q)$, for some $k \in \{1, \ldots, k^*\}$. Then, $k \leq k^*$ implies $\ell_k \leq \delta_{k-1}$, so $\delta_k = \delta_{k-1} + A\ell_k \leq (A+1)\delta_{k-1} \leq (A+1)^k A\delta_{\text{dF}}(P, Q)$, as desired. Now (iii) follows from $\delta^* = \delta_{k^*}$. □

We call an edge *long* if it has length at least δ^*, and *short* otherwise. Before we get into the details of the analysis, let us provide some intuition for our proof. In general, we would like to give a similar argument as in the discrete case: both the greedy traversal and every optimal traversal must match long edges uniformly, while short edge are irrelevant for the approximation factor. However, in the continuous setting, the situation is not as clear cut: an optimal traversal may match vertices and short edges against the interior of long

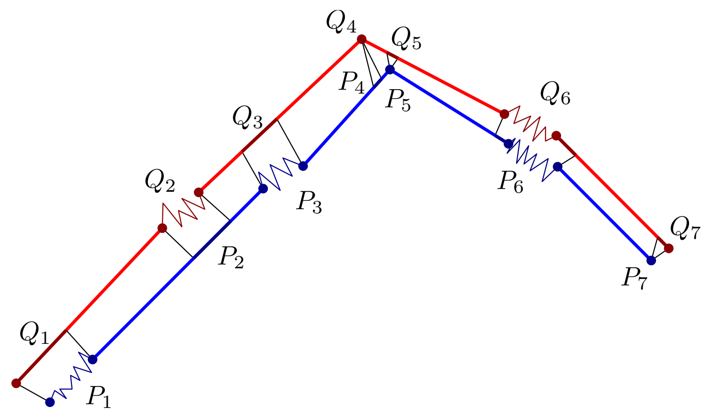

Figure 13: The subcurves on P and Q induced by an optimal traversal. The subcurves P_2, P_4, Q_3, and Q_5 are straight, the others are pointed.

edges. To deal with this, we fix an optimal traversal, and we mark the subcurves on P and Q during which the optimal traversal is at a vertex or at a short edge on either curve. Now, as in the discrete case, we would like to argue that these subcurves are "short" and that between two consecutive subcurves the greedy traversal and the optimal traversal behave essentially "uniformly". However, this does not have to be true: under certain circumstances, two adjacent subcurves on P or on Q may be "close" to each other, so that it is not clear how the greedy algorithm will deal with them. Therefore, we need to perform a more detailed analysis to understand the behavior of the subcurves. Our analysis shows that this situation can be handled by merging "close" consecutive subcurves in a controlled manner. The resulting sequence of modified subcurves has the desired properties, and we can carry out our strategy as planned. Details follow.

Let $S \subseteq [1, n]$ be the set of all parameters $s \in [1, n]$ such that at least one of $P(\varphi(s))$ or $Q(\psi(s))$ is a vertex or lies on a short edge. By construction, S consists of a finite number of pairwise disjoint closed intervals, $I_1, \ldots, I_k$, ordered from left to right. This induces a sequence of subcurves $P_i = P(\varphi(I_i))$ and $Q_i = Q(\psi(I_i))$, for $i = 1, \ldots, k$, see Figure 13.

A *subcurve* of P or Q is a function of the form $P_{|I}$ or $Q_{|I}$, where $I \subseteq [1, n]$ is a closed interval. If $I \subseteq [i, i+1]$, for some $i \in \{1, \ldots, n-1\}$, we call the subcurve a *subsegment*. A subsegment is *initial*, if $i \in I$, it is *final* if $i + 1 \in I$. A subcurve is *short* if it does not intersect the interior of a long edge. A short subcurve is *maximal* if it is not properly contained in another short subcurve. We call a subcurve *pointed* if it contains a vertex, and *straight* otherwise. Given a subcurve $P_{|I}$ of P, let $I' = \varphi^{-1}(I)$ and $J = \psi(I')$. We say that $Q_{|J}$ is *matched to* $P_{|I}$ by β. We write $|P_{|I}|$ for the length of a subcurve $P_{|I}$. For two points $p, p' \in P$, we denote by $d_P(p, p')$ the distance between p and p' *along* P. We extend this notation to subcurves in the obvious way. Our first technical lemma lets us bound the length of a subcurve that is matched to a subsegment.

Lemma 6.2. *Suppose that β matches a subsegment e of P to a subcurve Q_e of Q. Then $|Q_e| \geq |e| - (2/A)\delta^*$. An analogous statement holds with the roles of P and Q reversed.*

Proof. Let $e = ab$ and let x be the first and y be the last point of Q_e. Since β matches x to

a and y to b, we have

$$|e| = d(a,b) \le d(a,x) + d(x,y) + d(y,b) \le \delta_{\mathrm{F}} + |Q_e| + \delta_{\mathrm{F}} \le |Q_e| + (2/A)\delta^*,$$

by the triangle inequality and Lemma 6.1(i). □

The next technical lemma shows that the subcurves are "close" to each other.

Lemma 6.3. *For every point $p \in P_i$, $i \in \{1, \dots, k\}$ there is a $q \in Q_i$ with $d(p,q) \le (1/A)\delta^*$.*

Proof. By construction, there is a $q \in Q_i$ with $d(p,q) \le \delta_{\mathrm{F}} \le (1/A)\delta^*$, by Lemma 6.1(i). □

We now dig deeper into the structure of the subcurves P_i and Q_i; examples of the different situations can be found in Figure 13.

Lemma 6.4. *The subcurve P_1 consists of a (possibly empty) maximal short subcurve, followed by an initial segment of the first long edge; the subcurve P_k consists of a final segment of the last long edge, followed by a (possibly empty) maximal short subcurve. For $i = 2, \dots, k-1$, the subcurve P_i is either a subsegment of the interior of a long edge, or it consists of a final subsegment of a long edge, followed by a (possibly empty) maximal short subcurve, followed by an initial subsegment of the next long edge. The subsegments may be degenerate (i.e., consist of only one point). If a subsegment is not degenerate, it has length at most $(3/A)\delta^*$. Analogous statements hold for Q.*

Proof. Suppose a subcurve P_i, $i \in \{1, \dots, k\}$, contains a nondegenerate subsegment s of a long edge. By definition, s is matched by β to a short subcurve $Q_e \subset Q_i$. Then, by Lemma 6.1(ii) and Lemma 6.2, we have $|s| \le (2/A)\delta^* + |Q_e| \le (3/A)\delta^*$. In particular, since $(3/A)\delta^* < \delta^*$, no P_i contains a complete long edge.

The claim for P_1 follows, as P_1 contains an initial segment of the first long edge. The claim for P_k holds for analogous reasons. Now consider a subcurve P_i with $i \in \{2, \dots, k-1\}$. If P_i contains at least one vertex p, then P_i contains the maximal short subcurve of P containing p, and the claim follows. If P_i is straight (does not contain a vertex), then P_i must be a subsegment of a long edge: if P_i contains at least one point on a short edge, then by the continuity of φ, it would contain the whole edge, including its end vertices. □

Lemma 6.4 has several consequences for the position of the subcurves. Let C be an appropriate large constant with $1 \gg 1/C \gg 1/A$.

Lemma 6.5. *The following holds:*

(i) *for $i = 1, \dots, k$, at least one of P_i, Q_i is pointed;*

(ii) *for $i = 1, \dots, k$, we have $|P_i|, |Q_i| \le (7/A)\delta^*$.*

(iii) *for any two pointed subcurves P_i, P_j, $i \ne j$, we have $d_P(P_i, P_j) \ge (1 - 6/A)\delta^*$. An analogous statement holds for Q;*

(iv) *for any two straight subcurves* P_i, P_j, $i \neq j$, *we have* $d_P(P_i, P_j) \geq (1 - 8/A)\delta^*$. *An analogous statement holds for* Q*;*

(v) *for any subcurve* P_i, *there is at most one subcurve* P_j, $j \neq i$, *with* $d_P(P_i, P_j) \leq (1/C)\delta^*$. *In this case,* $j \in \{i-1, i+1\}$. *If* P_i *is pointed, then* Q_i *and* P_j *are straight, and* Q_j *is pointed. If* P_i *is straight, then* Q_i *and* P_j *are pointed, and* Q_j *is straight. An analogous statement holds for* Q.

Proof. (i): If neither P_i nor Q_i is pointed, then by Lemma 6.4 both are subsegments of the interiors of long edges, contradicting the definition.

(ii): By, (i) and Lemma 6.4, if P_i is straight, it is matched by β to a short subcurve Q_i on Q, and thus $|P_i| \leq (3/A)\delta^*$, by Lemma 6.1(ii) and Lemma 6.2. Otherwise, by Lemma 6.4, P_i consists of a short subcurve on P, plus two subsegments of length at most $(3/A)\delta^*$ each. Thus, $|P_i| \leq (7/A)\delta^*$. The argument for Q is analogous.

(iii): If P_i is pointed, then by Lemma 6.4, P_i consists of a final subsegment of a long edge e_P, followed by a (possibly empty) short subcurve, followed by an initial subsegment of a long edge e'_P. Let P_l be the subcurve that contains the startpoint of e_P. Again by Lemma 6.4, P_l consists of a final subsegment of a long edge, followed by a (possibly empty) short subcurve, followed by an initial subsegment on e_P. Furthermore, the subsegments of e_P on P_i and on P_l have length at most $(3/A)\delta^*$. Thus, for all pointed P_j, $j < i$,

$$d_P(P_j, P_i) \geq d_P(P_l, P_i) \geq \delta^* - 2(3/A)\delta = (1 - 6/A)\delta^*.$$

The argument for $j > i$ is analogous.

(iv): If P_i is straight, then Q_i is pointed, by (i). Let $l < i$ be maximum such that Q_l is pointed. By (iii), we have $d_Q(Q_i, Q_l) \geq (1 - 6/A)\delta^*$, and by Lemma 6.2, the subsegment on Q between Q_l and Q_i is matched to a subcurve P_σ of P of length at least $(1 - 8/A)\delta^*$. Thus, by (i), for every straight P_j with $j < i$, we have $d_P(P_j, P_i) \geq (1 - 8/A)\delta^*$. The argument for $j > i$ is analogous.

(v): Suppose that P_i is pointed and suppose there exists a subcurve P_j, $j < i$, with $d_P(P_i, P_j) \leq (1/C)\delta^*$. By monotonicity, we also have $d_P(P_{i-1}, P_i) \leq (1/C)\delta^*$, and by (iii) and since $1/C < 1 - 8/A$, the subcurve P_{i-1} is straight. Furthermore, for any other straight subcurve P_l, we have

$$\begin{aligned}
d_P(P_i, P_l) &\geq d_P(P_{i-1}, P_l) - d_P(P_{i-1}, P_i) - |P_i| && \text{(triangle inequality)}\\
&\geq (1 - 8/A)\delta^* - (1/C)\delta^* - (7/A)\delta^* && \text{((iii), assumption, (ii))}\\
&= (1 - 15/A - 1/C)\delta^* \\
&> (1/C)\delta^*. && (A, C \text{ large enough})
\end{aligned}$$

Thus, P_{i-1} is the only curve within distance $(1/C)\delta^*$ from P_i. It follows from (i) that Q_i is straight and that Q_{i-1} is pointed. The cases $j > i$ and P_i straight are analogous. □

To deal with the case that subcurves may be close together, as in Lemma 6.5(v), we modify our subcurves as follows: we go through the subcurves $P_1, \ldots, P_k$ in order. Let

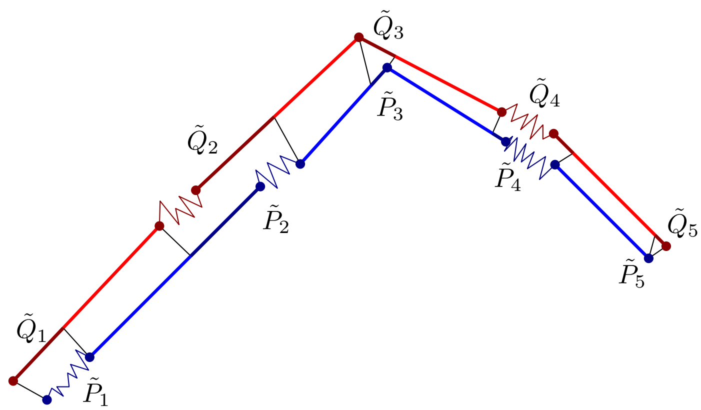

Figure 14: Joining close subcurves. The subcurves $\tilde{P}_2$, $\tilde{P}_3$, $\tilde{Q}_2$, and $\tilde{Q}_3$ are composite. The others are simple.

P_i be the current subcurve. If $d_P(P_i, P_{i+1}) > (1/C)\delta^*$, we proceed to P_{i+1}. Otherwise, if $d_P(P_i, P_{i+1}) \leq (1/C)\delta^*$, we unite P_i and P_{i+1} to a subcurve that goes from the startpoint of P_i to the endpoint of P_{i+1}, and we unite Q_i and Q_{i+1} to a subcurve from the startpoint of Q_i to the endpoint of Q_{i+1}. Then, we proceed to P_{i+2}.

Let $\tilde{P}_1, \ldots, \tilde{P}_{\tilde{k}}$ and $\tilde{Q}_1, \ldots, \tilde{Q}_{\tilde{k}}$ be the resulting sequences of subcurves. We call a subcurve $\tilde{P}_i$ or $\tilde{Q}_i$ *composite* if it was obtained by combining two original subcurves, and *simple* otherwise, see Figure 14. The next lemma collects properties of simple and composite subcurves.

Lemma 6.6. *For $i = 1, \ldots, \tilde{k}$, we have*

(i) *if $\tilde{P}_i$ is simple, then $|\tilde{P}_i|, |\tilde{Q}_i| \leq (7/A)\delta^*$, and for any $j \neq i$, $d_P(\tilde{P}_i, \tilde{P}_j) > (1/C)\delta^*$ and $d_Q(\tilde{Q}_i, \tilde{Q}_j) > (1/2C)\delta^*$;*

(ii) *if $\tilde{P}_i$ is composite, then $|\tilde{P}_i| \leq (2/C)\delta^*$ and $|\tilde{Q}_i| \leq (2/C)\delta^*$. Furthermore, for any $j \neq i$, we have $d_P(\tilde{P}_i, \tilde{P}_j) > (1 - 2/C)\delta^*$ and $d_Q(\tilde{Q}_i, \tilde{Q}_j) > (1 - 2/C)\delta^*$.*

Proof. (i): The bounds on $|\tilde{P}_i|, |\tilde{Q}_i|$ are due to Lemma 6.5(ii). If $\tilde{P}_{i-1}$ is simple, then $d_P(\tilde{P}_{i-1}, \tilde{P}_i) > (1/C)\delta^*$, as otherwise we would have combined the subcurves. If $\tilde{P}_{i-1}$ was obtained by combining two original subcurves P_l, P_{l+1}, then $d_P(P_l, P_{l+1}) \leq (1/C)\delta^*$, and hence $d_P(\tilde{P}_{i-1}, \tilde{P}_i) = d_P(P_{l+1}, \tilde{P}_i) > (1/C)\delta^*$, by Lemma 6.5(v). Similarly, we get $d_P(\tilde{P}_i, \tilde{P}_{i+1}) > (1/C)\delta^*$, and hence $d_P(\tilde{P}_i, \tilde{P}_j) > (1/C)\delta^*$ for all $j \neq i$.

Since the subsegment between $\tilde{Q}_i$ and $\tilde{Q}_{i-1}$ is matched to a subsegment of P with length at least $(1/C)\delta^*$, we have $d_Q(\tilde{Q}_{i-1}, \tilde{Q}_i) \geq (1/C - 2/A)\delta^*$, by Lemma 6.2. Similarly, $d_Q(\tilde{Q}_i, \tilde{Q}_{i+1}) \geq (1/C - 2/A)\delta^*$, so $d_Q(\tilde{Q}_i, \tilde{Q}_j) \geq (1/C - 2/A)\delta^* \geq (1/2C)\delta^*$ for all $j \neq i$.

(ii): Suppose that $\tilde{P}_i$ and $\tilde{Q}_i$ were obtained by combining the original subcurves P_l, P_{l+1} and Q_l, Q_{l+1}. By Lemma 6.5, we have $|P_l|, |P_{l+1}|, |Q_l|, |Q_{l+1}| \leq (7/A)\delta^*$. By construction, we have $d_P(P_l, P_{l+1}) \leq (1/C)\delta^*$, so by Lemma 6.2, $d_Q(Q_l, Q_{l+1}) \leq (2/A + 1/C)\delta^*$. The bounds on $|\tilde{P}_i|$ and $|\tilde{Q}_i|$ now follow, because $|\tilde{P}_i| = |P_l| + d_P(P_l, P_{l+1}) + |P_{l+1}|$, $|\tilde{Q}_i| = |Q_l| + d_Q(Q_l, Q_{l+1}) + |Q_{l+1}|$, and $1/C \gg 1/A$.

By Lemma 6.5(v), $\tilde{P}_i$ consists of a straight and a pointed subcurve. Thus, for $i \neq j$,

$$\begin{aligned} d_P(\tilde{P}_i, \tilde{P}_j) &\geq (1 - 8/A)\delta^* - |\tilde{P}_i| && \text{(triangle inequality, Lemma 6.5(iii,iv))} \\ &\geq (1 - 22/A - 1/C)\delta^* && \text{(first part)} \\ &\geq (1 - 1/2C)\delta^* && (1/C \gg 1/A) \end{aligned}$$

and similarly

$$\begin{aligned} d_Q(\tilde{Q}_i, \tilde{Q}_j) &\geq (1 - 8/A)\delta^* - |\tilde{Q}_i| \\ &\geq (1 - 24/A - 1/C)\delta^* \\ &\geq (1 - 1/2C)\delta^*. \end{aligned}$$

□

The invariant. We say that an edge e of P is *incident* to a subcurve $\tilde{P}_i$, $i \in \{1, \ldots, \tilde{k}\}$, if e and $\tilde{P}_i$ have at least one point in common, and similarly for Q. To analyze the greedy algorithm, we show that the traversal β_{greedy} maintains the following invariant.

Invariant 6.7. *Let (p, q) be an intermediate position of the greedy algorithm. If p is a vertex of $\tilde{P}_i$, $i \in \{1, \ldots, \tilde{k}\}$, then q is the closest point of some vertex of $\tilde{P}_i$ on an edge incident to $\tilde{Q}_i$. If q is a vertex of $\tilde{Q}_i$, $i \in \{1, \ldots, \tilde{k}\}$, then p is the closest point of some vertex of $\tilde{Q}_i$ on an edge incident to $\tilde{P}_i$.*

Invariant 6.7 holds after the first step, because the greedy algorithm proceeds to either p_2 and the closest point of p_2 on q_1q_2 or to q_2 and the closest point of q_2 on p_1p_2. Clearly, p_1p_2 is incident to the subcurve containing p_2 and q_1q_2 is incident to the subcurve containing p_2.

We focus on the situation that the greedy algorithm is at an intermediate position (p, q) such that p is a vertex of $\tilde{P}_i$, $i \in \{1, \ldots, \tilde{k}\}$, and such that q is the closest point of a vertex of $\tilde{P}_i$ on an edge incident to $\tilde{Q}_i$. The case that q is a vertex of Q_i is symmetric. Let p' be the vertex of P strictly after p, and q' the vertex of Q strictly after q. Let q^* be the closest point to p' on qq' and p^* the closest point to q' on pp'. We need two technical lemmas about closest points on the edges of P and Q.

Lemma 6.8. *Let $e \subset Q$ be the edge with $qq' \subset e$. If $q^* \neq q$, then q^* is the closest point for p' on e.*

Proof. Let $\ell(x)$, $x \in \mathbb{R}$, be some parametrization of the line spanned by e. Then the claim follows from the fact that the distance function $x \mapsto d(p', \ell(x))$ is bitonic. □

Lemma 6.9. *Suppose that p is a vertex of $\tilde{P}_i$, and that $q \in Q$ is the closest point for p on a given edge incident to $\tilde{Q}_i$. If $\tilde{P}_i$ is simple, then $d_Q(q, \tilde{Q}_i) \leq (16/A)\delta^*$. If $\tilde{P}_i$ is composite, then $d_Q(q, \tilde{Q}_i) \leq (5/C)\delta^*$ An analogous statement holds with the roles of P and Q exchanged.*

Proof. If q lies in $\tilde{Q}_i$, then $d_Q(q, \tilde{Q}_i) = 0$, and the claim holds. Thus, assume that q lies on a long edge e incident to $\tilde{Q}_i$. Let a be an endpoint of $\tilde{Q}_i$ that lies on e. Then,

$$\begin{aligned}
d_Q(q, \tilde{Q}_i) &\le d(q, a) && (q \text{ and } a \text{ lie on } e)\\
&\le d(q, p) + d(p, a) && (\text{triangle inequality})\\
&\le 2d(p, a) && (q \text{ is } p\text{'s closest point on } e)\\
&\le 2d(p, \tilde{Q}_i) + 2|\tilde{Q}_i| && (\text{triangle inequality})\\
&\le (2/A)\delta^* + 2|\tilde{Q}_i|. && (\text{Lemma 6.3})
\end{aligned}$$

The lemma follows by plugging in the bounds for $|\tilde{Q}_i|$ from Lemma 6.6. □

To show that Invariant 6.7 is maintained, we distinguish two cases, depending on whether $\tilde{P}_i$ is simple or composite.

Case 1. First, suppose that $\tilde{P}_i$ (and $\tilde{Q}_i$) is simple. We perform some quite straightforward calculations to bound the relevant distances.

Lemma 6.10. *We have*

(i) *If $p' \in \tilde{P}_i$, then $d(p', q^*) \le (17/A)\delta^*$;*

(ii) *If $p' \notin \tilde{P}_i$, then $d(p', \tilde{Q}_i) \ge (1/2C)\delta^*$;*

(iii) *If $q' \in \tilde{Q}_i$, then $d(p^*, q') \le (8/A)\delta^*$;*

(iv) *If $q' \notin \tilde{Q}_i$, then $d(q', \tilde{P}_i) \ge (1/3C)\delta^*$.*

Proof. (i): If $p' \in \tilde{P}_i$, then

$$\begin{aligned}
d(p', q^*) \le d(p', q) &\le d(p', \tilde{Q}_i) + d_Q(\tilde{Q}_i, q) && (q \text{ is on } qq', \text{ triangle inequality})\\
&\le (1/A)\delta^* + (16/A)\delta^* = (17/A)\delta^*. && (\text{Lemmas 6.3 and 6.9})
\end{aligned}$$

(ii): If $p' \notin \tilde{P}_i$, then

$$\begin{aligned}
d(p', \tilde{Q}_i) &\ge d(p', \tilde{P}_i) - d(\tilde{P}_i, \tilde{Q}_i) - |\tilde{Q}_i| && (\text{triangle inequality})\\
&\ge (1/C)\delta^* - (1/A)\delta^* - (7/A)\delta^* && (\text{Lemmas 6.6(i) and 6.3})\\
&\ge (1/2C)\delta^* && (1/C \gg 1/A)
\end{aligned}$$

(iii): If $q' \in \tilde{Q}_i$, then

$$\begin{aligned}
d(p^*, q') \le d(p, q') &\le |\tilde{P}_i| + d_Q(\tilde{P}_i, q') && (p \text{ is on } pp', \text{ triangle inequality})\\
&\le (7/A)\delta^* + (1/A)\delta^* = (8/A)\delta^*. && (\text{Lemmas 6.6(i) and 6.3})
\end{aligned}$$

(iv): If $q' \notin \tilde{Q}_i$, then

$$\begin{aligned} d(q', \tilde{P}_i) &\geq d(q', \tilde{Q}_i) - d(\tilde{Q}_i, \tilde{P}_i) - |\tilde{P}_i| && \text{(triangle inequality)} \\ &\geq (1/2C)\delta^* - (1/A)\delta^* - (7/A)\delta^* && \text{(Lemmas 6.6(i) and 6.3)} \\ &\geq (1/3C)\delta^*. && (1/C \gg 1/A) \end{aligned}$$

□

Now a simple case analysis shows that the invariant is maintained.

Lemma 6.11. *Invariant 6.7 holds in the next intermediate step.*

Proof. If $p' \in \tilde{P}_i$ and $q' \in \tilde{Q}_i$, then Invariant 6.7 clearly holds in the next step (in particular, by Lemma 6.8, if $q^* \neq q$, then q^* is the closest point of p' on an edge incident to $\tilde{Q}_i$).
If $p' \in \tilde{P}_i$ and $q' \notin \tilde{Q}_i$, then

$$d(p', q^*) \leq (17/A)\delta^* \leq (1/3C)\delta^* \leq d(\tilde{P}_i, q') \leq d(p^*, q'),$$

by Lemma 6.10(i,iv). Thus, the next intermediate position is (p', q^*), and if $q^* \neq q$, then q^* is the closest point of p' on an edge incident to $\tilde{Q}_i$, by Lemma 6.8.
If $p' \notin \tilde{P}_i$ and $q' \in \tilde{Q}_i$, then

$$d(p^*, q') \leq (8/A)\delta^* \leq (1/3C)\delta^* \leq d(p', \tilde{Q}_i) - |\tilde{Q}_i| - d(\tilde{Q}_i, q^*) \leq d(p', q^*),$$

by Lemma 6.10(ii,iii), Lemma 6.6(i), Lemma 6.9 and the triangle inequality. Thus, the next intermediate position is (p^*, q'), and p^* is the closest point of q' on an edge incident to $\tilde{P}_i$.

If $p' \notin \tilde{P}_i$ and $q' \notin \tilde{Q}_i$, then p' is the first vertex of $\tilde{P}_{i+1}$, q' is the first vertex of $\tilde{Q}_{i+1}$, p^* lies on the segment between $\tilde{P}_i$ and $\tilde{P}_{i+1}$, and q^* lies on the segment between $\tilde{P}_i$ and $\tilde{P}_{i+1}$. If the next intermediate position is (p^*, q'), then Invariant 6.7 clearly holds in the next step. If the next intermediate position is (p', q^*), it remains to argue that q^* is indeed the closest point for p' on the segment incident to $\tilde{Q}_i$ and $\tilde{Q}_{i+1}$. Since the optimal traversal β passes the segment between $\tilde{P}_i$ and $\tilde{P}_{i+1}$ and the segment between $\tilde{Q}_i$ and $\tilde{Q}_{i+1}$ together,

$$d(p', q^*) = \min\{d(p', q^*), d(p^*, q')\} \leq \delta_{\mathrm{F}} \leq (1/A)\delta^*,$$

by Lemma 6.1(i), whereas

$$\begin{aligned} d(p', q) &\geq d(p', \tilde{Q}_i) - |\tilde{Q}_i| - d(\tilde{Q}_i, q) && \text{(triangle inequality)} \\ &\geq (1/2C)\delta^* - (7/A)\delta^* - (16/A)\delta^* && \text{(Lemmas 6.10(ii), 6.6(i), 6.9)} \\ &\geq (1/3C)\delta^*. && (1/C \gg 1/A) \end{aligned}$$

Thus, $q \neq q^*$, and q^* is the closest point of p' on the segment between $\tilde{Q}_i$ and $\tilde{Q}_{i+1}$. □

Case 2. Now suppose that $\tilde{P}_i$ (and $\tilde{Q}_i$) is composite. The argument is completely analogous to the first case, but with different bounds.

Lemma 6.12. *We have*

(i) *If* $p' \in \tilde{P}_i$*, then* $d(p', q^*) \leq (6/C)\delta^*$*;*

(ii) *If* $p' \notin \tilde{P}_i$*, then* $d(p', \tilde{Q}_i) \geq (1 - 5/C)\delta^*$*;*

(iii) *If* $q' \in \tilde{Q}_i$*, then* $d(p^*, q') \leq (3/C)\delta^*$*;*

(iv) *If* $q' \notin \tilde{Q}_i$*, then* $d(q', \tilde{P}_i) \geq (1 - 5/C)\delta^*$*.*

Proof. (i): If $p' \in \tilde{P}_i$, then

$$\begin{aligned} d(p', q^*) &\leq d(p', q) \leq d(p', \tilde{Q}_i) + d_Q(\tilde{Q}_i, q) && (q \text{ is on } qq', \text{ triangle inequality}) \\ &\leq (1/A)\delta^* + (5/C)\delta^* && (\text{Lemmas 6.3 and 6.9}) \\ &\leq (6/C)\delta^*. && (1/C \gg 1/A) \end{aligned}$$

(ii): If $p' \notin \tilde{P}_i$, then

$$\begin{aligned} d(p', \tilde{Q}_i) &\geq d(p', \tilde{P}_i) - d(\tilde{P}_i, \tilde{Q}_i) - |\tilde{Q}_i| && (\text{triangle inequality}) \\ &\geq (1 - 2/C)\delta^* - (1/A)\delta^* - (2/C)\delta^* && (\text{Lemmas 6.6(ii), 6.3}) \\ &\geq (1 - 5/C)\delta^*. && (1/C \gg 1/A) \end{aligned}$$

(iii): If $q' \in \tilde{Q}_i$, then

$$\begin{aligned} d(p^*, q') &\leq d(p, q') \leq |\tilde{P}_i| + d_Q(\tilde{P}_i, q') && (p \text{ on } pp', \text{ triangle inequality}) \\ &\leq (2/C)\delta^* + (1/A)\delta^* \leq (3/C)\delta^*. && (\text{Lemmas 6.6(ii),6.3}) \end{aligned}$$

(iv): If $q' \notin \tilde{Q}_i$, then

$$\begin{aligned} d(q', \tilde{P}_i, q') &\geq d(q', \tilde{Q}_i) - d(\tilde{Q}_i, \tilde{P}_i) - |\tilde{P}_i| && (\text{triangle inequality}) \\ &\geq (1 - 2/C)\delta^* - (1/A)\delta^* - (2/C)\delta^* && (\text{Lemmas 6.6(ii),6.3}) \\ &= (1 - 5/C)\delta^*. && (1/C \gg 1/A) \end{aligned}$$

□

Lemma 6.13. *Invariant 6.7 holds in the next intermediate step.*

Proof. If $p' \in \tilde{P}_i$ and $q' \in \tilde{Q}_i$, then Invariant 6.7 clearly holds in the next step (in particular, by Lemma 6.8, if $q^* \neq q$, then q^* is the closest point of p' on an edge incident to $\tilde{Q}_i$).
If $p' \in \tilde{P}_i$ and $q' \notin \tilde{Q}_i$, then

$$d(p', q^*) \leq (6/C)\delta^* \leq (1 - 5/C)\delta^* \leq d(\tilde{P}_i, q') \leq d(p^*, q'),$$

by Lemma 6.12(i,iv). Thus, the next intermediate position is (p', q^*), and if $q^* \neq q$, then q^* is the closest point of p' on an edge incident to $\tilde{Q}_i$, by Lemma 6.8.

If $p' \notin \tilde{P}_i$ and $q' \in \tilde{Q}_i$, then

$$d(p^*, q') \leq (3/C)\delta^* \leq (1 - 8/C)\delta^* \leq d(p', \tilde{Q}_i) - |\tilde{Q}_i| - d(\tilde{Q}_i, q^*) \leq d(p', q^*),$$

by Lemma 6.12(ii,iii), Lemma 6.6(ii) and Lemma 6.9. Thus, the next intermediate position is (p^*, q'), and p^* is the closest point of q' on an edge incident to $\tilde{P}_i$.

If $p' \notin \tilde{P}_i$ and $q' \notin \tilde{Q}_i$, then p' is the first vertex of $\tilde{P}_{i+1}$, q' is the first vertex of $\tilde{Q}_{i+1}$, p^* lies on the segment between $\tilde{P}_i$ and $\tilde{P}_{i+1}$, and q^* lies on the segment between $\tilde{P}_i$ and $\tilde{P}_{i+1}$. If the next intermediate position is (p^*, q'), then Invariant 6.7 clearly holds in the next step. If the next intermediate position is (p', q^*), it remains to argue that q^* is indeed the closest point of p' on the segment incident to $\tilde{Q}_i$ and $\tilde{Q}_{i+1}$. Since the optimal traversal β passes the segment between $\tilde{P}_i$ and $\tilde{P}_{i+1}$ and the segment between $\tilde{Q}_i$ and $\tilde{Q}_{i+1}$ together, we have

$$d(p', q^*) = \min\{d(p', q^*), d(p^*, q')\} \leq \delta_{\mathrm{F}} \leq (1/A)\delta^*,$$

by Lemma 6.1(i), whereas

$$\begin{aligned} d(p', q) &\geq d(p', \tilde{Q}_i) - |\tilde{Q}_i| - d(\tilde{Q}_i, q) \\ &\geq (1 - 5/C)\delta^* - (2/C)\delta^* - (5/C)\delta^* \\ &= (1 - 12/C)\delta^*, \end{aligned}$$

by Lemmas 6.12(ii), 6.6(ii), 6.9, and the triangle inequality. Thus, $q \neq q^*$, and q^* is the closest point of p' on the segment between $\tilde{Q}_i$ and $\tilde{Q}_{i+1}$. □

Conclusion.

Theorem 6.14. *The greedy algorithm computes a $2^{O(n)}$-approximation for the continuous Fréchet distance in $O(n)$ time.*

Proof. The running time follows by construction. Since the greedy algorithm moves uniformly between the intermediate positions, δ_{greedy} is the maximum distance of any intermediate position. We have $d(p_1, q_1) \leq \delta_{\mathrm{F}}$, and for all other intermediate positions, Invariant 6.7 holds by Lemmas 6.11 and 6.13. Now let (p, q) be an intermediate position, and suppose that p is a vertex of $\tilde{P}_i$, $i \in \{1, \dots, \tilde{k}\}$, and that q is the closest point of some vertex of P_i on an edge incident to $\tilde{Q}_i$. Then,

$$\begin{aligned} d(p, q) &\leq d(p, \tilde{Q}_i) + |\tilde{Q}_i| + d(\tilde{Q}_i, q) \\ &\leq (1/A)\delta^* + (2/C)\delta^* + (5/C)\delta^* = O(\delta^*) \end{aligned}$$

by Lemma 6.3, Lemma 6.6, and Lemma 6.9. The case that q is a vertex of $\tilde{Q}_i$ is analogous. Thus, by Lemma 6.1(iii), we have $\delta_{\text{greedy}} = O(\delta^*) = 2^{O(n)}\delta_{\mathrm{F}}$. □

7 Conclusions

We have obtained several new results on the approximability of the discrete Fréchet distance. As our main results,

1. we showed a conditional lower bound for the *one-dimensional* case that there is no 1.399-approximation in strongly subquadratic time unless the Strong Exponential Time Hypothesis fails. This sheds further light on what makes the Fréchet distance a difficult problem.

2. we determined the approximation ratio of the *greedy* algorithm as $2^{\Theta(n)}$ in any dimension $d \geq 1$. This gives the first general linear time approximation algorithm for the problem; and

3. we designed an α-*approximation* algorithm running in time $O(n \log n + n^2/\alpha)$ for any $1 \leq \alpha \leq n$ in any constant dimension $d \geq 1$. This significantly improves the greedy algorithm, at the expense of a (slightly) worse running time.

Our lower bounds exclude only (too good) constant factor approximations with strongly subquadratic running time, while our best strongly subquadratic approximation algorithm has an approximation ratio of n^{ε}. It remains a challenging open problem to close this gap.

References

[1] Amir Abboud and Virginia Vassilevska Williams. Popular conjectures imply strong lower bounds for dynamic problems. In *Proc. 55th Annu. IEEE Sympos. Found. Comput. Sci. (FOCS)*, pages 434–443, 2014.

[2] Amir Abboud, Virginia Vassilevska Williams, and Oren Weimann. Consequences of faster alignment of sequences. In *Proc. 41st Internat. Colloq. Automata Lang. Program. (ICALP)*, volume 8572 of *LNCS*, pages 39–51, 2014.

[3] Amir Abboud, Ryan Williams, and Huacheng Yu. More applications of the polynomial method to algorithm design. In *Proc. 26th Annu. ACM-SIAM Sympos. Discrete Algorithms (SODA)*, pages 218–230, 2015.

[4] Pankaj K. Agarwal, Rinat Ben Avraham, Haim Kaplan, and Micha Sharir. Computing the discrete Fréchet distance in subquadratic time. *SIAM J. Comput.*, 43(2):429–449, 2014.

[5] Helmut Alt. Personal communication. 2012.

[6] Helmut Alt and Michael Godau. Computing the Fréchet distance between two polygonal curves. *Internat. J. Comput. Geom. Appl.*, 5(1–2):78–99, 1995.

[7] Karl Bringmann. Why walking the dog takes time: Fréchet distance has no strongly subquadratic algorithms unless SETH fails. In *Proc. 55th Annu. IEEE Sympos. Found. Comput. Sci. (FOCS)*, pages 661–670, 2014.

[8] Karl Bringmann and Marvin Künnemann. Improved approximation for Fréchet distance on c-packed curves matching conditional lower bounds. `arXiv:1408.1340`, 2014.

[9] Kevin Buchin, Maike Buchin, Wouter Meulemans, and Wolfgang Mulzer. Four soviets walk the dog - with an application to Alt's conjecture. In *Proc. 25th Annu. ACM-SIAM Sympos. Discrete Algorithms (SODA)*, pages 1399–1413, 2014.

[10] Kevin Buchin, Maike Buchin, Rolf van Leusden, Wouter Meulemans, and Wolfgang Mulzer. Computing the Fréchet distance with a retractable leash. In *Proc. 21st Annu. European Sympos. Algorithms (ESA)*, pages 241–252, 2013.

[11] Paul B. Callahan and S. Rao Kosaraju. A decomposition of multidimensional point sets with applications to k-nearest-neighbors and n-body potential fields. *J. ACM*, 42(1):67–90, 1995.

[12] Thomas Eiter and Heikki Mannila. Computing Discrete Fréchet Distance. Technical Report CD-TR 94/64, Christian Doppler Laboratory, 1994.

[13] Anka Gajentaan and Mark H. Overmars. On a class of $O(n^2)$ problems in computational geometry. *Comput. Geom. Theory Appl.*, 5(3):165–185, 1995.

[14] Michael R. Garey and David S. Johnson. *Computers and intractability. A guide to the theory of NP-completeness.* W. H. Freeman, 1979.

[15] Allan Grønlund and Seth Pettie. Threesomes, degenerates, and love triangles. In *Proc. 55th Annu. IEEE Sympos. Found. Comput. Sci. (FOCS)*, pages 621–630, 2014.

[16] Russell Impagliazzo and Ramamohan Paturi. On the complexity of k-SAT. *J. Comput. System Sci.*, 62(2):367–375, 2001.

[17] Russell Impagliazzo, Ramamohan Paturi, and Francis Zane. Which problems have strongly exponential complexity. *J. Comput. System Sci.*, 63(4):512–530, 2001.

[18] Mihai Pătraşcu and Ryan Williams. On the possibility of faster SAT algorithms. In *Proc. 21st Annu. ACM-SIAM Sympos. Discrete Algorithms (SODA)*, pages 1065–1075, 2010.

[19] Ramamohan Paturi, Pavel Pudlák, Michael E. Saks, and Francis Zane. An improved exponential-time algorithm for k-sat. *J. ACM*, 52(3):337–364, 2005.

[20] Liam Roditty and Virginia Vassilevska Williams. Fast approximation algorithms for the diameter and radius of sparse graphs. In *Proc. 45th Annu. ACM Sympos. Theory Comput. (STOC)*, pages 515–524, 2013.

[21] Ryan Williams. A new algorithm for optimal 2-constraint satisfaction and its implications. *Theoret. Comput. Sci.*, 348(2):357–365, 2005.