POLYLINE SIMPLIFICATION HAS CUBIC COMPLEXITY

Karl Bringmann[*] *and Bhaskar Ray Chaudhury*[†]

Abstract. In the classic polyline simplification problem, given a polygonal curve P consisting of n vertices and an error threshold $\delta \geq 0$, we want to replace P by a subsequence Q of minimal size such that the distance between the polygonal curves P and Q is at most δ. The distance between curves is usually measured using the Hausdorff or continuous Fréchet distance. These distance measures can be applied *globally*, i.e., to the whole curves P and Q, or *locally*, i.e., to each simplified subcurve and the line segment that it was replaced with separately (and then taking the maximum). This gives rise to four problem variants: Global-Hausdorff simplification (known to be NP-hard), Local-Hausdorff simplification (can be solved in time $O(n^3)$), Global-Fréchet simplification (can be solved in time $O(kn^5)$, where k is the size of the optimum simplification), and Local-Fréchet simplification (can be solved in time $O(n^3)$).

Our contribution is as follows:

- *Cubic time for all variants:* For Global-Fréchet simplification, we design an algorithm running in time $O(n^3)$. This shows that all three problems (Local-Hausdorff, Local-Fréchet, and Global-Fréchet) can be solved in cubic time. All these algorithms work over a general metric space such as $(\mathbb{R}^d, L_p)$, but the hidden constant depends on p and (linearly) on d.
- *Cubic conditional lower bound:* We provide evidence that in high dimensions, cubic time is essentially optimal for all three problems (Local-Hausdorff, Local-Fréchet, and Global-Fréchet). Specifically, improving the cubic time to $O(n^{3-\epsilon}\text{poly}(d))$ for polyline simplification over $(\mathbb{R}^d, L_p)$ for $p = 1$ would violate plausible conjectures. We obtain similar results for all $p \in [1, \infty), p \neq 2$.

In total, in high dimensions and over general L_p-norms we resolve the complexity of polyline simplification with respect to Local-Hausdorff, Local-Fréchet, and Global-Fréchet, by a providing new algorithm and conditional lower bounds.

[*] *Saarland University, Max Planck Institute for Informatics, Saarland Informatics Campus*, bringmann@cs.uni-saarland.de

[†] *Max Planck Institute for Informatics, Saarbrücken Graduate School of Computer Science, Saarland Informatics Campus*, braycha@mpi-inf.mpg.de

1 Introduction

We revisit the classic problem of polygonal line simplification, which is fundamental to computational geometry, computer graphics, and geographic information systems. The most frequently implemented and cited algorithms for curve simplification go back to the 70s (Douglas and Peucker [13]) and 80s (Imai and Iri [20]). These algorithms use the following standard[1] formalization of curve simplification. A *polygonal curve* or *polyline* is given by a sequence $P = \langle v_0, v_1, \ldots, v_n \rangle$ of points $v_i \in \mathbb{R}^d$, and represents the continuous curve walking along the line sequences $\overline{v_i v_{i+1}}$ in order.

Given a polyline $P = \langle v_0, v_1, \ldots, v_n \rangle$ and a value $\delta > 0$, we want to compute a subsequence $Q = \langle v_{i_0}, \ldots, v_{i_{k-1}} \rangle$, with $0 = i_0 < \ldots < i_{k-1} = n$, of minimum length k such that P and Q have a "distance" at most δ.

Several distance measures have been used for the curve simplification problem. The most generic distance measure on two *point sets* A, B is the *Hausdorff distance* δ_H. The (directed) Hausdorff distance from A to B is the maximum over all $a \in A$ of the distance from a to its closest point in B. This is used on curves P, Q by applying it to the images of the curves in the ambient space, i.e., to the union of all line segments $\overline{v_i v_{i+1}}$.

However, the most popular distance measure for curves in computational geometry is the *Fréchet distance* δ_F. This is the minimal length of a leash connecting a dog to its owner as they continuously walk along the two polylines without backtracking. In comparison to the Hausdorff distance, it takes the ordering of the vertices along the curves into account, and thus better captures an intuitive notion of distance among curves.

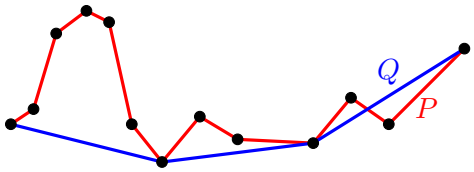
For both of these distance measures $\delta_* \in \{\delta_H, \delta_F\}$, we can apply them *locally* or *globally* in order to measure the distance between the original curve P and its simplification Q. In the global variant, we simply consider the distance $\delta_*(P, Q)$, i.e., we use the Hausdorff or Fréchet distance of P and Q. In the local variant, we consider the distance $\max_{1 \le \ell < k} \delta_*(P[i_{\ell-1} \ldots i_\ell], \overline{v_{i_{\ell-1}} v_{i_\ell}})$, i.e., for each simplified subcurve $P[i_{\ell-1} \ldots i_\ell] = \langle v_{i_{\ell-1}}, \ldots, v_{i_\ell} \rangle$ of P we compute the distance to the line segment $\overline{v_{i_{\ell-1}} v_{i_\ell}}$ that we simplified the subcurve to, and we take the maximum over these distances. This gives rise to four problem variants, depending on the distance measure: Local-Hausdorff, Local-Fréchet, Global-Hausdorff, and Global-Fréchet. See Section 2 for details.

Among these variants, Global-Hausdorff is unreasonable in that it essentially does not take the ordering of vertices along the curve into account. Moreover, it was recently shown that curve simplification under Global-Hausdorff is NP-hard [23]. For these reasons, we do not consider this measure in this paper.

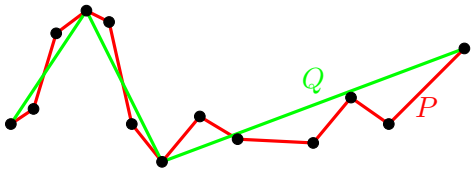
The classic algorithm by Imai and Iri [20] was designed for Local-Hausdorff simplification and solves this problem in time[2] $\mathcal{O}(n^3)$. By exchanging the distance computation in

[1] The problem was also studied without the restriction that vertices of the simplification belong to the original curve [18]. The choice whether the start- and endpoints of P and Q must coincide or not is typically irrelevant in this area; in this paper we assert that they coincide, but all results could also be proved without this assumption. Moreover, sometimes δ is given and k should be minimized, sometimes k is given and δ should be minimized; we focus on the former variant in this paper.

[2] Throughout this paper, in $\mathcal{O}$-notation we *hide any polynomial factors in d*, but we make exponential



(a) The simplification Q does not resemble P closely.



(b) The simplification Q resembles P closely.

Figure 1: Illustration of a good simplification. The simplifications in (a) and (b) have the same size. However the one in (b) represents P much more closely than the one in (a).

this algorithm for the Fréchet distance, one can obtain an $\mathcal{O}(n^3)$-time algorithm for Local-Fréchet [17]. Several papers obtained improvements for Local-Hausdorff simplification in small dimension d [22, 10, 7]; the fastest known running times are $2^{O(d)}n^2$ for L_1-norm, $\mathcal{O}(n^2)$ for L_∞-norm, and $\mathcal{O}(n^{3-\Omega(1/d)})$ for L_2-norm [7].

The remaining variant, Global-Fréchet, has only been studied very recently [23], although it is a reasonable measure: The local constraints (i.e., matching each v_{i_ℓ} to itself) are not necessary to enforce ordering along the curve, since Fréchet distance already takes the ordering of the vertices into account – in contrast to Hausdorff distance, for which the Local constraints are necessary to enforce any ordering. Therefore, Global-Fréchet simplification is also very natural and well motivated as Fréchet distance is a popular distance measure between curves in computational geometry. Van Kreveld et al. [23] presented an algorithm for Global-Fréchet simplification in time $\mathcal{O}(k^*n^5)$, where k^* is the output size, i.e., the size of the optimal simplification.

1.1 Contribution 1: Algorithm for Global-Fréchet

From the state of the art, one could get the impression that Global-Fréchet is a well-motivated, but computationally expensive curve simplification problem, in comparison to Local-Hausdorff and Local-Fréchet. We show that the latter intuition is wrong, by designing an $\mathcal{O}(n^3)$-time algorithm for Global-Fréchet simplification. This is an improvement by a factor $\Theta(k^*n^2)$ over the previous best algorithm by van Kreveld et al. [23].

Theorem 1.1 (Section 3)**.** *Global-Fréchet simplification can be solved in time* $\mathcal{O}(n^3)$.

This shows that all three problem variants (Local-Hausdorff, Local-Fréchet, and Global-Fréchet) can be solved in time $\mathcal{O}(n^3)$, and thus the choice of which problem variant to apply should not be made for computational reasons, at least in high dimensions.

factors in d explicit.

Our algorithm (as well as the algorithms for Local-Hausdorff and Local-Fréchet [20, 17]) works over a general metric space such as $\mathbb{R}^d$ with L_p-norm. The hidden constant depends on p, and has linear dependence on d (throughout this paper in $\mathcal{O}$-notation we hide polynomial factors in d). We assume the Real RAM model of computation, which allows us to perform exact distance computations, and to exactly solve equations of the form $\|x - a\|_p = r$ for given $a \in \mathbb{R}^d$, $r \geq 0$. See Section 3 for an overview of the algorithm.

1.2 Contribution 2: Conditional Lower Bound

Since all three variants can be solved in time $\mathcal{O}(n^3)$, the question arises whether any of them can be solved in time $\mathcal{O}(n^{3-\varepsilon})$. Tools to (conditionally) rule out such algorithms have been developed in recent years in the area of *fine-grained complexity*, see, e.g., the survey [24]. One of the most widely used fine-grained hypotheses is the following.

k-OV Hypothesis: *Problem:* Given sets $A_1, \ldots, A_k \subseteq \{0,1\}^d$ of size n, determine whether there exist vectors $a_1 \in A_1, \ldots, a_k \in A_k$ that are orthogonal, i.e., for each dimension $j \in [d]$ there is a vector $i \in [k]$ with $a_i[j] = 0$.
Hypothesis: For any $k \geq 2$ and $\varepsilon > 0$, the problem cannot be solved in time $\mathcal{O}(n^{k-\varepsilon})$.

Naively, k-OV can be solved in time $\mathcal{O}(n^k)$, and the hypothesis asserts that no polynomial improvement is possible, at least not with polynomial dependence on d. See [2] for the fastest known algorithms for k-OV.

Buchin et al. [8] used the 2-OV hypothesis to rule out $\mathcal{O}(n^{2-\varepsilon})$-time algorithms for Local-Hausdorff[3] in the L_1, L_2, and L_∞ norm. This yields a tight bound for L_∞, since an $\mathcal{O}(n^2)$-time algorithm is known [7]. However, for all other L_p-norms ($p \in [1, \infty)$), the question remained open whether $\mathcal{O}(n^{3-\varepsilon})$-time algorithms exist. To answer this question, one could try to generalize the conditional lower bound by Buchin et al. [8] to start from 3-OV. However, there is a barrier to such a reduction: As we show in Section 5, polyline simplification has fast nondeterministic and co-nondeterministic algorithms, while 3-OV is conjectured not to have both of these [9]. For similar reasons, Abboud et al. [3] introduced the Hitting Set hypothesis, in which they essentially consider a variant of 2-OV where we have a universal quantifier over the first set of vectors and an existential quantifier over the second one ($\forall\exists$-OV). From their hypothesis, however, it is not known how to prove higher lower bounds than quadratic. We therefore consider the following natural extension of their hypothesis. This problem was studied in a more general context by Gao et al. [16].

$\forall\forall\exists$-OV Hypothesis: *Problem:* Given sets $A, B, C \subseteq \{0,1\}^d$ of size n, determine whether for all $a \in A, b \in B$ there exists $c \in C$ such that $a, b, c,$ are orthogonal.
Hypothesis: For any $\varepsilon > 0$ the problem cannot be solved in time[4] $\mathcal{O}(n^{3-\varepsilon})$.

No algorithm violating this hypothesis is known, and even for much stronger hypotheses on variants of k-OV and Satisfiability, no such algorithms are known, see Section 6

[3] Their proof can be adapted to also work for Local-Fréchet and Global-Fréchet.
[4] Recall that in $\mathcal{O}$-notation we hide any polynomial dependence on d.

for details. This shows that the hypothesis is plausible, in addition to being a natural generalization of the hypothesis of Abboud et al. [3]. We establish a ∀∀∃-OV-based lower bound for curve simplification.

Theorem 1.2 (Section 4)**.** *Over* $(\mathbb{R}^d, L_p)$ *for any* $p \in [1, \infty)$ *with* $p \neq 2$*, Local-Hausdorff, Local-Fréchet, and Global-Fréchet simplification have no* $\mathcal{O}(n^{3-\varepsilon})$*-time algorithm for any* $\varepsilon > 0$*, unless the* ∀∀∃-OV *Hypothesis fails. This holds even for the problem of deciding whether the optimal simplification has size* ≤ 4 *or* ≥ 5*.*

In particular, this rules out improving the $2^{\mathcal{O}(d)}n^2$-time algorithm for Local-Hausdorff over L_1 [7] to a polynomial dependence on d. Note that the theorem statement excludes two interesting values for p, namely ∞ and 2. For $p = \infty$, an $\mathcal{O}(n^2)$-time algorithm is known for Local-Hausdorff [7], so proving the above theorem also for $p = \infty$ would immediately yield an algorithm breaking the ∀∀∃-OV Hypothesis. For $p = 2$, we do not have such a strong reason why it is excluded, however, we argue in Section 4 that at least a significantly different proof would be necessary in this case. This leaves open the possibility of a faster curve simplification algorithm for L_2, but such a result would need to exploit the Euclidean norm very heavily.

1.3 Further Related Work

Curve simplification has been studied in a variety of different formulations and settings, and it is well beyond the scope of this paper to give an overview. To list some examples, it was shown that the classic heuristic algorithm by Douglas and Peucker [13] can be implemented in time $\mathcal{O}(n \log n)$ [19], and that the classic $\mathcal{O}(n^3)$-time algorithm for Local-Hausdorff simplification by Imai and Iri [20] can be implemented in time $\mathcal{O}(n^2)$ in two dimensions [10, 22]. Further topics include curve simplification without self-intersections [12], Local-Hausdorff simplification with additional constraints on the angles between consecutive line segments [11], approximation algorithms [4], streaming algorithms [1], and the use of curve simplification in subdivision algorithms [18, 14, 15].

1.4 Organization

In Section 2 we formally define the problems studied in this paper. In Section 3 we present our new algorithm for Global-Fréchet simplification, and in Section 4 we show our conditional lower bounds. In Section 5, we provide an argument why the more common hypothesis, SETH, cannot be used (assuming NSETH) for showing our lower bounds in Section 4. We further discuss the used hypothesis in Section 6. Finally, in Section 7 we summarize the main results of the paper and mention some open problems for future research.

2 Preliminaries

Our ambient space is the metric space $(\mathbb{R}^d, L_p)$, where the distance between points $x, y \in \mathbb{R}^d$ is the L_p-norm of their difference, i.e., $\|x - y\|_p = \left(\sum_{i=1}^{d}(x[i] - y[i])^p\right)^{1/p}$. A *polyline* P of

size n is given by a sequence of points $\langle v_0, v_1, \ldots, v_n\rangle$, where each v_i lies in the ambient space. We associate with P the continuous curve that starts in v_0, walks along the line segments $\overline{v_i v_{i+1}}$ for $i = 0, \ldots, n-1$ in order, and ends in v_n. We also interpret P as a function $P\colon [0,n] \to \mathbb{R}^d$ where $P[i+\lambda] = (1-\lambda)v_i + \lambda v_{i+1}$ for any $\lambda \in [0,1]$ and $i \in \{0, \ldots, n-1\}$. We use the notation $P[t_1 \ldots t_2]$ to represent the sub-polyline of P between $P[t_1]$ and $P[t_2]$. Formally for any integers $0 \le i \le j \le n$ and reals $\lambda_1 \in [0,1)$ and $\lambda_2 \in (0,1]$,

$$P[i+\lambda_1 \ldots j+\lambda_2] = \langle (1-\lambda_1)v_i + \lambda_1 v_{i+1}, v_{i+1}, \ldots, v_j, (1-\lambda_2)v_j + \lambda_2 v_{j+1}\rangle$$

A *simplification* of P is a curve $Q = \langle v_{i_0}, v_{i_1}, \ldots, v_{i_m}\rangle$ with $0 = i_0 < i_1 < \ldots < i_m = n$. The size of the simplification Q is $m+1$. Our goal is to determine a simplification of small size that "closely" represents P (see Figure 1). To this end we define two popular measures of similarity between the curves, namely the Fréchet and Hausdorff distances.

Definition 2.1 (Fréchet distance)**.** *The (continuous) Fréchet distance $\delta_F(P_1, P_2)$ between two curves P_1 and P_2 of size n and m respectively is*

$$\delta_F(P_1, P_2) = \inf_f \max_{t\in[0,n]} \|P_1[t] - P_2[f(t)]\|_p$$

where $f\colon [0,n] \to [0,m]$ is continuous and monotone with $f(0) = 0$ and $f(n) = m$.

Alt and Godau [6] gave the characterization of the Fréchet distance in terms of the so-called free-space diagram.

Definition 2.2 (Free-Space)**.** *Given two curves P_1, P_2 and $\delta \ge 0$, the free-space $FS_\delta(P_1, P_2) \subseteq \mathbb{R}^2$ is the set $\{(x,y) \in ([0,m] \times [0,n]) \mid \|P_2[x] - P_1[y]\|_p \le \delta\}$.*

Consider the following decision problem. Given two curves P_1, P_2 of size n and m, respectively, and given $\delta \ge 0$, decide whether $\delta_F(P_1, P_2) \le \delta$. The answer to this question is yes if and only if (m,n) is reachable from $(0,0)$ by a monotone path through $FS_\delta(P_1, P_2)$. This "reachability" problem is known to be solvable by a dynamic programming algorithm in time $\mathcal{O}(nm)$, and the standard algorithm for computing the Fréchet distance is an adaptation of this decision algorithm [6]. In particular, if either P_1 or P_2 is a line segment, then the decision problem can be solved in linear time. Throughout the paper, whenever we construct a free space diagram between curves P_1 and P_2, we will have P_2 to be a line segment, i.e., $m = 1$. Therefore, we will always have the $FS_\delta(P_1, P_2) \subseteq [0,1] \times [0,n]$ where n is the size of the polyline P_1.

The Hausdorff distance between curves ignores the ordering of the points along the curve. Intuitively, if we remove the continuity and monotonicity condition from function f in Definition 2.1 we obtain the directed Hausdorff distance between the curves. Formally, it is defined as follows.

Definition 2.3 (Directed Hausdorff distance)**.** *The (directed)* Hausdorff *distance $\delta_H(P_1, P_2)$ between curves P_1 and P_2 of size n and m, respectively, is*

$$\delta_H(P_1, P_2) = \max_{t_1\in[0,n]} \min_{t_2\in[0,m]} \|P_1[t_1] - P_2[t_2]\|_p.$$

In order to measure the "closeness" between a curve and its simplification, these above similarity measures can be applied either *globally* to the whole curve and its simplification, or *locally* to each simplified subcurve $P[i_\ell \dots i_{\ell+1}]$ and the segment $\overline{v_{i_\ell}, v_{i_{\ell+1}}}$ to which it was simplified (taking the maximum over all ℓ). This gives rise to the following measures for curve simplification.

Definition 2.4 (Similarity for Curve Simplification)**.** *Given a curve $P = \langle v_0, v_1, \dots, v_n \rangle$ and a simplification $Q = \langle v_{i_0}, v_{i_1} \dots, v_{i_m} \rangle$ of P, we define their*

- *Global-Hausdorff distance as $\delta_H(P, Q)$,*
- *Global-Fréchet distance as $\delta_F(P, Q)$,*
- *Local-Hausdorff distance*[5] *as $\max_{0 \le \ell \le m-1} \delta_H(P[i_\ell \dots i_{\ell+1}], \overline{v_{i_\ell} v_{i_{\ell+1}}})$, and*
- *Local-Fréchet distance as $\max_{0 \le \ell \le m-1} \delta_F(P[i_\ell \dots i_{\ell+1}], \overline{v_{i_\ell} v_{i_{\ell+1}}})$.*

3 Algorithms for Global-Fréchet simplification

In this section we present an $\mathcal{O}(n^3)$ time algorithm for curve simplification under Global-Fréchet distance, i.e., we prove Theorem 1.1. We first give a brief overview of all the techniques and concepts used in this section.

Overview of the Algorithm. We first sketch the algorithm by Imai and Iri [20] for Local-Hausdorff simplification. Given a polyline $P = \langle v_0, \dots, v_n \rangle$ and a distance threshold δ, for all $i < i'$ we compute the Hausdorff distance between the subpolyline $P[i \dots i']$ and the line segment $\overline{v_i v_{i'}}$. This takes a total time of $\mathcal{O}(n^3)$, since the Hausdorff distance between a polyline and a line segment can be computed in linear time. We build a directed graph on vertices $\{0, 1, \dots, n\}$, with a directed edge from i to i' if and only if the Hausdorff distance between the subpolyline $P[i \dots i']$ and the line segment $\overline{v_i v_{i'}}$ is at most δ. We then determine the shortest path from 0 to n in this graph. This yields the simplification Q of smallest size, with Local-Hausdorff distance at most δ. The running time is dominated by the first step, and is thus $\mathcal{O}(n^3)$ time. Replacing the Hausdorff distance by Fréchet distance yields an $\mathcal{O}(n^3)$-time algorithm for Local-Fréchet simplification.

Note that these algorithms (mentioned in the paragraph above) are simple dynamic programming solutions. For Global-Fréchet, our cubic time algorithm also uses dynamic programming, but is significantly more non-trivial and involved.

In our algorithm, we compute the same dynamic programming table as the previously best algorithm [23]. This is a table of size $\mathcal{O}(k^* n^2)$, where k^* is the output size. The table entry $\mathrm{DP}(k, i, j)$ stores the earliest reachable point on the line segment $\overline{v_j v_{j+1}}$ with a size-k

[5] It can be checked that in this expression directed and undirected Hausdorff distance have the same value, and so for Local-Hausdorff we can without loss of generality use the directed Hausdorff distance. For Global-Hausdorff this choice makes a difference, but simplification under both directed and undirected Global-Hausdorff is NP-hard [23] and therefore we do not consider this problem in this paper.

simplification of $P[0 \dots i]$. More precisely, $\mathrm{DP}(k, i, j)$ is the minimal t, with $j \leq t \leq j+1$, such that there is a size-k simplification Q of $P[0 \dots i]$ with $\delta_F(Q, P[0 \dots t]) \leq \delta$. If such a point does not exist, we set $\mathrm{DP}(k, i, j) = \infty$.

A relatively simple algorithm computes a table entry in time $\mathcal{O}(n^3)$: We iterate over all possible second-to-last points $v_{i'}$ of the simplification Q, and over all possible previous line segments $\overline{v_{j'} v_{j'+1}}$, and check whether from i' on Q and $\mathrm{DP}(k-1, i', j')$ on P we can "walk" to i on Q and some $j \leq t \leq j+1$ on P, always staying within the required distance. Moreover, we compute the earliest such t (or equivalently the minimum such t). This can be done in time $O(n^3)$, which in total yields time $\mathcal{O}(k^* n^5)$. This is the algorithm from [23].

In order to obtain a speedup, we split the above procedure into two types: $j' = j$, i.e., the walks "coming from the left in the free-space diagram $FS_\delta(P, \overline{v_{i'} v_i})$" (see Figure 3), and $j' < j$, i.e., the walk "coming from the bottom the free-space diagram $FS_\delta(P, \overline{v_{i'} v_i})$" (see Figure 4). We maintain the table entry DP_1 to correctly handle the walks of the first kind. It can be seen that the simple algorithm computes their contribution to the output in time $\mathcal{O}(n)$. Moreover, it is easy to bring down this running time to $\mathcal{O}(1)$ per table entry, by maintaining a certain minimum.

We maintain table entries DP_2 to handle the walks of the second kind in total time $\mathcal{O}(n^3)$. This is the bulk of effort going into our new algorithm. Here, the main observation is that the particular values of $\mathrm{DP}(k-1, i', j')$ are irrelevant, and in particular we only need to store for each i', j' the smallest k' such that $\mathrm{DP}(k', i', j') \neq \infty$. Using this observation, and further massaging the problem, we arrive at the following subproblem that we call *Cell Reachability*: We are given n squares (or *cells*) numbered $1, \dots, n$ and stacked on top of each other. Between cell j and cell $j+1$ there is a *passage*, which is an interval on their common boundary through which we can pass from j to $j+1$. Finally, we are given an integral *entry-cost* λ_j for each cell j. The goal is to compute, for each cell j, its *exit-cost* μ_j, defined as the minimal entry-cost $\lambda_{j'}$, $j' < j$, such that we can walk from cell j' to cell j through the contiguous passages in a monotone fashion (i.e., the points at which we cross a passage are monotonically non-decreasing). See Figure 5 for an illustration of this problem.

To solve the Cell Reachability, we determine for each cell j and cost k the leftmost point $t_j(k)$ on the passage from cell $j-1$ to cell j at which we can arrive from some cell $j' < j$ with entry-cost at most k (using a monotone path). Among the sequence $t_j(1), t_j(2), \dots$ we only need to store the break-points, with $t_j(k) < t_j(k-1)$, and we design an algorithm to maintain these break-points in amortized time $\mathcal{O}(1)$ per cell j. This yields an $\mathcal{O}(n)$-time solution to Cell Reachability, which translates to an $\mathcal{O}(n^3)$-time solution for Global-Fréchet simplification.

We will now sketch the $\mathcal{O}(kn^5)$-time algorithm in [23], so that the reader is familiar with how we update the table DP. Thereafter, we present our $\mathcal{O}(n^3)$-time algorithm, where we show how to do update the table DP faster by the novel update procedure mentioned above.

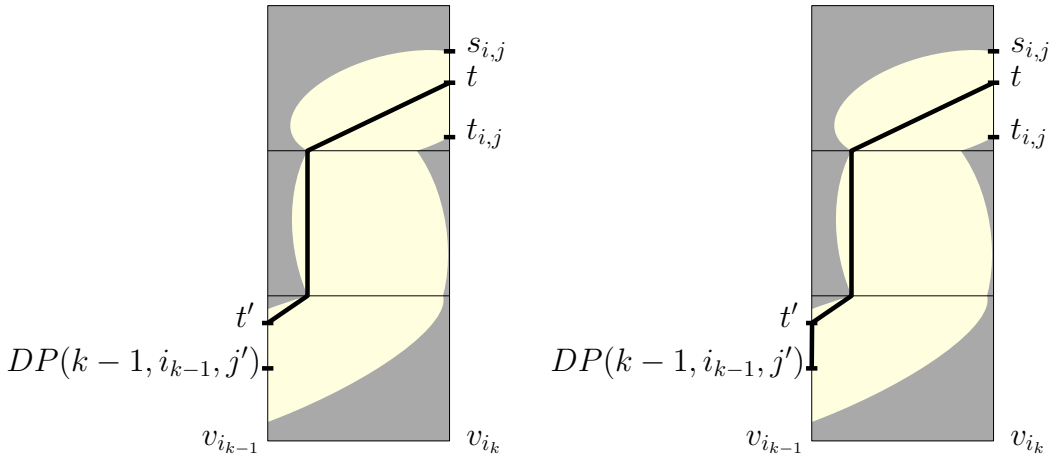


Figure 2: Illustration of the proof of Lemma 3.2. There exists a monotone path from $(0, t')$ to $(1, t)$ in $FS_\delta(P, \overline{v_{i_{k-1}} v_{i_k}})$ (left). Since $\mathrm{DP}(k-1, i_{k-1}, j') \leq t' \leq t$, there is a monotone path in $FS_\delta(P, \overline{v_{i_{k-1}} v_{i_k}})$ from $(0, \mathrm{DP}(k-1, i_{k-1}, j'))$ (right) to $(1, t)$ by moving from $(0, \mathrm{DP}(k-1, i_{k-1}, j'))$ to $(0, t')$ and then following the existing monotone path from $(0, t')$ to $(1, t)$.

3.1 An $O(kn^5)$ algorithm for Global Fréchet simplification

We start by describing the previously best algorithm by [23]. Let P be the polyline $\langle v_0, v_1, \ldots v_n \rangle$. Let $\mathrm{DP}(k, i, j)$ represent the earliest reachable point on $\overline{v_j v_{j+1}}$ with a size-k simplification of the polyline $P[0 \ldots i]$, i.e., $\mathrm{DP}(k, i, j)$ represents the smallest t such that $P[t]$ lies on the line-segment $\overline{v_j v_{j+1}}$ (i.e. $j \leq t \leq j+1$) and there is a simplification $\tilde{Q}$ of the polyline $P[0 \ldots i]$ of size at most k such that $\delta_F(\tilde{Q}, P[0 \ldots t]) \leq \delta$. If such a point does not exist then we set $\mathrm{DP}(k, i, j) = \infty$. To solve Global-Fréchet simplification, we need to return the minimum k such that $\mathrm{DP}(k, n, n-1) \neq \infty$. Let $P[t_{i,j}]$ and $P[s_{i,j}]$ be the first point and the last point respectively on the line segment $\overline{v_j v_{j+1}}$ such that $\|v_i - P[t_{i,j}]\|_p \leq \delta$ and $\|v_i - P[s_{i,j}]\|_p \leq \delta$. Observe that if $\mathrm{DP}(k, i, j) \neq \infty$ then $t_{i,j} \leq \mathrm{DP}(k, i, j) \leq s_{i,j}$ for all k. Before moving on to the algorithm we make some simple observations,

Observation 3.1. *If* $\mathrm{DP}(k, i, j) = \infty$ *then* $\mathrm{DP}(k', i, j) = \infty$ *for all* $k' < k$. *If* $\mathrm{DP}(k, i, j) = t_{i,j}$ *then* $\mathrm{DP}(k', i, j) = t_{i,j}$ *for all* $k' \geq k$.

Proof. If $k' < k$, then the minimization in $\mathrm{DP}(k, i, j)$ is over a superset compared to $\mathrm{DP}(k', i, j)$. Thus, $\mathrm{DP}(k', i, j) \geq \mathrm{DP}(k, i, j) = \infty$. Thus, $\mathrm{DP}(k, i, j) = \infty$. Similarly when $k' \geq k$ then the minimization in $\mathrm{DP}(k', i, j)$ is over a superset compared to $\mathrm{DP}(k, i, j)$. Thus, we have $t_{i,j} \leq \mathrm{DP}(k', i, j) \leq \mathrm{DP}(k, i, j)$. Thus, $\mathrm{DP}(k, i, j) = t_{i,j}$ implies $\mathrm{DP}(k', i, j) = t_{i,j}$ for all $k' \geq k$. □

We will crucially make use of the following characterization of the DP table entries.

Lemma 3.2. $\mathrm{DP}(k,i,j)$ *is the minimal* $t \in [t_{i,j}, s_{i,j}]$*, such that for some* $i' < i$ *and* $j' \leq j$*, we have* $\mathrm{DP}(k-1,i',j') \neq \infty$ *and* $\delta_F(P[\mathrm{DP}(k-1,i',j')\dots t], \overline{v_{i'}v_i}) \leq \delta$*. If no such* t *exists then* $\mathrm{DP}(k,i,j) = \infty$*.*

Proof. Let t be minimal in $[t_{i,j}, s_{i,j}]$ such that $\mathrm{DP}(k-1,i',j') \neq \infty$ and $\delta_F(P[\mathrm{DP}(k-1,i',j')\dots t], \overline{v_{i'}v_i}) \leq \delta$ for some $i' < i$ and $j' \leq j$. Since in particular $\mathrm{DP}(k-1,i',j') \neq \infty$, for one direction we note that there exists a simplification $\hat{Q}$ of the polyline $P[0\dots i']$ of size $k-1$ such that $\delta_F(\hat{Q}, P[0\dots \mathrm{DP}(k-1,i',j')]) \leq \delta$. By appending v_j to $\hat{Q}$ we obtain a simplification $\tilde{Q}$ of the polyline $P[0\dots i]$ such that $\delta_F(\tilde{Q}, P[0\dots \hat{t}]) \leq \max(\delta_F(\hat{Q}, P[0\dots \mathrm{DP}(k-1,i',j')]), \delta_F(P[\mathrm{DP}(k-1,i',j')\dots t], \overline{v_{i'}v_i})) \leq \delta$. It follows that $\mathrm{DP}(k,i,j) \leq t$. In particular if $\mathrm{DP}(k,i,j) = \infty$ then no such t exists.

For the other direction, let t' be such that $\mathrm{DP}(k,i,j) = t'$. Assume $t' \neq \infty$. Then, there exists a simplification $\tilde{Q} = \langle v_{i_0}, v_{i_1}, \dots v_{i_{k-1}}, v_{i_k}\rangle$ of the polyline $P[0\dots i]$ such that $\delta_F(\tilde{Q}, P[0\dots t']) \leq \infty$. Such a $\tilde{Q}$ exists if and only if there is a simplification $\hat{Q}$ of size $k-1$ of the polyline $\hat{P} = P[0\dots i_{k-1}]$ and a value $\hat{t} \leq t'$ such that,

(1) $\delta_F(\hat{Q}, P[0\dots \hat{t}]) \leq \delta$ and

(2) $\delta_F(P[\hat{t}\dots t'], \overline{v_{i_{k-1}}v_{i_k}}) \leq \delta$.

Let $\hat{j} \leq \hat{t} \leq \hat{j}+1$. Observe that (1) implies that $\mathrm{DP}(k-1,i_{k-1},\hat{j}) \neq \infty$. Also $t_{i_{k-1},\hat{j}} \leq \mathrm{DP}(k-1,i_{k-1},\hat{j}) \leq \hat{t} \leq s_{i_{k-1},\hat{j}}$. Now we show that $\delta_F(P[\hat{t}\dots t'], \overline{v_{i_{k-1}}v_{i_k}}) \leq \delta$ implies that $\delta_F(P[\mathrm{DP}(k-1,i_{k-1},\hat{j})\dots t'], \overline{v_{i_{k-1}}v_{i_k}}) \leq \delta$. This is obvious from inspecting $FS_\delta(P, \overline{v_{i_{k-1}}v_{i_k}})$ (see Figure 2). There exists a monotone path in $FS_\delta(P, \overline{v_{i_{k-1}}v_{i_k}})$ that starts from $(0, \mathrm{DP}(k-1,i_{k-1},\hat{j}))$, moves to $(0,\hat{t})$ and then follows the monotone path from $(0,\hat{t})$ to $(1,t')$ that exists. Therefore, $t \leq t' = \mathrm{DP}(k,i,j)$. Combining the two inequalities we have that $\mathrm{DP}(k,i,j) = t$. □

A dynamic programming algorithm follows more or less directly from Lemma 3.2. Note that for fixed $i' < i$ and $j' \leq j$ such that $\mathrm{DP}(k-1,i',j') \neq \infty$ we can determine the minimal t such that $(1,t)$ is reachable from $(0, \mathrm{DP}(k-1,i',j'))$ by a monotone path in $FS_\delta(P, \overline{v_{i'}v_i})$ in $\mathcal{O}(n)$ time. This follows from the standard algorithm for the decision version of the Fréchet distance between two polygonal curves of length at most n (in particular here one of the curves is of length 1). To determine $\mathrm{DP}(k,i,j)$ we enumerate over all $i' < i$ and $j' \leq j$ such that $\mathrm{DP}(k-1,i',j') \neq \infty$ and determine the minimum t that is reachable. The running time to determine $\mathrm{DP}(k,i,j)$ is thus $\mathcal{O}(n^3)$ by the loops for i', j' and the Fréchet distance check. Since there are $\mathcal{O}(kn^2)$ DP-cells to fill, the algorithm runs in total time $\mathcal{O}(kn^5)$ and uses space $\mathcal{O}(kn^2)$.

3.2 An $\mathcal{O}(n^3)$ algorithm for Global-Fréchet simplification

Now we improve the running time by a more careful understanding of the monotone paths through $FS_\delta(P, \overline{v_{i'}v_i})$ to $(1, \mathrm{DP}(k,i,j))$ for fixed i, j and i'. Let fbox_j denote the intersection of the free-space $FS_\delta(P, \overline{v_{i'}v_i})$ with the square with corner vertices $(0,j)$ and $(1,j+1)$. The following fact will be useful later.

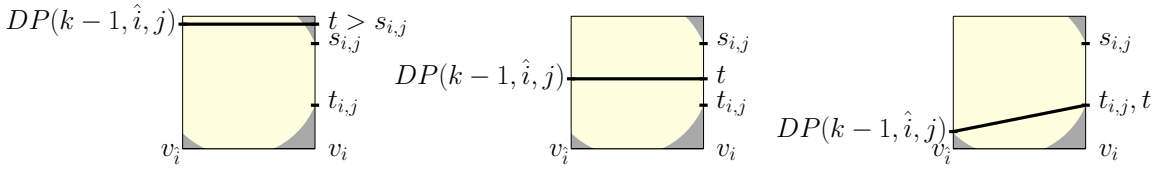


Figure 3: Illustration of the proof of Observation 3.4. For some $\hat{i} < i$, $t \in [t_{i,j}, s_{i,j}]$ is minimal such that $(1,t)$ is reachable from $(0, \mathrm{DP}(k-1,\hat{i},j))$ by a monotone path in $fbox_j$. If $\mathrm{DP}(k-1,\hat{i},j) > s_{i,j}$ (left) then no such t exists. If $t_{i,j} \le \mathrm{DP}(k-1,\hat{i},j) \le s_{i,j}$ (middle) then $t = \mathrm{DP}(k-1,\hat{i},j)$. If $\mathrm{DP}(k-1,\hat{i},j) < t_{i,j}$ (right) then $t = t_{i,j}$.

Fact 3.3. *$fbox_j$ is convex for all $j \in [n-1]$.*

Proof. Alt and Godau [6] showed that $fbox_j$ is an affine transformation of the unit ball, and this is convex for any L_p norm. □

Furthermore let ver_j denote the free space along vertical line segment with endpoints $(0,j)$ and $(0,j+1)$ and let hor_j denote the free space along the horizontal line segment $(0,j)$ to $(1,j)$ in the free space $FS_\delta(P, \overline{v_{i'}v_i})$. We consider the point $(0,j)$ to belong to ver_j, but not hor_j, to avoid certain corner cases. We split the monotone paths from $(0, \mathrm{DP}(k-1,i',j'))$ for $i' < i$ and $j' \le j$ to $(1, \mathrm{DP}(k,i,j))$ in $FS_\delta(P, \overline{v_{i'}v_i})$ into two categories: the ones that intersect ver_j and the ones that intersect hor_j. We first look at the monotone paths that intersect ver_j. Observe that if the monotone path intersects ver_j then $j' = j$. Let $\overline{\mathrm{DP}}_1(k,i,j) = \min_{i'<i} \mathrm{DP}(k-1,i',j)$. We now define,

$$\mathrm{DP}_1(k,i,j) = \begin{cases} \max(\overline{\mathrm{DP}}_1(k,i,j), t_{i,j}) & \text{if } \overline{\mathrm{DP}}_1(k,i,j) \le s_{i,j} \\ \infty & \text{otherwise} \end{cases} \tag{1}$$

We show a characterization of DP_1 similar to the characterization of DP in Lemma 3.2 and thus establishing that DP_1 correctly handles all paths intersecting ver_j.

Observation 3.4. *$\mathrm{DP}_1(k,i,j)$ is the minimal $t \in [t_{i,j}, s_{i,j}]$ such that $\mathrm{DP}(k-1,i',j) \neq \infty$ and $\delta_F(P[\mathrm{DP}(k-1,i',j) \ldots t], \overline{v_{i'}v_i}) \le \delta$ for some $i' < i$. If no such t exists then $\mathrm{DP}_1(k,i,j) = \infty$.*

Proof. Fix $\hat{i} < i$. First note that if there is a monotone path connecting $(0, \mathrm{DP}(k-1,\hat{i},j))$ to $(1,t)$ then $t \ge \mathrm{DP}(k-1,\hat{i},j)$. Now consider $fbox_j$ in the free-space $FS_\delta(P, \overline{v_{\hat{i}}v_i})$. As illustrated in Figure 3 there are three cases,

- If $\mathrm{DP}(k-1,\hat{i},j) > s_{i,j}$ then there is no monotone path from $(0, \mathrm{DP}(k-1,\hat{i},j))$ to $(1,t)$ for all $t \in [t_{i,j}, s_{i,j}]$.
- If $t_{i,j} \le \mathrm{DP}(k-1,\hat{i},j) \le s_{i,j}$. As mentioned at the beginning of the proof, $t \ge \mathrm{DP}(k-1,\hat{i},j)$. Since $fbox_j$ is convex, the line segment connecting $(0, \mathrm{DP}(k-1,\hat{i},j)$ and $(1, \mathrm{DP}(k-1,\hat{i},j))$ lies inside $fbox_j$ and hence lies inside $FS_\delta(P, \overline{v_{\hat{i}}v_i})$. Thus, the smallest $t \in [t_{i,j}, s_{i,j}]$ such that there is a monotone path from $(0, \mathrm{DP}(k-1,\hat{i},j))$ to $(1,t)$ in $FS_\delta(P, \overline{v_{\hat{i}}v_i})$ is $\mathrm{DP}(k-1,\hat{i},j)$.

- If $\mathrm{DP}(k-1,\hat{i},j) \leq t_{i,j}$. Again since $fbox_j$ is convex the line segment connecting $(0,\mathrm{DP}(k-1,\hat{i},j))$ and $(1,t_{i,j})$ lies inside $fbox_j$ and thus lies inside $FS_\delta(P,\overline{v_{\hat{i}}v_i})$. Thus, the smallest $t \in [t_{i,j},s_{i,j}]$ such that there is a monotone path from $(0,\mathrm{DP}(k-1,\hat{i},j))$ to $(1,t)$ in $FS_\delta(P,\overline{v_{\hat{i}}v_i})$ is $t_{i,j}$.

Therefore, for any $\hat{i} < i$ if $\mathrm{DP}(k-1,\hat{i},j) > s_{i,j}$ then there exists no $t \in [t_{i,j},s_{i,j}]$ such that $\delta_F(P[\mathrm{DP}(k-1,\hat{i},j)\ldots t],\overline{v_{\hat{i}}v_i}) \leq \delta$. Similarly, if $\mathrm{DP}(k-1,\hat{i},j) \leq s_{i,j}$, then the minimal $t \in [t_{i,j},s_{i,j}]$ such that $\delta_F(P[\mathrm{DP}(k-1,\hat{i},j)\ldots t],\overline{v_{\hat{i}}v_i}) \leq \delta$ is $\max(\mathrm{DP}(k-1,\hat{i},j),t_{i,j})$. Now let $t \in [t_{i,j},s_{i,j}]$ be minimal such that $\mathrm{DP}(k-1,i',j) \neq \infty$ and $\delta_F(P[\mathrm{DP}(k-1,i',j)\ldots t],\overline{v_{i'}v_i}) \leq \delta$ for some $i' < i$. It follows that if $\min_{i'<i}\mathrm{DP}(k-1,i',j) \leq s_{i,j}$, then $t = \max(\min_{i'<i}\mathrm{DP}(k-1,i',j),t_{i,j})$ and if $\min_{i'<i}\mathrm{DP}(k-1,i',j) > s_{i,j}$, then no such t exists. Since $\min_{i'<i}\mathrm{DP}(k-1,i',j) = \overline{\mathrm{DP}}_1(k,i,j)$ and $\mathrm{DP}_1(k,i,j) = \max(\overline{\mathrm{DP}}_1(k,i,j),t_{i,j})$ when $\overline{DP}_1(k,i,j) \leq s_{i,j}$ (by definition), we have that when $\overline{\mathrm{DP}}_1(k,i,j) \leq s_{i,j}$, $\mathrm{DP}_1(k,i,j) = \max(\overline{\mathrm{DP}}_1(k,i,j),t_{i,j}) = \max(\min_{i'<i}\mathrm{DP}(k-1,i',j),t_{i,j}) = t$. Similarly when $\overline{\mathrm{DP}}_1(k,i,j) > s_{i,j}$, then $DP(k,i,j) = \infty$ and t does not exist. □

We now look at the monotone paths that intersect hor_j. Observe that if the monotone path intersects hor_j then $j' < j$. Along this line, we define $\overline{\mathrm{DP}}_2(k,i,j) = 1$ if there exists some $i' < i$ and $j' < j$, such that $\mathrm{DP}(k-1,i',j') \neq \infty$ and there exists a monotone path from $(0,\mathrm{DP}(k-1,i',j'))$ to $(1,t_{i,j})$ in the free-space $FS_\delta(P,\overline{v_{i'}v_i})$ and otherwise we set $\overline{\mathrm{DP}}_2(k,i,j) = 0$. Hereafter we define,

$$\mathrm{DP}_2(k,i,j) = \begin{cases} t_{i,j} & \text{if } \overline{\mathrm{DP}}_2(k,i,j) = 1 \\ \infty & \text{otherwise} \end{cases}$$

We show a characterization of DP_2, similar to our characterization of DP in Lemma 3.2, thus establishing that DP_2 correctly handles all paths intersecting hor_j.

Observation 3.5. *$\mathrm{DP}_2(k,i,j)$ is the minimal $t \in [t_{i,j},s_{i,j}]$ such that $\mathrm{DP}(k-1,i',j') \neq \infty$ and $\delta_F(P[\mathrm{DP}(k-1,i',j')\ldots t],\overline{v_{i'}v_i}) \leq \delta$ for some $i' < i$ and $j' < j$. If no such t exists then $\mathrm{DP}_2(k,i,j) = \infty$.*

Proof. Let $t \in [t_{i,j},s_{i,j}]$ be minimal such that $\mathrm{DP}(k-1,i',j') \neq \infty$ and $\delta_F(P[\mathrm{DP}(k-1,i',j')\ldots t],\overline{v_{i'}v_i}) \leq \delta$ for some $i' < i$ and $j' < j$. If such t exists then $\overline{DP}_2(k,i,j) = 1$. Observe that for any $i' < i$ and $j' < j$, if there is a monotone path from $(0,\mathrm{DP}(k-1,i',j'))$ to $(1,t)$ in $FS_\delta(P,\overline{v_{i'}v_i})$, then the path intersects hor_j (at say z). Since $fbox_j$ is convex, the line segment connecting z and $(1,t_{i,j})$ lies inside $fbox_j$ and hence inside $FS_\delta(P,\overline{v_{i'}v_i})$. Thus, there is a monotone path from $(0,\mathrm{DP}(k-1,i',j'))$ to $(1,t_{i,j})$ in $FS_\delta(P,\overline{v_{i'}v_i})$ following the monotone path from $(0,\mathrm{DP}(k-1,i',j'))$ to z and then from z to $(1,t_{i,j})$ (see Figure 4). Since $t \geq t_{i,j}$ and is minimal, we have $t = t_{i,j} = \mathrm{DP}_2(k,i,j)$. Similarly if such t does not exist, then $\overline{\mathrm{DP}}_2(k,i,j) = 0$ and $\mathrm{DP}_2(k,i,j) = \infty$. □

Lemma 3.6. $\mathrm{DP}(k,i,j) = \min(\mathrm{DP}_1(k,i,j),\mathrm{DP}_2(k,i,j))$.

Proof. Follows directly from Lemma 3.2, 3.4, and 3.5. □

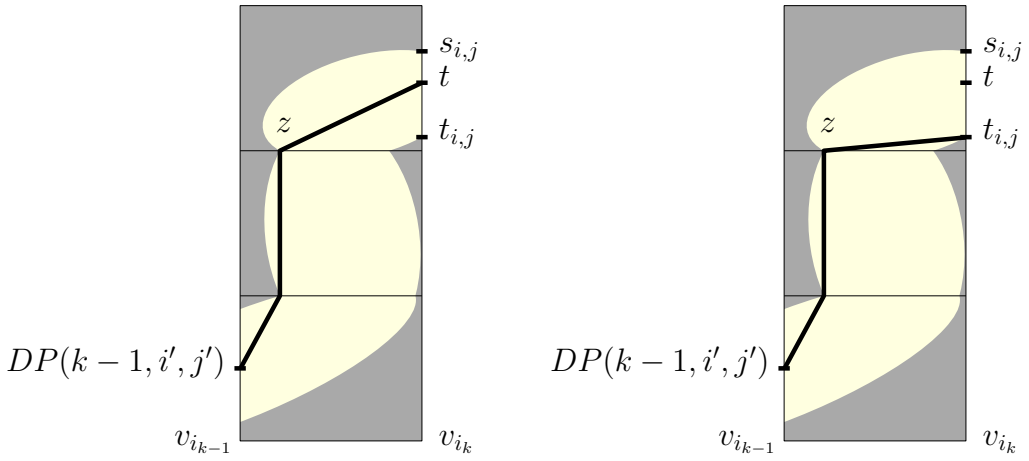


Figure 4: Illustration of the proof of Observation 3.5. For $t_{i,j} \leq t \leq s_{i,j}$, there is a monotone path from $(0, \mathrm{DP}(k-1,i',j'))$ to $(1,t)$ in the free-space $FS_\delta(P, \overline{v_{i'}v_i})$ (left) for some $i' < i$ and $j' < j$ that intersect hor_j at z. Then, there is also a monotone path from $(0, \mathrm{DP}(k-1,i',j'))$ to $(1,t_{i,j})$ (right) in the free-space $FS_\delta(P, \overline{v_{i'}v_i})$ following the same monotone path from $(0, \mathrm{DP}(k-1,i',j')$ to z and then from z to $(1,t_{i,j})$.

In particular this yields a dynamic programming formulation for $\mathrm{DP}(k,i,j)$, since both $\mathrm{DP}_1(k,i,j)$ and $\mathrm{DP}_2(k,i,j)$ depend on values of $\mathrm{DP}(k',i',j')$ with $k' < k$, $i' < i$ and $j' \leq j$.

We define $\kappa(i,j)$ as the minimal k such that $\mathrm{DP}(k,i,j) \neq \infty$. Similarly we define $\kappa_1(i,j)$ and $\kappa_2(i,j)$ as the minimal k such that $\mathrm{DP}_1(k,i,j) \neq \infty$ and $\mathrm{DP}_2(k,i,j) \neq \infty$ respectively. Note that $\kappa(i,j) = \min(\kappa_1(i,j), \kappa_2(i,j))$ (by Lemma 3.6). Also note that both $\kappa_1(i,j)$, and $\kappa_2(i,j)$ depends only on the values of $\mathrm{DP}(k',i',j')$ with $k' < k$, $i' < i$ and $j' \leq j$.

With these preparations, we can now present our dynamic programming algorithm, except for one subroutine κ_2-*subroutine*(i) that we describe in Section 3.3. In particular, for any i, κ_2-subroutine(i) determines $\kappa_2(i,j)$ for all $j \in [n]$ in time $T(n)$ only using the values of $\kappa(i',j)$ for all $i' < i$ and all $0 \leq j \leq n-1$. Now we show how to compute $\mathrm{DP}_1(k,i,j)$. Observe that for any i, j and k we can compute $\overline{\mathrm{DP}}_1(k,i,j)$ from $\overline{\mathrm{DP}}_1(k,i-1,j)$ and $\mathrm{DP}(k-1,i-1,j)$ as $\overline{\mathrm{DP}}_1(k,i,j) = \min(\overline{\mathrm{DP}}_1(k,i-1,j), \mathrm{DP}(k-1,i-1,j))$. Then, we can compute $\mathrm{DP}_1(k,i,j)$ using the formulation in 1 and set $\kappa_1(i,j)$ to the minimal k such that $\mathrm{DP}_1(k,i,j) \neq \infty$. This shows that we determine $\overline{\mathrm{DP}}_1(k,i,j)$ and $\kappa_1(i,j)$ in $\mathcal{O}(1)$ and $\mathcal{O}(n)$ time, respectively. Now we show how to compute $\mathrm{DP}_2(k,i,j)$. Notice that $\mathrm{DP}_2(k,i,j) = t_{i,j}$ if and only if $k \geq \kappa_2(i,j)$ and $\mathrm{DP}_2(k,i,j) = \infty$ otherwise. Also, we can set $\kappa(i,j)$ as $\min(\kappa_1(i,j), \kappa_2(i,j))$. Hence, we can determine $\mathrm{DP}_2(k,i,j)$ and $\kappa(i,j)$ in $\mathcal{O}(1)$ time. After that, we can also compute $\mathrm{DP}(k,i,j)$ by the formulation in Lemma 3.6 in $\mathcal{O}(1)$ time.

Algorithm 1 Solving curve simplification under Global-Fréchet distance

1: **Determine** $t_{i,j}$ and $s_{i,j}$ for all $0 \leq i \leq n$ and $0 \leq j \leq n-1$
2: **Determine** the largest j_0 such that $\|v_0 - v_j\|_p \leq \delta$ for all $j \leq j_0$
3: **Set** $\overline{\mathrm{DP}}_1(k,0,j), \mathrm{DP}(k,0,j)$ to 0 for all $j \leq j_0$ and to ∞ otherwise (for all $k \in [n+1]$)
4: **Set** $\kappa(0,j)$ to 1 for all $j \leq j_0$ and to ∞ otherwise
5: **Set** $\mathrm{DP}(0,i,j)$ to ∞ for all $i,j \in [n]$
6: **for all** $i = 1$ to n **do**
7: *Determine* $\kappa_2(i,j)$ *for all* $0 \leq j \leq n-1$ *using* κ_2*-subroutine*(i)
8: **for** $j = 0$ to $n-1$ **do**
9: **for** $k = 1$ to $n+1$ **do**
10: **Set** $\overline{\mathrm{DP}}_1(k,i,j)$ to $\min(\overline{\mathrm{DP}}_1(k,i-1,j), \mathrm{DP}(k-1,i-1,j))$
11: **Set** $\mathrm{DP}_1(k,i,j)$ to $\max(\overline{\mathrm{DP}}_1(k,i,j), t_{i,j})$ if $\overline{\mathrm{DP}}_1(k,i,j) \leq s_{i,j}$ and to ∞ otherwise
12: **Set** $\kappa_1(i,j)$ to the smallest k such that $\mathrm{DP}_1(k,i,j) \neq \infty$
13: **Set** $\kappa(i,j) = \min(\kappa_1(i,j), \kappa_2(i,j))$
14: **for** $k = 1$ to $n+1$ **do**
15: **Set** $\mathrm{DP}_2(k,i,j)$ to $t_{i,j}$ if $k \geq \kappa_2(i,j)$ and to ∞ otherwise
16: **Set** $\mathrm{DP}(k,i,j)$ to $\min(\mathrm{DP}_1(k,i,j), \mathrm{DP}_2(k,i,j))$
17: **Return** $\kappa(n,n-1)$

Algorithm 1 takes $\mathcal{O}(n \cdot T(n))$ time for determining $\kappa_2(i,j)$ for all $i,j \in [n]$. The time taken to compute $\kappa_1(i,j)$ and $\kappa(i,j)$ is $\mathcal{O}(n)$ and $\mathcal{O}(1)$ respectively. All the DP cells are updated in $\mathcal{O}(1)$ time. Since there are $\mathcal{O}(n^2)$ κ cells and $\mathcal{O}(n^3)$ DP cells, the total running time of our algorithm is $\mathcal{O}(n^3 + n \cdot T(n))$.

3.3 Implementing the κ_2-subroutine(i)

In this subsection we show how to implement step 7 of Algorithm 1 in time $T(n) = \mathcal{O}(n^2)$. Then, in total we have $\mathcal{O}(n^3)$ for solving Global-Fréchet simplification.

3.3.1 Cell Reachability

We introduce an auxiliary problem that we call *Cell Reachability*. We shall see later that an $\mathcal{O}(n)$ time solution to this problem ensures that the κ_2-subroutine(i) can be implemented in time $T(n) = \mathcal{O}(n^2)$.

Definition 3.7. *In an instance of the Cell Reachability problem, we are given*

- *A set of* n cells. *Each cell* j *with* $1 \leq j \leq n$ *is a unit square with corner points* $(0,j)$ *and* $(1,j+1)$. *We say that cells* j *and* $j+1$ *are* consecutive.
- *An integral* entry-cost $\lambda_j > 0$ *for every cell* j.
- *A set of* $n-1$ passages *between consecutive cells. The passage* p_j *is the horizontal line segment with endpoints* (j,a_j) *and* (j,b_j) *where* $b_j > a_j$.

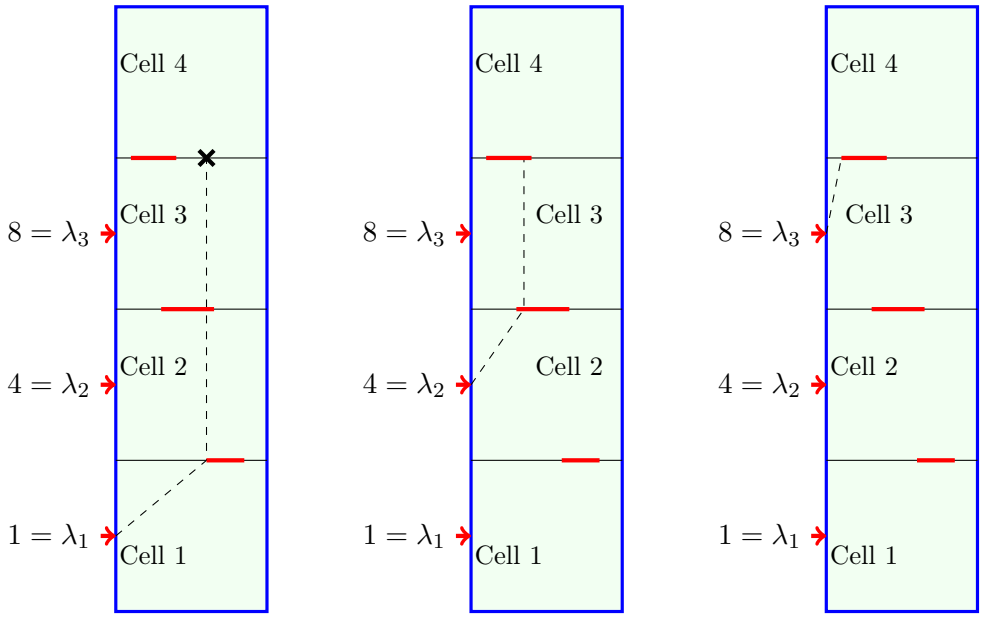


Figure 5: Illustrating an instance of Cell Reachability. The red horizontal line segments between the cells indicate the passages. Note that cell 4 is only reachable from cells 2 and 3. therefore $\mu_4 = \min(\lambda_2, \lambda_3) = \min(4, 8) = 4$.

We say that cell j is reachable *from cell j' with $j' < j$ if and only if there exists $x_{j'+1} \leq x_{j'+2} \ldots \leq x_j$ such that $x_k \in [a_k, b_k]$ for every $j' < k \leq j$. Intuitively, cell j is reachable from cell j' if and only if there is a monotone path through the passages from cell j' to cell j. We define the* exit-cost *μ_j of cell j as the minimal $\lambda_{j'}$ such that j is reachable from cell j', $j' < j$. The goal of the problem is to determine the sequence $\langle \mu_1, \mu_2, \ldots, \mu_n \rangle$. See Figure 5 for an illustration.*

We make a more refined notion of reachability. For any cells j and $j' < j$ we define the *first reachable point* $frp(j, j')$ on cell j from cell j' as the minimal t such that there exists $x_{j'+1} \leq x_{j'+2} \leq \ldots \leq x_j$ such that each $x_k \in [a_k, b_k]$ for every $j' < k \leq j$ and $x_j = t$ and we set $frp(j, j') = \infty$ if there exists no such t. Let $t_j(k)$ be the first reachable point on cell j from any cell j' with entry-cost at most k i.e. $t_j(k) = \min\{frp(j, j') \mid j' < j, \lambda_{j'} \leq k\}$. In particular we have $t_j(0) = \infty$, since $\lambda_{j'} > 0$ for all $j' < j$. We now make some simple observations about $t_j(k)$.

Observation 3.8. *μ_j is the minimal k such that $t_j(k) \neq \infty$.*

Proof. We have $t_j(k) \neq \infty$ if and only if cell j is reachable from some cell $j' < j$ with entry-cost $\lambda_{j'} \leq k$. Therefore, the minimal such $\lambda_{j'}$ is the minimal k at which $t_j(k) \neq \infty$. □

Thus, it suffices to show how to determine $t_j(\cdot)$ and μ_j from $t_j(\cdot)$ for all $j \in [n]$ in $\mathcal{O}(n)$ time. Our solution is similar to an algorithm by Alt et al. [5, Lemma 2.3].

Observation 3.9. *We have $t_j(k+1) \leq t_j(k)$ for any $j \in [n]$ and $k \geq 0$.*

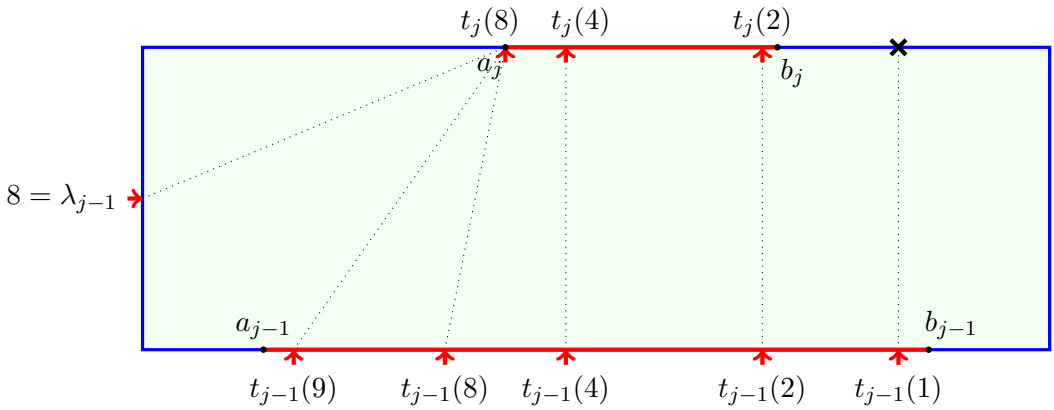


Figure 6: Illustration of the proof of Lemma 3.10. Determining the function $t_j(\cdot)$ from a_j,b_j,$t_{j-1}(\cdot)$, and λ_{j-1}. For all $k \geq \lambda_{j-1} = 8$ we have $t_j(k) = a_j$. For $k = 2$ and $k = 4$ we have $k < \lambda_{j-1}$ and $t_{j-1}(k) \leq b_j$, implying $t_j(k) = t_j(k-1)$. Lastly for $k = 1$ we have $t_{j-1}(k) > b_j$, implying $t_j(k) = \infty$.

Proof. The minimum in the definition of $t_j(k+1)$ is taken over a superset compared to $t_j(k)$. □

Lemma 3.10. *For any $j \in [n]$ and $k \geq 0$ we have*

$$t_j(k) = \begin{cases} a_j & \text{if } k \geq \lambda_{j-1} \\ a_j & \text{if } k < \lambda_{j-1} \text{ and } t_{j-1}(k) \leq a_j \\ t_{j-1}(k) & \text{if } k < \lambda_{j-1} \text{ and } t_{j-1}(k) \in (a_j, b_j] \\ \infty & \text{if } k < \lambda_{j-1} \text{ and } t_{j-1}(k) > b_j \end{cases}$$

Proof. See Figure 6 for an illustration. Note that $frp(j, j-1) = a_j$. Therefore, if $\lambda_{j-1} \leq k$, then $t_j(k) = \min_{j' < j, \lambda_{j'} \leq k} frp(j, j') \leq frp(j, j-1) = a_j$. Since $t_j(k) \geq a_j$, we conclude that $t_j(k) = a_j$. Now we discuss the cases when $k < \lambda_{j-1}$. Let $t_j(k) = frp(j, j')$. Since $\lambda_{j-1} > k$ we have $j' < j-1$. Therefore, there exist $x_{j'+1} \leq x_{j'+2} \leq \ldots \leq x_{j-1} \leq x_j$ such that $x_k \in [a_k, b_k]$ for every $j' < k \leq j$ with $x_j = t_j(k)$. Note that $t_{j-1}(k) \leq x_{j-1} \leq x_j = t_j(k)$. Thus, $t_j(k) \geq \max(t_{j-1}(k), a_j)$. In particular, if $t_{j-1}(k) > b_j$, then $t_j(k) = \infty$. Now we look into the case when $t_{j-1}(k) \leq b_j$. Observe that if $t_{j-1}(k) \leq b_j$ then there exists $\hat{j} < j-1$ and there exists $x_{\hat{j}+1} \leq x_{\hat{j}+2} \leq \ldots \leq x_{j-1} = t_{j-1}(k)$ such that $x_k \in [a_k, b_k]$ for every $\hat{j} < k \leq j-1$. Setting $x_j = \max(a_j, t_{j-1}(k))$ and there exists $x_{\hat{j}+1} \leq x_{\hat{j}+2} \leq \ldots \leq x_{j-1} \leq x_j$ such that $x_k \in [a_k, b_k]$ for every $\hat{j} \leq k \leq j$ and hence $t_j(k) \leq \max(a_j, t_{j-1}(k))$. Combining the two inequalities we get that $t_j(k) = \max(a_j, t_{j-1}(k))$ when $t_{j-1}(k) \leq b_j$. □

Lemma 3.10 yields a recursive definition for $t_j(\cdot)$. To ensure that we can solve an instance of *cell reachability* in $\mathcal{O}(n)$ time, if suffices to determine $t_j(\cdot)$ from $t_{j-1}(\cdot)$ and μ_j from $t_j(\cdot)$ in $\mathcal{O}(1)$ amortized time. To this end, let $S_j = \{k \geq 0 \mid t_j(k) < t_j(k-1)\}$ and let L_j be the *doubly linked list* storing the pairs $(k, t_j(k))$ for every $k \in S_j$, sorted in descending

order of k (or equivalently in increasing order of $t_j(k)$). To develop some intuition, note that for any k and j, if we have $t_j(k) = t_j(k-1)$, then this means that every cell $j' \geq j$ that is reachable from a cell $\hat{j} \leq j$ with entry-cost at most k is also reachable from some cell $\tilde{j} \leq j$ with entry-cost at most $k-1$. Since we are only interested in reachability from a cell of minimum entry-cost, we can ignore reachability from all cells below cell j with entry costs k. Therefore, it suffices to focus on the set S_j and the corresponding μ_j. In particular we can determine μ_j from S_j as following,

Lemma 3.11. *The minimal positive k in S_j is equal to μ_j.*

Proof. Since $t_j(0) = \infty$, the minimal positive k in S_j is the minimal k such that $t_j(k) \neq \infty$. By Observation 3.8 this is equal to μ_j. □

We now outline a simple algorithm to determine L_j from L_{j-1}. Again see Figure 6 for illustration. The algorithm first determines k_{left}, the minimal k such that $t_j(k) = a_j$, by moving the head of the list L_{j-1} to the right as long as $k \geq \lambda_{j-1}$ or $t_{j-1}(k) \leq a_j$ (correctness follows directly from Lemma 3.10). Observe that $t_j(k) = t_j(k_{left}) = a_j$ for all $k \geq k_{left}$. Next it determines k_{right}, the minimal k such that $t_j(k) \leq b_j$ by moving the tail of L_{j-1} to the minimal k such that $t_{j-1}(k) \leq b_j$. Note that at this point we have already inserted (k_{left}, a_j) so k_{right} is guaranteed to exits (again correctness follows from Lemma 3.10). Observe that $t_j(k) = t_j(0) = \infty$ for all $k < k_{right}$. Thus, we have $\mu_j = k_{right}$. Now we are left with updating L_j for pairs with $k \in (k_{left}, k_{right})$. Note that for $k \in (k_{left}, k_{right})$, we have $t_j(k) = t_{j-1}(k)$ (by Lemma 3.10) and therefore $t_j(k) = t_j(k-1)$ if and only if $t_{j-1}(k) = t_{j-1}(k-1)$. Thus, the sublist of L_j corresponding to the values of $k \in (k_{left}, k_{right})$ is same as the sublist of L_{j-1} corresponding to the values of $k \in (k_{left}, k_{right})$. Finally, the algorithm appends a new node to L_j storing $(0, \infty)$ (since $t_j(0) = \infty$).

Algorithm 2 Determining L_j from L_{j-1}

```
L ← L_{j-1}
k_left ← λ_j
while k ≥ λ_j or t ≤ a_j, where (k, t) = L.front() do
    k_left ← min(k_left, k)
    L.popfront()
L.pushfront((k_left, a_j))
while t > b_j, where (k, t) = L.back() do
    L.popback()
Set μ_j = k, where (k, t) = L.back().
L.pushback((0, ∞))
L_j ← L
```

The number of operations performed to determine L_j from L_{j-1} and determining μ_j from L_j is $\mathcal{O}(1+d)$ where d is the number of pairs deleted from L_{j-1}. Since every deleted pair was previously inserted, we can pay for the deletions by paying an extra token per insertion. Note that there are two insertions per update. Hence, the total time taken to determine L_j and μ_j for all $j \in [n]$ is $\mathcal{O}(n)$.

Theorem 3.12. *Cell Reachability can be solved in $\mathcal{O}(n)$ time.*

3.3.2 Implementing κ_2-subroutine(i) using Cell Reachability

Recall the definition of $\kappa_2(\cdot,\cdot)$ and what our goal is now: For a fixed $i' < i$, let $\kappa(i,j,i')$ be the minimal k such that for some $j' < j$, we have $\mathrm{DP}(k-1,i',j') \neq \infty$ and $\delta_F(P[\mathrm{DP}(k-1,i',j')\ldots t_{i,j}], \overline{v_{i'}v_i}) \leq \delta$. Note that $\kappa_2(i,j) = \min_{i'<i} \kappa(i,j,i')$. In order to show that the κ_2-subroutine(i) can be implemented in $\mathcal{O}(n^2)$, it suffices to show that for fixed $i' < i$ we can determine $\kappa(i,j,i')$ for all $j \in [n-1]$ in $\mathcal{O}(n)$ time.

Observation 3.13. *Let the line segment with endpoints (a_j, j) and (b_j, j) denote the free-space on hor_j in $FS_\delta(P, \overline{v_{i'}v_i})$ where $i' < i$. Then, for any $j' < j$ there is a monotone path from $(0, \mathrm{DP}(\kappa(i',j'),i',j'))$ to $(1, t_{i,j})$ in the free-space $FS_\delta(P, \overline{v_{i'}v_i})$ if and only if there exist $x_{j'+1} \leq x_{j'+2} \leq \ldots x_j$ with each $x_k \in [a_k, b_k]$ for all $j' < k \leq j$.*

Proof. The "only if" direction is straightforward. Note that the monotone path from $(0, \mathrm{DP}(\kappa(i',j'),i',j'))$ to $(1, t_{i,j})$ in the free-space $FS_\delta(P, \overline{v_{i'}v_i})$ intersects hor_k for all $j' < k \leq j$. Let x_k be the intersection of the path with hor_k for $j' < k \leq j$. Since the path lies inside the free-space $FS_\delta(P, \overline{v_{i'}v_i})$ we have $x_k \in [a_k, b_k]$ for every $j' < k \leq j$. Since the path is monotone we have $x_{j'+1} \leq x_{j'+2} \leq \ldots \leq x_j$.
Now we show the "if" direction. Assume there exist $x_{j'+1} \leq x_{j'+2} \leq \ldots \leq x_j$ and $x_k \in [a_k, b_k]$ for every $j' < k \leq j$. Since every $fbox_k$ is convex for every $j' < k < j$, the line segment with endpoints as (x_k, k) and $(x_{k+1}, k+1)$ lies inside $fbox_k$. By the same convexity argument it follows that the line segment with endpoints $(0, \mathrm{DP}(\kappa(i',j'),i',j')$ and $(x_{j'+1}, j'+1)$ lies inside $fbox_{j'}$ and the line segment with endpoints (x_j, j) and $(1, t_{i,j})$ also lies inside $fbox_j$. Therefore, we have a monotone path namely $\langle(0, \mathrm{DP}(\kappa(i',j'),i',j'), (x_{j'+1}, j'+1), (x_{j'+2}, j'+2)\ldots(x_j, j)(1, t_{i,j})\rangle$ inside the free-space $FS_\delta(P, \overline{v_{i'}v_i})$ from $(0, \mathrm{DP}(\kappa(i',j'),i',j')$ to $(1, t_{i,j})$. □

Observation 3.14. *Fix any i and j. Consider any $k > 0$ and any $i' < i$ and $j' < j$. If there is a monotone path from $(0, \mathrm{DP}(k,i',j'))$ to $(1, t_{i,j})$ in the free-space $FS_\delta(P, \overline{v_{i'}v_i})$ intersecting hor_j, then there is also a monotone path from $(0, \mathrm{DP}(\kappa(i',j'),i',j'))$ to $(1, t_{i,j})$ in the free-space $FS_\delta(P, \overline{v_{i'}v_i})$ intersecting hor_j.*

Proof. This is obvious by inspecting the free-space $FS_\delta(P, \overline{v_{i'}v_i})$ as follows. Since the monotone path intersects hor_j, we have $j' < j$. Observe that both $\mathrm{DP}(k, i', j')$ and $\mathrm{DP}(\kappa(i',j'),i',j')$ lie in the interval $[t_{i',j'}, s_{i',j'}]$. Also let z be the point at which the monotone path intersects $hor_{j'+1}$. Then, there is a monotone path in $FS_\delta(P, \overline{v_{i'}, v_i})$ from z to $(1, t_{i,j})$. Since $fbox_{j'}$ is convex (By Fact 3.3) the line segment joining $(0, \mathrm{DP}(\kappa(i',j'),i',j'))$ and z is contained in $fbox_{j'}$. Therefore, there is a monotone path from $(0, \mathrm{DP}(\kappa(i',j'),i',j'))$ to $(1, t_{i,j})$ by walking from $(0, \mathrm{DP}(\kappa(i',j'),i',j'))$ to z and then following the monotone path from z to $(1, t_{i,j})$. □

Observations 3.13 and 3.14 imply that $\kappa(i,j,i')$ is the minimal value of $1 + \kappa(i',j')$ over all $j' < j$ such that there exist $x_{j'+1} \leq x_{j'+2} \leq \ldots \leq x_j$ with every $x_k \in [a_k, b_k]$ for all

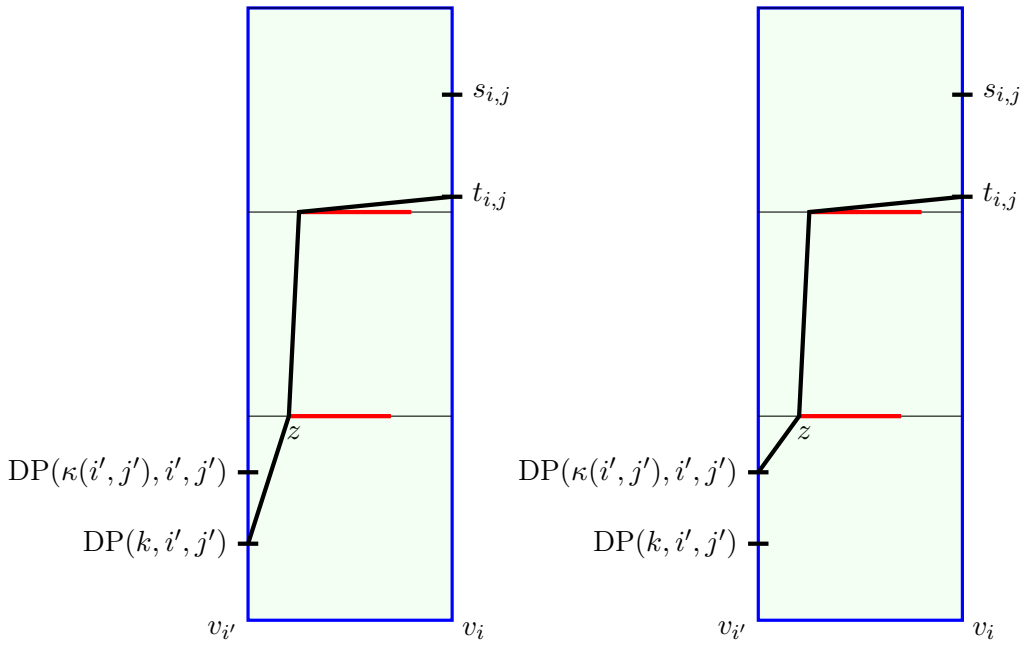
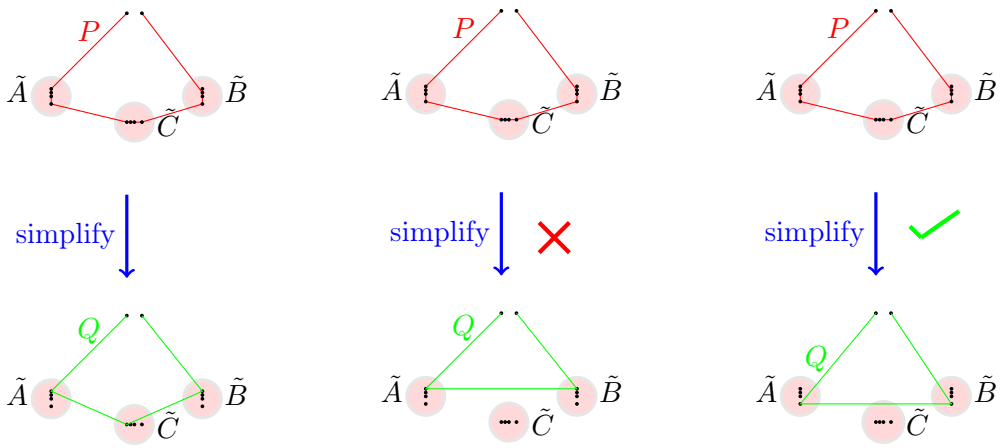


Figure 7: Illustration of the proof of Observation 3.14. For any $i' < i$, $j' < j$ and any k, there is a monotone path from $(0, \mathrm{DP}(k, i', j'))$ to $(1, t_{i,j})$ in $FS_\delta(P, \overline{v_{i'}v_i})$ (left) that intersects hor_j at z. Then, there is a monotone path from $(0, \mathrm{DP}(\kappa(i', j'), i', j'))$ to $(1, t_{i,j})$ in $FS_\delta(P, \overline{v_{i'}v_i})$ (right) by walking from $(0, \mathrm{DP}(\kappa(i', j'), i', j')$ to z and then following the existing monotone path from z to $(1, t_{i,j})$.



(a) $|Q| = 5$ (b) Incorrect choice of $\tilde{a}$ and $\tilde{b}$ (c) Correct choice of $\tilde{a}$ and $\tilde{b}$

Figure 8: Illustration of the reduction. P is the given polyline that visits vertices in $\tilde{A}$, $\tilde{C}$ and $\tilde{B}$ in the exact same order. Note that a simplification Q of size five is always possible as shown in (a). However a simplification of size four exists if and only if there exists $\tilde{a} \in A$ and $\tilde{b} \in B$ such that all vertices in $\tilde{C}$ are close to $\overline{\tilde{a}\tilde{b}}$ as illustrated in (b) and (c).

$j' < k \leq j$.

Note that now we are "almost" in an instance of the Cell Reachability problem where the passage p_j corresponds to the free space on hor_j and each $\lambda_j = 1 + \kappa(i', j)$. The only difference is that the free space on some hor_j could be empty (while in the Cell Reachability problem we never had empty passages). However if the free space on any hor_j is empty then there exists no monotone path in the free-space $FS_\delta(P, \overline{v_{i'}v_i})$ from any point below hor_j to any point above hor_j. Thus, we can split the instance into two disjoint instances of Cell Reachability. Thus, for any fixed i' we can determine $\kappa(i, j, i')$ in $\mathcal{O}(n)$ time, and therefore we can implement κ_2-subroutine(i) for any $i \in [n]$ in $T(n) = \mathcal{O}(n^2)$.

4 Conditional Lower Bound for Curve Simplification

In this section we show that an[6] $\mathcal{O}(n^{3-\varepsilon})$ time algorithm for Global-Fréchet, Local-Fréchet, or Local-Hausdorff simplification over $(\mathbb{R}^d, \|\|_p)$ for any $p \in [1, \infty)$, $p \neq 2$, would yield an $\mathcal{O}(n^{3-\varepsilon})$ algorithm for ∀∀∃-OV.

We first briefly present the ideas behind our conditional lower bounds. For better exposition, we will mention the previous conditional lower bounds known for polyline simplification and then highlight the novelty in our conditional lower bound. Then we move on to the details of our reduction.

Ideas behind the conditional lower bound. Let us first briefly sketch the previous conditional lower bound by Buchin et al. [8]. See Figure 8 for an illustration. Given a 2-OV

[6]Recall that throughout the paper in $\mathcal{O}$-notation we hide polynomial factors in d.

instance on vectors $A, B \subseteq \{0,1\}^d$, they construct corresponding point sets $\tilde{A}, \tilde{B} \subset \mathbb{R}^{d'}$ (for some $d' = \mathcal{O}(d)$), forming two clusters that are very far apart from each other. They also add a start- and an endpoint, which can be chosen far away from these clusters (in a new direction). Near the midpoint between $\tilde{A}$ and $\tilde{B}$, another set of points $\tilde{C}$ is constructed. The final curve then starts in the startpoint, walks through all points in $\tilde{A}$, then through all points in $\tilde{C}$, then through all points in $\tilde{B}$, and ends in the endpoint. This setup ensures that any reasonable size-4 simplification must consist of the startpoint, one point $\tilde{a} \in \tilde{A}$, one point $\tilde{b} \in \tilde{B}$, and the endpoint. All points in $\tilde{A}$ are close to $\tilde{a}$, so they are immediately close to the simplification, similarly for $\tilde{B}$. Thus, the constraints are in the points $\tilde{C}$. Buchin et al. [8] construct $\tilde{C}$ such that it contains one point for each dimension $\ell \in [d]$, which "checks" that the vectors corresponding to the chosen points $\tilde{a}, \tilde{b}$ are orthogonal in dimension ℓ, i.e., one of a or b has a 0 in dimension ℓ.

We instead want to reduce from ∀∀∃-OV, so we are given an instance A, B, C and want to know whether for all $a \in A, b \in B$ there exists $c \in C$ such that a, b, c are orthogonal. In our adapted setup, the set $\tilde{C}$ is in one-to-one correspondence to the set of vectors C (the global picture is still as in Figure 8). That is, choosing a size-4 simplification implements an existential quantifier over $a \in A, b \in B$. The constraints that all $\tilde{c} \in \tilde{C}$ are close to the line segment from $\tilde{a}$ to $\tilde{b}$ implements a universal quantifier over $c \in C$ (see Figure 8 for an illustration). Naturally, we want the distance from $\tilde{c}$ to the line segment $\overline{\tilde{a}\tilde{b}}$ to be large if a, b, c are orthogonal, and to be small otherwise. This simulates the negation of ∀∀∃-OV, so any curve simplification algorithm can be turned into an algorithm for ∀∀∃-OV.

The restriction $p \in [1, \infty)$ with $p \neq 2$ in Theorem 1.2 already is a hint that the specific construction of points is subtle. Indeed, let us sketch one critical issue in the following. We want the points $\tilde{C}$ to lie in the middle between $\tilde{A}$ and $\tilde{B}$, which essentially means that we want to consider the distance from $(\tilde{a} + \tilde{b})/2$ to $\tilde{c}$. Now consider just a single dimension of ∀∀∃-OV. Then, our task boils down to constructing points a_0, a_1 and b_0, b_1 and c_0, c_1, corresponding to the bits in this dimension, such that $\|(a_i + b_j)/2 - c_k\|_p = \beta_1$ if $i = j = k = 1$ and β_0 otherwise, with $\beta_1 < \beta_0$. Writing $a'_i = a_i/2$ and $b'_j = b_j/2$ for simplicity, in the case $p = 2$ we can simplify

$$\begin{aligned}
\|a'_i + b'_j - c_k\|_2^2 &= \sum_{\ell=1}^{d'} (a'_i[\ell] + b'_j[\ell] - c_k[\ell])^2 \\
&= \sum_{\ell=1}^{d'} \Big((a'_i[\ell] + b'_j[\ell])^2 + (a'_i[\ell] - c_k[\ell])^2 + (b'_j[\ell] - c_k[\ell])^2 - a'_i[\ell]^2 - b'_j[\ell]^2 - c_k[\ell]^2 \Big) \\
&= \|a'_i + b'_j\|_2^2 + \|a'_i - c_k\|_2^2 + \|b'_j - c_k\|_2^2 - \|a'_i\|_2^2 - \|b'_j\|_2^2 - \|c_k\|_2^2 \\
&= f_1(i,j) + f_2(j,k) + f_3(i,k), \qquad (2)
\end{aligned}$$

for some functions[7] $f_1, f_2, f_3 \colon \{0,1\} \times \{0,1\} \to \mathbb{R}$. Note that by assumption this is equal to β_1^2 if $i = j = k = 1$ and β_0^2 otherwise, with $\beta_1 < \beta_0$. After a linear transformation, we thus obtain a representation of the form (2) for the function $f(i,j,k) = i \cdot j \cdot k$ for $i, j, k \in \{0,1\}$. However, it can be checked that such a representation is impossible[8]. Therefore, for $p = 2$

[7] This holds for $f_1(i,j) := \|a'_i + b'_j\|_2^2 - \|a'_i\|_2^2$, $f_2(j,k) := \|b'_j - c_k\|_2^2 - \|b'_j\|_2^2$, and $f_3(i,k) := \|a'_i - c_k\|_2^2 - \|c_k\|_2^2$.

[8] For instance, we can express this situation by a linear system of equations in 12 variables (the 4 image

our outlined reduction cannot work - provably!

We nevertheless make this reduction work in the cases $p \in [1, \infty)$, $p \neq 2$. The above argument shows that the construction is necessarily subtle. Indeed, constructing the right points requires some technical effort, which we now elaborate.

4.1 Overview of the Reduction

We now give an overview of the reduction. Consider any instance (A, B, C) of $\forall\forall\exists$-OV where A,B,$C \subseteq \{0,1\}^d$ have size n. We write $A = \{a_1, a_2, \dots a_n\}$, $B = \{b_1, b_2, \dots b_n\}$, and $C = \{c_1, c_2, \dots c_n\}$. We will efficiently construct $3n+1$ points in $\mathbb{R}^D$ with $D \in \mathcal{O}(d)$ namely the sets of points $\tilde{A} = \{\tilde{a}_1, \tilde{a}_2, \dots \tilde{a}_n\}$, $\tilde{B} = \left\{\tilde{b}_1, \tilde{b}_2, \dots \tilde{b}_n\right\}$, and $\tilde{C} = \{\tilde{c}_1, \tilde{c}_2, \dots \tilde{c}_n\}$, and one more point s. We also determine $\delta \geq 0$, such that the following properties are satisfied.

($\mathbf{P}_1$) For any $\tilde{a} \in \tilde{A}, \tilde{b} \in \tilde{B}, \tilde{c} \in \tilde{C}$, there is a point x on the line segment $\overline{\tilde{a}\tilde{b}}$ with $\|x - \tilde{c}\|_p \leq \delta$ if and only if $\|\frac{\tilde{a}+\tilde{b}}{2} - \tilde{c}\|_p \leq \delta$.

($\mathbf{P}_2$) For any $\tilde{a} \in \tilde{A}, \tilde{b} \in \tilde{B}, \tilde{c} \in \tilde{C}$, we have $\|\frac{\tilde{a}+\tilde{b}}{2} - \tilde{c}\|_p \leq \delta$ if and only if $\sum_{\ell \in [d]} a[\ell] \cdot b[\ell] \cdot c[\ell] \neq 0$.

($\mathbf{P}_3$) $\|x - y\|_p \leq \delta$ holds for all $x, y \in \tilde{A}$, and for all $x, y \in \tilde{B}$, and for all $x, y \in \tilde{C}$.

($\mathbf{P}_4$) For any $y_1, y_2 \in \{s\} \cup \tilde{B} \cup \tilde{C}$ and any point x on the line segment $\overline{y_1 y_2}$ we have $\|x - \tilde{a}\|_p > \delta$ for all $\tilde{a} \in \tilde{A}$.

($\mathbf{P}_5$) For any $y_1, y_2 \in \{s\} \cup \tilde{A} \cup \tilde{C}$ and any point x on the line segment $\overline{y_1 y_2}$ we have $\|x - \tilde{b}\|_p > \delta$ for all $\tilde{b} \in \tilde{B}$.

($\mathbf{P}_6$) For any $y \in \tilde{B} \cup \tilde{A}$ and any point x on the line segment $\overline{sy}$ we have $\|x - \tilde{c}\|_p > \delta$ for all $\tilde{c} \in \tilde{C}$.

We postpone the exact construction of these points. Our hard instance for curve simplification will be $Q = \langle s, \tilde{a}_1, \tilde{a}_2, \dots, \tilde{a}_n, \tilde{c}_1, \tilde{c}_2, \dots, \tilde{c}_n, \tilde{b}_1, \tilde{b}_2, \dots, \tilde{b}_n, s \rangle$.

Lemma 4.1. *Let $\hat{Q} = \langle s, \tilde{a}_i, \tilde{b}_j, s \rangle$ for some $\tilde{a}_i \in \tilde{A}$ and $\tilde{b}_j \in \tilde{B}$. If $\|\frac{\tilde{a}_i+\tilde{b}_j}{2} - \tilde{c}\|_p \leq \delta$ for all $\tilde{c} \in \tilde{C}$ then the Local-Frechet distance between Q and $\hat{Q}$ is at most δ.*

Proof. Both Q and $\hat{Q}$ have the same starting point s. By property $\mathbf{P}_3$ we have $\|\tilde{a} - \tilde{a}_i\|_p \leq \delta$ for all $\tilde{a} \in \tilde{A}$, and $\|\tilde{b} - \tilde{b}_j\|_p \leq \delta$ for all $\tilde{b} \in \tilde{B}$. Thus, it follows that $\delta_F(\langle s, \tilde{a}_1, \dots, \tilde{a}_i \rangle, \overline{s\tilde{a}_i}) \leq \delta$ and $\delta_F(\langle \tilde{b}_j, \dots, \tilde{b}_n, s \rangle, \overline{\tilde{b}_j s}) \leq \delta$. It remains to show that $\delta_F(Q_{ij}, \overline{\tilde{a}_i \tilde{b}_j}) \leq \delta$ where $Q_{ij} = \langle \tilde{a}_i, \dots, \tilde{a}_n, \tilde{c}_1, \dots, \tilde{c}_n, \tilde{b}_1, \dots, \tilde{b}_j \rangle$. To this end, first note that both polylines Q_{ij} and $\overline{\tilde{a}_i \tilde{b}_j}$ have the same endpoints. We now outline monotone walks on both Q_{ij} and $\overline{\tilde{a}_i \tilde{b}_j}$.

values for each function f_i) and 8 equations (for the values of f on $i, j, k \in \{0, 1\}$) and verify that it has no solution.

(1) Walk on Q_{ij} from $\tilde{a}_i$ to $\tilde{a}_n$ and remain at $\tilde{a}_i$ on $\overline{\tilde{a}_i\tilde{b}_j}$.

(2) Walk uniformly on both polylines, up to $\frac{\tilde{a}_i+\tilde{b}_j}{2}$ on $\overline{\tilde{a}_i\tilde{b}_j}$ and up to $\tilde{c}_1$ on Q_{ij}.

(3) Walk on Q_{ij} from $\tilde{c}_1$ to $\tilde{c}_n$ and remain at $\frac{\tilde{a}_i+\tilde{b}_j}{2}$ on $\overline{\tilde{a}_i\tilde{b}_j}$.

(4) Walk uniformly on both curves up to $\tilde{b}_j$ on $\overline{\tilde{a}_i\tilde{b}_j}$ and up to $\tilde{b}_1$ on Q_{ij}.

(5) Walk on Q_{ij} until $\tilde{b}_j$ and remain at $\tilde{b}_j$ on $\overline{\tilde{a}_1\tilde{b}_j}$.

We now argue that we always stay within distance δ throughout the walks. For (1) and (5) this follows due to property $\mathbf{P}_3$. For (2) and (4) it follows due to the fact that we always remain within distance δ while walking with uniform speed on two line segments, as long as their startpoints and their endpoints are within distance δ. By the assumption $\|\frac{\tilde{a}_i+\tilde{b}_j}{2} - \tilde{c}\|_p \leq \delta$ for all $\tilde{c} \in \tilde{C}$, we always stay within distance δ also for (3). □

Observe that property $\mathbf{P_3}$ implies that there is a simplification of size five namely $\hat{Q} = \langle s, \tilde{a}, \tilde{c}, \tilde{b}, s\rangle$ for any $\tilde{a} \in \tilde{A}$, $\tilde{b} \in \tilde{B}$, and $\tilde{c} \in \tilde{C}$, such that the distance between $\hat{Q}$ and Q is at most δ under Local-Fréchet, Global-Fréchet and Local-Hausdorff distance. We now show that a smaller simplification is only possible if there exist $a \in A$, $b \in B$ such that for all $c \in C$ we have $\sum_{\ell\in[d]} a[\ell] \cdot b[\ell] \cdot c[\ell] \neq 0$.

Lemma 4.2. *Let $\hat{Q}$ be a simplification of the polyline Q of size 4. Then, the following statements are equivalent*:

(1) The Global-Fréchet distance between Q and $\hat{Q}$ is at most δ.

(2) The Local-Fréchet distance between Q and $\hat{Q}$ is at most δ.

(3) The Local-Hausdorff distance between Q and $\hat{Q}$ is at most δ.

(4) There exist some $\tilde{a} \in \tilde{A}$, $\tilde{b} \in \tilde{B}$, such that $\hat{Q} = \langle s, \tilde{a}, \tilde{b}, s\rangle$, and $\|\frac{\tilde{a}+\tilde{b}}{2} - \tilde{c}\|_p \leq \delta$ for every $\tilde{c} \in \tilde{C}$.

(5) There exist $a \in A$, $b \in B$ such that for all $c \in C$ we have $\sum_{\ell\in[d]} a[\ell] \cdot b[\ell] \cdot c[\ell] \neq 0$.

Proof. We first show that (1), (2), and (3) are equivalent to (4). To this end, we first show that each of (1), (2), and (3) imply (4). Since for any $y_1, y_2 \in s \cup \tilde{B} \cup \tilde{C}$ there is no point on the line segment $\overline{y_1y_2}$ that has distance at most δ to any $\tilde{a} \in \tilde{A}$ (by property $\mathbf{P}_4$), $\hat{Q}$ must contain at least one point from $\tilde{A}$. A symmetric argument can be made for the fact that $\hat{Q}$ must contain at least one point from $\tilde{B}$ (property $\mathbf{P}_5$). Since the size of $\hat{Q}$ is 4, we have $\hat{Q} = \langle s, \tilde{a}, \tilde{b}, s\rangle$ for some $\tilde{a} \in \tilde{A}$ and $\tilde{b} \in \tilde{B}$. By property $\mathbf{P}_6$ there is no point on the line segments $\overline{s\tilde{a}}$, and $\overline{\tilde{b}s}$ that has distance at most δ to any $\tilde{c} \in C$. Therefore, the Global-Fréchet distance or the Local-Fréchet distance, or the Local-Hausdorff distance between Q, and $\hat{Q}$

is at most δ only if for all $\tilde{c} \in \tilde{C}$ there is a point on the line segment $\overline{\tilde{a}\tilde{b}}$ that has distance at most δ to $\tilde{c}$. By property $\mathbf{P}_1$, this implies that $\|\frac{\tilde{a}+\tilde{b}}{2} - \tilde{c}\|_p \leq \delta$ for all $\tilde{c} \in \tilde{C}$.

Now we show that (4) implies (1), (2), and (3). First observe that (2) implies (1) and (3), since the Local-Fréchet distance between a curve and its simplification is at least the Global-Fréchet distance, and at least the Local-Hausdorff distance between the same. Thus, it suffices to show that (4) implies (2). This directly follows from Lemma 4.1. Finally, (4) and (5) are equivalent due to property $\mathbf{P}_2$. □

Observing that we can construct Q and determine δ in $\mathcal{O}(nd)$ time, the above lemma directly yields the following theorem.

Theorem 4.3. *For any $\varepsilon > 0$, there is no $\mathcal{O}(n^{3-\varepsilon})$ algorithm for Global-Fréchet, Local-Fréchet, and Local-Hausdorff simplification over $(\mathbb{R}^d, \|\|_p)$ for any $p \in [1, \infty)$, $p \neq 2$ unless* ∀∀∃-OV *Hypothesis fails. This holds even for the problem of deciding whether the optimal simplification has size ≤ 4 or ≥ 5.*

Proof. Given an instance A,B,C of ∀∀∃-OV, we can construct the curve Q and determine δ in $\mathcal{O}(nd)$. By Lemma 4.2, the simplification problem on (Q, δ) is equivalent to ∀∀∃-OV on A, B, C. Thus, any $\mathcal{O}(n^{3-\varepsilon})$ time algorithm for the curve simplification problem would yield an $\mathcal{O}(n^{3-\varepsilon})$ algorithm for ∀∀∃-OV. □

It remains to construct the point s, the sets $\tilde{A}$, $\tilde{B}$ and $\tilde{C}$ and determine δ. We first introduce some notation. For vectors x and y and $\alpha \in [-\frac{1}{2}, \frac{1}{2}]$, we define $P_{xy}(\alpha)$ as $(\frac{1}{2} - \alpha)x + (\frac{1}{2} + \alpha)y$. Moreover let $u_i \in \mathbb{R}^d$. We write $v = \left[u_1 u_2 \dots u_m\right]$ for the vector $v \in \mathbb{R}^{md}$ with $v[(j-1)d + k] = u_j[k]$ for any $j \in [m]$ and $k \in [d]$.

Fact 4.4. *Let $u_1, u_2, \dots, u_m \in \mathbb{R}^d$ and $v = \left[u_1 u_2 \dots u_m\right]$. Then, we have $\|v\|_p^p = \sum_{i \in [m]} \|u_i\|_p^p$.*

4.2 Coordinate gadgets

In this section, our aim is to construct points $\mathbf{A}_i$, $\mathbf{B}_i$, $\mathbf{C}_i$ for $i \in \{0, 1\}$ such that the distance $\|\mathbf{C}_i - P_{\mathbf{A}_j\mathbf{B}_k}(0)\|_p$ only depends on whether the bits $i, j, k \in \{0, 1\}$ seen as coordinates of vectors are orthogonal. In other words, the points $\mathbf{A}_i$, $\mathbf{B}_i$, $\mathbf{C}_i$ form a *coordinate gadget*. Formally we will prove the following lemma.

Lemma 4.5. *For any $p \neq 2$*

$$\|\mathbf{C}_i - P_{\mathbf{A}_j\mathbf{B}_k}(0)\|_p^p = \begin{cases} \beta_1 & \text{if } i = 1, j = 1, k = 1 \\ \beta_2 & \text{otherwise} \end{cases}$$

where $\beta_1 < \beta_2$.

In Section 4.3 we will use this lemma to construct the final point sets $\tilde{A}$, $\tilde{B}$ and $\tilde{C}$.

Let θ_1, θ_2, θ_3, θ_4 and θ_5 be positive constants. We construct the points $\mathbf{A}_1$, $\mathbf{B}_1$, $\mathbf{C}_1$ and $\mathbf{A}_0$, $\mathbf{B}_0$, $\mathbf{C}_0$ in $\mathbb{R}^9$ as follows:

$$\begin{aligned}
\mathbf{A}_0 &= \big[-\theta_1, \quad 0, \quad -\theta_2, \quad 0, \quad \theta_3, \quad 2\theta_3, \quad \theta_4, \quad -2\theta_4, \quad 0 \big] \\
\mathbf{A}_1 &= \big[\theta_1, \quad 2\theta_1, \quad \theta_2, \quad -2\theta_2, \quad -\theta_3, \quad 0, \quad -\theta_4, \quad 0, \quad 0 \big] \\
\mathbf{B}_0 &= \big[-\theta_1, \quad 0, \quad \theta_2, \quad 2\theta_2, \quad \theta_3, \quad -2\theta_3, \quad -\theta_4, \quad 0, \quad 0 \big] \\
\mathbf{B}_1 &= \big[\theta_1, \quad -2\theta_1, \quad -\theta_2, \quad 0, \quad -\theta_3, \quad 0, \quad \theta_4, \quad 2\theta_4, \quad 0 \big] \\
\mathbf{C}_0 &= \big[0, \quad 0, \quad 0, \quad 0, \quad 0, \quad 0, \quad 0, \quad 0, \quad \theta_5 \big] \\
\mathbf{C}_1 &= \big[-\theta_1, \quad 0, \quad -\theta_2, \quad 0, \quad -\theta_3, \quad 0, \quad -\theta_4, \quad 0, \quad 0 \big]
\end{aligned}$$

From these points we can compute the points $P_{\mathbf{A}_i\mathbf{B}_j}(0)$ for all $i, j \in \{0, 1\}$.

$$\begin{aligned}
P_{\mathbf{A}_0\mathbf{B}_0}(0) &= \big[-\theta_1, \quad 0, \quad 0, \quad \theta_2, \quad \theta_3, \quad 0, \quad 0, \quad -\theta_4, \quad 0 \big] \\
P_{\mathbf{A}_1\mathbf{B}_0}(0) &= \big[0, \quad \theta_1, \quad \theta_2, \quad 0, \quad 0, \quad -\theta_3, \quad -\theta_4, \quad 0, \quad 0 \big] \\
P_{\mathbf{A}_1\mathbf{B}_1}(0) &= \big[\theta_1, \quad 0, \quad 0, \quad -\theta_2, \quad -\theta_3, \quad 0, \quad 0, \quad \theta_4, \quad 0 \big] \\
P_{\mathbf{A}_0\mathbf{B}_1}(0) &= \big[0, \quad -\theta_1, \quad -\theta_2, \quad 0, \quad 0, \quad \theta_3, \quad \theta_4, \quad 0, \quad 0 \big]
\end{aligned}$$

Observe that $\|\mathbf{C}_0 - P_{\mathbf{A}_i\mathbf{B}_j}(0)\|_p^p = \sum_{r\in[5]} \theta_r^p$ for all $i, j \in \{0, 1\}$. Thus, all the points $P_{\mathbf{A}_i\mathbf{B}_j(0)}$ are equidistant from $\mathbf{C}_0$ irrespective of the exact values of θ_r for $r \in [5]$. Note that when $\theta_r = \theta$ for all $r \in [5]$, then $\|\mathbf{C}_1 - P_{\mathbf{A}_i\mathbf{B}_j}(0)\|_p^p = 4\theta^p + 2^p\theta^p$ for all $i, j \in \{0, 1\}$. Thus, all the points $P_{\mathbf{A}_i\mathbf{B}_j}(0)$ are equidistant from $\mathbf{C}_1$ when all the θ_r are the same. We now determine θ_r for $r \in [5]$ such that all but one point in $\big\{P_{\mathbf{A}_i\mathbf{B}_j}(0) | i, j \in \{0, 1\}\big\}$ are equidistant and far from $\mathbf{C}_1$. More precisely,

$$\|\mathbf{C}_1 - P_{\mathbf{A}_i\mathbf{B}_j}(0)\|_p^p = \begin{cases} \beta_1 & \text{if } i = 1, j = 1 \\ \beta_2 & \text{otherwise} \end{cases}$$

and $\beta_1 < \beta_2$. We first quantify the distances from $\{\mathbf{C}_0, \mathbf{C}_1\}$ to each of the points in $\big\{P_{\mathbf{A}_j\mathbf{B}_k}(0) \mid j, k \in \{0, 1\}\big\}$.

Lemma 4.6. *We have*

$$\|\mathbf{C}_i - P_{\mathbf{A}_j\mathbf{B}_k}(0)\|_p^p = \begin{cases} \sum_{r\in[5]} \theta_r^p & \text{if } i = 0 \\ 2\theta_2^p + 2^p\theta_3^p + 2\theta_4^p & \text{if } i = 1, j = 0, k = 0 \\ 2\theta_1^p + 2^p\theta_2^p + 2\theta_3^p & \text{if } i = 1, j = 1, k = 0 \\ 2\theta_1^p + 2\theta_3^p + 2^p\theta_4^p & \text{if } i = 1, j = 0, k = 1 \\ 2^p\theta_1^p + 2\theta_2^p + 2\theta_4^p & \text{if } i = 1, j = 1, k = 1 \end{cases}$$

We now set the exact values of θ_r for $r \in [5]$. We define values depending on p. When $1 \le p < 2$ we set

$$\theta_1 = (2^{p-1} - 1)^{\frac{1}{p}}, \theta_2 = 0, \theta_3 = 1, \theta_4 = 0, \theta_5 = 2^{\frac{p-1}{p}}$$

Now we make the following observation,

Observation 4.7. *When $1 \leq p < 2$, then*

$$\|\mathbf{C}_i - P_{\mathbf{A}_j\mathbf{B}_k}(0)\|_p^p = \begin{cases} 2^p(2^{p-1}-1) & \text{if } i=1, j=1, k=1 \\ 2^p & \text{otherwise} \end{cases}$$

Proof. Substituting the values of θ_k for every $k \in [4]$ in Lemma 4.6 we have that

$$\begin{aligned}
\|\mathbf{C}_0 - P_{\mathbf{A}_j\mathbf{B}_k}(0)\|_p^p &= 2^{p-1} - 1 + 0^p + 1^p + 0^p + 2^{p-1} && = 2^p \\
\|\mathbf{C}_1 - P_{\mathbf{A}_0\mathbf{B}_0}(0)\|_p^p &= 2 \cdot 0^p + 2^p \cdot 1^p + 2 \cdot 0^p && = 2^p \\
\|\mathbf{C}_1 - P_{\mathbf{A}_1\mathbf{B}_0}(0)\|_p^p &= 2 \cdot (2^{p-1}-1) + 2^p \cdot 0^p + 2 \cdot 1^p && = 2^p \\
\|\mathbf{C}_1 - P_{\mathbf{A}_0\mathbf{B}_1}(0)\|_p^p &= 2 \cdot (2^{p-1}-1) + 2 \cdot 1^p + 2^p \cdot 0^p && = 2^p \\
\|\mathbf{C}_1 - P_{\mathbf{A}_1\mathbf{B}_1}(0)\|_p^p &= 2^p \cdot (2^{p-1}-1) + 2 \cdot 0^p + 2^p \cdot 0^p && = 2^p(2^{p-1}-1)
\end{aligned}$$ □

In case $p > 2$. Then, we set

$$\theta_1 = 0, \theta_2 = (2^p-2)^{\frac{1}{p}}, \theta_3 = (2^p-4)^{\frac{1}{p}}, \theta_4 = (2^p-2)^{\frac{1}{p}}, \theta_5 = (2^{2p} - 3 \cdot 2^p)^{\frac{1}{p}}$$

We make a similar observation,

Observation 4.8. *When $p > 2$, then*

$$\|\mathbf{C}_i - P_{\mathbf{A}_j\mathbf{B}_k}(0)\|_p^p = \begin{cases} 2^{p+2} - 8 & \text{if } i=1, j=1, k=1 \\ 2^{2p} - 8 & \text{otherwise} \end{cases}$$

Proof. Substituting the values of θ_k for every $k \in [4]$ in Lemma 4.6 we have that

$$\begin{aligned}
\|\mathbf{C}_0 - P_{\mathbf{A}_j\mathbf{B}_k}(0)\|_p^p &= 0^p + (2^p-2) + (2^p-4) + (2^p-2) + (2^{2p} - 3 \cdot 2^p) && = 2^{2p} - 8 \\
\|\mathbf{C}_1 - P_{\mathbf{A}_0\mathbf{B}_0}(0)\|_p^p &= 2 \cdot (2^p-2) + 2^p \cdot (2^p-4) + 2 \cdot (2^p-2) && = 2^{2p} - 8 \\
\|\mathbf{C}_1 - P_{\mathbf{A}_1\mathbf{B}_0}(0)\|_p^p &= 2 \cdot 0^p + 2^p \cdot (2^p-2) + 2 \cdot (2^p-4) && = 2^{2p} - 8 \\
\|\mathbf{C}_1 - P_{\mathbf{A}_0\mathbf{B}_1}(0)\|_p^p &= 2 \cdot 0^p + 2 \cdot (2^p-4) + 2^p \cdot (2^p-2) && = 2^{2p} - 8 \\
\|\mathbf{C}_1 - P_{\mathbf{A}_1\mathbf{B}_1}(0)\|_p^p &= 2^p \cdot 0^p + 2 \cdot (2^p-2) + 2 \cdot (2^p-2) && = 2^{p+2} - 8
\end{aligned}$$ □

Combining Observations 4.7 and 4.8 we arrive at Lemma 4.5.

4.3 Vector gadgets

For every $a \in A$, $b \in B$, and $c \in C$ we introduce vectors a', b', c', and a'', b'', c'', and then concatenate the respective vectors to form $\tilde{a}$, $\tilde{b}$, and $\tilde{c}$ respectively. Intuitively, a', b', c' primarily help us to ensure properties $\mathbf{P}_1$, and $\mathbf{P}_2$, while a'', b'', c'' help us ensure the remaining properties.

4.3.1 The vectors a', b', c', and s'

We construct the vector s', and the vectors a', b', and c' for every $a \in A$, $b \in B$, and $c \in C$ respectively, in $\mathbb{R}^{9d}$ as follows:

$$a' = \left[\mathbf{A}_{a[1]}, \mathbf{A}_{a[2]}, \dots \mathbf{A}_{a[d]}\right] \tag{3}$$
$$b' = \left[\mathbf{B}_{b[1]}, \mathbf{B}_{b[2]}, \dots \mathbf{B}_{b[d]}\right] \tag{4}$$
$$c' = \left[\mathbf{C}_{c[1]}, \mathbf{C}_{c[2]}, \dots \mathbf{C}_{c[d]}\right] \tag{5}$$
$$s' = \left[0, 0, \dots, 0\right] \tag{6}$$

We also define the sets $A' = \{a' \mid a \in A\}$, $B' = \{b' \mid b \in B\}$ and $C' = \{c' \mid c \in C\}$. We now make a technical observation about the vectors in A', B', and C', that will be useful later. We set $\eta_1 = \max_{i \in [5]} \theta_i$.

Observation 4.9. *For any $x, y \in A' \cup B' \cup C'$, we have $\|x - y\|_p \leq \eta_2$ where $\eta_2 := 36d\eta_1$.*

Proof. Note that the absolute value of every coordinate of the vectors $\mathbf{A}_1, \mathbf{B}_1, \mathbf{C}_1$ and also $\mathbf{A}_0, \mathbf{B}_0, \mathbf{C}_0$ is bounded by $2\eta_1$ (Since every coordinate is of the form $\pm\theta_r$ or $\pm 2\theta_r$ or 0). Also every coordinate of a', b', and c', is a coordinate of one of $\mathbf{A}_1, \mathbf{B}_1, \mathbf{C}_1, \mathbf{A}_0, \mathbf{B}_0$ and $\mathbf{C}_0$. Therefore, for any $x, y \in A' \cup B' \cup C'$ we have $\max_{\ell \in [9d]} |x[\ell] - y[\ell]| \leq 4\eta_1$. Hence, we have $\|x - y\|_p \leq \sum_{\ell \in [9d]} |x[\ell] - y[\ell]| \leq 9d \cdot 4\eta_1 = 36d\eta_1 = \eta_2$. □

Note that $a \in A$, $b \in B$, and $c \in C$ are non orthogonal if and only if $\#_{111}^{c,a,b} > 0$ where $\#_{111}^{c,a,b} = |\{i \mid i \in [d], a[i] = b[i] = c[i] = 1\}|$. The following lemma shows a connection between non-orthogonality and small distance $\|c' - P_{a'b'}(0)\|_p$.

Lemma 4.10. *For any $a \in A$, $b \in B$, and $c \in C$ we have $\|c' - P_{a'b'}(0)\|_p^p = d\beta_2 - (\beta_2 - \beta_1)\#_{111}^{c,a,b}$.*

Proof. By Lemma 4.5, for any $\alpha \in [-\frac{1}{2}, \frac{1}{2}]$

$$\|\mathbf{C}_{c[\ell]} - P_{\mathbf{A}_{a[\ell]}\mathbf{B}_{b[\ell]}}(0)\|_p^p = \begin{cases} \beta_1 & \text{if } c[\ell] = a[\ell] = b[\ell] = 1 \\ \beta_2 & \text{otherwise} \end{cases}$$

By Observation 4.4 we have

$$\begin{aligned} \|c' - P_{a'b'}(0)\|_p^p &= \sum_{\ell \in [d]} \|\mathbf{C}_{c[\ell]} - P_{\mathbf{A}_{a[\ell]}\mathbf{B}_{b[\ell]}}(0)\|_p^p \\ &= \beta_2(d - \#_{111}^{c,a,b}) + \beta_1 \#_{111}^{c,a,b} \\ &= d\beta_2 - (\beta_2 - \beta_1)\#_{111}^{c,a,b}. \end{aligned}$$

□

4.3.2 The vectors a'',b'', c'', and s''

We construct the vector s'' and the vectors a'',b'', and c'' for every $a \in A$,$b \in B$, and $c \in C$, respectively, in $\mathbb{R}^3$ as follows,

$$\begin{aligned} a'' &= [\gamma_1, 0, 0] \\ b'' &= [\gamma_1, \gamma_2, 0] \\ c'' &= [0, \frac{\gamma_2}{2}, 0] \\ s'' &= [0, \frac{\gamma_2}{2}, \gamma_2] \end{aligned}$$

where γ_1, γ_2 are positive constants. We are now ready to define the final points of our construction, s and $\tilde{a}$, $\tilde{b}$ and $\tilde{c}$ for any $a \in A$, $b \in B$ and $c \in C$ respectively.

$$\begin{aligned} \tilde{a} &= [a', a''] \\ \tilde{b} &= [b', b''] \\ \tilde{c} &= [c', c''] \\ s &= [s', s''] \end{aligned}$$

We set

$$\gamma_1 = \eta_1, \delta = (\gamma_1^p + d\beta_2 - (\beta_2 - \beta_1))^{\frac{1}{p}}, \gamma_2 = \max\left(4\delta, \eta_2\left(1 + \frac{(\gamma_1^p + d\beta_2)^{\frac{1}{p}}}{(\gamma_1^p + d\beta_2)^{\frac{1}{p}} - \delta}\right)\right)$$

Note that we have constructed the point sets $\tilde{A}$, $\tilde{B}$, $\tilde{C}$, and the point s and determined δ in total time $\mathcal{O}(nd)$. Therefore, now it suffices to show that our point set and δ satisfy the properties $\mathbf{P}_1$, $\mathbf{P}_2$, $\mathbf{P}_3$, $\mathbf{P}_4$, $\mathbf{P}_5$, and $\mathbf{P}_6$. To this end we first show how the distance $\|\tilde{c} - P_{\tilde{a}\tilde{b}}(\alpha)\|_p$ is related with $\#_{111}^{c,a,b}$ (the non-orthogonality of the vectors a,b, and c) by the following lemma.

Lemma 4.11. *For any $a \in A$, $b \in B$, and $c \in C$ we have,*

- $\|\tilde{c} - P_{\tilde{a}\tilde{b}}(0)\|_p^p = \gamma_1^p + \beta_2 d - (\beta_2 - \beta_1)\#_{111}^{c,a,b}$.
- *If $\#_{111}^{c,a,b} = 0$ then $\|\tilde{c} - P_{\tilde{a}\tilde{b}}(\alpha)\|_p^p > \delta$ for all $\alpha \in [-\frac{1}{2}, \frac{1}{2}]$.*

Proof. Note that

$$\begin{aligned} \tilde{c} - P_{\tilde{a}\tilde{b}}(\alpha) &= [c' - P_{a'b'}(\alpha), -\gamma_1, -\gamma_2\alpha, 0] \\ &= [c' - P_{a'b'}(0), -\gamma_1, -\gamma_2\alpha, 0] - [P_{a'b'}(\alpha) - P_{a'b'}(0), 0, 0, 0] \end{aligned}$$

Thus, substituting α as 0,

$$\begin{aligned} \|\tilde{c} - P_{\tilde{a}\tilde{b}}(0)\|_p^p &= \|[c' - P_{a'b'}(0), -\gamma_1]\|_p^p \\ &= \gamma_1^p + \|c' - P_{a'b'}(0)\|_p^p \\ &= \gamma_1^p + d\beta_2 - (\beta_2 - \beta_1)\#_{111}^{c,a,b} \qquad \text{(by Lemma 4.10)} \end{aligned}$$

Furthermore, by reverse triangle inequality we have

$$\begin{aligned}\|\tilde{c} - P_{\tilde{a}\tilde{b}}(\alpha)\|_p &\geq \|\big[c' - P_{a'b'}(0), -\gamma_1, -\gamma_2\alpha, 0\big]\|_p - \|\big[P_{a'b'}(\alpha) - P_{a'b'}(0), 0, 0, 0\big]\|_p \\ &= \|\big[c' - P_{a'b'}(0), -\gamma_1, -\gamma_2\alpha\big]\|_p - \|\big[P_{a'b'}(\alpha) - P_{a'b'}(0)\big]\|_p.\end{aligned}$$

We bound the two summands on the right hand side. Note that $\|\big[c'-P_{a'b'}(0), -\gamma_1, -\gamma_2\alpha\big]\|_p \geq \max((\gamma_1^p+d\beta_2-(\beta_2-\beta_1)\#_{111}^{c,a,b})^{\frac{1}{p}}, |\alpha|\gamma_2)$. We also have $\|\big[P_{a'b'}(\alpha)-P_{a'b'}(0)\big]\|_p = |\alpha|\|b-a\|_p \leq |\alpha|\eta_2$ (by Observation 4.9). Therefore, when $\#_{111}^{c,a,b} = 0$, for any $\alpha \in [-\frac{1}{2}, \frac{1}{2}]$ we have,

$$\|\tilde{c} - P_{\tilde{a}\tilde{b}}(\alpha)\|_p \geq \max((\gamma_1^p + d\beta_2)^{\frac{1}{p}}, |\alpha|\gamma_2) - |\alpha|\eta_2$$

Now we consider two cases. If $|\alpha| < \frac{1}{\eta_2}((\gamma_1^p + d\beta_2)^{\frac{1}{p}} - \delta)$, then

$$\begin{aligned}\|\tilde{c} - P_{\tilde{a}\tilde{b}}(\alpha)\|_p &> (\gamma_1^p + d\beta_2)^{\frac{1}{p}} - ((\gamma_1^p + d\beta_2)^{\frac{1}{p}} - \delta) \\ &= \delta\end{aligned}$$

Similarly if $|\alpha| \geq \frac{1}{\eta_2}((\gamma_1^p + d\beta_2)^{\frac{1}{p}} - \delta)$, we have

$$\begin{aligned}\|\tilde{c} - P_{\tilde{a}\tilde{b}}(\alpha)\|_p &\geq |\alpha|\gamma_2 - |\alpha|\eta_2 \\ &= |\alpha|(\gamma_2 - \eta_2) \\ &\geq \frac{1}{\eta_2}(\gamma_1^p + d\beta_2)^{\frac{1}{p}} - \delta) \cdot \eta_2\left(\frac{(\gamma_1^p + d\beta_2)^{\frac{1}{p}}}{(\gamma_1^p + d\beta_2)^{\frac{1}{p}} - \delta}\right) \quad (\text{ substituting } \gamma_2 \text{ and } \alpha) \\ &= (\gamma_1^p + d\beta_2)^{\frac{1}{p}} \\ &> \delta.\end{aligned}$$

Combining the two cases, we arrive at the second result of the lemma. □

We now verify properties $\mathbf{P}_1$, $\mathbf{P}_2$, $\mathbf{P}_3$, $\mathbf{P}_4$, $\mathbf{P}_5$, and $\mathbf{P}_6$.

Lemma 4.12 ($\mathbf{P}_2$). *For any $a \in A$, $b \in B$, and $c \in C$ we have $\|\tilde{c} - P_{\tilde{a}\tilde{b}}(0)\|_p \leq \delta$ if and only if $\#_{111}^{c,a,b} \geq 1$ or equivalently when $\sum_{\ell\in[d]} a[\ell] \cdot b[\ell] \cdot c[\ell] \neq 0$.*

Proof. By Lemma 4.11 we have that $\|\tilde{c}-P_{\tilde{a}\tilde{b}}(0)\|_p = (\gamma_1^p+d\beta_2-(\beta_2-\beta_1)\#_{111}^{c,a,b})^{\frac{1}{p}}$. Therefore, if $\#_{111}^{c,a,b} \geq 1$ then $\|\tilde{c} - P_{\tilde{a}\tilde{b}}(0)\|_p \leq \delta$. Conversely if $\#_{111}^{c,a,b} = 0$ then $\|\tilde{c} - P_{\tilde{a}\tilde{b}}(0)\|_p = (\gamma_1^p + d\beta_2)^{\frac{1}{p}} > (\gamma_1^p + d\beta_2 - (\beta_2 - \beta_1))^{\frac{1}{p}} = \delta$. □

Lemma 4.13 ($\mathbf{P}_1$). *For any $a \in A$, $b \in B$, and $c \in C$ we have $\|\tilde{c} - P_{\tilde{a}\tilde{b}}(\alpha)\|_p \leq \delta$ for any $\alpha \in [-\frac{1}{2}, \frac{1}{2}]$, if and only if $\|\tilde{c} - P_{\tilde{a}\tilde{b}}(0)\|_p \leq \delta$.*

Proof. The "if" statement is trivial as $\|\tilde{c} - P_{\tilde{a}\tilde{b}}(\alpha)\|_p \leq \delta$ for $\alpha = 0$. For the "only if" case, since $\|\tilde{c} - P_{\tilde{a}\tilde{b}}(0)\|_p > \delta$, from Lemma 4.12 it follows that $\#_{111}^{c,a,b} = 0$. By Lemma 4.11 we obtain $\|\tilde{c} - P_{\tilde{a}\tilde{b}}(\alpha)\|_p^p > \delta$ for all $\alpha \in [-\frac{1}{2}, \frac{1}{2}]$. Therefore, there exists no $\alpha \in [-\frac{1}{2}, \frac{1}{2}]$ such that $\|\tilde{c} - P_{\tilde{a}\tilde{b}}(\alpha)\|_p \leq \delta$. □

Lemma 4.14 ($\mathbf{P}_3$). *We have $\|x-y\|_p \leq \delta$ for all $x, y \in \tilde{A}$, and for all $x, y \in \tilde{B}$, and for all $x, y \in \tilde{C}$.*

Proof. We prove the case of $x, y \in \tilde{A}$; the other cases are analogous. Consider any $\tilde{a}_1, \tilde{a}_2 \in \tilde{A}$. Note that $\|\tilde{a}_1 - \tilde{a}_2\|_p = \|a'_1 - a'_2\|_p$. By Observation 4.9, we have $\|a'_1 - a'_2\|_p \leq \eta_2 < \gamma_1 \leq \delta$. □

We now prove properties $\mathbf{P}_4$, $\mathbf{P}_5$ and $\mathbf{P}_6$.

Lemma 4.15 ($\mathbf{P}_4$,$\mathbf{P}_5$, and $\mathbf{P}_6$). *For any $a \in A$, $b \in B$, and $c \in C$, and $\alpha \in [-\frac{1}{2}, \frac{1}{2}]$, the following properties hold.*

1. *For any $y_1, y_2 \in \{s\} \cup \tilde{B} \cup \tilde{C}$, we have $\|\tilde{a} - P_{y_1 y_2}(\alpha)\|_p > \delta$ for all $\tilde{a} \in \tilde{A}$.*
2. *For any $y_1, y_2 \in \{s\} \cup \tilde{A} \cup \tilde{C}$, we have $\|\tilde{b} - P_{y_1 y_2}(\alpha)\|_p > \delta$ for all $\tilde{b} \in \tilde{B}$.*
3. *For any $y \in \tilde{A} \cup \tilde{B}$, we have $\|\tilde{c} - P_{sy}(\alpha)\|_p > \delta$ for all $\tilde{c} \in \tilde{C}$.*

Proof. Since we set γ_2 to at least 4δ, we have $\frac{\gamma_2}{2} > \delta$. We first prove (1). For any $y_1, y_2 \in \{s\} \cup \tilde{B} \cup \tilde{C}$ we have $y_1[9d+2] \geq \frac{\gamma_2}{2}$ and $y_2[9d+2] \geq \frac{\gamma_2}{2}$. Therefore, for any $\alpha \in [-\frac{1}{2}, \frac{1}{2}]$ we have $P_{y_1 y_2}(\alpha)[9d+2] \geq \frac{\gamma_2}{2}$. For any $\tilde{a} \in \tilde{A}$ we have $\tilde{a}[9d+2] = 0$. Hence, we obtain $\|\tilde{a} - P_{\tilde{y}_1 \tilde{y}_2}(\alpha)\|_p \geq |\tilde{a}[9d+2] - P_{y_1 y_2}(\alpha)[9d+2]| \geq \frac{\gamma_2}{2} > \delta$.

We now make a symmetric argument for (2). For any $y_1, y_2 \in \{s\} \cup \tilde{A} \cup \tilde{C}$ we have $y_1[9d+2] \leq \frac{\gamma_2}{2}$ and $y_2[9d+2] \leq \frac{\gamma_2}{2}$. Therefore, for any $\alpha \in [-\frac{1}{2}, \frac{1}{2}]$ we have $P_{y_1 y_2}(\alpha)[9d+2] \leq \frac{\gamma_2}{2}$. For any $\tilde{b} \in \tilde{B}$ we have $\tilde{b}[9d+2] = \gamma$. Like earlier we obtain $\|\tilde{b} - P_{\tilde{y}_1 \tilde{y}_2}(\alpha)\|_p \geq |\tilde{b}[9d+2] - P_{y_1 y_2}(\alpha)[9d+2]| \geq \frac{\gamma_2}{2} > \delta$.

We now show (3). For this we state a simple observation.

Observation 4.16. *For any $\alpha \in [-\frac{1}{2}, \frac{1}{2}]$ we have,*

- $\|c'' - P_{b'' s''}(\alpha)\|_p \geq \frac{\gamma_2}{3} > \delta$.
- $\|c'' - P_{a'' s''}(\alpha)\|_p \geq \frac{\gamma_2}{3} > \delta$.

Proof. Observe that

$$c'' - P_{b'' s''}(\alpha) = [-(\tfrac{1}{2} - \alpha)\gamma_1, (\tfrac{\alpha}{2} - \tfrac{1}{4})\gamma_2, -(\tfrac{1}{2} + \alpha)\gamma_2]$$
$$c'' - P_{a'' s''}(\alpha) = [-(\tfrac{1}{2} - \alpha)\gamma_1, -(\tfrac{\alpha}{2} - \tfrac{1}{4})\gamma_2, -(\tfrac{1}{2} + \alpha)\gamma_2]$$

It follows that for $\alpha \in [-\frac{1}{2}, \frac{1}{2}]$ we have

$$\|c'' - P_{b'' s''}(\alpha)\|_p \geq \max\left(|(\tfrac{\alpha}{2} - \tfrac{1}{4})\gamma_2|, |(\tfrac{1}{2} + \alpha)\gamma_2|\right) = \gamma_2 \cdot \max(|\tfrac{\alpha}{2} - \tfrac{1}{4}|, |\tfrac{1}{2} + \alpha|) = \tfrac{\gamma_2}{3}$$
$$\|c'' - P_{a'' s''}(\alpha)\|_p \geq \max\left(|(\tfrac{\alpha}{2} - \tfrac{1}{4})\gamma_2|, |(\tfrac{1}{2} + \alpha)\gamma_2|\right) = \gamma_2 \cdot \max(|\tfrac{\alpha}{2} - \tfrac{1}{4}|, |\tfrac{1}{2} + \alpha|) = \tfrac{\gamma_2}{3}$$

Again since we set γ_2 to at least 4δ, we have $\frac{\gamma_2}{3} > \delta$. □

For any $y \in \tilde{A} \cup \tilde{B}$, we define $y'' = a''$ if $y = \tilde{a} \in \tilde{A}$ and $y'' = b''$ if $y = \tilde{b} \in \tilde{B}$. Then, by Observation 4.16 we have $\|\tilde{c} - P_{sy}(\alpha)\|_p \geq \|c'' - P_{y'' s''}(\alpha)\|_p > \delta$. This finishes the proof of Lemma 4.15, and thus of Theorem 1.2. □

5 Barriers to SETH-based lower bounds for Polyline Simplification (under Local-Hausdorff)

In this section we highlight a barrier for reductions from SAT or 3-OV to polyline simplification. Specifically, we will show that polyline simplification under Local Hausdorff has nondeterministic and co-nondeterministic algorithms running in quadratic time. This will rule out any superquadratic lower bound for polyline simplification based on SETH or k-OV (under deterministic fine-grained reductions and assuming a hypothesis known as Nondeterministic SETH [9]).

We start by introducing some basic terminology. Let $\mathcal{P}$ be a problem and T be a time bound. We say that problem $\mathcal{P}$ is in NTIME[T] if it admits an $\mathcal{O}(T)$-time nondeterministic algorithm. We say that $\mathcal{P}$ is in coNTIME[T] (co-nondeterministic time T) if its complement $\overline{\mathcal{P}}$ admits an $\mathcal{O}(T)$-time nondeterministic algorithm.

Intuitively, a *fine-grained reduction* from a problem $\mathcal{P}_1$ at time T_1 to a problem $\mathcal{P}_2$ at time T_2 (denoted by $(\mathcal{P}_1, T_1) \leq_{\text{FGR}} (\mathcal{P}_2, T_2)$) shows that any algorithm solving $\mathcal{P}_2$ in time $\mathcal{O}((T_2)^{1-\varepsilon})$ for $\varepsilon > 0$ can be transformed into an algorithm solving $\mathcal{P}_1$ in time $\mathcal{O}((T_2)^{1-\varepsilon'})$ for $\varepsilon' > 0$. For a formal definition of fine-grained reductions we refer the reader to [9]. For the purpose of this paper we want the reduction not to use randomization, and we highlight this by writing $(\mathcal{P}_1, T_1) \leq^{\text{det}}_{\text{FGR}} (\mathcal{P}_2, T_2)$. These reductions are known to transfer running time savings not just in the deterministic setting, but also in nondeterministic and co-nondeterminstic settings [9].

Carmosino et al. [9] posed and justified the following hypothesis.

Nondeterministic SETH (NSETH) [9]: For any $\varepsilon > 0$, there is a k such that k-SAT does not have a $\mathcal{O}((2-\varepsilon)^n)$ time co-nondeterministic algorithm.

Intuitively, since SAT may not have a faster co-nondeterministic algorithm (in comparison to its deterministic algorithm), it is unlikely that any problem $\mathcal{P}$ that is fine-grained reducible from SAT admits faster nondeterministic and co-nondeterministic algorithms (in comparison to its deterministic algorithm). This intuition is summarized by the following lemma.

Lemma 5.1 ([9]). *Let $\mathcal{P} \in (\text{N} \cap \text{coN})\text{TIME}[T]$ and assume NSETH. Then, there exists no reduction $(\text{SAT}, 2^n) \leq^{\text{det}}_{\text{FGR}} (\mathcal{P}, T^{1+\gamma})$ for any $\gamma > 0$. Similarly, there exists no reduction $(k\text{-}OV, n^k) \leq^{\text{det}}_{\text{FGR}} (\mathcal{P}, T^{1+\gamma})$ for any $k \geq 2$ and $\gamma > 0$.*

We will next show $\mathcal{O}(n^2)$-time nondeterministic and co-nondeterministic algorithms for polyline simplification under Local-Hausdorff. Thus, by Lemma 5.1 it follows that we cannot show cubic time hardness via a deterministic fine-grained reduction from SAT or 3-OV. In particular, this shows why we introduce a new hypothesis in this paper instead of working with the more standard SETH.

We will consider the *decision variant* of polyline simplification, where we are given as input a polyline P, a distance $\delta \geq 0$, and a number $k \geq 1$, and we want to decide whether there exists a simplification Q of P of size at most k and Local-Hausdorff distance at most δ to P. We denote by G_δ the *simplification graph* defined on vertex set $[n]$, which contains

a directed edge from i to j if and only if $\delta_H(\overline{v_iv_j}, P[i \dots j]) \leq \delta$, for any $i < j$. Moreover, by $d_G(i,j)$ we denote the length of the shortest path from vertex i to vertex j in graph G. Now we state a well know result in polyline simplification under local measures.

Lemma 5.2 ([20]). *There exists a simplification Q of size at most k of P that has Local-Hausdorff distance at most δ to P if and only if $d_{G_\delta}(1,n) \leq k$.*

Lemma 5.2 will be crucial for us to outline quadratic-time nondeterministic and co-nondeterministic algorithms for polyline simplification under Local-Hausdorff. We briefly sketch the two algorithms: For the nondeterministic algorithm, we nondeterministically guess a simplification Q of size at most k of P. Then, we can verify that the Local-Hausdorff distance between P and Q is at most δ in $\mathcal{O}(n)$-time. For the co-nondeterminstic algorithm, we non-deterministically guess the edges $\overline{E}$ that are not in G_δ and for each such edge (i,j), we can non-deterministically guess a certificate (an integer $w(i,j) \in \{i, \dots, j\}$) such that we can verify in $\mathcal{O}(n)$-time that the edges in $\overline{E}$ are not in G_δ. Then, in $\mathcal{O}(n^2)$-time we can verify that the length of the shortest path in G_δ (even if G_δ contains all the edges other than $\overline{E}$) is larger than k. We will now elaborate the two algorithms.

The following observation will help us in designing appropriate "certificates" to be guessed by the nondeterministic and co-nondeterministic algorithms.

Observation 5.3. *For $P = \langle v_0, v_1, \dots, v_n \rangle$ and any $i < j$ we have $\delta_H(\overline{v_iv_j}, P[i \dots j]) > \delta$ if and only if there exists an integer $k \in \{i, \dots, j\}$ such that the distance from v_k to $\overline{v_iv_j}$ is larger than δ.*

Proof. The "if" direction follows immediately since there is a point, namely $P[k] \in P[i \dots j]$, that has distance larger than δ to $\overline{v_iv_j}$.

We now show the "only if" direction. Assume that all vertices v_k such that $i \leq k \leq j$ have distance at most δ to $\overline{v_iv_j}$. Now consider any integer $k \in \{i, \dots, j\}$. Since both v_k and v_{k+1} have distance at most δ to $\overline{v_iv_j}$, we obtain that all points on $\overline{v_kv_{k+1}}$ have distance at most δ to $\overline{v_iv_j}$ (this follows from convexity of the free-space). Thus, all points in the sub-polyline $P[i \dots j]$ have distance at most δ to $\overline{v_iv_j}$. Therefore, $\delta_H(\overline{v_iv_j}, P[i \dots j]) \leq \delta$. □

We need some more notation. Let $\overrightarrow{K_n}$ denote the *complete directed graph* on the vertex set $[n]$ where there is a directed edge from i to j if $i < j$. Also we will use the notation $\overrightarrow{K_n} \setminus E$ to denote the graph that we are left with following the removal of the edges in E from $\overrightarrow{K_n}$.

With Lemma 5.2 and Observation 5.3 we are ready to establish that polyline simplification under Local-Hausdorff has faster nondeterministic and co-nondeterministic algorithms.

Theorem 5.4. *The decision problem of polyline simplification under Local-Hausdorff belongs to* $(\mathrm{N} \cap \mathrm{coN})\mathrm{TIME}[n^2]$.

Proof. We first outline an $\mathcal{O}(n)$-time nondeterministic algorithm. We guess nondeterministically a simplification Q of size at most k of P. We then check whether the Local-Hausdorff

distance between P and Q is at most δ. This check takes time $\mathcal{O}(n)$ as checking whether $\delta_H(P[i \dots j], \overline{v_i v_j}) \leq \delta$ can be done in time $\mathcal{O}(j-i)$, and adding up over all edges $\overline{v_i v_j}$ of Q yields total time $\mathcal{O}(n)$. If the answer to the checks is "yes", we accept otherwise we reject.

For the co-nondeterministic algorithm, we first nondeterministically guess a subset $E \subseteq E(\overrightarrow{K}_n)$. Moreover, for any $(i,j) \in E$ we guess an integer $w(i,j) \in \{i, \dots, j\}$. We perform two checks:

(C1) Check for all $(i,j) \in E$ whether the distance from $v_{w(i,j)}$ to $\overline{v_i v_j}$ is larger than δ, and

(C2) Check whether $d_{\overrightarrow{K_n} \setminus E}(1,n) > k$.

If the answer to both the checks are "yes", then we accept and otherwise we reject. We will now show that for any "yes" instance we always reject and for any "no" instance there is a particular good guess for which we accept.

Consider a "yes" instance. Suppose that some guessed edge $(i,j) \in E$ belongs to G_δ. Then, by Observation 5.3 there is no $w(i,j) \in \{i, \dots, j\}$ such that the distance from $v_{w(i,j)}$ to $\overline{v_i v_j}$ is larger than δ. Thus, the check (C1) fails and we reject. Otherwise, if every edge in E does not belong to G_δ, then we have that the edge set of G_δ belong to $\overrightarrow{K}_n \setminus E$ and thus $d_{\overrightarrow{K}_n \setminus E}(1,n) \leq d_{G_\delta}(1,n)$. Since we consider a "yes" instance, by Lemma 5.2 we have that $d_{G_\delta}(1,n) \leq k$. It follows that check (C2) fails. Hence, every "yes" instance is rejected.

Now consider a "no" instance. Suppose that our guessed subset $E \subseteq E(\overrightarrow{K}_n)$ is such that $\overrightarrow{K}_n \setminus E = G_\delta$. Note that for every $(i,j) \in E$ we have $\delta_H(\overline{v_i v_j}, P[i \dots j]) > \delta$ and thus by Observation 5.3 there exists a $w(i,j) \in \{i, \dots, j\}$ such that the distance of $v_{w(i,j)}$ to $\overline{v_i v_j}$ is larger than δ. Suppose that we guessed such vertices $w(i,j)$. Then, check (C1) clearly passes. Moreover, we have $d_{\overrightarrow{K}_n \setminus E}(1,n) = d_{G_\delta}(1,\delta)$, and since we consider a "no" instance, by Lemma 5.2 we have $d_{G_\delta}(1,n) > k$. Thus, check (C2) also passes. Hence, for any "no" instance there exists a guess for which we accept. □

Now combining Theorem 5.4 and Lemma 5.1 we have the following theorem for polyline simplification under Local-Hausdorff distance.

Theorem 5.5. *There exists no reduction* $(\mathrm{SAT}, 2^n) \leq^{\mathrm{det}}_{\mathrm{FGR}} (\text{Polyline Simplification}, n^3)$ *or* $(3\text{-}OV, n^3) \leq^{\mathrm{det}}_{\mathrm{FGR}} (\text{Polyline Simplification}, n^3)$ *unless NSETH fails.*

6 Discussion of the ∀∀∃-OV Hypothesis

The ∀∀∃-OV Hypothesis, that we introduced in this paper, is a special case of the following more general hypothesis (by setting $k = 3$ and $Q_1 = Q_2 = \forall$).

Quantified-k-OV Hypothesis: *Problem:* Fix quantifiers $Q_1, \dots, Q_{k-1} \in \{\forall, \exists\}$. Given sets $A_1, \dots, A_k \subseteq \{0,1\}^d$ of size n, determine whether $Q_1 a_1 \in A_1 \colon \dots Q_{k-1} a_{k-1} \in A_{k-1} \colon \exists a_k \in A_k$ such that $a_1, \dots, a_k$ are orthogonal.
Hypothesis: For any $k \geq 1$, any $Q_1, \dots, Q_{k-1}$, and any $\varepsilon > 0$, the problem cannot be solved in time $\mathcal{O}(n^{k-\varepsilon})$.

These problems were studied by Gao et al. [16], who showed that (even for every fixed k and $Q_1, \ldots, Q_{k-1}$) the Quantified-k-OV hypothesis implies the 2-OV hypothesis. Unfortunately, there is no reduction known in the opposite direction. In fact, Carmosino et al. [9] established barriers for a reduction in the other direction, see also the discussion of the Hitting Set[9] hypothesis in [3]. Hence, we cannot base the hardness of Quantified-k-OV on the more standard k-OV hypothesis.

It is well-known that the following Strong Exponential Time Hypothesis implies the k-OV hypothesis [25].

Strong Exponential Time Hypothesis (SETH) [21]: *Problem:* Given a q-CNF formula ϕ over variables $x_1, \ldots, x_n$, determine whether there exist $x_1, \ldots, x_n$ such that ϕ evaluates to true.
Hypothesis: For any $\varepsilon > 0$, there exists $q \geq 3$, such that the problem cannot be solved in time $O(2^{(1-\varepsilon)n})$.

Similarly, we can pose a hypothesis for Quantified Satisfiability, that implies the Quantified-k-OV hypothesis (by essentially the same proof as in [25]).

Quantified-SETH: *Problem:* Given a q-CNF formula ϕ over variables $x_1, \ldots, x_n$, determine whether for all $x_1, \ldots, x_{\alpha(1)n}$ there exist $x_{\alpha(1)n+1}, \ldots, x_{\alpha(2)n}$ such that ... such that for all $x_{\alpha(2s)n+1}, \ldots, x_{\alpha(2s+1)n}$ there exist $x_{\alpha(2s+1)n+1}, \ldots, x_n$ such that ϕ evaluates to true.
Hypothesis: For any $s \geq 0$, any $0 \leq \alpha(1) < \ldots < \alpha(2s+1) < 1$, and any $\varepsilon > 0$, there exists $q \geq 3$, such that the problem cannot be solved in time $O(2^{(1-\varepsilon)n})$.

Although Quantified Satisfiability is one of the fundamental problems studied in complexity theory (known to be PSPACE-complete), no algorithm violating Quantified-SETH is known.

Hence, Quantified-SETH and the Quantified-k-OV hypothesis are two hypotheses that are even stronger than the $\forall\forall\exists$-OV Hypothesis that we used in this paper to prove a conditional lower bound. The fact that even these stronger hypotheses have not been falsified in decades of studying these problems, we view as evidence that the $\forall\forall\exists$-OV Hypothesis is a plausible conjecture.

7 Conclusion

In this paper, we studied the complexity of three variants of the classical polyline simplification problem, namely, Local-Hausdorff simplification, Local-Fréchet simplification and Global-Fréchet simplification. We showed an $\mathcal{O}(n^3)$-time algorithm for the Global-Fréchet simplification, substantially improving the previous best algorithm. Although, the dynamic programming algorithms for the Local-Hausdorff simplification and Local-Fréchet simplification are significantly simpler than the dynamic programming algorithm for the Global-Fréchet simplification, we show that both the all the three variant (including the local variants) cannot be solved substantially faster unless $\forall\forall\exists$-OV Hypothesis is false. However,

[9] The Hitting Set problem considered in [3] is equivalent to $\forall\exists$-OV.

our lower bound does not apply when our underlying metric space is the Euclidean space. However, any subcubic algorithm needs to exploit the Euclidean distance heavily! Thus, settling the complexity of polyline simplification when the ambient space is the Euclidean space is a very interesting open problem.

Our lower bounds are applicable only in higher dimensions (when $d \in \mathcal{O}(\log(n))$). For small values of d, there are faster algorithms known for Local-Hausdorff simplification, however, to the best of our knowledge, there are no faster (sub-cubic) algorithms known for both Local-Fréchet and Global-Fréchet simplification (even when $d = 2$). We believe that faster algorithms should be possible in smaller dimensions and leave this as another tractable venue for future research.

References

[1] M. A. Abam, M. de Berg, P. Hachenberger, and A. Zarei. Streaming algorithms for line simplification. *Discrete & Computational Geometry*, 43(3):497–515, 2010.

[2] A. Abboud, R. R. Williams, and H. Yu. More applications of the polynomial method to algorithm design. In *SODA*, pages 218–230. SIAM, 2015.

[3] A. Abboud, V. V. Williams, and J. R. Wang. Approximation and fixed parameter subquadratic algorithms for radius and diameter in sparse graphs. In *SODA*, pages 377–391. SIAM, 2016.

[4] P. K. Agarwal, S. Har-Peled, N. H. Mustafa, and Y. Wang. Near-linear time approximation algorithms for curve simplification. *Algorithmica*, 42(3-4):203–219, 2005.

[5] H. Alt, A. Efrat, G. Rote, and C. Wenk. Matching planar maps. In *SODA*, pages 589–598. ACM/SIAM, 2003.

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

[7] G. Barequet, D. Z. Chen, O. Daescu, M. T. Goodrich, and J. Snoeyink. Efficiently approximating polygonal paths in three and higher dimensions. *Algorithmica*, 33(2):150–167, 2002.

[8] K. Buchin, M. Buchin, M. Konzack, W. Mulzer, and A. Schulz. Fine-grained analysis of problems on curves. *EuroCG, Lugano, Switzerland*, 2016.

[9] M. L. Carmosino, J. Gao, R. Impagliazzo, I. Mihajlin, R. Paturi, and S. Schneider. Nondeterministic extensions of the strong exponential time hypothesis and consequences for non-reducibility. In *ITCS*, pages 261–270. ACM, 2016.

[10] W. Chan and F. Chin. Approximation of polygonal curves with minimum number of line segments or minimum error. *International Journal of Computational Geometry & Applications*, 06(01):59–77, 1996.

[11] D. Z. Chen, O. Daescu, J. Hershberger, P. M. Kogge, N. Mi, and J. Snoeyink. Polygonal path simplification with angle constraints. *Comput. Geom.*, 32(3):173–187, 2005.

[12] M. de Berg, M. van Kreveld, and S. Schirra. Topologically correct subdivision simplification using the bandwidth criterion. *Cartography and Geographic Information Systems*, 25(4):243–257, 1998.

[13] D. H. Douglas and T. K. Peucker. Algorithms for the reduction of the number of points required to represent a digitized line or its caricature. *Cartographica*, 10(2):112–122, 1973.

[14] R. Estkowski and J. S. B. Mitchell. Simplifying a polygonal subdivision while keeping it simple. In *SoCG*, pages 40–49. ACM, 2001.

[15] S. Funke, T. Mendel, A. Miller, S. Storandt, and M. Wiebe. Map simplification with topology constraints: Exactly and in practice. In *ALENEX*, pages 185–196. SIAM, 2017.

[16] J. Gao, R. Impagliazzo, A. Kolokolova, and R. R. Williams. Completeness for first-order properties on sparse structures with algorithmic applications. In *SODA*, pages 2162–2181. SIAM, 2017.

[17] M. Godau. A natural metric for curves - computing the distance for polygonal chains and approximation algorithms. In *STACS*, pages 127–136, 1991.

[18] L. J. Guibas, J. Hershberger, J. S. B. Mitchell, and J. Snoeyink. Approximating polygons and subdivisions with minimum link paths. *Int. J. Comput. Geometry Appl.*, 3(4):383–415, 1993.

[19] J. Hershberger and J. Snoeyink. An $O(n \log n)$ implementation of the douglas-peucker algorithm for line simplification. In *SoCG*, pages 383–384. ACM, 1994.

[20] H. Imai and M. Iri. Polygonal approximations of a curve — formulations and algorithms. In *Computational Morphology*, volume 6 of *Machine Intelligence and Pattern Recognition*, pages 71 – 86. North-Holland, 1988.

[21] R. Impagliazzo, R. Paturi, and F. Zane. Which problems have strongly exponential complexity? *J. Comput. Syst. Sci.*, 63(4):512–530, 2001.

[22] A. Melkman and J. O'Rourke. On polygonal chain approximation. In *Computational Morphology*, volume 6 of *Machine Intelligence and Pattern Recognition*, pages 87 – 95. North-Holland, 1988.

[23] M. J. van Kreveld, M. Löffler, and L. Wiratma. On optimal polyline simplification using the hausdorff and fréchet distance. In *SoCG*, volume 99 of *LIPIcs*, pages 56:1–56:14. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, 2018.

[24] V. Vassilevska Williams. On some fine-grained questions in algorithms and complexity. In *Proceedings of the ICM*, 2018.

[25] R. Williams. A new algorithm for optimal constraint satisfaction and its implications. In *ICALP*, pages 1227–1237, 2004.