Approximate Discrete Fréchet distance: simplified, extended and structured

Ivor van der Hoog
DTU Compute, Technical University of Denmark, Denmark

Eva Rotenberg
DTU Compute, Technical University of Denmark, Denmark

Sampson Wong
University of Sydney, Australia

Abstract

The Fréchet distance is one of the most studied distance measures between curves $P$ and $Q$. The data structure variant of the problem is a longstanding open problem: Efficiently preprocess $P$, so that for any $Q$ given at query time, one can efficiently approximate their Fréchet distance. There exist conditional lower bounds that prohibit $(1 + \varepsilon)$-approximate Fréchet distance computations in subquadratic time, even when preprocessing $P$ using any polynomial amount of time and space. As a consequence, the problem has been studied under various restrictions: restricting $Q$ to be a (horizontal) segment, or requiring $P$ and $Q$ to be so-called realistic input curves.

We give a data structure for $(1 + \varepsilon)$-approximate discrete Fréchet distance in any metric space $X$ between a realistic input curve $P$ and any query curve $Q$. After preprocessing the input curve $P$ (of length $|P| = n$) in $O(n \log n)$ time, we may answer queries specifying a query curve $Q$ and an $\varepsilon$, and output a value $d(P, Q)$ which is at most a $(1 + \varepsilon)$-factor away from the true Fréchet distance between $Q$ and $P$. Thus, we give the first data structure that adapts to $\varepsilon$-values specified at query time, and the first data structure to handle query curves with arbitrarily many vertices. Our query time is asymptotically linear in $|Q| = m, \frac{1}{\varepsilon}, \log n$, and the realism parameter $c$ or $\kappa$.

The method presented in this paper simplifies and generalizes previous contributions to the static problem variant. Particularly, we show how to go from approximately deciding the Fréchet distance, to computing an approximate value without the (typically used) Well Separated Pair Decomposition of $(P, Q)$, which has $O(2^d n +dn \log n)$ construction time in $\mathbb{R}^d$, or parametric search, which, when applied to sublinear decision algorithms, commonly squares the decision time. We obtain subquadratic total running time in any metric space, even $\mathbb{R}^d$ for non-constant $d$. Our algorithm takes into account that one may not have oracle-access to exact distances in $X$, assuming oracle-access to approximate distances for some parameterised query cost. We obtain efficient queries (and therefore static algorithms) for Fréchet distance computation in high-dimensional spaces and other metric spaces (e.g., when $X$ is a graph under the shortest path metric). Our method supports subcurve queries at no additional cost.

2012 ACM Subject Classification Theory of computation → Design and analysis of algorithms; Theory of computation → Computational geometry

Keywords and phrases Fréchet, Hausdorff, data structures, approximation algorithms

Digital Object Identifier 10.4230/LIPIcs.SoCG.2023.XX

Related Version ArXiv:2201.02121

Funding Ivor van der Hoog: Supported by Independent Research Fund Denmark grant 2020-2023 (9131-00044B) “Dynamic Network Analysis”.
Eva Rotenberg: Partially supported by Independent Research Fund Denmark grant 2020-2023 (9131-00044B) “Dynamic Network Analysis” and the Carlsberg Young Researcher Fellowship CF21-0302 “Graph Algorithms with Geometric Applications”.

© . licensed under Creative Commons License CC-BY 4.0

Leibniz International Proceedings in Informatics
Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany
1 Introduction

The Fréchet distance $D_F(P,Q)$ is a popular metric for measuring the similarity between (polygonal) curves $P$ and $Q$ that reside in some metric space $X$. We assume that $P$ has $n$ vertices and $Q$ has $m$ vertices. Traditionally, $X$ is $\mathbb{R}^2$ under the $L_2$ metric, and $n > m$. The Fréchet distance is often intuitively defined through the following metaphor: suppose that we have two curves that are traversed by a person and their dog. Over all possible traversals by both the person and the dog, consider the length of their connecting leash (measured by the metric over $X$). What is the minimum length of this connecting leash? The Fréchet distance has many applications; in particular in the analysis and visualization of movement data [8, 11, 32, 42]. It is a versatile distance measure that can be used for a variety of objects, such as handwriting [37], coastlines [34], outlines of geometric shapes in geographic information systems [17], trajectories of moving objects, such as vehicles, animals or sports players [4, 11, 28, 38], air traffic [3] and also protein structures [31]. The two most-studied variants are the continuous and discrete Fréchet distance (based on whether the entities traverse a polygonal curve in a continuous manner or vertex-by-vertex).

Fréchet distance computation and lower bounds. Alt and Godau [1] were the first to study the Fréchet distance from a computational perspective. They compute the continuous Fréchet distance in $\mathbb{R}^2$ under the $L_2$ metric in $O(mn \log(n + m))$ time. Eiter and Manila [28] showed how to compute the discrete Fréchet distance in $\mathbb{R}^2$ in $O(nm)$ time, which was later improved to $O(nm \log \log nm) / \log nm$ by Buchin et al. [9]. Typically, $O(nm)$ (or ‘quadratic’) time is considered costly. However, Bringmann [5] showed that this cost is unavoidable: conditioned on SETH, one cannot compute even a $(1 + \varepsilon)$-approximation of $D_F(P,Q)$ between curves in $\mathbb{R}^2$ under the $L_1, L_2, L_\infty$ metric faster than $\Omega((nm)^{1 - \delta})$ time for any $\delta > 0$. This lower bound was extended by Bringmann and Mulzer [7] to intersecting curves in $\mathbb{R}^1$. Buchin, Ophelders and Speckmann [10] show this lower bound for pairwise disjoint planar curves in $\mathbb{R}^2$ and intersecting curves in $\mathbb{R}^1$. Recently, at SoCG 2022, Driemel, van der Hoog and Rotenberg [22] extended the lower bound to where $P$ and $Q$ are paths in a weighted planar graph (under the shortest path metric).

Avoiding lower bounds. The lower bounds can be circumvented when allowing relatively large approximation factors. Bringmann and Mulzer [7] approximate the discrete Fréchet distance in $\mathbb{R}^d$ for constant $d$. They present an $\alpha$-approximation algorithm for the discrete Fréchet distance, that runs in time $O(n \log n + n^2/\alpha)$, for any $\alpha$ in $[1, n]$. This was improved by Chan and Rahmati [13] to $O(n \log n + n^2/\alpha^2)$ for any $\alpha$ in $[1, \sqrt{n \log n}]$. Recently at SODA 2023, van der Horst, van Kreveld, Ophelders, and Speckmann [10] show how to compute an $\alpha$-approximation in the same setting in $O((n + nm/\alpha) \log^3 n)$ time for the continuous variant.

Driemel, Har-Peled and Wenk [14] propose another way to circumvent such lower bounds, assuming that both input curves come from a well-behaved class of curves. They introduce three classes (each with their own parameter) where a curve $P$ is:

- $\kappa$-straight if for every $i, j$ the length of the subcurve from $p_i$ to $p_j$ is $\ell([p_i, p_j]) \leq \kappa \cdot d(p_i, p_j)$,
- $c$-packed if for every ball $B$ in $X$ with radius $r$: the length $\ell(P \cap B) \leq c \cdot r$,
- $\phi$-low-dense if for every ball $B$ in $X$ with radius $r$, there exist at most $\phi$ subcurves $P[i, j] \subseteq P \cap B$ such that $P[i, j] \geq r$. 


Any $c$-straight curve is also $O(c)$-packed. Parametrized by $\varepsilon$, $\phi \in O(1)$, $c$ and $\kappa = O(c)$, Driemel, Har-peled and Wenk \[19\] compute a $(1 + \varepsilon)$-approximation of the continuous Fréchet distance between a pair of $\kappa$-straight or $c$-packed curves in $\mathbb{R}^d$ under the $L_1, L_2, L_\infty$ metric for constant $d$ in $O(\frac{cm}{\varepsilon} + cn \log n)$ time. Their result for $c$-packed and $c$-straight curves was improved by Bringmann and Künnemann \[6\] to $O(\frac{cm}{\varepsilon} \log \varepsilon^{-1} + cn \log n)$. Driemel, van der Hoog and Rotenberg \[22\] study the problem where $\mathcal{X}$ is a weighted graph $G = (V, E)$ and the distance metric is the shortest path metric. If $P$ is a $c$-straight path and $Q$ is any walk in $\mathcal{X}$, they compute a $(1 + \varepsilon)$-approximation of $D_\mathcal{F}(P, Q)$ in $O(\frac{|G|}{\varepsilon} \log |G| + |E| \log |E| + \frac{cm}{\varepsilon} \log |G|)$ time. Note that the complexity of $G$ may easily exceed $n$ and $m$. Realistic input assumptions have been applied to other geometric problems, e.g. for robotic navigation in $\phi$-low-dense environments \[41\], and map matching of $\phi$-low-dense graphs \[14\] or $c$-packed graphs \[28\].

Deciding, computing, and subcurves. We make a distinction between three problem variants: for deciding the Fréchet distance, we are given a value $\rho$ and two curves $P$ and $Q$ and we ask whether $D_\mathcal{F}(P, Q) \leq \rho$. This decision variant is often solved through navigating an $n$ by $m$ ‘free space diagram’. To convert the decision variant into computing the Fréchet value (the ‘value variant’), two techniques are commonly used. The first technique is binary search for values of $\rho$, chosen from a well-separated pair decomposition (WSPD) between $P$ and $Q$. However, computing a WSPD in $\mathbb{R}^d$ under the $L_1, L_2, L_\infty$ metric takes $O(2^d(n + m) + d(n + m) \log(n + m))$ time \[20\] Ch. 4 (hence, most results assume constant dimension)\[2\]. The second technique is parametric search \[35\]. For decision variants that have a sublinear running (or query) time of $T$, the running time of parametric search is (usually) $O(T^2)$ \[29\] \[42\]. The final variant that we consider are subcurve variants. For both the decision or value variant, subcurve computations restrict $P$ to a subcurve from a point $s$ to a point $t$. The goal is to report the Fréchet distance $D_\mathcal{F}(P[s, t], Q)$ for the subcurve of $P$ from $s$ to $t$. Typically, this is obtained by storing $P$ in a balanced tree, with a data structure on each node. The output can subsequently be found with a factor $O(\log n)$ more time.

Data structures for Fréchet distance. A big open question is whether we can store $P$ in a data structure, for efficient (approximate) Fréchet distance queries for any query $Q$. This topic received considerable attention throughout the years \[2, 12, 15, 18, 20, 26, 28, 29\]. Recently at SODA 2023, Gudmundsson, Seybold and Wong \[28\] answer this question negatively for arbitrary curves in $\mathbb{R}^2$: showing that even when preprocessing $P$ using arbitrary polynomial space and preprocessing time, the $\Omega((nm)^{1-\delta})$ conditional lower bound for queries holds.

Surprisingly, even in very restricted settings efficient results are difficult to obtain. Gudmundsson et al. \[29\] assume that $Q$ is a single horizontal segment in $\mathbb{R}^2$. They present an $O(n \log n)$ sized data structure that can decide if the continuous Fréchet distance to any horizontal segment $Q$ is smaller than a given value $\rho$ in $O(\log^4 n)$ time. It follows through parametric search, that they can compute $D_\mathcal{F}(P, Q)$ in $O(\log^4 n)$ time. Through another application of parametric search, they do subcurve queries in $O(\log^3 n)$ time. De Berg et al. \[16\] present an $O(n^2)$ size data structure that restricts ‘only’ the orientation of the query segment to be horizontal. Queries are supported in $O(\log^2 n)$ time, and even subcurve queries are allowed (in that case, using $O(n^2 \log^2 n)$ space).

---

1 We note that for high $d$, one can alternatively compute a WSPD through sorting all pairs in $P \times Q$. 
Recently at ESA 2022, Buchin et al. [12] improve this result to using only \(O(n \log^2 n)\) space, where queries take \(O(\log n)\) time. For arbitrary query segments, they present an \(O(n^{4+\delta})\) size data structure that supports (subcurve) queries to arbitrary segments in \(O(\log^4 n)\) time. Driemel and Har-peled [13] create a data structure to store any curve in \(\mathbb{R}^d\) for constant \(d\). They preprocess \(P\) in \(O(n \log^3 n)\) time and \(O(n \log n)\) space. For any query \((Q, \varepsilon, i, j)\) they can create a \((3+\varepsilon)\)-approximation of \(D_F(P[i, j], Q)\) in \(O(m^2 \log n \log (m \log n))\) time.

There exist some data structures for specifically the discrete Fréchet distance. Driemel, Psarros and Schmidt [21] fix \(\varepsilon\) and an upper bound on \(m\) beforehand (and assume \(m < < n\)). They store any curve \(P\) in \(\mathbb{R}^d\) for constant \(d\) using \(O((m \log (\frac{1}{\varepsilon})^{\delta})^m)\) space and preprocessing, to answer \((1+\varepsilon)\)-approximate Fréchet distance queries in \(O(m^2 + \log \frac{1}{\varepsilon})\) time. Filtser [27] gives the corresponding data structure for the discrete Fréchet distance. Recently, at SODA 2022, Filtser and Filtser [25] study the same setting: storing \(P\) in \(O((\frac{1}{\varepsilon})^d m \log \frac{1}{\varepsilon})\) space, to answer \((1+\varepsilon)\)-approximate Fréchet distance queries in \(O(m \cdot d)\) time.

**Contribution.** For any \(c\)-packed curve \(P = (p_1, \ldots, p_n)\) in some metric space \(\mathcal{X}\), both specified at construction time, we construct a data structure for \((1+\varepsilon)\)-approximate discrete Fréchet distance between (subcurves of) \(P\) and other curves in \(\mathcal{X}\). We facilitate different queries, each specifying an integer \(m\), an arbitrary curve \(Q\) with \(m\) vertices, and a desired \(\varepsilon\)-value. Decision queries furthermore specify some \(\rho \geq 0\) and output one of two (not mutually exclusive) conclusions: \(D_F(P, Q) > \rho\) or \(D_F(P, Q) \leq (1+\varepsilon)\rho\).

We revisit and simplify the argument by Driemel, Har-Peled and Wenk [19] for the static problem. We show that it suffices that \(P\) is \(c\)-packed, i.e. no realistic input assumption is required on \(Q\). Additionally, our argument assumes access only to a \((1+\alpha)\)-approximate distance oracle with \(T_\alpha\) query time. Thus, we generalise the previous argument to any metric space with approximate distance oracles with non-constant query time. Examples include \(\mathcal{X} = \mathbb{R}^d\) subject to the \(L_2\) metric under floating point arithmetic, and \(\mathcal{X}\) as a weighted graph under the shortest path metric. We can preprocess \(P\) using \(O(n)\) space and time, and answer any decision query \((Q, \varepsilon, \rho)\) using \(O(\frac{2n}{\varepsilon} \log n)\) distance queries (which take constant time in \(\mathbb{R}^d\) for constant \(d\)). Our approach immediately works for subcurve queries. Next, we show how to use decision queries to compute a \((1+\varepsilon)\)-approximation of \(D_F(P, Q)\) using a factor \(O(\log n)\) more time (both for preprocessing and queries). Previous works compute a \((1+\varepsilon)\)-approximation using a Well-Separated Pair Decomposition (WSPD) between \(P\) and \(Q\). Crucially, we show that it suffices to map only \(P\) to a curve \(P' \subset \mathbb{R}^1\) and compute only a WSPD between \(P'\) and \(P'\) itself in \(O(n \log n)\) time. This technique may be of independent interest, and already improves computation times for Fréchet distance in \(\mathbb{R}^{d \geq 0}\) [3] [19].

We give the first data structure for \((1+\varepsilon)\)-approximate discrete Fréchet distance queries between curves with arbitrarily many vertices. Additionally, we are the first to support constant-factor approximate Fréchet distance queries in \(o(m^2)\) time, while allowing the number \(m\) to be part of the query input. In contrast to previous techniques, we do not require that \(\varepsilon\) is fixed at preprocessing time. One strength of our technique is that they are remarkably simple. We do not rely upon complicated techniques such as parametric search [28] [29], higher-dimensional envelopes [12], or advanced path-simplification structures [13] [21] [25]. Thus, our techniques are not only generally applicable, but also implementable. In the appendix we demonstrate that our algorithmic skeleton is broadly applicable. In Appendix A we apply our framework to approximate the Hausdorff distance \(D_H(P, Q)\) (which, surprisingly, is more difficult to do). In Appendix B we apply our WSPD-technique to the map matching problem, improving the query time of the recent SODA paper by Gudmundsson, Seybold and Wong [25] from \(O(m \log m \cdot (\log^4 n + c^4 \varepsilon^{-8} \log^2 n))\) to \(O(m (\log n + \log \varepsilon^{-1}) \cdot (\log^2 n + c^2 \varepsilon^{-4} \log n))\).
2 Preliminaries

Notation. We consider some metric space $\mathcal{X}$ (e.g., $\mathcal{X} \leftarrow \mathbb{R}^2$, or $\mathcal{X}$ is some weighted graph). For any two elements $a, b \in \mathcal{X}$ we denote by $d(a, b)$ their distance in $\mathcal{X}$. A curve $P$ in $\mathcal{X}$ is any ordered sequence of points in $\mathcal{X}$ (Figure 1). We refer to such points as vertices. For any curve $P$ with $n$ vertices, for any integers $i, j \in [n]$ with $i < j$ we denote by $P[i, j]$ the subcurve from $p_i$ to $p_j$. We denote by $|P[i, j]| = (j - i + 1)$ the size of the subcurve (the number of vertices in the subcurve) and by $\ell(P[i, j]) := \sum_{k=i}^{j-1} d(p_k, p_{k+1})$ its length.

Distance and distance oracles. Throughout this paper, we assume that for any $\alpha > 0$ we have access to some $(1 + \alpha)$-approximate distance oracle. This is a data structure $D^a_{\alpha}$ that for any two $a, b \in \mathcal{X}$ can report a value $d^a(a, b) \in [(1 - \alpha)d(a, b), (1 + \alpha)d(a, b)]$ in $O(T_{\alpha})$ time. To distinguish between inaccuracy as a result of our algorithm and as a result of our oracle, we refer to $d^a(a, b)$ as the perceived value (as opposed to an approximate value).

- Oracles 1. We present some examples of distance oracles:
  - For $\mathcal{X} \subseteq \mathbb{R}^d$ under the $L_1, L_2, L_\infty$ metric in real-RAM we can compute the exact $d(a, b)$ in $O(d)$ time. Thus, for any $\alpha$, we have an oracle $D^a_{\alpha}$ with $T_{\alpha} = O(d)$ query time.
  - For $\mathcal{X} \subseteq \mathbb{R}^d$ under the $L_2$ metric executed in word-RAM, we can compute $d(a, b)$ in $O(d^2)$ expected time. Thus, we have an oracle $D^a_{\alpha}$ with $O(d^2)$ expected query time.
  - For any $\mathcal{X} \subseteq \mathbb{R}^d$ under the $L_2$ metric in word-RAM, we can $(1 + \alpha)$-approximate the distance between two points in $T_{\alpha} = O(d \log \alpha^{-1})$ worst case time using Taylor expansions.
  - For $\mathcal{X}$ a planar weighted graph, Long and Pettie [35] show how to store $\mathcal{X}$ with $N$ vertices using $O(N^{1+o(1)})$ space, to answer exact distance queries in $O((\log(N))^{2+o(1)})$ time.
  - For $\mathcal{X}$ as an arbitrary weighted graph, Thorup [39] shows that it is possible to compute $(1+\alpha)$-approximate distance oracle in $O(N/\alpha \log N)$ time and space, and with a query-time of $O(1/\alpha)$. Note that $\alpha$ must be known at the time of the construction.

![Figure 1](image-url)

(a) $\mathcal{X}$ may be Euclidean space. (b) $\mathcal{X}$ may be a weighted graph under the shortest path metric. Observe that in full generality when $\mathcal{X}$ is a graph, any sequence of vertices may be a curve.

Discrete similarity measures between curves. The Hausdorff distance can be defined between any two sets $P$ and $Q$. The discrete directed Hausdorff distance from $P$ to $Q$ is:

$$D_H(P \rightarrow Q) := \max_{p_i \in P} \min_{q_j \in Q} d(p_i, q_j).$$

2 Comparing real-valued input in word-RAM requires expected $O(d \log(d \cdot n))$ bits per coordinate (Theorem 3.1 [24]). Assuming $O(\log(d \cdot n))$ word size, comparisons take expected $O(d^2)$ time.
The discrete Hausdorff distance $D_H(P, Q)$ is the maximum of $D_H(P \rightarrow Q)$ and $D_H(Q \rightarrow P)$. To define the discrete Fréchet distance we first define discrete walks: given two curves $P$ and $Q$ in $\mathcal{X}$, we denote by $[n] \times [m]$ the $n \times m$ integer lattice. We say that an ordered sequence $F$ of points in $[n] \times [m]$ is a discrete walk if for every consecutive pair $(i, j), (k, l) \in F$, we have $k \in \{i - 1, i, i + 1\}$ and $l \in \{j - 1, j, j + 1\}$. It is furthermore $xy$-monotone when we restrict to $k \in \{i, i + 1\}$ and $l \in \{j, j + 1\}$. Let $F$ be a discrete walk from $(1, 1)$ to $(n, m)$. The cost of $F$ is the maximum over $(i, j) \in F$ of $d(p_i, q_j)$. The discrete Fréchet distance is the minimum over all $xy$-monotone walks $F$ from $(1, 1)$ to $(n, m)$ of its associated cost:

$$D_F(P, Q) := \min_F \text{cost}(F) = \min_F \max_{(i, j) \in F} d(p_i, q_j).$$

The discrete weak Fréchet distance $D^w_F$ is defined analogously, not requiring $F$ to be $xy$-monotone. Apart from lower bounds, all mentioned results for Fréchet distance will apply to the weak variant. In this paper we, given a $(1 + \alpha)$-approximate distance oracle, also define what we call the perceived similarity between curves: $D^\circ_H, D^\circ_F$ and $D^\circ_w$. These similarity measures are defined analogously, where $d(p_i, q_j)$ is replaced by $d^\circ(p_i, q_j)$ in their definition.

**Free space matrix (FSM).** The FSM for a fixed $\rho^* \geq 0$ is a $|P| \times |Q|$, $(0, 1)$-matrix where the cell $(i, j)$ is zero if and only if the distance between the $i$'th point in $P$ and the $j$'th point in $Q$ is at most $\rho^*$. Per definition, $D_F(P, Q) \leq \rho^*$ if and only if there exists a ($xy$-monotone) discrete walk $F$ from $(1, 1)$ to $(n, m)$ where for all $(i, j) \in F$: the cell $(i, j)$ is zero.

**Problem statement.** Our data structure input is a curve $P = (p_1, p_2, \ldots, p_n)$. For a given similarity measure $D_\alpha$, we make a distinction between three types of (approximate) queries: decision queries receive as input a curve $Q = (q_1, q_2, \ldots, q_m)$ in $\mathcal{X}$ and a real value $\rho$. We want to output a Boolean indicating whether $D_\alpha(P, Q) \leq \rho$. Value queries receive as input a curve $Q$ and want to output a value $c$ with $c = D_\alpha(P, Q)$. Subcurve queries compute the similarity measure to a subcurve $P[i, j]$. The number of vertices $m$ of $Q$ is part of the query input and may vary. Formally, given a similarity measure $D_\alpha$, we preprocess $P$ subject to:

- **A-decision**$(Q, \varepsilon, \rho)$: for $\rho \geq 0$ and $0 < \varepsilon < 1$ outputs a Boolean concluding either $D_\alpha(P, Q) > \rho$, or $D_\alpha(P, Q) \leq (1 + \varepsilon)\rho$ (these two options are not mutually exclusive).
- **A-value**$(Q, \varepsilon)$: for $0 < \varepsilon < 1$ outputs a value in $[(1 - \varepsilon)D_\alpha(P, Q), (1 + \varepsilon)D_\alpha(P, Q)]$.
- **Subcurve**$(Q, \rho, \varepsilon, i, j)$: considers a subcurve $P[i, j]$. We define two queries equivalent to the two above for similarity between the subcurve $P[i, j]$ and a query $Q$.

We want a solution that is efficient in time and space, where time and space is measured in units of $\varepsilon, n, m, \rho$ and the distance oracle query time $T_\alpha$.

**Curve simplification.** To efficiently compute the approximate Fréchet distance between two curves, Driemel, Har-peled and Wenk [19] introduce the concept of $\mu$-simplified curves. For a parameter $\mu \in \mathbb{R}$ one can construct a curve $P^\mu$ as follows. Start with the initial vertex $p_1$, and set this as the current vertex $p_1$. Next, scan the polygonal curve to find the first vertex $p_j$ such that $\ell(P[i, j]) > \mu$. Add $p_j$ to $P^\mu$, and set $p_j$ as the current vertex. Continue this process until we reach the end of the curve. Finally, add the last vertex $p_n$ to $P^\mu$. Driemel, Har-peled and Wenk [19] observe any $\mu$-simplified curve $P^\mu$ can be computed in linear time and $D_F(P^\mu, P) < \varepsilon$. This leads to the following approximate decision algorithm:

1. Given $P, \varepsilon, Q$ and $\rho$, construct $P^{\varepsilon/2}$ and $Q^{\varepsilon/2}$ in $O(n + m)$ time.
2. Denote by $Z$ the number of zeroes in the FSM between $P^{\varepsilon/2}$ and $P^{\varepsilon/2}$ for $\rho^* = (1 + \varepsilon/2)\rho$. 

The problem for $\rho^* \geq 0$ is a $|P| \times |Q|$, $(0, 1)$-matrix where the cell $(i, j)$ is zero if and only if the distance between the $i$'th point in $P$ and the $j$'th point in $Q$ is at most $\rho^*$. Per definition, $D_F(P, Q) \leq \rho^*$ if and only if there exists a ($xy$-monotone) discrete walk $F$ from $(1, 1)$ to $(n, m)$ where for all $(i, j) \in F$: the cell $(i, j)$ is zero.
3. Using DFS over the FSM, and $O(Z)$ distance computations, test if $D^p(\overline{P^q}, \overline{Q^q}) \leq \rho^*$. 
   - They prove that: if yes then $D^p(P, Q) \leq (1 + \varepsilon)\rho$. If no then $D^p(P, Q) > \rho$.
   - They upper bound $Z$ by $O\left(\frac{c(n+m)}{\varepsilon}\right)$.

In this paper, we show that it suffices to assume that only $P$ is $c$-packed and upper bound $Z$ by $O\left(\frac{cm}{\varepsilon}\right)$ instead. We extend the analysis to work with approximate distance oracles. Finally, we show a data structure to execute step 3 in time independent of $|P| = n$.

**From decision variant to computing Fréchet distance.** Suppose that we are given an algorithm to compute $A$-decision($Q, \varepsilon, \rho$) in $O(T)$ time. Then this may be used to answer $A$-value($Q, \varepsilon, \rho$) in $O(T^2)$ time through parametric search [28, 29], which is not so attractive. Driemel, Har-peled and Wenk [19] have a static algorithm for $A$-decision($Q, \varepsilon, \rho$) that takes $O\left(\frac{c(n+m)}{\varepsilon}\right)$ distance computations. They perform an approximate binary search for the appropriate $\rho$ over events where either the simplification of the input curves change, or when the reachability in the free space matrix changes. For the continuous Fréchet distance, we have vertex-vertex events and vertex-edge events. We will focus on the vertex-vertex events as these are the most relevant to the discrete Fréchet distance.

For vertex-vertex events, they begin by constructing a well-separated pair decomposition (WSPD) on $(P, Q)$. This takes $O(2^d n + dn \log n)$ time (assuming $n \geq m$) and partitions $(P, Q)$ into $O(2^d n)$ pairs $(P_i, Q_i)$. For each pair $P_i \subseteq P, Q_i \subseteq Q$ there exists a value $c_i$ with $c_i < d(p, q) < 2 \cdot c_i$ for all $(p, q) \in P_i \times Q_i$. This provides a set of $O(2^d n)$ candidate choices for $\rho$. Naively, binary searching over these values takes $O(T \cdot \log(2^d n))$ time. Some further observations reduce their running time to $O(d \cdot n \cdot (\frac{2^d}{\varepsilon} + c \log n))$ total time (assuming distance computation takes $O(d)$ time and $n \geq m$). Bringmann and Künnemann go from the decision variant to computations using [19]. Hence, their running time implicitly becomes $O(2^d n + d \cdot \frac{c n}{\varepsilon} + d \cdot cn \log(2^d n))$. Using a WSPD has three downsides:

1. It is only known how to compute a WSPD for doubling metrics [43].
2. For non-constant dimensions $d$, computing a WSPD dominates the running time.
3. Computing a WSPD between $P$ and $Q$ takes $\Omega(|P|)$ time: which is undesirable for queries.

Due to downside 1, Driemel, van der Hoog and Rotenberg [19] approximate the Fréchet distance in a graph through binary search over the sorted edges of $G$ (which is costly).

**Results.** In Section 3 we study computing the Fréchet distance in a data structure setting. Just as for previous works, our results also apply to the weak Fréchet distance as we show:

**Theorem 1.** Let $\mathcal{X}$ be a metric space and $D^p_X$ be a $(1 + \alpha)$-approximate distance oracle with $O(T_{\alpha})$ query time. Let $P = (p_1, \ldots, p_n)$ be any $c$-packed curve in $\mathcal{X}$. We can store $P$ using $O(n)$ space and preprocessing time, such that for any curve $Q = (q_1, \ldots, q_m)$ in $\mathcal{X}$ and any $\rho > 0$ and $0 < \varepsilon < 1$, we can answer $A$-decision($Q, \varepsilon, \rho$) for the Fréchet distance in:

$$O\left(\frac{c \cdot m}{\varepsilon} \cdot (T_{\varepsilon/6} + \log n)\right) \text{ time.}$$

**Corollary 2.** Using Oracles [7] it follows for a $c$-packed $P$ and the Fréchet distance that:

- For $\mathcal{X} = \mathbb{R}^d$ under $L_1, L_2, L_\infty$ metric in the real-RAM model, we can store $P$ using $O(n)$ space and preprocessing to answer $A$-decision($Q, \varepsilon, \rho$) in $O\left(\frac{cn}{\varepsilon} \cdot (d + \log n)\right)$ time.

- Improving the static $O(d \cdot \frac{cm}{\varepsilon} + d \cdot cn \log n)$ time by [40] when $n > m$. 
For $X = \mathbb{R}^d$ under $L_2$ in word-RAM, we can store $P$ using $O(n)$ space and preprocessing, to answer $A$-decision($Q, \varepsilon, \rho$) in $O\left(\frac{cm}{\varepsilon} \cdot (d \log \varepsilon^{-1} + \log n)\right)$ worst case time.

- Improving upon the static expected $O(d^2 \cdot \frac{cm}{\varepsilon} + d^2 \cdot cn \log n)$ time by [6].

- For $X$ a planar graph under the shortest path metric, we can store $P$ using $O(N^{1+o(1)})$ space and preprocessing, to answer $A$-decision($Q, \varepsilon, \rho$) in $O\left(\frac{cm}{\varepsilon} \cdot \log^{2+o(1)} N\right)$ time.

- Generalising the static $O\left(N^{1+o(1)} + \frac{cm}{\varepsilon} \log^{2+o(1)} N\right)$ time for $c$-straight curves by [23].

- For $X$ a graph under the shortest path metric, we can fix $\varepsilon$ and store $P$ using $O\left(\frac{N}{\varepsilon} \log N\right)$ space and preprocessing, to answer $A$-decision($Q, \varepsilon, \rho$) in $O\left(\frac{cm}{\varepsilon} \cdot (\varepsilon^{-1} + \log n)\right)$ time.

- Generalising the static $O\left(\frac{N}{\varepsilon} \log N + \frac{cm}{\varepsilon}\right)$ time for $c$-straight curves by [23].

We also use Theorem [1] to answer value queries. To this end, we develop a new technique that circumvents computing a WSPD between $X$ and $Q$, as we obtain the following:

**Theorem 3.** Let $X$ be a metric space and $D_X$ be a $(1 + \alpha)$-approximate distance with $O(T_n)$ query time. Let $P = (p_1, \ldots, p_n)$ be any c-packed curve in $X$. We can store $P$ using $O(n)$ space and $O(n \log n)$ preprocessing time, such that for any curve $Q = (q_1, \ldots, q_m)$ in $X$ and any $0 < \varepsilon < 1$, we can answer $A$-Value($Q, \varepsilon$) for the Fréchet distance in:

$$O\left(\frac{c \cdot m}{\varepsilon} \cdot \log n \cdot \left(T_{n/6} + \log \frac{c \cdot m}{\varepsilon} + \log n\right)\right) \text{ time.}$$

**Corollary 4.** Using Oracles [1], it follows for a c-packed $P$ and the Fréchet distance that:

- For $X = \mathbb{R}^d$ in the real-RAM model, we can store $P$ using $O(n)$ space and $O(n \log n)$ preprocessing, to answer $A$-value($Q, \varepsilon$) in $O\left(\frac{cm}{\varepsilon} \log n (d + \log \frac{cm}{\varepsilon} + \log n)\right)$ time.

- Improving the static $O(2^d n + d \cdot \frac{cm}{\varepsilon} + d^2 \cdot cn \log n)$ time by [6] when $n$ or $d$ is large.

- Improving the dynamic $O((1 + \frac{1}{\varepsilon})^d m \log \frac{1}{\varepsilon})$ space solution with $O(m \cdot d)$ query time by [23], for $Q$ of arbitrary length, when $P$ is packed (even for non-constant $c$).

- For $X = \mathbb{R}^d$ under the $L_2$ metric in word-RAM, we can store $P$ using $O(n)$ space and preprocessing, to answer $A$-value($Q, \varepsilon$) in $O\left(\frac{cm}{\varepsilon} \log n (d \log n + \log \frac{cm}{\varepsilon})\right)$ time.

- Improving upon the static expected $O(2^d n + d^2 \cdot \frac{cm}{\varepsilon} + d^3 \cdot cn \log n)$ time by [6].

- For $X$ a planar graph under the SP metric, we can store $P$ using $O(N^{1+o(1)})$ space and preprocessing, to answer $A$-value($Q, \varepsilon$) in $O\left(\frac{cm}{\varepsilon} \cdot \log n \cdot (\log^{2+o(1)} N + \log \frac{cm}{\varepsilon})\right)$ time.

- Improving the static $O\left(N^{1+o(1)} + |E| \log |E| + \frac{cm}{\varepsilon} \log^{2+o(1)} N \log |E|\right)$ time by [23].

- For $X$ a graph under the SP metric, we can fix $\varepsilon$ and store $P$ using $O\left(\frac{N}{\varepsilon} \log N\right)$ space and preprocessing, to answer $A$-value($Q, \varepsilon$) in $O\left(\frac{cm}{\varepsilon} \cdot \log n \cdot (\varepsilon^{-1} + \log \frac{cm}{\varepsilon} + \log n)\right)$ time.

- Improving the static $O\left(\frac{N}{\varepsilon} \log N + |E| \log |E| + \frac{cm}{\varepsilon} \log |E|\right)$ time by [23].

We briefly note that all our results are also immediately applicable to subcurve queries:

**Corollary 5.** All results obtained in Section [1] can answer the subcurve variants of the $A$-decision and $A$-value queries for any $i, j \in [n]$ at no additional cost.

The appendix is dedicated to applying our techniques beyond the Fréchet distance. In Appendix [A] we show that we can approximate the Hausdorff distance also. In Appendix [B] we show that your WSPD technique can be used to improve the query time from [28] from $O(m \log m \cdot (\log^4 n + c^2 \log^2 n))$ to $O(m \log n \cdot (\log^4 n + c^2 \log^2 n \cdot \varepsilon^{-1}) \cdot (\log^3 n + c^2 \varepsilon^{-4} \log n))$. 
3 Approximate Discrete Fréchet distance

In this section we simultaneously approximate the weak Fréchet distance and the Fréchet distance. For ease of exposition, we focus on the Fréchet distance, and indicate where algorithms are different for the weak Fréchet distance. We denote by $D^P_{\alpha}$ a $(1+\alpha)$-approximate distance oracle over our metric space $X$. We receive as input some curve $P = (p_1,\ldots,p_n)$ in $X$ which is c-packed in $X$ for some $c \geq 1$. The goal is to preprocess $P$ to:

- answer A-decision$(Q,\varepsilon,\rho)$ for any curve $Q = (q_1,\ldots,q_m)$, $\rho > 0$ and $0 < \varepsilon < 1$,
- answer A-value$(Q,\varepsilon)$ for any curve $Q = (q_1,\ldots,q_m)$ and $0 < \varepsilon < 1$.

We obtain this result in four steps. In Section 3.1 we define a $(0,1)$-matrix $M^{A\times Q}_\rho$ for any two curves $A$ and $Q$ and any $\rho^* > 0$. We show that if $A$ is the $\mu$-simplified curve $P^\mu$ for some convenient $\mu$, then the number of zeroes in $M^{P^\mu\times Q}_\rho$ is bounded. In Section 3.2 we show that we can answer A-decision$(Q,\varepsilon,\rho)$ through inspecting all the zeroes the matrix $M^{P^\mu\times Q}_\rho$ for some convenient choice of $\mu$ and $\rho^*$. In Section 3.3 we show a data structure that stores $P$ to answer A-decision$(Q,\varepsilon,\rho)$ through obtaining and navigating $M^{P^\mu\times Q}_\rho$ for convenient $\mu$ and $\rho^*$. Finally, in Section 3.4 we extend this solution to answer A-value$(Q,\varepsilon)$.

3.1 Perceived free space matrix and free space complexity

We define the perceived free space matrix to help answer A-decision queries. Given two curves $(A,Q)$ and some $\rho$, we construct an $|A| \times |Q|$ matrix we call the perceived free space matrix $M^{A\times Q}_\rho$. The $i$'th column corresponds to the $i$'th element $A[i]$ in $A$. We assign to each matrix cell $M^{P^\mu\times Q}_\rho[i,j]$ the integer $0$ if $d^\mu(A[i],q_j) \leq \rho$ and integer $1$ otherwise.

▶ Observation 6. For all $0 < \varepsilon < 1$, $\rho^* > 0$, and curves $A$ and $Q$, the discrete Fréchet distance between $A$ and $Q$ is at most $\rho^*$ if and only if there exists an $(xy$-monotone) discrete walk $F$ from $(1,1)$ to $(|A|,|Q|)$ where $\forall(i,j) \in F, M^{A\times Q}_\rho[i,j] = 0$.

Computing $D^P_{\alpha}(P^\mu,Q)$. Previous results for approximating the Fréchet distance upper bound for any choice of $\rho$, the number of zeroes in the FSM between $P^\mu$ and $Q^\mu$. We show something stronger: we consider the perceived FSM instead, and introduce a new parameter $k \geq 1$ (which we leverage to be enable approximate distance oracles). For any value $\rho$ and some simplification value $\mu \geq \frac{\kappa^\mu}{\varepsilon^2}$, we subsequently upper bound the number of zeroes in the perceived FSM $M^{P^\mu\times Q}_\rho$ for the conveniently chosen $\rho^* = (1 + \frac{\kappa}{\varepsilon})\rho$.

▶ Lemma 7. Let $P = (p_1,\ldots,p_n)$ be a $c$-packed curve in $X$. For any $\rho > 0$ and $0 < \varepsilon < 1$, denote $\rho^* = (1 + \frac{\kappa}{\varepsilon})\rho$. For any $k \geq 1$, denote by $P^\mu$ its $\mu$-simplified curve for $\mu \geq \frac{\kappa^\mu}{\varepsilon^2}$. For any curve $Q = (q_1,\ldots,q_m) \subset X$ the matrix $M^{P^\mu\times Q}_\rho$ contains at most $8 \cdot \frac{c^\mu}{\varepsilon^2}$ zeroes per row.

Proof. The proof is by contradiction. Suppose that the $j$'th row of $M^{P^\mu\times Q}_\rho$ contains strictly more than $8 \cdot \frac{c^\mu}{\varepsilon^2}$ zeroes. Let $P_0 \subset P^\mu$ be the vertices corresponding to these zeroes. Consider the ball $B_1$ centered at $q_j$ with radius $|B_1| = (1 + \alpha)\rho^*$ and the ball $B_2$ with radius $2|B_1|$ (Figure 2). Each $p_i \in P_0$ must be contained in $B_1$ (since $d(p_i,q_j) \leq (1 + \alpha)\rho^*$). For each $p_i \in P_0$ denote by $S_i$ the segment of $P$ starting at $p_i$ of length $\mu$. Observe that since $\varepsilon < 1$:

$$|S_i| \subset B_2.$$ 

This lower bounds the length of $P \cap B_2$:

$$\ell(P \cap B_2) \geq \sum_{p \in P_0} \ell(S_i) = \sum_{p \in P_0} \mu \geq 2(1+\alpha)(1+\frac{\varepsilon}{2}) \cdot \frac{c \cdot k}{\varepsilon} \cdot \mu \geq 2(1+\alpha) \cdot c \cdot \rho^* \geq c \cdot |B_2|.$$

This contradicts the assumption that $P$ was $c$-packed. ▶
Figure 2 (a) For some value of $\mu$, the $\mu$-simplified curve is $(p_1, p_5, p_6, \ldots, p_{27}, p_n)$. We show the matrix $M_\rho \mu \times Q$. (b) For the point $q_3$, we claim that there are more than $Z = 8 \frac{k}{\varepsilon}$ zeroes in its corresponding row. Thus, the ball $B_1$ with radius $(1 + \alpha)\rho^*$ contains more than $Z$ points. (c) For each of these points, there is a unique segment along $P$ contained in the ball $B_2$.

**Corollary 8.** Let $P = (p_1, \ldots, p_n)$ be a c-packed curve in $X$. For any $\rho > 0$ and $0 < \varepsilon < 1$, denote $\rho^* = (1 + \frac{1}{2}\varepsilon)\rho$. For any $k \geq 1$, denote by $P_\mu \rho$ its $\mu$-simplified curve for $\mu \geq \frac{c}{5\varepsilon}$. For any curve $Q = (q_1, \ldots, q_m) \subset X$ the matrix $M_\rho \mu \times Q$ contains at most $8 \cdot \frac{k}{\varepsilon} \cdot m$ zeroes.

### 3.2 Perceived distance implies distance

We show that we can use $D_X^\rho(P_\mu, Q)$ for some $\mu$-simplification to answer A-decision($Q, \rho, \varepsilon$):

**Lemma 9.** For any $\rho > 0$ and $0 < \varepsilon < 1$, choose $\rho^* = (1 + \frac{1}{2}\varepsilon)\rho$ and $\mu \leq \frac{1}{6}\varepsilon\rho$. Let $X$ be a metric space and $D_X^\rho \times Q$ be a $(1 + \varepsilon)$-approximate distance oracle. For a c-packed curve $P = (p_1, \ldots, p_n)$ in $X$ and any curve $Q = (q_1, \ldots, q_m)$ in $X$:

= If $D_X^\rho(P, Q) \leq \rho^*$ then $D_X(P, Q) \leq (1 + \varepsilon)\rho$.

= If $D_X^\rho(P, Q) \geq \rho^*$ then $D_X(P, Q) \geq \rho$.

**Proof.** Per definition of $D_X^\rho \times Q$: $\forall (p, q) \in P \times Q$, $d^\rho(p, q) \in [(1 - \frac{1}{6}\varepsilon)d(p, q), (1 + \frac{1}{6}\varepsilon)d(p, q)]$.

It follows from $0 < \varepsilon < 1$ that:

$$\forall (p, q) \in P \times Q: \quad d(p, q) \leq \left(1 + \frac{1}{6}\varepsilon\right)d^\rho(p, q) \quad \wedge \quad d^\rho(p, q) \leq \left(1 + \frac{1}{6}\varepsilon\right)d(p, q).$$

**Suppose that** $D_X^\rho(P_\mu, Q) \leq \rho^*$. There exists a (monotone) discrete walk $F$ through $P_\mu \times Q$ such that for each $(i, j) \in F$: $d^\rho(P_\mu \rho[i], q_j) \leq \rho^* = (1 + \frac{1}{2}\varepsilon)\rho$. It follows that:

$$d(P_\mu \rho[i], q_j) \leq \left(1 + \frac{1}{6}\varepsilon\right)d^\rho(P_\mu \rho[i], q_j) \leq \left(1 + \frac{1}{6}\varepsilon\right)\left(1 + \frac{1}{2}\varepsilon\right)\rho \leq \left(1 + \frac{5}{6}\varepsilon\right)\rho.$$

We will prove that this implies $D_X(P, Q) \leq (1 + \varepsilon)\rho$. We use $F$ to construct a discrete walk $F'$ through $P \times Q$. For each consecutive pair $(a, b), (c, d) \in F$ note that since $F$ is a discrete walk, $P_\mu[a]$ and $P_\mu[c]$ are either the same vertex or incident vertices on $P_\mu$. Denote by $P_{ac}$ the vertices of $P$ in between $P_\mu[a]$ and $P_\mu[c]$. It follows that:

$$\forall p' \in P_{ac}: \quad d(p', q_b) \leq d(P_\mu \rho[a], q_b) + \mu \leq (1 + \frac{5}{6}\varepsilon)\rho + \frac{1}{6}\varepsilon\rho = (1 + \varepsilon)\rho.$$

Now consider the following sequence of pairs of points:

$L_{ac} = (P_\mu \rho[a], q_b) \cup \{(p', q_b) | p' \in P_{ac} \} \cup (P_\mu \rho[c], q_d)$. We add the lattice points corresponding to $L_{ac}$ to $F'$. It follows that we create a discrete walk $F'$ in the lattice $|P| \times |Q|$ where for each $(i, j) \in F'$: $d(pi, q_j) \leq (1 + \varepsilon)\rho$. Thus, $D_X(P, Q) \leq (1 + \varepsilon)\rho$. 

\[ q_3 \quad \begin{array}{cccc} 0 & 0 & 0 & 1 & 0 & 0 \\ q_2 & 1 & 1 & 1 & 0 & 0 & 1 \\ q_1 & 0 & 0 & 0 & 1 & 1 & 1 \end{array} \]

\[ P_\mu :: p_1 \ p_5 \ p_6 \ldots p_{27} \ p_n \]
Suppose otherwise that $D_F(P, Q) \leq \rho$. We will prove that $D_F^\circ(P^\mu, Q) \leq \rho^*$. Indeed, consider a discrete walk $F'$ in the lattice $|P| \times |Q|$ where for each $(i, j) \in F'$: $d(p_i, q_j) \leq \rho$. We construct a discrete walk $F$ in $|P^\mu| \times |Q|$. Consider each $(i, j) \in F'$, If $p_i = P^\mu[a]$ for some integer $a$, we add $(a, j)$ to $F$. Otherwise, denote by $P^\mu[a]$ the last vertex on $P^\mu$ that precedes $p_i$: we add $(a, j)$ to $F$. Note that per definition of $\mu$-simplification, $d(P^\mu[a], q_j) \leq d(p_i, q_j) + \mu \leq (1 + \frac{1}{\rho^*})\rho$. It follows from the definition of our approximate distance oracle that $d^\circ(P^\mu[a], q_j) \leq (1 + \frac{1}{6}\epsilon)(1 + \frac{1}{6}\epsilon)\rho < (1 + \frac{1}{\rho^*})\rho = \rho^*$. Thus, we may conclude that $D_F^\circ(P^\mu, Q) \leq \rho^*$. ▶

3.3 Answering A-decision($Q, \varepsilon, \rho$)

We showed in Sections 3.1 and 3.2 that for any $c$-packed curve $P$, $\rho > 0$ and $0 < \varepsilon < 1$ we can choose suitable values $\frac{\varepsilon}{6} \leq \mu \leq \frac{\varepsilon}{6}$. This upper bounds the number of zeroes in $M^\rho_{|P^\mu| \times |Q|}$. Moreover, for $\rho^* = (1 + \frac{1}{\rho^*})\rho$ we know that computing $D_F^\circ(P^\mu, Q)$ implies an answer to A-decision($Q, \varepsilon, \rho$). We now construct a data structure over $P$ such that for any value $\mu$, we can efficiently obtain $P^\mu$ and answer A-decision($Q, \varepsilon, \rho$). Note that we assume that the preprocessing input specifies the distance between consecutive points on $P$. This assumption may be lifted by incorporating the distance oracle query time into the preprocessing time.

Lemma 10. Let $P = (p_1, \ldots, p_n)$ be a curve in $\mathcal{X}$. We can store $P$ using $O(n)$ space and preprocessing time, such that for any value $\mu \geq 0$ and any integer $N$ we can report the first $N$ vertices on the $\mu$-simplification $P^\mu$ in $O(N\log n)$ time.

Proof. For each $1 < i \leq n$ we create a half-open interval $[\ell(P[1, i-1]), \ell(P[1, i])]$ in $\mathbb{R}^1$. Each point in $P$ receives a pointer to its unique interval. This results in an ordered set of $O(n)$ disjoint intervals on which we build a balanced binary tree in $O(n)$ time.

For any given value $\mu$, we can now obtain the first $N$ vertices of $P^\mu$ as follows: First, we add $p_1$. Then, we inductively add subsequent vertices. Suppose that we just added $p_i$ to our output. We choose the value $a = \ell(P[1, i]) + \mu$. We binary search in $O(\log n)$ time for the point $p_j$ where the interval $[\ell(P[1, j-1]), \ell(P[1, j])]$ contains $a$. Per definition of the intervals: the length $\ell(P[i, j]) \geq \mu$. Moreover, for all $x \in (i, j)$ the length $\ell(P[i, x]) < x$. Thus, $p_j$ is the successor of $p_i$ and we recurse if necessary. ▶

Theorem 1. Let $\mathcal{X}$ be a metric space and $D_\mathcal{X}^\alpha$ be a $(1 + \alpha)$-approximate distance oracle with $O(T_n)$ query time. Let $P = (p_1, \ldots, p_n)$ be any $c$-packed curve in $\mathcal{X}$. We can store $P$ using $O(n)$ space and preprocessing time, such that for any curve $Q = (q_1, \ldots, q_m)$ in $\mathcal{X}$ and any $\rho > 0$ and $0 < \varepsilon < 1$, we can answer A-decision($Q, \varepsilon, \rho$) for the Fréchet distance in:

$$O\left(\frac{c \cdot m}{\varepsilon} \cdot (T_{\varepsilon/6} + \log n)\right)$$

time.

Proof. We store $P$ in the data structure of Lemma 10 using $O(n)$ space and preprocessing time. Given a query A-decision($Q, \varepsilon, \rho$) we choose $\alpha = \frac{\varepsilon}{3C}, \rho^* = (1 + \frac{1}{\rho^*})\rho$ and $\mu = \frac{1}{3C}$. We test if $D_F^\circ(\mu^{\mu}, Q) \leq \rho^*$. By Lemma 10 if $D_F^\circ(\mu^{\mu}, Q) \leq \rho^*$ then $D_F(P, Q) \leq (1 + \varepsilon)\rho$ and otherwise $D_F(P, Q) > \rho$. We consider the matrix $M^{\rho^*}_{|P| \times |Q|}$.

By Observation 6 $D_F^\circ(\rho^{\mu}, Q) \leq \rho^*$ if and only if there exists a discrete walk $F$ from $(1, 1)$ to $[|P^\mu|, |Q|]$ where for each $(i, j) \in F$: $M^{\rho^*}_{|P^\mu| \times |Q|}[i, j] = 0$. We will traverse this matrix in a depth-first manner as follows: starting from the cell $(1, 1)$, we test if $M^{\rho^*}_{|P^\mu| \times |Q|}[1, 1] = 0$. If so, we push $(1, 1)$ onto a stack. Each time we pop a tuple $(i, j)$ from the stack, we inspect their $O(1)$ neighbors $\{(i+1, j), (i, j+1), (i+1, j+1)\}$ (we inspect twice as many neighbors for the weak Fréchet distance). If $M^{\rho^*}_{|P^\mu| \times |Q|}[i', j'] = 0$, we push $(i', j')$ onto our stack. It takes
Approximate Discrete Fréchet distance: simplified, extended and structured

$O(\log n)$ time to obtain the $i + 1$th vertex of $P^\mu$, and $O(T_{\varepsilon/6})$ to determine the value of e.g., $M_{\rho}^{P^\mu \times Q}[i + 1, j]$. Thus each time we pop the stack, we spend $O(T_{\varepsilon/6} + \log n)$ time.

By Corollary 8 (noting $\varepsilon < 1$ and setting $k = 6$), we push at most $O(\frac{m}{\varepsilon})$ tuples onto our stack. Therefore, we spend $O(\frac{m}{\varepsilon}(T_{\varepsilon/6} + \log n))$ total time. By Observation 6, $D_F^\varepsilon(P^\mu, Q) \leq \rho^*$ if and only if we push $([P^\mu], [Q])$ onto our stack. We test this in $O(1)$ additional time per operation. Thus, the theorem follows.

Note that Lemma 10 can for any $\mu$ also report the $\mu$-simplification of $P[i, j]$ for any $(i, j)$ at query time. Therefore, Theorem 3 also immediately works for the subcurves query variants.

3.4 Answering A-value($Q, \varepsilon$)

Finally, we show how to answer the A-value($Q, \varepsilon, \rho$) query. Recall that previous methods to approximate the value $D_F(P, Q)$ rely upon inefficient data structures: when $X = \mathbb{R}^d$ a $d$-dimensional WSPD of $P \times Q$, and when $X$ is a graph the sorted set of edges $E$. We circumvent this, by leveraging the variable $k \geq 1$ introduced in the definition of $\mu \geq \frac{\varepsilon}{k}$:

**Theorem 3.** Let $X$ be a metric space and $D_X^0$ be a $(1 + \alpha)$-approximate distance with $O(T_n)$ query time. Let $P = (p_1, \ldots, p_n)$ be any $c$-packed curve in $X$. We can store $P$ using $O(n)$ space and $O(n \log n)$ preprocessing time, such that for any curve $Q = (q_1, \ldots, q_m)$ in $X$ and any $0 < \varepsilon < 1$, we can answer $A$-Value($Q, \varepsilon$) for the Fréchet distance in:

$$O\left(\frac{c \cdot m}{\varepsilon} \cdot \log n \cdot \left(T_{\varepsilon/6} + \log \frac{c \cdot m}{\varepsilon} + \log n\right)\right)$$

time.

**Proof.** We preprocess $P$ using Lemma 10 in $O(n)$ space and time. We map $P$ onto $\mathbb{R}^1$ as follows: each vertex $p_i$ gets mapped to the value $\lambda_i = \ell(P[1, i])$. Denote by $\Lambda = \{\lambda_i\}$. As a second data structure, we compute a 1-dimensional WSPD on $\Lambda$ and itself in $O(n \log n)$ time using $O(n)$ space 20. We obtain a partition of $P \times P$ into $O(n)$ sets $(P_s, P'_s)$ where for each $s$, there exists a $c_s \in \mathbb{R}$ such that for all $p_j \in P_s$ and $p_j \in P'_s$: $\ell(P[i, j]) \in [c_s \cdot 2 \cdot c_s]$. We denote by $I_s = [c_s \cdot 2 \cdot c_s]$ the corresponding interval and obtain a sorted set of intervals $\mathcal{I}$.

Given a query $(Q, \varepsilon, \rho)$, we set $\alpha \leftarrow \varepsilon/6$ and obtain $D_X^\alpha$. We (implicitly) rescale each interval $I \in \mathcal{I}$ by a factor $\frac{\varepsilon}{2}$, creating for $I_s$ the interval $I_s^\varepsilon = [\frac{6 \cdot c_s}{\varepsilon}, \frac{12 \cdot c_s}{\varepsilon}]$. This creates a sorted set $\mathcal{I}^\varepsilon$ of pairwise disjoint intervals. Intuitively, these are the intervals over $\mathbb{R}^1$ where for $\rho \in I_s^\varepsilon$, the $\mu$-simplification $P^\mu$ for $\mu = \frac{\rho}{\varepsilon}$ may change.

We binary search over $\mathcal{I}^\varepsilon$. For each boundary point $\lambda$ of an interval $I_s^\varepsilon$ we query $A$-decision($Q, \varepsilon, \lambda$): discarding half of the remaining intervals in $\mathcal{I}^\varepsilon$. It follows that in $O\left(\frac{c \cdot m}{\varepsilon} \cdot \log n \cdot (T_{\varepsilon/6} + \log n)\right)$ time, we obtain one of two things:

(a) an interval $I_s^\varepsilon$ where $\exists \rho^* \in I_s^\varepsilon$ that is a $(1 + \varepsilon)$-approximation of $D_F(P, Q)$, or
(b) a maximal interval $I^*$ disjoint of the intervals in $\mathcal{I}^\varepsilon$ where $\exists \rho^* \in I^*$ that is a $(1 + \varepsilon)$-approximation of $D_F(P, Q)$.

Denote by $\lambda$ the left boundary of $I_s^\varepsilon$ or $I^*$: it lower bounds $D_F(P, Q)$. Note that if $I^*$ precedes all of $\mathcal{I}^\varepsilon$, $\lambda = 0$. We now compute a $(1 + \varepsilon)$-approximation of $D_F(P, Q)$ as follows:
FindApproximation($\lambda$):
1. Compute $C = \frac{d^*(p_a,q_b)}{(1 + \frac{1}{2}\epsilon)}$.
2. Initialize $\rho^* \leftarrow (1 + \frac{1}{2}\epsilon) \cdot \max\{C, \lambda\}$ and set a constant $\mu \leftarrow \frac{\epsilon}{6} \cdot \lambda$.
3. Push the lattice point $(1,1)$ onto a stack.
4. Whilst the stack is not empty do:
   - Pop a point $(i,j)$ and consider the $O(1)$ neighbors $(p_a,q_b)$ of $(p_i,q_j)$ in $M^{P^\mu \times Q}_{\rho^*}$;
   - If $d^*(p_a,q_b) \leq \rho^*$, push $(a,b)$ onto the stack.
   - Else, store $d^*(p_a,q_b)$ in a min-heap.
   - If we push $(p_n,q_m)$ onto the stack do:
     - Output $\nu = \frac{\rho^*}{(1 + \frac{1}{2}\epsilon)}$.
5. If the stack is empty, we extract the minimal $d^*(p_a,p_b)$ from the min-heap.
   - Update $\rho^* \leftarrow (1 + \frac{1}{2}\epsilon) \cdot d^*(p_a,q_b)$, push $(a,b)$ onto the stack and go to line 4.

Correctness. Suppose that our algorithm pushes $(p_n,q_m)$ onto the stack and let at this time of the algorithm, $\rho^* = (1 + \frac{1}{2}\epsilon)\nu$. Per definition of the algorithm, $\nu \geq \lambda$ is the minimal value for which the matrix $M^{P^\mu \times Q}_{\rho^*}$ contains a walk $F$ from $(1,1)$ to $(n,m)$ where for each $(i,j) \in F$: $M^{P^\mu \times Q}_{\rho^*}[i,j] = 0$. Indeed, each time we increment $\rho^*$ by the minimal value required to extend any walk in $M^{P^\mu \times Q}_{\rho^*}$. Moreover, we fixed $\mu \leq \frac{\epsilon}{6}\lambda$ and thus $\mu \leq \frac{\epsilon}{6}\nu$. Thus we may apply Lemma 9 to defer that $\nu$ is the minimal value for which $D_F(P,Q) \leq (1 + \epsilon)\nu$.

Running time. We established that the binary search over $I^\epsilon$ took $O(\frac{\epsilon n m}{\epsilon} \log n (T_{c/\epsilon} + \log n))$ time. We upper bound the running time of our final routine. For each pair $(p_i,q_j)$ that we push onto the stack we spend at most $O(T_{c/\epsilon} + \log \frac{\epsilon n m}{\epsilon} + \log n)$ time as we:
- Obtain the $O(1)$ neighbors of $(p_i,q_j)$ through our data structure in $O(\log n)$ time,
- Perform $O(1)$ distance oracle queries in $O(T_{c/\epsilon})$ time, and
- Possibly insert $O(1)$ neighbors into a min-heap. The min-heap has size at most $K$: the number of elements we push onto the stack. Thus, this takes $O(\log K)$ insertion time.

What remains is to upper bound the number of items we push onto the stack. Note that we only push an element onto the stack, if for the current value $\rho^*$ the matrix $M^{P^\mu \times Q}_{\rho^*}$ contains a zero in the corresponding cell. We now refer to our earlier case distinction.

Case (a): Since $\epsilon < 1$ we know that $\rho^* \in [\lambda, 4 \cdot \lambda]$. We set $\mu \leq \frac{\epsilon}{6}\lambda$. So $\mu \geq \frac{1}{\epsilon}\epsilon\rho^*$ for $k = 24$. Thus, we may immediately apply Corollary 9 to conclude that we push at most $O(\frac{\epsilon n m}{\epsilon})$ elements onto the stack.

Case (b): Denote by $\gamma = \frac{\epsilon}{6}\nu$. Per definition of our re-scaled intervals, the half-open interval $(\mu, \gamma)$ does not intersect with any interval in the non-scaled set $\mathcal{I}$. It follows that $P^\mu = P^\gamma$ and that for two consecutive vertices $p_i, p_l \in P^\mu$: $\ell(P[i,l]) > \gamma$. From here, we essentially redo Lemma 7 for this highly specialized setting. The proof is by contradiction, where we assume that for $\rho^* = (1 + \frac{1}{2}\epsilon)\nu$ there are more than $8 \cdot 6 \cdot \frac{\epsilon}{\epsilon}$ in the $j$'th row of $M^{P^\mu \times Q}_{\rho^*}$. Denote by $P_0 \subset P^\mu$ the vertices corresponding to these zeroes. We construct a ball $B_1$ centered at $q_j$ with radius $2\rho^*$ and a ball $B_2$ with radius $2 |B_1|$. We construct a subcurve $S_i$ of $P$ starting at $p_i \in P_0$ of length $\gamma$. The critical observation is, that our above analysis implies that all the sub-curves $S_i$ do not coincide (since each of them start with a vertex in $P^\mu$). Since $\epsilon < 1$, each segment $S_i$ is contained in $B_2$. However, this implies that $B_2$ is not $c$-packed since: $\ell(P \cap B_2) \geq \sum \ell(S_i) = \sum \gamma > 8 \cdot 6 \frac{\epsilon}{\epsilon}\gamma \geq 4 \cdot c \cdot \rho^* \geq 2 \cdot c \cdot |B_2|$. Thus, we always push at most $O(\frac{\epsilon n m}{\epsilon})$ elements onto our stack and this implies our running time.
References

1. Helmut Alt and Michael Godau. Computing the Fréchet distance between two polygonal curves. *International Journal of Computational Geometry & Applications*, 5(01n02):75–91, 1995.

2. Boris Aronov, Omrit Filtser, Michael Horton, Matthew J. Katz, and Khadijeh Sheikhan. Efficient nearest-neighbor query and clustering of planar curves. In Zachary Friggstad, Jörg-Rüdiger Sack, and Mohammad R Salavatipour, editors, *Algorithms and Data Structures*, pages 28–42, Cham, 2019. Springer International Publishing.

3. Alessandro Bombelli, Lluis Soler, Eric Trumbauer, and Kenneth D Mease. Strategic air traffic planning with Fréchet distance aggregation and rerouting. *Journal of Guidance, Control, and Dynamics*, 40(5):1117–1129, 2017.

4. Sotiris Brakatsoulas, Dieter Pfoser, Randall Salas, and Carola Wenk. On map-matching vehicle tracking data. In *Proceedings of the 31st international conference on Very large data bases*, pages 853–864, 2005.

5. Karl Bringmann. Why walking the dog takes time: Fréchet distance has no strongly subquadratic algorithms unless SETH fails. In 2014 *IEEE 55th Annual Symposium on Foundations of Computer Science*, pages 661–670. IEEE, 2014.

6. Karl Bringmann and Marvin Künnemann. Improved approximation for Fréchet distance on c-packed curves matching conditional lower bounds. 2015.

7. Karl Bringmann and Wolfgang Mulzer. Approximability of the discrete Fréchet distance. *Journal of Computational Geometry*, 7(2):46–76, 2016.

8. Kevin Buchin, Maike Buchin, David Duran, Britanny Terese Fasy, Roel Jacobs, Vera Sacristán, Rodrigo I Silveira, Frank Staals, and Carola Wenk. Clustering trajectories for map construction. In *Proceedings of the 25th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems*, pages 1–10, 2017.

9. Kevin Buchin, Maike Buchin, Wouter Meulemans, and Wolfgang Mulzer. Four soviets walk the dog: Improved bounds for computing the Fréchet distance. *Discrete & Computational Geometry*, 58(1):180–216, 2017.

10. Kevin Buchin, Tim Ophelders, and Bettina Speckmann. Seth says: Weak Fréchet distance is faster, but only if it is continuous and in one dimension. In *Proceedings of the Thirtieth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 2887–2901. SIAM, 2019.

11. Maike Buchin, Bernhard Kilgus, and Andrea Kölzsch. Group diagrams for representing trajectories. *International Journal of Geographical Information Science*, 34(12):2401–2433, 2020.

12. Maike Buchin, Ivor van der Hoog, Tim Ophelders, Lena Schlipf, Rodrigo I Silveira, and Frank Staals. Efficient Fréchet distance queries for segments. *European Symposium on Algorithms*, 2022.

13. Timothy M Chan and Zahed Rahmati. An improved approximation algorithm for the discrete Fréchet distance. *Information Processing Letters*, 138:72–74, 2018.

14. Daniel Chen, Anne Driemel, Leonidas J. Guibas, Andy Nguyen, and Carola Wenk. Approximate map matching with respect to the Fréchet distance. In Matthias Müller-Hannemann and Renato Fonseca F. Werneck, editors, *Proceedings of the Thirteenth Workshop on Algorithm Engineering and Experiments, ALENEX 2011, Holiday Inn San Francisco Golden Gateway, San Francisco, California, USA, January 22, 2011*, pages 75–83. SIAM, 2011. doi:10.1137/1.9781611972917.8

15. Mark De Berg, Atlas F Cook IV, and Joachim Gudmundsson. Fast Fréchet queries. *Computational Geometry*, 46(6):747–755, 2013.
16 Mark de Berg, Ali D Mehrabi, and Tim Ophelders. Data structures for Fréchet queries in trajectory data. In 29th Canadian Conference on Computational Geometry (CCCG’17), pages 214–219, 2017.
17 Thomas Devogele. A new merging process for data integration based on the discrete Fréchet distance. In Advances in spatial data handling, pages 167–181. Springer, 2002.
18 Anne Driemel and Sariel Har-Peled. Jaywalking your dog: computing the Fréchet distance with shortcuts. SIAM Journal on Computing, 42(5):1830–1866, 2013.
19 Anne Driemel, Sariel Har-Peled, and Carola Wenk. Approximating the Fréchet distance for realistic curves in near linear time. Discret. Comput. Geom., 48(1):94–127, 2012. doi:10.1007/s00454-012-9402-z
20 Anne Driemel and Ioannis Psarros. (2+ε)-ANN for time series under the Fréchet distance. Workshop on Algorithms and Data structures (WADS), 2021.
21 Anne Driemel, Ioannis Psarros, and Melanie Schmidt. Sublinear data structures for short Fréchet queries. CoRR, abs/1907.04420, 2019. URL: http://arxiv.org/abs/1907.04420, arXiv:1907.04420
22 Anne Driemel, Ivor van der Hoog, and Eva Rotenberg. On the Fréchet distance in graphs. In International Symposium on Computational Geometry (SoCG 2023). Schloss Dagstuhl-Leibniz-Zentrum für Informatik, 2023.
23 Thomas Eiter and Heikki Mannila. Computing discrete Fréchet distance. Technical Report CD-TR 94/64, Christian Doppler Laboratory for Expert Systems, TU Vienna, Austria, 1994.
24 Jeff Erickson, Ivor Van Der Hoog, and Tillmann Miltzow. Smoothing the gap between np and er. In 2020 IEEE 61st Annual Symposium on Foundations of Computer Science (FOCS), pages 1022–1033. IEEE, 2020.
25 Arnold Filtser and Omrit Filtser. Static and streaming data structures for Fréchet distance queries. In Dániel Marx, editor, Proceedings of the 2021 ACM-SIAM Symposium on Discrete Algorithms, SODA 2021, Virtual Conference, January 10 - 13, 2021, pages 1150–1170. SIAM, 2021. doi:10.1137/1.9781611976465.71
26 Arnold Filtser, Omrit Filtser, and Matthew J. Katz. Approximate nearest neighbor for curves – simple, efficient, and deterministic. In 47th International Colloquium on Automata, Languages, and Programming (ICALP 2020), volume 168 of Leibniz International Proceedings in Informatics (LIPIcs), pages 48:1–48:19, 2020. URL: https://drops.dagstuhl.de/opus/volltexte/2020/12455, doi:10.4230/LIPIcs.ICALP.2020.48
27 Omrit Filtser. Universal approximate simplification under the discrete Fréchet distance. Inf. Process. Lett., 132:22–27, 2018. doi:10.1016/j.ipl.2017.10.002
28 Joachim Gudmundsson, Martin P. Seybold, and Sampson Wong. Map matching queries on realistic input graphs under the Fréchet distance. Symposium on Discrete Algorithms (SODA), 2023.
29 Joachim Gudmundsson, André van Renssen, Zeinab Saeidi, and Sampson Wong. Fréchet distance queries in trajectory data. In The Third Iranian Conference on Computational Geometry (ICCG 2020), pages 29–32, 2020.
30 Sariel Har-Peled. Geometric approximation algorithms. Number 173. American Mathematical Soc., 2011.
31 Minghui Jiang, Ying Xu, and Binhai Zhu. Protein structure–structure alignment with discrete Fréchet distance. Journal of bioinformatics and computational biology, 6(01):51–64, 2008.
32 Maximilian Konzack, Thomas McKetterick, Tim Ophelders, Maike Buchin, Luca Giuggioli, Jed Long, Trisalyn Nelson, Michel A Westenberg, and Kevin Buchin. Visual analytics
of delays and interaction in movement data. *International Journal of Geographical Information Science*, 31(2):320–345, 2017.

33 Yaowei Long and Seth Pettie. Planar distance oracles with better time-space tradeoffs. In Dániel Marx, editor, *Proceedings of the 2021 ACM-SIAM Symposium on Discrete Algorithms, SODA 2021, Virtual Conference, January 10 - 13, 2021*, pages 2517–2537. SIAM, 2021. [doi:10.1137/1.9781611976465.149].

34 Ariane Mascret, Thomas Devogele, Iwan Le Berre, and Alain Hénaff. Coastline matching process based on the discrete Fréchet distance. In *Progress in Spatial Data Handling*, pages 383–400. Springer, 2006.

35 Nimrod Megiddo. Applying parallel computation algorithms in the design of serial algorithms. *J. ACM*, 30(4):852–865, 1983.

36 Roniel S. De Sousa, Azzedine Boukerche, and Antonio A. F. Loureiro. Vehicle trajectory similarity: Models, methods, and applications. *ACM Comput. Surv.*, 53(5), September 2020. [doi:10.1145/3406096].

37 E Sriraghavendra, K Karthik, and Chiranjib Bhattacharyya. Fréchet distance based approach for searching online handwritten documents. In *Ninth International Conference on Document Analysis and Recognition (ICDAR 2007)*, volume 1, pages 461–465. IEEE, 2007.

38 Han Su, Shuncheng Liu, Bolong Zheng, Xiaofang Zhou, and Kai Zheng. A survey of trajectory distance measures and performance evaluation. *The VLDB Journal*, 29(1):3–32, 2020.

39 Mikkel Thorup. Compact oracles for reachability and approximate distances in planar digraphs. *Journal of the ACM (JACM)*, 51(6):993–1024, 2004.

40 Thijs van der Horst, Marc van Kreveld, Tim Ophelders, and Bettina Speckmann. A subquadratic $n^{2/3}$-approximation for the continuous Fréchet distance. *Symposium on Discrete Algorithms (SODA)*, 2023.

41 A. Frank van der Stappen. *Motion planning amidst fat obstacles*. University Utrecht, 1994.

42 Rene Van Oostrum and Remco C Veltkamp. Parametric search made practical. In *Proceedings of the eighteenth annual symposium on Computational geometry*, pages 1–9, 2002.

43 Dong Xie, Feifei Li, and Jeff M Phillips. Distributed trajectory similarity search. *Proceedings of the VLDB Endowment*, 10(11):1478–1489, 2017.

44 Daming Xu. *Well-separated pair decompositions for doubling metric spaces*. PhD thesis, Carleton University, 2005.
This section is dedicated to showing that our approach also works to compute a $(1 + \epsilon)$-approximation of the Hausdorff distance $D_H(P, Q)$. Perhaps surprisingly, computing the Hausdorff distance is somewhat more complicated than computing the Fréchet distance. The intuition behind this, is as follows. Because $P$ is $c$-packed, we can upper bound for any decision variable $\rho$ the number of zeroes in the free-space matrix. For the Fréchet distance, we are interested in a connected walk through the matrix that only consists of zeroes, hence we can find such a walk using depth-first search. For the decision variant of the Hausdorff distance, we require instead that the free space matrix contains a zero in every row and in every column. Since this does not have the same structure as a connected path, we require more work and more time to verify this. To this end, we assume that in the preprocessing phase we may compute a Voronoi diagram in $O(f(n))$ time, and that the Voronoi diagram has $O(g(n))$ query time. In $\mathbb{R}^d$ under the $L_1, L_2, L_\infty$ metric, a Voronoi diagram on $n$ points can be computed in $f(n) = O(n \log n + n^{d/2})$ time: it subsequently has $g(n) = O(\log n)$ query time. In a graph under the shortest path metric a Voronoi diagram on $n$ points can be computed in $O(|G| \log |G|)$ time and it has $O(1)$ query time [22].

Before we go into the details, we make one brief remark: for ease of exposition, throughout this section, we assume that our distance oracle $D^\star_X$ is an exact distance oracle. The reasoning behind this is that we want to simplify our analysis to focus on illustrating the difference between computing the Hausdorff and Fréchet distance. Approximate distance oracles may be assumed by introducing $\rho^{\star} = (1 + \frac{\epsilon}{2})\rho$ and $\mu = \frac{\epsilon\rho}{4}$ in the exact way as in Section 3. Again our approach immediately also works for subcurve query variants.

**Hausdorff matrix.** For values $\rho^{\star} \geq 0$ and some $0 < \epsilon < 1$ we choose $\mu = \epsilon\rho^{\star}$. We define $H^{\rho^{\star} \times Q}_\rho$ as a $(0, 1)$-matrix of dimensions $|P^{\mu}|$ by $|Q|$ where rows correspond to the ordered vertices in $Q$ and columns to the ordered vertices in $P^{\mu}$. Let $p_i \in P$ and $p_i$ be its predecessor in $P^{\mu}$. For any $q_j \in Q$, we set the cell corresponding to $(p_i, q_j)$ to zero if there exists a $p' \in P[i, s]$ such that $d(p', q_j) \leq \rho^{\star}$. Otherwise, we set the cell to one. This defines $H^{\rho^{\star} \times Q}_\rho$.

**Lemma 11.** Let $X$ be any metric space and $P = (p_1, \ldots, p_m)$ and $Q = (q_1, \ldots, q_m)$ be curves in $X$. For any $\rho^{\star} \geq 0$, any $0 < \epsilon < 1$ and $\mu \leq \epsilon\rho^{\star}$: If there exists a zero in each row and each column of $H^{\rho^{\star} \times Q}_\rho$ then $D_H(P, Q) \leq (1 + \epsilon)\rho^{\star}$. Otherwise, $H^{\rho^{\star} \times Q}_\rho D_H(P, Q) > \rho^{\star}$.

**Proof.** Suppose that there exists a zero in each row and each column of $H^{\rho^{\star} \times Q}_\rho$. Consider a column in $H^{\rho^{\star} \times Q}_\rho$ corresponding to a vertex $p_s \in P^{\rho^{\star}}$ and let its $j$th row contain a zero. Then for $p_s$ and all $p'$ in between $p_s$ and its successor: $d(p', q_j) \leq \rho^{\star} + \mu \leq (1 + \epsilon)\rho^{\star}$. Thus $D_H(P \rightarrow Q) \leq (1 + \epsilon)\rho^{\star}$. Similarly, consider the $j$th row in $H^{\rho^{\star} \times Q}_\rho$ and let it contain a zero in its column corresponding to $p_j$. Then $d(p_i, q_j) \leq (1 + \epsilon)\rho^{\star}$ and thus $D_F(Q \rightarrow P) \leq \rho^{\star}$. ▶

From this point on, we observe that the proof of Lemma 11 immediately applies to the matrix $H^{\rho^{\star} \times Q}_\rho$ (indeed, for any zero in the $j$th row we still obtain a point in the ball $B_1$ where the point may be some $p'$ succeeding $p_i$. The segment $S_i$ subsequently must be entirely contained in $B_2$ and so the proof follows). Thus, we conclude:

**Corollary 12.** Let $P = (p_1, \ldots, p_n)$ be a $c$-packed curve in $X$. For any $\rho^{\star} > 0$ and any $k > 1$, denote by $P^{\mu}$ its $\mu$-simplified curve for $\mu \geq \frac{\epsilon\rho^{\star}}{k}$. For any curve $Q = (q_1, \ldots, q_m) \subset X$ the matrix $H^{\rho^{\star} \times Q}_\rho$ contains at most $8 \cdot \frac{k}{\epsilon} \cdot m$ zeroes.
A.1 Answering A-decision\((Q, \varepsilon, \rho^*)\).

\textbf{Theorem 13.} Let \(X\) be a metric space and \(P = (p_1, \ldots, p_n)\) be any \(c\)-packed curve in \(X\). Suppose that we can construct a Voronoi diagram on \(P\) in \(O(f(n))\) time using \(O(s(n))\) space and that it has \(O(g(n))\) query time. We can store \(P\) using \(O((n + s(n)) \log n)\) space and \(O((n + f(n)) \log n)\) preprocessing time, such that for any curve \(Q = (q_1, \ldots, q_m)\) in \(X\), any \(\rho^* \geq 0\) and \(0 < \varepsilon < 1\), we can answer A-Decision\((Q, \varepsilon, \rho)\) for the Hausdorff distance in:

\[
O\left(\frac{c \cdot m}{\varepsilon} \cdot \log n \cdot g(n)\right) \quad \text{time.}
\]

\textbf{Proof.} At preprocessing, we construct the data structure of Lemma 10. In addition, we prove the first \(O\) takes \(|q|\) items on the stack. Per item on the stack, we take \(\rho^*\) and \(8\) at most \(\rho^*\) are at most \(\varepsilon\). Indeed: if \(\rho^*\) decomposition, we create a Voronoi diagram on the corresponding (sub)curve. This takes \(O((n + f(n)) \log n)\) total time and \(O((n + s(n)) \log n)\) total space.

At query time, we receive \(Q\) and a value \(\rho^* \geq 0\). We set \(\mu = \varepsilon \rho^*\) and in \(O\left(\frac{c \cdot m}{\varepsilon} \log n\right)\) time we compute the first \(8\varepsilon m\) vertices of \(P^{\mu}\).

If we discover that \(P^{\mu}\) has more than \(8\varepsilon m\) elements, we report that \(D_H(P, Q) > \rho\).

Indeed: if \(D_H(P, Q) \leq \rho\) then each column in \(H^{P^{\mu} \times Q}_{\rho^*}\) must contain a zero. Since there are at most \(m\) rows, then by the pigeonhole principle \(D_H(P, Q) \leq \rho\) only if \(P^{\mu}\) contains at most \(8\varepsilon m\) vertices. We store \(P^{\mu}\) in a balanced binary tree \(T\) (ordered along \(P^{\mu}\)) in \(O(|P^{\mu}|) = O\left(\frac{c \cdot m}{\varepsilon} \log n\right)\) time. We now iterate over each of the vertices in \(Q\) and do the following subroutine:

\textbf{Subroutine}(\(q_j \in Q, T\)): We execute each subroutine one by one, passing \(T\) along. We maintain a stack of pairs of indices.

If after all subroutines \(T\) is not empty, we output \(D_H(P, Q) > \rho\).

Otherwise, we output \(D_H(P, Q) \leq (1 + \varepsilon)\rho\).

1. Push the pair \((1, n)\) onto a stack.
2. Whilst the stack is not empty, do:
   - Set \((i, j) \leftarrow \text{pop}\) and obtain \(P[i, j]\) as \(O(\log n)\) roots in our hierarchical decomposition.
   - For each of the roots, query their Voronoi diagram with \(q_j\) in \(O(\log n \cdot V_{\varepsilon/8})\) total time.
   - Let \(p' \in P[i, j]\) be the vertex realising the minimal distance to \(q_j\).
     - If \(d(p', q) > \rho\): continue the loop with the next stack item.
     - Otherwise, find its predecessor \(p_s\) in \(P^{\mu}\) in \(O(\log n)\) time.
     - Set the cell corresponding to \((p_s, q_j)\) in \(H^{P^{\mu} \times Q}_{\rho^*}\) to zero.
     - Remove \(p_s\) from \(T\) in \(O\left(\frac{c \cdot m}{\varepsilon} \right) = O(\log n)\) time.
     - Let \(p_l\) be the successor of \(p_s\) in \(P^{\mu}\).
     - Push \((i, s)\) and \((l, j)\) onto the stack.
3. If only \((1, n)\) was ever pushed onto the stack, terminate and output \(D_H(P, Q) > \rho\) (no further subroutines are required).

\textbf{Runtime.} Each time we push two items onto the stack, the subroutine has found in the \(j\)’th row of \(H^{P^{\mu} \times Q}_{\rho^*}\) a new cell that has a zero. Thus, the subroutine can push at most \(O\left(\frac{c}{\varepsilon}\right)\) items on the stack. Per item on the stack, we take \(O(\log n \cdot g(n))\) time. Thus, our algorithm takes \(O\left(\frac{c \cdot m}{\varepsilon} \cdot \log n \cdot g(n)\right)\) total time.
Correctness. Finally, we show that our algorithm always outputs a correct conclusion through a case distinction on when we output an answer.

Let us output our answer after line 3 of the subroutine. Then we have found a vertex \( q_j \in Q \) whose row in \( H_{\rho,\mu}^{P^\ast \times Q} \) contains no zeroes and by Lemma\(^\text{[1]}\) \( D_H(P, Q) > \rho \).

Let us output an answer because \( T \) contains some vertex \( p_s \in P^\ast \) after all the subroutines. Then we have found a vertex \( p_s \in P^\ast \) whose column in \( H_{\rho,\mu}^{P^\ast \times Q} \) contains no zeroes and by Lemma\(^\text{[1]}\) \( D_H(P, Q) > \rho \). Indeed: suppose for the sake of contradiction that the corresponding column contains a zero in row \( j \). Let \( p_l \) be the successor of \( p_s \) on \( P^\ast \). Because there is a zero in \( H_{\rho,\mu}^{P^\ast \times Q}[s,j] \) it must be that there exists a vertex \( p_l \in P[s,l-1] \) with \( d(p',q_j) \leq \rho \). During Subroutine\((q_j \in Q,T)\), we always have at least one interval \((a,b)\) on the stack where \( i \in [a,b] \). Each time such \((a,b)\) gets found, we find a zero in the \( j\)’th row and push a pair \((c,d)\) on the stack with \( i \in [c,d] \). Thus, we eventually must find \( p_l \). However, when we find \( p_l \) we remove \( p_s \) from \( T \) which is a contradiction.

Finally, if none of the above two conditions hold it must be that every row and every column in \( H_{\rho,\mu}^{P^\ast \times Q} \) contains a zero and by Lemma\(^\text{[1]}\) \( D_H(P, Q) \leq (1+\varepsilon)\rho \).

A.2 Answering A-value\((Q,\varepsilon)\)

\(\triangledown\) Theorem 14. Let \( \mathcal{X} \) be a metric space and \( P = (p_1,\ldots,p_n) \) be any \( c \)-packed curve in \( \mathcal{X} \). Suppose that we can construct a Voronoi diagram on \( P \) in \( O(f(n)) \) time using \( O(s(n)) \) space and that it has \( O(q(n)) \) query time. We can store \( P \) using \( O((n+s(n)) \log n) \) space and \( O((n+f(n)) \log n) \) preprocessing time, such that for any curve \( Q = (q_1,\ldots,q_m) \in \mathcal{X} \), and \( 0 < \varepsilon < 1 \), we can answer the subcurve query \( A\text{-Value}(Q,\varepsilon) \) for the Hausdorff distance in:

\[
O\left(\frac{c \cdot m}{\varepsilon} \cdot g(n) \cdot \log^2 n \right)
\]
time.

Proof. This proof is simpler than in Section\(^\text{3}\) since we have access to a Voronoi diagram. During preprocessing, we construct the data structures used in Theorem\(^\text{3}\). Secondly, we note that constructing a WSPD on \( P \) with itself is dominated by Voronoi diagram construction time. Thus, we compute a WSPD on \( P \) with itself in \( O(f(n)) \) time. This WSPD is stored as a set of sorted intervals \( \mathcal{I} = \{I_s = [c_s,2 \cdot c_s]\} \). We create the set \( \mathcal{H} = \{H_s = [\frac{1}{2} c_s,4 \cdot c_s]\} \).

Note that \( \mathcal{H} \) is a set of intervals which may overlap. We keep \( \mathcal{H} \) sorted by \( c_s \).

Given a query \((Q,\varepsilon)\) we first do the following: we compute the Hausdorff distance from \( Q \) to \( P \) in \( O(m \cdot g(n)) \) time (by querying for every point in \( Q \), the Voronoi diagram in \( P \)). Let this Hausdorff distance be \( \lambda \), we create an interval \( H^* = [\lambda,2\lambda] \).

We now claim, that \( \rho = D_H(P, Q) \) is contained in either an interval in \( \mathcal{H} \) or in \( H^* \). Indeed, consider for each \( q \in Q \) a disk with radius \( \rho \) centered at \( q_j \). Each of these disks must contain at least one point of \( P \). Moreover, there must exist at least one \( q_j \in Q \) which has a point \( p_i \) on its border (else, the Hausdorff distance is smaller than \( \rho \)). Consider the disk \( D_j \) centered at \( q_j \) with radius \( \frac{\rho}{2} \). There exist two cases:

(a) \( D_j \) contains a point \( p_i \in P \). Then \( \rho \in [\frac{\rho}{2},d(p_l,p_i),4 \cdot d(p_l,p_i)] \) and so \( \rho \) is in \( \mathcal{H} \).

(b) \( D_j \) contains no points in \( P \). Then \( D_H(Q \rightarrow P) \) is at least \( \frac{\rho}{2} \) (and at most \( \rho \)) so \( \rho \in H^* \).

Having observed this, we simply do a binary search over \( \mathcal{H} \) (and check \( H^* \) separately). For each interval \( H_s = [\frac{1}{2} c_s,4 \cdot c_s] \), we query: \( A\text{-Decision}(Q,\varepsilon,4 \cdot c_s) \). We choose \( \rho^* = 4 \cdot c_s \) and \( \mu = \varepsilon \frac{1}{2} c_s \). By Corollary\(^\text{[2]}\), the matrix \( H_{\rho,\mu}^{P^\ast \times Q} \) contains \( O(\frac{\mu^2}{\varepsilon}) \) zeroes, which we identify with the algorithm from Theorem\(^\text{[3]}\) in \( O(\frac{c^2 m}{\varepsilon} \cdot g(n) \cdot \log^2 n) \) time. If \( s \) is the smallest integer \( s \) for which the matrix \( H_{\rho,\mu}^{P^\ast \times Q} \) contains a zero in every row and every column, then we obtain a \((1+\varepsilon)\)-approximation of the Hausdorff distance. We obtain \( s \) by inspecting all zeroes in the matrix \( H_{\rho,\mu}^{P^\ast \times Q} \), and computing the minimal required pointwise distance. \(\triangledown\)
The technique in Section 3.4 avoids the use of parametric search when minimising the Fréchet distance. We apply this technique to map matching under the discrete Fréchet distance and the discrete Hausdorff distance.

In the map matching problem, the input is a Euclidean graph (a graph $P$ with $n$ vertices embedded in $\mathbb{R}^2$). The goal is to preprocess $P$, so that any query curve $Q$ in $\mathbb{R}^2$ can be ‘mapped’ onto $P$. That is, we want to find a path $\pi$ in the graph $P$, such that the distance $D_* (\pi, Q)$ (our similarity measure, derived from the underlying $L_2$ metric in $\mathbb{R}^2$) is minimized.

Under the continuous Fréchet distance, it was previously was shown that one can pre-process a $c$-packed graph in quadratic time and nearly-linear space for efficient approximate map matching queries [28]. The query time is $O(m \log m \cdot (\log^4 n + c^4 \varepsilon^{-8} \log^2 n))$, where $n$ and $m$ are the number of edges in the graph and query curve, respectively.

We show that, using our techniques, under the discrete Fréchet distance and discrete Hausdorff distance, the query time improves to $O(m \log n \cdot (\log^4 n + c^4 \varepsilon^{-4} \log^2 n))$, where $n$ and $m$ are the number of edges in the graph and query curve, respectively. In particular, we reduce the polynomial dependence on $c$, $\log n$ and $\log m$ in the query time. We obtain this, by replacing the parametric search in [28] by our WSPD-technique. Since $P$ is a graph and not a single curve, we do something slightly different as we compute a two-dimensional WSPD on $P$ with itself (in the exact same manner as in Appendix A).

**Theorem 15.** Let $X = \mathbb{R}^2$ and $P$ be a $c$-packed graph in $X$. We can store $P$ using $O(n \log^2 n + c \varepsilon^{-4} n \log n \cdot \log \varepsilon^{-1})$ space and $O(c^2 \varepsilon^{-4} n^2 \log^2 n \cdot \log \varepsilon^{-1})$ preprocessing time, such that for any curve $Q$ in $X$, we can return, in $O(m \log n + \log \varepsilon^{-1} \cdot (\log^2 n + c^4 \varepsilon^{-4} \log^2 n))$ time a $(1 + \varepsilon)$-approximation of $\min_{\pi \in P} D_F(\pi, Q)$ (or $\min_{\pi \in P} D_H(\pi, Q)$).

**Proof (Sketch).** We explain how to modify the proofs by Gudmundsson et al. [28] to use our techniques instead. We build the data structure in the same way as in Gudmundsson et al. [28]. We modify Lemma 13 of [28] so that, instead of computing the the continuous Fréchet distance in $O(n \log n)$ time using a free space diagram, we compute the discrete Fréchet distance in $O(n)$ time. Specifically, in Lemma 13, the authors show how to compute a map matching between $P$ and $\overline{ab}$ between any two vertices $a, b \in P$. We note that the dynamic program to achieve this has linear running time instead of the traditional quadratic running time, since one of the two curves has only two vertices.

Therefore, Theorem 3 of [28] implies that one can construct a data structure using $O(n \log^2 n + c \varepsilon^{-4} n \log n \cdot \log \varepsilon^{-1})$ space and $O(c^2 \varepsilon^{-4} n^2 \log^2 n \cdot \log \varepsilon^{-1})$ preprocessing time, to answer the decision variant of the query problem in $O(m \log n + c^4 \varepsilon^{-4} \log^2 n)$ time. This holds for both the discrete Fréchet distance and the discrete Hausdorff distance.

What remains, is to apply the decision variant to efficiently obtain a $(1 + \varepsilon)$-approximation. Previously in [28], parametric search was applied, which introduces a factor of $O(\log m)$ and squares the polynomial dependencies on $\log n$, $c$ and $\varepsilon^{-1}$. We avoid parametric search.

In preprocessing time, we precompute a Voronoi diagram on $P$ in $O(n \log n)$ time, and a 2-dimensional WSPD on $P$ and $P$ in $O(n \log n)$ time [30]. The WSPD partitions $P \times P$ into $O(n)$ sets $(P_s, P'_s)$ where for each $s$ there exists a distance $c_s$ such that for all pairs of points $(p_i, p_j)$ contained in $(P_s, P'_s)$, we have $c_s \leq d(p_i, p_j) \leq 2c_s$. We sort the values $c_s$ and define $H_s$ to be the interval $[\frac{1}{2} \cdot c_s, 4 \cdot c_s]$. This gives a sorted set $H_s$.

Given a query $(Q, \varepsilon)$ we first do the following: we compute the Hausdorff distance from $Q$ to $P$ in $O(m \cdot \log n)$ time (by querying for every point in $Q$, the Voronoi diagram in $P$ and taking the maximum). Let this Hausdorff distance be $\lambda$, we create an interval $H^* = [\lambda, 2\lambda]$. 

**B Map matching**
Let \( \rho \) be the map matching distance between \( Q \) and \( P \). We claim that \( \rho \) contained in either an interval in \( \mathcal{H} \) or in \( \mathcal{H}^* \). Indeed, consider for each \( q \in Q \) a disk with radius \( \rho \) centered at \( q_j \). Each of these disks must contain at least one point of \( P \) (else, we cannot map \( q_j \) to \( P \) with distance at most \( \rho \)). Moreover, there must exist at least one \( q_j \in Q \) which has a point \( p_i \) on its border (else, we may decrease \( \rho \) and still maintain a discrete map matching to \( P \)). Consider the disk \( D_j \) centered at \( q_j \) with radius \( \frac{1}{2} \rho \). There exist two cases:

(a) \( D_j \) contains a point \( p_l \in P \). Then \( \rho \in \left[ \frac{1}{2} \cdot d(p_l, p_i), 4 \cdot d(p_l, p_i) \right] \) and so \( \rho \) is in \( \mathcal{H} \).

(b) \( D_j \) contains no points in \( P \). Then \( D_{\mathcal{H}}(Q \rightarrow P) \) is at least \( \frac{1}{2} \rho \). Observe that \( D_{\mathcal{H}}(Q \rightarrow P) \) is upper bounded by \( \rho \), and so \( \rho \in \mathcal{H}^* \).

We now use decision variant to binary search over \( \mathcal{H} \) to find an interval containing \( \rho \) (we check \( \mathcal{H}^* \) separately). It follows that we obtain an interval \([\alpha, \beta]\) with \( \rho \in [\alpha, \beta] \) and \( \frac{\beta}{\alpha} \in O(1) \). We use the standard approach to refine the interval into a \((1 + \varepsilon)\)-approximation using \( O(\log \varepsilon^{-1}) \) decision queries. This completes the description of the query procedure.

Finally, we perform an analysis of the preprocessing time and space, and the query time. The preprocessing is dominated by the construction of Theorem 3 of [28], which requires \( O(n \log^2 n + c \varepsilon^{-4} \log(1/\varepsilon) n \log n) \) space and \( O(c^2 \varepsilon^{-4} \log(1/\varepsilon) n^2 \log^2 n) \) preprocessing time. The query procedure consists of a binary search, which requires \( O(\log n) \) applications of the query decider to identify the interval \( I_q \) or \( I_j \). We require a further \( O(\log \varepsilon^{-1}) \) applications of the query decider to refine the \( O(1) \)-approximation to a \((1 + \varepsilon)\)-approximation. Since the decider takes \( O(m(\log^2 n + c^2 \varepsilon^{-4} \log n)) \) time per application, the overall query procedure can be answered in \( O(m \cdot (\log n + \log \varepsilon^{-1})(\log^2 n + c^2 \varepsilon^{-4} \log n)) \) time. \( \square \)