Locality-sensitive hashing of curves

Locality-Sensitive Hashing of Curves*†

Anne Driemel[1] and Francesco Silvestri[2]

1 Department of Mathematics and Computer Science, Eindhoven University of Technology, Eindhoven, The Netherlands
adriemel@tue.nl

2 Department of Information Engineering, University of Padova, Padova, Italy; and
Theoretical Computer Science, IT University of Copenhagen, Copenhagen Denmark
silvestri@dei.unipd.it

Abstract

We study data structures for storing a set of polygonal curves in $\mathbb{R}^d$ such that, given a query curve, we can efficiently retrieve similar curves from the set, where similarity is measured using the discrete Fréchet distance or the dynamic time warping distance. To this end we devise the first locality-sensitive hashing schemes for these distance measures. A major challenge is posed by the fact that these distance measures internally optimize the alignment between the curves. We give solutions for different types of alignments including constrained and unconstrained versions. For unconstrained alignments, we improve over a result by Indyk from 2002 [17] for short curves. Let n be the number of input curves and let m be the maximum complexity of a curve in the input. In the particular case where $m \leq \frac{\alpha}{4d} \log n$, for some fixed $\alpha > 0$, our solutions imply an approximate near-neighbor data structure for the discrete Fréchet distance that uses space in $O(n^{1+\alpha} \log n)$ and achieves query time in $O(n^{\alpha} \log^2 n)$ and constant approximation factor. Furthermore, our solutions provide a trade-off between approximation quality and computational performance: for any parameter $k \in [m]$, we can give a data structure that uses space in $O(2^{2k} m^{k-1} n \log n + nm)$, answers queries in $O(2^{2k} m^k \log n)$ time and achieves approximation factor in $O(m/k)$.

1998 ACM Subject Classification F.2.2 Nonnumerical Algorithms and Problems

Keywords and phrases Locality-Sensitive Hashing, Fréchet distance, Dynamic Time Warping

Digital Object Identifier 10.4230/LIPIcs.SoCG.2017.37

1 Introduction

We study nearest-neighbor searching for polygonal curves under the discrete Fréchet distance or the dynamic time warping distance. This problem has various applications in machine learning, information retrieval and classification where the recorded instances are curves. Dynamic time warping has shown to be useful for classification of various types of data: surgical processes [11], whale singing [5], chromosomes [23], fingerprints [22], electrocardiogram (ECG) frames [15], and vessel trajectories [30]. Originally conceived for speech recognition, it is now being deployed as universal similarity measure for time series in the field of data

* A full version of the paper is available at https://arxiv.org/abs/1703.04040.

† Driemel has been supported by NWO Veni project "Clustering time series and trajectories (10019853)". Silvestri has been supported by the European Research Council project "Scalable Similarity Search" (no. 614331) and by MIUR of Italy under project AMANDA.

mining. The Fréchet distance is considered a useful similarity measure for trajectories of moving objects [6, 12, 20, 31].

Indyk and Motwani [19, 14] introduced the idea that hashing could enable faster nearest-neighbor searching in high-dimensional Euclidean spaces using a hashing scheme where near points are more likely to collide than far ones. They showed that such an approach can be used for the (c, r)-*near neighbor problem* which is defined as follows. Preprocess a set S of n points into a data structure that answers queries in the following way: if there exists a point $p \in S$ that lies within distance r from the query point q, then the data structure reports a point $p' \in S$ that lies within distance cr from q. In this paper, we study such locality-sensitive hashing schemes for the space of curves.

1.1 State of the art

In 2002, Indyk gave a deterministic and approximate near-neighbor data structure for the discrete Fréchet distance [17]. This data structure is to date the only result known for this task and represents the state of the art. The data structure achieves approximation factor $O(\log m + \log\log n)$, where m is the maximum length of a curve and n is the maximum number of elements in the data structure. Further, the data structures uses space in $O\Big(|X|^{\sqrt{m}}(m^{\sqrt{m}}n)^2\Big)$, where $|X|$ is the size of the domain on which the curves are defined. The query time is $O\big(m^{O(1)} \log n\big)$. The data structure precomputes all answers to queries with curves of length $\sqrt{m}$, leading to a very high space consumption.[1]

In the group of ℓ_p distances, the Fréchet distance most resembles the ℓ_∞-distance, which is notoriously hard to embed into a low-dimensional ℓ_p-space, see also the discussion by Indyk in [16]. Indyk's data structure for the discrete Fréchet distance is in fact an extension of his data structure for the ℓ_∞-distance [16]. Any subset of ℓ_∞^d can be embedded into the Fréchet metric[2] [18]. This embedding implies that, unless the strong exponential-time hypothesis fails, there exists no data structure for near-neighbor searching under the discrete Fréchet distance that achieves preprocessing time in $O\big(n^{2-\varepsilon} \operatorname{poly} m\big)$, query time in $O\big(n^{1-\varepsilon} \operatorname{poly} m\big)$ for any $\varepsilon > 0$, and approximation factor $c < 3$. This suggests that the problem becomes hard for long curves, i.e., $m \in \omega(\log n)$. Recently, Backurs and Sidiropoulos showed how to embed finite subsets of the Hausdorff distance into ℓ_∞ using constant distortion and constant dimension of the host space [2]. However, for the Fréchet distance, no non-trivial embeddings are known, see also the discussion in [18]. It is possible to embed any finite metric space into ℓ_p, for example, using the embedding due to Bourgain [25]. However, the high cost of computing the embedding makes it unfit for use in a nearest-neighbor data structure. Another known approach to proximity searching in metric spaces is to exploit a low doubling-dimension [1, 13]. However, the doubling dimension of the Fréchet distance is infinite, even if the metric space is restricted to curves of constant length [9]. Recently Bartal *et al.* [3] gave lower bounds for embedding doubling spaces. Their result implies that a metric embedding of the Fréchet distance into an ℓ_p-space would have at least super-constant distortion. However, as noted earlier, it is not even known how to obtain such an embedding.

In general, there is little known in terms of data structures for the Fréchet distance.

[1] Indyk also claims (without proof) a slightly different bound using a trade-off parameter $t \geq 2$: approximation factor $O((\log m + \log\log n)^{(t-1)})$, space $O\Big(\big(m^2|X|\big)^{tm^{1/t}} n^{2t}\Big)$ and query time $(m + \log n)^{O(t)}$. The space bound decreases at the cost of approximation and query time as soon as $t < \log m$; however, the trade-off disappear for larger values of t since all bounds increase in t as soon as $t \geq \log m$.

[2] In particular, one can use $3d$ vertices to express each d-dimensional vector as a curve on a real line.

The authors are aware of the following few results which were developed for the classic (continuous) Fréchet distance. De Berg, Cook and Gudmundsson [7] study range counting queries for the set of subcurves that lie within distance r to a query line segment. Their data structure uses a partition tree to store compressed subcurves. For any parameter $n \leq s \leq n^2$, the space used by the data structure is in $O(s \operatorname{poly} \log n)$. The queries are computed in time in $O\left(\frac{n}{\sqrt{s}} \operatorname{poly} \log n\right)$ and uses a constant approximation factor. However, the data structure does not support more complex query curves than line segments. A second data structure is due to Driemel and Har-Peled [8]. This data structure answers queries for the Fréchet distance of a subcurve to a query curve (the subcurve is specified in the query). If the queries are line segments, an approximation factor of $(1 + \varepsilon)$ can be achieved with logarithmic query time and linear space. Unlike the ℓ_∞-metric, which can be evaluated in time that is linear in the dimension, evaluating a single Fréchet distance is believed to take time that is at least roughly quadratic in the complexity of the curves (the number of vertices) in the worst case [4]. The high time complexity can be credited to the fact that the distance measure optimizes over all possible monotone alignments of the two input sequences. Computing the discrete Fréchet distance, as well as dynamic time warping, can be solved via dynamic programming. In both cases, the naive linear scan leads to $O(nm^2)$ query time for finding the nearest neighbor. For dynamic time warping (DTW) no data structures exist that give provable guarantees, however there exist many heuristics, see the work of Rakthanmanon *et al.* [26] (and references therein). Since DTW does not satisfy the triangle inequality, it cannot be embedded into an ℓ_p-space.

1.2 Our results

Our first result is a basic LSH scheme for the discrete Fréchet distance, which leads to a very efficient LSH with approximation factor that is linear in the number of curve vertices. The scheme is described in Section 3 and it is surprisingly simple: We snap the curves to a randomly shifted grid and remove consecutive duplicate vertices. It turns out that this simple scheme alleviates the alignment problem which sets the Fréchet distance computation apart from the ℓ_∞-distance. Next, we show in Section 4 that it is even possible to get constant approximation, at the cost of a lower collision probability for near curves. The second scheme randomly perturbs the vertices of the input curves independently and snaps the vertices to a fixed grid instead of a randomly shifted grid. It is natural to ask if there exists an LSH scheme exhibiting a full-spectrum trade-off between collision probability and approximation. We positively answer to this question in Section 5, with a scheme based on a random partition of the input curves, inspired by Indyk's data structure [17], followed by the application of the basic scheme to each subsequence independently.[3] (Due to space constraints, we refer to the full version of our paper [10] for more details.)

All the LSH schemes achieve zero false-positives, meaning that no collisions happen between far curves. When applied to the (c, r)-near neighbor problem, we obtain the results summarized in Table 1 (see also Section 2.3). It is interesting to compare our bounds with the state of the art. The basic scheme of Theorem 7 provides a data structure using almost linear space and $O(m \log n)$ query time by allowing a linear approximation $c = O(m)$. This query time always beats the trivial exact solution of scanning all input curves for each query, which needs $O(nm^2)$ time. In comparison, Indyk's result [17] provides a better approximation when

[3] Indeed, in Theorem 10, the collision probability for near curves is bounded by 2^{-3M} for $K = M$ (using Stirling's approximation for the binomial coefficient), however when summarizing our bounds we use the simplified bound from Corollary 11.

Table 1 Our approximate near-neighbor data structure results for the discrete Fréchet distance in comparison with the result by Indyk, assuming $d = O(1)$ for simplicity. The first four rows refer to the standard discrete Fréchet distance d_F, while the last two rows $d_{w,\mathrm{aF}}$ and $d_{w,\mathrm{sF}}$ refer to the anchored and speed constraints respectively. The input consists of n polygonal curves in $\mathbb{R}^d$, each of complexity at most m. The corresponding query results are achieved with high probability. The parameters $k \geq 1$ and $\ell \geq 1$ trade-off space/query time and approximation, and parameter w constrains the possible alignments. The first entry in bi-criteria $(\cdot, \cdot)$ denotes the distance approximation, while the second is the alignment approximation.

	Space	Query time	Approximation	Reference
d_F	$O\left(\lvert X\rvert^{\sqrt{m}}(m^{\sqrt{m}}n)^2\right)$	$O\left(m^{O(1)}\log n\right)$	$O(\log m + \log\log n)$	[17]
	$O(n\log n + nm)$	$O(m\log n)$	$O(m)$	Thm. 7
	$O(2^{4md}n\log n + nm)$	$O(2^{4md}m\log n)$	$O(1)$	Thm. 9
	$O(2^{2k}m^{k-1}n\log n + nm)$	$O(2^{2k}m^k\log n)$	$O(m/k)$	Cor. 11
$d_{w,\mathrm{aF}}$	$O\left(\left(\sqrt{2}w\right)^{2m/\ell} n\log n + nm\right)$	$O\left(\left(\sqrt{2}w\right)^{2m/\ell} m\log n\right)$	bi-criteria $\left(4d^{\frac{3}{2}}\ell, 2\ell - 2\right)$	Thm. 12
$d_{w,\mathrm{sF}}$	$O\left(\left(\sqrt{2}w\ell\right)^{2m/\ell} n\log n + nm\right)$	$O\left(\left(\sqrt{2}w\ell\right)^{2m/\ell} m\log n\right)$	bi-criteria $\left(4d^{\frac{3}{2}}\ell, \ell\right)$	Thm. 13

$m = \Omega(\log\log n)$ but it uses exponential space and slightly higher query time. More generally, when curves are short $m = o(\log n)$, our basic result provides a good alternative to Indyk's result due to the improved space. In the particular case where $m \leq \frac{\alpha}{4d}\log n$ for some fixed $\alpha > 0$, we can answer queries using a constant approximation factor in $O(n^\alpha \log^2 n)$ time and using $O(n^{1+\alpha}\log n)$ space, using Theorem 9. When curves have constant complexity, the basic LSH gives the first efficient data structure with constant approximation. We recall that a data structure for the (c, r)-approximate near neighbor problem can be used as a building block for solving the c-approximate nearest neighbor problem [28].

We then address LSH for the discrete Fréchet distance under alignment constraints in Section 6. It is natural to constrain the alignments of curves: this preserves important characteristics of the input curves and it also reduces the actual time to compute the distance between curves (see e.g., [27, 21]). We target the anchored and bounded speed constraints that require, respectively, a vertex to be aligned with at most w vertices or to be aligned with vertices whose indices differ by at most $w/2$, for a suitable parameter $w \geq 1$ (for formal definitions see Section 2.2). Our scheme provides the first data structures for the (c, r)-near neighbor problem with alignment constraints. Further, they exhibit a bi-criteria approximation: it is possible to reduce space and query time with a weaker approximation on the distance but also on the alignment parameter w. Bounds are summarized in Table 1.

In Section 7, we study which one of our schemes work for DTW. We show that the basic LSH applies to DTW with the same linear approximation, space and query bounds of the discrete Fréchet distance. In contrast, the techniques to improve the approximation factor under the Fréchet distance do not provide improvements for DTW. The LSH schemes for constrained distances also yields linear approximation for DTW distance, but maintains the trade-off between space/query time and the approximation on the alignment parameter w.

2 Preliminaries

2.1 Distance measures for curves

A *time series* (or *trajectory*)[4] is a series $(p_1, t_1), \ldots, (p_m, t_m)$ of measurements p_i of a signal taken at times t_i. We assume $0 = t_1 < t_2 < \ldots < t_m = 1$ and m is finite. A time series may be

[4] Usually, these are referred to as time series when $d = 1$ and trajectories when $d > 1$.

viewed as a continuous function $P : [0,1] \to \mathbb{R}^d$ by linearly interpolating $p_1, \ldots, p_m$ in order of t_i, $i = 1, \ldots m$. We obtain a polygonal curve with *vertices* $p_1 = P(t_1), \ldots, p_m = P(t_m)$ and segments between p_i and p_{i+1} called *edges* $\overline{p_i p_{i+1}} = \{xp_i + (1-x)p_{i+1} | x \in [0,1]\}$. We will simply refer to P as a *curve*. We denote the space of all curves in $\mathbb{R}^d$ with Δ^d.

We now recall the definitions of discrete Fréchet distance and of the dynamic time warping distance between two curves. To this end we define the concept of traversal. Given two polygonal curves $P = p_1, \ldots, p_{m_1}$ and $Q = q_1, \ldots, q_{m_2}$, a *traversal*

$$T = \{(i_1, j_1), (i_2, j_2), \ldots, (i_\ell, j_\ell)\}$$

is a sequence of pairs of indices referring to a *pairing* of vertices from the two curves with the following properties:

(i) $i_1 = 1$, $j_1 = 1$, $i_\ell = m_1$, and $j_\ell = m_2$

(ii) $\forall (i_k, j_k) \in T : (i_{k+1} - i_k) \in \{0,1\} \wedge (j_{k+1} - j_k) \in \{0,1\}$.

(iii) $\forall (i_k, j_k) \in T : (i_{k+1} - i_k) + (j_{k+1} - j_k) \geq 1$.

Intuitively, one can think of the traversal as a prescribed schedule for simultaneously traversing the two curves, starting at the first vertex of each curve, in every step the traversal advances by one vertex, either on one of the curves, or on both curves simultaneously, finally the traversal has to end at the last vertices of the two curves.

We consider the maximum distance of two vertices paired by a traversal as the cost incurred by this traversal. Let $\mathcal{T}$ be the set of possible traversals for two curves P and Q, then the Fréchet distance corresponds to the minimal cost of a traversal of the two curves. Likewise, if we define the cost of a traversal as the sum of distances between paired vertices, then the traversal with minimum cost corresponds to the dynamic time warping distance.

▶ **Definition 1.** Let $\mathcal{T}$ be the set of possible traversals for two curves P and Q. The *discrete Fréchet distance* $d_F(P,Q)$ between curves P and Q is defined as

$$d_F(P,Q) = \min_{T \in \mathcal{T}} \max_{(i_k, j_k) \in T} \|p_{i_k} - q_{j_k}\|.$$

▶ **Definition 2.** Let $\mathcal{T}$ be the set of possible traversals for two curves P and Q. The *dynamic time warping (DTW) distance* $d_{DTW}(P,Q)$ between curves P and Q is defined as

$$d_{\mathrm{DTW}}(P,Q) = \min_{T \in \mathcal{T}} \sum_{(i_k, j_k) \in T} \|p_{i_k} - q_{j_k}\|.$$

The discrete Fréchet distance satisfies the triangle inequality and is a pseudo-metric. This is not true for the DTW distance, since it does not satisfy the triangle inequality.

We refer to a traversal realizing the distance of two curves as an *optimal traversal*. We can interpret a traversal as the edges of a bipartite graph where the nodes are the vertices of the two curves and the edges connect the pairs. The following simple lemma holds for all distance measures. As a consequence, we assume in the paper that an optimal traversal consists of disconnected stars, that we call *components*.

▶ **Lemma 3.** *For any two curves $P = p_1, \ldots, p_{m_1}$ and $Q = q_1, \ldots, q_{m_2}$, there always exists an optimal traversal T with the following two properties:*

(i) *T consists of at most $m = \min\{m_1, m_2\}$ disconnected components.*

(ii) *Each component is a star, i.e., all edges of this component share a common vertex.*

Proof. The first part is immediate, since we can charge each component to a vertex of the shorter curve that is contained in it. To see the second part of the claim, assume for the sake of contradiction that an optimal traversal has the pairs $(i, j), (i, j + 1)(i + 1, j + 1)$ for some i, j (or the symmetric configuration $(i, j), (i + 1, j)(i + 1, j + 1)$). In this case, the middle pair $(i, j + 1)$ can be removed without increasing the cost and without invalidating the traversal properties. We can apply this reasoning repeatedly until each component is a star. ◀

2.2 Distances measures with constraints

Anchored distances. A traversal T is said *w-anchored traversal* if each vertex is paired with a vertex at distance at most $w/2$ (for simplicity we assume w to be even): namely, $|i - j| \leq w/2$ for each $(i, j) \in T$. Parameter w is called the *width* of the traversal. Such a traversal exists only if $|m_1 - m_2| \leq w/2$, otherwise there would be unpaired vertices (e.g., the last vertex of the longest curve). For two curves P and Q with lengths satisfying $|m_1 - m_2| \leq w/2$, we define the *w-anchored discrete Fréchet distance $d_{w,aF}(P, Q)$* and *w-anchored DTW distance $d_{w,aDTW}(P, Q)$* as in Definitions 1 and 2 where $\mathcal{T}$ is defined as the set of all possible w-anchored traversals.

Speed-constrained distances. A traversal T is a *w-speed traversal* if each vertex is aligned with at most w vertices of the other curve: in other terms, the bipartite graph representing the traversal has degree at most w. Parameter w is called the *speed* of the traversal. (We overload the meaning of w since the width and speed parameters play a similar role in our algorithms.) Note that a w-anchored traversal is a $(w + 1)$-speed traversal, but the opposite is not necessary true. A w-speed traversal exists only if $1/w \leq m_1/m_2 \leq w$. For two curves P and Q with lengths satisfying $1/w \leq m_1/m_2 \leq w$, we defined the *w-speed discrete Fréchet distance $d_{w,sF}(P, Q)$* and *w-speed DTW distance $d_{w,sDTW}(P, Q)$* as in Definitions 1 and 2 where $\mathcal{T}$ is defined as the set of all possible w-speed traversals.

2.3 Locality-sensitive hashing

We use the notion of asymmetric locality-sensitive hashing (see, e.g. [29]), defined as follows:

▶ **Definition 4.** Let $\mathcal{S}$ be the set of curves in $\mathbb{R}^d$ and let $d : \mathcal{S} \times \mathcal{S} \to \mathbb{R}^+$ be a distance measure defined on them. Given real values $r > 0$, $c > 1$, $0 \leq \alpha_1 \leq 1$ and $0 \leq \alpha_2 \leq 1$ with $\alpha_1 > \alpha_2$, a family $\mathcal{H}$ of pairs of hash functions (h_1, h_2) is called *$(r, c, \alpha_1, \alpha_2)$-sensitive* if for any two curves $P, Q \in \mathcal{S}$

(i) if $d(P, Q) \leq r$, then $Pr_{(h_1,h_2)\in\mathcal{H}}(h_1(P) = h_2(Q)) \geq \alpha_1$;

(ii) if $d(P, Q) > cr$, then $Pr_{(h_1,h_2)\in\mathcal{H}}(h_1(P) = h_2(Q)) \leq \alpha_2$.

When $h_1 = h_2$, we have the traditional definition of (symmetric) locality-sensitive hashing. The above scheme is *asymmetric* in the sense that there are two different schemes and the guarantees only hold for curves P and Q where P was hashed using the first scheme and Q was hashed using the second scheme. This is useful, e.g., if the application of the LSH is a nearest neighbor data structure, where comparisons only need to be done between input objects and query objects.

The results reported in Table 1 follow by applying the standard framework for solving the (c, r)-near neighbor problem with an $(r, c, \alpha_1, \alpha_2)$-sensitive hashing scheme $\mathcal{H}$. For the sake of completeness, we sketch this process here.[5] A new family $\mathcal{H}'$ of hashing is constructed by

[5] We observe that the LSH schemes presented in this paper have long hash values (curves or array of curves). However, they can be shortened with traditional hashing (i.e., by mapping each value in $[0, O(n)]$), that allows for a more efficient search in the hash tables generated by the LSH. This technique increases α_2 by an additive $O(1/n)$ term.

concatenating $k = \max\{1, \log_{\alpha_2}(1/n)\}$ hash functions from $\mathcal{H}$, so that the collision probability of far points is at most $1/n$. Then, each point in S is inserted into $L = (1/\alpha_1)^k$ hash tables, each corresponding to a different randomly chosen hash function from $\mathcal{H}'$. For a query point q, the algorithm searches among all points that collide with q in the L hash tables and stops as soon as a cr-near neighbor is found. When $\alpha_2 > 0$, the data structure requires $O\big(n^{1+\rho} + nm\big)$ memory words and query time $O(\Gamma n^{\rho})$, where $\Gamma = \Omega(m)$ is the time required for computing the distance between two curves and $\rho = \log\alpha_1/\log\alpha_2$. When $\alpha_2 = 0$, the data structure requires $O(n/\alpha_1)$ memory words and query time $O(m/\alpha_1)$. Note that in this case the query time does not include Γ: the algorithm does not need to compute distances between q and points in the buckets since there are no false positives. For a given query, the data structures returns an approximate cr-near neighbor with constant probability. To obtain high probability (i.e., at least $1-1/n$) we repeat the above process $\log n$ times, leading to $\log n$ different data structures. This increases space and query time by a $O(\log n)$ term.

3 Linear approximation factor

We first present the basic LSH scheme in Section 3.1, and then in Section 3.2 we analyze its correctness and performance for the discrete Fréchet distance. The basic LSH has an approximation factor that is linear in the number of vertices that a curve can have.

3.1 Algorithm

We use a randomly shifted grid in our hashing scheme. Let the canonical d-dimensional grid of resolution δ be defined as an evenly spaced point set in $\mathbb{R}^d$, as follows:

$$G_\delta = \Big\{(x_1, \ldots, x_d) \in \mathbb{R}^d \mid \forall\, 1 \le i \le d\ \exists\, j \in \mathbb{N}\ :\ x_i = j \cdot \delta\Big\}.$$

Consider a family of such grids parametrized by a shift t:

$$\widehat{G}^t_\delta = \{p + t \mid p \in G_\delta\}.$$

Choosing t uniformly at random from the half-open hypercube $[0,\delta)^d$ we obtain a family of randomly shifted grids. Let $P \in \Delta^d$ be a polygonal curve with vertices $p_1, \ldots, p_m$ and let $h^t_\delta : \Delta^d \to \Delta^d$ be a hash function. The curve $h^t_\delta(P)$ is defined as the result of the following two-stage construction.

(i) We snap the curve to the grid $\widehat{G}^t_\delta$. More precisely, we replace each vertex p_i with its closest grid point $p'_i = \arg\min_{q\in\widehat{G}^t_\delta} \|p_i - q\|$ to obtain the curve P'.

(ii) We remove consecutive duplicates in P'. That is, we remove the vertex p'_i if it is identical to p'_{i-1}.

Let $\mathcal{H}^{\mathrm{L}}_\delta$ be the family of hash functions h^t_δ constructed this way.

3.2 Analysis

▶ **Lemma 5.** *Let $P, Q \in \Delta^d$ be two curves with m_1 and m_2 points, respectively, and let $m = \min\{m_1, m_2\}$. For any $\delta > 0$, it holds that*

$$Pr_{\mathcal{H}^{\mathrm{L}}_\delta}\big(h^t_\delta(P) = h^t_\delta(Q)\big) \ge 1 - \left(2dm \cdot \frac{d_F(P,Q)}{\delta}\right).$$

Proof. We bound the probability that P and Q do not hash to the same sequence. To this end, consider an optimal traversal T of P and Q with respect to the discrete Fréchet distance.

By Lemma 3, we can assume that $|T| \leq m$ and each component is a star. Let ℓ denote the number of components of T. For $1 \leq k \leq \ell$ denote with E_k the event that not all vertices of the k-th component are snapped to the same grid point. This happens only if at least one pair of vertices is separated in at least one dimension by the random shift t.

Since the component is a star, there exists a vertex v of either P or Q, such that v is involved in all pairs of T in the k-th component. Therefore, all vertices in this component have distance at most $d_F(P, Q)$ to v. Since t is uniformly distributed in $[0, \delta)^d$, the probability that any pair is separated along any fixed dimension is $2d_F(P, Q)/\delta$. As a consequence, event E_k happens with probability at most $2d \cdot d_F(P, Q)/\delta$.

By a union bound over the ℓ components in T, we have that the probability of P and Q not being hashed to the same sequence is bounded by

$$Pr\left(\bigcup_{1\leq k\leq \ell} E_k\right) \leq \sum_{1\leq k\leq \ell} Pr(E_k) = 2dm \cdot \frac{d_F(P,Q)}{\delta}$$

and the lemma follows. ◀

▶ **Lemma 6.** *For any value of δ and for any $P, Q \in \Delta^d$, if there exists a value of $t \in [0, \delta)^d$ such that $h_\delta^t(P) = h_\delta^t(Q)$, then it holds that $d_F(P, Q) \leq \sqrt{d} \cdot \delta$.*

Proof. In the case that $h_\delta^t(P) = h_\delta^t(Q)$, it holds that $d_F(h_\delta^t(P), h_\delta^t(Q)) = 0$. Snapping a curve to the randomly shifted grid changes the position of each vertex by at most $\frac{\sqrt{d}}{2} \cdot \delta$. Therefore, it holds that $d_F(P, h_\delta^t(P)) \leq \frac{\sqrt{d}}{2} \cdot \delta$ and similarly $d_F(Q, h_\delta^t(Q)) \leq \frac{\sqrt{d}}{2} \cdot \delta$. By the triangle inequality, $d_F(P, Q) \leq d_F(h_\delta^t(P), P) + d_F(h_\delta^t(P), h_\delta^t(Q)) + d_F(h_\delta^t(Q), Q) \leq \sqrt{d} \cdot \delta$. ◀

The next theorem follows by plugging in the bounds of Lemmas 5 and 6.

▶ **Theorem 7.** *Let $P, Q \in \Delta^d$ be two curves with m_1 and m_2 points, respectively, and let $m = \min\{m_1, m_2\}$, $\delta = 4dmr$ and $c = 4d^{\frac{3}{2}}m$. It holds that:*

(i) *if $d_F(P, Q) < r$, then $Pr_{\mathcal{H}_\delta^L}(h_\delta^t(P) = h_\delta^t(Q)) > \frac{1}{2}$;*

(ii) *if $d_F(P, Q) > cr$, then $Pr_{\mathcal{H}_\delta^L}(h_\delta^t(P) = h_\delta^t(Q)) = 0$.*

4 Constant approximation factor

In the previous section we analyzed a very efficient LSH with linear approximation factor. On the other end of the spectrum, we can also design an LSH with constant approximation factor, but higher running time. Conceptually, the easiest way to do this is to randomly and independently perturb the vertices of each curve and snap them to a fixed grid.

4.1 Algorithm

The described scheme is asymmetric. We assume that we have two types of curves, which we call input curves and query curves. Consider an input curve $P = p_1, \ldots, p_m$, and let G_δ be the canonical d-dimensional grid of resolution δ defined in the previous section. Let $t_P = t_1, \ldots, t_m$ be a sequence of independent random variables which are uniformly distributed in $\left[-\frac{\delta}{2}, \frac{\delta}{2}\right]^d$. We perturb the vertices of P: Let $P' = p'_1, \ldots, p'_m$ be the perturbed curve with $p'_i = p_i + t_i$. We snap the curve P' to the grid G_δ. More precisely, we replace each vertex p'_i with its closest grid point $p''_i = \arg\min_{q \in G_\delta} \|p'_i - q\|$ to obtain the curve P''. In the next step we remove consecutive duplicates in P''. That is, we remove the vertex p''_i if it is identical to p''_{i-1}. We define $h_\delta^{t_P}(P)$ to be the result of this algorithm.

For a query curve Q, the hash function is the same. However, a different random sequence t_Q is used for randomly perturbing the curve. We let $\mathcal{H}^{\mathfrak{C}}_\delta$ denote the LSH scheme defined this way: namely, $\mathcal{H}^{\mathfrak{C}}_\delta$ contains all pairs $(h^{t_P}_\delta, h^{t_Q}_\delta)$, where vectors t_P and t_Q consist of entries independent and identically distributed in $\left[-\frac{\delta}{2}, \frac{\delta}{2}\right]^d$.

4.2 Analysis

▶ **Lemma 8.** *Let $P, Q \in \Delta^d$ be two curves with m_1 and m_2 points, respectively. Let $m = \min\{m_1, m_2\}$ and let $M = \max\{m_1, m_2\}$. For any $\delta > 0$, it holds that*

$$Pr_{\mathcal{H}^{\mathfrak{C}}_\delta}\left(h^{t_P}_\delta(P) = h^{t_Q}_\delta(Q)\right) \geq \left(\frac{1}{2}\right)^{dm} \cdot \left(\frac{1}{2} - \frac{d_F(P,Q)}{\delta}\right)^{dM}$$

In particular, if $\delta > 4d_F(P,Q)$, then the probability is strictly lower bounded by $2^{-2d(m_1+m_2)}$.

Proof. Note that for $d_F(P,Q) \geq \frac{\delta}{2}$ the claim is trivially true. Therefore, assume that $d_F(P,Q) < \frac{\delta}{2}$. For simplicity assume first that $d = 1$. We bound the probability that P and Q do not hash to the same sequence. To this end, consider an optimal traversal T of P and Q with respect to the discrete Fréchet distance. By Lemma 3, we can assume that $|T| \leq m_1 + m_2$ and each component is a star. Let ℓ denote the number of components of T. For $1 \leq k \leq \ell$ denote with E_k the event that not all vertices of the k-th component are snapped to the same grid point. Assume that the center of the k-th star is a vertex p_i of P and that the other vertices of the component are vertices $q_j, \ldots, q_{j+c_k}$ of Q. The analysis for the case where the center is a vertex of Q is analogous. There must be a grid point in either one of the two intervals to the left and to the right of p_i: $I_l = [p_i - \frac{\delta}{2}, p_i)$ and $I_r = [p_i, p_i + \frac{\delta}{2})$. We analyze the case that there is a grid point in I_r, the other case is analogous. Let X_i be the event that $p'_i \in I_r$. Since t_P is uniformly random in $\left[-\frac{\delta}{2}, \frac{\delta}{2}\right]^{m_1}$, it holds that $Pr(X_i) \geq \frac{1}{2}$. Now, let Y_j be the event that $q'_j \in I_r$. If q_j was in p_i's component, then there are two cases. Either q_j lies in I_l or in I_r. In the first case, we have

$$Pr(Y_i) \geq \frac{\frac{\delta}{2} - |p_i - q_j|}{\delta} \geq \frac{1}{2} - \frac{d(P,Q)}{\delta},$$

and in the second case we have $Pr(Y_i) \geq \frac{1}{2}$. We can bound the probability that all vertices in the k-th component snap to the same grid point

$$Pr\left(\overline{E_k}\right) \geq Pr(X_i \cap Y_j \cap \cdots \cap Y_{j+c_k}) \geq \frac{1}{2} \cdot \left(\frac{1}{2} - \frac{d(P,Q)}{\delta}\right)^{c_k}.$$

If all components are preserved, then the two curves will hash to the same sequence, therefore

$$\begin{aligned} Pr\left(h^{t_P}_\delta(P) = h^{t_Q}_\delta(Q)\right) &\geq Pr\left(\bigcap_{1\leq k\leq \ell} \overline{E_k}\right) \geq \prod_{1\leq k\leq \ell} Pr\left(\overline{E_k}\right) \\ &\geq \prod_{1\leq k\leq \ell} \frac{1}{2}\left(\frac{1}{2} - \frac{d(P,Q)}{\delta}\right)^{c_k} \geq \left(\frac{1}{2}\right)^{\ell}\left(\frac{1}{2} - \frac{d(P,Q)}{\delta}\right)^{m_1+m_2-\ell}. \end{aligned}$$

The last inequality follows since $\left(\sum_{1\leq k\leq \ell} c_k\right) = m_1 + m_2 - \ell$. Indeed, each center of a component can be charged to this component and the remaining vertices make up the sum of the leaves of all components. The lemma is now implied for $d = 1$ observing that $\ell \leq \min\{m_1, m_2\}$, as implied by Lemma 3. We get the lemma for general d by observing that the dimensions are independent. ◀

The next theorem follows by plugging in the bounds of Lemma 8 and by using same arguments as in the proof of Lemma 6.

▶ **Theorem 9.** *Let $P, Q \in \Delta^d$ be two curves with m_1 and m_2 points, respectively, and let $\delta = 4dr$ and $c = 4d^{3/2}$. It holds that*

(i) *if $d_F(P,Q) < r$, then $Pr_{\mathcal{H}^c_\delta}\left(h^{t_P}_\delta(P) = h^{t_Q}_\delta(Q)\right) > \left(\frac{1}{2}\right)^{2d(m_1+m_2)}$;*

(ii) *if $d_F(P,Q) > cr$, then $Pr_{\mathcal{H}^c_\delta}\left(h^{t_P}_\delta(P) = h^{t_Q}_\delta(Q)\right) = 0$.*

5 Trade-off between approximation factor and query time

In the previous two sections we have seen schemes with linear and constant approximations. We now suggest a scheme exhibiting a trade-off between the collision probability of near points and the approximation factor. The basic idea is to randomly partition the input curves and to concatenate the outcome of the basic LSH (Section 3) applied to the different parts of the curves.

5.1 Algorithm

The scheme is asymmetric. Again, we assume that we have two types of curves, which we call input and query curves. The difference in how they are handled lies in the way we create the partition. For an input curve $P = p_1, \ldots, p_m$, we randomly sample a partition into K subsequences. To this end, we denote a partition of P with $\Phi^s(P) = \left(\widehat{P}_1, \ldots, \widehat{P}_K\right)$ where the subsequences are defined by a monotone sequence $s \in [m]^{K-1}$ as follows.

$$\widehat{P}_1 = p_1, \ldots, p_{s_1}; \qquad \forall\, 1 < i < K \;:\; \widehat{P}_i = p_{s_{i-1}}, \ldots, p_{s_i}; \qquad \widehat{P}_K = p_{s_{K-1}}, \ldots, p_m.$$

There are at most $\binom{m+K-1}{K-1}$ ways to partition a curve of length m in this way. We denote with $\mathcal{P}_K$ the family of all valid partitions for a given m. Let $t = t_1, \ldots, t_K$ be a sequence of independent random values evenly distributed in $[0,\delta)^d$. Once we have partitioned the input curve P into K (overlapping) subsequences, we apply the basic LSH to each individual subsequence and concatenate the resulting curves:

$$g^{t,s}_{\delta,K}(P) = h^{t_1}_\delta\left(\widehat{P}_1\right) \oplus h^{t_2}_\delta\left(\widehat{P}_2\right) \oplus \cdots \oplus h^{t_K}_\delta\left(\widehat{P}_K\right).$$

A query curve $Q = q_1, \ldots, q_m$ is subdivided into K equal-sized subsequences (deterministically), where the last subsequence may be shorter and two consecutive sequences overlap by one element. We denote with $\Phi^*(Q)$ this partitioning into equal-sized subsequences. For query curves, we define $g^{t,*}_{\delta,K}(Q)$ to be the resulting curve given by applying the basic LSH to each individual subsequence and concatenating the resulting curves. For any given $\delta > 0$ and $K \geq 1$, we denote with $\mathcal{H}^{\mathsf{T}}_{\delta,K}$ the family of asymmetric hash functions created this way: that is, $\mathcal{H}^{\mathsf{T}}_{\delta,K}$ consists of tuples $(g^{t,s}_{\delta,K}, g^{t,*}_{\delta,K})$ where the entries of t are independently and identically distributed in $[0,\delta)^d$ and $\Phi^s(P)$ is uniformly chosen at random from $\mathcal{P}_K$.

5.2 Analysis

We have the following theorem which generalizes Theorem 7. Using the parameter K we get a tradeoff between approximation factor and query time.

▶ **Theorem 10.** *Let $P, Q \in \mathcal{S}$ be two curves with m_1 and m_2 points, respectively, and let $M = \max\{m_1, m_2\}$. Let $K \geq 1$ be a given integer and let $\delta = 4dr \cdot \left\lceil \frac{M}{K} \right\rceil$ and $c = 4d^{\frac{3}{2}} \cdot \left\lceil \frac{M}{K} \right\rceil$. It holds that*

(i) *if* $d_F(P,Q) < r$*, then* $Pr_{\mathcal{H}^{\mathcal{T}}_{\delta,K}}\left(g^{t,s}_{\delta,K}(P) = g^{t,*}_{\delta,K}(Q)\right) \geq \left(\frac{1}{2}\right)^K \cdot \binom{M+K-1}{K-1}^{-1}$*;*

(ii) *if* $d_F(P,Q) > cr$*, then* $Pr_{\mathcal{H}^{\mathcal{T}}_{\delta,K}}\left(g^{t,s}_{\delta,K}(P) = g^{t,*}_{\delta,K}(Q)\right) = 0$.

Proof. We first prove (i). Let T be an optimal traversal of P and Q. We say two partitions $\Phi^s(P)$ and $\Phi^r(Q)$ are *consistent* with respect to T if and only if $(s_i, r_i) \in T$ for all $1 \leq i \leq K-1$. Let E denote the event that the partition $\Phi^s(P)$ used in the hash functions is consistent with $\Phi^*(Q)$ with respect to T. By construction this happens for at least one of the partitions in $\mathcal{P}_K$. Therefore, $Pr(E) \geq \frac{1}{|\mathcal{P}_K|}$. Now, let E_i be the event that $h^{t_i}_\delta\left(\widehat{P}_i\right) = h^{t_i}_\delta\left(\widehat{Q}_i\right)$. By Lemma 5 we have that

$$Pr(E_i \mid E) \geq 1 - \left(2dm' \cdot \frac{d_F\left(\widehat{P}_i, \widehat{Q}_i\right)}{\delta}\right) \geq 1 - \left(2d\left\lceil \frac{M}{K} \right\rceil \cdot \frac{d_F(P,Q)}{\delta}\right) \geq \frac{1}{2}.$$

Note that we can assume $m' \leq \lceil \frac{M}{K} \rceil$ in the above inequality, since m' is the length of the shorter of the two subsequences in the lemma. By construction, the length of $\widehat{Q}_i$ will be at most $\lceil \frac{M}{K} \rceil$.

Since the values t_i are chosen pairwise independent, we have

$$Pr_{\mathcal{H}^{\mathcal{T}}_{\delta,K}}\left(g^{t,s}_{\delta,K}(P) = g^{t,*}_{\delta,K}(Q)\right) \geq \left(\prod_{1 \leq i \leq K} Pr(E_i \mid E)\right) \cdot Pr(E) \geq \left(\frac{1}{2}\right)^K \cdot \frac{1}{|\mathcal{P}_K|}.$$

Using $|\mathcal{P}_K| \leq \binom{M+K-1}{K-1}$, the first part of the claim follows.

As for the second part of the claim, we can use Lemma 6 applied to the subsequences. If there exists a partition of P, and there exist $t = t_1, \ldots, t_K$, such that for all $0 \leq i \leq K$ $h^{t_i}_\delta\left(\widehat{P}_i\right) = h^{t_i}_\delta\left(\widehat{Q}_i\right)$, then it holds by Lemma 6 that $d_F\left(\widehat{P}_i, \widehat{Q}_i\right) \leq \sqrt{d} \cdot \delta$. In this case, we can combine the traversals of the subsequences to a traversal of the entire curves. This combined traversal has the same cost, therefore it follows that $d_F(P,Q) \leq \sqrt{d} \cdot \delta$. Consequently, if $d_F(P,Q) > cr = 4d^{\frac{3}{2}}Mr/K = \sqrt{d} \cdot \delta$, then it cannot happen that $g^{t,s}_\delta(P) = g^{t,*}_\delta(Q)$ for any combination of $t = t_1, \ldots, t_K$ and s. ◀

▶ **Corollary 11.** *Let $P, Q \in \mathcal{S}$ be two curves with m_1 and m_2 points, respectively, and let $M = \max\{m_1, m_2\}$. Let $K \geq 1$ be a given integer and let $\delta = 4dr \cdot \lceil \frac{M}{K} \rceil$ and $c = 4d^{\frac{3}{2}} \cdot \lceil \frac{M}{K} \rceil$. It holds that*

(i) *if* $d_F(P,Q) < r$*, then* $Pr_{\mathcal{H}^{\mathcal{T}}_{\delta,K}}\left(g^{t,s}_{\delta,K}(P) = g^{t,*}_{\delta,K}(Q)\right) > \left(\frac{1}{4}\right)^K \cdot \left(\frac{1}{M}\right)^{K-1}$*;*

(ii) *if* $d_F(P,Q) > cr$*, then* $Pr_{\mathcal{H}^{\mathcal{T}}_{\delta,K}}\left(g^{t,s}_{\delta,K}(P) = g^{t,*}_{\delta,K}(Q)\right) = 0$.

6 Handling constrained alignments

We now focus on LSH for discrete Fréchet distance with constraints on the alignment. We first target the w-anchored distance in Section 6.1, and then the w-speed distance in Section 6.2. As in the previous sections, the schemes are asymmetric and consist of a partitioning of the curve into subsequences and on the application of the basic LSH scheme to each subsequence. However, the partitions are different since they leverage on random processes on both input and query curves, consecutive subsequences do not overlap, and the constraints are exploited. We let $\ell \geq 1$ denote an arbitrary given integer that allows to trade-off the collision probability of near curves with a bi-criteria approximation on the distance and on the anchored alignment.

6.1 LSH for anchored distances

Consider an input curve $P = p_1, \dots, p_m$ and let $r_P = r_{P,1}, r_{P,2}, \dots r_{P,m}$ and $t = t_1, t_2, \dots, t_m$ denote sequences of independent and identically distributed random variables in $[1, w/2]$ and $[0, \delta)^d$ respectively, where δ is a suitable parameter defined later. The partition of P consists of a fixed partitioning into subsequences of length ℓ, followed by a random perturbation of subsequence lengths. Specifically, the following three operations are performed:

(i) Partition P into subsequences $\widehat{P'}_1, \dots, \widehat{P'}_{K'}$ with $K' = \lceil m/\ell \rceil$ of size ℓ, with the possible exception of the last subsequence. Let $s' \in [m]^{K'+1}$ be the vector denoting the final indexes of each subsequence, that is $\widehat{P}_i = p_{(s'_{i-1}+1)}, \dots, p_{s'_i}$: we have $s'_0 = 0$, $s'_{K'} = m$ and $s'_i = i\ell$ for each $1 \leq i < K'$.

(ii) Random perturb the final index of each subsequence with the random vector r_P: for each $1 \leq i < K'$, set $s_i = \min\{s_i + r_{p,2}, m\}$.

(iii) Clean the partition by removing overlaps among subsequences: for each $1 \leq i < K'$ and starting from $i = 1$, remove each subsequence where $s'_i \leq s'_j$ for some $j < i$. We let $\Phi^{r_P}(P) = \left(\widehat{P}_1, \dots, \widehat{P}_K\right)$ denote the resulting partition of P with $K \leq \lceil m/\ell \rceil$ and let $s_P \in [m]^{K+1}$ be the resulting vector denoting the final indexes of each subsequence (note that each subsequence has now length at most $\ell + w$).

Once curve P has been partitioned into K subsequences, we apply the basic LSH in Section 3 to each subsequence using the random shifts given by sequence t. Specifically, we snap the i-th subsequence $\widehat{P}_i$ on a grid of side δ shifted by the random value t_i and remove consecutive duplicates within each subsequence; the remaining values denote the hash value of $\widehat{P}_i$ and we denote them with $h_\delta^{t_i}\left(\widehat{P}_i\right)$. The final hash value $g_{w,\delta,\ell}^{t,r_P}(P)$ of curve P is the array containing the hash of each subsequence, specifically:

$$g_{w,\delta,\ell}^{t,r_P}(P) = \left(h_\delta^{t_1}\left(\widehat{P}_1\right), h_\delta^{t_2}\left(\widehat{P}_2\right), \dots, h_\delta^{t_K}\left(\widehat{P}_K\right)\right).$$

We observe that the final hash value is not a curve as in previous sections, but an array of curves. Equality between two curves then holds only if the two hash values have the same length and coincide in each position (i.e., the hash values $((a, b), (c))$ and $((a), (b, c))$ do not collide, but they collide if their are considered as a single curve (a, b, c)). This enforces the alignment constraint.

The hash process of a query curve Q is the same: however, a different random sequence r_Q is used to partition the curve, while the same sequence t of random shifts is kept. Due to the different random bits in r_Q the proposed LSH scheme is asymmetric. We let $\mathcal{H}^{\mathtt{A}}_{w,\delta,\ell}$ denote the hash family consisting of all possible pairs of hash functions $\left(g_{w,\delta,\ell}^{t,r_P}, g_{w,\delta,\ell}^{t,r_Q}\right)$.

The next theorem shows that the scheme has a bi-criteria approximation: In addition to the distance approximation c, the scheme has also an approximation on the alignment. As an example, we observe that two curves with a w-anchored distance larger than cr can still collide if they have a $w + 2(\ell - 1)$-anchored distance lower than cr.

▶ **Theorem 12.** *Let $P, Q \in \mathcal{S}$ be two curves with m_1 and m_2 points, respectively and let $m = \min\{m_1, m_2\}$. Let $\ell \geq 1$ be an arbitrary integer, $\delta = 4dr\ell$, and $c = 4d^{\frac{3}{2}}\ell$. Then, it holds that:*

(i) *if $d_{w,aF}(P, Q) < r$, then $Pr_{\mathcal{H}^{\mathtt{A}}_{w,\delta,\ell}}\left(g_{w,\delta,\ell}^{t,r_P}(P) = g_{w,\delta,\ell}^{t,r_Q}(Q)\right) > \left(1/\sqrt{2}w\right)^{2m/\ell}$;*

(ii) *if $d_{(w+2(\ell-1)),aF}(P, Q) > cr$, then $Pr_{\mathcal{H}^{\mathtt{A}}_{w,\delta,\ell}}\left(g_{w,\delta,\ell}^{t,r_P}(P) = g_{w,\delta,\ell}^{t,r_Q}(Q)\right) = 0$.*

6.2 LSH for bounded-speed distances

Consider an input curve $P = p_1, \ldots, p_m$ and let $r_P = r_{P,1}, r_{P,2}, \ldots, r_{P,m}$ and $t = t_1, t_2, \ldots t_m$ denote sequences of independent and identically distributed random variables in $[1, w\ell]$ and $[0, \delta)^d$ respectively. We random partition curve P into non overlapping subsequences of length given by the random sequence r_p. Specifically, let $\Phi^s(P) = \left(\widehat{P}_1, \ldots, \widehat{P}_K\right)$ denote a partition of P with $m/(w\ell) \leq K \leq m$ and let $s \in [m]^{K+1}$ be the vector denoting the initial and final indexes of a subsequence, that is $\widehat{P}_i = p_{s_{i-1}+1}, \ldots, p_{s_i}$. Then, s satisfies the following conditions : (i) $s_0 = 0$ and $s_K = m$, (ii) for $1 \leq i \leq K$, $s_i - s_{i-1} = r_{p,1}$, which implies that $s_i = \sum_{j=1}^{i} r_{p,1}$. Once we have partitioned curve P into K subsequences, we continue as in the w-anchored LSH by applying the basic LSH to each subsequence using the random shifts given by sequence t. For a query curve Q, the hash process is the same, but a different random sequence r_Q is used to partition the curve. We let $\mathcal{H}^{\mathsf{S}}_{w,\delta,\ell}$ denote the hash family consisting of all possible pairs of hash functions $\left(g^{t,r_P}_{w,\delta,\ell}, g^{t,r_Q}_{w,\delta,\ell}\right)$. The next theorem shows that the scheme has a bi-criteria approximation (note that the alignment approximation in point (ii) differs from the one for the anchored distance).

▶ **Theorem 13.** *Let $P, Q \in \mathcal{S}$ two curves with m_1 and m_2 points, respectively and let $m = \min\{m_1, m_2\}$. Let $\ell \geq 1$ be an arbitrary integer, $\delta = 4dr\ell$, and $c = 4d^{\frac{3}{2}}\ell$. Then, it holds that:*

(i) *if $d_{w,sF}(P,Q) < r$, then $Pr_{\mathcal{H}^s_{w,\delta,\ell}}\left(g^{t,r_P}_{w,\delta,\ell}(P) = g^{t,r_Q}_{w,\delta,\ell}(Q)\right) > \left(1/\sqrt{2}w\ell\right)^{2m/\ell}$;*

(ii) *if $d_{w\ell,sF}(P,Q) > cr$, then $Pr_{\mathcal{H}^s_{w,\delta,\ell}}\left(g^{t,r_P}_{w,\delta,\ell}(P) = g^{t,r_Q}_{w,\delta,\ell}(Q)\right) = 0$.*

7 Extensions to dynamic time warping

All our schemes can be applied to DTW without any algorithmic change, and in this section we analyze some of them. We first investigate in Section 7.1 the basic scheme in Section 3.1 for this distance. Then, we provide a few insights on DTW with constrained alignments in Section 7.2. We do not analyze the techniques proposed in Sections 4 and 5 since they have the same linear approximation of the basic LSH, and – in contrast to our previous results for the Fréchet distance – do not provide a sublinear approximation for DTW.

7.1 Analysis of the basic LSH

▶ **Lemma 14.** *Let $P, Q \in \Delta^d$ be two curves with m_1 and m_2 points, respectively. For any $\delta \geq 0$, it holds that*

$$Pr_{\mathcal{H}^L_\delta}\left(h^t_\delta(P) = h^t_\delta(Q)\right) \geq 1 - \left(d \cdot \frac{d_{DTW}(P,Q)}{\delta}\right).$$

▶ **Lemma 15.** *Let $P, Q \in \Delta^d$ be two curves with m_1 and m_2 points, respectively, and let $M = \max\{m_1, m_2\}$ and $\delta \geq 0$. If there exists a value of $t \in [0, \delta)^d$ such that $h^t_\delta(P) = h^t_\delta(Q)$, then it holds that $d_{DTW}(P,Q) \leq 2M\sqrt{d} \cdot \delta$.*

Lemma 14 above can be proven by a similar analysis as in the proof of Lemma 5: let T be an optimal traversal of P and Q with respect to their DTW distance; let $\ell = |T|$ and denote with d_k the distance $\|p_{i_k} - q_{j_k}\|$ for $1 \leq k \leq \ell$; we have that $d_{\mathrm{DTW}}(P,Q) = \sum_{1 \leq k \leq \ell} d_k$; now, we can use a union bound over all pairs in the traversal, instead of components. The proof of Lemma 15 is somewhat technical since DTW does not satisfy the triangle inequality. The following theorem can be obtained by plugging in the bounds of Lemmas 14 and 15.

▶ **Theorem 16.** *Let $P, Q \in \Delta^d$ be two curves with m_1 and m_2 points, respectively, and let $M = \max\{m_1, m_2\}$, $\delta = 2dr$ and let $c = 4d^{\frac{3}{2}}M$.*

(i) *if $d_{DTW}(P,Q) < r$, then $Pr_{\mathcal{H}^t_\delta}(h^t_\delta(P) = h^t_\delta(Q)) > \frac{1}{2}$;*

(ii) *if $d_{DTW}(P,Q) > cr$, then $Pr_{\mathcal{H}^t_\delta}(h^t_\delta(P) = h^t_\delta(Q)) = 0$.*

7.2 Handling constrained alignments

The schemes in Section 6 for w-anchored/speed traversals automatically apply to DTW distance, with the same collision probabilities stated in Theorems 12 and 13. However, the approximation factor is $4d^{3/2}(m_1 + m_2)$, where m_1 and m_2 are curve lengths. The claim follows by mimicking the proofs for the Fréchet distance and use the bounds in Theorem 16.

8 Conclusion

To the best of our knowledge, this is the first paper providing LSH schemes for curves. When applied to the near neighbor problem, our techniques improve the state of the art for the discrete Fréchet distance [17] under different settings, and provide the first data structure with theoretical guarantees for DTW. The methods presented are simple enough that they may be practical. We do not know if our bounds are tight. It would be interesting to know if lower bounds can be obtained for the studied problem and/or to improve the upper bounds. All of the presented LSH schemes exhibit the property that no collisions happen between far points (i.e., $\alpha_2 = 0$). An open question is to understand if it is possible to slightly increase this collision probability (say $\alpha_2 = 1/n$) to get a better approximation factor. Another interesting direction would be to reduce space by exploiting the independence in the approach described in Section 4.1, or by using a multiprobe approach [24]. Finally, we remark that our results only partially extend to DTW. As such, it is still open to get a sublinear approximation for DTW. We hope that our work inspires further work in one of these directions.

Acknowledgments. The authors would like to thank Rasmus Pagh and the anonymous reviewers for useful comments. This research was initiated at the Dagstuhl Seminar 16101 "Data Structures and Advanced Models of Computation on Big Data, 2016".

References

1 S. Arya, D. Mount, A. Vigneron, and J. Xia. Space-time tradeoffs for proximity searching in doubling spaces. In *Proc. 16th European Symp. Algorithms (ESA)*, pages 112–123, 2008.

2 A. Backurs and A. Sidiropoulos. Constant-distortion embeddings of Hausdorff metrics into constant-dimensional l_p spaces. In *Proc. 19th Workshop on Approximation Algorithms for Combinatorial Optimization Problems (APPROX)*, volume 60, pages 1:1–1:15, 2016.

3 Y. Bartal, L. A. Gottlieb, and O. Neiman. On the impossibility of dimension reduction for doubling subsets of ℓ_p. In *Proc. 13th Symp. on Computational Geometry (SOCG)*, pages 60:60–60:66, 2014.

4 K. Bringmann. Why Walking the Dog Takes Time: Fréchet Distance Has No Strongly Subquadratic Algorithms Unless SETH Fails. In *Proc. 55th Symp. on Foundations of Computer Science (FOCS)*, pages 661–670, 2014.

5 J. C. Brown and P. J. O. Miller. Automatic classification of killer whale vocalizations using dynamic time warping. *J. of the Acoustical Society of America*, 122(2):1201–1207, 2007.

6 J. Campbell, J. Tremblay, and C. Verbrugge. Clustering player paths. In *Proc. 10th Int'l Conf. on the Foundations of Digital Games (FDG)*, 2015.

7 M. de Berg, A. F. Cook, and J. Gudmundsson. Fast Fréchet queries. *Comput. Geom.*, 46(6):747–755, 2013.

8 A. Driemel and S. Har-Peled. Jaywalking your dog: Computing the Fréchet distance with shortcuts. *SIAM J. Computing*, 42(5):1830–1866, 2013.

9 A. Driemel, A. Krivošija, and C. Sohler. Clustering time series under the Fréchet distance. In *Proc. 27th Symp. on Discrete Algorithms (SODA)*, pages 766–785, 2016.

10 A. Driemel and F. Silvesstri. Locality-sensitive hashing of curves. Arxiv:1703.04040, 2017.

11 G. Forestier, F. Lalys, L. Riffaud, B. Trelhu, and P. Jannin. Classification of surgical processes using dynamic time warping. *J. Biomedical Informatics*, 45(2):255–264, 2012.

12 J. Gudmundsson and N. Valladares. A GPU approach to subtrajectory clustering using the Fréchet distance. *IEEE Trans. on Parallel and Distributed Systems*, 26(4):924–937, 2015.

13 Anupam Gupta, Robert Krauthgamer, and James R Lee. Bounded geometries, fractals, and low-distortion embeddings. In *Proc. 44th Symp. Found. Comp. Science (FOCS)*, pages 534–543, 2003.

14 S. Har-Peled, P. Indyk, and R. Motwani. Approximate nearest neighbor: Towards removing the curse of dimensionality. *Theory of Computing*, 8(1):321–350, 2012.

15 B. Huang and W. Kinsner. ECG frame classification using dynamic time warping. In *Proc. Canadian Conf. on Electrical and Computer Engineering*, volume 2, pages 1105–1110, 2002.

16 P. Indyk. On approximate nearest neighbors in non-euclidean spaces. In *Proc. 39th Symp. on Foundations of Computer Science*, pages 148–155, 1998.

17 P. Indyk. Approximate nearest neighbor algorithms for Fréchet distance via product metrics. In *Proc. 18th Symp. on Computational Geometry (SOCG)*, pages 102–106, 2002.

18 P. Indyk and J. Matoušek. Low-distortion embeddings of finite metric spaces. In *Handbook of Discrete and Computational Geometry*, pages 177–196. CRC Press, 2004.

19 P. Indyk and R. Motwani. Approximate nearest neighbors: Towards removing the curse of dimensionality. In *Proc. 30th Symp. Theory of Computing (STOC)*, pages 604–613, 1998.

20 R. J. Kenefic. Track clustering using Fréchet distance and minimum description length. *J. of Aerospace Information Systems*, 11(8):512–524, 2014.

21 E. Keogh and C. A. Ratanamahatana. Exact indexing of dynamic time warping. *Knowledge and Information Systems*, 7(3):358–386, 2005.

22 Z. M. Kovacs-Vajna. A fingerprint verification system based on triangular matching and dynamic time warping. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(11):1266–1276, 2000.

23 B. Legrand, C. S. Chang, S. H. Ong, S. Y. Neo, and N. Palanisamy. Chromosome classification using dynamic time warping. *Pattern Recognition Letters*, 29(3):215–222, 2008.

24 Q. Lv, W. Josephson, Z. Wang, M. Charikar, and K. Li. Multi-probe lsh: Efficient indexing for high-dimensional similarity search. In *Proc. 33rd Int'l Conf. on Very Large Data Bases*, VLDB'07, pages 950–961. VLDB Endowment, 2007.

25 J. Matoušek. Embedding finite metric spaces into euclidean spaces. In *Lectures on Discrete Geometry*, chapter 15. Springer, 2002.

26 T. Rakthanmanon, B. Campana, A. Mueen, G. Batista, B. Westover, Q. Zhu, J. Zakaria, and E. Keogh. Searching and mining trillions of time series subsequences under dynamic time warping. In *Proc. 18th Conf. Knowl. Disc. and Data Mining*, pages 262–270, 2012.

27 C. A. Ratanamahatana and E. Keogh. Three myths about dynamic time warping data mining. In *Proc. SIAM Conf. on Data Mining (SDM)*, pages 506–510, 2005.

28 G. Shakhnarovich, T. Darrell, and P. Indyk, editors. *Nearest-Neighbor Methods in Learning and Vision: Theory and Practice.* MIT Press, 2006.

29 A. Shrivastava and P. Li. Asymmetric LSH (ALSH) for sublinear time maximum inner product search (MIPS). In *Proc. 27th Conf. on Neural Information Processing Systems (NIPS)*, pages 2321–2329, 2014.

30 G. K. D. Vries. *Kernel methods for vessel trajectories.* PhD thesis, Univ. Amsterdam, 2012.

31 H. Zhu, J. Luo, H. Yin, X. Zhou, J. Z. Huang, and F. B. Zhan. Mining trajectory corridors using Fréchet distance and meshing grids. In *Proc. 14th Pacific-Asia Conference on Knowledge Discovery and Data Mining (PAKDD)*, pages 228–237, 2010.