

# Fréchet Distance for Curves, Revisited

Boris Aronov<sup>\*1</sup>, Sariel Har-Peled<sup>2</sup>, Christian Knauer<sup>3</sup>, Yusu Wang<sup>4</sup>, and Carola Wenk<sup>5</sup>

<sup>1</sup> Dept. of Comp. and Info. Sci., Polytechnic Univ., Brooklyn, NY 11201;  
aronov@cis.poly.edu

<sup>2</sup> Dept. of Comp. Sci., University of Illinois, 1304 West Springfield Ave., Urbana, IL 61801; sariel@cs.uiuc.edu

<sup>3</sup> Freie Universität Berlin, Inst. of Comp. Sci., Takustr. 9, 14195 Berlin, Germany;  
christian.knauer@inf.fu-berlin.de

<sup>4</sup> Dept. of Comp. Sci. and Engineering, The Ohio State Univ., Columbus, OH 43016;  
yusu@cse.ohio-state.edu

<sup>5</sup> Dept. of Comp. Sci., Univ. of Texas at San Antonio, One UTSA Circle,  
San Antonio, TX 78249; carola@cs.utsa.edu

**Abstract.** We revisit the problem of computing the Fréchet distance between polygonal curves, focusing on the *discrete* Fréchet distance, where only distance between vertices is considered. We develop efficient approximation algorithms for two natural classes of curves:  $\kappa$ -*bounded* curves and *backbone* curves, the latter of which are widely used to model molecular structures. We also propose a pseudo-output-sensitive algorithm for computing the discrete Fréchet distance exactly. The complexity of the algorithm is a function of the complexity of the free-space boundary, which is quadratic in the worst case, but tends to be lower in practice.

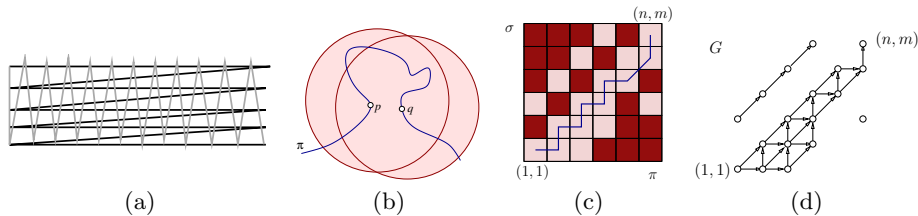
## 1 Introduction

The Fréchet distance is a natural measure of similarity between two curves [AG95]. An intuitive definition of the Fréchet distance is to imagine that a dog and its handler are walking on their respective curves. Both can control their speed but can only go forward. The Fréchet distance of these two curves is the minimal length of any leash necessary for the dog and the handler to move from the starting points of the two curves to their respective endpoints. The Fréchet distance and its variants have been widely used in many applications such as dynamic time-warping [KP99], speech recognition [KHM<sup>+</sup>98], signature verification [PP90], and matching of time series in databases [KKS05].

Alt and Godau [AG95] present an algorithm to compute the Fréchet distance between two polygonal curves of  $n$  and  $m$  vertices, respectively, in  $O(nm \log(nm))$  time. Improving this roughly quadratic-time solution for general curves seems to be hard, and so far, no algorithm, exact or approximate, with running time lower than  $O(nm)$  has been found for this problem for general curves. A slightly

---

\* Research supported in part by NSF ITR Grant CCR-00-81964 and by a grant from US-Israel Binational Science Foundation.



**Fig. 1.** (a) Light and dark curves are close under Hausdorff, but far under Fréchet distance. (b)  $\pi$  is  $\kappa$ -bounded if and only if for any  $p, q \in \pi$ , subchain  $\pi(p, q)$  lies inside the shaded region: the radius of the two disks (centered at  $p$  and  $q$ , respectively) is  $\kappa d(p, q)/2$ . (c) The free-space diagram  $D(\pi, \sigma, \delta)$  and a viable path. (d) The directed graph  $G$  corresponds to the white cells in (c).

simpler version of the Fréchet distance is the *discrete Fréchet distance*, which only considers vertices of polygonal curves. Its computation takes  $\Theta(n^2)$  time and space using dynamic programming [EM94], and no subquadratic algorithm is known either. Both the continuous and discrete Fréchet distance are related to the edit distance problem, for which no substantially subquadratic algorithm is known.

On the other hand, another similarity measure, the *Hausdorff distance*, can be computed faster in the plane and approximated efficiently in higher dimensions. Unfortunately, the Hausdorff distance does not reflect curve similarity well (see Figure 1(a) for an example). Alt et al. [AKW04] showed that the Hausdorff distance and the Fréchet distance are the same for a pair of closed convex curves. They also showed that the two measures are closely related for  $\kappa$ -bounded curves (see Figure 1(b) for definition). In particular, they showed that the Fréchet distance between any two  $\kappa$ -bounded curves is bounded by  $\kappa + 1$  times the Hausdorff distance between them. This leads to a  $(\kappa + 1)$ -approximation algorithm for the Fréchet distance for any pair of  $\kappa$ -bounded curves that runs in near linear time in  $\mathbb{R}^2$ . Little is known about computing the Fréchet distance for other types of curves, including even  $x$ -monotone curves.

The problem of minimizing the Fréchet distance under various classes of transformations has also been studied [AKW01, Wen02, CM05], however the runtimes are very high and practical solutions remain elusive. Fréchet distance has also been extended to graphs (maps) [AERW03, BPSW05], to piecewise smooth curves [Rot05], to simple polygons [BBW06], and to surfaces [AB05]. Finally, Fréchet distance was used for high-dimensional approximate nearest neighbor search [Ind02], for efficient curve simplification [AHPMW05], and for curve morphing [EGHPM01].

*Our results.* Given the apparent difficulty of improving the worst-case time complexity of computing the (continuous or discrete) Fréchet distance between two unrestricted polygonal curves, we aim at developing algorithms for more realistic cases. First, in Section 3 we consider efficient approximation algorithms for the *discrete Fréchet distance*. Most current algorithms for computing the Fréchet

distance rely on a so-called *decision* procedure which determines whether the Fréchet distance between the two given curves is larger or smaller than a given value. We observe that an approximate solution to the decision problem can lead to an approximation of the discrete Fréchet distance, and curve simplification can help us approximate the decision problem efficiently. We apply this idea to two common families of curves:  $\kappa$ -bounded curves and backbone curves. In the former case, given an arbitrary polygonal curve  $\pi$  and a  $\kappa$ -bounded polygonal curve  $\sigma$ , of complexity  $n$  and  $m$ , respectively, we can  $(1 + \varepsilon)$ -approximate their discrete Fréchet distance in  $O((m + n\kappa^d\varepsilon^{-d})\log(nm))$  time in  $d$  dimensions. In the second case, both curves are so-called *backbone curves*, used widely to model molecular structures, such as protein backbones and DNA/RNA. We  $(1 + \varepsilon)$ -approximate their discrete Fréchet distance in near linear time in two dimensions, and in  $O(nm^{1/3}\log(nm)/\varepsilon^2)$  time in three dimensions.

In Section 4, we shift our focus back to the exact computation of the discrete Fréchet distance. Previously, the problem of deciding whether the Fréchet distance was smaller than some threshold was cast as finding some viable path in the so-called *free-space diagram*. We observe that such a path can be computed using only a subset  $\mathcal{S}$  of cells in the free-space diagram. The size of  $\mathcal{S}$  is  $nm$  in the worst case, but should be smaller in practical settings. Based on this observation, we present algorithms that compute the discrete Fréchet distance under the  $L_\infty$  norm in  $O(|\mathcal{S}|\log(nm) + (n + m)\log^{2d}(nm))$  time in  $d$  dimensions. The case of the  $L_2$  norm can be handled as well but with worse performance; running time and details are omitted from this version.

## 2 Preliminaries

*Fréchet distance.* A (parameterized) curve in  $\mathbb{R}^d$  can be represented as a continuous function  $f: [0, 1] \rightarrow \mathbb{R}^d$ . A (monotone) reparametrization  $\alpha$  is a continuous non-decreasing function  $\alpha: [0, 1] \rightarrow [0, 1]$  with  $\alpha(0) = 0$  and  $\alpha(1) = 1$ . Given two curves  $f, g: [0, 1] \rightarrow \mathbb{R}^d$ , their *Fréchet distance*,  $\delta_F(f, g)$ , is defined as

$$\delta_F(f, g) := \inf_{\alpha, \beta} \max_{t \in [0, 1]} d(f(\alpha(t)), g(\beta(t))).$$

where  $d(x, y)$  denotes the Euclidean distance between points  $x$  and  $y$ , and  $\alpha$  and  $\beta$  range over all monotone reparametrizations.

*Discrete Fréchet distance.* A simpler variant of the Fréchet distance for two polygonal curves  $\pi = \langle p_1, p_2, \dots, p_n \rangle$  and  $\sigma = \langle q_1, q_2, \dots, q_m \rangle$  is the *discrete Fréchet distance*, denoted by  $\delta_D(\pi, \sigma)$ . Imagine that both the dog and its handler can only stop at vertices of  $\pi$  and  $\sigma$ , and at any step, each of them can either stay at their current vertex or jump to the next one. The discrete Fréchet distance is defined as the minimal leash necessary at these discrete moments.

To formally define the discrete Fréchet distance, we first consider a discrete analog of  $(\alpha, \beta)$ , i.e, the correspondences between continuous reparametrizations. In particular, an *order-preserving complete correspondence* between  $\pi$  and  $\sigma$  is a

set  $M \subseteq \{(p, q) \mid p \in \pi, q \in \sigma\}$  of pairs of vertices which is (a) *order-preserving*: if  $(p_i, q_j) \in M$ , then no  $(p_s, q_t) \in M$  for  $s < i$  and  $t > j$ , nor for  $s > i$  and  $t < j$ ; and (b) *complete*: for any  $p \in \pi$  (respectively,  $q \in \sigma$ ), there exists some pair involving  $p$  (respectively,  $q$ ) in  $M$ . The *discrete Fréchet distance* between  $\pi$  and  $\sigma$ ,  $\delta_D(\pi, \sigma)$ , is then

$$\delta_D(f, g) := \min_M \max_{(p, q) \in M} d(p, q),$$

where  $M$  range over all order-preserving complete correspondences between  $\pi$  and  $\sigma$ .

It is well known that discrete and continuous versions of the Fréchet distance relate to each other as follows:

$$\delta_F(\pi, \sigma) \leq \delta_D(\pi, \sigma) \leq \delta_F(\pi, \sigma) + \max\{\ell_1, \ell_2\},$$

where  $\ell_1$  and  $\ell_2$  are the lengths of the longest edges in  $\pi$  and  $\sigma$ , respectively. This suggests using  $\delta_D$  to approximate  $\delta_F$ . Unfortunately, it seems that computing  $\delta_D(\pi, \sigma)$  is asymptotically almost as hard as computing  $\delta_F(\pi, \sigma)$ .

*Decision problem.* In the original paper [AG95], to compute  $\delta_F(\pi, \sigma)$ , first, the *decision problem* “Given a parameter  $\delta \geq 0$ , is  $\delta_F(\pi, \sigma) \leq \delta$ ?” is solved by a dynamic programming algorithm that runs in  $\Theta(nm)$  time and space. This algorithm is then used as a subroutine to search for  $\delta_F(\pi, \sigma)$  using the parametric search paradigm within  $O(nm \log nm)$  time [AG95, AST94]. Our algorithms follow a similar framework combining decision problem and binary search (instead of parametric search). Thus we describe below how to solve the decision problem for the discrete case.

Given two polygonal chains  $\pi$  and  $\sigma$  and a distance threshold  $\delta \geq 0$ , we construct the following *free-space diagram*  $D = D(\pi, \sigma, \delta)$ :  $D$  is an  $n \times m$  matrix (grid) and a cell  $D[i, j]$  has value 1 if  $d(p_i, q_j) \leq \delta$ , and value 0 otherwise. We refer to 1-cells as *white* and 0-cells as *black*. A *viable path* in  $D$  is a path connecting  $(1, 1)$  to  $(n, m)$ , visiting only white cells of  $D$ , and moving in one step from  $(i, j)$  to either  $(i, j + 1)$ ,  $(i + 1, j)$ , or  $(i + 1, j + 1)$ . It is easy to check that a complete order-preserving correspondence  $M$  induces a viable path in  $D$  and vice versa (see Figure 1 (c)). Hence the problem of deciding “ $\delta_D(\pi, \sigma) \leq \delta$ ?” is equivalent to deciding the existence of a viable path in  $D$ .

Given  $D$ , one can extract a viable path, if it exists, in  $\Theta(nm)$  time using dynamic programming. Alternatively, one can traverse a directed graph  $G$  defined as follows: The nodes of  $G$  are the white cells of  $D$ . A white cell is connected to its top, right, and/or top-right neighbor cells by a directed edge, if they are white. See Figure 1(d). The size of  $G$  is  $O(|W|)$ , where  $|W|$  is the number of white cells of  $D$ . Given  $G$ , testing “ $\delta_D(\pi, \sigma) \leq \delta$ ?” corresponds to a connectivity check in  $G$  (from  $(1, 1)$  to  $(n, m)$ ) that can be performed in time  $O(|W|)$ .

*Approximations.* We say that  $\tau$  is an  $(1 + \varepsilon)$ -approximation of  $\delta(\pi, \sigma)$  if

$$(1 - \varepsilon)\tau \leq \delta(\pi, \sigma) \leq (1 + \varepsilon)\tau.$$

An algorithm  $(1 + \varepsilon)$ -approximates the decision problem “Is  $\delta(\pi, \sigma) \leq \tau$ ?”, if it returns YES whenever  $\delta(\pi, \sigma) < (1 - \varepsilon)\tau$  and NO whenever  $\delta(\pi, \sigma) > (1 + \varepsilon)\tau$ . If  $\tau$  is a  $(1 + \varepsilon)$ -approximation of  $\delta(\pi, \sigma)$ , the algorithm is allowed to return either YES or NO. We also call such an algorithm a *fuzzy decision procedure*.

### 3 Approximation Algorithms Based on Simplification

In this section, we first introduce a general framework for approximating the discrete Fréchet distance by using a fuzzy decision procedure. Based on this framework we then develop efficient approximation algorithms, using curve simplification and packing arguments, for two families of common curves:  $\kappa$ -bounded curves and backbone curves.

#### 3.1 Approximation via a fuzzy decision procedure

Given a set  $P$  of  $N$  points in  $\mathbb{R}^d$ , a *well-separated pairs decomposition (WSPD)* of  $P$  with separation constant  $s$  is a collection  $\{(A_i, B_i)\}$  of pairs of subsets of  $P$ , with the property that (1) for every pair of points  $x, y \in P$ , there is an index  $i$ , so that  $x \in A_i$  and  $y \in B_i$  and (2) the minimum distance between  $A_i$  and  $B_i$  is at least  $s$  times the diameter of either set. The *size* of a WSPD is  $\sum_i (|A_i| + |B_i|)$ . For a constant  $s$ , a WSPD of size  $O(N)$  can be computed in  $O(N \log N)$  time [CK95]. For every pair  $(A_i, B_i)$  in the WSPD, we choose an arbitrary pair of points  $(p_i, q_i)$ , with  $p_i \in A_i$  and  $q_i \in B_i$ , as its *representatives*. We set  $s = 10$ . It is easy to check that the distance between any two points  $x, y \in P$  is  $(1 + \frac{1}{5})$ -approximated by the distance between the representatives of the corresponding WSPD pair.

Consider an optimization problem whose solution  $\delta^*$  is a distance determined by a pair of points in  $P$ . Let  $X$  be the set of all distances induced by pairs of points of  $P$ . We now describe how to solve the optimization problem approximately by using an exact decision procedure. We start by constructing a WSPD as above and considering the set  $Y := \{d(p_i, q_i)\}$  of  $O(N)$  distances between representative points of the decomposition pairs. By definition of WSPD, every distance in  $X$  is  $(1 + \frac{1}{5})$ -approximated by some distance in  $Y$ . Hence some value in  $Y$  is a  $(1 + \frac{1}{5})$ -approximation of  $\delta^*$ , as  $\delta^*$  is defined by a pair of points in  $P$ . Next, form a larger set  $Y'$  of distances by adding to  $Y$ , for each value  $y \in Y$ , the two values  $\frac{4}{5}y$  and  $\frac{6}{5}y$ . We then perform a binary search on  $Y'$  using the decision procedure to identify the smallest interval  $\mathcal{J} = [a, b]$  that contains  $\delta^*$ . (The cost is dominated by  $O(\log N)$  invocations of the decision procedure and the  $O(N \log N)$  time to construct the WSPD.) Notice that  $b \leq \frac{3}{2}a$ , as by above discussion,  $\delta^*$  is contained in the interval  $[\frac{4}{5}y, \frac{6}{5}y]$  for some  $y \in Y$  and we have included both  $\frac{4}{5}y$  and  $\frac{6}{5}y$  in  $Y'$ . We now perform another (numerical) binary search on this interval to identify the interval  $[a', b'] \subseteq [a, b]$  containing  $\delta^*$  with  $b' \leq (1 + \varepsilon)a'$ , giving rise to a  $(1 + \varepsilon)$ -approximation of  $\delta^*$ . The second binary search invokes the decision procedure  $O(\log \varepsilon^{-1})$  times.

Interestingly, the decision procedure does not have to be exact—the above binary search can be adapted to work with a fuzzy decision procedure with the same performance guarantees. We omit the details from current version.

**Theorem 1.** *Let  $P$  be a set of  $N$  points in  $\mathbb{R}^d$ , and let  $Z$  be any optimization problem, for which the optimal answer is a distance induced by a pair of points of  $P$ . Given a fuzzy decision procedure for  $Z$ , one can  $(1 + \varepsilon)$ -approximate the optimal solution in*

$$O(N \log N + T_{\text{FDECISION}}(N, 1/10) \log N + T_{\text{FDECISION}}(N, \varepsilon/4) \log \varepsilon^{-1})$$

*time, where  $T_{\text{FDECISION}}(N, c)$  is the running time of the fuzzy decision procedure on  $N$  points when the required approximation factor is  $1 + c$ .*

Returning to the computation of discrete Fréchet distance, observe that there must exist some  $p^* \in \pi$  and  $q^* \in \sigma$  such that  $d(p^*, q^*) = \delta_{\text{D}}(\pi, \sigma)$ . Hence, by applying the above theorem to the set of all vertices from  $\pi$  and  $\sigma$ , we only need a fuzzy decision procedure for  $\delta_{\text{D}}(\pi, \sigma)$  in order to approximate  $\delta_{\text{D}}(\pi, \sigma)$ .

### 3.2 Approximation with simplifications

The remaining question is how to implement a fuzzy decision procedure efficiently. We show that curve simplification together with a packing argument can be used to achieve guaranteed efficiency for the two classes of common curves that we investigate.

*Greedy simplification.* Given a polygonal chain  $\pi = \langle p_1, \dots, p_n \rangle$ , we simplify  $\pi$  to obtain  $\tilde{\pi} = \langle \hat{p}_1, \dots, \hat{p}_k \rangle$ , where vertices of  $\tilde{\pi}$  form a subsequence of  $\pi$ , with  $\hat{p}_1 = p_1$  and  $\hat{p}_k = p_n$ . Let  $I_{\pi}(i) = j$  if  $\hat{p}_i = p_j \in \pi$ ; the subscript  $\pi$  is omitted when it is clear from context. We say that  $\tilde{\pi}$   $\mu$ -simplifies  $\pi$  if (i)  $I(i) < I(k)$  for  $i < k$ , and (ii)  $d(\hat{p}_i, p_k) \leq \mu$  for any  $k$  such that  $I(i) \leq k < I(i+1)$ . (This definition is slightly different from the standard one in the literature.) We construct  $\tilde{\pi}$ , a  $\mu$ -simplification of  $\pi$ , in a greedy manner: Start with  $\hat{p}_1 = p_1$ . At some stage, suppose we have already computed  $\hat{p}_i = p_j$ . In order to find  $I(i+1)$ , we check each vertex of  $\pi$  starting from  $p_j$  in order, and stop when we reach the first edge  $p_k p_{k+1}$  of  $\pi$  such that  $d(p_j, p_k) \leq \mu$  and  $d(p_j, p_{k+1}) > \mu$ . We set  $\hat{p}_{i+1} = p_{k+1}$  and continue, until we reach  $p_n$ , at which point we add  $p_n$  as the final vertex of  $\tilde{\pi}$ . The entire procedure takes linear time. By construction, the following observation is straightforward.

**Observation 2** *Any edge  $\hat{p}_i \hat{p}_{i+1}$  in  $\tilde{\pi}$  other than the last edge,  $d(\hat{p}_i, \hat{p}_{i+1}) > \mu$ .*

The following fact follows easily by an explicit construction. We omit the details.

**Lemma 1.** *If  $\tilde{\pi}$  and  $\tilde{\sigma}$  be  $\mu$ -simplifications of curves  $\pi$  and  $\sigma$ , respectively, then*

$$\delta_{\text{D}}(\pi, \sigma) - 2\mu \leq \delta_{\text{D}}(\tilde{\pi}, \tilde{\sigma}) \leq \delta_{\text{D}}(\pi, \sigma) + \mu.$$

The above lemma implies that if the answer to  $\delta_D(\tilde{\pi}, \tilde{\sigma}) \leq \delta$  is YES, then,  $\delta_D(\pi, \sigma) \leq \delta + 2\mu$ . If it is NO, then  $\delta_D(\pi, \sigma) \geq \delta - \mu$ . Thus the decision problem for  $\delta_D(\tilde{\pi}, \tilde{\sigma})$   $(1 + 2\mu/\delta)$ -approximates that for  $\delta_D(\pi, \sigma)$ . We next show that  $\delta_D(\tilde{\pi}, \tilde{\sigma})$  can be answered asymptotically much faster for two special classes of curves, giving rise to efficient fuzzy decision procedure for them.

### 3.3 Fréchet distance for $\kappa$ -bounded curves

As defined by Alt et al. [AKW04],  $\pi$  is  $\kappa$ -bounded if  $\pi(x, y) \subseteq B(x, \frac{\kappa}{2}d(x, y)) \cup B(y, \frac{\kappa}{2}d(x, y))$ , for all  $x, y \in \pi$ , where  $\pi(x, y)$  is the arc of  $\pi$  between  $x$  and  $y$  and  $B(x, r)$  is the radius- $r$  Euclidean ball centered at  $x$ <sup>6</sup>. See Figure 1(b) for an illustration in two dimensions. Examples of  $\kappa$ -bounded curves include  $\kappa$ -straight curves [AKW04] which in turn include *curves with increasing chords* [Rot94] and *self-approaching curves* [AAI<sup>+</sup>01].

We now describe how to construct a fuzzy decision procedure for the problem “ $\delta_D(\pi, \sigma) \leq \delta$ ?” for two polygonal curves  $\pi, \sigma$  where  $\sigma$  is  $\kappa$ -bounded. We first  $\mu$ -simplify  $\pi$  and  $\sigma$  into  $\tilde{\pi}$  and  $\tilde{\sigma}$  respectively, using  $\mu := \varepsilon\delta/2$ . By Lemma 1, the decision problem for  $\delta_D(\tilde{\pi}, \tilde{\sigma})$  is an  $(1 + \varepsilon)$ -approximation decision procedure for  $\delta_D(\pi, \sigma)$ . Hence we now focus on checking whether  $\delta_D(\tilde{\pi}, \tilde{\sigma}) \leq \delta$ . Let  $n, m, r, s$  be the size of  $\pi, \sigma, \tilde{\pi}$ , and  $\tilde{\sigma}$  respectively;  $r \leq n$  and  $s \leq m$ .

*Decision problem for  $\delta_D(\tilde{\pi}, \tilde{\sigma})$ .* Let  $\tilde{D}$  be the free-space diagram for  $\tilde{\pi}$  and  $\tilde{\sigma}$  with respect to  $\delta$ . Recall that  $\delta_D(\tilde{\pi}, \tilde{\sigma}) \leq \delta$  if and only if there exists a viable path in  $\tilde{D}$ . This can be tested in  $O(|W|)$  time once  $W$ , the set of white cells of  $\tilde{D}$ , is given. We first bound the size of  $W$ .

For every  $\hat{p} \in \tilde{\pi}$ , let  $N(\hat{p})$  be the set of vertices from  $\tilde{\sigma}$  contained in  $B(\hat{p}, \delta)$ . Obviously,  $|W| = \sum_{\hat{p} \in \tilde{\pi}} |N(\hat{p})|$ . Consider any two points  $q_1, q_2 \in \tilde{\sigma}$  that lie in  $B(\hat{p}, \delta)$  for some  $\hat{p} \in \tilde{\pi}$ . If  $q_1q_2$  is an edge of  $\tilde{\sigma}$ ,  $d(q_1, q_2) \geq \mu$  by Observation 2. Otherwise

$$\sigma(q_1, q_2) \subseteq B(q_1, \frac{\kappa}{2}d(q_1, q_2)) \cup B(q_2, \frac{\kappa}{2}d(q_1, q_2)),$$

as  $\sigma$  is  $\kappa$ -bounded. Furthermore, in this case, let  $q_1q \subset \tilde{\sigma}$  be the edge with  $q \in \sigma(q_1, q_2)$ ;  $d(q_1, q) \geq \mu$  by Observation 2. It then follows that  $(1 + \kappa/2)d(q_1, q_2) \geq \mu$  and therefore  $d(q_1, q_2) \geq 2\mu/(\kappa + 2)$ . Hence  $N(\hat{p}) = O((\kappa\delta/\mu)^d)$  by a straightforward packing argument. This means that the number of white cells is  $|W| = O(s(\kappa\delta/\mu)^d) = O(n(\kappa\delta/\mu)^d)$  given that  $\sigma$  is a  $\kappa$ -bounded curve.

We still need to compute  $N(\hat{p})$  efficiently, that is, to enumerate the set of vertices of  $\tilde{\sigma}$  contained in  $B(\hat{p}, \delta)$  for every  $\hat{p} \in \tilde{\pi}$ . This can be done by a spherical range query, which unfortunately, there is no known efficient algorithms. We hence replace exact spherical range queries by approximate ones by rounding vertices of  $\tilde{\sigma}$  to vertices of some grid of appropriate size. Overall, the set of white cells in  $\tilde{D}$  can be computed in  $O(r + s + r(\kappa\delta/\mu)^d)$  time. Details are omitted from the conference version.

<sup>6</sup> We have slightly abused the notation by treating a curve section as a point set.

Putting everything together and substituting  $\mu = \varepsilon\delta/2$ , we have a  $(1 + \varepsilon)$ -approximation decision procedure for  $\delta_{\mathbb{D}}(\pi, \sigma)$  that runs in  $O(n + m + n\kappa^d\varepsilon^{-d})$  time and space in  $\mathbb{R}^d$ . An application of Theorem 1 now yields

**Theorem 3.** *A  $(1 + \varepsilon)$ -approximation of  $\delta_{\mathbb{D}}(\pi, \sigma)$  for a polygonal curve  $\pi$  and a  $\kappa$ -bounded curve  $\sigma$ , of size  $n$  and  $m$  respectively, can be computed in  $O((m + n\kappa^d\varepsilon^{-d}) \log(n/\varepsilon))$  time and  $O(n + m + n\kappa^d/\varepsilon^d)$  space in  $d$  dimensions.*

### 3.4 Fréchet distance for protein backbones

In molecular biology, it is common to model a protein backbone by a polygonal chain, where each  $C_{\alpha}$  atom becomes a vertex, and each edge represents a covalent bond between two consecutive amino acids. All the bonds have approximately the same bond length, and no two atoms (thus vertices) can get too close to each other due to van der Waals interactions. This is the motivation behind the study of *backbone* curves, which have the following properties:

- P1. For any two non-consecutive vertices  $u$  and  $v$  of the curve,  $d(u, v) \geq 1$ ,
- P2. Every edge of the curve has length between  $c_1$  and  $c_2$ , where  $c_2 > c_1 > 0$  are constants.

Although proteins lie in three-dimensional space, there are simplified models for protein backbones in both two and three dimensions [GIP99, KS94].

Given backbone curves  $\pi$  and  $\sigma$  in  $\mathbb{R}^d$  and given a distance threshold  $\delta \geq 0$ , we want to test whether  $\delta_{\mathbb{D}}(\pi, \sigma) \leq \delta$ . We  $\mu$ -simplify  $\pi$  and  $\sigma$  to obtain  $\tilde{\pi}$  and  $\tilde{\sigma}$  as in the previous section, for  $\mu = \varepsilon\delta/2$ , and construct the free-space diagram  $\tilde{D}$  for  $\tilde{\pi}$  and  $\tilde{\sigma}$  with respect to  $\delta$ .  $\tilde{D}$  is an  $r \times s$  grid, where by Observation 2 and property P2,  $r = |\tilde{\pi}| \leq c_2n/\mu$  and  $s = |\tilde{\sigma}| \leq c_2m/\mu$ . Given  $\tilde{D}$ , we can compute  $W$ , the set of white cells in  $\tilde{D}$  by the same approach as the one for  $\kappa$ -bounded curves in  $O(r + s + |W|)$  time and space. Once  $\tilde{D}$  and  $W$  are given, the decision problem can be solved in time proportional to  $|W|$ . Below we present an upper bound for  $|W|$ .

*Bounding  $|W|$ .* A straightforward bound<sup>7</sup> for  $|W|$  is  $O(\min\{r\delta^d, s\delta^d\})$ , as by a packing argument and property P1, there are at most  $O(\delta^d)$  vertices lying in  $\delta$ -neighborhood of any vertex of  $\tilde{\pi}$  and  $\tilde{\sigma}$ . If  $\delta < 1$ , then the number of white cells is  $O(n + m)$ . Hence we now assume that  $\delta \geq 1$ .

We can improve this bound on  $|W|$  by a more careful counting analysis. Assume without loss of generality that  $r \leq s$ . For any vertex  $\hat{p} \in \tilde{\pi}$  and its  $\delta$ -neighborhood  $B(\hat{p}, \delta)$ , let  $E(\hat{p})$  be the set of edges of  $\tilde{\sigma}$  intersecting the ball  $B(\hat{p}, \delta)$ . The number of vertices of  $\tilde{\sigma}$  in  $B(\hat{p}, \delta)$  is upper bounded by  $O(|E(\hat{p})|)$ . Furthermore, given any edge  $e = (\hat{q}_i, \hat{q}_{i+1}) \in \tilde{\sigma}$ , let  $\sigma(e) = \sigma(q_{I_q(i)}, q_{I_q(i+1)})$  (that is, the subchain  $\sigma(e) \subseteq \sigma$  that simplified into edge  $e$  in chain  $\tilde{\sigma}$ ).  $E(\hat{p})$

<sup>7</sup> In what follows, the big- $O$  notation may hide factors depending on constants  $c_1$  and  $c_2$ .

can be partitioned into two sets: (i)  $E_1 = \{e \in E(\hat{p}) \mid \sigma(e) \subseteq B(\hat{p}, \delta)\}$ , and (ii)  $E_2 = \{e \in E(\hat{p}) \mid \text{at least one vertex of } \sigma(e) \text{ lies outside } B(\hat{p}, \delta)\}$ .

By property P2, the number of vertices in  $\sigma(e)$  is at least  $\mu/c_2$  for any  $e \in \tilde{\sigma}$ . Therefore  $|E_1| = O(c_2\delta^d/\mu)$ . On the other hand, for every edge  $e \in E_2$ , there is at least one vertex of  $\sigma(e)$  that lies in the spherical shell of  $B(\hat{p}, \delta + c_2) \setminus B(\hat{p}, \delta)$ , as the length of edges in  $\sigma$  is at most  $c_2$ . Since the volume of this spherical shell is  $O(c_2(c_2 + \delta)^{d-1})$ , the size of  $E_2$  is bounded by  $O((c_2(c_2 + \delta)^{d-1}/(c_1^{d-1})))$ . Therefore,  $|E(\hat{p})| = |E_1| + |E_2| = O(\delta^{d-1} + \delta^d/\mu)$ . Summing over all  $r$  vertices of  $\tilde{\pi}$ , we obtain  $|W| = O(\frac{n}{\mu}(\delta^{d-1} + \delta^d/\mu))$ . As this number cannot exceed the size of  $\tilde{D}$  which is  $O(rs) = O(nm/\mu^2)$ , we have  $|W| = \min\{nm/\mu^2, O(\frac{n}{\mu}(\delta^{d-1} + \delta^d/\mu))\}$ .  $|W|$  is maximized when the two balancing terms are equal:  $\frac{nm}{\varepsilon^2\delta^2} = \frac{\delta^{d-2}}{\varepsilon^2}$ , that is, when  $\delta = m^{1/d}$ . This implies that  $|W| = O(nm^{1-2/d}/\varepsilon^2)$ .

Finally, by applying Theorem 1 and putting everything together, we conclude with the following result. We remark that similar but more involved argument can also be used to approximate the continuous Fréchet distance for two backbone curves. We omit the details from current conference version.

**Theorem 4.** *Given two backbone curves  $\pi$  and  $\sigma$  of  $n$  and  $m \geq n$  vertices respectively, we can compute a  $(1 + \varepsilon)$ -approximation of  $\delta_D(\pi, \sigma)$  in time  $O((n + m)\varepsilon^{-2} \log(nm))$  in the plane, and  $O(nm^{1/3}\varepsilon^{-2} \log(nm))$  in three dimensions.*

## 4 Pseudo-Output-Sensitive Algorithm

In this section, we present a pseudo-output-sensitive algorithm for computing  $\delta_D(\pi, \sigma)$  for general curves  $\pi$  and  $\sigma$  of size  $n$  and  $m$ , respectively. Although in the worst-case the runtime may still be  $\Theta(nm)$  for solving the decision problem, we believe that our observation should help produce efficient algorithms for the Fréchet distance in practice. In what follows, we describe results for the  $L_\infty$  norm (which yield a constant-factor approximation for the  $L_2$  norm). The case of the  $L_2$  norm is more involved and the running times are slower; details are omitted from the conference version.

*Binary search.* We show how to compute  $\delta^* = \delta_D(\pi, \sigma)$  using a variant of binary search. Assume we have algorithms to solve the decision problem “Is  $\delta^* = \delta_D(\pi, \sigma) \leq \delta?$ ” in time  $A(n + m)$  and to answer the *distance selection* query “Given a set of  $N$  points  $P$  and a rank  $k$ , what is the  $k$ th smallest distance among all pairwise distances from  $P$ ?” in  $B(N)$  time. Combining the two algorithms by performing a binary search on the set of all inter-vertex distances, we can find  $\delta^*$  in  $O((A(n + m) + B(n + m)) \log(nm))$  time. For the  $L_\infty$  norm, the distance-selection problem can be solved in  $O(dN \log^{d-1} N)$  in  $\mathbb{R}^d$  [Sal89]. For the decision problem, a straightforward bound for the runtime of  $A$  is  $O(|W|)$  plus the time to compute  $W$ , the set of white cells in the free-space diagram  $D = D(\pi, \sigma, \delta)$  for a threshold  $\delta > 0$ . Below we provide a tighter bound for  $A$  although its worst-case complexity is still  $\Theta(nm)$ .

*Boundary cells.* Given an  $n \times m$  matrix  $D$  representing the free-space diagram with respect to some threshold  $\delta$ , a *boundary cell* is a white cell whose immediate neighbor above or below it is black. So if  $D[i, j]$  is a boundary cell, then the edge  $q_j q_{j+1} \subset \sigma$  (or  $q_j q_{j-1}$ ) intersects the boundary of  $B(p_i, \delta)$  exactly once (one endpoint must lie inside and one must be outside); we say that the edge  $q_j q_{j+1}$  *crosses* the boundary of  $B(p_i, \delta)$ . Let  $\mathcal{S} = \mathcal{S}(\pi, \sigma, \delta)$  denote the set of boundary cells of  $D(\pi, \sigma, \delta)$ . Although in the worst case  $|\mathcal{S}| = \Omega(|W|) = \Omega(nm)$ , we expect it to be much smaller than  $|W|$  in practice. For example, consider the case when vertices of  $\sigma$  form lines of a cubic lattice of size  $n^{1/3} \times n^{1/3} \times n^{1/3}$  and  $\delta$  is roughly  $n^{1/3}/2$ . For a vertex  $p$  at the center of this cube, the number of white cells in the corresponding column in  $D$  is  $\Theta(n)$ , while the number of boundary cells is  $\Theta(n^{2/3})$ . The remaining questions are (i) how to compute the set of boundary cells  $\mathcal{S}(\pi, \sigma, \delta)$  and (ii) how to solve the decision problem once  $\mathcal{S}$  is given.

*Computing  $\mathcal{S}$ .* Given  $p \in \mathbb{R}^d$  and  $\delta$ , let  $\mathcal{S}(p, \delta)$  denote the set of edges from  $\sigma$  crossing the boundary of  $B(p, \delta)$ . Since one endpoint of each edge has to be inside  $B(p, \delta)$  and the other endpoint outside, this is different from the standard segment/ball intersection problem. To compute  $\mathcal{S}$ , we need to perform  $n$  *edge/ball crossing queries*, one for each vertex from  $\pi$ . Under the  $L_\infty$  norm,  $B(p, \delta)$  is a cube centered at  $p$ , and the basic operation is an edge/cube crossing query, where all cubes are congruent. We can preprocess the set of edges in  $\sigma$  by building a standard multi-level data structure for their endpoints (similar to the multi-level range tree for orthogonal range reporting problem). In particular, there are altogether  $2d$  levels: the first  $d$  levels are used to locate edges with one endpoint inside the query cube, and the second  $d$  levels are used to find those with the second endpoint outside of the query cube. The entire data structure has size  $O(m \log^{2d-1} m)$  and can be built in  $O(m \log^{2d} m)$  time. Given any query cube (in fact, the query can be any orthogonal box), the set of edges crossing it can be reported in  $O(\log^{2d} m + k)$  time, where  $k$  is the number of such edges. The query time can be improved to  $O(\log^{2d-1} m + k)$  using the fractional-cascading technique [CG86, Lue78].

Once  $\mathcal{S}$  is given, we can solve the decision problem using dynamic programming in  $O(|\mathcal{S}|)$  time and space. The technical details are omitted due to lack of space. Putting everything together, we conclude with the following theorem:

**Theorem 5.** *Given any two polygonal curves  $\pi$  and  $\sigma$  in  $\mathbb{R}^d$ , with  $n$  and  $m$  vertices, respectively, one can compute  $\delta_D(\pi, \sigma)$  under the  $L_\infty$ -norm in  $O(\Phi \log(nm) + (n+m) \log^{2d}(nm))$  time and  $O(\Phi + (n+m) \log^{2d-1}(nm))$  space, where  $\Phi$  is the maximum number of boundary cells for any threshold  $\delta$ .*

## 5 Conclusion and Discussion

In this paper, we considered the problem of computing the discrete Fréchet distance between two polygonal curves either approximately or exactly. Our main contribution is a simple approximation framework that leads to efficient  $(1 + \varepsilon)$ -approximation algorithms for two families of common curves:  $\kappa$ -bounded

curves and backbone curves. We also considered the exact algorithm for general curves, and proposed a pseudo-output-sensitive algorithm by observing that only a subset of the white cells from the free-space diagram are necessary for the decision problem. It will be interesting to investigate whether there are families of curves that are guaranteed to have small  $\Phi$ , which is the maximum number of boundary cells, over all possible values of the threshold  $\delta$ . We are currently working on extending the pseudo-output-sensitive algorithms to the continuous weak Fréchet distance.

It might be hard to develop algorithms that are significantly sub-quadratic in the worst case, given that no such algorithm exists for the related and widely studied problem of computing the edit distance for strings. Hence our future directions will focus on practical variants of the Fréchet distance that can handle outliers, partial matching, and/or efficient multiple-curve alignment. Another important direction is to develop efficient (approximation) algorithms for minimizing the Fréchet distance under transformations such as rigid motions.

## References

- [AAI<sup>+</sup>01] O. Aichholzer, F. Aurenhammer, C. Icking, R. Klein, E. Langetepe, and G. Rote. Generalized self-approaching curves. *Discrete Applied Mathematics*, 109:3–24, 2001.
- [AB05] H. Alt and M. Buchin. Semi-computability of the Fréchet distance between surfaces. In *Proc. 21th European Workshop on Computational Geometry*, 2005.
- [AERW03] H. Alt, A. Efrat, G. Rote, and C. Wenk. Matching planar maps. *J. Algorithms*, 49:262–283, 2003.
- [AG95] H. Alt and M. Godau. Computing the Fréchet distance between two polygonal curves. *Internat. J. Comput. Geom. Appl.*, 5:75–91, 1995.
- [AHPMW05] P. K. Agarwal, S. Har-Peled, N. Mustafa, and Y. Wang. Near-linear time approximation algorithms for curve simplification in two and three dimensions. *Algorithmica*, 2005. To appear.
- [AKW01] H. Alt, C. Knauer, and C. Wenk. Matching polygonal curves with respect to the Fréchet distance. In *Proceedings 18th International Symposium on Theoretical Aspects of Computer Science*, pages 63–74, 2001.
- [AKW04] H. Alt, C. Knauer, and C. Wenk. Comparison of distance measures for planar curves. *Algorithmica*, 38(1):45–58, 2004.
- [AST94] P. K. Agarwal, M. Sharir, and S. Toledo. Applications of parametric searching in geometric optimization. *J. Algorithms*, 17:292–318, 1994.
- [BBW06] K. Buchin, M. Buchin, and C. Wenk. Computing the Fréchet distance between simple polygons in polynomial time. In *Proc. 22st Annu. ACM Sympos. Comput. Geom.*, pages 80–87, 2006.
- [BPSW05] S. Brakatsoulas, D. Pfoser, R. Salas, and C. Wenk. On map-matching vehicle tracking data. In *Proc. 31st VLDB Conference*, pages 853–864, 2005.
- [CG86] B. Chazelle and L. J. Guibas. Fractional cascading: I. A data structuring technique. *Algorithmica*, 1:133–162, 1986.
- [CK95] Paul B. Callahan and S. Rao Kosaraju. A decomposition of multidimensional point sets with applications to k-nearest-neighbors and n-body potential fields. *J. ACM*, 42(1):67–90, 1995.

- [CM05] M. Clausen and A. Mosig. Approximately matching polygonal curves with respect to the Fréchet distance. *Comput. Geom. Theory Appl.*, 30:113–127, 2005.
- [EGHPM01] A. Efrat, L.J. Guibas, S. Har-Peled, and T.M. Murali. Morphing between polylines. In *Proc. 12th ACM-SIAM Sympos. Discrete Algorithms*, pages 680–689, 2001.
- [EM94] T. Eiter and H. Mannila. Computing discrete Fréchet distance. Technical Report CD-TR 94/64, Christian Doppler Laboratory for Expert Systems, TU Vienna, Austria, 1994.
- [GIP99] D. Goldman, S. Istrail, and C. H. Papadimitriou. Algorithmic aspects of protein structure similarity. In *Proc. 40th Annu. IEEE Sympos. Found. Comput. Sci.*, pages 512–522, 1999.
- [Ind02] P. Indyk. Approximate nearest neighbor algorithms for Fréchet distance via product metrics. In *Proc. 18th Annu. ACM Sympos. Comput. Geom.*, pages 102 – 106, 2002.
- [KHM<sup>+</sup>98] S. Kwong, Q. H. He, K. F. Man, K. S. Tang, and C. W. Chau. Parallel genetic-based hybrid pattern matching algorithm for isolated word recognition. *Int. J. Pattern Recognition & Artificial Intelligence*, 12(5):573–594, August 1998.
- [KKS05] M.S. Kim, S.W. Kim, and M. Shin. Optimization of subsequence matching under time warping in time-series databases. In *Proc. ACM symp. Applied comput.*, pages 581–586, 2005.
- [KP99] E. J. Keogh and M. J. Pazzani. Scaling up dynamic time warping to massive dataset. In *Proc. of the Third Euro. Conf. Princip. Data Mining and Know. Disc.*, pages 1–11, London, UK, 1999.
- [KS94] A. Kolinski and J. Skolnick. Monte carlo simulations of protein folding: Lattice model and interaction scheme. In *Proteins*, volume 18, pages 338–352, 1994.
- [Lue78] G. S. Lueker. A data structure for orthogonal range queries. In *Proc. 19th Annu. IEEE Sympos. Found. Comput. Sci.*, pages 28–34, 1978.
- [PP90] M. Parizeau and R. Plamondon. A comparative analysis of regional correlation, dynamic time warping, and skeletal tree matching for signature verification. *IEEE Trans. Pattern Anal. Mach. Intell.*, 12(7):710–717, 1990.
- [Rot94] G. Rote. Curves with increasing chords. *Math. Proc. Camb. Phil. Soc.*, 115:1–12, 1994.
- [Rot05] G. Rote. Computing the Fréchet distance between piecewise smooth curves. Technical Report ECG-TR-241108-01, May 2005.
- [Sal89] J. S. Salowe.  $L_\infty$  interdistance selection by parametric search. *Information Processing Letters*, 30:9–14, 1989.
- [Wen02] C. Wenk. *Shape Matching in Higher Dimensions*. PhD thesis, Dept. of Comput. Sci., Freie Universität Berlin, 2002.