Faster Approximate Covering of Subcurves Under the Fréchet Distance

Frederik Brüning
Department of Computer Science, Universität Bonn, Germany

Jacobus Conradi
Department of Computer Science, Universität Bonn, Germany

Anne Driemel
Hausdorff Center for Mathematics, Universität Bonn, Germany

Abstract

Subtrajectory clustering is an important variant of the trajectory clustering problem, where the start and endpoints of trajectory patterns within the collected trajectory data are not known in advance. We study this problem in the form of a set cover problem for a given polygonal curve: find the smallest number k of representative curves such that any point on the input curve is contained in a subcurve that has Fréchet distance at most a given Δ to a representative curve. We focus on the case where the representative curves are line segments and approach this NP-hard problem with classical techniques from the area of geometric set cover: we use a variant of the multiplicative weights update method which was first suggested by Brönniman and Goodrich for set cover instances with small VC-dimension. We obtain a bicriteria-approximation algorithm that computes a set of $O(k \log(k))$ line segments that cover a given polygonal curve of n vertices under Fréchet distance at most $O(\Delta)$. We show that the algorithm runs in $\widetilde{O}(k^2n + kn^3)$ time in expectation and uses $\widetilde{O}(kn + n^3)$ space. For input curves that are c-packed and lie in the plane, we bound the expected running time by $\widetilde{O}(k^2c^2n)$ and the space by $\widetilde{O}(kn + c^2n)$. In addition, we present a variant of the algorithm that uses implicit weight updates on the candidate set and thereby achieves near-linear running time in n without any assumptions on the input curve, while keeping the same approximation bounds. This comes at the expense of a small (polylogarithmic) dependency on the relative arclength.

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

Keywords and phrases Clustering, Set cover, Fréchet distance, Approximation algorithms

Digital Object Identifier 10.4230/LIPIcs.ESA.2022.28

Related Version *Full Version*: `https://arxiv.org/abs/2204.09949` [7]

Funding This work has been funded by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) – AA 1111/2-2 (FOR 2535 Anticipating Human Behavior).

1 Introduction

The advancement of tracking technology made it possible to record the movement of single entities at a large scale in various application areas ranging from vehicle navigation over sports analytics to the socio-ecological study of animal and human behaviour. The types of trajectories that are analyzed range from GPS-trajectories [25] to full-body-motion trajectories [22] and complex gestures [24], and even include the positions of the focus point of attention from a human eye [15, 21].

In many such applications, a flood of data presents us with the challenging task of extracting useful information. If a long trajectory is given as a sequence of positions in some parameter space, it is rarely known in advance which specific movement patterns occur. In particular, it is challenging to find the start and endpoints of such patterns, which is why popular clustering algorithms heuristically partition the trajectories into smaller subtrajectories. An example is the popular algorithm by Lee, Han and Whang [23].

Since the criteria according to which one should detect, group and represent behaviour patterns vary greatly among different kinds of application, there are many different variants of the subtrajectory clustering problem, see also the survey papers [12, 26, 27]. One line of research uses the well-established Fréchet distance to define similarity between subcurves, for example the works of Agarwal et al. [1], Buchin et al. [11] and Akitaya et al. [2].

In an attempt to unify previous definitions of the underlying algorithmic problem, Akitaya et al. [2] define the following geometric set cover problem. Given a polygonal curve, the goal is to "cover" the whole curve with a minimum number of simpler representative curves, such that each point of the trajectory is contained in a subcurve with small Fréchet distance to its closest representative curve. This is in line with traditional clustering formulations such as metric k-center, where clusters may overlap. In this paper, we study the set cover problem introduced by Akitaya et al. and improve upon their results.

Preliminaries. For any $n > 1$, a sequence of points $p_1, \ldots, p_n \in \mathbb{R}^d$ defines a **polygonal curve** P by linearly interpolating consecutive points, that is, for each i, we obtain the **edge** $e_i : [0,1] \to \mathbb{R}^d; t \mapsto (1-t)p_i + tp_{i+1}$. We may write $e_i = \overline{p_i\, p_{i+1}}$ for edges. We may think of P as a continuous function $P : [0,1] \to \mathbb{R}^d$ by fixing n values $0 = t_1 < \ldots < t_n = 1$, and defining $P(t) = e_i\left(\frac{t-t_i}{t_{i+1}-t_i}\right)$ for $t_i \le t \le t_{i+1}$. We call the set $(t_1, \ldots, t_n)$ the **vertex parameters** of the parametrized curve $P : [0,1] \to \mathbb{R}^d$. For $n = 1$, we may slightly abuse notation to view a point p_1 in $\mathbb{R}^d$ as a polygonal curve defined by an edge of length zero with $p_2 = p_1$. We call the number of vertices n the **complexity** of the curve. For any two $a, b \in [0,1]$ we denote with $P[a,b]$ the **subcurve** of P that starts at $P(a)$ and ends at $P(b)$. Note, that $a > b$ is specifically allowed and results in a subcurve in reverse direction. We call the subcurves of edges **subedges**. Let $\mathbb{X}_\ell^d = (\mathbb{R}^d)^\ell$, and think of the elements of this set as the set of all polygonal curves of ℓ vertices in $\mathbb{R}^d$.

For two parametrized curves P and Q, we define their **Fréchet distance** as

$$d_F(P,Q) = \inf_{\alpha,\beta:[0,1]\to[0,1]} \sup_{t\in[0,1]} \|P(\alpha(t)) - Q(\beta(t))\|,$$

where α and β range over all functions that are non-decreasing, surjective and continuous. We call the pair (α, β) a traversal. Every traversal has a distance $\sup_{t\in[0,1]} \|P(\alpha(t)) - Q(\beta(t))\|$ associated to it.

We call a curve X in $\mathbb{R}^d$ **c-packed**, if for any point p and and radius r, the length of X inside the disk is bounded by $||X \cap \mathrm{b}_r(p)|| \le cr$, where $\mathrm{b}_r(p) = \{x \in \mathbb{R}^d \mid \|p - x\| \le r\}$. Let X be a set. We call a set $\mathcal{R}$ where any $r \in \mathcal{R}$ is of the form $r \subseteq X$ a **set system** with **ground set** X.

We say a subset $A \subseteq X$ is **shattered** by $\mathcal{R}$ if for any $A' \subseteq A$ there exists an $r \in \mathcal{R}$ such that $A' = r \cap A$. The **VC-dimension** of $\mathcal{R}$ is the maximal size of a set A that is shattered by $\mathcal{R}$. For a weight function w on the ground set X and a real value $\varepsilon > 0$, we say that a subset $C \subset X$ is an **ε-net** if every set of $\mathcal{R}$ of weight at least $\varepsilon \cdot w(X)$ contains at least one element of C. For any $A \subseteq X$, we write $w(A)$ short for $\sum_{a\in A} w(a)$.

Computational Model. We describe our algorithms in the real-RAM model of computation, which allows to store real numbers and to perform simple operations in constant time on them. We call the following operations **simple operations**. The arithmetic operations $+, -, \times, /$. The comparison operations $=, \neq, >, \ge, \le, <$, for real numbers with output 0 or 1. In addition to the simple operations, we allow the square-root operation. In the full version [7], we describe how to circumvent the square-root operation with little extra cost.

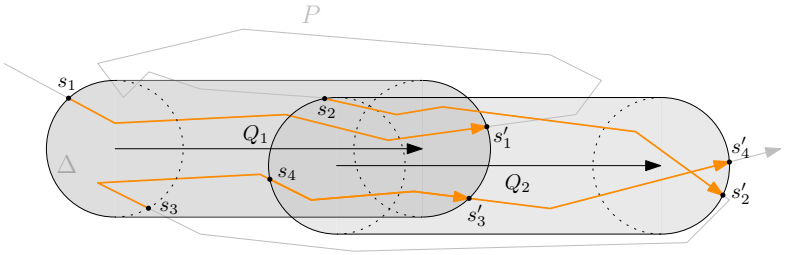

Figure 1 Illustration of the Δ-coverage of a set $C = \{Q_1, Q_2\}$ and a curve P. Here we have $\Psi_\Delta(P, C) = [s_1, s'_1] \cup [s_2, s'_2] \cup [s_3, s'_4]$, since the subcurves $P[s_1, s'_1]$ and $P[s_3, s'_3]$ have Fréchet distance Δ to Q_1, the subcurves $P[s_2, s'_2]$ and $P[s_4, s'_4]$ have Fréchet distance Δ to Q_2 and each other subcurve of P that has Fréchet distance at most Δ to Q_1 or Q_2 is a subcurve of $P[s_i, s'_i]$ for some $1 \leq i \leq 4$.

Problem definition. We study the same problem as Akitaya, Chambers, Brüning and Driemel [2]. Let $P : [0, 1] \to \mathbb{R}^d$ be a polygonal curve of n vertices and let $\ell \in \mathbb{N}$ and $\Delta \in \mathbb{R}$ be fixed parameters. Define the **Δ-coverage** of a set of center curves $C \subset \mathbb{X}_\ell^d$ as follows:

$$\Psi_\Delta(P, C) = \bigcup_{q \in C} \bigcup_{0 \leq t \leq t' \leq 1} \{s \in [t, t'] \mid d_F(P[t, t'], q) \leq \Delta\}.$$

The Δ-coverage corresponds to the part of the curve P that is covered by the set of all subtrajectories that are within Fréchet distance Δ to some curve in C. If for some P, C, Δ it holds that $\Psi_\Delta(P, C) = [0, 1]$, then we call C a **Δ-covering** of P. The problem is to find a Δ-covering $C \subset \mathbb{X}_\ell^d$ of P of minimum size. We study bicriterial approximation algorithms for this problem, which we formalize as follows.

▶ **Definition 1** ((α, β)-approximate solution). *Let $P \in \mathbb{X}_n^d$ be a polygonal curve, $\Delta \in \mathbb{R}_+$ and $\ell \in \mathbb{N}$. A set $C \subseteq \mathbb{X}_\ell^d$ is an (α, β)-approximate solution to the Δ-coverage problem on P, if C is an $\alpha\Delta$-covering of P and there exists no Δ-covering $C' \subseteq \mathbb{X}_\ell^d$ of P with $\beta|C'| < |C|$.*

Related work. Buchin, Buchin, Gudmundsson, Löffler and Luo were the first to consider the problem of clustering subtrajectories under the Fréchet distance [10]. They consider the problem of finding a single cluster of subtrajectories with certain qualities, like the number of distinct subtrajectories, or the length of the longest subtrajectory assigned to it. In their paper, they suggested a sweepline approach in the parameter space of the curves and obtain constant-factor approximation algorithms for finding the largest cluster. They also show NP-completeness of the corresponding decision problems. This hardness result extends to $(2 - \varepsilon)$-approximate algorithms. For their 2-approximation algorithm, Buchin et al. [10] develop an algorithm that finds a legible cluster center among the subcurves of the input curve. Gudmundsson and Wong [19] present a cubic conditional lower bound for this problem and show that it is tight up to a factor of $O(n^{o(1)})$, where n is the number of vertices.

The algorithmic ideas presented in [10] were implemented and extended by Gudmundsson and Valladares [18] who obtained practical speed ups using GPUs. In a series of papers, these ideas were also applied to the problem of reconstructing road maps from GPS data [8, 9]. In a similar vain, Buchin, Kilgus and Kölzsch [11] studied the trajectories of migrating animals and defined so-called group diagrams which are meant to represent the underlying migration patterns in the form of a graph. In their algorithm, to build the group diagram, they repeatedly find the largest cluster and remove it from the data, inspired by the classical greedy set cover algorithm.

The above cited works however do not offer theoretical guarantees when used for computing a clustering of subtrajectories, nor do they explicitly formulate a clustering objective. Agarwal, Fox, Munagala, Nath, Pan, and Taylor [1] define an objective function for clustering subtrajectories based on the metric facility location problem, which consists of a weighted sum over different quality measures such as the number of centers and the distances between cluster centers and their assigned trajectories. While they show NP-hardness for determining whether an input curve can be covered with respect to the Fréchet distance, they also present a $O(\log^2 n)$-approximation algorithm for clustering κ-packed curves (for some constant κ) under the discrete Fréchet distance, where n denotes the total complexity of the input. The overall running time of their algorithm is roughly quadratic in n, cubic in κ and depends logarithmically on the spread of the vertex coordinates.

In our paper, we focus on the clustering formulation previously studied by Akitaya, Chambers, Brüning, and Driemel [2]. They present a pseudo-polynomial algorithm that computes a bi-criterial approximation in the sense of Definition 1 with expected running time in $\tilde{O}(k(\frac{\lambda}{\Delta})^2 + \frac{\lambda}{\Delta}n)$, where λ denotes the total arclength of the input trajectory. The algorithm finds an (α, β)-approximate solution with $\alpha \in O(1)$ and $\beta \in O(\ell^2 \log(k\ell))$. In combination with our Lemma 2, below, this can be directly improved to $O(\ell \log(k))$. It should be noted that in this problem formulation some complexity constraint on the eligible cluster centers is needed to prevent the entire input curve being a cluster center in a trivial clustering.

Our contribution. Our main result is an algorithm that computes an (α, β)-approximate solution with $\alpha \in O(1)$ and $\beta = O(\ell \log k)$, where k is the size of an optimal solution. For general curves, the algorithm runs in $\widetilde{O}(k^2 n + kn^3)$ time in expectation and uses $\widetilde{O}(kn + n^3)$ space. (The $\widetilde{O}(\cdot)$ notation hides polylogarithmic factors in n to simplify the exposition.) If the input curve is a c-packed polygonal curve in the plane, the expected running time can be bounded by $\widetilde{O}(k^2 c^2 n)$ and the space is in $\widetilde{O}(kn + c^2 n)$. In higher dimensions, the bound for c-packed curves becomes quadratic in n. Our second result is an algorithm that achieves near-linear running time in n – even for general polygonal curves – while keeping the same approximation bounds at the expense of a small dependency on the arclength in the running time. The algorithm needs in expectation $\widetilde{O}(nk^3 \log^4(\frac{\lambda}{\Delta k}))$ time and $\widetilde{O}(nk \log(\frac{\lambda}{\Delta k}))$ space, where λ is the total arclength of the input curve. Here, we stated our results for general ℓ using the reduction described below (Lemma 2).

In our algorithms we use a variant of the multiplicative weights update method [5], which has been used earlier for set cover problems with small VC-dimension [6, 13]. The difficulty in our case is that the set system initially has high VC-dimension, as shown by Akitaya et al [2] – namely $\Theta(\log n)$ in the worst case. We circumvent this by defining an intermediate set cover problem where the VC-dimension is significantly reduced. We then show how to compute a finite set system using a carefully chosen set of candidate curves on which the multiplicative weight update method can be applied. A key idea that enables our results is a curve simplification that requires the curve to be locally maximally simplified, a notion that is borrowed from de Berg, Cook, and Gudmundsson [14]. To the best of our knowledge, our candidate generation yields the first strongly polynomial algorithm for approximate subtrajectory clustering under the continuous Fréchet distance. In the full version [7], we also discuss how our candidate set can be used for the related problem of maximizing the coverage. Our second algorithm improves the dependency on the relative arclength from quadratic to polylogarithmic as compared to [2].

Reduction to line segments. In the remainder of the paper, we will focus on finding a Δ-covering with line segments, that is $\ell = 2$. The following lemma provides the reduction for general ℓ at the expense of an increased approximation factor.

▶ **Lemma 2.** *Let $P \in \mathbb{X}_n^d$ be a polygonal curve, $\Delta \in \mathbb{R}_+$ and $\ell \in \mathbb{N}$. Let $C \subseteq \mathbb{X}_\ell^d$ be a Δ-covering of P of minimum cardinality. There exists a set of line segments $C' \subseteq \mathbb{X}_2^d$ that is a Δ-covering of P with $|C'| \leq (\ell - 1)|C|$.*

Proof. Choose as set C' the union of the set of edges of the polygonal curves of C. Clearly, this set has the claimed cardinality and is a Δ-covering of P. ◀

Roadmap. In Section 2 we develop a structured variant of our problem that allows us to apply the multiplicative weight update method in the style of Brönniman and Goodrich [6] in an efficient way. Our intermediate goal is to obtain a structured set of candidates for a modified coverage problem that is on the one hand easy to compute and on the other hand sufficient to obtain good approximation bounds for the original problem. We first define our notion of curve simplification. A crucial property of this simplification is that the relevant subcurves of the input are within small Fréchet distance to subcurves of constant complexity of the simplification. We then define a structured notion of Δ-coverage and a candidate space, which lets us take advantage of this fact. We show that we can narrow our choice down even further, to a finite set of subedges of the simplification, and still sufficiently preserve the quality of the solution. In Section 3, we present our main algorithm. The algorithm uses the concepts and techniques developed in Section 2 in combination with the multiplicative weights update method. In Section 4, we analyze the approximation factor and running time of this algorithm. Crucially, we show that the VC-dimension of the induced set system which is implicitly used by our algorithm is small by design.

2 Structuring the solution space

In this section, we introduce key concepts that allow us to transfer the problem to a set cover problem on a finite set system with small VC-dimension and still obtain good approximation bounds. The main result of this section is Theorem 14.

Simplifications and containers. We start by defining the notion of curve-simplification that we will use throughout the paper.

▶ **Definition 3** (simplification). *Let P be a polygonal curve in $\mathbb{R}^d$. Let $(t_1, \ldots, t_n)$ be the vertex-parameters of P, and $p_i = P(t_i)$ the vertices of P. Consider an index set $1 \leq i_1 < \ldots < i_k \leq n$ that defines vertices p_{i_j}. We call a curve S defined by such an ordered set of vertices $(p_{i_1}, \ldots, p_{i_k}) \in (\mathbb{R}^d)^k$ a **simplification** of P. We say the simplification is Δ-**good**, if the following properties hold:*

- **(i)** $\|p_{i_j} - p_{i_{j+1}}\| \geq \frac{\Delta}{3}$ *for* $1 \leq j < k$
- **(ii)** $d_F(P[t_{i_j}, t_{i_{j+1}}], \overline{p_{i_j}\, p_{i_{j+1}}}) \leq 3\Delta$ *for all* $1 \leq j < k$.
- **(iii)** $d_F(P[t_1, t_{i_1}], \overline{p_{i_1}\, p_{i_1}}) \leq 3\Delta$ *and* $d_F(P[t_{i_k}, t_n], \overline{p_{i_k}\, p_{i_k}}) \leq 3\Delta$
- **(iv)** $d_F(P[t_{i_j}, t_{i_{j+2}}], \overline{p_{i_j}\, p_{i_{j+2}}}) > 2\Delta$ *for all* $1 \leq j < k - 1$

Our intuition is the following. Property (i) guarantees that S does not have short edges. Property (ii) and (iii) together tell us, that the simplification error is small. Property (iv) tells us, that the simplification is (approximately) maximally simplified, that is, we cannot remove a vertex, and hope to stay within Fréchet distance 2Δ to P.

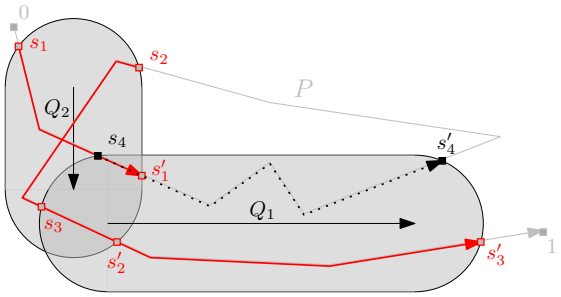

Figure 2 Example of the structured Δ-coverage of a set $C = \{Q_1, Q_2\}$ and a curve P. Here we have $\Psi'_\Delta(P, C) = [s_1, s'_1] \cup [s_2, s'_3]$ since the subcurves $P[s_1, s'_1]$ and $P[s_2, s'_2]$ have Fréchet distance Δ to Q_1 and $P[s_3, s'_3]$ has Fréchet distance Δ to Q_2. Note that $[s_4, s'_4]$ is not part of the coverage since the subcurve $P[s_4, s'_4]$ consists of 4 edges.

▶ **Definition 4** (Container). *Let P be a polygonal curve, let $\pi = P[s,t]$ be a subcurve of P, and let $(t_1, \ldots, t_n)$ be the vertex-parameters of P. For a simplification S of P defined by index set $I = (i_1, \ldots, i_k)$, define the* ***container*** *$c_S(\pi)$ of π on S as $S[t_a, t_b]$, with $a = \max(\{i_1\} \cup \{i \in I \mid t_i \leq s\})$ and $b = \min(\{i \in I \mid t_i \geq t\} \cup \{i_k\})$.*

The following lemma has been proven by de Berg et al. [14]. We restate and reprove it here with respect to our notion of simplification.

▶ **Lemma 5** ([14]). *Let P be a polygonal curve in $\mathbb{R}^d$, and let S be a Δ-good simplification of P. Let Q be an edge in $\mathbb{R}^d$ and let π be a subcurve of P with $d_F(Q, \pi) \leq \Delta$. Then $c_S(\pi)$ consists of at most 3 edges.*

Proof. Assume for the sake of contradiction, that $c_S(\pi)$ contains 4 edges, that is it has three internal vertices s_1, s_2, s_3. By Definition 4 these three vertices are also interior vertices of π. As the Fréchet distance $d_F(Q, \pi) \leq \Delta$, there are points $q_1, q_2, q_3 \in Q$, that get matched to s_1, s_2 and s_3 respectively during the traversal, with $\|s_i - q_i\| \leq \Delta$. This implies $d_F(\pi[s_1, s_3], \overline{q_1\, q_3}) \leq \Delta$. It also implies, that $d_F(\overline{s_1\, s_3}, \overline{q_1\, q_3}) \leq \Delta$. But then

$$d_F(\overline{s_1\, s_3}, P[s_1, s_3]) = d_F(\overline{s_1, s_3}, \pi[s_1, s_3]) \leq d_F(\overline{s_1\, s_3}, \overline{q_1\, q_3}) + d_F(\pi[s_1, s_3], \overline{q_1\, q_3}) \leq 2\Delta,$$

contradicting the assumption that S is a Δ-good simplification. ◀

Structured coverage and candidate space. We want to make use of the property of Δ-good simplifications shown in Lemma 5. For this we adapt the notion of Δ-coverage from Section 1 as follows.

▶ **Definition 6.** *Let S be a polygonal curve in $\mathbb{R}^d$. Let $(t_1, \ldots, t_n)$ be the vertex-parameters of S. Let $\ell \in \mathbb{N}$ and $\Delta \in \mathbb{R}$ be fixed parameters. Define the* ***structured Δ-coverage*** *of a set of center curves $C \subset \mathbb{X}^d_\ell$ as*

$$\Psi'_\Delta(S, C) = \bigcup_{q \in C} \bigcup_{(i,j) \in J} \Psi^{(i,j)}_\Delta(S, q)$$

where

$$\Psi^{(i,j)}_\Delta(S, q) = \{s \in [t, t'] \mid t_i \leq t \leq t_{i+1};\ t \leq t';\ t_{j-1} \leq t' \leq t_j;\ d_F(S[t, t'], q) \leq \Delta\},$$

and where $J = \{1 \leq i \leq j \leq n \mid 1 \leq j - i \leq 4\}$.

If it holds that $\Psi'_\Delta(S, C) = [0, 1]$, then we call C a ***structured Δ-covering*** *of S.*

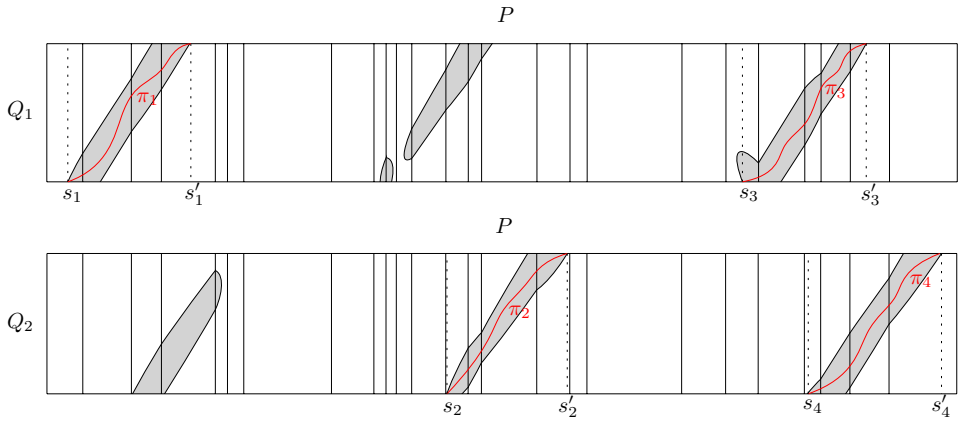

Figure 3 Free space diagrams of the curves P and Q_1 (resp. Q_2) depicted in Figure 1. The monotone paths π_i illustrate that the Fréchet distance between $P[s_i, s_i']$ and Q_1 (resp. Q_2) is equal to Δ for $1 \leq i \leq 4$.

▶ **Observation 7.** *In general for any polygonal curve S and set of center curves C it holds that $\Psi'_\Delta(S, C) \subseteq \Psi_\Delta(S, C)$.*

We now want to restrict the candidate set to subedges of a simplification of the input curve, thereby imposing more structure on the solution space. For this we begin by defining a more structured parametrization of the set of edges of a polygonal curve.

▶ **Definition 8** (Edge space)**.** *We define the **edge space** $\mathbb{T}_n = \{1, \ldots, n-1\} \times [0, 1]$. We denote the set of edges of P with $E(P)$.*

▶ **Definition 9** (Candidate space)**.** *Let $E = \{e_1, \ldots, e_{n-1}\}$ be an ordered set of edges in $\mathbb{R}^d$. We define the **candidate space** induced by E as the set $\mathcal{Z}_E = \{(i_1, t_1, i_2, t_2) \in \mathbb{T}_n \times \mathbb{T}_n \mid i_1 = i_2\}$. We associate an element $(i, t_1, i, t_2) \in \mathcal{Z}_E$ with the subedge $\overline{e_i(t_1)\, e_i(t_2)}$.*

The following theorem summarizes and motivates the above definitions of structured coverage and candidate space. Namely, we can restrict the search space to subedges of the simplification S and still obtain a good covering of P. Moreover, we can evaluate the coverage of our solution solely based on S. The structured coverage only allows subcurves of S that consist of at most three edges to contribute to the coverage. This technical restriction is necessary to obtain a small VC-dimension in our main algorithm later on, and it is well-motivated by Lemma 5.

▶ **Theorem 10.** *Let S be a Δ-good simplification of a curve P. Let C be a set of subedges of edges of S. If C is a structured 8Δ-covering of S, then C is an 11Δ-covering of P. Moreover, if k is the size of an optimal Δ-covering of P, then there exists such a set C of size at most $3k$.*

Partial traversals and coverage. Our algorithm and analysis use the notion of the free space diagram which was first introduced by Alt and Godau [3]. It is instructive to consider this concept in the context of the coverage problem. Refer to Figure 3.

▶ **Definition 11** (Free space diagram)**.** *Let P and Q be two polygonal curves parametrized over $[0, 1]$. The free space diagram of P and Q is the joint parametric space $[0, 1]^2$ together with a not necessarily uniform grid, where each vertical line corresponds to a vertex of P and each horizontal line to a vertex of Q. The Δ-**free space** of P and Q is defined as*

$$\mathcal{D}_\Delta(P, Q) = \left\{(x, y) \in [0, 1]^2 \mid \|P(x) - Q(y)\| \leq \Delta\right\}$$

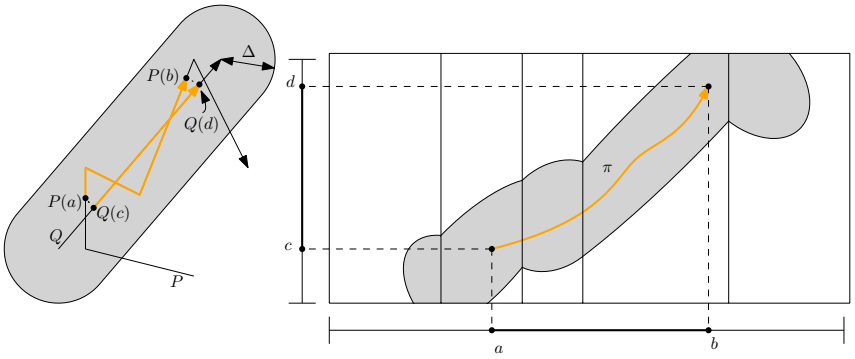

Figure 4 An illustration of a Δ-feasible $(2,4)$-partial traversal π from (a,c) to (b,d) of P and Q. π covers all points between a and b on P, and all points between c and d on Q.

This is the set of points in the parametric space, whose corresponding points on P and Q are at a distance at most Δ. The edges of P and Q segment the free space into cells. We call the intersection of $\mathcal{D}_\Delta(P,Q)$ with the boundary of cells the ***free space intervals****.*

Alt and Godau [3] showed that the Δ-free space inside any cell is convex and has constant complexity. More precisely, it is an ellipse intersected with the cell. Furthermore, the Fréchet distance between two curves is less than or equal to Δ if and only if there exists a path $\pi : [0,1] \to \mathcal{D}_\Delta(P,Q)$ that starts at $(0,0)$, ends in $(1,1)$ and is monotone in both coordinates.

▶ **Definition 12** (Partial traversal)**.** *Let P be a polygonal curve in $\mathbb{R}^d$, and let $(t_1,\ldots,t_n)$ be the vertex-parameters of P. Let $1 \le i < j \le n$ be integer values. Let Q be an edge in $\mathbb{R}^d$. We define an (i,j)-****partial traversal*** *as a pair of continuous, monotone increasing and surjective functions, $f : [0,1] \to [a,b]$ and $g : [0,1] \to [c,d]$, where $t_i \le a \le t_{i+1}$, $t_{j-1} \le b \le t_j$, $0 \le a \le b \le 1$, and $0 \le c \le d \le 1$. We say that (f,g) is a partial traversal from (a,c) to (b,d).*

▶ **Definition 13** (Δ-feasible)**.** *We say that a partial traversal is Δ-****feasible*** *if the image of the path $\pi : [0,1] \to [0,1]^2$ defined by $\pi(t) = (f(t),g(t))$ is contained inside the Δ-free space $\mathcal{D}_\Delta(P,Q)$. We say that π* ***covers*** *a point t on P if $t \in [a,b]$ and we say that π covers a point t on Q if $t \in [c,d]$.*

A finite set of candidates. By Theorem 10, it is sufficient to find a structured covering using a suitable simplification of the input curve. However the corresponding search space would still be infinite, even for a single edge. We will next define a finite set of candidates and show that it contains a good solution. In particular, our goal is to prove the following theorem.

▶ **Theorem 14.** *Let P be a polygonal curve of complexity n in $\mathbb{R}^d$ and let $\Delta > 0$ be given. Let S be a Δ-good simplification of P. Assume there exists a Δ-covering C of P of cardinality k. Then, there exists an algorithm that computes in $O(n^3)$ time and space a set of candidates $B \subset \mathcal{Z}_{E(S)} \subset \mathbb{X}_2^d$ with $|B| \in O(n^3)$, such that B contains a structured 8Δ-covering C_B of S of size at most $12k$. Moreover, C_B is a 11Δ-covering of P.*

The main steps to constructing this set of candidates B are as follows. We first define a special set of subcurves of the simplification S. Intuitively, these are the containers of S of subcurves of P that may contribute to the coverage.

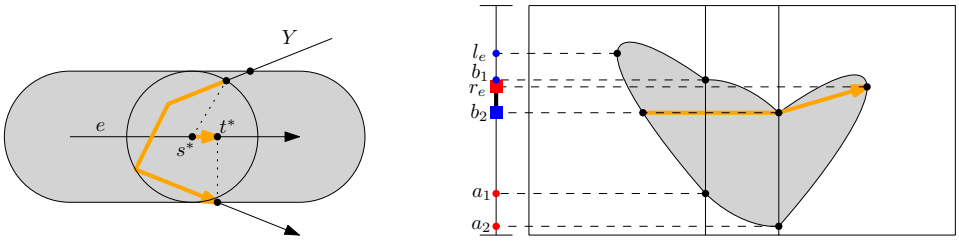

Figure 5 Examples of extremal points. Shown on the right are the two free space intervals $[a_1, b_1]$ and $[a_2, b_2]$ as well as the left- and rightmost points l_e and r_e of the Δ-free space of e and Y. The extremal points are defined by b_2 and r_e. All points considered for the first extremal point are shown in blue. Similarly all points considered for the second extremal point are shown in red. A traversal from the first extremal point to the second extremal point is illustrated. On the left the resulting subedge $e[s^*, t^*]$ and the maximal subcurve of Y that can be matched are illustrated.

▶ **Definition 15** (Generating subcurves). *Let S be a Δ-good simplification of a polygonal curve P. Let $(t_1, \ldots, t_m)$ be the vertex-parameters of S. For any $1 \leq i$, $1 \leq j \leq 3$ and $i + j \leq m$, we say the subcurve $S[t_i, t_{i+j}]$ is a **generating subcurve**. In particular, this defines all subcurves of at most three edges starting and ending at vertices of S.*

Now, for every generating subcurve Y of S and every edge e of S, we can identify an interval defining a subedge of e, that maximizes the Δ-coverage on Y over all subedges of e. For this reason, we call the endpoints of this interval extremal. Using this definition we define the finite candidate set induced by S via generating triples.

▶ **Definition 16** (Δ-extremal points). *Given a value of $\Delta > 0$, a polygonal curve $Y : [0,1] \to \mathbb{R}^d$ of m edges and an edge $e : [0,1] \to \mathbb{R}^d$, such that they permit a Δ-feasible $(1,m)$-partial traversal. As e is a single edge, the Δ-free space of Y and e consists of a single row. Let $[a_i, b_i]$ be the ith vertical free space interval of the Δ-free space of Y and e. Denote by $l = (l_Y, l_e)$ the leftmost point in the Δ-free space of Y and e and $r = (r_Y, r_e)$ the rightmost point (in case l is not unique, chose the point with smallest y-coordinate, and r as the point with the biggest y-coordinate). We define the **Δ-extremal points** induced by Y on e as the tuple $\mathcal{E}_\Delta(Y, e) = (s, t) \in [0,1]^2$ with $s = \min(\{l_e\} \cup \{b_1, \ldots, b_{n-1}\})$ and $t = \max(\{r_e\} \cup \{a_1, \ldots, a_{n-1}\})$. We explicitly allow that $t < s$.*

▶ **Definition 17** (Generating triples). *Let S be a Δ-good simplification of a polygonal curve P. We define the set of **generating triples** T_S as a set of triples (e, Y_1, Y_2), where e is any edge of S, and Y_1 and Y_2 are generating subcurves of S (not necessarily distinct). We include the triple (e, Y_1, Y_2) in the set T_S if and only if there are points $p \in e$, $p_1 \in Y_1$ and $p_2 \in Y_2$ such that $||p - p_1|| \leq 8\Delta$ and $||p - p_2|| \leq 8\Delta$.*

▶ **Definition 18** (Candidate set). *Let $\Delta > 0$ be a given value and let S be a Δ-good simplification of a polygonal curve P. Let T_S be the set of generating triples of S. We define the **candidate set** induced by S with respect to Δ as the set of line segments*

$$B = \{e[s_1, t_2] \mid \exists\ (e, S_1, S_2) \in T_S,\ \text{s.t. } \mathcal{E}_{8\Delta}(S_i, e) = (s_i, t_i) \text{ for } i \in \{1, 2\}\}$$

Clearly, the set B can be computed in $O(|T_S|)$ time and space, if the set T_S is given.

In the full version [7], we show that, for any suitable covering, we can deform each subedge of the solution to one of our candidates while retaining the coverage on a fixed subcurve. However, while retaining coverage on one subcurve, we may lose coverage on

another subcurve in the same cluster. We show, through a case analysis, how to deal with all subcurves at once while increasing the number of clusters by a factor of at most 4. We use this fact together with Theorem 10 to prove Theorem 14.

3 The main algorithm

We describe the main algorithm below with pseudocode specified in Algorithm 1 and Algorithm 2. Specifications of the missing subroutines are given in Table 1. Several additional building blocks of the algorithm are described in the full version [7]: computing candidates, computing the structured coverage, testing feasibility and computing simplifications.

Algorithm. The algorithm receives as input a polygonal curve P in $\mathbb{R}^d$ and a parameter $\Delta \geq 0$. The goal is to compute an small set of edges C, such that all points on P are covered by the Δ'-coverage of C on P for some $\Delta' \in O(\Delta)$. The algorithm ApproxCover (see Algorithm 1), when called with input P and Δ, first computes a Δ-good simplification S of P and generates a finite subset B of the candidate space $\mathcal{Z}_{E(S)} \subset \mathbb{X}_2^d$ defined on the edges of this simplification. For this, we use the construction of the candidate set presented in Section 2. The algorithm then performs an exponential search with the variable k that controls the target size of the solution. Starting with a constant k, the algorithm tries to find a solution of size approximately k and if this fails, the algorithm doubles k and continues. For finding a solution with fixed target size, the algorithm kApproxCover is used (see Algorithm 1). This algorithm is called with the simplification S, the candidate set B and set of parameters r, Δ', k', and $i_{\max}$. The algorithm kApproxCover uses a variant of the multiplicative weight update method with a maximum number of (proper) iterations bounded by $i_{\max}$. In the ith iteration, we take a sample from a discrete probability distribution $\mathcal{D}_i$ that is defined on B via a weight function $w_i : B \to \mathbb{R}$, where the probability of an element $e \in B$ being in the sample is defined as $w_i(e)/\sum_{e \in B} w_i(e)$. For the initial distribution $\mathcal{D}_1$, all weights are set to 1, which corresponds to the uniform distribution over B. During the course of the algorithm, we repeatedly update this distribution thereby generating distributions $\mathcal{D}_1, \mathcal{D}_2, \ldots$ (up to $\mathcal{D}_{i_{\max}}$, unless the algorithm finds a solution in an earlier iteration). The update step performed by a call to subroutine UpdateWeight proceeds by doubling the weight of the subset F of B. This can be done in $O(|B|)$ time and space by storing the cumulative probability distribution.

With this basic mechanism in place, the algorithm kApproxCover now proceeds as follows. In each iteration, the algorithm computes a set $C \subset B$ by taking k' independent draws from the current distribution $\mathcal{D}_i$. Then, the algorithm checks, if C is a solution to our problem by a call to the subroutine PointNotCovered. The subroutine should either return that all points on S are in the Δ'-coverage of the solution C, or return a point t on S that is not covered in this way. This can be done by computing the structured coverage $\Psi'_{\Delta'}(S, C)$ explicitly. In the former case, the algorithm returns the solution and terminates. In the latter case, we compute the subset F of candidates B that would cover t with respect to the subcurves that contain t and which have at most 3 edges. To compute F, we simply iterate over all elements of B and check if t is covered by a call to IsFeasible (see Algorithm 2). (For technical reasons, we parametrize the curve P via the edge space of the set of edges of P, so that we can locate the edge that contains t in constant time.) It is important that F is not a multiset, so repeated additions of an element will not increase its weight.

At this point we would like to perform the weight update step which we described above with respect to the set F, however, we only do this if the weight of the set F is small. If the total weight of the set F is larger than a $\frac{1}{r}$-fraction of the total weight of B, then we simply skip the update step and continue by taking another sample from the current distribution.

Algorithm 1 Main algorithm.

procedure ApproxCover($P \in \mathbb{X}_n^d$, $\Delta \in \mathbb{R}$)
  $S \leftarrow$ SimplifyCurve(P, Δ)
  $B \leftarrow$ GenerateCandidates(S, Δ)
  $k \leftarrow 1$
  $\gamma \leftarrow 110d + 412$ ▷ bound on the VC-dimension
  repeat
    $k \leftarrow 2k$ ▷ increase target size for solution
    $r \leftarrow 2k$, $\Delta' \leftarrow \alpha\Delta$, $k' \leftarrow \lceil 16k\gamma \log(16k\gamma) \rceil$, $i_{\max} \leftarrow 5k \log_2(\frac{|B|}{k})$
    $C \leftarrow$ kApproxCover$(S, B, r, \Delta', k', i_{\max})$ ▷ search solution with this size
  until $C \neq \emptyset$ ▷ until we find a solution
  return C

procedure kApproxCover($S \in \mathbb{X}_n^d$, $B \subset \mathbb{X}_2^d$, $r, \Delta' \in \mathbb{R}$, k', $i_{\max} \in \mathbb{N}$)
  Let $\mathcal{D}_1$ be the uniform distribution over B with weight function $w_1 : B \to \{1\}$
  $i \leftarrow 1$
  repeat
    $C \leftarrow$ sample k' elements from $\mathcal{D}_i$
    $t \leftarrow$ PointNotCovered(C, S, Δ')
    if $t = -1$ **then return** C ▷ if all points covered, return solution found
    $F \leftarrow \emptyset$ ▷ otherwise, compute feasible set of t
    for each $Q \in B$ **do**
      if IsFeasible(Q, S, t, Δ') **then** add Q to F
    if $\Pr_{\mathcal{D}_i}[F] \leq \frac{1}{r}$ **then**
      $\mathcal{D}_{i+1} \leftarrow$ WeightUpdate$(\mathcal{D}_i, F)$ ▷ increase the probability of F
      $i \leftarrow i + 1$
  until $i > i_{\max}$
  return $\emptyset$ ▷ no solution found for this target size

Algorithm 2 Subroutine IsFeasible which is called by the main algorithm.

procedure IsFeasible($Q \in \mathbb{X}_2^d, S \in \mathbb{X}_n^d, t \in \mathbb{T}_n, \Delta' \in \mathbb{R}$)
  $(t', i') \leftarrow t$ ▷ locate edge of t on S
  $J = \{1 \leq i \leq j \leq n \mid 1 \leq j - i \leq 4; i \geq i' - 3; j \leq i' + 4\}$ ▷ find generating subcurves
  for $(i, j) \in J$ **do** ▷ check if Q covers t on S
    if $t \in \Psi_{\Delta'}^{i,j}(S, Q)$ **then return** true
  return false

Table 1 Specification of additional subroutines used in the main algorithm.

Procedure	Input	Output
SimplifyCurve	$P \in \mathbb{X}_n^d$, $\Delta \geq 0$	Δ-good simplification of P (Def. 3)
GenerateCandidates	$S \in \mathbb{X}_n^d$, $\Delta \geq 0$	candidate set (Def. 18)
PointNotCovered	$C \subset \mathbb{X}_2^d$, $S \in \mathbb{X}_n^d$, $\Delta \geq 0$	either $t \in \mathbb{T}_n \setminus \Psi'_\Delta(S, C)$ or -1 if this set is empty
WeightUpdate	distribution $\mathcal{D}$ given by weight function $w : B \to \mathbb{R}$, $F \subset B$	$\mathcal{D}'$ with $w' : B \to \mathbb{R}$ where weight is doubled for all elements of F

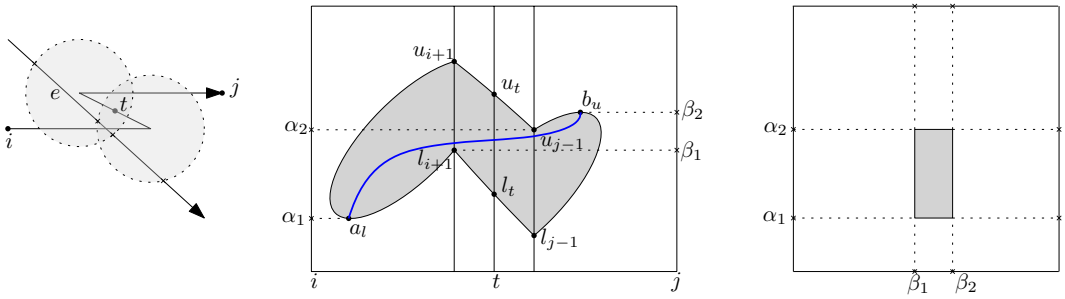

Figure 6 Example for the construction of the rectangle $R = [\alpha_1, \alpha_2] \times [\beta_1, \beta_2]$ for fixed P, i, j, t, Δ and e. The left image shows the curves $P[t_i, t_j]$ and e with two circles of radius Δ around $P(t_{i+1})$ and $P(t_{j-1})$. The middle image shows the corresponding Δ-free space diagram with a (i, j)-partial traversal from a_l to b_u and the right image shows the rectangle R in the parameter space $[0, 1]^2$ of e.

4 Analysis of the main algorithm

The algorithm described in Section 3 is based on the set cover algorithm by Brönniman and Goodrich [6]. A crucial step in the analysis of this algorithm is the analysis of the VC-dimension of the dual set system. In our case this is a set system formed by the sets F computed in the main algorithm. For the formal analysis of this set system, we introduce the notion of feasible sets.

▶ **Definition 19** (Feasible set). *Let $S : \mathbb{T}_n \to \mathbb{R}^d$ be a polygonal curve and let $B \subset \mathbb{X}_2^d$ be a candidate set of edges and let $\Delta \geq 0$ be a real value. For any point $t \in \mathbb{T}_n$, we define the **feasible set** of t as the set of elements $Q \in B$ that admit an (i, j)-partial traversal with S that fully covers Q and that covers t on S, with the additional condition that $j - i \leq 3$. We denote the feasible set of t with $F_\Delta(t)$.*

Note that for any fixed S and Δ the set of feasible sets $\{F_\Delta(t) \mid t \in \mathbb{T}_n\}$ is exactly the set system determined by the subroutine IsFeasible described in Algorithm 2. We claim that any feasible set can be split into sets corresponding to the edges of the simplification, where each set consists of a constant union of rectangles in the candidate space restricted to the respective edge. Figure 6 illustrates one of those rectangles. The following lemma provides the formal statement.

▶ **Lemma 20.** *Let P be a polygonal curve in $\mathbb{R}^d$ and let $e \in \mathbb{X}_2^d$ be an edge. Let $(t_1, \ldots, t_n)$ be the vertex-parameters of P. For any integer values $1 \leq i < j \leq \min(i + 3, n)$ and real value $t \in [0, 1]$ with $t_i \leq t \leq t_j$, either there exist $\alpha_1, \alpha_2, \beta_1, \beta_2$ such that*

$$R := \{(\alpha, \beta) \in [0, 1]^2 \mid t \in \Psi_\Delta^{i,j}(P, e[\alpha, \beta])\} = [\alpha_1, \alpha_2] \times [\beta_1, \beta_2],$$

or the set R is empty. Moreover, each α_v (respectively β_v) for $v \in \{1, 2\}$ can be written as $\alpha_v = c_v + \sqrt{d_v}$ (respectively $\beta_v = e_v + \sqrt{f_v}$), where the parameters c_v and d_v (respectively e_v and f_v) can be computed by an algorithm that takes $(i, j), t$ and e as input and needs $O(d)$ simple operations.

To prove a VC-dimension bound of $O(d)$, we combine the above lemma with the following general theorem which can be attributed to Goldberg and Jerrum [16]. We use the variant by Anthony and Bartlett [4], which is stated as follows.

▶ **Theorem 21** (Theorem 8.4 [4]). *Suppose h is a function from $\mathbb{R}^a \times \mathbb{R}^b$ to $\{0, 1\}$ and let $H = \{x \to h(\alpha, x) \mid \alpha \in \mathbb{R}^a\}$ be the class determined by h. Suppose that h can be computed by an algorithm that takes as input the pair $(\alpha, x) \in \mathbb{R}^a \times \mathbb{R}^b$ and returns $h(\alpha, x)$ after no more than t simple operations. Then, the VC-dimension of H is $\leq 4a(t + 2)$.*

▶ **Lemma 22.** *Let $S : \mathbb{T}_n \to \mathbb{R}^d$ be a polygonal curve and let $\Delta \in \mathbb{R}_+$. Consider the set system $\{F_\Delta(t) \mid t \in \mathbb{T}_n\}$ with ground set $\mathbb{X}_2^d$. The VC-dimension of this set system is in $O(d)$.*

Proof. Define a function $h : \mathbb{T}_n \times \mathbb{X}_2^d \to \{0, 1\}$ with $h(t, Q) = 1$ if and only if a call to IsFeasible(Q, S, t, Δ) returns true. We analyse the VC-dimension of the class of functions determined by h:

$$H = \{x \to h(t, x) \mid t \in \mathbb{T}_n\}$$

As a consequence, we obtain the same bounds on the VC-dimension of the corresponding set system $\mathcal{R}$ with ground set $\mathbb{X}_2^d$ where a set $r_t \in \mathcal{R}$ is defined by a $t \in \mathbb{T}_n$ with

$$r_t = \{Q \in \mathbb{X}_2^d \mid h(t, Q) = 1\}$$

In order to show the lemma, we first argue that for any given $t \in \mathbb{T}_n$ and $Q \in \mathbb{X}_d^2$ the expression $h(t, Q)$ can be evaluated with $O(d)$ simple operations.

Let $(t', i') = t$ and recall the index set $J = \{(i, j) \mid i' - 3 \leq i \leq i' \leq j \leq i + 3\}$ as in the procedure IsFeasible. Note that $|J| = 9$ and that J can be determined by $O(1)$ simple operations from i'. Note that IsFeasible returns true if and only if $t \in \Psi_\Delta^{i,j}(S, Q)$ for some $(i, j) \in J$. So, for fixed (i, j), consider the set

$$R = \{(\alpha, \beta) \in [0, 1]^2 \mid t \in \Psi_\Delta^{i,j}(S, Q[\alpha, \beta])\}$$

Lemma 20 implies that R is either empty or can be written as a rectangle $[\alpha_1, \alpha_2] \times [\beta_1, \beta_2]$. Note that $t \in \Psi_\Delta^{i,j}(S, Q)$ if and only if R is non-empty and $(0, 1) \in R$. By Lemma 20, this test can be performed using $O(d)$ simple operations. Thus, we can apply Theorem 21 and conclude that the VC-dimension of H is in $O(d)$. ◀

With proper bounds on the VC-dimension in place, we obtain the following main result. The proof is based on the well-known $\frac{1}{r}$-net theorem by Haussler and Welzl [20], which provides a bound on the probability that our sample chosen in line 5 is a $\frac{1}{r}$-net of the weighted set system, based on the VC-dimension of this set system. We use this to bound the expected number of iterations of the main loop in kApproxCover within our analysis of the multiplicative weights update algorithm.

▶ **Theorem 23.** *Given a polygonal curve $P \in \mathbb{X}_n^d$ and $\Delta \in \mathbb{R}_+$, there exists an algorithm that computes an (α, β)-approximate solution to the Δ-coverage problem on P with $\alpha = 11$ and $\beta = O(\log k^*)$, where k^* is the minimum size of a solution to the Δ-coverage problem on P. The algorithm needs in expectation $\widetilde{O}((k^*)^2 n + k^* n^3)$ time and $\widetilde{O}((k^*)n + n^3)$ space.*

In the full version [7], we show improved bounds for c-packed curves. The only modification to the algorithm is a more careful generation of the triples that generate the candidate set.

▶ **Theorem 24.** *Let $P \in \mathbb{X}_n^d$ be c-packed and $\Delta \in \mathbb{R}_+$. Let k^* be the minimum size of a solution to the Δ-coverage problem on P. There exists an algorithm that outputs an $(11, O(\log(k^*)))$-approximate solution. The algorithm needs*

1. *$\tilde{O}((k^*)^2 n + nc^2 k^*)$ expected time and $\tilde{O}(k^* n + nc^2)$ space in $\mathbb{R}^2$,*
2. *$\tilde{O}((k^*)^2 n + nc^2 k^* + n^2)$ expected time and $\tilde{O}(k^* n + nc^2)$ space in $\mathbb{R}^d$.*

In the full version [7], we also show that the property of the feasible sets as testified by Lemma 20 can be exploited to implicitly update the weights of a much larger set of candidates chosen from a uniform grid in the candidate space, thereby circumventing the explicit computation of the candidates. This improves the overall dependency on the complexity of the input curve in the running time, when compared to the previous algorithm – at the cost of a logarithmic factor of the relative arc-length of the curve.

▶ **Theorem 25.** *Let $P \in \mathbb{X}_n^d$ and $\Delta \in \mathbb{R}_+$. Let k^* be the minimum size of a solution to the Δ-coverage problem on P. Let further $\lambda(P)$ be the arc length of the curve P. There exists an algorithm that outputs a $(12, O(\log(k^*))$-approximate solution. The algorithm needs in expectation $O(nk^{*3}(\log^4(\frac{\lambda(P)}{\Delta}) + \log^3(\frac{n}{k^*})) + n\log^2(n))$ time and $O(nk^* \log(\frac{n\lambda(P)}{\Delta k^*}))$ space.*

5 Conclusions

With the algorithm variants presented in this paper, we can find bicriteria-approximate solutions to the Δ-coverage problem on a polygonal curve P. The new algorithms improve upon previously known algorithms for the Δ-coverage problem both in terms of known running time and space requirement bounds [2], as well as approximation factors. To the best of our knowledge, our candidate generation leads to the first strongly polynomial algorithm for subtrajectory clustering under the continuous Fréchet distance that does not depend on the relative arclength λ/Δ of the input curve or the spread of the coordinates. The running time is at most cubic in n, the number of vertices of the input curve (Theorem 23). In practice, we expect this to be lower as testified by our analysis for c-packed curves (Theorem 24). The work of Gudmundsson et. al. [17] suggest that in practice most curves are c-packed for a c that is considerably smaller than the complexity of the curve. However, it remains to be seen if this also holds for the typically long curves which appear as input in the subtrajectory clustering setting. We also present a variant of the algorithm with implicit weight updates which achieves a linear dependency on n (Theorem 25) and this holds in general, without any c-packedness assumption on the input.

There are several avenues for future research. We mention some of them here. An interesting question that remains open for now is whether the implicit weight update can be performed directly on the candidate set (Definition 18). For this, we need to develop a dynamic data structure that can maintain the distribution on this candidate set to perform updates with rectangles and to sample from it. Another future research direction is to improve the dependency of the approximation factor on the parameter that controls the complexity of the input curves. Currently, the dependency is linear, and we did not try to improve it, since our focus was on clustering with line segments. Another interesting question is, how the low complexity center curves obtained by our algorithm can be best connected to center curves of higher complexity or even a geometric graph while retaining the Δ-covering.

References

1 Pankaj K. Agarwal, Kyle Fox, Kamesh Munagala, Abhinandan Nath, Jiangwei Pan, and Erin Taylor. Subtrajectory clustering: Models and algorithms. In *Proceedings of the 37th ACM SIGMOD-SIGACT-SIGAI Symposium on Principles of Database Systems*, PODS '18, pages 75–87, New York, NY, USA, 2018. Association for Computing Machinery. doi:10.1145/3196959.3196972.

2 Hugo A. Akitaya, Frederik Brüning, Erin Chambers, and Anne Driemel. Subtrajectory clustering: Finding set covers for set systems of subcurves, 2021. doi:10.48550/ARXIV.2103.06040.

3 Helmut Alt and Michael Godau. Computing the Fréchet distance between two polygonal curves. *Int. J. Comput. Geom. Appl.*, 5:75–91, 1995. doi:10.1142/S0218195995000064.

4 Martin Anthony and Peter L. Bartlett. *Neural Network Learning: Theoretical Foundations*. Cambridge University Press, 1999. doi:10.1017/CBO9780511624216.

5 Sanjeev Arora, Elad Hazan, and Satyen Kale. The multiplicative weights update method: a meta-algorithm and applications. *Theory of Computing*, 8(1):121–164, 2012. doi:10.4086/toc.2012.v008a006.

6 Hervé Brönnimann and Michael T Goodrich. Almost optimal set covers in finite VC-dimension. *Discrete & Computational Geometry*, 14(4):463–479, 1995. doi:10.1007/BF02570718.

7 Frederik Brüning, Jacobus Conradi, and Anne Driemel. Faster approximate covering of subcurves under the fréchet distance, 2022. doi:10.48550/ARXIV.2204.09949.

8 Kevin Buchin, Maike Buchin, David Duran, Brittany Terese Fasy, Roel Jacobs, Vera Sacristan, Rodrigo I. Silveira, Frank Staals, and Carola Wenk. Clustering trajectories for map construction. In *Proceedings of the 25th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems*, SIGSPATIAL '17, New York, NY, USA, 2017. Association for Computing Machinery. doi:10.1145/3139958.3139964.

9 Kevin Buchin, Maike Buchin, Joachim Gudmundsson, Jorren Hendriks, Erfan Hosseini Sereshgi, Vera Sacristán, Rodrigo I. Silveira, Jorrick Sleijster, Frank Staals, and Carola Wenk. Improved map construction using subtrajectory clustering. In *LocalRec'20: Proceedings of the 4th ACM SIGSPATIAL Workshop on Location-Based Recommendations, Geosocial Networks, and Geoadvertising, LocalRec@SIGSPATIAL 2020, November 3, 2020, Seattle, WA, USA*, pages 5:1–5:4, 2020. doi:10.1145/3423334.3431451.

10 Kevin Buchin, Maike Buchin, Joachim Gudmundsson, Maarten Löffler, and Jun Luo. Detecting commuting patterns by clustering subtrajectories. *Int. J. Comput. Geom. Appl.*, 21(3):253–282, 2011. doi:10.1142/S0218195911003652.

11 Maike Buchin, Bernhard Kilgus, and Andrea Kölzsch. Group diagrams for representing trajectories. *International Journal of Geographical Information Science*, 34(12):2401–2433, 2020. doi:10.1080/13658816.2019.1684498.

12 Maike Buchin and Carola Wenk. Inferring movement patterns from geometric similarity. *J. Spatial Inf. Sci.*, 21(1):63–69, 2020. doi:10.5311/JOSIS.2020.21.724.

13 Kenneth L Clarkson. Las vegas algorithms for linear and integer programming when the dimension is small. *Journal of the ACM (JACM)*, 42(2):488–499, 1995. doi:10.1145/201019.201036.

14 Mark de Berg, Atlas F. Cook, and Joachim Gudmundsson. Fast Fréchet queries. *Computational Geometry*, 46(6):747–755, 2013. doi:10.1016/j.comgeo.2012.11.006.

15 Andrew T Duchowski. A breadth-first survey of eye-tracking applications. *Behavior Research Methods, Instruments, & Computers*, 34(4):455–470, 2002. doi:10.3758/BF03195475.

16 Paul W. Goldberg and Mark R. Jerrum. Bounding the Vapnik-Chervonenkis dimension of concept classes parameterized by real numbers. *Machine Learning*, 18:131–148, 1995. doi:10.1007/BF00993408.

17 Joachim Gudmundsson, Yuan Sha, and Sampson Wong. Approximating the packedness of polygonal curves, 2020. doi:10.48550/ARXIV.2009.07789.

18 Joachim Gudmundsson and Nacho Valladares. A GPU approach to subtrajectory clustering using the fréchet distance. *IEEE Trans. Parallel Distributed Syst.*, 26(4):924–937, 2015. doi:10.1109/TPDS.2014.2317713.

19 Joachim Gudmundsson and Sampson Wong. Cubic upper and lower bounds for subtrajectory clustering under the continuous fréchet distance, 2021. doi:10.48550/ARXIV.2110.15554.

20 David Haussler and Emo Welzl. Epsilon-nets and simplex range queries. *Discrete & Computational Geometry*, 2(2):127–151, 1987. doi:10.1007/BF02187876.

21 Kenneth Holmqvist, Marcus Nyström, Richard Andersson, Richard Dewhurst, Halszka Jarodzka, and Joost Van de Weijer. *Eye tracking: A comprehensive guide to methods and measures*. OUP Oxford, 2011. URL: https://global.oup.com/academic/product/eye-tracking-9780199697083.

22 Catalin Ionescu, Dragos Papava, Vlad Olaru, and Cristian Sminchisescu. Human3.6m: Large scale datasets and predictive methods for 3d human sensing in natural environments. *IEEE transactions on pattern analysis and machine intelligence*, 36(7):1325–1339, 2013. doi:10.1109/TPAMI.2013.248.

23 Jae-Gil Lee, Jiawei Han, and Kyu-Young Whang. Trajectory clustering: a partition-and-group framework. In *Proceedings of the ACM SIGMOD International Conference on Management of Data, Beijing, China, June 12-14, 2007*, pages 593–604, 2007. doi:10.1145/1247480.1247546.

24 Sen Qiao, Y. Wang, and J. Li. Real-time human gesture grading based on OpenPose. *2017 10th International Congress on Image and Signal Processing, BioMedical Engineering and Informatics (CISP-BMEI)*, pages 1–6, 2017. doi:10.1109/CISP-BMEI.2017.8301910.

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

26 Sheng Wang, Zhifeng Bao, J Shane Culpepper, and Gao Cong. A survey on trajectory data management, analytics, and learning. *ACM Computing Surveys (CSUR)*, 54(2):1–36, 2021. doi:10.1145/3440207.

27 Guan Yuan, Penghui Sun, Jie Zhao, Daxing Li, and Canwei Wang. A review of moving object trajectory clustering algorithms. *Artificial Intelligence Review*, 47(1):123–144, 2017. doi:10.1007/s10462-016-9477-7.