Subtrajectory Clustering and Coverage Maximization in Cubic Time, or Better

Jacobus Conradi
University of Bonn, Germany

Anne Driemel
University of Bonn, Germany

Abstract

Many application areas collect unstructured trajectory data. In subtrajectory clustering, one is interested to find patterns in this data using a hybrid combination of segmentation and clustering. We analyze two variants of this problem based on the well-known SetCover and CoverageMaximization problems. In both variants the set system is induced by metric balls under the Fréchet distance centered at polygonal curves. Our algorithms focus on improving the running time of the update step of the generic greedy algorithm by means of a careful combination of sweeps through a candidate space. In the first variant, we are given a polygonal curve P of complexity n, distance threshold Δ and complexity bound ℓ and the goal is to identify a minimum-size set of center curves $\mathcal{C}$, where each center curve is of complexity at most ℓ and every point p on P is covered. A point p on P is covered if it is part of a subtrajectory π_p of P such that there is a center $c \in \mathcal{C}$ whose Fréchet distance to π_p is at most Δ. We present an approximation algorithm for this problem with a running time of $\mathcal{O}((n^2\ell + \sqrt{k_\Delta}n^{5/2})\log^2 n)$, where k_Δ is the size of an optimal solution. The algorithm gives a bicriterial approximation guarantee that relaxes the Fréchet distance threshold by a constant factor and the size of the solution by a factor of $\mathcal{O}(\log n)$. The second problem variant asks for the maximum fraction of the input curve P that can be covered using k center curves, where $k \leq n$ is a parameter to the algorithm. For the second problem variant, our techniques lead to an algorithm with a running time of $\mathcal{O}((k+\ell)n^2\log^2 n)$ and similar approximation guarantees. Note that in both algorithms $k, k_\Delta \in O(n)$ and hence the running time is cubic, or better if $k \ll n$.

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.2025.12

Related Version *Full Version*: `https://arxiv.org/abs/2504.17381`

Funding *Jacobus Conradi*: Partially funded by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) – 313421352 (FOR 2535 Anticipating Human Behavior) and the iBehave Network: Sponsored by the Ministry of Culture and Science of the State of North Rhine-Westphalia. Affiliated with Lamarr Institute for Machine Learning and Artificial Intelligence.
Anne Driemel: Affiliated with Lamarr Institute for Machine Learning and Artificial Intelligence.

1 Introduction

Trajectory analysis is a broad field ranging from gait analysis of full-body motion trajectories [16, 20, 13], via traffic analysis [21] and epidemiological hotspot detection [25] to Lagrangian analysis of particle simulations [24]. Using the current availability of cheap tracking technology, vast amounts of unstructured spatio-temporal data are collected. Clustering is a fundamental problem in this area and can be formulated under various similarity metrics, among them the Fréchet distance. However, the unstructured nature of trajectory data poses the additional challenge that the data first needs to be segmented before the trajectory segments, which we call subtrajectories, can be clustered in a metric space. Subtrajectory clustering aims to find clusters of subtrajectories that represent complex patterns that reoccur along the trajectories [1, 8, 17], such as commuting patterns in traffic data.

We study two variants of a problem posed by Akitaya, Brüning, Chambers, and Driemel [3], which are based on the well-known set cover and coverage maximization problems. Both rely on the definition of a geometric set system built using metric balls under the Fréchet distance. Given a polygonal curve, we are asked to produce a set of "center" curves that either

1. has minimum cardinality and covers the entirety of the input curve, or
2. covers as much as possible of the input curve under cardinality constraints.

In either setting a point p of the input-curve is considered as "covered", if there is a subcurve π_p of the input curve that contains p and has small Fréchet distance to some center curve. Note that a point p may be covered by multiple center curves. For the precise formulation refer to Section 2. This formulation extends immediately to a set of input curves – instead of one – where one is now tasked to cover the entirety (respectively as much as possible) of all points on all input curves combined.

In this paper we study the continuous variant of the problem where we need to cover the continuous set of points given by the input polygonal curve. It is instructive, however, to first consider a discrete variant of the set cover instance as follows. Given a discrete point sequence $A = a_1, \dots, a_n$, consider the set of contiguous subsequences of the form $A_{ij} = a_i, \dots, a_j$. We say a_k is contained in the set defined by A_{ij} if there exist $i' \leq k \leq j'$ such that $A_{i'j'}$ is similar to A_{ij}. Note that a full specification of this set system has size $\Theta(n^3)$ in the worst case. This phenomenon is shared with the continuous setting where all known discretizations lead to a cubic-size set system. In this paper, we examine the question of whether we can obtain subcubic-time algorithms by internally representing the set system implicitly.

2 Preliminaries

An **edge** is a map $e : [0,1] \to \mathbb{R}^d$ defined by two points $p, q \in \mathbb{R}^d$ as the linear interpolation $t \mapsto p + t(q-p)$. A **polygonal curve** is a map $P : [0,1] \to \mathbb{R}^d$ defined by an ordered set $(p_1, \dots, p_n) \subset \mathbb{R}^d$ as the result of concatenating the edges $(e_1, \dots, e_{n-1})$, where e_i is the edge from p_i to p_{i+1}. Any point $P(t)$ is said to lie on the edge e_i if t lies in the preimage of e_i. Any point $P(t)$ lies on at most two edges, exactly if it defines the endpoint of edge e_i which coincides with the start point of e_{i+1}. The number of points n defining the polygonal curve P is its **complexity**, and we denote it by $|P|$. The set of all polygonal curves in $\mathbb{R}^d$ of complexity at most ℓ we denote by $\mathbb{X}^d_\ell$. For any polygonal curve P and values $0 \leq s \leq t \leq 1$ we define $P[s,t]$ to be the subcurve of P from s to t, which itself is a polygonal curve of complexity at most $|P|$. For two polygonal curves P and Q we define the **continuous Fréchet distance** to be $\mathrm{d}_\mathcal{F}(P,Q) = \inf_{f,g} \max_{t\in[0,1]} \|P(f(t)) - Q(g(t))\|$ where f and g range over all non-decreasing surjective functions from $[0,1]$ to $[0,1]$.

▶ **Definition 1** (Δ-coverage [3]). *Let P be a polygonal curve in $\mathbb{R}^d$, and let $\Delta \in \mathbb{R}_{>0}$ and $\ell \in \mathbb{N}_{\geq 2}$ be given. For any $\mathcal{Q} \subset \mathbb{X}^d_\ell$, define the Δ-coverage of $\mathcal{Q}$ on P (refer to Figure 1) as*

$$\mathrm{Cov}_P(\mathcal{Q}, \Delta) = \bigcup_{Q \in \mathcal{Q}} \left(\bigcup_{0 \leq s \leq t \leq 1,\, \mathrm{d}_\mathcal{F}(P[s,t],Q) \leq \Delta} [s,t] \right) \subset [0,1].$$

For a curve $Q \in \mathbb{X}^d_\ell$ we denote the Δ-coverage $\mathrm{Cov}_P(\{Q\}, \Delta)$ by $\mathrm{Cov}_P(Q, \Delta)$.

Problem 1: Subtrajectory Covering (SC). Let P be a polygonal curve in $\mathbb{R}^d$, and let $\Delta \in \mathbb{R}_{>0}$ and $\ell \in \mathbb{N}_{\geq 2}$ be given. The task is to identify a set $\mathcal{Q}$ of curves in $\mathbb{X}^d_\ell$ minimizing $|\mathcal{Q}|$ such that $\mathrm{Cov}_P(\mathcal{Q}, \Delta) = [0,1]$.

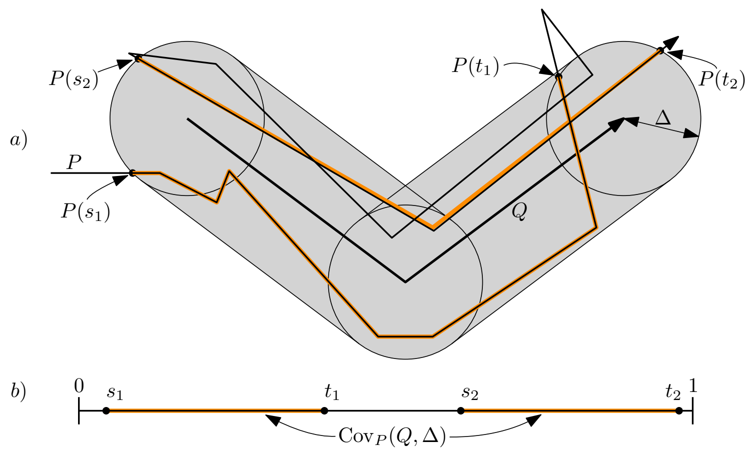

Figure 1 *a*): Example of all points on P that lie on subcurve of P that have Fréchet distance at most Δ to a curve Q of complexity 3. *b*): The set $\mathrm{Cov}_P(Q,\Delta) \subset [0,1]$.

▶ **Definition 2** (Bicriteria approximation for SC)**.** *Let P be a polygonal curve in $\mathbb{R}^d$, and let $\Delta \in \mathbb{R}_{>0}$ and $\ell \in \mathbb{N}_{\geq 2}$ be given. Let $C^* \subset \mathbb{X}^d_\ell$ be a set of minimum cardinality such that $\mathrm{Cov}_P(C^*,\Delta) = [0,1]$. Let $\alpha,\beta \geq 1$ be given. An algorithm that outputs a set $C \subset \mathbb{X}^d_\ell$ of size $\alpha|C^*|$ such that $\mathrm{Cov}_P(C,\beta\Delta) = [0,1]$ is called an (α,β)-approximation for SC.*

Problem 2: Subtrajectory Coverage Maximization (SCM). Let P be a polygonal curve in $\mathbb{R}^d$, and let $\Delta \in \mathbb{R}_{>0}$, $\ell \in \mathbb{N}_{\geq 2}$ and $k \in \mathbb{N}_{\geq 1}$ be given. The task is to identify a set $\mathcal{Q}$ of k curves in $\mathbb{X}^d_\ell$ maximizing the Lebesgue measure $\|\mathrm{Cov}_P(\mathcal{Q},\Delta)\|$.

▶ **Definition 3** (Bicriteria approximation for SCM)**.** *Let P be a polygonal curve in $\mathbb{R}^d$, and let $\Delta \in \mathbb{R}_{>0}$ and $\ell \in \mathbb{N}_{\geq 2}$ be given. Let $C^* \subset \mathbb{X}^d_\ell$ be a set of size k such that $\|\mathrm{Cov}_P(C^*,\Delta)\|$ is maximum. Let $\alpha \geq 0$ and $\beta \geq 1$ be given. An algorithm that outputs a set $C \subset \mathbb{X}^d_\ell$ of size k such that $\|\mathrm{Cov}_P(C,\beta\Delta)\| \geq \alpha\,\|\mathrm{Cov}_P(C^*,\Delta)\|$ is called an (α,β)-approximation for SCM.*

2.1 Prior work on Subtrajectory Covering and Coverage Maximization

We first discuss prior work on the SC problem. The SC problem was introduced by Akitaya, Brüning, Chambers, and Driemel [3]. Their results were later improved by Brüning, Conradi, and Driemel [4]. Both works identified a curve S that approximates the input P, such that any solution to the problem induces an approximate solution of similar size consisting of only subcurves of S. This set of subcurves defines a set system which turns out to have low VC-dimension. This enables randomized set cover algorithms with improved approximation factors and expected running time in $\mathcal{O}(n(k_\Delta)^3(\log^4(\Lambda/\Delta) + \log^3(n/k_\Delta)) + n\log^2 n)$, where Λ corresponds to the arc length of the curve P. While the complexity of the main algorithm in [3] depends on the relative arclength of the input curve, the latter [4] also identified a set of points on S that define $\mathcal{O}(n^3)$ subcurves of complexity 2 which induce the set-system of the SetCover instance resulting in an expected running time of $\tilde{\mathcal{O}}(k_\Delta n^3)$, where $\tilde{\mathcal{O}}(\cdot)$ hides polylogarithmic factors. The approximation quality of both approaches depends on ℓ, k_Δ as well as on the VC-dimension of the set system. A drawback of these algorithms is that they generate cluster centers that are line segments only and that the algorithm is randomized with large constants in the approximation factors. Conradi and Driemel [9] took a different approach by focusing on the greedy set cover algorithm, which successively adds the candidate curve maximizing the added coverage to a partial solution until the entirety of $[0,1]$ is covered. Their algorithm computes $\mathcal{O}(n^3\ell)$ candidate curves that have complexity up to ℓ instead of 2. Applying the greedy algorithm, this results in a deterministic $(\mathcal{O}(\log n), 11)$-approximation algorithm with running time $\tilde{\mathcal{O}}(k_\Delta n^4\ell + n^4\ell^2)$. Recently, van der Hoog, van der Horst and Ophelders showed [23] that the cardinality of the candidate curve set can be further reduced to a size of $\mathcal{O}(n^2\log n)$. They improve the quality of the approximating curve S resulting in a $(\mathcal{O}(\log n), 4)$-approximation with running time in $\tilde{\mathcal{O}}(k_\Delta n^3)$.

As for the SCM problem, by standard sub-modular function maximization arguments [19, 15], the greedy algorithm used in all these approaches [3, 4, 9, 23] yields an $(\mathcal{O}(1), \mathcal{O}(1))$-approximation. Using this reduction, the candidate curve set identified in [23] yields a $(\mathcal{O}(1), 4)$-approximation to the SCM problem with a running time of $\tilde{\mathcal{O}}(kn^3)$.

2.2 Structure and contributions

In Section 3 we describe how to obtain a set of $\tilde{\mathcal{O}}(n^2)$ candidate center curves $\mathcal{C}$ based on known transformations [23]. Unlike the recent line of work discussed above [3, 4, 9, 23], we do not improve any further on the size of the set $\mathcal{C}$. Instead, our focus lies on the update step of the greedy set cover algorithm. In this step, we have to find the element $c \in \mathcal{C}$ that maximizes the coverage that is added to a partial solution. In Section 4 we describe our first contribution: a partition of the set $\mathcal{C}$ into few ordered lists, called **sweep-sequences**, which we each process via a sweep algorithm computing their coverage reusing prior computations. Unfortunately, the coverage does not immediately allow efficient maintenance along a sweep-sequence. Here, we introduce our second contribution: a candidate-specific approximation of the coverage, called the **proxy coverage**. In Section 5, we describe how sweep-sequences allow efficient maintenance of the proxy coverage and with Theorem 26 describe the culmination of our techniques enabling our algorithm: for a weighted set $A \subset [0,1]$ of points we are able to compute the total weight of points that lie inside the proxy coverage of every element in the sweep-sequence in logarithmic time per element.

In Section 6 we present two $(\mathcal{O}(\log n), 4)$-approximation algorithms for Subtrajectory Covering. The algorithms discretize the ground-set $[0,1]$ by the arrangement (of size $\tilde{\mathcal{O}}(n^3)$) of the intervals representing the coverage of each element in $\mathcal{C}$. Computing the arrangement together with standard greedy set cover techniques and the subroutine from Theorem 26 results in a first approximation algorithm for SC with running time in $\tilde{\mathcal{O}}(|\mathcal{C}|n + k_\Delta|\mathcal{C}|) = \tilde{\mathcal{O}}(n^3)$ (see Corollary 30). Our third contribution is a sampling technique improving the running time: we identify a representative subset (comparable to a sampling) of size roughly $\mathcal{O}(\sqrt{k_\Delta}n^{3/2})$ of the arrangement. This enables a faster algorithm by first covering the representative subset and with it almost the entire arrangement in an initial pass. The remaining $\mathcal{O}(\sqrt{k_\Delta}n^{5/2})$ elements of the arrangement are then identified and covered resulting in the following theorem.

▶ **Theorem 4.** *There is a $(96\ln(n) + 128, 4)$-approximation for SC. Given a polygonal curve P of complexity n, $\Delta > 0$ and $\ell \leq n$, its running time is in $\mathcal{O}\left(\left(n^2\ell + \sqrt{k_\Delta}n^{\frac{5}{2}}\right)\log^2 n\right)$, where k_Δ is the size of the smallest subset $C^* \subset \mathbb{X}_\ell^d$ such that $\mathrm{Cov}_P(C^*, \Delta) = [0,1]$.*

As trivially $k_\Delta \leq \lceil \frac{n}{\ell} \rceil$, Theorem 4 yields an algorithm with (near-)cubic running time. Further, in the case that $k_\Delta \in \mathcal{O}(n^{1-\varepsilon})$, it yields an algorithm with subcubic running time. This compares favorably to the best known algorithm with running time $\tilde{\mathcal{O}}(k_\Delta n^3)$ [23].

In Section 7, we sketch how our techniques can be applied to obtain faster algorithms for the SCM problem which compares favorably to the best known algorithm with running time $\tilde{\mathcal{O}}(kn^3)$ [23]. All omitted proofs and discussions can be found in the full version.

▶ **Theorem 5.** *Let $\varepsilon \in \left(0, \frac{1}{5}\right]$. There is a $(\frac{e-1}{16e}, 4+\varepsilon)$-approximation algorithm for SCM, where e is the base of the natural logarithm. Given a polygonal curve P of complexity n, $\Delta > 0$, $\ell \leq n$ and $k > 0$, its running time is in $\mathcal{O}((k+\ell)n^2\varepsilon^{-2}\log^2 n \log \varepsilon^{-1})$.*

2.3 Related work

Prior to the line of work on the SC and SCM problem discussed in Section 2.1, Buchin et al. [7] presented one of the earlier works on subtrajectory clustering for both the discrete and continuous Fréchet distance. Their work focuses on finding the largest subtrajectory cluster, where different variants of "largest" are considered. They present hardness results for $(2-\varepsilon)$-approximations for any ε and a matching polynomial time 2-approximation algorithm. Gudmundsson and Wong [12] present a cubic lower bound conditioned on SETH for the same problem and show that this lower bound is tight. In subsequent experimental work [11, 5, 6, 8] the formulation from [7] was studied, but these works do not offer any theoretical guarantees.

Agarwal et al. [1] base their formulation of the subtrajectory clustering problem on FacilityLocation. In this formulation there is an opening cost associated with every center curve, a cost for every point on the input that is assigned to a cluster, and a cost for points that are not assigned. They present conditional NP-hardness results and an $O(\log^2 n)$-approximation for well-behaved classes of curves under the discrete Fréchet distance.

3 Set system discretization

In this section we show that, given a curve P of complexity n, there is a curve S of complexity n and a set $\mathcal{C}_S(P)$ of at most $\mathcal{O}(n^2 \log n)$ subcurves of S, each of complexity at most ℓ, such that for any set of curves $C \subset \mathbb{X}^d_\ell$ there is a subset $\hat{C} \subset \mathcal{C}_S(P)$ of size $\mathcal{O}(|C|)$ such that $\mathrm{Cov}_P(C, \Delta) \subset \mathrm{Cov}_P(\hat{C}, 4\Delta)$. This result is known and follows the line of research in [4, 9, 23]. All of these approaches compute a "maximally simplified" version S of P, see also [10]. Our definition of a simplification follows the notion of *pathlet-preserving simplifications* from [23], as it yields the best constants.

▶ **Definition 6.** *Let a polygonal curve P, $\Delta \geq 0$, and $\ell \geq 1$ be given. A curve S is a simplification of P if $\mathrm{d}_{\mathcal{F}}(S, P) \leq 2\Delta$, and for any curve $Q \in \mathbb{X}^d_\ell$ and $P[s,t]$ with $\mathrm{d}_{\mathcal{F}}(Q, P[s,t]) \leq \Delta$ there is a subcurve $S[s',t']$ with $\mathrm{d}_{\mathcal{F}}(P[s,t], S[s',t']) \leq 2\Delta$ and $|S[s',t']| \leq \ell + 2$.*

▶ **Theorem 7** ([23]). *For any curve P of complexity n a simplification S of P of complexity $\mathcal{O}(n)$ can be computed in $\mathcal{O}(n \log n)$ time.*

By definition of the simplification S of P and the triangle inequality for any curve $\pi \in \mathbb{X}^d_\ell$ there is a subcurve $\hat{\pi}$ of S of complexity at most $\ell + 2$ such that $\mathrm{Cov}_P(\pi, \Delta) \subset \mathrm{Cov}_P(\hat{\pi}, 4\Delta)$.

▶ **Definition 8** (Free space diagram). *Let S and P be two polygonal curves parametrized over $[0,1]$. The free space diagram of S and P is defined as the joint parametric space $[0,1]^2$ together with a not necessarily uniform grid, where each vertical grid line corresponds to a vertex of P and each horizontal grid line to a vertex of S. The Δ-**free space** (refer to Figure 2) of S and P is defined as*

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

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

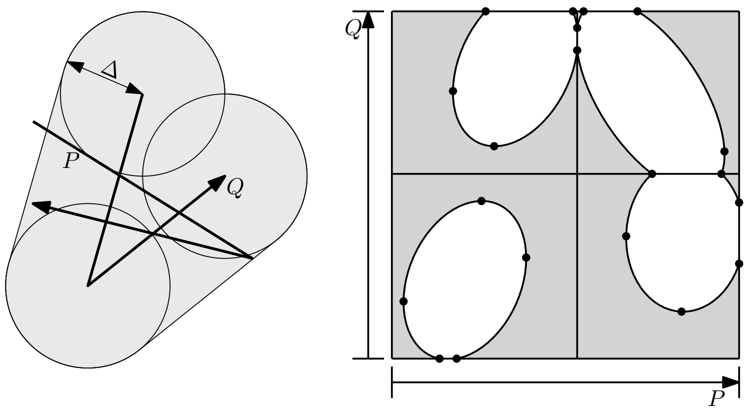

Figure 2 Illustration of the Δ-free space of two curves P and Q as well as their extremal points.

As was noted by Alt and Godau [2], the Fréchet distance of S and P is at most Δ if and only if there is a monotone (in both x- and y-direction) path from $(0,0)$ to $(1,1)$ in Δ-free space of S and P. They further observed that the free space inside any cell is described by an ellipse intersected with the cell and thus is convex and has constant complexity. Observe that two subcurves $P[a,c]$ and $S[b,d]$ have Fréchet distance at most Δ if and only if there is a monotone (in both x- and y-direction) path from (a,b) to (c,d) in Δ-free space of S and P.

▶ **Definition 9** (Extremal points [4]). *As the Δ-free space of two curves S and P is convex in any cell C, the set of points of $\mathcal{D}_\Delta(S,P)$ minimizing the x-coordinate in C are described by either a single point or a vertical line segment. We call either the single point or the start and end point of the line segment the leftmost points of $\mathcal{D}_\Delta(S,P)$ in C. Similarly C has at most two bottommost, rightmost and topmost points. The set of all these at most eight points of $\mathcal{D}_\Delta(S,P)$ in every cell defines the set of extremal points of $\mathcal{D}_\Delta(S,P)$.*

Similar to [23], we define specific subcurves via the extremal points of the 4Δ-free space.

▶ **Definition 10** ((I)-, (II)- and (III)-subcurves). *Let P be a polygonal curve and let S be a simplification of P. Let Δ and ℓ be given. Define the following three types of subcurves of S via the (sorted) set $\mathcal{E}(S,P)$ of y-coordinates of extremal points in $\mathcal{D}_\Delta(S,P)$. For every edge e let $\mathcal{E}(e,P)$ be the subset of $\mathcal{E}(S,P)$ corresponding to the extremal points in $\mathcal{D}_\Delta(e,P)$ (refer to Figure 3).*

(I): *A (I)-subcurve of S is a subcurve of S that starts at some ith vertex of S and ends at the $(i+j)$th vertex for $j \in \{2^m \mid 0 \le m \le \lfloor \log_2(\ell) \rfloor\}$.*

(II): *A (II)-subcurve of S is either a subcurve $e[0,\varepsilon_i]$ or $e[\varepsilon_i,1]$ of an edge e of S defined by a vertex of e and a value $\varepsilon_i \in \mathcal{E}(e,P)$ or its reversal $\mathrm{rev}(e[0,\varepsilon_i])$ or $\mathrm{rev}(e[\varepsilon_i,1])$.*

(III): *A (III)-subcurve of S is either a subcurve $e[\varepsilon_i,\varepsilon_{i+j}]$ of an edge e of S defined by two values $\varepsilon_i,\varepsilon_{i+j} \in \mathcal{E}(e,P)$ such that $j \in \{2^m \mid 0 \le m \le \lfloor \log_2(|\mathcal{E}(e,P)|) \rfloor\}$, or its reversal $\mathrm{rev}(e[\varepsilon_i,\varepsilon_{i+j}])$.*

The set containing all Type (I)-, (II)- and (III)-subcurves which we denote by $\mathcal{C}_S(P)$ describes the set of candidate curves similar to the candidate set from [23].

▶ **Theorem 11** (adapted from [23]). *Let P be a polygonal curve, and let S be a simplification of P. Then for any curve $\pi \in \mathbb{X}^d_\ell$ there is a set $S_\pi \subset \mathcal{C}_S(P)$ of size at most 8 such that*

$$\mathrm{Cov}_P(\pi,\Delta) \subset \bigcup_{s\in S_\pi} \mathrm{Cov}_P(s,4\Delta).$$

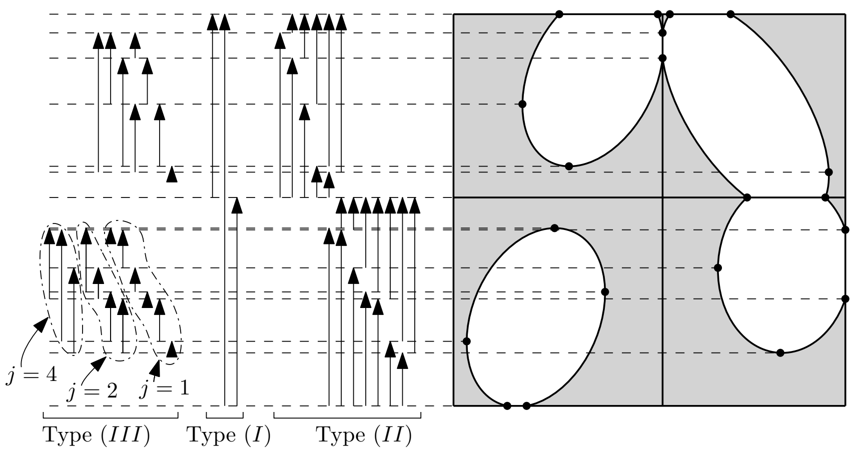

Figure 3 Illustration of all Type (I)-, (II)- and (III)-subcurves of Q (as vertical lines) that are not reversals, induced by the 4Δ-free space of Q and P from Figure 2. Further denoted are the values j, which induced the set of Type (III)-subcurves on the first edge of Q.

4 Sweep-sequences and the proxy coverage

In this section we define an ordering of Type (II) and Type (III) subcurves in the set $\mathcal{C}_S(P)$ and introduce an approximate Δ-coverage called the **proxy coverage**. Later on, we show that a coarse representation of the proxy coverage can be maintained along the ordering.

Throughout this section we are given a polygonal curve P of complexity n, values $\Delta > 0$ and $\ell \leq n$, as well as a simplification S of P. As $|\mathcal{E}(e, P)| = \mathcal{O}(n)$ for any edge e, there are $\mathcal{O}(n \log \ell)$ Type (I)-, $\mathcal{O}(n^2)$ Type (II)- and $\mathcal{O}(n^2 \log n)$ Type (III)-subcurves in $\mathcal{C}_S(P)$.

▶ **Definition 12** (Sweep-sequence). *Let $E = (e_1, \dots)$ be a sorted list of values in $\mathbb{R}$. We say $\mathfrak{s}$ is a sweep-sequence of E if $\mathfrak{s}$ is a list of tuples of E such that either for all consecutive tuples (e_a, e_b) and (e_c, e_d) it holds that $a \leq b$, $c \leq d$, $0 \leq a - c \leq 1$ and $0 \leq b - d \leq 1$ or for all consecutive tuples it holds that $a \geq b$, $c \geq d$, $0 \leq c - a \leq 1$ and $0 \leq d - b \leq 1$.*

We are able to construct few sweep-sequences that already capture the entirety of $\mathcal{C}_S(P)$.

▶ **Lemma 13.** *Let e be an edge of S. In time $\mathcal{O}(n \log n)$, one can compute a set $\mathfrak{S}_e$ of $\mathcal{O}(\log n)$ sweep-sequences of $\mathcal{E}(e, P)$, each of length $\mathcal{O}(n)$, such that for any Type (II)- or Type (III)-subcurve $\pi \in \mathcal{C}_S(P)$ on the edge e there is a tuple $(\varepsilon_i, \varepsilon_j)$ in one of the sweep-sequences in $\mathfrak{S}_e$ such that $\pi = e[\varepsilon_i, \varepsilon_j]$ if $\varepsilon_i \leq \varepsilon_j$ (refer to Figure 4), and $\pi = \mathrm{rev}(e[\varepsilon_j, \varepsilon_i])$ if $\varepsilon_i > \varepsilon_j$. Further, for the first and last pair (e_a, e_b) in every sweep-sequence it holds that $a = b$.*

We focus our analysis on sweep-sequences of $\mathcal{E}(e, P)$ where for every tuple $(\varepsilon_a, \varepsilon_b)$ it holds that $a \leq b$ (refer to Figure 4). This analysis carries over to all sweep-sequences of $\mathcal{E}(e, P)$ by setting $e \leftarrow \mathrm{rev}(e)$, and thus to all sweep-sequences in $\mathfrak{S}_e$.

We now turn to defining the proxy coverage on P which approximates the 4Δ-coverage.

▶ **Definition 14.** *Let e be an edge of S. Define $l_i(\cdot)$ to be the function mapping any y to the x-coordinate of the leftmost point of the ith cell in $\mathcal{D}_{4\Delta}(e, P)$ at height y. If this point does not exist, $l_i(y) = \infty$. Similarly define $r_j(y)$ to be the x-coordinate of the rightmost point at height y, and ∞ otherwise.*

We assume that for every edge e and every $i < n$ the functions $l_i(\cdot)$ and $r_i(\cdot)$ of the free space $\mathcal{D}_\Delta(e, P)$ can be evaluated in $\mathcal{O}(1)$ time.

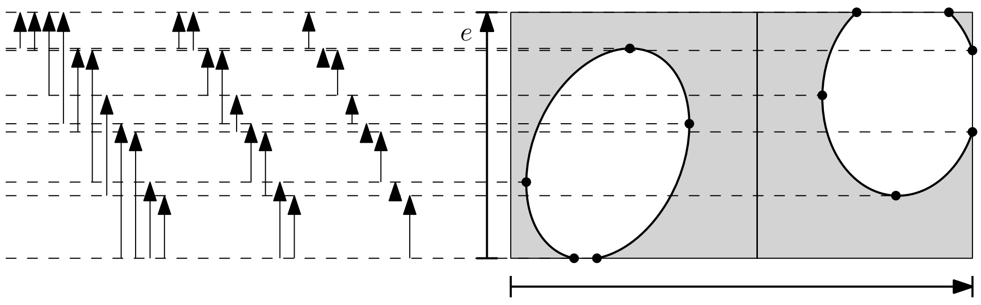

Figure 4 Illustration of three of the eight sweep-sequences in $\mathfrak{S}_e$ of the edge e that are constructed for Type (III)-subcurves. From left to right one for $j = 4$, one for $j = 2$ and one for $j = 1$.

▶ **Definition 15** (Combinatorial representation). *Let e be an edge of S and let $0 \leq s \leq t \leq 1$ be given. The combinatorial representation $\mathcal{R}(e[s,t])$ of the 4Δ-coverage $\mathrm{Cov}_P(e[s,t], 4\Delta)$ of $e[s,t]$ is the set of all inclusionwise-maximal pairs of indices (i,j) such that there are points s' and t' on the ith and jth edge of P respectively such that there is a monotone path from (s', s) to (t', t) in $\mathcal{D}_\Delta(e, P)$. An index pair (i,j) includes (i', j') if $i \leq i'$ and $j' \leq j$.*

The combinatorial representation can be partitioned into two sets, the global group $\mathcal{G}(e[s,t])$ consisting of all index-pairs $(i,j) \in \mathcal{R}(e[s,t])$ such that $i < j$ and the local group $\mathcal{L}(e[s,t])$ consisting of all index-pairs $(i,i) \in \mathcal{R}(e[s,t])$.

▶ **Observation 16.** *Let $e[s,t]$ be a subcurve of an edge e of S. Then (refer to Figure 5)*

$$\mathrm{Cov}_P(e[s,t], 4\Delta) = \bigcup_{(i,j)\in\mathcal{R}(e[s,t])} [l_i(s), r_j(t)] = \bigcup_{(i,j)\in\mathcal{L}(e[s,t])\cup\mathcal{G}(e[s,t])} [l_i(s), r_j(t)]$$

Let an edge e of S together with $0 \leq s \leq t \leq 1$ be given. We say an index i is bad for $e[s,t]$, if all topmost points in cell i of the free space of e and P are to the left of both $l_i(s)$ and $r_i(t)$, and both $l_i(s)$ and $r_i(t)$ are to the left of all bottom most points in cell i. Call this set of bad indices $\mathcal{B}(e[s,t])$. If $i \notin \mathcal{B}(e[s,t])$, i is said to be good for $e[s,t]$. Intuitively, an index is bad if the free-space inside the cell is a diagonal from the top left to the bottom right. For Lemma 21 to hold, the definition a bad index is stronger and depends on s and t.

▶ **Observation 17.** *Let i be good for $e[s,t]$ and P. If $l_i(s) \neq \infty \neq r_i(t)$ then $l_i(s) \leq r_i(t)$.*

We now turn to defining our central tool, the proxy coverage, which approximates the Δ-coverage and is easy to maintain. It is defined via a reduced combinatorial representation.

▶ **Definition 18** (Reduced global group). *Let e be an edge of S and let $0 \leq s \leq t \leq 1$ be given. Based on the global group $\mathcal{G}(e[s,t])$ we define the reduced global group $\widetilde{\mathcal{G}}(e[s,t])$ which results from $\mathcal{G}(e[s,t])$ after merging all pairs of index pairs (a,b) and (c,d) if either $a < c < b < d$, or $b = c$ and $b = c$ is good for e (see Figure 5).*

▶ **Definition 19** (Proxy coverage). *For edge e of S, $0 \leq s \leq t \leq 1$ and $1 \leq i,j < n$ define*

$$\hat{l}_{i,e[s,t]}(s) = \begin{cases} l_i(s), & \text{if } i \text{ is good for } e[s,t] \\ r_i(s), & \text{if } i \text{ is bad for } e[s,t] \end{cases} \qquad \hat{r}_{j,e[s,t]}(t) = \begin{cases} l_j(t), & \text{if } j \text{ is good for } e[s,t] \\ r_j(t), & \text{if } j \text{ is bad for } e[s,t]. \end{cases}$$

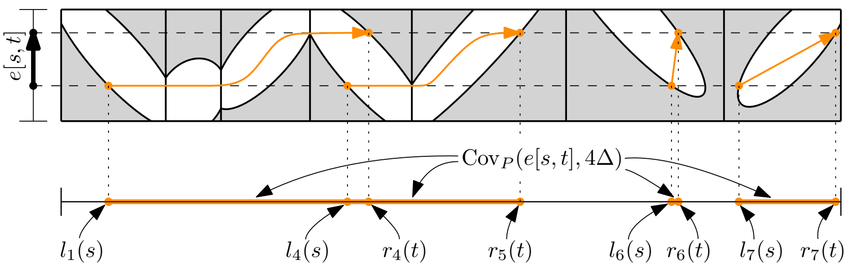

Figure 5 Illustration of the 4Δ-coverage of $e[s,t]$. The global group of $e[s,t]$ is $\{(1,4),(4,5)\}$. The reduced global group of $e[s,t]$ is $\{(1,4),(4,5)\}$ also, as 4 is bad for $e[s,t]$.

With these at hand, define the ***proxy coverage*** *of subedges of S as the union*

$$\widehat{\mathrm{Cov}}_P(e[s,t]) = \left(\bigcup_{(i,i)\in\mathcal{L}(e[s,t])\setminus\mathcal{B}(e[s,t])} [l_i(s), r_i(t)]\right) \cup \left(\bigcup_{(i,j)\in\widetilde{\mathcal{G}}(e[s,t])} [\hat{l}_{i,e[s,t]}(s), \hat{r}_{j,e[s,t]}(t)]\right),$$

where by a slight abuse of notation let $\mathcal{L}(e[s,t])\setminus\mathcal{B}(e[s,t])$ be all index-pairs (i,i) in $\mathcal{L}(e[s,t])$ such that i is not in $\mathcal{B}(e[s,t])$.

We observe that the proxy coverage can be expressed as a disjoint union via $\widetilde{\mathcal{G}}(e[s,t])$.

▶ **Lemma 20.** *Let e be an edge and let $0 \le s \le t \le 1$. Then $\widehat{\mathrm{Cov}}_P(e[s,t])$ is the disjoint union (refer to Figure 6)*

$$\widehat{\mathrm{Cov}}_P(e[s,t]) = \left(\bigsqcup_{i\in\mathcal{L}(e[s,t])\setminus\mathcal{B}(e[s,t])} [l_i(s), r_i(t)]\right) \sqcup \left(\bigsqcup_{(i,j)\in\widetilde{\mathcal{G}}(e[s,t])} [\hat{l}_{i,e[s,t]}(s), \hat{r}_{j,e[s,t]}(t)]\right).$$

▶ **Lemma 21.** *If i is bad for $e[s,t]$, then i is good for $\mathrm{rev}(e[s,t])$ and $l_i(s), r_i(t) \in [l_i(t), r_i(s)]$.*

▶ **Lemma 22.** *Let e be an edge of S and let $0 \le s \le t \le 1$ be given. Then (refer to Figure 6)*

$$\mathrm{Cov}_P(e[s,t],4\Delta) \subset \widehat{\mathrm{Cov}}_P(e[s,t]) \cup \widehat{\mathrm{Cov}}_P(\mathrm{rev}(e[s,t])) \subset \mathrm{Cov}_P(\{e[s,t],\mathrm{rev}(e[s,t])\},4\Delta).$$

We extend the proxy coverage $\widehat{\mathrm{Cov}}$ to also be defined for Type (I)-subcurves of S, where we set $\widehat{\mathrm{Cov}}_P(S[s,t]) = \mathrm{Cov}_P(S[s,t],4\Delta)$. Overall, we see that for any $\pi \in \mathcal{C}_S(P)$ – not just Type (II)- and (III)-subcurves of S – there are at most two elements $\pi_1, \pi_2 \in \mathcal{C}_S(P)$ with

$$\mathrm{Cov}_P(\pi,4\Delta) \subset \widehat{\mathrm{Cov}}_P(\pi_1) \cup \widehat{\mathrm{Cov}}_P(\pi_2) \subset \mathrm{Cov}_P(\{\pi_1,\pi_2\},4\Delta).$$

▶ **Theorem 23.** *Let P be a polygonal curve, and let S be a simplification of P. Then for any curve $\pi \in \mathbb{X}^d_\ell$ there is a set $S_\pi \subset \mathcal{C}_S(P)$ of size at most 16 such that*

$$\mathrm{Cov}_P(\pi,\Delta) \subset \bigcup_{s\in S_\pi} \widehat{\mathrm{Cov}}_P(s).$$

Proof. This follows from the definition of the proxy coverage, Lemma 22 and Theorem 11. ◀

5 Maintaining the proxy coverage along a sweep-sequence

In this section we present an algorithm that given $\mathfrak{s} \in \mathfrak{S}_e$ maintains a symbolic representation of $\widehat{\mathrm{Cov}}_P(e[s,t])$ in (roughly) logarithmic time per element in $\mathfrak{s}$. To this end we maintain the set $\mathcal{U}(e[s,t])$ of inclusion-wise maximal index-pairs (i,j) such that there is a path from cell i at height s to cell j at some height $\hat{t} \leq t$, which helps maintain $\widetilde{\mathcal{G}}(e[s,t])$ and $\mathcal{L}(e[s,t]) \setminus \mathcal{B}(e[s,t])$. We want to highlight that maintaining $\mathcal{G}(e[s,t])$ and $\mathcal{L}(e[s,t])$ as a symbolic representation of the 4Δ-Coverage can take total time in $\Theta(n|\mathfrak{S}_e|)$ (Figure 7) motivating our techniques.

▶ **Theorem 24.** *At the beginning and end of the loop in Lines 7 – 15 of* Maintain *of Algorithm 1, the set of index-pairs $\widetilde{\mathcal{G}}$ coincides with $\widetilde{\mathcal{G}}(e[s,t])$, $\mathcal{L}$ coincides with $\mathcal{L}(e[s,t]) \setminus \mathcal{B}(e[s,t])$ and $\mathcal{U}$ coincides with $\mathcal{U}(e[s,t])$. Further, executing* Maintain *takes $\mathcal{O}(n \log n)$ time.*

▶ **Corollary 25.** *Let $\mathfrak{s} \in \mathfrak{S}_e$ be a sweep-sequence. There are $m = \mathcal{O}(n)$ (not necessarily distinct) index-pairs $p_1 = (i_1, j_1), \ldots, p_m = (i_m, j_m)$ together with m contiguous subsets $I_i = \{(s_{a_i}, t_{a_i}), \ldots, (s_{b_i}, t_{b_i})\} \subset \mathfrak{s}$ such that for every $(s,t) \in \mathfrak{s}$ it holds that*

$$\widetilde{\mathcal{G}}(e[s,t]) \cup (\mathcal{L}(e[s,t]) \setminus \mathcal{B}(e[s,t])) = \bigcup_{1 \leq i \leq m, (s,t) \in I_i} \{p_i\}.$$

Further for $p_k = (i_k, j_k)$ the index i_k is either always good or always bad for $e[s,t]$ for $(s,t) \in I_k$, and the index j_k is either always good or always bad for $e[s,t]$ for $(s,t) \in I_k$. The index-pair p_i as well as the values a_i and b_i can be computed for all i in total time $\mathcal{O}(n \log n)$.

Overall, we can maintain a symbolic representation of $\widehat{\mathrm{Cov}}_P(\cdot)$ during a sweep-sequence in total time $\mathcal{O}(n \log n)$ which enables the use of the maintained sets for batch point-queries.

▶ **Theorem 26.** *Let $\mathfrak{s} \in \mathfrak{S}_e$, and let $Q \subset [0,1]$ together with $w_Q : Q \to \mathbb{N}$ be a weighted point-set. For any $Q' \subset Q$ let $w_Q(Q') = \sum_{q \in Q'} w_q(q)$ denote its weight. There is an algorithm which computes $w_Q(Q \cap \widehat{\mathrm{Cov}}_{A_S}(e[s,t]))$ for every $(s,t) \in \mathfrak{s}$ in $\mathcal{O}(|Q| \log |Q| + n \log n)$ time.*

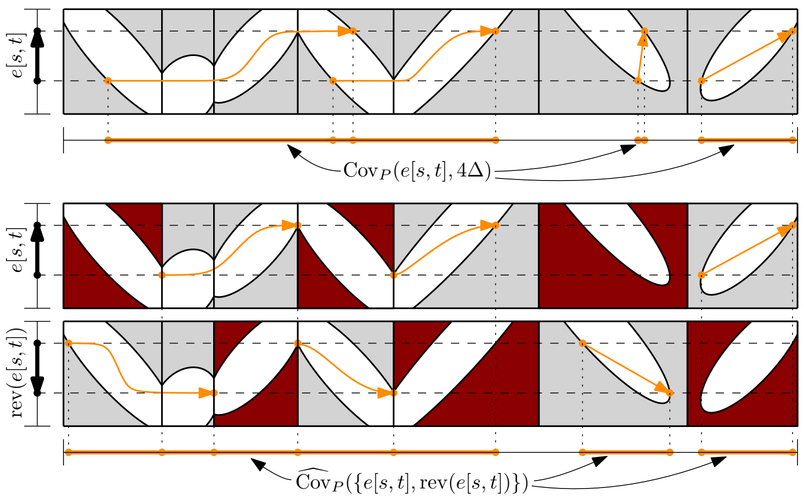
$e[s,t]$
$\mathrm{Cov}_P(e[s,t], 4\Delta)$
$e[s,t]$
$\mathrm{rev}(e[s,t])$
$\widehat{\mathrm{Cov}}_P(\{e[s,t], \mathrm{rev}(e[s,t])\})$

Figure 6 Illustration of the proxy coverage of $e[s,t]$ and $\mathrm{rev}(e[s,t])$ compared to the 4Δ-coverage of $e[s,t]$. Cells with bad index for $e[s,t]$ and $\mathrm{rev}(e[s,t])$ are highlighted in red.

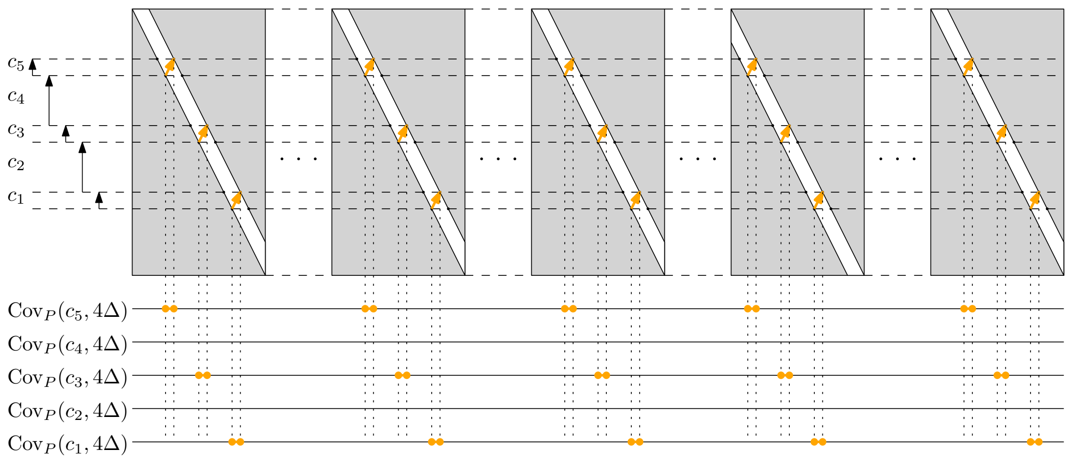
c_5
c_4
c_3
c_2
c_1
$\mathrm{Cov}_P(c_5, 4\Delta)$
$\mathrm{Cov}_P(c_4, 4\Delta)$
$\mathrm{Cov}_P(c_3, 4\Delta)$
$\mathrm{Cov}_P(c_2, 4\Delta)$
$\mathrm{Cov}_P(c_1, 4\Delta)$

Figure 7 Illustration of how the combinatorial description of the 4Δ-coverage of two neighbouring elements in a sweep-sequence may differ by up to n index-pairs. Instead, $\widetilde{\mathrm{Cov}}(c_i) = \emptyset$ for all i.

Algorithm 1 Maintenance of the reduced global group.

```
procedure MAINTAIN($\mathcal{D}_{4\Delta}(e, P)$, sweep-sequence $\mathfrak{s} = \{(s_1, t_1), \ldots, (s_m, t_m)\}$ of $\mathcal{E}(e, P)$)
    for $i < n$ do compute the contiguous sets $B_i = \{(s, t) \in \mathfrak{s} \mid i$ is bad for $e[s, t]\}$
    Compute coverage $\mathrm{Cov}_P(e[s_m, t_m], 4\Delta)$ represented as $\mathcal{O}(n)$ disjoint intervals
    Compute $\mathcal{L}(e[s_m, t_m])$ and $\mathcal{G}(e[s_m, t_m])$ from $\mathrm{Cov}_P(e[s_m, t_m], 4\Delta)$
    $\mathcal{U}, \mathcal{L}, \mathcal{G}, (s, t) \leftarrow \mathcal{L}(e[s_m, t_m]) \cup \mathcal{G}(e[s_m, t_m]), \mathcal{L}(e[s_m, t_m]), \mathcal{G}(e[s_m, t_m]), (s_m, t_m)$
    Compute $\mathcal{B}(e[s, t])$ from $B_i$ and with it $\widetilde{\mathcal{G}} \leftarrow \widetilde{\mathcal{G}}(e[s_m, t_m])$
    for $(s', t')$ in REVERSED(S) do                                        ▷ sweeping window events
        if $s' \neq s$ then                                                        ▷ $s' = s - 1$
            Compute $\mathcal{B}(e[s', t]) \setminus \mathcal{B}(e[s, t])$ and $\mathcal{B}(e[s, t]) \setminus \mathcal{B}(e[s', t])$ via the sets $B_i$
            Update $\mathcal{U}, \mathcal{L}, \mathcal{G}$ and $\widetilde{\mathcal{G}}$ via ADVANCESTART$(s, s', t)$ from Algorithm 2
            Update $\mathcal{B}(e[s, t])$ to $\mathcal{B}(e[s', t])$ via the sets $B_i$, and $s \leftarrow s'$
        if $t' \neq t$ then                                                        ▷ $t' = t - 1$
            Compute $\mathcal{B}(e[s, t']) \setminus \mathcal{B}(e[s, t])$ and $\mathcal{B}(e[s, t]) \setminus \mathcal{B}(e[s, t'])$ via the sets $B_i$
            Update $\mathcal{U}, \mathcal{L}, \mathcal{G}$ and $\widetilde{\mathcal{G}}$ via ADVANCEEND$(s, t, t')$ from Algorithm 2
            Update $\mathcal{B}(e[s, t])$ to $\mathcal{B}(e[s', t])$ via the sets $B_i$, and $t \leftarrow t'$
```

Proof. Define following values for all i, j along the sweep-sequence $\mathfrak{s}$ where for $(s, t) \in \mathfrak{s}$:

$$L_i((s,t)) = \sum_{\substack{q \in Q,\ q \text{ in cell } i \\ \hat{l}_{i,e[s,t]}(s) \neq \infty,\ \hat{l}_{i,e[s,t]}(s) \leq q}} w_Q(q), \qquad R_j((s,t)) = \sum_{\substack{q \in Q,\ q \text{ in cell } j \\ \hat{r}_{j,e[s,t]}(t) \neq \infty,\ q \leq \hat{r}_{j,e[s,t]}(t)}} w_Q(q),$$

$$M_i((s,t)) = \sum_{\substack{q \in Q,\ i \notin \mathcal{B}(e[s,t]),\ q \text{ in cell } i \\ l_i(s) \neq \infty,\ r_i(t) \neq \infty,\ l_i(s) \leq q \leq r_i(t)}} w_Q(q), \qquad D(i,j) = \sum_{\substack{q \in Q,\ q \text{ in cell } m \\ i \leq m \leq j}} w_Q(q).$$

The value $L_i((s,t))$ corresponds to the weight of points in cell i that lie right of $\hat{l}_{i,e[s,t]}(s)$. Similarly, $R_i((s,t))$ corresponds to the weight of points in cell i that lie left of $\hat{r}_{i,e[s,t]}(t)$. The value $M_i((s,t))$ corresponds to the weight of points in cell i that lie in between $\hat{l}_{i,e[s,t]}(s)$ and $\hat{r}_{i,e[s,t]}(t)$ if i is good for $e[s,t]$. Lastly $D(i,j)$ corresponds to the total weight of points that lie on edge $i, \ldots, j$. Then for any $(s,t) \in \mathfrak{s}$ and any $(i,i) \in \mathcal{L}(e[s,t]) \setminus \mathcal{B}(e[s,t])$ it holds that $M_i((s,t)) = w_Q(Q \cap [l_i(s), r_i(t)])$. Similarly, for any $(i,j) \in \widetilde{\mathcal{G}}(e[s,t])$ it holds that

Algorithm 2 Advancing the start/end point of the sweeping window.

```
1:  procedure ADVANCESTART($s$,$s'$,$t$)
2:      Get $\mathcal{U}$, $\mathcal{L}$, $\mathcal{G}$, $\widetilde{\mathcal{G}}$, $\mathcal{B}(e[s,t])$, $\mathcal{B}(e[s',t]) \setminus \mathcal{B}(e[s,t])$, $\mathcal{B}(e[s,t]) \setminus \mathcal{B}(e[s',t])$ and $\mathcal{D}_{4\Delta}(e,P)$
3:      if $s$ is induced by a lowest point in cell $i$ of $\mathcal{D}_{4\Delta}(e,P)$ then
4:          Identify smallest $j > i$ such that $l_j(s) \neq \infty$
5:          Update index-pairs in $\mathcal{U}$ and in $\mathcal{G}$ starting at $i$ to start at $j$
6:      if $s'$ is induced by a upper boundary of free space interval of cell $i$ of $\mathcal{D}_{4\Delta}(e,P)$ then
7:          Identify largest $j < i$ such that $l_j(s)$ is in the interior of cell $j$
8:          Update index-pairs in $\mathcal{U}$ and in $\mathcal{G}$ starting at $i$ to start at $j$
9:      if $s'$ is induced by a highest point in cell $i$ of $\mathcal{D}_{4\Delta}(e,P)$ then add $(i,i)$ to $\mathcal{U}$
10:     for changed index-pair $(i,j)$ in $\mathcal{U}$ and $\mathcal{G}$ do
11:         if $i \neq j$ and $r_j(t) \neq \infty$ then move index-pair to $\mathcal{G}$, else move it to $\mathcal{U}$
12:         Remove any dominated index-pairs involving $(i,j)$ from $\mathcal{G}$ and $\mathcal{U}$
13:     Update $\widetilde{\mathcal{G}}$ from $\mathcal{G}$ after merging overlapping intervals via $\mathcal{B}(e[s,t])$ and $\mathcal{B}(e[s',t])$
14:     Update $\mathcal{L}$ via the changes to $\mathcal{U}$, $\mathcal{B}(e[s,t])$ and $\mathcal{B}(e[s,t'])$
15: procedure ADVANCEEND($s$,$t$,$t'$)
16:     dual to ADVANCESTART; refer to full version in appendix
```

$L_i((s,t)) + D(i+1,j-1) + R_j((s,t)) = w_Q(Q \cap [\hat{l}_{i,e[s,t]}(s), \hat{r}_{i,e[s,t]}(t)])$. Then, by Lemma 20

$$w_Q(Q \cap \widehat{\mathrm{Cov}}_{A_S}(e[s,t])) =$$

$$= \left(\sum_{i \in \mathcal{L}(e[s,t]) \setminus \mathcal{B}(e[s,t])} w_Q(Q \cap [l_i(s), r_i(t)]) \right) + \left(\sum_{(i,j) \in \widetilde{\mathcal{G}}(e[s,t])} w_Q(Q \cap [\hat{l}_{i,e[s,t]}(s), \hat{r}_{j,e[s,t]}(t)]) \right)$$

$$= \left(\sum_{i \in \mathcal{L}(e[s,t]) \setminus \mathcal{B}(e[s,t])} M_i((s,t)) \right) + \left(\sum_{(i,j) \in \widetilde{\mathcal{G}}(e[s,t])} L_i((s,t)) + D(i+1,j-1) + R_j((s,t)) \right).$$

Observe that $D(i,j)$ can be provided via a data-structure that first computes $d_i = \sum_{q \in Q, q \text{ in cell } i} w_Q(q)$ for every i in total time $\mathcal{O}(|Q| \log n)$ and stores them in a balanced binary tree as leaves, where every inner node stores the sum of the values of its children. For every $i \leq j$ the value $D(i,j) = \sum_{i \leq m \leq j} d_m$ can then be returned in $\mathcal{O}(\log n)$ time by identifying in $\mathcal{O}(\log n)$ time all $\mathcal{O}(\log n)$ maximal subtrees whose children lie in the interval $[i,j]$ and then returning the sum of the stored values in the root of each maximal subtree.

Next we show that $L_i(\cdot)$ (resp. $R_j(\cdot)$ and $M_i(\cdot)$) can correctly be maintained when performing the sweep of $\mathfrak{s}$. To this end let

$$I_q = \{(s,t) \in \mathfrak{s} \mid q \text{ is in cell } i \text{ and } \hat{l}_{i,e[s,t]}(s) \neq \infty, \hat{l}_{i,e[s,t]} \leq q\}.$$

Refer to Figure 8. As the free space in every cell is convex, throughout $\mathfrak{s}$ the first indices are monotone and the y-coordinates of the left-most and right-most points are stored in $\mathcal{E}(A_e)$, for every $q \in Q$ the contiguous subsets

$$L_q = \{(s,t) \in \mathfrak{s} \mid q \text{ is in cell } i \text{ and } l_i(s) \neq \infty, l_i(s) \leq q\} \text{ and}$$
$$R_q = \{(s,t) \in \mathfrak{s} \mid q \text{ is in cell } i \text{ and } l_i(s) \neq \infty, r_i(s) \geq q\}$$

can be computed in $\mathcal{O}(\log n)$ time. Similarly the set $B_i = \{(s,t) \in \mathfrak{s} \mid i \text{ is bad for } e[s,t]\}$ is also a contiguous subset of $\mathfrak{s}$ and can be computed in $\mathcal{O}(\log n)$ time. Let Q_i denote all $q \in Q$ that lie on the ith edge of P. Then for any $q \in Q_i$

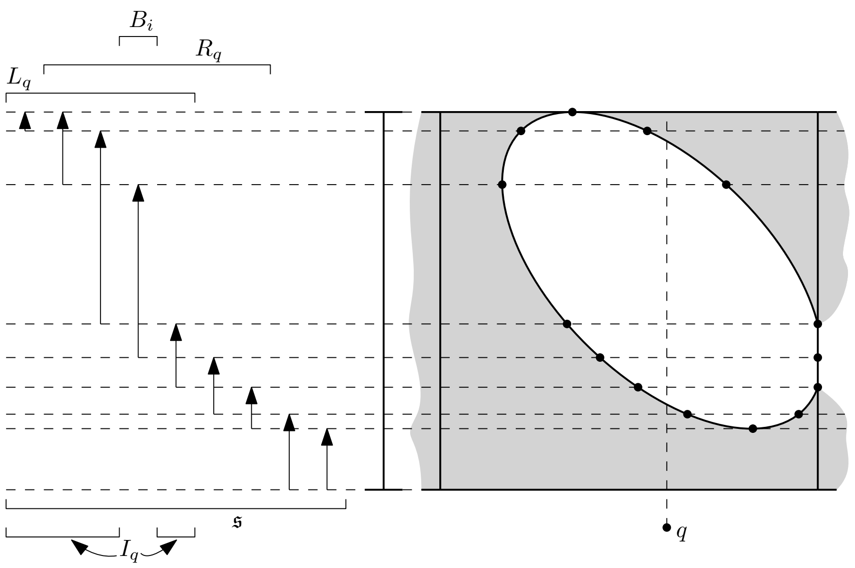

Figure 8 Construction of the set I_q encoding when $\hat{l}_{i,e[s,t]}(s) \leq q$ from Proof of Theorem 26 via the sets L_q and R_q encoding when $l_i(s) \leq q$ and $r_i(s) \geq q$ and the set B_i for all $(s,t) \in \mathfrak{s}$.

$$I_q = (L_q \cap (\mathfrak{s} \setminus B_i)) \cup ((\mathfrak{s} \setminus R_q) \cap B_i),$$

and thus I_q consists of $\mathcal{O}(1)$ contiguous disjoint subsets of $\mathfrak{s}$. Thus all sets I_q (represented as $\mathcal{O}(1)$ contiguous subsets of $\mathfrak{s}$) can be computed in time $\mathcal{O}(|Q| \log n + n \log n)$. Further,

$$L_i((s,t)) = \sum_{q \in Q_i} \mathbb{1}_{q \in I_q} w_Q(q),$$

and hence $L_i(\cdot)$ can be maintained in total time $\mathcal{O}(|Q| + n \log n)$ after one initial computation of all I_q, by adding $w_Q(q)$ for $q \in Q_i$ whenever (s,t) enters the $\mathcal{O}(1)$ contiguous disjoint subsets of I_q, and subtracting $w_Q(q)$ for $q \in Q_i$ whenever (s,t) exits the $\mathcal{O}(1)$ contiguous disjoint subsets of I_q. Sorting the boundaries of all I_q preparing them for the maintenance of $L_i(\cdot)$ takes $\mathcal{O}(|Q| \log |Q|)$ time. Similarly $M_i(\cdot)$ and $R_j(\cdot)$ can be maintained.

Overall, the values $L_i(\cdot)$, $R_j(\cdot)$, $M_i(\cdot)$ and $D(i,j)$ are correctly maintained in total time $\mathcal{O}(|Q| \log |Q|)$ time such that they can be evaluated in $\mathcal{O}(\log n)$. By Corollary 25 there are only $\mathcal{O}(n)$ total updates to $\widetilde{\mathcal{G}}(\cdot)$ and $\mathcal{L}(\cdot) \setminus \mathcal{B}(\cdot)$. Hence, $w_Q(\cdot)$ can be correctly maintained along $\mathfrak{s}$ by updating it whenever $L_i(\cdot)$, $R_j(\cdot)$, $M_i(\cdot)$, $\widetilde{\mathcal{G}}(\cdot)$, $\mathcal{L}(\cdot) \setminus \mathcal{B}(\cdot)$ or $\mathcal{B}(\cdot)$ change. Thus computing $w_Q(e[s,t])$ for all $(s,t) \in S$ takes $\mathcal{O}(|Q| \log |Q| + n \log n)$ time. ◀

6 Ground-set discretization and sampling

We now present two $(\mathcal{O}(\log n), 4)$-approximation algorithm for SC that uses the efficient batch point queries from Theorem 26.

▶ **Definition 27** (Atomic intervals). *Let P be a polygonal curve, and let $\Delta > 0$ and ℓ be given. Let S be a simplification of P. Let G be the set of all intersection-points of horizontal lines at height h for y-coordinates $h \in \mathcal{E}(S,P)$ of extremal points of $\mathcal{D}_{4\Delta}(S,P)$ with the boundary of the free space $\mathcal{D}_{4\Delta}(S,P)$. From this, define the set of atomic intervals $\mathcal{A}(S,P)$ as the set of intervals describing the arrangement of $[0,1]$ defined by the set $\{x \in [0,1] \mid \exists y \in [0,1] : (x,y) \in G\}$ (Figure 9).*

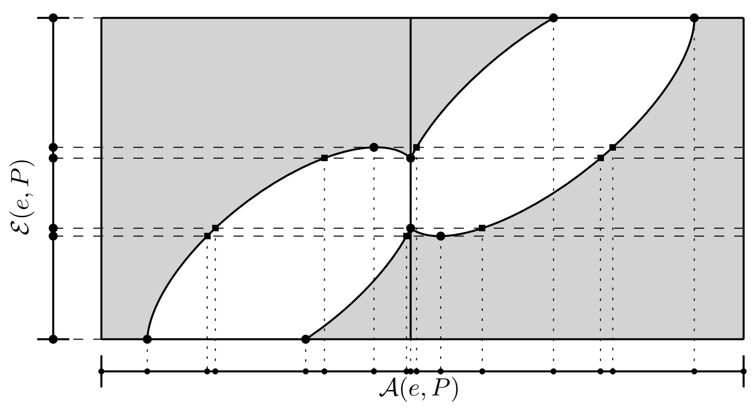

Figure 9 Illustration of atomic intervals $\mathcal{A}(e,P)$ induced by $\mathcal{E}(e,P)$.

▶ **Observation 28.** *As $|\mathcal{E}(S,P)| \leq 8n^2$, and each horizontal line intersects at most n cells, and the free space in every cell is convex it follows, the set of all midpoints of atomic intervals $\mathcal{A}(S,P)$ is a point set $A \subset [0,1]$ of size $16n^3$ such that for any $C \subset \mathcal{C}_S(P)$ it holds that*

$$\widehat{\mathrm{Cov}}_P(C) = [0,1] \iff A \subset \widehat{\mathrm{Cov}}_P(C).$$

▶ **Lemma 29.** *Let P be a polygonal curve of complexity n and let $\Delta > 0$ and $\ell \leq n$ be given. Let S be a simplification of P. Let $\mathcal{E}(S,P)$ be the extremal points of $\mathcal{D}_{4\Delta}(S,P)$. Let $A \subset [0,1]$ be a given set of size at most cn^3 points. For every edge e of S let W_e be the set of atomic intervals a in $\mathcal{A}(e,P)$ such that $w_A(a) = |A \cap a| \neq 0$. Let k_Δ be the size of the smallest set $\mathcal{C}^* \subset \mathbb{X}_\ell^d$ such that $\mathrm{Cov}_P(\mathcal{C}^*,\Delta) = [0,1]$. There is an algorithm that computes a set $\mathcal{C} \subset \mathcal{C}_S(P)$ of size $16(3\ln(n) + \ln c + 1)k_\Delta$ such that $A \subset \widehat{\mathrm{Cov}}_{A_S}(\mathcal{C})$ in time*

$$\mathcal{O}\left(n^2\ell\log^2 n + |A|\log n + k_\Delta n^2\log^3 n + \log n\sum_e |W_e|\right).$$

Proof Sketch. The algorithm operates in rounds. In every round we start with a subset A' of A, where initially $A' = A$, and in every round for the partial solution R of the greedy algorithm it holds that $A' = A \setminus \widehat{\mathrm{Cov}}_P(R)$. Further in every round and for every edge e of S we maintain the set W'_e of atomic intervals a in $\mathcal{A}(e,P)$ such that $w_{A'}(a) = |A' \cap a| \neq 0$. These weights in combination with Theorem 26 allow us to identify the Type (II)- or (III)-subedge r of S maximizing $|A' \cap \widehat{\mathrm{Cov}}_P(r)| = |(A \setminus \widehat{\mathrm{Cov}}_P(R)) \cap \widehat{\mathrm{Cov}}_P(r)|$. We similarly identify the Type (I)-curve r maximizing $|A' \cap \widehat{\mathrm{Cov}}_P(r)|$ by initially computing $\widehat{\mathrm{Cov}}_P(r)$ for every Type (I)-curve and then in every round recomputing $|A' \cap \widehat{\mathrm{Cov}}_P(r)|$. Afterwards we add the subcurve maximizing $|A' \cap \widehat{\mathrm{Cov}}_P(r)|$ to R. Lastly we update $A' \leftarrow A' \setminus \widehat{\mathrm{Cov}}_P(r)$. This results in almost all intervals in W'_e whose weights are updated, to have their weights set to zero and are thus removed from W'_e. The number of updates setting such a weight to zero, together with the initial application of Theorem 26 dominate the running time. Lastly, by standard greedy SetCover arguments [14, 22, 18] and Theorem 23 the number of rounds until the algorithm terminates ($A' = \emptyset$) is bounded by $16(\ln(|A|) + 1)$. ◀

▶ **Corollary 30.** *There is an $(48\ln(n) + 64, 4)$-approximation algorithm for SC. Given a polygonal curve P of complexity n, $\Delta > 0$ and $\ell \leq n$, its running time is in $\mathcal{O}\left(n^3\log^3 n\right)$.*

Proof. Lemma 29 together with Observation 28 and Theorem 23 yields the claim. ◀

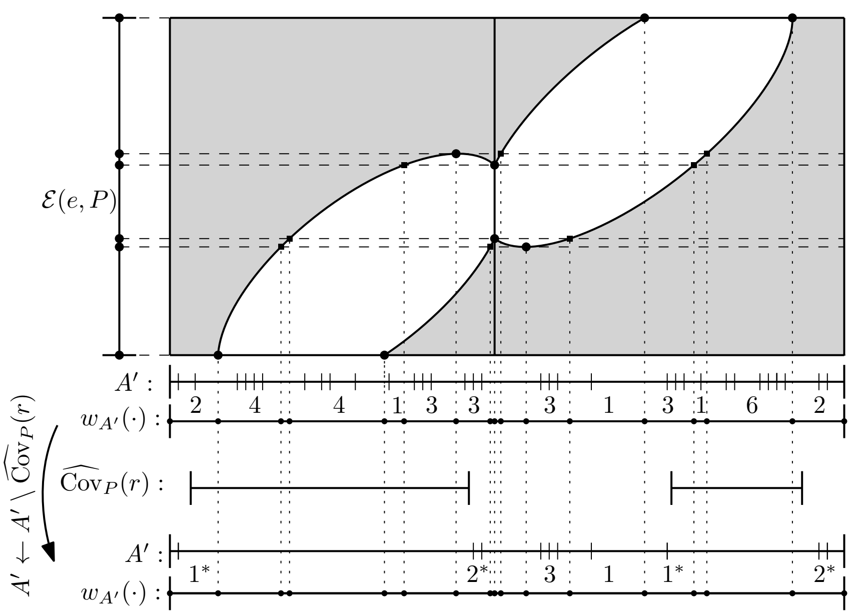

Figure 10 Illustration to Proof Sketch of Lemma 29. Modification of non-zero weights when adding r to R. All weights that are updated but not necessarily set to 0 are highlighted with a *.

The running time of Corollary 30 is dominated by the size of $\mathcal{A}(S,P)$. We now present techniques enabling identification of a representative subset of $\mathcal{A}(S,P)$.

▶ **Theorem 31.** *Let n lists L_i be given, each containing m_i sorted values such that all values are distinct and in every list L_i and for every $j \leq m_i$ identifying the item at position j takes $\mathcal{O}(\log m_i)$ time and also for given v determining the maximal index j such that the jth item is less than v takes $\mathcal{O}(\log m_i)$ time. Then for every $K \leq \sum_i m_i$ in $\mathcal{O}(Kn \log \max_i m_i)$ time one can determine K values $v_1 < \ldots < v_K$ such that*

1. *for every $x \in \bigcup_i L_i$ there is an i such that $x \in [v_i, v_{i+1}]$ and*
2. *$|[v_i, v_{i+1}] \cap \bigcup_i L_i| = O\left(\frac{\sum_i m_i}{K}\right)$ for all i.*

Proof Sketch. We recursively compute a value m such that both a constant fraction of values among all lists is smaller than m and a constant fraction of values among all lists is bigger than m. To find m we compute the middle element of every list in $\mathcal{O}(n \log \max_i m_i)$ time. Then m is (roughly) the middle element of these middle elements. ◀

▶ **Lemma 32.** *For every $\alpha \in [0,3]$ in $\mathcal{O}(n^{1+\alpha} \log n)$ time one can determine $\mathcal{O}(n^\alpha)$ so called **α-coarse** intervals partitioning $[0,1]$, each containing at most $\mathcal{O}(n^{3-\alpha})$ intervals of $\mathcal{A}(S,P)$.*

Proof Sketch. By the structure of $\mathcal{D}_{4\Delta}(e,P)$ we can compute how many atomic intervals of $\mathcal{A}(e,P)$ are contained on any edge of P in $\mathcal{O}(\log n)$ time. Similarly, on any edge of P we can compute the ith atomic interval boundary via $\mathcal{O}(1)$ binary searches of $\mathcal{E}(e,P)$. Hence after $\mathcal{O}(n \log n)$ precomputation time per edge e of S, we can output the ith interval boundary of $\mathcal{A}(e,P)$ in $\mathcal{O}(\log n)$ time for any i. Hence Theorem 31 implies the claim. ◀

We now use Lemma 32 together with Lemma 29 to obtain the following lemma and theorem. The overarching approach for the algorithm is as follows: First use Lemma 32 to compute roughly $\mathcal{O}(n^{3/2})$ points from $\mathcal{A}(S,P)$. These points we cover with a set R_1 of

$\tilde{\mathcal{O}}(k_\Delta)$ elements from $\mathcal{C}_S(P)$ in time $\tilde{\mathcal{O}}(k_\Delta n^{5/2})$ via Lemma 29. Between any two consecutive points of the $\mathcal{O}(n^{3/2})$ points there are at most $\mathcal{O}(n^{3/2})$ other points from $\mathcal{A}(S,P)$, and at most $\tilde{\mathcal{O}}(k_\Delta n)$ of these sets are **not** covered by R_1. This leaves us with $\tilde{\mathcal{O}}(k_\Delta n^{5/2})$ uncovered points in $\mathcal{A}(S,P)$. We again invoke Lemma 29, covering these remaining points with a set R_2 of $\tilde{\mathcal{O}}(k_\Delta)$ elements from $\mathcal{C}_S(P)$ in time $\tilde{\mathcal{O}}(k_\Delta n^{5/2})$.

▶ **Lemma 33.** *Let $\alpha \in [0,3]$ and $K > 0$ be given. Let $C \subset \mathcal{C}_S(P)$ be a set covering all midpoints of α-coarse intervals of size $\mathcal{O}(K \log n)$. Then there are $\mathcal{O}(Kn^{4-\alpha}\log n)$ atomic intervals in $\mathcal{A}(S,P)$ and $\mathcal{O}(Kn^{4-\alpha}\log n + Kn^2\log n)$ atomic intervals in $\bigcup_e \mathcal{A}(e,P)$ that are not contained in $\widehat{\mathrm{Cov}}_{A_S}(C)$. They can be computed in $\mathcal{O}(Kn^{4-\alpha}\log^2 n + Kn^2\log^2 n)$ time.*

▶ **Theorem 4.** *There is a $(96\ln(n)+128, 4)$-approximation for SC. Given a polygonal curve P of complexity n, $\Delta > 0$ and $\ell \le n$, its running time is in $\mathcal{O}\left(\left(n^2\ell + \sqrt{k_\Delta}n^{\frac{5}{2}}\right)\log^2 n\right)$, where k_Δ is the size of the smallest subset $C^* \subset \mathbb{X}^d_\ell$ such that $\mathrm{Cov}_P(C^*,\Delta) = [0,1]$.*

Proof Sketch. The algorithm maintains an internal guess K of k_Δ, which initially is 1. We describe a subroutine that decides, whether $K < k_\Delta$ or outputs a solution of size $\mathcal{O}(K\log n)$. If the subroutine outputs that $K < k_\Delta$ we set $K \leftarrow 2K$, which then implies the claim.

First set $\alpha = \frac{3}{2} + \frac{\log K}{2\log n} + \frac{\log\log n}{\log n}$. Next compute a set of α-coarse intervals via Lemma 32. From them compute their midpoints M and for every edge e of S compute the subset W_e of $\mathcal{A}(e,P)$ containing such a midpoint. Then W_e is the set of atomic intervals in $\mathcal{A}(e,P)$ with $w_M(\cdot) \neq 0$. The algorithm then attempts to cover the mid-points of these intervals via $\mathcal{O}(K\log n)$ rounds of the algorithm from Lemma 29. If the algorithm does not terminate after $\mathcal{O}(K\log n)$ rounds we return that $K < k_\Delta$. Otherwise we compute the set of atomic intervals $A \subset \mathcal{A}(S,P)$ and for every e the set of atomic intervals in $W'_e \subset \mathcal{A}(e,P)$ that are not fully contained in the solution returned from the first call of the algorithm from Lemma 29 via Lemma 33. We then invoke the algorithm from Lemma 29 on the midpoints M' of A and W'_e which is precisely the set of atomic intervals in $\mathcal{A}(e,P)$ with $w_{M'}(\cdot) \neq 0$. As $K \in \mathcal{O}(k_\Delta)$ and thus $K \in \mathcal{O}(n)$, the running time of the subroutine is

$$\mathcal{O}(n^{1+\alpha}\log n + Kn^{4-\alpha}\log^3 n + Kn^2\log^2 n) = \mathcal{O}(K^{\frac{1}{2}}n^{\frac{5}{2}}\log^2 n).$$

And thus overall the running time of the algorithm is

$$\mathcal{O}\left(n^2\ell\log^2 n + \sum_{K=1}^{\log(k_\Delta)}\left(\left(2^K\right)^{\frac{1}{2}}n^{\frac{5}{2}}\log^2 n\right)\right) = \mathcal{O}\left(n^2\ell\log^2 n + \sqrt{k_\Delta}n^{\frac{5}{2}}\log^2 n\right).$$ ◀

7 Subtrajectory Coverage Maximization

The techniques discussed in this paper can also be used to obtain the following result.

▶ **Theorem 5.** *Let $\varepsilon \in \left(0, \frac{1}{5}\right]$. There is a $\left(\frac{e-1}{16e}, 4+\varepsilon\right)$-approximation algorithm for SCM, where e is the base of the natural logarithm. Given a polygonal curve P of complexity n, $\Delta > 0$, $\ell \le n$ and $k > 0$, its running time is in $\mathcal{O}((k+\ell)n^2\varepsilon^{-2}\log^2 n\log\varepsilon^{-1})$.*

Proof Sketch. We first compute an ε-approximation of the 4Δ-free space, which allows describing $\|\widehat{\mathrm{Cov}}_P(\cdot)\|$ as a sum of linear functions. A sum of linear functions is a linear function itself. As we can maintain a combinatorial description of the proxy coverage we are able to parsimoniously maintain this linear function during a scan of a sweep-sequence in $\mathfrak{S}_e$, which allows evaluating $\|\widehat{\mathrm{Cov}}_P(\cdot)\|$ for all elements in a single sweep-sequence in time roughly $\mathcal{O}(n\log n)$. Using this subroutine once per round and sweep-sequence results in a running time of $\mathcal{O}(kn^2\log^2 n)$. ◀

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, Houston, TX, USA, June 10-15, 2018*, PODS '18, pages 75–87, 2018. doi:10.1145/3196959.3196972.

2 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.

3 Frederik Brüning, Hugo A. Akitaya, Erin W. Chambers, and Anne Driemel. Subtrajectory clustering: Finding set covers for set systems of subcurves. *Comput. Geom. Topol.*, 2(1):1:1–1:48, February 2023. doi:10.57717/cgt.v2i1.7.

4 Frederik Brüning, Jacobus Conradi, and Anne Driemel. Faster approximate covering of subcurves under the fréchet distance. In *30th Annual European Symposium on Algorithms (ESA 2022)*, volume 244 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 28:1–28:16, Dagstuhl, Germany, 2022. Schloss Dagstuhl – Leibniz-Zentrum für Informatik. doi:10.4230/LIPIcs.ESA.2022.28.

5 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, 2017. doi:10.1145/3139958.3139964.

6 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.

7 Kevin Buchin, Maike Buchin, Joachim Gudmundsson, Maarten Löffler, and Jun Luo. Detecting commuting patterns by clustering subtrajectories. *International Journal of Computational Geometry and Applications*, 21(3):253–282, 2011. doi:10.1142/S0218195911003652.

8 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.

9 Jacobus Conradi and Anne Driemel. Finding complex patterns in trajectory data via geometric set cover, 2023. doi:10.48550/arXiv.2308.14865.

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

11 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.

12 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.

13 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.

14 David S Johnson. Approximation algorithms for combinatorial problems. In *Proceedings of the fifth annual ACM symposium on Theory of computing*, pages 38–49, 1973. doi:10.1145/800125.804034.

15 Andreas Krause and Daniel Golovin. Submodular function maximization. In Lucas Bordeaux, Youssef Hamadi, and Pushmeet Kohli, editors, *Tractability: Practical Approaches to Hard Problems*, pages 71–104. Cambridge University Press, 2014. doi:10.1017/CBO9781139177801.004.

16 Lily Lee and W Eric L Grimson. Gait analysis for recognition and classification. In *Proceedings of Fifth IEEE International Conference on Automatic Face Gesture Recognition*, pages 155–162. IEEE, 2002. doi:10.1109/AFGR.2002.1004148.

17 Anqi Liang, Bin Yao, Bo Wang, Yinpei Liu, Zhida Chen, Jiong Xie, and Feifei Li. Subtrajectory clustering with deep reinforcement learning. *VLDB J.*, 33(3):685–702, 2024. doi:10.1007/s00778-023-00833-w.

18 László Lovász. On the ratio of optimal integral and fractional covers. *Discrete mathematics*, 13(4):383–390, 1975. doi:10.1016/0012-365X(75)90058-8.

19 G. L. Nemhauser, L. A. Wolsey, and M. L. Fisher. An analysis of approximations for maximizing submodular set functions - I. *Mathematical Programming*, 14(1):265–294, 1978. doi:10.1007/BF01588971.

20 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.

21 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.

22 Sherman K Stein. Two combinatorial covering theorems. *Journal of Combinatorial Theory, Series A*, 16(3):391–397, 1974. doi:10.1016/0097-3165(74)90062-4.

23 Ivor van der Hoog, Thijs van der Horst, and Tim Ophelders. Faster and deterministic subtrajectory clustering, 2024. doi:10.48550/arXiv.2402.13117.

24 Erik van Sebille, Stephen M. Griffies, Ryan Abernathey, Thomas P. Adams, Pavel Berloff, Arne Biastoch, Bruno Blanke, Eric P. Chassignet, Yu Cheng, Colin J. Cotter, Eric Deleersnijder, Kristofer Döös, Henri F. Drake, Sybren Drijfhout, Stefan F. Gary, Arnold W. Heemink, Joakim Kjellsson, Inga Monika Koszalka, Michael Lange, Camille Lique, Graeme A. MacGilchrist, Robert Marsh, C. Gabriela Mayorga Adame, Ronan McAdam, Francesco Nencioli, Claire B. Paris, Matthew D. Piggott, Jeff A. Polton, Siren Rühs, Syed H.A.M. Shah, Matthew D. Thomas, Jinbo Wang, Phillip J. Wolfram, Laure Zanna, and Jan D. Zika. Lagrangian ocean analysis: Fundamentals and practices. *Ocean Modelling*, 121:49–75, 2018. doi:10.1016/j.ocemod.2017.11.008.

25 Yiqun Xie, Shashi Shekhar, and Yan Li. Statistically-robust clustering techniques for mapping spatial hotspots: A survey. *ACM Comput. Surv.*, 55(2), January 2022. doi:10.1145/3487893.