Voronoi-like diagrams

Towards linear-time algorithms for tree-like abstract Voronoi diagrams

Doctoral Dissertation submitted to the
Faculty of Informatics of the Università della Svizzera Italiana
in partial fulfillment of the requirements for the degree of
Doctor of Philosophy

presented by
Kolja Junginger

under the supervision of
Prof. Evanthia Papadopoulou

August 2020

Dissertation Committee

Prof. Stefan Wolf	Università della Svizzera Italiana, Lugano, Switzerland
Prof. Piotr Didyk	Università della Svizzera Italiana, Lugano, Switzerland
Prof. Rolf Klein	Universität Bonn, Germany
Prof. Stefan Felsner	Technische Universität Berlin, Germany

Dissertation accepted on 4 August 2020

Research Advisor	PhD Program Director
Prof. Evanthia Papadopoulou	**The PhD program Director *pro tempore***

I certify that except where due acknowledgement has been given, the work presented in this thesis is that of the author alone; the work has not been submitted previously, in whole or in part, to qualify for any other academic award; and the content of the thesis is the result of work which has been carried out since the official commencement date of the approved research program.

Kolja Junginger
Lugano, 4 August 2020

Abstract

Computational Geometry is a subfield of Algorithm Design and Analysis with a focus on the design and analysis of algorithms related to discrete geometric objects. The Voronoi diagram is one of the most important structures in Computational Geometry providing proximity information, which is applicable to many different fields of science. For a given set of points in the plane – called *sites* – the classic Voronoi diagram subdivides the plane into regions, such that all points within one region have the same nearest site. Abstract Voronoi diagrams provide a unifying model for various concrete Voronoi diagrams for different sites and different metrics. For example the sites can be disjoint line segments or non-enclosing circles, disjoint convex polygons of constant size and the metrics include all L_p metrics, the Karlsruhe metric and other convex distance measures. In the abstract framework the diagram is not defined via the sites and the distances, but from a bisecting curve system satisfying certain properties.

Updating the classic Voronoi diagram of point sites, after deletion of one site, can be done in linear time as it is well known since 1989. However, this problem has remained open since then for generalized sites other than points and for abstract Voronoi diagrams.

In this dissertation we present a simple, expected linear-time algorithm to update an abstract Voronoi diagram after deletion of one site. To this aim we introduce the concept of a *Voronoi-like diagram*, a relaxed version of a Voronoi construct that has a structure similar to an abstract Voronoi diagram, without however being one. In our algorithm Voronoi-like diagrams serve as intermediate structures, which are considerably simpler to compute. We formalize the concept of a Voronoi-like diagram and prove that it is well-defined and robust under an insertion operation, thus, enabling its use in incremental constructions.

Further, we show that our randomized algorithm can be used to compute the order-(k+1) subdivision within the face of an order-k abstract Voronoi region in expected time linear in the complexity of the region's boundary. Moreover, our randomized algorithm can be adapted to compute the farthest abstract Voronoi diagram in expected linear-time, after the sequence of its faces at infinity

is known.

Finally, we have investigated the possibility to apply Voronoi-like diagrams also to a deterministic algorithmic framework for possible use in deriving deterministic versions of the above mentioned randomized algorithms. We formulate open problems, the solution of which will make progress towards this goal.

Acknowledgements

First of all, I thank my supervisor Evanthia Papadopoulou, who devoted an incredible amount of her time to this research – more than any usual PhD student receives. Without her patience and optimism this project would not have been successful.

I want to thank everyone in my dissertation committee – Rolf Klein, Stefan Felsner, Stefan Wolf and Piotr Didyk – for taking the time to read and evaluate this thesis. I am particularly thankful for Rolf Klein's positive feedback and the valuable discussions about abstract Voronoi diagram during our Voronoi++ meetings. I thank Stefan Felsner, who gave me my first research opportunity during my Master thesis, a very positive experience, which motivated me to start a PhD. I also thank him for the very interesting and helpful discussion about permutations properties, which appear also in this thesis.

I want to thank Chee Yap for his wonderful and long visit to Lugano, bringing all his wisdom about science and many other aspects of life to us – an unforgettable experience. Of course, I thank my two close colleagues Martin Suderland and Ioannis Mantas for turning the daily office work so lively, and also for sharing so much more. I thank Martin for all the hikes and bike tours together and for convincing me to start badminton. I thank Ioannis for making me discover Europe's most beautiful summer destination – Greek islands. I thank my first colleague Lena Arseneva for welcoming me to Lugano and PhD life and for introducing me to Computational Geometry during my first year. Thanks to all the people in the open space for the regular spontaneous coffee breaks in the kitchen (including passionate discussions) and for keeping the coffee machine(s) alive! I thank Lugano for its perfect climate, the beautiful landscape, the lovely (Swiss;-) Italian lifestyle and the great food.

Finally, from all my heart I thank my family and the most important person in my life – María del Ángel.

Contents

List of Figures

Chapter 1

Introduction

Computational Geometry is a subfield of Algorithm Design and Analysis with a focus on the design and analysis of algorithms related to discrete geometric objects. One main objective is to design algorithms performing with the fastest possible running time, in an asymptotic sense, with respect to the input size. One of the most well-studied objects in Computational Geometry is the Voronoi diagram.

The *Voronoi diagram*[1] of a set S of n simple geometric objects in the plane, called *sites*, is a geometric partitioning structure that reveals proximity information for the input sites. Sites are usually points, line segments or polygons. In case of the ordinary *nearest-neighbor* Voronoi diagram $\mathcal{V}(S)$ every point within a Voronoi region has the same nearest site. More formally, the *Voronoi region* of a site $s \in S$ is defined as follows:

$$\mathrm{VR}(s,S) = \{x \in \mathbb{R}^2 \mid d(x,s) < d(x,s')\ \forall s' \in S \setminus \{s\}\},$$

where $d(x,s) = \min\{d(x,y) \mid y \in s\}$ is the minimum distance between the point x and the site s. The Voronoi diagram $\mathcal{V}(S)$ is formally defined as

$$\mathcal{V}(S) = \mathbb{R}^2 \setminus \bigcup_{s \in S} \mathrm{VR}(s,S).$$

Refer to Figure 1.1 for an example where the sites S are points, see Figure 1.2 where the sites are (disjoint) line segments and Figure 1.3 where the sites are circles (and $d(x,y)$ is the Euclidean metric, respectively). The latter is equivalent to the additively weighted Voronoi diagram of points, where the weights are the circle radii. In the last three cases the Voronoi diagram $\mathcal{V}(S)$ has combinatorial complexity $O(n)$.

[1] Named after the mathematician Georgy Feodosevich Voronoy (1868-1908).

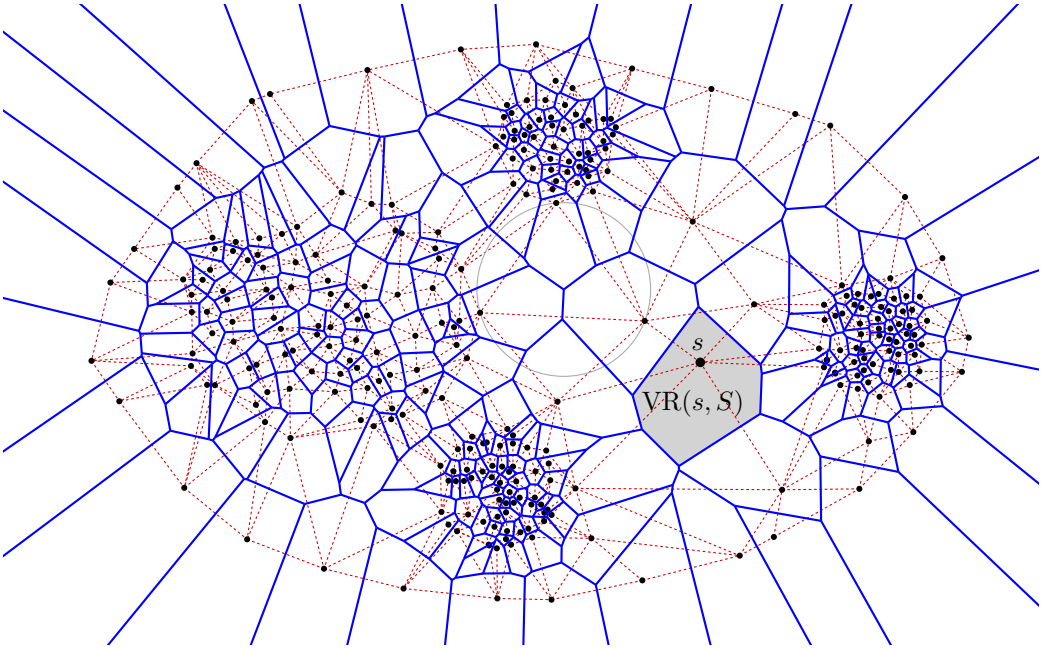

Figure 1.1. A nearest-neighbor Voronoi diagram $\mathcal{V}(S)$ (solid, blue) of point sites S (black) in the Euclidean metric and a Voronoi region $\mathrm{VR}(s,S)$ (shaded, grey). Its dual Delaunay triangulation is shown in dashed, red lines. The circumcircle of each triangle (e.g., the gray circle) is empty of all point sites.

Common variants of the nearest-neighbor Voronoi diagram include the *farthest-site*, and the *order-k* Voronoi diagram of S $(1 \leq k < n)$. Every point within an order-k Voronoi region has the same k nearest sites. For $k = 1$ we have the nearest-neighbor diagram and for $k = n-1$ we have the farthest-site diagram.

More formally, the *farthest Voronoi region* of a site $s \in S$ is

$$\mathrm{FVR}(s) = \{x \in \mathbb{R}^2 \mid d(x,s) > d(x,s'), \forall s' \in S \setminus \{s\}\}$$

and the farthest Voronoi diagram is the subdivision of the plane defined by the farthest Voronoi regions, see Figure 1.4. In the case of point sites, only those sites lying on the boundary of the convex hull of S have non-empty farthest Voronoi regions, all of which are unbounded. The sites in the interior of the convex hull have empty farthest Voronoi regions. Moreover, the sequence of the sites along the convex hull equals the sequence of the unbounded faces at infinity . The farthest Voronoi diagram is a tree and its combinatorial complexity is $O(n)$.

The *order-k Voronoi region* of a subset of sites $H \in S, |H| = k$, is

$$\mathrm{VR}_k(H,S) = \{x \in \mathbb{R}^2 \mid d(x,h) < d(x,s), \forall h \in H, \forall s \in S \setminus H\}$$

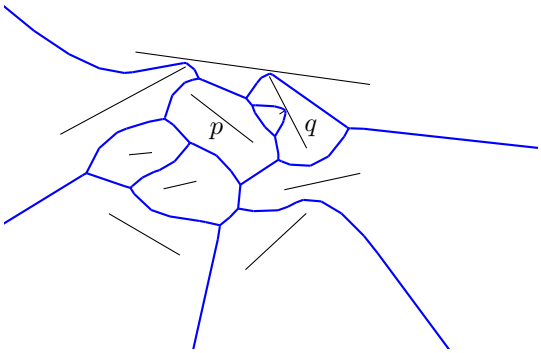

Figure 1.2. A Voronoi diagram $\mathcal{V}(S)$ of line segments as sites.

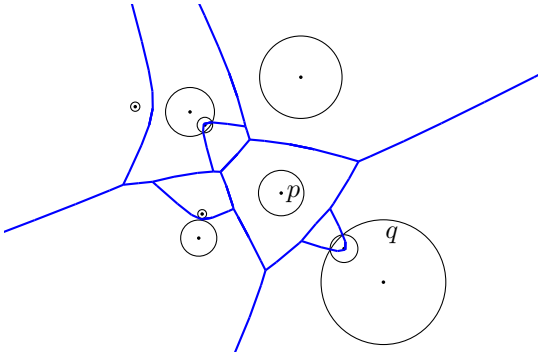

Figure 1.3. A Voronoi diagram $\mathcal{V}(S)$ of circles as sites.

and the order-k Voronoi diagram is the subdivision of the plane defined by the order-k Voronoi regions, see Figure 1.5. The combinatorial complexity of the order-k Voronoi diagram is $O(k(n-k))$.

The concept of Voronoi diagrams finds numerous applications in many fields of science like epidemiology, crystallography, meteorology, geography, social sciences, ecology, biology, physics, in Shannon's communication model, and many more. In geometry, Shamos and Hoey [1975] presented applications of Voronoi diagrams to compute the minimum spanning tree, the largest empty circle and the smalling enclosing circle of a point set. They also gave the first $O(n \log n)$ time construction algorithm for Voronoi diagrams of point sites. Other important geometric applications are point location, roundness computation and reconstruction of curves and surfaces from point clouds. Other areas in computer science that make use of Voronoi diagrams include robotics, computer graphics, wireless networking and machine learning. Refer to the monograph of Okabe, Boots, Sugihara and Chiu [2000] for a broad overview of applications.

Optimal $O(n \log n)$ construction algorithms are also well known for other standard cases like the nearest or farthest Voronoi diagram of points and disjoint line segments in various metrics, see e.g., Aurenhammer, Klein and Lee [2013] for references, extensive information on Voronoi diagrams and their generalizations.

In case of a set of *points* as sites S, the dual structure of the Voronoi diagram $\mathcal{V}(S)$, is the famous *Delaunay triangulation*[2] $\mathrm{DT}(S)$, which is a triangulation of S, such that the circumcircle of each triangle is empty of points from S, see the dashed, red graph in Figure 1.1. The Delaunay triangulation is the straight line dual graph of the Voronoi diagram $\mathcal{V}(S)$, i.e., two sites $p, q \in S$ are connected by

[2]Named after the mathematician Boris Nikolaevich Delaunay (1890 - 1980).

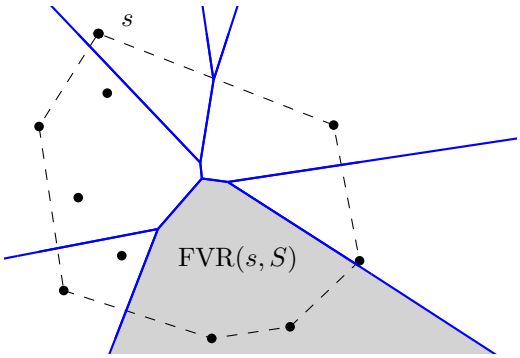

Figure 1.4. A farthest Voronoi diagram $\mathrm{FVD}(S)$ (solid) of point-sites and their convex hull (dashed).

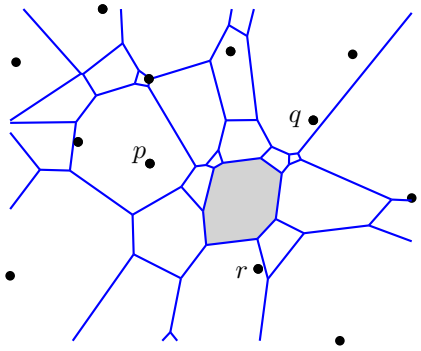

Figure 1.5. An order-3 Voronoi diagram $\mathcal{V}_3(S)$ of point-sites, where $\mathrm{VR}_3(\{p,q,r\},S)$ is shown shaded.

a line segment if and only if their corresponding Voronoi regions are adjacent. See e.g., Aurenhammer et al. [2013] for references and more information. For a triangulation of line segments, which is dual to the Voronoi diagram of line segments, see e.g., Chew and Kedem [1989] and Brévilliers, Chevallier and Schmitt [2007].

Differences of point sites and generalized sites

There are some similarities, but also fundamental differences between point sites and generalized sites, like line segments or circles, which we want to point out in the following (see the monographs of de Berg, Schwarzkopf, van Kreveld and Overmars [2000] and of Aurenhammer, Klein and Lee [2013]).

In case of *points* as sites, the bisector of two sites is a straight line and thus, all the edges of the Voronoi diagram $\mathcal{V}(S)$ are straight line segments (or half lines) and every Voronoi region is convex. Refer to Figure 1.1.

In the case of disjoint *line segments* as sites, the bisector between two sites can consist of up to seven pieces, which are either line segments or parabolic arcs. In the case of *circles* as sites, bisectors are hyperbolas. In both cases each Voronoi region is simply connected and (weakly) star-shaped. Two regions can be adjacent in $\Omega(n)$ edges, but the total complexity of the diagram is still $O(n)$. For example, in Figures 1.2 and 1.3 the Voronoi regions of the sites p and q are adjacent in two Voronoi edges, respectively.

Abstract Voronoi diagrams (AVDs)

Voronoi diagrams for different sites or metrics have often been treated separately, but also several approaches of unifying them exist. Such frameworks allow to establish properties and algorithms for a whole class of Voronoi diagrams at once.

Abstract Voronoi diagrams offer a unifying framework for various concrete and well-known instances of Voronoi diagrams and were introduced by Klein [1989]. For example the sites can be points (Figure 1.1), disjoint line segments (Figure 1.2), non-enclosing circles or equivalently additively weighted point sites with non-enclosing circles (Figure 1.3) or disjoint convex polygons of constant size. The metrics include all L_p metrics, the Karlsruhe metric or other convex distance measures.

In the abstract framework the diagram is not defined via the sites and distances, but from a bisecting curve system satisfying certain combinatorial properties. The bisectors are unbounded *Jordan curves* that partition the plane into two *domains of dominance* and the abstract Voronoi regions are defined by intersections of these domains (see Figures 2.1 and 2.2). The bisectors are assumed to fulfill certain properties for every subset of sites, among them, most commonly, that each Voronoi region is non-empty and connected and all Voronoi regions together cover the plane. For details refer to the axioms in Section 2.1.

A set $\mathcal{J}$ of abstract bisectors fulfilling these axioms is called *admissible*. Assuming an admissible bisector system, the nearest-neighbor abstract Voronoi diagram can be computed in $O(n \log n)$ (deterministic or expected) time, see Klein [1989] and Klein, Mehlhorn and Meiser [1993], respectively. Once it is shown that the bisector system of a concrete Voronoi diagram fulfills the AVD axioms, combinatorial properties and construction algorithms for abstract Voronoi diagrams become directly applicable to the concrete case. In Section 2.1 we review AVDs in more detail.

1.1 Basic properties of Voronoi diagrams and algorithms

In this section we present the most common techniques to construct *point-site* Voronoi diagrams in the Euclidean metric, and the general algorithmic paradigms on which these techniques are based on.

We begin with a short review of the most basic properties of point-site Voronoi diagrams, see Aurenhammer et al. [2013] for proofs and more. From Euler's formula it can be concluded that the Voronoi diagram $\mathcal{V}(S)$ of n points S with n Voronoi regions consists of at most $2n-2$ vertices and $3n-6$ edges. The com-

binatorial complexity of $\mathcal{V}(S)$, which is defined as the sum of all vertices, edges and faces is thus $O(n)$. The number of edges bounding one Voronoi region is less than 6 on average. Every Voronoi region of a site $s \in S$ is convex and a Voronoi region is unbounded if and only if s lies on the boundary of the convex hull of S. Two sites p, q of S are connected by an edge in the Delaunay triangulation of S if and only if their Voronoi regions VR(p) and VR(q) in $\mathcal{V}(S)$ share an edge.

For the following algorithms, it is important to know that there is an $\Omega(n \log n)$ lower bound for the construction time for the Voronoi diagram $\mathcal{V}(S)$, even if the points S are already sorted according to their x-coordinates. This can be shown by reductions from sorting n numbers and from the ε-closeness problem in the horizontally sorted case.

Divide & Conquer. Shamos and Hoey [1975] give the first $O(n \log n)$ time construction algorithm based on the Divide & Conquer paradigm. The sites S are divided by a line into two sets L and R of about the same size, then their Voronoi diagrams $\mathcal{V}(L)$ and $\mathcal{V}(R)$ are computed recursively. In the *Conquer* phase $\mathcal{V}(L)$ and $\mathcal{V}(R)$ are merged into $\mathcal{V}(S)$ in time $O(|S|)$ by computing a *merge curve*, which is the (vertically monotone) polygonal chain consisting of all Voronoi edges of $\mathcal{V}(S)$ that border Voronoi regions from both L and R. The computation of the merge curve is done by tracing the regions of the diagrams $\mathcal{V}(L)$ and $\mathcal{V}(R)$, respectively, without revisiting parts twice, i.e., without so-called *backtracking*. This way of computing the merge curve is often called *clockwise-counterclockwise scan* in the literature.

Plane sweep. There is also a plane sweep algorithm for constructing $\mathcal{V}(S)$, first found by Fortune [1987] and was later further developed. See Aurenhammer et al. [2013] for a nice and detailed presentation of the following sketched algorithm. The *sweep line* is a horizontal line H that moves from left to right over the plane, where H is treated like an extra site. The *beach line* is the bisector between the points left of the sweep line and the sweep line itself. The beach line is a vertically monotone curve made up of parabolic arcs. One can imagine the sweep line to move continuously, however, only at certain discrete events it is necessary to update the Voronoi diagram that is computed so far up to the current beach line. Those *events* are the sites S and the intersections of Voronoi edges (giving rise to a Voronoi vertex). The events and the beach line must be stored in dynamic data structures accordingly and must be updated at each event. The size of the beach line can be shown to be $O(n)$ and there are $O(n)$ many events, each of which can be handled in $O(\log n)$ time, yielding an $O(n \log n)$ algorithm.

Lifting to $\mathbb{R}^3$. The Voronoi diagram $\mathcal{V}(S)$ can also be computed by an elegant lifting transformation to the paraboloid $P = \{(x, y, z) \in \mathbb{R}^3 \mid x^2 + y^2 = z^2\}$ in three-dimensional space. Each site $(x, y) \in S$ in the plane is mapped to x^2+y^2 on P in $\mathbb{R}^3$, yielding the lifted sites S^P. It can be shown that the vertical projection of the (lower) convex hull of S^P back to the plane equals the Delaunay triangulation DT(S). The Voronoi diagram $\mathcal{V}(S)$ can be directly obtained in linear time from DT(S), because it is its dual graph. Since the convex hull of n points in $\mathbb{R}^3$ can be constructed in $O(n \log n)$ time (see, e.g., Preparata and Shamos [1985]), it can be concluded that also $\mathcal{V}(S)$ can be constructed in $O(n \log n)$ time.

Randomized incremental construction (RIC). A Voronoi diagram can also be constructed incrementally by adding one site after the other to the already computed diagram. In a brute force manner, finding the location of the newly added site takes $O(n)$ time and tracing the Voronoi region of the new site again $O(n)$, which leads to a simple $O(n^2)$ time algorithm. However, considering that the average size of a Voronoi region is only 6, a randomization of the insertion order together with a smart way of point location can bring the running time down to expected $O(n \log n)$, see Guibas, Knuth and Sharir [1992] and Boissonnat and Teillaud [1993]. As usual, the algorithm is better described for the dual Delaunay triangulation. The point location can be realized with a *history graph* (also *influence graph* or *Delaunay tree*), which is a rooted directed acyclic graph and offers point location in expected $O(\log n)$ time. The nodes of the history graph are all Delaunay triangles that have ever been created during the algorithm. A triangle that has been destroyed in some step is connected by an edge to the triangles that have been created in this step. The out-degrees and thus, the size of the history graph can be sufficiently bounded, even though it contains many more triangles than the final actual Delaunay triangles. Note that the expected $O(n \log n)$ running time of this algorithm holds independently of the distribution of the point sites. The expectation value is the average of all insertion orders. Only a bad (and unlikely) insertion order can cause quadratic running time, not the input point configuration. See Aurenhammer et al. [2013] for a more detailed presentation of this algorithm.

1.1.1 Randomized incremental constructions - a general framework

The randomized algorithm for constructing Voronoi diagrams and Delaunay triangulations that we have reviewed in the last paragraph (in the end of Section 1.1) is only one example of a vast literature on randomized algorithms in computational geometry.

After Clarkson [1985] introduced his random sampling via the post office problem, this technique was further developed by him and others into a general framework, see Clarkson [1985], Clarkson [1987], Clarkson and Shor [1988], Clarkson and Shor [1989], Sharir [2003]. This framework is based on random incremental insertions of *objects* and uses a *conflict graph*, which stores the (conflicting) relations between the so far constructed geometric objects called *ranges* (e.g., Voronoi regions) and the objects that have not been inserted yet (e.g., sites). Devillers [1996] compares in his survey the conflict graph and influence graph, reviews their properties and some algorithmic applications. See also the monographs of Boissonnat and Yvinec [1998] and Mulmuley [1994] for a detailed discussion. The framework has been modified and applied to various randomized incremental constructions for geometric problems, of which the following list is only a small subset: Boissonnat et al. [1990], Boissonnat and Teillaud [1993], Boissonnat et al. [1992], Seidel [1990], Chew [1990], Guibas et al. [1992], Clarkson et al. [1993], Klein et al. [1993], Mehlhorn et al. [2001].

Seidel [1993] gives an excellent overview of some of the existing RIC algorithms, for which he shows how to analyze their running time with the simple so-called *backward analysis*. We will review backward analysis in Section 2.3.2 in more detail on the example of Chew [1990], who was the first one to perform this kind of analysis, constructing the Voronoi diagram of a convex polygon in expected linear time.

1.2 Linear-time algorithms

As we have seen in Section 1.1, for many Voronoi diagrams there exist $O(n \log n)$ construction algorithms e.g., for point sites and for abstract Voronoi diagrams. On the other side, we have to keep in mind that there is an $\Omega(n \log n)$ lower bound for the construction time for the Voronoi diagram of points – even if the points are already sorted according to their x-coordinates.

However, in some situations where even more information is already available Voronoi diagrams can be computed faster. For several *tree-like* Voronoi diagrams in the plane, linear-time construction algorithms have been well-known to exist.

The first technique was introduced by Aggarwal, Guibas, Saxe and Shor [1989] for the Voronoi diagram of points in convex position, which can be computed in linear time if the order of points along their convex hull is given, see Figure 1.6. Their technique can be used to derive linear-time algorithms for other fundamental tree-structured problems for point sites including the following.

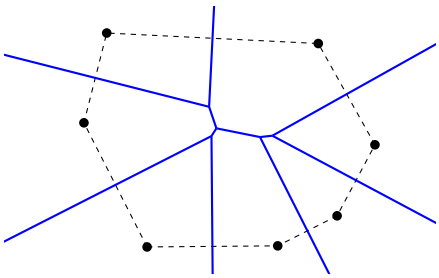

Figure 1.6. A Voronoi diagram of points in convex position and the boundary of their convex hull (dashed).

Tree-like Voronoi problems:

(1) Updating a Voronoi diagram after deletion of one site in optimal time, i.e., linear in the number of the Voronoi neighbors of the deleted site, see Figure 1.7.

(2) Computing the order-(k+1) subdivision within an order-k Voronoi region, i.e., to compute $\mathcal{V}_{k+1}(S)$ within $\mathrm{VR}_k(H,S)$.

(3) Computing the farthest Voronoi diagram in linear time, after the sequence of its faces at infinity is known (or after the convex hull of the sites is known).

There is also a much simpler randomized approach for the same problems introduced by Chew [1990]. This technique can also be adapted to derive simple expected linear-time algorithms for the same problems (1)-(3). The adaptation for the farthest Voronoi diagram of a convex hull appears in more detail in the monograph of de Berg, Cheong, van Kreveld and Overmars [2008].

For abstract Voronoi diagrams as well, there exist linear-time algorithms in special cases. Klein and Lingas [1994] adapted the linear-time framework of Aggarwal et al. [1989] to abstract Voronoi diagrams, under restrictions, showing that a *Hamiltonian abstract Voronoi diagram* can be computed in linear time, given the order of Voronoi regions along an unbounded simple curve, which visits each region *exactly once* and can intersect each bisector only *once*. This construction has been extended recently by Bohler et al. [2014] to include forest structures under similar conditions, where no region can have multiple faces within the domain enclosed by a curve.

Khramtcova and Papadopoulou [2017] gave an expected linear-time algorithm[3] for the concrete farthest-segment Voronoi diagram, in which case one has

[3]A preliminary version (Khramtcova and Papadopoulou [2015]) of their paper contained a

to deal with disconnected Voronoi regions and they presented an innovative way to treat the faces separately.

Recently, Barba [2019] has shown that the geodesic farthest-point Voronoi diagram of points in a simple polygon is also computable in expected linear-time. One issue in this case is the non-constant complexity of geodesic bisectors of point-sites.

The medial axis of a simple polygon is another well-known problem to admit a linear-time construction, shown by Chin et al. [1999] after it had been long-standing open problem. They decompose the polygon in several stages, ending up with xy-monotone histograms, for which they can compute the medial axis in linear time using an extension of the framework of Aggarwal et al. [1989].

Updating the Voronoi diagram of point sites in three dimensions after deletion of a site has been considered by Buchin et al. [2013]. They show how to delete a vertex from the dual structure, the three-dimensional Delaunay triangulation, and compute its update.

In Chapter 2 we review those of the above linear-time algorithms that are most related to this thesis in more detail.

Open problem. For problems (1)-(3), deterministic linear-time solutions so far only exist for points. For generalized (non-point) sites or for abstract Voronoi diagrams, deterministic linear-time construction algorithms for problems (1)-(3) are still unknown, despite serious efforts in the past. Neither for line segments nor circles as sites deterministic linear-time solutions are known. Since 1989 problems (1)-(3) have been well-known open problems for non-point sites and for abstract Voronoi diagrams.

Before the randomized algorithm of Khramtcova and Papadopoulou [2015] for the farthest line-segment Voronoi diagram no *expected* linear-time solutions for any non-point site Voronoi diagram was known for the above three problems. Moreover, prior to this dissertation, for problems (1)-(3), no *expected* linear-time solutions for abstract Voronoi diagrams were known.

non-trivial gap when considering the linear-time framework of Aggarwal et al. [1989], thus, a deterministic linear-time construction for the farthest-segment Voronoi diagram remained an open problem.

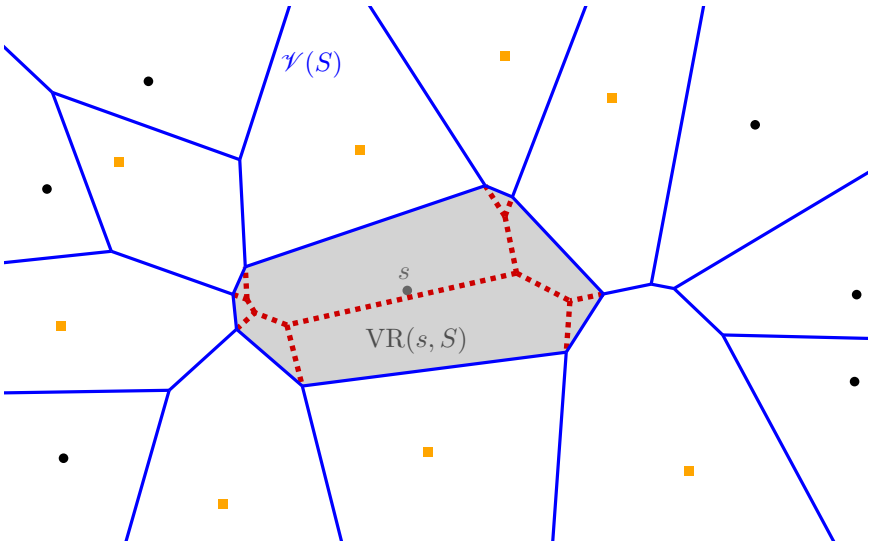

Figure 1.7. The update of $\mathcal{V}(S)$ to $\mathcal{V}(S \setminus \{s\})$ can be obtained by computing the red dashed tree $\mathcal{V}(S \setminus \{s\}) \cap \mathrm{VR}(s,S)$. The Voronoi neighbors of site s are shown as orange squares.

1.3 Focus of this dissertation: Linear-time algorithms (in expected sense) for tree-like diagrams

In this dissertation the major problem we consider is the construction of certain tree-like (two-dimensional) Voronoi diagrams in expected linear time after the sequence of their faces along an enclosing curve is known. A Voronoi diagram within a simply connected domain D is called *tree-like* if its graph structure within a bounded domain D is a tree, or a forest if the domain D is unbounded (see Definition 1). We consider the framework of abstract Voronoi diagrams to simultaneously address the various concrete instances under their umbrella. Our main focus are simple randomized algorithms for the tree-like problems (1)-(3) as inspired by the approach of Chew [1990].

1.3.1 Site deletion

The problem of *site deletion* (1) is probably the most important of the three problems. To update the Voronoi diagram $\mathcal{V}(S)$ after the deletion of site s, we need to compute the Voronoi diagram $\mathcal{V}(S \setminus \{s\}) \cap \mathrm{VR}(s,S)$, which in case of point sites is a tree, see Figure 1.7; for abstract Voronoi diagrams, the diagram $\mathcal{V}(S \setminus \{s\}) \cap \mathrm{VR}(s,S)$ is *tree-like*.

The importance of fast site deletion becomes apparent when considering a dynamic setting, in which after having computed the Voronoi diagram $\mathcal{V}(S)$, either sites are removed from the set S or additional sites are added to S and we want to update the diagram after this removal or addition efficiently. Here we focus only on the time for the insertion of the Voronoi region of s after the location of s in the diagram is known and we do not discuss the problem of efficiently updating a data structure for point location, which is treated for example by Kaplan, Mulzer, Roditty, Seiferth and Sharir [2017].

After the location of a new site is known, updating the diagram after adding the new site can be done by very simple *tracing* in time linear in the size of the added Voronoi region (see e.g., Aurenhammer et al. [2013]). In contrast, the update after deletion of a site is significantly more difficult. Even in the simplest case of point sites under the Euclidean metric the linear-time solution of Aggarwal et al. [1989] is an involved doubly recursive divide-and-conquer algorithm, based on two sophisticated divide phases. However, Chew [1990] offers a much simpler randomized version. For more general sites other challenging issues arise for the problem of site deletion as we will point out in Section 1.3.2.

Site deletion in practice

In the C++ library CGAL [1995] for geometric algorithms, site deletion is implemented via vertex removal in the dual Delaunay triangulation, which uses the well-known triangle flip algorithm of Lawson [1977] to update the diagram after deletion of a site. This algorithm runs in quadratic time, i.e., in time $O(h^2)$, where h is the number of Voronoi neighbors of the deleted site. For $3 \leq h \leq 7$, CGAL uses algorithms of Devillers [2011], who optimizes vertex removal for small degrees that perform well in practice. He underlines the importance of small degrees, because the average degree of a vertex in a Delaunay triangulation is only 6. For segments or other generalized sites, deletion is not implemented in CGAL.

1.3.2 Challenges in computing tree-like Voronoi diagrams of non-point sites with disconnected regions

In the following we point out the difficulties that arise in computing tree-like Voronoi diagrams of generalized (non-point) sites with disconnected regions. We explain the main difficulties in the context of site deletion, however, the same problems occur in the other tree-like Voronoi problems that we investigate in this dissertation.

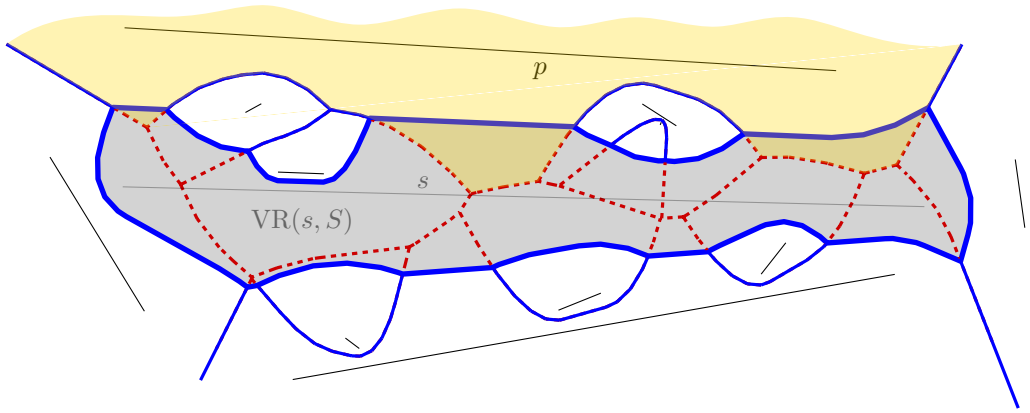

Figure 1.8. A Voronoi diagram of a set of line segments S. The Voronoi region $\mathrm{VR}(p, S \setminus \{s\})$ of one site p (yellow, shaded) can have multiple disconnected faces within $\mathrm{VR}(s, S)$, which is shown shaded, gray. The site p induces three edges along $\partial\mathrm{VR}(s, S)$.

Consider the boundary of a Voronoi region $\mathrm{VR}(s, S)$, denoted $\partial\mathrm{VR}(s, S)$. If the set of sites S are points, then each edge along $\partial\mathrm{VR}(s, S)$ is induced by exactly one site in S, i.e., no repetition can occur. See e.g., the regions of the point sites depicted as orange squares in Figure 1.7, where each site corresponds to exactly one region in the dashed red graph and to exactly one blue Voronoi edge on $\partial\mathrm{VR}(s, S)$.

In contrast, if $\mathcal{V}(S)$ is a Voronoi diagram of more general set of sites like line segments, circles or an abstract Voronoi diagram, then the sequence of site-occurrences along $\partial\mathrm{VR}(s, S)$ can include *repetitions*, i.e, two Voronoi regions $\mathrm{VR}(s, S)$ and $\mathrm{VR}(p, S)$ can be adjacent in several – even in $\Omega(n)$ – Voronoi edges, a fact that we have illustrated already in Figures 1.2 and 1.3 in the beginning of this chapter. In Figure 1.8 $\mathrm{VR}(s, S)$ and $\mathrm{VR}(p, S)$ are adjacent in three Voronoi edges. The boundary $\partial\mathrm{VR}(s, S)$ forms a Davenport-Schinzel sequence[4] of order 2 with respect to site occurrences. This repetition of sites constitutes a major difference from the respective problem for points.

What else does this repetition of sites imply? As shown in the example in Figure 1.8, the diagram $\mathcal{V}(S \setminus \{s\}) \cap \mathrm{VR}(s, S)$ contains *disconnected* Voronoi regions, i.e., for a site $p \in S \setminus \{s\}$, the region $\mathrm{VR}(p, S \setminus \{s\}) \cap \mathrm{VR}(s, S)$ can consist of several

[4] A finite sequence of elements is called a *Davenport-Schinzel sequence* of order d if no consecutive elements are the same and if it does not contain an alternating subsequence $a, b, a, b, \ldots$ of length $d + 2$ of two distinct elements $a \neq b$. The length of a Davenport-Schinzel sequence of order 2 on n distinct values is well known to be bounded by $2n - 1$, see, e.g., Sharir and Agarwal [1995].

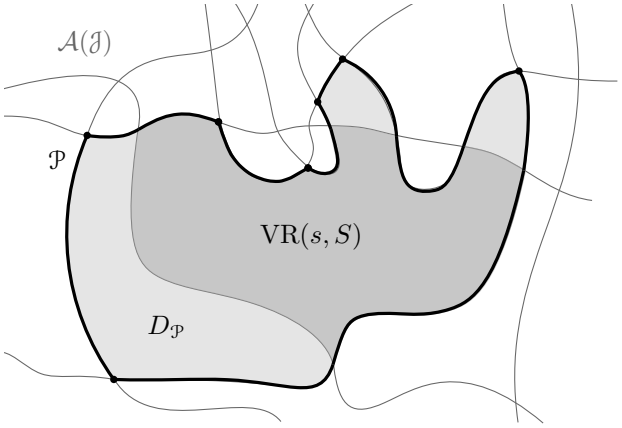

Figure 1.9. A boundary curve $\mathcal{P}$ is a monotone path in the arrangement of abstract bisectors $\mathscr{A}(\mathcal{J})$; its domain $D_{\mathcal{P}}$ contains $\mathrm{VR}(s,S)$.

faces.

The fact of site repetition introduces serious complications. For example, the diagram $\mathcal{V}(S')\cap\mathrm{VR}(s,S'\cup\{s\})$ for a subset $S'\subset S\setminus\{s\}$ may contain various faces that are not related to the diagram $\mathcal{V}(S\setminus\{s\})\cap\mathrm{VR}(s,S)$. Conversely, an arbitrary sub-sequence of the Voronoi edges $\partial\mathrm{VR}(s,S)$ need not correspond to any Voronoi diagram. At first sight, a linear-time algorithm may seem infeasible.

1.4 Summary of the main results of this dissertation

The main contribution of this dissertation is the formulation of an abstract tree-like diagram, within a simply connected domain, called a *Voronoi-like diagram*. It is a relaxed version of a Voronoi diagram, it serves as an intermediate structure that is much simpler to compute and allows to derive simple randomized linear-time techniques for the tree-like problems (1)-(3) (listed in Section 1.2) for abstract Voronoi diagrams.

A Voronoi-like diagram is defined relative to a Voronoi region $\mathrm{VR}(s,S), s\in S$ and it is a subgraph of the arrangement $\mathscr{A}(\mathcal{J})$ of the admissible bisector system $\mathcal{J}$. It allows to efficiently compute $\mathcal{V}(S\setminus\{s\})$ truncated within $\mathrm{VR}(s,S)$, i.e., $\mathcal{V}(S\setminus\{s\})\cap\mathrm{VR}(s,S)$. A Voronoi-like diagram is defined on a *boundary curve* $\mathcal{P}$, which confines a *domain* $D_{\mathcal{P}}$ that is subdivided by the Voronoi-like diagram. In Figure 1.9, the boundary curve is shown bold black and its domain shaded gray. Given a subset of sites $S'\subseteq S$, a *boundary curve* $\mathcal{P}$ is a *monotone path* in the arrangement $\mathscr{A}(\mathcal{J})$, which encloses $\mathrm{VR}(s,S)$, but continues to share common edges with $\partial\mathrm{VR}(s,S)$ as related to S' (see Definitions 3 and 5). Furthermore, there are

many different boundary curves for S', all enclosing $\partial\mathrm{VR}(s,S')$, but being simpler to compute. The Voronoi-like diagram $\mathcal{V}_l(\mathcal{P})$ of a boundary curve $\mathcal{P}$ is defined simply as a subgraph in the arrangement of the bisector system $\mathcal{J}$ whose faces are bounded by *monotone paths* in this arrangement (see Definition 6). In contrast, the boundary of a real Voronoi region is an envelope in the same arrangement. Real Voronoi regions are enclosed in their Voronoi-like counterparts.

We prove that Voronoi-like diagrams are well-defined, i.e., for a given boundary curve $\mathcal{P}$, its Voronoi-like diagram $\mathcal{V}_l(\mathcal{P})$ exists and it is unique. We show that Voronoi-like diagrams are robust under an *insertion operation*, thus, making possible a randomized incremental construction for $\mathcal{V}(S \setminus \{s\}) \cap \mathrm{VR}(s,S)$ in linear time. This solves problem (1) in expected linear time, the update of a Voronoi diagram after deletion of one site. Further, we design randomized linear-time algorithms for computing the order-(k+1) subdivision within an order-k Voronoi region (problem (2)) and computing the farthest Voronoi diagram in linear time, after the sequence of its faces at infinity is known (problem (3)).

Formulating the definition of our Voronoi-like diagram, proving the correctness of the algorithms and finding sufficient properties of the geometric structures has been a long, challenging and non-trivial process.

1.5 Dissertation Outline

After a literature review in Chapter 2, the remaining Chapters 3 – 7 of this dissertation comprise the following topics: the definition of our Voronoi-like diagram, establishing the correctness of an insertion operation in this diagram, the question of the existence and uniqueness of the Voronoi-like structure, its properties, randomized algorithms and other generalizations, and finally the investigation of a possible applicability in the deterministic algorithmic framework of Aggarwal, Guibas, Saxe and Shor [1989]. In the following we give a more detailed overview of the chapters.

Chapter 2 – Literature review

In Section 2.1 we give the definition of Abstract Voronoi diagrams and their higher-order versions and an overview of construction algorithms. In Section 2.2 we give a brief review of the notions of arrangements of Jordan curves and related Davenport-Schinzel sequences. Finally, we give an overview of linear-time algorithms (randomized and deterministic) for certain classes of Voronoi diagrams, concrete and abstract. Among them there are linear-time algorithms

for point-site Voronoi diagrams, the farthest line-segment Voronoi diagram (Section 2.3), and for Abstract Voronoi diagrams in special cases (Section 2.3.3).

Chapter 3 – Voronoi-like diagrams

In Chapter 3 we introduce the concept of a *Voronoi-like diagram*, a relaxed version of a Voronoi construct that has a structure similar to an abstract Voronoi diagram, without however being one.

In Section 3.1 we give some preliminaries and show that the diagram $\mathcal{V}(S \setminus \{s\}) \cap \mathrm{VR}(s, S)$ is *tree-like*. In Section 3.2 we introduce the concepts of a *monotone path*, an *envelope* and a *boundary curve*. We present the definition of a *Voronoi-like diagram* and give basic properties, e.g., that *Voronoi-like regions* are supersets of real Voronoi regions. In Section 3.3 we prove that a Voronoi-like diagram is robust under an *insertion operation*, therefore, enabling its use in incremental constructions of Voronoi diagrams.

Chapter 4 – Existence and Uniqueness

In Chapter 4 we prove that Voronoi-like diagrams are well-defined. In Section 4.1 we prove that if a Voronoi-like diagram exists, then it is unique. Additionally to its uniqueness, in Section 4.2 we prove that, for a given boundary curve $\mathcal{P}$, its Voronoi-like diagram $\mathcal{V}_l(\mathcal{P})$ always exists (via a proof by construction). Combining *existence* with *uniqueness* establishes the Voronoi-like structure as a well-defined geometric partitioning tool that can handle disconnected Voronoi regions.

Chapter 5 – A randomized algorithm

In Chapter 5 we present an easy randomized algorithm for solving the fundamental tree-like problem (1) of site deletion in an abstract Voronoi diagram. However, proving the correctness of the algorithm was a long and challenging process, which includes the formulation of the definitions in previous chapters and the various properties.

In Section 5.1 we show that our diagrams allow to solve the fundamental problem (1) of updating an abstract Voronoi diagram, after deletion of one site, in expected time linear in the number of changes in the diagram (Theorem 7). In this construction algorithm Voronoi-like diagrams serve as intermediate structures, which are considerably simpler and faster to compute than their real Voronoi counterparts.

In Section 5.2 we give the time analysis of the randomized algorithm by partitioning all possible insertion orders into groups of similar permutations and analyze the average work of an insertion within one group.

Chapter 6 – Extensions of the algorithm

In Chapter 6 we extend the algorithm of Chapter 5 to solve the tree-like problems (2) and (3) that we have presented in Section 1.3.

In Section 6.1 we consider problem (2) and we show how our randomized algorithm can be adapted to compute the order-$(k+1)$ subdivision within the face of an order-k abstract Voronoi region in expected time linear in the complexity of the region's boundary. The non-trivial proof process established new properties of the Voronoi-like structures that are interesting in their own right. First, in Section 6.1.1 we point out the relation between different boundary curves and their Voronoi-like structures and show how we can use one in the place of another to end up with simple expected linear-time construction algorithms. Then, in Section 6.1.2 we use the relations of the previous section in order to tackle problem (2) for abstract Voronoi diagrams, resulting in a surprisingly simple algorithm (Theorem 9). This reduces by a logarithmic factor the time complexity of the iterative construction to compute the order-k abstract Voronoi diagram i.e., from $O(k^2 n \log n)$ to $O(k^2 n + n \log n)$ steps.

In Section 6.2, we show how our approach can be adapted to compute in expected linear time the farthest abstract Voronoi diagram, after the sequence of its faces at infinity is known (Theorem 10).

Chapter 7

In Chapter 7 we investigate the application of Voronoi-like diagrams in the linear-time framework of Aggarwal, Guibas, Saxe and Shor [1989] to possibly solve the problem of site deletion (and the other tree-like problems) in *deterministic* linear time – a well-known and still open problem.

We formulate three challenges towards this application: the formulation of a coloring rule for the first divide phase, establishing a combinatorial lemma for the second divide phase and merging two Voronoi-like diagrams in linear time.

The solutions of these open problems would yield significant progress towards a deterministic linear-time algorithm.

1.6 Publications

The following is a list of papers that have been written within my time as a PhD student.

- Kolja Junginger and Evanthia Papadopoulou. Deletion in Abstract Voronoi Diagrams in Expected Linear Time. In *34th International Symposium on Computational Geometry* (SoCG 2018), Dagstuhl, Germany.

  - Journal version is submitted and under review.
  - Preliminary version:
    Kolja Junginger and Evanthia Papadopoulou. Deletion in Abstract Voronoi diagrams in expected linear time. In *34th European Workshop on Computational Geometry* (EuroCG 18), Berlin, Germany, March 21 - 23, 2018.
  - arXiv longer version (including the farthest abstract Voronoi diagram):
    Kolja Junginger and Evanthia Papadopoulou. *Deletion in abstract Voronoi diagrams in expected linear time*, CoRR abs/1803.05372. URL: `http://arxiv.org/abs/1803.05372`

  This publication appears in Chapters 3, 4 and 5.

- Kolja Junginger and Evanthia Papadopoulou. Abstract tree-like Voronoi diagrams in expected linear time. In *CGWeek Young Researchers Forum 2019*, Computational Geometry week, Portland, Oregon, 2019.

  - Conference version ready for submission.
  - Journal version in preparation.

  This publication appears in Chapters 4 and 6.

- Kolja Junginger, Ioannis Mantas and Evanthia Papadopoulou. On Selecting Leaves with Disjoint Neighborhoods in Embedded Trees, *Conference on Algorithms and Discrete Applied Mathematics*, Kharagpur, India, 2019.

  - Journal version submitted to *Discrete Applied Mathematics*.

  This publication is not part of this dissertation. However, we cite the main result of this publication in Chapter 7 (Theorem 11).

- Kolja Junginger, Ioannis Mantas, Evanthia Papadopoulou, Martin Suderland and Chee Yap. *Certified Approximations Algorithms for the Fermat Points and k-Ellipses. 36th European Workshop on Computational Geometry* (EuroCG 18), Würzburg, Germany, March 16 – 18, 2020.

  This publication is not part of this dissertation.

Chapter 2

Literature review

In this chapter we review the most fundamental literature and definitions, which form the basis for this dissertation.

In Section 2.1 we give the definition of Abstract Voronoi diagrams and their higher-order versions and an overview of construction algorithms. These definitions are the basis for the rest of this thesis, since all results presented in this dissertation are given in the setting of Abstract Voronoi diagrams.

In Section 2.2 we give a brief review of the notions of arrangements of Jordan curves and related Davenport-Schinzel sequences. Our Voronoi-like diagrams presented in Chapter 3 are defined on arrangements and thus make use of these concepts.

Finally, in Section 2.3 we give an overview of linear-time algorithms (randomized and deterministic) for certain classes of Voronoi diagrams, concrete and abstract. We summarize linear-time algorithms for point-site Voronoi diagrams (Sections 2.3.1, 2.3.2) and the farthest line-segment Voronoi diagram (Sections 2.3.4). In Section 2.3.3 we review linear-time algorithms for Abstract Voronoi diagrams in special cases. This overview of the most important and most related known linear-time algorithms allows the reader to classify the contributions of this dissertation. With the results given later in this thesis we further contribute to the existing linear-time algorithms for Voronoi diagrams.

2.1 Abstract Voronoi diagrams

Abstract Voronoi diagrams (AVDs) were introduced by Klein [1989]. They offer a unifying framework for various concrete and well-known instances of Voronoi diagrams.

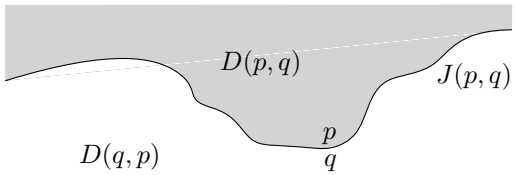

Figure 2.1. A bisector $J(p,q)$ and its dominance regions; $D(p,q)$ is shown shaded.

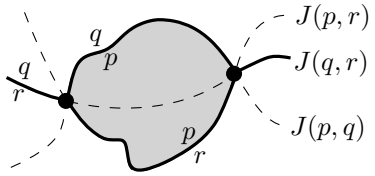

Figure 2.2. The Voronoi diagram $\mathcal{V}(\{p,q,r\})$ in solid lines. The shaded region is $\mathrm{VR}(p,\{p,q,r\})$.

Instead of sites and distance measures, they are defined in terms of a system of bisecting curves, homeomorphic to unbounded lines, that satisfy some simple combinatorial properties. Given a set S of n abstract sites (indices), the bisector $J(p,q)$ of two sites $p,q \in S$ is an unbounded Jordan curve, homeomorphic to a line, that divides the plane into two open domains: the *dominance region of* p, $D(p,q)$ (having label p), and the *dominance region of* q, $D(q,p)$ (having label q), see Figure 2.1.

The *Voronoi region* of p is

$$\mathrm{VR}(p,S) = \bigcap_{q \in S \setminus \{p\}} D(p,q).$$

The *(nearest-neighbor) abstract Voronoi diagram* of S is

$$\mathcal{V}(S) = \mathbb{R}^2 \setminus \bigcup_{p \in S} \mathrm{VR}(p,S).$$

Following the traditional model of abstract Voronoi diagrams (see e.g. Klein [1989]; Bohler et al. [2015]; Bohler, Liu, Papadopoulou and Zavershynskyi [2016]; Bohler, Klein and Liu [2016]) the system of bisectors is assumed to satisfy the following axioms, for every subset $S' \subseteq S$:

(A1) Each nearest Voronoi region $\mathrm{VR}(p,S')$ is non-empty and pathwise connected.

(A2) Each point in the plane belongs to the closure of a nearest Voronoi region $\mathrm{VR}(p,S')$.

(A3) After stereographic projection to the sphere, each bisector can be completed to a Jordan curve through the north pole.

(A4) Any two bisectors $J(p,q)$ and $J(r,t)$ intersect transversally and in a finite number of points. (It is possible to relax this axiom, see Klein et al. [2009]).

Assuming axioms (A1)-(A4), $\mathcal{V}(S)$ is a plane graph of structural complexity $O(n)$ and its regions are simply-connected. It can be computed in $O(n \log n)$ expected time, with a randomized incremental construction, see Klein, Mehlhorn and Meiser [1993]. Moreover, $\mathcal{V}(S)$ can be computed in deterministic $O(n \log n)$ time, if acyclic partitions of S are available ([Klein, 1989, Chapter 3]), which applies to many concrete cases, see [Klein, 1989, Chapter 4]. A partition (L,R) of the abstract sites S is called *acyclic*, if the borderline consisting of Voronoi edges separating the Voronoi regions of sites in L from the Voronoi regions of sites in R does not contain any cycles.

Examples of concrete Voronoi diagrams that fall under the AVD umbrella include: disjoint line segments and disjoint convex polygons of constant size in the L_p norms, or under the Hausdorff metric; point sites in any convex distance metric or the Karlsruhe metric; additively weighted points that have non-enclosing circles; power diagrams with non-enclosing circles. See Bohler, Liu, Papadopoulou and Zavershynskyi [2016].

Higher-order and the farthest abstract Voronoi diagram

The above axiomatic framework can also be used to define the farthest abstract Voronoi diagram, see Mehlhorn, Meiser and Rasch [2001], where it was investigated for the first time.

The *farthest abstract Voronoi region* of a site $p \in S$ is defined as

$$\mathrm{FVR}(p,S) = \bigcap_{q \in S \setminus \{p\}} D(q,p).$$

The *farthest abstract Voronoi diagram* of S is defined as

$$\mathrm{FVD}(S) = \mathbb{R}^2 \setminus \bigcup_{p \in S} \mathrm{FVR}(p,S).$$

It is a tree of complexity $O(n)$, however, regions may be disconnected and a farthest Voronoi region may consist of $\Theta(n)$ disjoint faces. It can be constructed in expected $O(n \log n)$ time by a randomized incremental construction as Mehlhorn, Meiser and Rasch [2001] have shown.

The *order-k abstract Voronoi diagram* was first introduced by Bohler, Cheilaris, Klein, Liu, Papadopoulou and Zavershynskyi [2015]. For a subset of sites $H \subset S$, $|H| = k$, the *order-k abstract Voronoi region* is defined as

$$\mathrm{VR}_k(H,S) = \bigcap_{q \in H, p \in S \setminus H} D(q,p).$$

The *order-k abstract Voronoi diagram* of S is defined as

$$\mathrm{V}_k(S) = \mathbb{R}^2 \setminus \bigcup_{H \subset S, |H|=k} \mathrm{VR}_k(H, S).$$

Note that $\mathrm{V}_1(S) = \mathcal{V}(S)$ and $\mathrm{V}_{n-1}(S) = \mathrm{FVD}(S)$. The combinatorial complexity of $\mathrm{V}_k(S)$ is $O(k(n-k))$. More precisely, Bohler et al. [2015] proved that the number of faces of $\mathrm{V}_k(S)$ has an upper bound of $2k(n-k)$ and a lower bound of $n-k+1$, which are both tight.

The order-k abstract Voronoi diagram can be computed with the *iterative construction* of Lee [1982] in time $O(k^2 n \log n)$, by first computing $\mathcal{V}_1(S)$, then computing $\mathcal{V}_2(S)$ by constructing in each face the order-2-subdivision, etc. until $\mathcal{V}_k(S)$. Bohler, Liu, Papadopoulou and Zavershynskyi [2016] explain the adaption of Lee's algorithm to abstract Voronoi diagrams. The iterative construction is particularly useful for small k and e.g., if the diagrams of all orders up to k are needed.

For larger k, other algorithms might be preferred. Bohler, Liu, Papadopoulou and Zavershynskyi [2016] show how the order-k abstract Voronoi diagram can be constructed in expected $O(kn^{1+\varepsilon})$ by a randomized divide-and-conquer algorithm and Bohler, Klein and Liu [2016] give a randomized incremental algorithm with $O(k(n-k)\log^2 n + n\log^3 n)$ expected running time.

2.2 Arrangements and Davenport-Schinzel sequences

Our Voronoi-like diagrams presented in Chapter 3 are defined on arrangements of an admissible bisector system. In this chapter we review the more general notion of *arrangements of Jordan curves* and *Jordan arcs*. Moreover, we give a brief review of *Davenport-Schinzel sequences*, which are used to analyze the complexity of the faces of such arrangements.

2.2.1 Davenport-Schinzel sequences

Davenport and Schinzel [1965] consider homogeneous linear differential equations, the solution of which are functions with bounded algebraic degree, and they study the point-wise maximum of this set of functions. From a geometric viewpoint the maximum that they analyze is the upper envelope of a set of n x-monotone curves in the plane. They ask for the maximum length of such sequences (defined by the functions on the upper envelope), which can be studied purely combinatorially and which have many geometric applications.

More formally, for $n, d \in \mathbb{N}$, they consider a sequence W over a set of symbols A of size n. The sequence W is called *Davenport-Schinzel sequence* or short a *DSS of order d over n symbols*, if it satisfies the following properties:

- No two consecutive elements are the same.
- There is no alternating subsequence of length $d+2$, i.e., the sequence W does not contain a subsequence $\ldots, a, \ldots, b, \ldots, a, \ldots, b, \ldots$ of length $d+2$ for any two symbols $a, b \in A$.

Denote by

$$\lambda_d(n) = \max\{|W| \,:\, W \text{ is a DSS of order } d \text{ over } n \text{ symbols}\} \tag{2.1}$$

the maximum length of a DSS of order d over n symbols.

In the following we review some of the bounds of $\lambda_d(n)$, which involves the *inverse Ackermann function* $\alpha(n)$ (see e.g., Sharir and Agarwal [1995]). The function $\alpha(n)$ is nondecreasing and unbounded, but extremely slowly growing. For all practical values of n, it holds $\alpha(n) \leq 4$ (up to a value n, which is an exponential tower of height 65536 made up of 2's).

Clearly, $\lambda_1(n) = n$, and further Davenport and Schinzel [1965] show that

$$\lambda_2(n) = 2n - 1 \text{ and}$$

$$\lambda_3(n) = O(n \log n).$$

Later, it was shown that

$$\lambda_3(n) = \Theta(n \cdot \alpha(n)),$$

$$\lambda_4(n) = \Theta(n \cdot 2^{\alpha(n)}).$$

Moreover, upper bounds for $d \geq 4$ have been established, all of which are near linear for fixed d, but a superlinear lower bound exists for $d = 3, 4$ and all even orders $d \geq 4$, see Sharir and Agarwal [1995], Hart and Sharir [1986], Agarwal, Sharir and Shor [1989].

2.2.2 Arrangements of Jordan curves

In a more general geometric setting, Sharir and Agarwal [1995] analyze a set $\mathcal{J} = \{\gamma_1, \ldots, \gamma_n\}$ of n *Jordan arcs* or *Jordan curves*, each pair of which intersect transversally in at most d points. The Jordan curves can be closed or unbounded

curves separating the plane into two domains. Jordan arcs are continuous pieces of Jordan curves.

The *arrangement* $\mathcal{A}(\mathcal{J})$ of $\mathcal{J}$ is the planar subdivision induced by the arcs of $\mathcal{J}$ (Sharir and Agarwal [1995]). More precisely, they define $\mathcal{A}(\mathcal{J})$ as a plane graph consisting of the following vertices, edges and faces: its vertices are the endpoints of the arcs of $\mathcal{J}$ and their pairwise intersection points, its edges are maximal (relatively open) connected pieces of the arcs $\mathcal{J}$ that do not contain a vertex, its faces are the connected components of $\mathbb{R}^2 \setminus \mathcal{J}$. The *combinatorial complexity* of a face is the number of vertices (or edges) on its boundary, and the combinatorial complexity of $\mathcal{A}(\mathcal{J})$ is the total complexity of all of its faces, which is clearly $O(dn^2)$, see Sharir and Agarwal [1995].

However, the combinatorial complexity of a single face is much smaller. In the general case of Jordan arcs it is $O(\lambda_{d+2}(n))$ (by [Sharir and Agarwal, 1995, Theorem 5.3]) and in the case that $\mathcal{J}$ is a collection of closed or unbounded Jordan curves it is $O(\lambda_d(n))$ (by [Sharir and Agarwal, 1995, Theorem 5.7]).

Applying the last result to the arrangement $\mathcal{A}(\mathcal{J}(S))$, where $\mathcal{J}(S)$ is an admissible bisector system of unbounded Jordan curves from Section 2.1, and considering that related bisectors can intersect at most twice [Klein, 1989, Lemma 3.5.2.5] we can conclude that the boundary of an abstract Voronoi region $\mathrm{VR}(s,S)$ forms a Davenport-Schinzel sequence of order $d = 2$. Thus, the complexity of $\partial\mathrm{VR}(s,S)$ (the number of its Voronoi edges) is at most $2|S_s| - 1$, where S_s is the set of sites that induce Voronoi edges on $\partial\mathrm{VR}(s,S)$.

2.3 Linear-time algorithms for Voronoi diagrams

In this section we give an overview of linear-time algorithms (randomized and deterministic) for some classes of Voronoi diagrams, concrete and abstract. We review (expected) linear-time algorithms for point-site Voronoi diagrams in Sections 2.3.1 and 2.3.2, the farthest line-segment Voronoi diagram in Section 2.3.4 and for Abstract Voronoi diagrams in special cases in Section 2.3.3.

2.3.1 A deterministic linear-time algorithm for points

In 1989 Aggarwal et al. [1989] introduced a technique to compute the Voronoi diagram of points in convex position in linear time, given the order of points along their convex hull. It can be used to derive linear-time algorithms for other fundamental problems:

(1) updating a Voronoi diagram of points after deletion of one site in time linear in the number of the Voronoi neighbors of the deleted site;

(2) computing the order-(k+1) subdivision within a face of an order-k Voronoi region in time linear in the complexity of the region boundary.

(3) computing the farthest Voronoi diagram of point-sites in linear time, after computing their convex hull;

Figure 1.7 illustrates problem (1): the update after deletion of a site s .

In the following we sketch the algorithmic framework for the application, where the sites S are in convex position and their cyclic sequence along the convex hull is known, see Figure 1.6. The linear-time algorithm is a doubly recursive divide-and-conquer algorithm based on a *colouring rule*. In a first phase the sites S are coloured into *red* R and *blue* B, such that the number of blue sites is roughly equal to the number of red sites and such that the red sites fulfill the following disjointness property: When inserting the Voronoi regions of each two consecutive red sites r_1, r_2 in the blue diagram, their regions in $\mathcal{V}(B \cup \{r_1, r_2\})$ are not adjacent. Then the Voronoi diagram of the blue sites ($B = S \setminus R$) is computed recursively in linear time. With the help of the blue Voronoi diagram $\mathcal{V}(B)$ the red sites are further subdivided into *garnet* and *crimson* such that the number of crimson sites is at least a constant fraction of the red sites; and such that the crimson sites have all pairwise disjoint regions when they are inserted into the blue diagram. This second divide phase of the red sites is realized by a *combinatorial lemma*. The disjointness of the crimson regions allows that all of them can be inserted one by one into the blue diagram in total linear time. The diagram of the garnet sites is again recursively computed in linear-time and finally merged with the diagram of the blue and crimson sites in linear time. Since the combinatorial lemma ensures that the crimson sites are a constant fraction of all red sites the runtime of the algorithm obeys the formula $T(n) = T(qn)+O(n), q < 1$, and is thus linear in $n = |S|$.

In Chapter 7 we give pseudo-code for the above sketched linear-time framework of Aggarwal et al. [1989].

The problems (1)–(3) of this section have been solved in deterministic linear time only for point-sites. Neither for line segments nor circles as sites a deterministic linear time algorithm is known. Since 1989 these fundamental problems have remained open for non-point sites and for abstract Voronoi diagrams.

2.3.2 A simple expected linear-time algorithm for points

Chew [1990] gave a much simpler randomized algorithm applicable to the same problems (1)–(3) in Section 2.3.1. To analyze the runtime of his randomized algorithm Chew is, according to Seidel [1993], probably the first one who uses the famous backward analysis in computational geometry.

Chew's algorithm for computing the Voronoi diagram of points in convex position (given their convex hull) can be sketched as follows. Refer to Figure 1.6. The initial cyclic sequence of Voronoi regions is given by the convex hull; in phase 1 the sites are deleted in a random order while the neighbor of the deleted site on the current convex hull is stored at the time of deletion. In a second phase the algorithm inserts the Voronoi regions of the sites in reverse order. When inserting a site in phase 2, it is known from phase 1 where in the diagram its region is located. See [de Berg et al., 2000, Chapter 7.4] for a more detailed description.

Let us analyze the running time of this algorithm. In the following we review backward analysis in case of this algorithm and explain how this simple trick serves to conclude that one insertion takes expected constant time. Seidel [1993] gives a nice overview of backward analysis for the above algorithm as well as for various applications to other problems in computational geometry.

Let's say at step i during phase 2 we insert site s into the Voronoi diagram $\mathcal{V}(S' \setminus \{s\})$ and obtain $\mathcal{V}(S')$ ($i = |S'|$). Since the location of the inserted Voronoi region is known in advance (from phase 1), the time for inserting one region is proportional to its size, i.e., to the number of Voronoi edges bounding this region. Now we focus on the question what the *expected* size of this region is. To answer this, we view the algorithm *backwards*, going from step i to step $i-1$. In the backwards view we go from $\mathcal{V}(S')$ to $\mathcal{V}(S' \setminus \{s\})$. The time needed for step i is the expected size of $\mathrm{VR}(s, S')$, which is the expected size of a region in $\mathcal{V}(S')$ that is chosen uniformly at random. Since the diagram $\mathcal{V}(S')$ has linearly many edges, to be precise $2i-3$ edges, the expected size of $\mathrm{VR}(s, S')$ is twice this number, divided by the number of Voronoi regions, which in turn is $i = |S'|$. Thus, the expected size of $\mathrm{VR}(s, S')$ is $2(2i-3)/i < 4 = O(1)$. We conclude that the expected time for one insertion step is constant and thus, the algorithm for n sites runs in expected $O(n)$ time.

2.3.3 Linear-time algorithms for abstract Voronoi diagrams in special cases

In the setting of abstract Voronoi diagrams linear-time algorithms are known as well, however, under restrictions. The first example is the *Hamiltonian abstract*

Voronoi diagram by Klein and Lingas [1994] who adapted the linear-time framework of Aggarwal et al. [1989] to abstract Voronoi diagrams, under the following restrictions. They assume to be given an unbounded simple curve H, such that

- H visits each Voronoi region *exactly once* and intersects each bisector *exactly once*.

Given the order of Voronoi regions along H they show that under these restrictions the Hamiltonian abstract Voronoi diagram within the domain enclosed by H can be computed in deterministic linear time. The same result applies to the case when H is a bounded simple curve with the above property.

Bohler, Klein, Lingas and Liu [2018] recently extended this construction to *forest-like abstract Voronoi diagrams* under similar conditions. Within a domain D enclosed by a simple closed curve they require the following property.

- $\mathcal{V}(S) \cap D$ is a tree. For all $S' \subseteq S, \mathcal{V}(S') \cap D$ is a forest and each Voronoi region has exactly one face.

This condition implies properties similar to the Hamiltonian AVDs: Each bisector intersects D exactly once and any two bisectors, $J(p,q), J(p,r)$ with a common site intersect at most once within D. Further, ∂D visits each region exactly once. No Voronoi region can have multiple faces within the domain D.

Open problem. Removing the above restrictions and still have a deterministic linear-time algorithm is a highly non-trivial open problem and directly related to the open problems (1)-(3) presented in Section 1.2.

In this dissertation we make progress towards this goal by designing an *expected* linear-time algorithm for the related problems (1)-(3) and abstract Voronoi diagrams, see Chapters 5 and 6.

2.3.4 A randomized algorithm for the farthest line-segment Voronoi diagram

Recently, Khramtcova and Papadopoulou [2017] adapted the expected linear-time algorithm of Chew [1990] and made it applicable to the computation of the farthest line-segment Voronoi diagram, if the sequence of its faces at infinity (or a big enclosing circle) is known.

First, we define the farthest line-segment Voronoi diagram and give some basic properties. Here the sites are a set of line segments S (that are allowed to

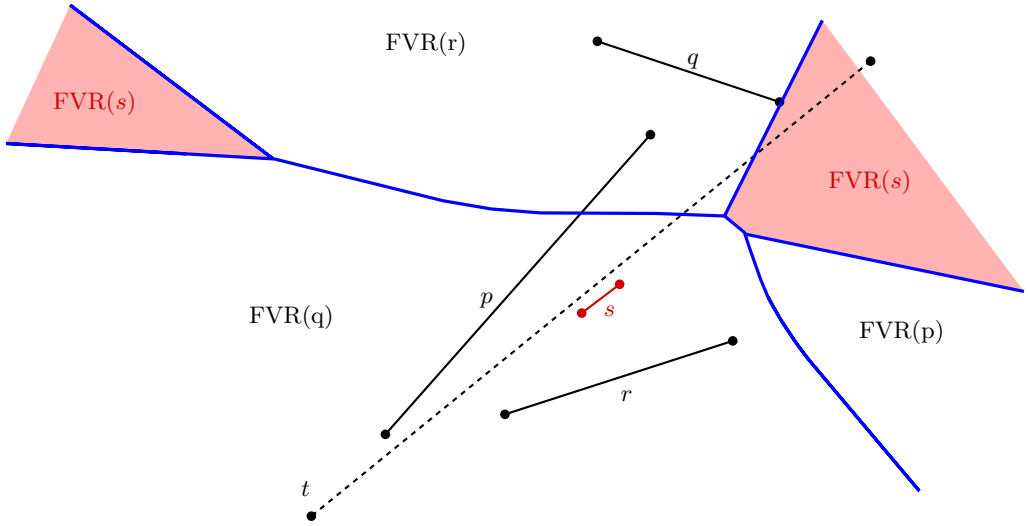

Figure 2.3. A farthest line-segment Voronoi diagram (blue) with a disconnected Voronoi region FVR(s) (red, shaded) and an empty Voronoi region FVR(t).

touch and intersect)[1]. The distance between a point x and a line segment s is $d(x,s) = \min\{d(x,y) \mid y \in s\}$ and the *farthest Voronoi region* of a segment $s \in S$ is $\mathrm{FVR}(s) = \{x \in \mathbb{R}^2 \mid d(x,s) > d(x,s'), s' \in S \setminus \{s\}\}$. The *farthest line-segment Voronoi diagram* $\mathrm{FVD}(S)$ is the graph subdividing the plane into the farthest Voronoi regions. No feature of the intersections of the sites is part of the diagram $\mathrm{FVR}(s)$. A single Voronoi region can have $\Omega(n)$ faces, but the complete diagram is still of size $O(n)$ and has a tree structure. The convex hull of the line segments S is not related to the diagram and a segment can have an empty Voronoi region, see Figure 2.3. The diagram $\mathrm{FVD}(S)$ can be constructed in time $O(n \log n)$, even if the line segments S touch or intersect. Refer to Aurenhammer, Drysdale and Krasser [2006] for details on these properties and algorithms.

Khramtcova and Papadopoulou [2017] contributed with their result further to the existing linear-time constructions for points by handling non-point sites and disconnected Voronoi regions[2]. In their algorithm, the (disconnected) faces are inserted one-by-one in contrast to inserting complete (connected) Voronoi

[1] In case of *intersecting* line-segments, the farthest line-segment Voronoi diagram is not part of the farthest abstract Voronoi diagram framework (their bisectors are not Jordan curves).

[2] The time analysis of their randomized algorithm is based on a questionable backward time analysis. However, the expected linear running time is correct as we have shown recently (not published yet). The algorithm of Junginger and Papadopoulou [2018a] for site deletion in Abstract Voronoi diagrams referred to and was based on the time analysis of Khramtcova and Papadopoulou [2017]. We revise this time analysis in Chapter 5.

regions in the case of point-sites. The realization of the insertion of the faces is an innovative feature. It creates tree-like diagrams in the intermediate steps of the algorithm, which are not instances of any type of Voronoi diagram. These intermediate diagrams are related to our *Voronoi-like diagrams* that we introduce in Chapter 3.

However, in their case definitions are geometric, relying on star-shapeness and visibility properties, which are specific to line-segment Voronoi regions and do not extend to the model of abstract Voronoi diagrams that we consider.

2.4 Conclusion

In this chapter we have seen the definition of *abstract Voronoi diagrams*, which is fundamental for the rest of this thesis, since all results are presented in the setting of an admissible bisector system of Jordan curves. Our Voronoi-like diagrams presented in Chapter 3 are defined on *arrangements* of these abstract bisectors, and thus make use of the notion of arrangements of Jordan curves that we have reviewed as well.

We have given an overview of (randomized and deterministic) *linear-time algorithms* for certain classes of Voronoi diagrams, among them concrete point-sites and the farthest line-segment Voronoi diagrams as well as abstract Voronoi diagrams. We have again underlined the challenging long-standing open problem of a linear-time algorithm for tree-like abstract Voronoi diagrams without further restrictions on the bisector system.

Being inspired by the ideas of the intermediate (non-Voronoi) tree structures in the randomized algorithm of Khramtcova and Papadopoulou [2017], in Chapter 3 we will generalize and adapt them to the abstract setting in order to define the most important object of this thesis, the Voronoi-like diagram. Using this diagram we will be able to further contribute to the existing linear-time algorithms for Voronoi diagrams by designing randomized linear-time algorithms for abstract Voronoi diagrams in Chapters 5 and 6.

Chapter 3

The definition of Voronoi-like diagrams and an insertion operation

In this chapter we introduce the Voronoi-like diagram, a relaxed version of a Voronoi construct that has a structure similar to an abstract Voronoi diagram, without however being one. A Voronoi-like diagram is defined on a *boundary curve* enclosing a simply connected domain. A Voronoi-like diagram is defined relative to a Voronoi region $\mathrm{VR}(s,S), s \in S$ and it is a subgraph of the arrangement of the admissible bisector system. *Voronoi-like regions* are supersets of real Voronoi regions, and their boundaries correspond to *monotone paths* in the arrangement of related bisectors, rather than to *envelopes* as in a real Voronoi diagram. It allows to efficiently compute $\mathcal{V}(S \setminus \{s\})$ truncated within $\mathrm{VR}(s,S)$, i.e., $\mathcal{V}(S \setminus \{s\}) \cap \mathrm{VR}(s,S)$.

In Section 3.1 we give some preliminaries and analyze the structure of the diagram $\mathcal{V}(S \setminus \{s\}) \cap \mathrm{VR}(s,S)$, showing that it is *tree-like*.

In Section 3.2 we introduce the concepts of a *monotone path*, an *envelope* and a *boundary curve*. We present the definition of a *Voronoi-like diagram* and give basic properties.

In Section 3.3 we prove that a Voronoi-like diagram is robust under an *insertion operation*, therefore, enabling its use in incremental constructions of Voronoi diagrams.

This chapter is based on the following publication:

Kolja Junginger and Evanthia Papadopoulou. Deletion in Abstract Voronoi Diagrams in Expected Linear Time. In *34th International Symposium on Computational Geometry* (SoCG 2018), Dagstuhl, Germany.

We also published a corresponding longer version with all proofs on arXiv:

Kolja Junginger and Evanthia Papadopoulou. *Deletion in abstract Voronoi dia-*

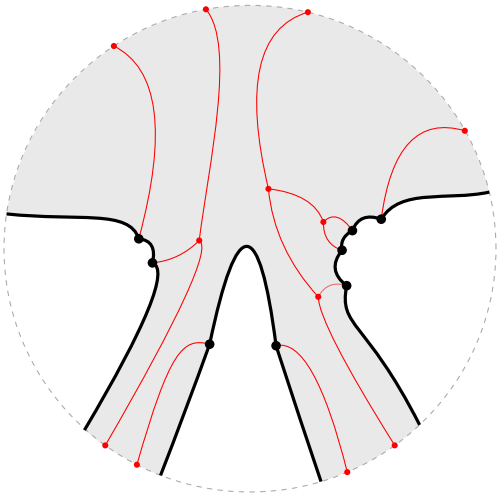

Figure 3.1. $\mathcal{V}(S \setminus \{s\}) \cap \mathrm{VR}(s,S)$ in red, where $\mathrm{VR}(s,S)$ is unbounded (gray, shaded) and its boundary is shown in black bold.

grams in expected linear time, CoRR abs/1803.05372. URL: `http://arxiv.org/abs/1803.05372`

3.1 Preliminaries

Given a set S of n abstract sites (indices), the bisector $J(p,q)$ of two sites $p,q \in S$ is an unbounded Jordan curve, which is homeomorphic to a line and divides the plane into two open domains: the *dominance region of* p, $D(p,q)$ (having label p), and the *dominance region of* q, $D(q,p)$ (having label q), see Figure 2.1.

Given such a system of bisectors the *Voronoi region* of site p is defined as the common intersections of its corresponding dominance regions, i.e.,

$$\mathrm{VR}(p,S) = \bigcap_{q \in S \setminus \{p\}} D(p,q).$$

All the Voronoi regions together define a subdivision of the plane, the *Voronoi diagram* of S, more formally defined as

$$\mathcal{V}(S) = \mathbb{R}^2 \setminus \bigcup_{p \in S} \mathrm{VR}(p,S),$$

see, e.g., Figure 2.2.

We always assume that the system of bisectors $\mathcal{J} = \{J(p,q) : p \neq q \in S\}$ is *admissible*, i.e., it satisfies axioms (A1)–(A4) of Section 2.1 for every subset

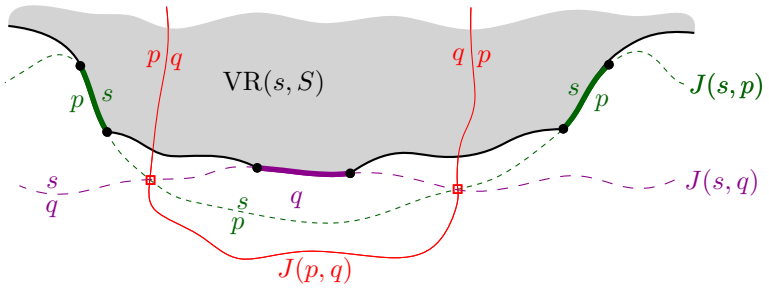

Figure 3.2. $\mathrm{VR}(p, S \setminus \{s\}) \cap \mathrm{VR}(s, S)$ cannot be connected because of $J(p, q)$.

$S' \subseteq S$. Let $\overline{X}$ denote the closure of a region X. For the complete chapter we fix one site $s \in S$, of which we think of a site that we want to delete from the diagram $\mathcal{V}(S)$.

To update $\mathcal{V}(S)$, after deleting site $s \in S$, we compute the diagram of the remaining sites $\mathcal{V}(S \setminus \{s\})$ within $\mathrm{VR}(s, S)$, i.e., we compute $\mathcal{V}(S \setminus \{s\}) \cap \mathrm{VR}(s, S)$. The structure of this diagram is *tree-like* (see the following definition), which is proven in Lemma 1.

Definition 1. *We call a Voronoi diagram $\mathcal{V}(S)$ in a simply-connected domain D* tree-like*, if its graph structure is a tree, or if it is a forest and D is unbounded, having exactly one distinct face for each arc of $\partial D \setminus \mathcal{V}(S)$ while its leaves are the vertices of $\partial D \cap \mathcal{V}(S)$ and points at infinity.*

Figure 3.1 illustrates $\mathcal{V}(S \setminus \{s\}) \cap \mathrm{VR}(s, S)$ (in red) for an unbounded region $\mathrm{VR}(s, S)$, and Figure 3.8(a) illustrates the same for a bounded region, where the region's boundary is shown in bold.

Bisectors in $\mathcal{J}$ that have a site p in common are called *p-related* or simply *related*; related bisectors can intersect at most twice [Klein, 1989, Lemma 3.5.2.5]. When two related bisectors $J(p, q)$ and $J(p, r)$ intersect, bisector $J(q, r)$ also intersects with them at the same point(s), and these points are the Voronoi vertices of $\mathcal{V}(\{p, q, r\})$, see Fig. 2.2. Since any two related bisectors in $\mathcal{J}$ intersect at most twice, the sequence of site occurrences along $\partial \mathrm{VR}(p, S)$, $p \in S$, forms a Davenport-Schinzel sequence of order 2 (by [Sharir and Agarwal, 1995, Theorem 5.7]).

An alternative proof for the next lemma has recently appeared in [Bohler et al., 2019, Lemma 4] and is based on the fact that the Voronoi region $\mathrm{VR}(s, S)$ is simply connected.[1]

[1] First, Bohler, Klein and Liu [2016] published the statement for *bounded* regions only (including, however, a more general statement for order-k Voronoi regions). The complete proof

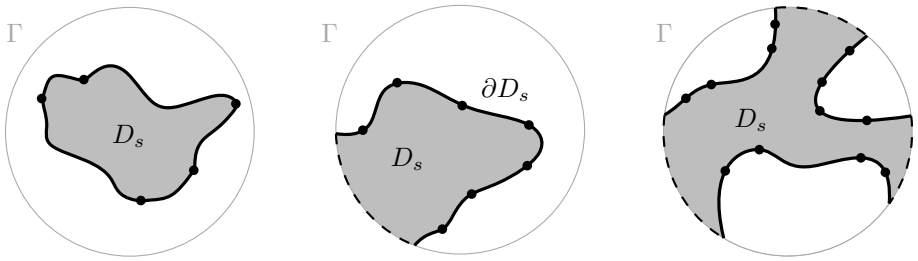

Figure 3.3. The domain $D_s = \mathrm{VR}(s,S) \cap D_\Gamma$ is shown gray and shaded. The Voronoi region $\mathrm{VR}(s,S)$ can be bounded, unbounded or have several openings to infinity, corresponding to several Γ-arcs of D_s (dashed).

Lemma 1. *$\mathcal{V}(S \setminus \{s\}) \cap VR(s,S)$ is tree-like, i.e., it is a forest having exactly one face for each Voronoi edge of $\partial VR(s,S)$. Its leaves are the Voronoi vertices of $\partial VR(s,S)$, and points at infinity if $VR(s,S)$ is unbounded. If $VR(s,S)$ is bounded then $\mathcal{V}(S \setminus \{s\}) \cap VR(s,S)$ is a tree. See Figure 3.1.*

Proof. Every face in $\mathcal{V}(S \setminus \{s\}) \cap \mathrm{VR}(s,S)$ must touch the boundary $\partial\mathrm{VR}(s,S)$ because Voronoi regions are non-empty and connected; this implies that the diagram is a forest. Every Voronoi edge $e \subseteq J(s,p)$ on $\partial\mathrm{VR}(s,S)$ must be entirely in $\mathrm{VR}(p, S \setminus \{s\})$. Thus, no leaf can lie in the interior of a Voronoi edge of $\partial\mathrm{VR}(s,S)$. On the other hand, each Voronoi vertex of $\partial\mathrm{VR}(s,S)$ must be a leaf of the diagram as its incident edges are induced by different sites.

Now we show that no two edges of $\partial\mathrm{VR}(s,S)$ can be incident to the same face of $\mathcal{V}(S \setminus \{s\}) \cap \mathrm{VR}(s,S)$. Consider two edges on $\partial\mathrm{VR}(s,S)$ induced by the same site $p \in S \setminus \{s\}$. Then there exists an edge between them, induced by a site $q \neq p$, such that the bisector $J(s,q)$ has exactly two intersections with $J(p,s)$ as shown in Figure 3.2. The bisector $J(p,q)$ intersects with them at the same two points. Since the bisector system is admissible, and thus $\mathrm{VR}(p, \{s,p,q\})$ is connected, $J(p,q)$ connects these endpoints through $D(p,s) \cap D(q,s)$ as shown in Figure 3.2, thus, $J(p,q) \cap \mathrm{VR}(s, \{s,p,q\})$ consists of two unbounded connected components. This implies that $D(p,q) \cap \mathrm{VR}(s,S)$ must have two disjoint faces, each of which is incident to exactly one of the two edges of p. Thus $\mathrm{VR}(p, S \setminus \{s\}) \cap \mathrm{VR}(s,S)$ cannot be connected and the two edges of p must be incident to different faces of $\mathcal{V}(S \setminus \{s\}) \cap \mathrm{VR}(s,S)$.

If $\mathrm{VR}(s,S)$ is unbounded, two consecutive edges of $\partial\mathrm{VR}(s,S)$ can extend to infinity, in which case there is at least one edge of $\mathcal{V}(S \setminus \{s\}) \cap \mathrm{VR}(s,S)$ extending

in [Bohler et al., 2019, Lemma 4] including *unbounded* regions has appeared later, after we published our proof in Junginger and Papadopoulou [2018b] including both bounded and unbounded regions (however, only for the nearest case, $k = 1$).

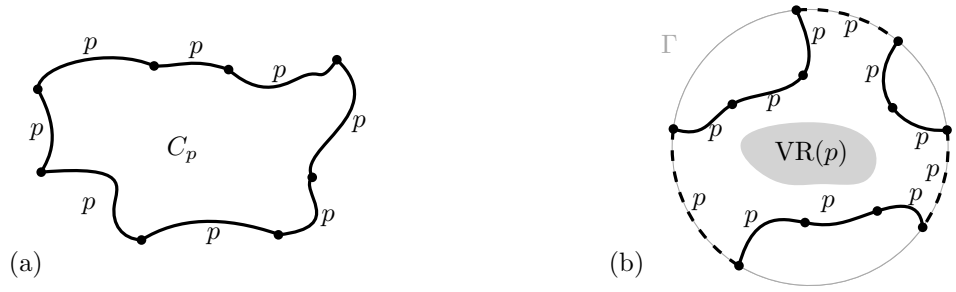

Figure 3.4. (a) A p-inverse cycle has the label p on the outside of the cycle. (b) A p-cycle that is incident to Γ.

to infinity between them; thus, leaves can be points at infinity. If $\mathrm{VR}(s,S)$ is bounded, all leaves of $\mathcal{V}(S \setminus \{s\}) \cap \mathrm{VR}(s,S)$ must lie on $\partial\mathrm{VR}(s,S)$. Since no face is incident to more than one edge of $\partial\mathrm{VR}(s,S)$, in this case $\mathcal{V}(S \setminus \{s\}) \cap \mathrm{VR}(s,S)$ cannot be disconnected, and thus is a tree. □

We denote by $\partial\mathrm{VR}(s,S)$ the boundary of a Voronoi region $\mathrm{VR}(s,S)$. Recall that the sequence of site-occurrences along $\partial\mathrm{VR}(s,S)$ can include *repetitions* and this constitutes a major difference from the respective problem for points, where no repetition can occur. $\mathcal{V}(S \setminus \{s\}) \cap \mathrm{VR}(s,S)$ contains *disconnected* Voronoi regions, i.e., for a site $p \in S \setminus \{s\}$, the region $\mathrm{VR}(p, S \setminus \{s\}) \cap \mathrm{VR}(s,S)$ can consist of several faces – a fact that introduces serious complications. For example, $\mathcal{V}(S') \cap \mathrm{VR}(s, S' \cup \{s\})$ for $S' \subset S \setminus \{s\}$ may contain various faces that are not related to $\mathcal{V}(S \setminus \{s\}) \cap \mathrm{VR}(s,S)$, and conversely, an arbitrary sub-sequence of $\partial\mathrm{VR}(s,S)$ need not correspond to any Voronoi diagram.

Let Γ be a closed Jordan curve in the plane large enough to enclose all the intersections of bisectors in $\mathcal{J}$, and such that each bisector crosses Γ exactly twice and transversally. Without loss of generality, we restrict all computations within Γ.[2] Let the interior of Γ be denoted as D_Γ. Our *domain of computation* is $D_s = \mathrm{VR}(s,S) \cap D_\Gamma$, see Figure 3.3 and we compute $\mathcal{V}(S \setminus \{s\}) \cap D_s$. The curve Γ can be interpreted as $J(p, s_\infty)$, for all $p \in S$, where s_∞ is an additional site at infinity.

In the following we make some observations regarding an admissible bisector system, which are used as tools in proofs throughout this dissertation. Let C_p be a cycle of p-related bisectors in the arrangement of bisectors $\mathcal{J} \cup \Gamma$. If for every edge in C_p the label p appears on the outside of the cycle then C_p is called *p-inverse*, see Figure 3.4(a). If the label p appears only inside C_p then C_p is called

[2]The presence of Γ is conceptual and its exact position unknown; we never compute coordinates on Γ.

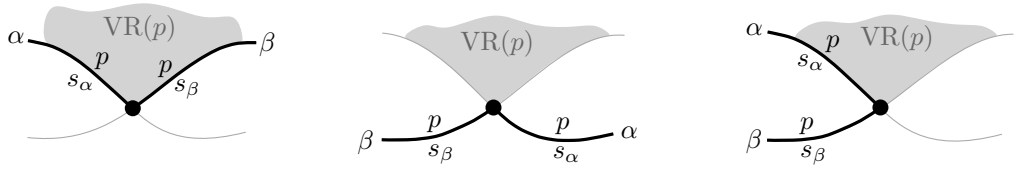

Figure 3.5. In (a) the arcs α, β fulfill the p-monotone path condition. In (b) and (c) they do not fulfill it.

a *p-cycle*, see Figure 3.4(b). Recall that Γ can be considered a p-related bisector, for all sites $p \in S$, where the label p is in the interior of Γ. Thus, a p-cycle may contain pieces of Γ, whereas a p-inverse cycle cannot contain any such piece.

Lemma 2. *In an admissible bisector system there is no p-inverse cycle.*

Proof. Suppose a p-inverse cycle exists in the admissible bisector system. Let C_p denote a minimal such cycle, where no p-related bisector may intersect the interior of C_p and let D_p denote the interior of C_p. Such a minimal cycle must exist because if a bisector $J(p,q)$ intersects D_p, then it defines another (smaller) p-inverse cycle that is contained in $C_p \cup D_p$ and whose interior is not intersected by $J(p,q)$. Let $S' \subseteq S$ denote the set of sites that define the edges of C_p. Considering S', the farthest Voronoi region of p is $\mathrm{FVR}(p,S') = \bigcap_{q \in S' \setminus \{p\}} D(q,p)$. But by its definition, D_p must be identical to one face of $\mathrm{FVR}(p,S')$. Since farthest Voronoi regions must be unbounded (see e.g., Mehlhorn et al. [2001]), we derive a contradiction. □

The following *transitivity lemma* is a consequence of transitivity of dominance regions [Bohler et al., 2015, Lemma 2] and the fact that bisectors $J(p,q), J(q,r)$, $J(p,r)$ intersect at the same point(s).

Lemma 3. *Let $z \in \mathbb{R}^2$ and $p,q,r \in S$. If $z \in D(p,q)$ and $z \in \overline{D(q,r)}$, then $z \in D(p,r)$.*

We make a general position assumption that no three p-related bisectors intersect at the same point. This implies that Voronoi vertices have degree 3.

3.2 Voronoi-like diagrams

In this section we first introduce the concepts of a *monotone path*, an *envelope* and a *boundary curve*. Then, we present the *Voronoi-like diagram* based on these concepts and give basic properties.

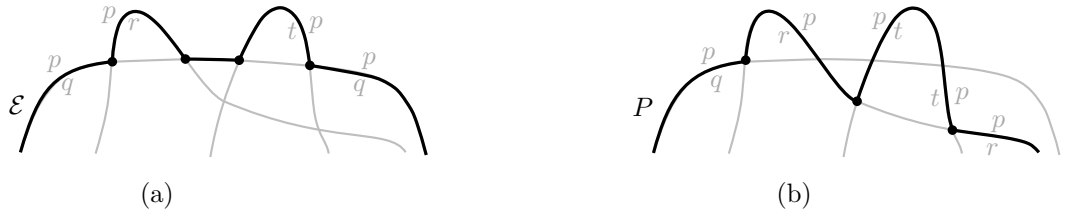

Figure 3.6. (a) The envelope $\mathcal{E} = \text{env}(\mathcal{J}_{p,\{q,r,t\}})$. (b) A p-monotone path P in $\mathcal{J}_{p,\{q,r,t\}}$.

Let $\mathcal{S}$ denote the sequence of Voronoi edges bounding the Voronoi region $\text{VR}(s,S)$ within D_Γ, along $\partial\text{VR}(s,S)$, i.e., $\mathcal{S} = \partial\text{VR}(s,S) \cap D_\Gamma$. We consider $\mathcal{S}$ as a cyclically ordered set of *arcs*, where each arc is a Voronoi edge of $\partial\text{VR}(s,S)$. Each arc $\alpha \in \mathcal{S}$ is *induced* by a site $s_\alpha \in S \setminus \{s\}$, where $\alpha \subseteq J(s,s_\alpha)$.

A single site in S may induce several arcs on $\mathcal{S}$. Recall, that the sequence of site occurrences along $\partial\text{VR}(s,S)$ is a Davenport-Schinzel sequence of order 2 and thus, the number of Voronoi edges in $\mathcal{S}$ is $2n_s-1$, where n_s is the number of sites neighboring s (i.e., inducing Voronoi edges on $\partial\text{VR}(s,S)$).

We can interpret the arcs in $\mathcal{S}$ as sites that induce a Voronoi diagram $\mathcal{V}(\mathcal{S})$ within the domain $D_s = \text{VR}(s,S) \cap D_\Gamma$ such as $\mathcal{V}(\mathcal{S}) = \mathcal{V}(S \setminus \{s\}) \cap D_s$, see, e.g., Figure 3.8(a) illustrating $\mathcal{S}$ in bold black and the diagram $\mathcal{V}(\mathcal{S})$ in red. By Lemma 1, each face of $\mathcal{V}(S \setminus \{s\}) \cap D_s$ is incident to exactly one arc in $\mathcal{S}$. In this respect, we can define the Voronoi region of an arc α in $\mathcal{S}$, denoted as $\text{VR}(\alpha,\mathcal{S})$, as the face of $\mathcal{V}(S \setminus \{s\}) \cap D_s$ incident to α. Then $\mathcal{V}(\mathcal{S}) = D_s \setminus \bigcup_{\alpha\in\mathcal{S}} \text{VR}(\alpha,\mathcal{S})$.

For a site $p \in S$ and a subset of sites $S' \subseteq S$, let $\mathcal{J}_{p,S'} = \{J(p,q) \mid q \in S', q \neq p\}$ denote the set of all p-related bisectors involving sites in S'.

Definition 2. *Consider the arrangement of bisectors $\mathcal{J}_{p,S'}$, $S' \subseteq S$, which is denoted as $\mathcal{A}(\mathcal{J}_{p,S'})$ (see Section 2.2). A* path *P in $\mathcal{J}_{p,S'}$ is a connected sequence of alternating edges and vertices of $\mathcal{A}(\mathcal{J}_{p,S'})$. An* arc *$\alpha$ of P is a maximally connected set of consecutive edges and vertices of the arrangement along P, which belong to the same bisector. The common endpoint of two consecutive arcs of P is a* vertex of P. *An arc of P is also called an* edge.

Two consecutive arcs in a path P are pieces of different bisectors. We use the notation $\alpha \in P$ to refer to an arc α of P.

Definition 3. *A path P in $\mathcal{J}_{p,S'}$, $S' \subseteq S$, is called* p-monotone *if any two consecutive arcs $\alpha,\beta \in P$, where $\alpha \subseteq J(p,s_\alpha)$ and $\beta \subseteq J(p,s_\beta)$, coincide locally with the Voronoi edges of $\partial VR(p,\{p,s_\alpha,s_\beta\})$ that are incident to the common endpoint of α,β, within a neighborhood around this endpoint (see Figure 3.5).*

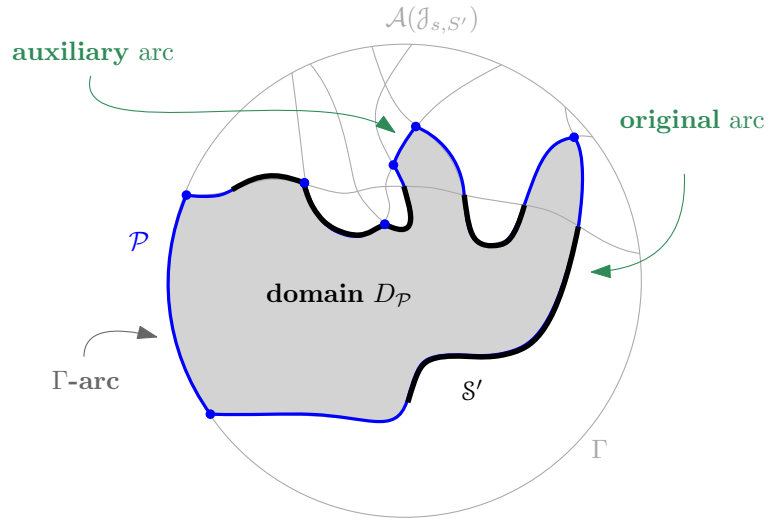

Figure 3.7. A boundary curve $\mathcal{P}$ for $\mathcal{S}'$ that consists of five original arcs, each of which contains an arc of $\mathcal{S}'$ (shown in black, bold), one auxiliary arc, which does not contain an arc of $\mathcal{S}'$, and one Γ-arc (subset of Γ).

A p-monotone path of interest is the *envelope* of $\mathcal{J}_{p,S'}$.

Definition 4. *The* envelope of $\mathcal{J}_{p,S'}$*, with respect to site* p*, is* $env(\mathcal{J}_{p,S'}) = \partial VR(p, S' \cup \{p\})$*, which is also called a* p*-envelope, see Figure 3.6(a).*

Figure 3.6 illustrates two p-monotone paths, where the path in Figure 3.6(a) is a p-envelope.

The system of bisectors $\mathcal{J}_{p,S'}$ may have an arrangement $\mathcal{A}(\mathcal{J}_{p,S'})$ that consists of several connected components. For convenience, and in order to unify these connected components, we include the curve Γ in the corresponding system of bisectors. Then, $\text{env}(\mathcal{J}_{p,S'} \cup \Gamma)$ is a closed p-monotone path that consists of the connected components in $\text{env}(\mathcal{J}_{p,S'})$, as interleaved by arcs of Γ.[3]

Definition 5. *Consider a set of arcs* $\mathcal{S}' \subseteq \mathcal{S}$ *and let* $S' = \{s_\alpha \in S \mid \alpha \in \mathcal{S}'\} \subseteq S \setminus \{s\}$ *be its corresponding set of sites. A closed* s*-monotone path* $\mathcal{P}$ *in* $\mathcal{J}_{s,S'} \cup \Gamma$ *such that* $\mathcal{P}$ *contains all arcs in* $\mathcal{S}'$ *is called a* boundary curve for $\mathcal{S}'$*, see Figure 3.7.*

The part of the plane enclosed by a boundary curve $\mathcal{P}$ *is called its* domain*, and it is denoted by* $D_{\mathcal{P}}$*. Given* $\mathcal{P}$*, let* $S_{\mathcal{P}} = S'$*.*

A set of arcs $\mathcal{S}' \subset \mathcal{S}$ can admit several different boundary curves. One such boundary curve is its s-envelope $\text{env}(\mathcal{S}') = \text{env}(\mathcal{J}_{s,S'} \cup \Gamma)$. The set $\mathcal{S}$ has only one

[3] Recall that we interpret the curve Γ as $J(p, s_\infty)$, for all $p \in S$, where s_∞ is an additional site at infinity and $D(p, s_\infty) = D_\Gamma$, (the interior of Γ).

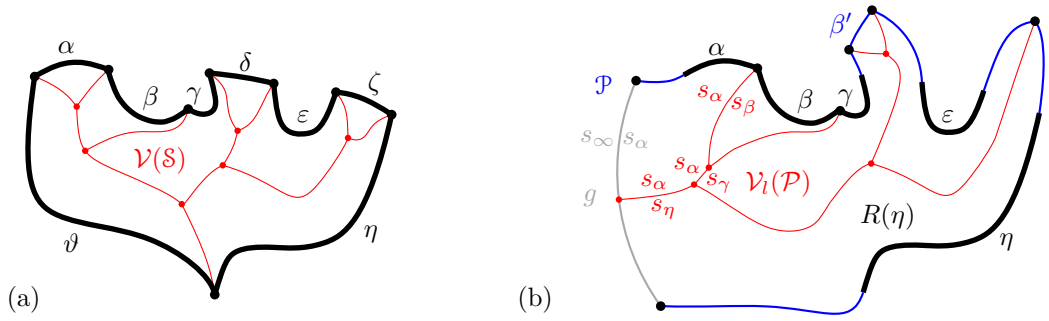

Figure 3.8. (a) illustrates $\mathcal{S} = \partial\mathrm{VR}(s,S)$ in bold (black) and $\mathcal{V}(\mathcal{S})$ in red; $\mathcal{S} = (\alpha,\beta,\gamma,\delta,\varepsilon,\zeta,\eta,\vartheta)$. (b) illustrates a boundary curve $\mathcal{P}$ for $\mathcal{S}' \subseteq \mathcal{S}$ in blue and gray, and $\mathcal{V}_l(\mathcal{P})$ in red. The gray arc g is a Γ-arc, and the blue arc β' is an auxiliary arc; the remaining arcs are original and their core parts $\mathcal{S}' = (\alpha,\beta,\gamma,\varepsilon,\eta)$ are illustrated in bold. Some bisectors illustrate their labels.

boundary curve, which is the s-envelope $\mathrm{env}(\mathcal{S}) = \partial(\mathrm{VR}(s,S) \cap D_\Gamma)$. Figure 3.8(b) illustrates a boundary curve for $\mathcal{S}' \subseteq \mathcal{S}$ in Figure 3.8(a).

A boundary curve $\mathcal{P}$ in $\mathcal{J}_{s,\mathcal{S}'} \cup \Gamma$ consists of bisector pieces in $\mathcal{J}_{s,\mathcal{S}'}$, called *boundary arcs*, and pieces of Γ, called Γ-*arcs*. Γ-arcs correspond to openings of the domain $D_\mathcal{P}$ to infinity. Among the boundary arcs, those that contain an arc of $\mathcal{S}'$ are called *original* and any others are called *auxiliary arcs*, see Figure 3.7. Original boundary arcs are expanded versions of the arcs in $\mathcal{S}$. To distinguish between them, we call the elements of $\mathcal{S}$ *core arcs* and use an * in their notation. In Figures 3.7 and 3.8, the core arcs are illustrated in bold. $|\mathcal{P}|$ denotes the number of boundary arcs in $\mathcal{P}$.

We now define the Voronoi-like diagram of a boundary curve $\mathcal{P}$.

Definition 6. *Given a boundary curve $\mathcal{P}$ on a set of arcs $\mathcal{S}' \subseteq \mathcal{S}$, whose corresponding set of sites is $S' = \{s_\alpha \in S \mid \alpha \in \mathcal{S}'\} \subseteq S \setminus \{s\}$, the* Voronoi-like diagram *of $\mathcal{P}$ is a plane graph $\mathcal{V}_l(\mathcal{P})$ on the arrangement of $\mathcal{J}(S') = \{J(p,q) \in \mathcal{J} \mid p,q \in S'\}$ that induces a subdivision on the domain $D_\mathcal{P}$ as follows (see Figure 3.8(b)):*

1. *For each boundary arc $\alpha \in \mathcal{P}$, there is exactly one distinct face $R(\alpha)$ in $\mathcal{V}_l(\mathcal{P})$ such that $\partial R(\alpha)$ consists of the arc α plus an s_α-monotone path in $\mathcal{J}_{s_\alpha,\mathcal{S}'} \cup \Gamma$.*

2. *The faces of the boundary arcs cover the entire domain of $\mathcal{P}$: $\bigcup_{\alpha\in\mathcal{P}\setminus\Gamma} \overline{R(\alpha)} = \overline{D_\mathcal{P}}$.*

The Voronoi-like diagram of $\mathcal{P}$ is $\mathcal{V}_l(\mathcal{P}) = D_\mathcal{P} \setminus \bigcup_{\alpha\in\mathcal{P}} R(\alpha)$. The Voronoi-like region *$R(\alpha)$ in $\mathcal{V}_l(\mathcal{P})$ is also denoted as $R(\alpha,\mathcal{P})$.*

If the boundary curve $\mathcal{P}$ coincides with the s-envelope $\mathcal{E}$ of $\mathcal{S}'$, then the Voronoi-like diagram $\mathcal{V}_l(\mathcal{E})$ coincides with the real Voronoi diagram $\mathcal{V}(\mathcal{E}) = \mathcal{V}(S') \cap D_{\mathcal{E}}$ (see Corollary 1 in the sequel). In the Voronoi diagram $\mathcal{V}(\mathcal{E})$, the face incident to a boundary arc $\alpha \in \mathcal{E}$ can be regarded as the Voronoi region of the arc α which is denoted as $\mathrm{VR}(\alpha, \mathcal{E})$. In the sequel we show that $\mathcal{V}_l(\mathcal{E}) = \mathcal{V}(\mathcal{E})$, i.e., $\mathrm{VR}(\alpha, \mathcal{E}) = R(\alpha, \mathcal{E})$ for any arc $\alpha \in \mathcal{E}$. Furthermore, if $\mathcal{P}$ is an arbitrary boundary curve on $\mathcal{S}'$, the Voronoi-like regions in $\mathcal{V}_l(\mathcal{P})$ are related to the real Voronoi regions of $\mathcal{V}(\mathcal{E})$ as supersets, for the original and any other common arcs between $\mathcal{P}$ and $\mathcal{E}$. In Figure 3.8(b) the Voronoi-like region $R(\eta)$ in $\mathcal{V}_l(\mathcal{P})$ is a superset of its corresponding Voronoi region $\mathrm{VR}(\eta, \mathcal{S})$, which is illustrated in Figure 3.8(a); similarly for e.g., $R(\alpha, \mathcal{P})$ in Figure 3.8(b), $R(\alpha, \mathcal{P}) \supseteq \mathrm{VR}(\alpha, \mathcal{S})$.

Lemma 4. *Let $\alpha \in \mathcal{P}$ be a boundary arc such that $\tilde{\alpha} \subseteq \alpha$ appears on the s-envelope $\mathcal{E}$. Then, $R(\alpha, \mathcal{P}) \supseteq VR(\tilde{\alpha}, \mathcal{E})$. Further, if α is original, then $R(\alpha, \mathcal{P}) \supseteq VR(\tilde{\alpha}, \mathcal{E}) \supseteq VR(\alpha^*, \mathcal{S})$.*

Proof. By the definition of a Voronoi region, no piece of a bisector $J(s_\alpha, \cdot)$ can appear in the interior of a Voronoi region in $\mathcal{V}(S') \cap D_{\mathcal{E}}$. Thus no piece of $J(s_\alpha, \cdot)$ can appear in $\mathrm{VR}(\tilde{\alpha}, \mathcal{E})$, for any $\tilde{\alpha} \in \mathcal{E}$. Since $\alpha \supseteq \tilde{\alpha}$, by the definition of a Voronoi-like region it follows that $R(\alpha, \mathcal{P}) \supseteq \mathrm{VR}(\tilde{\alpha}, \mathcal{E})$. For an original arc α, since $S' \subseteq S$, by the standard monotonicity property of Voronoi regions, we also have $\mathrm{VR}(\tilde{\alpha}, \mathcal{E}) \supseteq \mathrm{VR}(\alpha^*, \mathcal{S})$. □

As a corollary to the superset property of Lemma 4, the adjacencies of the real Voronoi diagram $\mathcal{V}(\mathcal{E})$ are preserved in $\mathcal{V}_l(\mathcal{P})$, for all the original arcs in $\mathcal{P}$. As a result, $\mathcal{V}_l(\mathcal{E})$ must coincide with the Voronoi diagram $\mathcal{V}(\mathcal{E}) = \mathcal{V}(S') \cap D_{\mathcal{E}}$.

Corollary 1. *$\mathcal{V}_l(\mathcal{E}) = \mathcal{V}(\mathcal{E}) = \mathcal{V}(S') \cap D_{\mathcal{E}}$, for the s-envelope $\mathcal{E}$ of $\mathcal{S}' \subseteq \mathcal{S}$.*

In the following we derive some basic properties of Voronoi-like regions and their interaction with the bisectors in $\mathcal{J}$ that we use throughout this dissertation. The following property establishes that an s-bisector $J(s, s_\alpha)$ can never intersect $R(\alpha, \mathcal{P})$ in $\mathcal{V}_l(\mathcal{P})$.

Lemma 5. *For any arc $\alpha \in \mathcal{P}$, $R(\alpha, \mathcal{P}) \subseteq D(s, s_\alpha)$.*

Proof. The contrary would yield a forbidden s_α-inverse cycle defined by a component of $J(s, s_\alpha) \cap R(\alpha, \mathcal{P})$ and the incident portion of $\partial R(\alpha, \mathcal{P})$. □

Lemma 6. *For a boundary curve $\mathcal{P}$, the domain $\overline{D_{\mathcal{P}}}$ may not contain a p-cycle of $\mathcal{J}(S_{\mathcal{P}}) \cup \Gamma$, for any site $p \in S_{\mathcal{P}}$.*

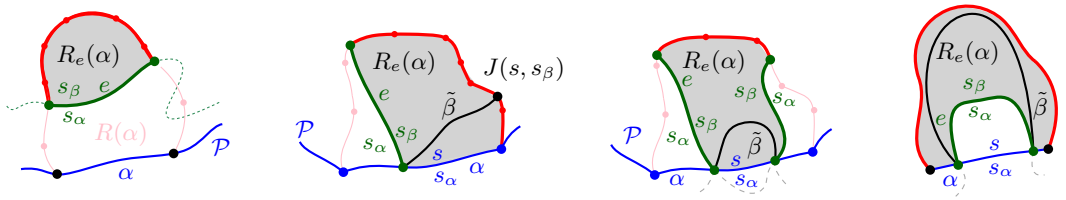

Figure 3.9. Illustrations for Lemma 7 – the cut property. The shaded gray region $R_e(\alpha)$ lies in $D(s_\beta, s_\alpha)$.

Proof. Let $p \in S_\mathcal{P}$ define an original arc along $\mathcal{P}$. This arc is bounding $\mathrm{VR}(p, S_\mathcal{P} \cup \{s\})$, thus, it must have a portion within the interior of $\mathrm{VR}(p, S_\mathcal{P})$ in $\mathcal{V}(S_\mathcal{P})$. Hence, $\mathrm{VR}(p, S_\mathcal{P})$ has a non-empty intersection with $\mathbb{R}^2 \setminus \overline{D_\mathcal{P}}$. But $\mathrm{VR}(p, S_\mathcal{P}) \cap D_\Gamma$ must be enclosed within any p-cycle of $\mathcal{J}(S_\mathcal{P}) \cup \Gamma$, by its definition. Thus, no such p-cycle can be contained in $\overline{D_\mathcal{P}}$. Refer to Figure 3.10. □

Next, we give a key property of a Voronoi-like region, which we call the *cut property*.

Suppose bisector $J(s_\alpha, s_\beta)$ appears within a Voronoi-like region $R(\alpha, \mathcal{P})$, see Fig. 3.9. Let e be a connected component of $J(s_\alpha, s_\beta) \cap R(\alpha, \mathcal{P})$ and let $R_e(\alpha)$ denote the portion of region $R(\alpha, \mathcal{P})$ that is *cut out by* e, as shown in Fig. 3.9. In particular, if e does not intersect α, then $R_e(\alpha)$ is the portion of the region at the opposite side of e as α (case (a), see Fig. 3.9(a)). Otherwise, let $\tilde{\beta}$ be the component of $J(s, s_\beta) \cap R(\alpha, \mathcal{P})$ intersecting α at the same point as e. If e and $\tilde{\beta}$ have only one endpoint on α, let $R_e(\alpha)$ be the portion of the region that contains $\tilde{\beta}$ (case (b), see Fig. 3.9(b)). If e intersects α once but $\tilde{\beta}$ intersects it twice, then there are two components of $J(s_\alpha, s_\beta) \cap R(\alpha, \mathcal{P})$ incident to α; let $R_e(\alpha)$ denote the portion of $R(\alpha, \mathcal{P})$ between these two components (case (c), see Fig. 3.9(c)). Otherwise, both e and $\tilde{\beta}$ intersect α twice, let $R_e(\alpha)$ be the portion of the region incident to $\partial R(\alpha, \mathcal{P})$ (case (d), see Fig. 3.9(d)).

Lemma 7. *Suppose bisector $J(s_\alpha, s_\beta)$ appears within $R(\alpha, \mathcal{P})$ (see Fig. 3.9). For any connected component e of $J(s_\alpha, s_\beta) \cap R(\alpha, \mathcal{P})$, it holds $R_e(\alpha) \subseteq D(s_\beta, s_\alpha)$. Thus, if e does not intersect α, the label s_α must appear on the same side of e as α.*

Note that $\partial R_e(\alpha)$ may contain Γ-arcs.

Proof. Let e be an arbitrary component of $J(s_\alpha, s_\beta) \cap R(\alpha, \mathcal{P})$. Suppose for the sake of contradiction that $R_e(\alpha) \not\subseteq D(s_\beta, s_\alpha)$. Then $J(s_\beta, s_\alpha)$ must intersect the interior of $R_e(\alpha)$ with a component e' of $J(s_\beta, s_\alpha) \cap R(\alpha, \mathcal{P})$, which is different from e. Among any such component let e' be the first one following e along $J(s_\beta, s_\alpha)$.

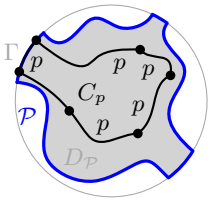

Figure 3.10. A p-cycle (possibly with Γ-arcs) within $\overline{D_\mathcal{P}}$ does not exist.

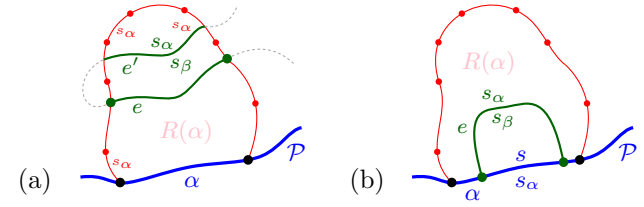

Figure 3.11. (a) Illustration for the proof of Lemma 7: The component e' has a contradictory edge labeling.

Since e' cannot intersect e, nor can it intersect $\tilde{\beta}$ (in cases (b) and (c) as a direct implication of Lemma 5), it follows that e' cannot intersect the arc α. But then e' creates an s_α-cycle with $\partial R_e(\alpha)$ within $\overline{D_\mathcal{P}}$ (see Figure 3.11(a)), contradicting Lemma 6 □

By Lemma 7, any component of $J(s_\alpha, s_\beta) \cap R(\alpha, \mathcal{P})$ must appear sequentially along $\partial R(\alpha, \mathcal{P})$. In addition, if any such component exists, then $J(s, s_\beta)$ must also intersect $D_\mathcal{P}$, indicating an arc of s_β that is *missing* from $\mathcal{P}$. In particular, the presence of any component of $J(s_\alpha, s_\beta) \cap R(\alpha, \mathcal{P})$ implies that there is a component of $J(s, s_\beta) \cap D_\mathcal{P}$ that is *missing from* $\mathcal{P}$. We show this formally in Section 4.1 (Lemma 17).

In the following theorem we conclude that $\mathcal{V}_l(\mathcal{P})$ exists and is unique, thus, it is well-defined. The proof is deferred to Chapter 4.

Theorem 1. *Given a boundary curve $\mathcal{P}$ of $\mathcal{S}' \subseteq \mathcal{S}$, $\mathcal{V}_l(\mathcal{P})$ exists and is unique.*

The complexity of $\mathcal{V}_l(\mathcal{P})$ is $O(|\mathcal{P}|)$, as it is a planar graph with exactly one face per boundary arc and vertices of degree 3 (or 1).

3.3 Insertion in a Voronoi-like diagram

Consider a boundary curve $\mathcal{P}$ for $\mathcal{S}' \subset \mathcal{S}$ and its Voronoi-like diagram $\mathcal{V}_l(\mathcal{P})$. Unless noted otherwise, we assume a counterclockwise traversal of $\mathcal{P}$.

Let β^* be an arc in $\mathcal{S} \setminus \mathcal{S}'$. Since β^* is a core arc, it must be contained in the closure of the domain $D_\mathcal{P}$. We define an insertion operation $\oplus$, which inserts arc β^*, in $\mathcal{P}$ deriving $\mathcal{P}_\beta = \mathcal{P} \oplus \beta^*$. Let $\beta \supseteq \beta^*$ be the connected component of $J(s, s_\beta) \cap \overline{D_\mathcal{P}}$ that contains β^* (see Figure 3.12).

Given $\mathcal{P}$ and β^*, we define the original arc $\beta \supseteq \beta^*$ as the connected component of $J(s, s_\beta) \cap \overline{D_\mathcal{P}}$ that contains β^* (see Figure 3.12). Let $\mathcal{P}_\beta$ denote the

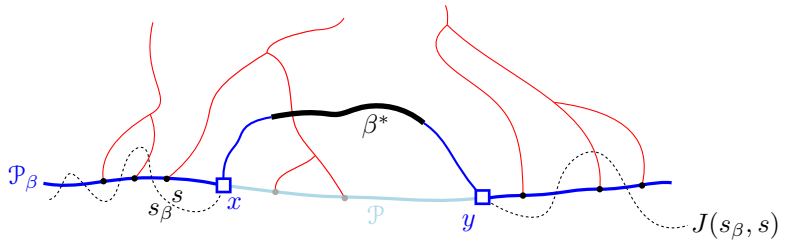

Figure 3.12. $\mathcal{P}_\beta = \mathcal{P} \oplus \beta$, core arc β^* is bold, black. Endpoints of β are x, y.

boundary curve derived by $\mathcal{P}$ by deleting its portion between the endpoints of β and substituting it with arc β (see Figure 3.12). We say that $\mathcal{P}_\beta$ is obtained from $\mathcal{P}$ by *inserting* the core arc β^*, or equivalently, by inserting the original arc β. Thus, we define an insertion operation $\oplus$, which inserts the core arc β^* in $\mathcal{P}$, deriving $\mathcal{P}_\beta = \mathcal{P} \oplus \beta^* = \mathcal{P} \oplus \beta$.

The insertion operation $\oplus$ further computes the boundary $\partial R(\beta, \mathcal{P}_\beta)$, inserts $R(\beta, \mathcal{P}_\beta)$ in $\mathcal{V}_l(\mathcal{P})$, and derives the Voronoi-like diagram $\mathcal{V}_l(\mathcal{P}_\beta) = \mathcal{V}_l(\mathcal{P}) \oplus \beta$. When the distinction is not important, we often slightly abuse notation and use the notation β to refer interchangeably to either the original arc β or its core portion β^* in $\mathcal{S} \setminus \mathcal{S}'$.

Figure 3.13 enumerates all possible cases of inserting arc β in $\mathcal{P}$ and is summarized in the following observation.

Observation 1. *All possible cases of inserting arc $\beta^* \subseteq \beta$ in $\mathcal{P}$ (see Figure 3.13). $D_{\mathcal{P}_\beta} \subseteq D_{\mathcal{P}}$.*

(a) Arc β straddles the endpoint of two consecutive boundary arcs; no arcs in $\mathcal{P}$ are deleted.

(b) Auxiliary arcs in $\mathcal{P}$ are deleted by β; their regions are also deleted from $\mathcal{V}_l(\mathcal{P}_\beta)$.

(c) An arc $\alpha \in \mathcal{P}$ is split into two arcs by β; $R(\alpha)$ in $\mathcal{V}_l(\mathcal{P})$ will also be split.

(d) A Γ-arc is split in two by β; $\mathcal{V}_l(\mathcal{P}_\beta)$ may switch from being a tree to being a forest.

(e) A Γ-arc is deleted or shrunk by inserting β. $\mathcal{V}_l(\mathcal{P}_\beta)$ may become a tree.

(f) $\mathcal{P}$ already contains a boundary arc $\bar{\beta} \supseteq \beta^*$; then $\beta = \bar{\beta}$ and $\mathcal{P}_\beta = \mathcal{P}$.

Note that $\mathcal{P}_\beta$ may contain fewer, the same number, or even one extra auxiliary arc compared to $\mathcal{P}$.

Lemma 8. *The curve $\mathcal{P}_\beta = \mathcal{P} \oplus \beta$ is a boundary curve for $\mathcal{S}' \cup \{\beta^*\}$.*

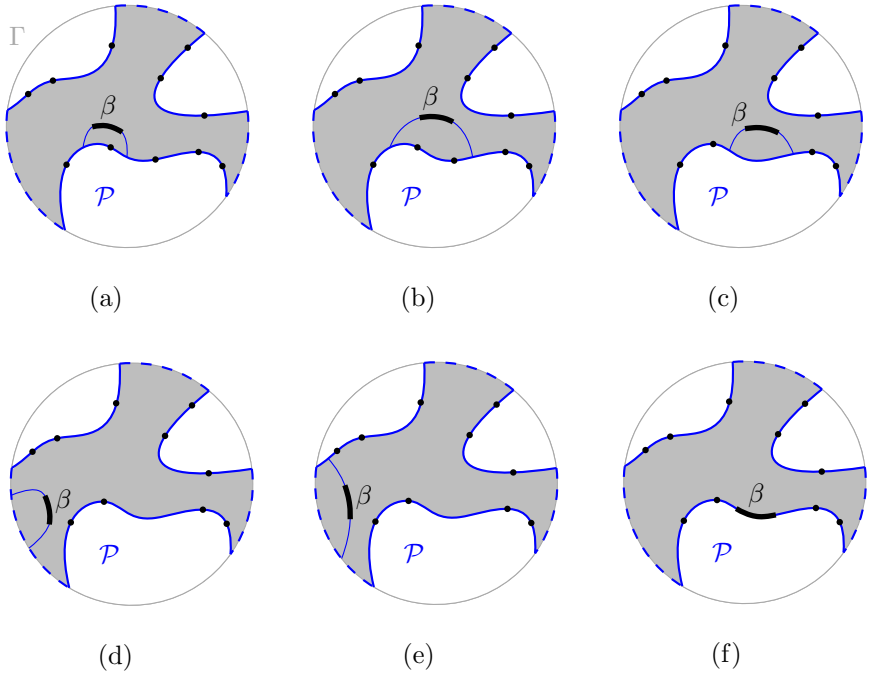

Figure 3.13. Insertion cases for an arc β.

Proof. Since $\mathcal{P}$ is a (closed) s-monotone path in $\mathcal{J}_{s,S'} \cup \Gamma$, $\mathcal{P}_\beta$ is also an s-monotone path in $\mathcal{J}_{s,S' \cup \{s_\beta\}} \cup \Gamma$, by construction. Since every core arc in $\mathcal{S}$ appears on the envelope $\text{env}(\mathcal{J}_{s,S} \cup \Gamma)$, no core arc can appear in the portion of $\mathcal{P}$ that gets deleted by the insertion of β. Thus, no original arc in $\mathcal{P}$ can get deleted, and $\mathcal{P}_\beta$ contains all core arcs in $\mathcal{S}' \cup \beta^*$. □

Given $\mathcal{V}_l(\mathcal{P})$ and $\beta^* \in \mathcal{S} \setminus \mathcal{S}'$, we define a *merge curve* $J(\beta)$, within $\mathcal{V}_l(\mathcal{P})$, which delimits the boundary of the region $R(\beta, \mathcal{P}_\beta)$ for the original arc $\beta \supseteq \beta^*$. We define $J(\beta)$ incrementally, starting at an endpoint of β. Let x and y denote the endpoints of β, where x, β, y are in counterclockwise order around $\mathcal{P}_\beta$; refer to Figure 3.14.

Definition 7. *Given $\mathcal{V}_l(\mathcal{P})$ and arc $\beta \subseteq J(s, s_\beta)$, the* merge curve *$J(\beta)$ is a path $(v_1, \ldots, v_m)$ in the arrangement of s_β-related bisectors, $\mathcal{J}_{s_\beta, S_\mathcal{P}} \cup \Gamma$, connecting the endpoints of β, $v_1 = x$ and $v_m = y$. Each edge $e_i = (v_i, v_{i+1})$ is an arc of a bisector $J(s_\beta, \cdot)$, called a* bisector edge, *or an arc on Γ. For $i = 1$: if $x \in J(s_\beta, s_\alpha)$, then $e_1 \subseteq J(s_\beta, s_\alpha)$; if $x \in \Gamma$, then $e_1 \subseteq \Gamma$. Given v_i, vertex v_{i+1} and edge e_{i+1} are defined as follows (see Figure 3.14). We assume a clockwise ordering of $J(\beta)$.*

1. *If $e_i \subseteq J(s_\beta, s_\alpha)$, let v_{i+1} be the other endpoint of the component $J(s_\beta, s_\alpha) \cap R(\alpha)$ incident to v_i. If $v_{i+1} \in J(s_\beta, \cdot) \cap J(s_\beta, s_\alpha)$, then $e_{i+1} \subseteq J(s_\beta, \cdot)$. If $v_{i+1} \in \Gamma$,*

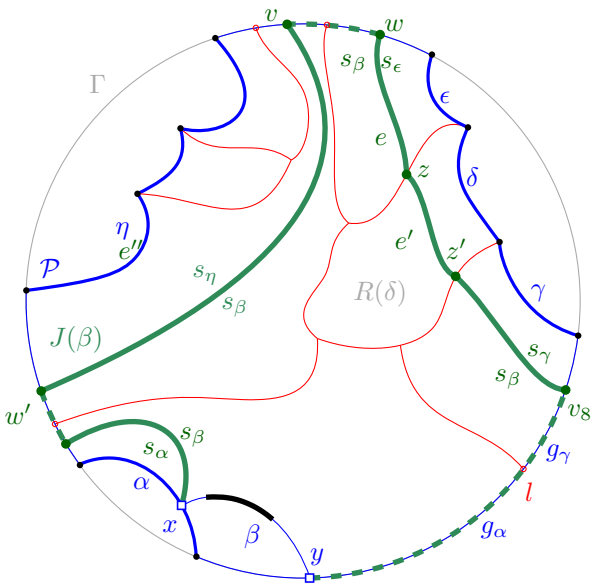

Figure 3.14. The merge curve $J(\beta)$ (thick, green) on $\mathcal{V}_l(\mathcal{P})$ (thin, red).

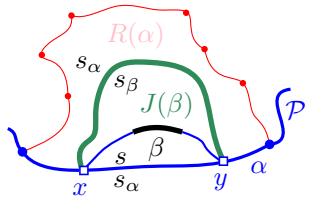

Figure 3.15. If β splits α, $J(\beta) \subset R(\alpha)$ would yield a forbidden s_α-inverse cycle.

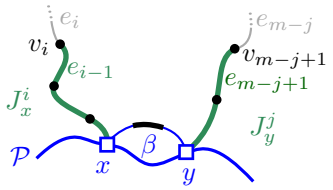

Figure 3.16. J_x^i and J_y^j in Section 3.3.1.

then $e_{i+1} \subseteq \Gamma$. (In Figure 3.14, see $e_i = e', v_i = z, v_{i+1} = z'$.)

2. *If $e_i \subseteq \Gamma$, let g be the Γ-arc incident to v_i. Let $e_{i+1} \subseteq J(s_\beta, s_\gamma)$, where $R(\gamma)$ is the first region, incident to g clockwise from v_i, such that $J(s_\beta, s_\gamma)$ intersects $g \cap \overline{R(\gamma)}$; let v_{i+1} be this intersection point. (In Figure 3.14, see $v_i = v$ and $v_{i+1} = w$.)*

A vertex v along $J(\beta)$, is called *valid* if v is a vertex in the arrangement of the bisector system $\mathcal{J}_{s_\beta, s_\mathcal{P}} \cup \Gamma$ or v is an endpoint of β. The following theorem shows that $J(\beta)$ is well-defined and that it forms an s_β-monotone path. We defer its proof to the end of this section.

Theorem 2. *The merge curve $J(\beta)$ is a unique s_β-monotone path in the arrangement of s_β-related bisectors $\mathcal{J}_{s_\beta, s_\mathcal{P}} \cup \Gamma$ connecting the endpoints of β. If arc β splits a single arc $\alpha \in \mathcal{P}$ (Observation 1(c)) then $J(\beta)$ must intersect $R(\alpha, \mathcal{P})$ in two different components, $e_1, e_{m-1} \subseteq J(s_\alpha, s_\beta)$. $J(\beta)$ can intersect any other region in $\mathcal{V}_l(\mathcal{P})$ at most once. $J(\beta)$ cannot intersect $R(\alpha', \mathcal{P})$ for any auxiliary arc $\alpha' \in \mathcal{P} \setminus \mathcal{P}_\beta$, nor can it intersect arc β in its interior.*

We define the region $R(\beta)$ as the area enclosed between $J(\beta)$ and β. Let $\mathcal{V}_l(\mathcal{P}) \oplus \beta$ be the subdivision obtained from $\mathcal{V}_l(\mathcal{P})$ by inserting $J(\beta)$ and deleting any portion of $\mathcal{V}_l(\mathcal{P})$ that is enclosed between $J(\beta)$ and the portion of $\mathcal{P}$ deleted by β. Formally, let $\mathcal{V}_l(\mathcal{P}) \oplus \beta = \big((\mathcal{V}_l(\mathcal{P}) \setminus R(\beta)) \cup J(\beta)\big) \cap D_{\mathcal{P}_\beta}$.

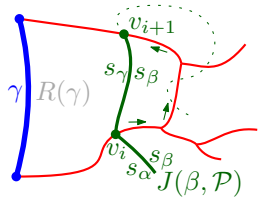

Figure 3.17. Impossible configuration of $J(s_\beta, s_\gamma)$. Scanning $\partial R(\gamma)$ from v_i counterclockwise, Lemma 7 assures that v_{i+1} is the first encountered intersection of $J(s_\beta, s_\gamma)$ with $\partial R(\gamma)$.

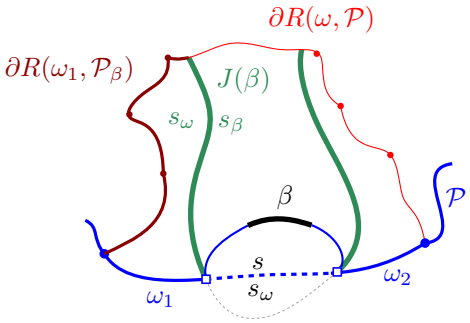

Figure 3.18. Illustration for the case that β splits ω in two arcs ω_1 and ω_2.

Theorem 3. *$\mathcal{V}_l(\mathcal{P}) \oplus \beta$ is the Voronoi-like diagram $\mathcal{V}_l(\mathcal{P}_\beta)$.*

Proof. By construction, $\mathcal{V}_l(\mathcal{P}) \oplus \beta$ defines a subdivision of the domain $D_{\mathcal{P}_\beta}$ and contains one face $R(\alpha)$ for each arc $\alpha \in \mathcal{P}_\beta$.

By Theorem 2, $J(\beta)$, and thus $\partial R(\beta) \setminus \beta$, is an s_β-monotone path. For any arc $\alpha \in \mathcal{P}$ such that $J(\beta)$ passes through $R(\alpha, \mathcal{P})$, the updated boundary of $R(\alpha) \setminus \alpha$ in $\mathcal{V}_l(\mathcal{P}) \oplus \beta$ remains an s_α-monotone path, by the definition of $J(\beta)$. This includes the case when β splits an arc $\alpha \in \mathcal{P}$ in two arcs $\alpha_1, \alpha_2 \in \mathcal{P}_\beta$. In this case, by Theorem 2, $J(\beta)$ splits $R(\alpha, \mathcal{P})$ in two regions $R(\alpha_1)$ and $R(\alpha_2)$ in $\mathcal{V}_l(\mathcal{P}) \oplus \beta$, whose boundaries remain s_α-monotone paths. Thus, for any arc $\alpha \in \mathcal{P}_\beta$, the boundary of the region $R(\alpha)$ in $\mathcal{V}_l(\mathcal{P}) \oplus \beta$, $\partial R(\alpha) \setminus \alpha$, is an s_α-monotone path, satisfying the first requirement of Definition 6.

Since $J(\beta)$ can enter a region in $\mathcal{V}_l(\mathcal{P})$ at most once (except from Observation 1(c) which is settled above) $J(\beta)$ cannot *cut out* of any region some face in the interior of $D_{\mathcal{P}}$. In addition, the regions of arcs in $\mathcal{P} \setminus \mathcal{P}_\beta$ get entirely deleted from $\mathcal{V}_l(\mathcal{P})$ as $J(\beta)$ cannot pass through them, by Theorem 2. Thus, any face of $\mathcal{V}_l(\mathcal{P}) \oplus \beta$ must be incident to $\mathcal{P}_\beta$, and thus, the second requirement of Definition 6 is also satisfied. Hence, $\mathcal{V}_l(\mathcal{P}) \oplus \beta$ satisfies the requirements of a Voronoi-like diagram on the boundary curve $\mathcal{P}_\beta$. Since, by Theorem 1, the Voronoi-like diagram of a boundary curve is unique, it follows that $\mathcal{V}_l(\mathcal{P}) \oplus \beta = \mathcal{V}_l(\mathcal{P}_\beta)$. □

The tracing of the merge curve $J(\beta)$ within $\mathcal{V}_l(\mathcal{P})$, given the endpoints of β, can be done similarly to any ordinary Voronoi diagram, see e.g., [Aurenhammer et al., 2013, Ch. 7.5.3], thanks to Lemma 7. In particular, when computing $J(\beta)$ and we enter some region $R(\gamma)$ at a point v_i, we scan $\partial R(\gamma)$ counterclockwise for the first intersection with $J(s_\beta, s_\gamma)$ to determine v_{i+1}. This is possible in a Voronoi-like diagram because of the cut property, Lemma 7, which assures that

no other intersection of $J(s_\beta, s_\gamma)$ with $\partial R(\gamma)$ is possible, between v_i and v_{i+1} as shown in Figure 3.17. Thus, we derive the following lemma.

Lemma 9. *Let $e_i = (v_i, v_{i+1})$ be an edge of $J(\beta)$ in $R(\gamma)$. Given v_i, we can determine v_{i+1} by sequentially scanning $\partial R(\gamma)$ counterclockwise from v_i (away from γ). No intersection of $J(s_\beta, s_\gamma)$ with $\partial R(\gamma)$ is possible between v_i and v_{i+1}.*

To identify the first edge of $J(\beta)$, special care is required in cases (c), (d), and (e) of Observation 1. In these cases, β may not overlap with any feature of $\mathcal{V}_l(\mathcal{P})$, thus, a starting point for tracing $J(\beta)$ is not readily available. Details are given in the following lemma. The statement of this lemma is an adaptation from a respective statement of Khramtcova and Papadopoulou [2017] for the farthest-segment Voronoi diagram, in the more general environment of an abstract Voronoi-like diagram.

Let $\tilde{\mathcal{P}}$ denote a finer version of $\mathcal{P}$ derived by intersecting its Γ-arcs with $\mathcal{V}_l(\mathcal{P})$. That is, any Γ-arcs of $\mathcal{P}$ may be partitioned into finer pieces by the incident faces of $\mathcal{V}_l(\mathcal{P})$. Since the complexity of $\mathcal{V}_l(\mathcal{P})$ is $O(|\mathcal{P}|)$, it follows that $|\tilde{\mathcal{P}}|$ is also $O(|\mathcal{P}|)$ (recall that $|\cdot|$ denotes complexity).

We first define some parameters for the time complexity analysis and fix a ccw order throughout the definition.

Definition 8. *Let α and γ denote the original arcs preceding and following β on $\mathcal{P}_\beta$. We use the following parameters:*

1. *Let $d_1(\beta, \mathcal{P}_\beta)$ denote the number of auxiliary arcs that appear on $\mathcal{P}_\beta$ from α to β (or equivalently from β to γ).*

2. *Let $d_2(\beta, \mathcal{P}_\beta)$ denote the number of auxiliary arcs in $\mathcal{P}$ that get deleted by the insertion of β.*

3. *In case (c) of Observation 1, where an arc ω is split by the insertion of β in two arcs (ω_1, ω_2), let $r(\beta, \mathcal{P}_\beta) = \min\{|\partial R(\omega_1, \mathcal{P}_\beta)|, |\partial R(\omega_2, \mathcal{P}_\beta)|\}$. In all other cases, $r(\beta, \mathcal{P}_\beta) = 0$.*

4. *In case (d) of Observation 1, where β splits a Γ-arc, let $\tilde{d}(\beta, \mathcal{P}_\beta)$ denote the number of fine Γ-arcs on $\tilde{\mathcal{P}}_\beta$ from α to β (i.e., the number of faces in $\mathcal{V}_l(\mathcal{P}_\beta)$ incident to Γ from α to β). In all other cases, $\tilde{d}(\beta, \mathcal{P}_\beta) = 0$.*

Lemma 10. *Given α, γ, and $\mathcal{V}_l(\mathcal{P})$, in all cases of Observation 1, the merge curve $J(\beta)$ can be computed in time $O(|J(\beta)| + d_1(\beta, \mathcal{P}_\beta) + \tilde{d}(\beta, \mathcal{P}_\beta) + d_2(\beta, \mathcal{P}_\beta) + r(\beta, \mathcal{P}_\beta))$.*

Proof. We assume a ccw traversal. We first determine the endpoints of arc β, by scanning sequentially the arcs in $\mathcal{P}$ from α to β, in time $O(d_1(\beta,\mathcal{P}_\beta)+d_2(\beta,\mathcal{P}_\beta))$. This scan further determines which insertion case of Observation 1 is concerned.

Let $T(\beta)$ denote the portion of $\mathcal{V}_l(\mathcal{P})$ that is enclosed by $J(\beta)$ and $\mathcal{P}\setminus\mathcal{P}_\beta$, i.e., the portion deleted by the insertion of β. By Theorem 2, $T(\beta)$ is a plane forest that is incident to the following faces of $\mathcal{V}_l(\mathcal{P})$: (1) one face for each bisector edge of $J(\beta)$; (2) one face for each auxiliary arc $\alpha' \in \mathcal{P}\setminus\mathcal{P}_\beta$. We infer that $T(\beta)$ has complexity $O(|J(\beta)| + d_2(\beta,\mathcal{P}_\beta))$.

To compute $J(\beta)$ we essentially trace $T(\beta)$ in time $O(|T(\beta)|)$ as in an ordinary Voronoi diagram, which is possible due to Lemma 9. However, we first need to determine one leaf of $T(\beta)$. The graph $T(\beta)$ can have a leaf on $\mathcal{P}$ or not.

First, we consider the case that $T(\beta)$ has a leaf on $\mathcal{P}$. Then, in all cases except in cases (d) and (e) a leaf is determined by sequentially scanning the arcs in $\mathcal{P}$ from α to β, in time $O(d_1(\beta,\mathcal{P}_\beta)+d_2(\beta,\mathcal{P}_\beta))$. If β has exactly one endpoint on Γ (which is possible in case (e)), then additionally fine Γ-arcs of $\tilde{\mathcal{P}}\setminus\tilde{\mathcal{P}}_\beta$ have to be scanned to find a leaf. Since these Γ-arcs are all part of $T(\beta)$, this can be done in time $O(|T(\beta)|)$. Otherwise, β has both endpoints on Γ (case (d)) and all fine Γ-arcs of $\tilde{\mathcal{P}}_\beta$ between α and β have to be scanned to find a leaf, which adds the term $O(\tilde{d}(\beta,\mathcal{P}_\beta))$ to the time. In case (d) neither auxiliary arcs get deleted nor does any boundary arc get split.

In the remainder of the proof we consider the special case that $T(\beta)$ has no leaf on $\mathcal{P}$, i.e. β is enclosed in a single Voronoi-like region, which is possible in cases (c)-(e) of Observation 1.

In case (c) of Observation 1, where the insertion of arc β splits arc ω in two parts, ω_1 and ω_2, we sequentially scan $\partial R(\omega)$, starting at both endpoints of ω in parallel until we determine an intersection with $J(s_\omega, s_\beta)$, see Figure 3.18. The scanned sequence of edges becomes a portion of $\partial R(\omega_i)$, where $i \in \{1,2\}$ determines the shorter of the two sequences. This adds the term $r(\beta,\mathcal{P}_\beta) = \min\{|\partial R(\omega_1,\mathcal{P}_\beta)|, |\partial R(\omega_2,\mathcal{P}_\beta)|\}$ to the overall time complexity. In case (c) no auxiliary arcs get deleted by the insertion of β, i.e., $d_2(\beta,\mathcal{P}_\beta) = 0$.

In cases (d) and (e) (if $T(\beta)$ has no leaf on $\mathcal{P}$), arc β has at least one endpoint on a fine Γ-arc and β lies entirely in a region $R(\delta)$. In this case $J(\beta)$ has constant length ($m = 3$ or $m = 4$), and thus, it can be computed in $O(1)$ time (after identifying δ). In particular, if β has one endpoint on Γ and one on δ (case (e)), then δ is already identified and $J(\beta)$ equals $J(s_\delta, s_\beta) \cap D_\Gamma$ plus one Γ-arc, see Fig. 3.19. Otherwise, β must have both endpoints on Γ (case (d)), and $J(\beta)$ consists of $J(s_\delta, s_\beta) \cap D_\Gamma$ plus the two incident Γ-arcs, see Fig. 3.20. In this case we can identify δ in $O(\tilde{d}(\beta,\mathcal{P}_\beta))$ time by scanning the fine Γ-arcs on $\tilde{\mathcal{P}}$ from α

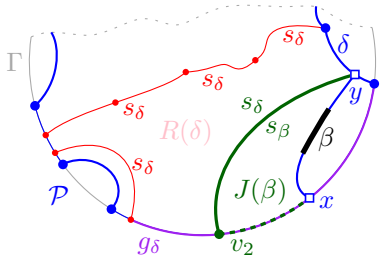

Figure 3.19. Endpoint x lies on a fine Γ-arc g_δ bounding $R(\delta)$, and $y \in \delta$. If $v_2 \in g_\delta$ then $J(\beta) = (x, v_2, y)$.

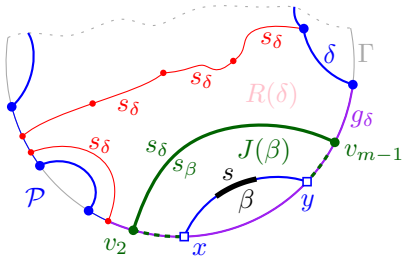

Figure 3.20. Both x, y lie on a fine Γ-arc g_δ bounding $R(\delta)$. If $v_2, v_{m-1} \in g_\delta$ then $J(\beta) = (x, v_2, v_{m-1}, y)$.

to β. □

Lemma 10 expresses the time complexity necessary to insert arc β in $\mathcal{V}_l(\mathcal{P})$ in terms of the resulting diagram $\mathcal{V}_l(\mathcal{P}_\beta)$, except from term $d_2(\beta, \mathcal{P}_\beta)$.

3.3.1 Proving Theorem 2

We first establish that $J(\beta)$ cannot intersect arc β, other than its endpoints. To this aim we use Lemma 5.

First observe that the open portion of β is contained in $D_\mathcal{P}$, thus, it is partitioned into smaller pieces by the regions of $\mathcal{V}_l(\mathcal{P})$. By Lemma 5 bisector $J(s_\beta, s_\alpha)$ cannot intersect $J(s, s_\beta)$ within any region $R(\alpha)$ of $\mathcal{V}_l(\mathcal{P})$, because if it did, $J(s, s_\alpha)$ would also pass through the same intersection point in $R(\alpha)$ contradicting that $R(\alpha) \subseteq D(s, s_\alpha)$. Thus, no edge of $J(\beta)$ can intersect arc β in its interior.

We use the following observation throughout the proofs in this section.

Lemma 11. *For any $p \in S$, $D(s,p) \cap D_\mathcal{P}$ is connected. Thus, any components of $J(s,\cdot) \cap D_\mathcal{P}$ must appear sequentially along $\mathcal{P}$.*

Proof. If we assume the contrary we obtain a forbidden s-inverse cycle defined by $J(s,\cdot)$ and $\mathcal{P}$. □

We now establish that $J(\beta)$ cannot pass through any region of auxiliary arcs in $\mathcal{P} \setminus \mathcal{P}_\beta$ that are deleted by the insertion of β.

Lemma 12. *Let $\alpha \in \mathcal{P}$ but $\alpha \notin \mathcal{P}_\beta$. Then $R(\alpha) \subset D(s_\beta, s_\alpha)$.*

Proof. By Lemma 5, it holds that $R(\alpha) \subseteq D(s, s_\alpha)$. Let $R_s = R(\alpha) \cap D(s, s_\beta)$ and $R_\beta = R(\alpha) \cap D(s_\beta, s)$. By transitivity of dominance regions we have $R_\beta \subseteq D(s_\beta, s_\alpha)$. By Lemma 11, R_s is not incident to α. Thus, if $J(s_\beta, s_\alpha)$ intersected R_s then it

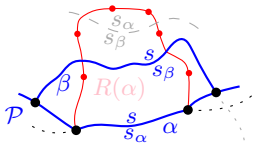

Figure 3.21. Illustrations for Lemma 12.

would create a forbidden s_α-cycle contradicting Lemma 6, see the dashed gray line in Figure 3.21. This implies that also $R_s \subseteq D(s_\beta, s_\alpha)$. Thus, $R(\alpha) = R_s \cup R_\beta \subseteq D(s_\beta, s_\alpha)$. □

In the following we prove that $J(\beta)$ is an s_β-monotone path connecting the endpoints of β. To this aim we perform a bi-directional induction on the vertices of $J(\beta)$.

Let $J_x^i = (v_1, v_2, \ldots, v_i), 1 \le i < m$, be the subpath of $J(\beta)$ starting at $v_1 = x$ up to vertex v_i, including a small neighborhood of e_i incident to v_i, see Figure 3.16. Note that vertex v_i uniquely determines e_i, however, its other endpoint is not yet specified. Similarly, let $J_y^j = (v_m, v_{m-1}, \ldots, v_{m-j+1}), 1 \le j < m$, denote the subpath of $J(\beta)$, starting at v_m up to vertex v_{m-j+1}, including a small neighborhood of edge e_{m-j}. For any bisector edge $e_\ell \in J(\beta)$, let α_ℓ denote the boundary arc that *induces* e_ℓ, i.e., $e_\ell \subseteq J(s_{\alpha_\ell}, s_\beta) \cap R(\alpha_\ell)$.

Induction hypothesis: Suppose J_x^i and J_y^j, $i, j \ge 1$, are disjoint s_β-monotone paths. Suppose further that each bisector edge of J_x^i and of J_y^j passes through a distinct region of $\mathcal{V}_l(\mathcal{P})$: α_ℓ is distinct for ℓ, $1 \le \ell \le i$ and $m - j \le \ell < m$, except possibly $\alpha_i = \alpha_{m-j}$ and $\alpha_1 = \alpha_{m-1}$.

Induction step: Assuming that $i + j < m$, we prove that at least one of J_x^i or J_y^j can respectively grow to J_x^{i+1} or J_y^{j+1} at a valid vertex (Lemmas 13, 14), and it enters a new region of $\mathcal{V}_l(\mathcal{P})$ that has not been visited so far (Lemma 16). A finish condition when $i + j = m$ is given in Lemma 15. The base case for $i = j = 1$ is trivially true.

Suppose that $e_i \subseteq J(s_{\alpha_i}, s_\beta)$ and $v_i \in \partial R(\alpha_i)$. To show that v_{i+1} is a valid vertex it is enough to show that (1) v_{i+1} can not be on α_i, and (2) if v_i is on a Γ-arc then v_{i+1} can be determined on the same Γ-arc. However, we cannot easily derive these conclusions directly. Instead we show that if v_{i+1} is not valid then v_{m-j} will have to be valid.

In the following lemmas we assume that the induction hypothesis holds.

Lemma 13. *Suppose $e_i \subseteq J(s_{\alpha_i}, s_\beta)$ but $v_{i+1} \in \alpha_i$, that is, e_i hits arc $\alpha_i \in \mathcal{P}$, and thus, v_{i+1} is not a valid vertex. Then vertex v_{m-j} must be a valid vertex in $\mathscr{A}(\mathcal{J}_{s_\beta, S_\mathcal{P}})$, and v_{m-j} can not be on $\mathcal{P}$.*

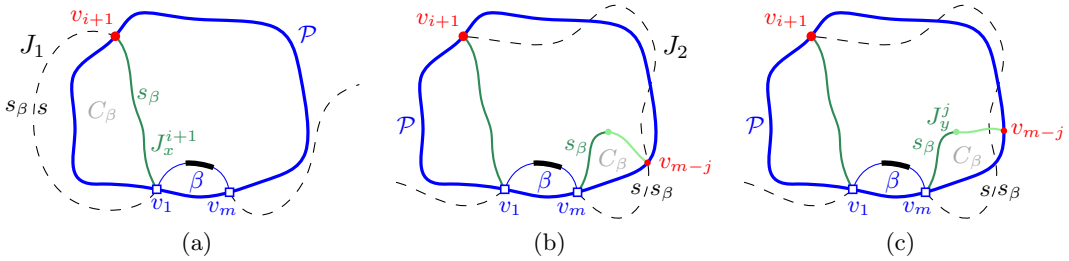

Figure 3.22. The assumption that edge $e_i = (v_i, v_{i+1})$ of the merge curve J_x^i hits a boundary arc of $\mathcal{P}$ as in Lemma 13.

Proof. Suppose vertex v_{i+1} of e_i lies on arc α_i as shown in Figure 3.22(a). Vertex v_{i+1} is the intersection point of related bisectors $J(s, s_{\alpha_i})$, $J(s_\beta, s_{\alpha_i})$ and thus also of $J(s, s_\beta)$. Thus, $v_1, v_m, v_{i+1} \in J(s, s_\beta)$. By the induction hypothesis, no other vertex of J_x^i nor J_y^j can be on $J(s, s_\beta)$. Vertices v_1, v_{i+1}, v_m appear on $\mathcal{P}$ in clockwise order, because J_x^{i+1} cannot intersect β. Arc β partitions $J(s, s_\beta)$ in two parts: J_1 incident to v_1 and J_2 incident to v_m. We claim that v_{i+1} must lie on J_2, as otherwise, J_x^{i+1} and J_1 would form a forbidden s_β-inverse cycle, see the dashed black and the green solid curve in Figure 3.22(a), contradicting Lemma 2. This cycle must be s_β-inverse because $J_x^{i+1} \subseteq \overline{D_\mathcal{P}}$, and all components of $J(s, \cdot) \cap D_\mathcal{P}$ must appear sequentially along $\mathcal{P}$ by Lemma 11.

Thus, v_{i+1} lies on J_2. Further, by Lemma 11, the components of $J_2 \cap D_\mathcal{P}$ appear on $\mathcal{P}$ clockwise after v_{i+1} and before v_m, as shown in Figure 3.22(b), which illustrates $J(s, s_\beta)$ as a black dashed curve.

Now consider J_y^j. We show that v_{m-j} cannot be on $\mathcal{P}$. First observe that v_{m-j} can not lie on $\mathcal{P}$, clockwise after v_m and before v_1, since J_y^{j+1} cannot cross β. Now we prove that v_{m-j} cannot lie on $\mathcal{P}$ clockwise after v_1 and before v_{i+1}. To see that, note that edge e_{m-j} cannot cross any non-Γ edge of J_x^{i+1}, because by the induction hypothesis, α_{m-j} is distinct from all $\alpha_\ell, \ell \leq i$. In addition, by the definition of a Γ-arc, v_{m-j} cannot lie on any Γ-arc of J_x^i. Finally, we show that v_{m-j} cannot lie on $\mathcal{P}$ clockwise after v_{i+1} and before v_m. If v_{m-j} lay on the boundary arc α_{m-j} then we would have $v_{m-j} \in J(s, s_\beta)$. This would define an s_β-inverse cycle C_β, formed by J_y^{j+1} and $J(s_\beta, s)$, see Figure 3.22(b), similarly to the first paragraph of this proof. If v_{m-j} lay on a Γ-arc then there would also be a forbidden s_β-inverse cycle formed by J_y^{j+1} and $J(s, s_\beta)$ because in order to reach Γ, edge e_i must cross $J(s, s_\beta)$. See the dashed black and the green curve in Figure 3.22(c). Thus $v_{m-j} \notin \mathcal{P}$.

Since $v_{m-j} \in \partial R(\alpha_{i+1})$ but $v_{m-j} \notin \mathcal{P}$, it must be a vertex of $\mathscr{A}(\mathcal{J}_{s_\beta, S_\mathcal{P}})$. □

The proof for the following lemma is similar.

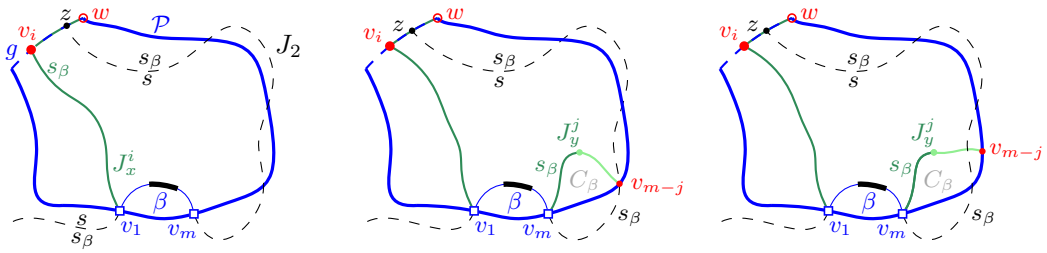

Figure 3.23. The assumption that $v_i \in \Gamma$ and v_{i+1} of the merge curve J_x^i cannot be determined as in Lemma 14.

Lemma 14. *Suppose vertex v_i is on a Γ-arc $g \in \mathcal{P}$ but v_{i+1} cannot be determined because no bisector $J(s_\beta, s_\gamma)$ intersects $\overline{R(\gamma)} \cap g$, clockwise from v_i. Then vertex v_{m-j} must be a valid vertex in $\mathscr{A}(\mathcal{J}_{s_\beta, S_\mathcal{P}})$ and v_{m-j} can not be on $\mathcal{P}$.*

Proof. We truncate the Γ-arc g to its portion clockwise from v_i; let w be the endpoint of g clockwise from v_i, see Figure 3.23(a). If no $J(s_\beta, s_\gamma) \cap R(\gamma)$ intersects g, as we assume in this lemma, then $R(\gamma) \cap g \subseteq D(s_\beta, s_\gamma)$, for any region $R(\gamma)$ incident to g. Thus, $w \in D(s_\beta, s)$. However, $v_i \in D(s, s_\beta)$, since, by Lemma 5, $R(\alpha_{i-1}) \subseteq D(s, s_{\alpha_{i-1}})$ and v_i is incident to $J(s_\beta, s_{\alpha_{i-1}}) \cap R(\alpha_{i-1})$. Thus, $J(s, s_\beta)$ must intersect g at some point z clockwise from v_i. Arc β partitions $J(s, s_\beta)$ in two parts: J_1 incident to v_1 and J_2 incident to v_m. Lemma 11 implies that all components of $J_2 \cap D_\mathcal{P}$ appear on $\mathcal{P}$ clockwise after v_i and before v_m, as shown by the black dashed curve in Figure 3.23(a); also z lies on J_2.

Now we can show that vertex v_{m-j} of J_y^j cannot be on $\mathcal{P}$ analogously to the proof of Lemma 13. The only difference is that we must additionally show that v_{m-j} cannot lie on $\mathcal{P}$ clockwise after v_i and before w. But this holds already by the assumption in the lemma statement. Refer to Figures 3.23(b) and (c).

We conclude that v_{m-j} cannot lie on $\mathcal{P}$ and it is a valid vertex of $\mathscr{A}(\mathcal{J}_{s_\beta, S_\mathcal{P}})$. □

Lemma 15 in the sequel provides a finish condition for the induction, when J_x^i and J_y^j are incident to a common region or to a common Γ-arc. When it is met, the merge curve $J(\beta)$ is a concatenation of J_x^i and J_y^j.

Lemma 15. *Suppose $i + j > 2$ and either (1) or (2) holds: (1) v_i and v_{m-j+1} are incident to a common region $R(\alpha_i)$ and $e_i, e_{m-j} \subseteq J(s_\beta, s_{\alpha_i})$, i.e., $\alpha_i = \alpha_{m-j}$; or (2) v_i and v_{m-j+1} are on a common Γ-arc g of $\mathcal{P}$ and $e_i, e_{m-j} \subseteq \Gamma$. Then $v_{i+1} = v_{m-j+1}$, $v_{m-j} = v_i$, and $m = i + j$.*

Proof. Let $\alpha = \alpha_i$. Suppose (1) holds, then $e_i, e_{m-j} \subseteq J(s_\beta, s_\alpha)$, see Figure 3.24(a). The boundary $\partial R(\alpha_i)$ is partitioned in four parts, using a counterclockwise traver-

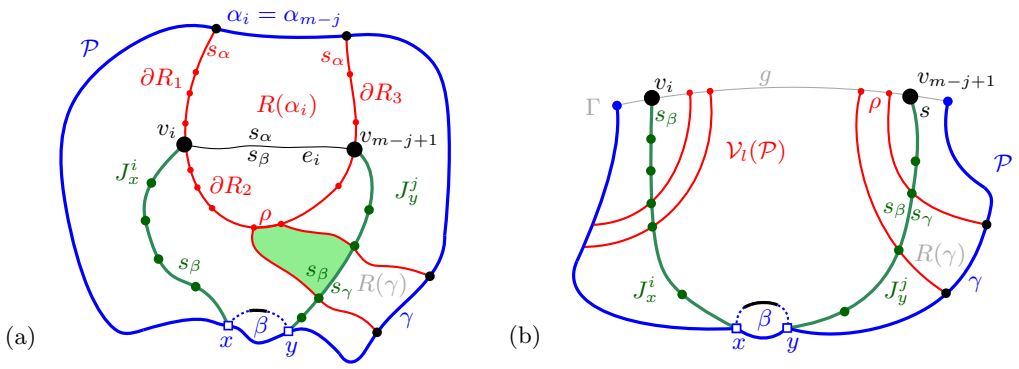

Figure 3.24. Illustrations for Lemma 15. (a) corresponds to condition (1) and (b) to condition (2).

sal starting at α_i: ∂R_1, from the endpoint of arc α_i to v_i; ∂R_2, from v_i to v_{m-j+1}; ∂R_3, from v_{m-j+1} to the next endpoint of α_i; and arc α_i. We show that e_i and e_{m-j} cannot hit any of these parts; thus, $e_i = e_{m-j}$.

1. Edge e_i cannot hit ∂R_1 and edge e_{m-j} cannot hit ∂R_3 by the cut property, Lemma 7.

2. We prove that edge e_i cannot hit ∂R_2. Analogously for edge e_{m-j}. Let ρ be any edge on ∂R_2. (If $v_i \in \rho$ or $v_{m-j+1} \in \rho$, assume that ρ is truncated with endpoint v_i or v_{m-j+1} respectively).

   (a) Suppose that ρ is a bisector edge, $\rho \subseteq J(s_\alpha, s_\gamma)$, see Figure 3.24(a). Then at least one of J_y^j, J_x^i, or β must pass through $R(\gamma)$. Suppose that J_y^j does, as shown in Figure 3.24(a). Then by the cut property (Lemma 7) $\rho \subseteq D(s_\beta, s_\gamma)$. By transitivity (Lemma 3) it also holds that $\rho \subseteq D(s_\beta, s_\alpha)$. Thus, e_i cannot hit ρ. Symmetrically for J_x^i. If only β passes through $R(\gamma)$, then we can use Lemma 12 to derive that $\rho \subseteq D(s_\beta, s_\gamma)$; the rest follows.

   (b) Suppose that $\rho \subseteq \Gamma$. Then either ρ itself is part of an edge of J_y^j or of J_x^i, or β passes through $R(\alpha)$ and ρ is at opposite side of it than α. In the former case, $\rho \subseteq D(s_\beta, s_\alpha)$ by the definition of a Γ-edge in the merge curve. In the latter case, the same is derived by Lemma 5 and transitivity (Lemma 3). Thus, e_i cannot hit ρ.

3. Edge e_i (resp. e_{m-j}) cannot hit ∂R_3 because if it did, e_i and e_{m-j} would not appear sequentially on $R(\alpha_i)$ contradicting Lemma 7.

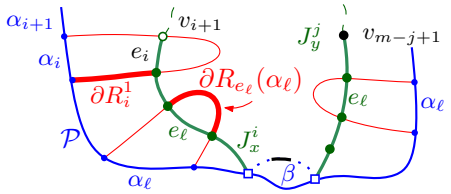

Figure 3.25. Illustration for Lemma 16.

4. It remains to show that e_i and e_{m-j} cannot both hit α_i. But this is already shown in Lemma 13.

Now suppose (2) holds, see Figure 3.24(b). Let $R(\gamma)$ be a region in $\mathcal{V}_l(\mathcal{P})$ incident to the Γ-arc g and let $\rho = R(\gamma) \cap g$ be the Γ-arc bounding $R(\gamma)$, which lies between v_i and v_{m-j+1}. At least one of J_y^j or J_x^i or β must pass through $R(\gamma)$. By the exact same arguments as before, $\rho \subseteq D(s_\beta, s_\gamma)$. We infer that there is no bisector $J(s_\beta, s_\gamma)$ in $R(\gamma)$, for any region $R(\gamma)$ incident to g between v_i and v_{m-j+1}. Thus, $e_{i+1} = e_{m-j+1} \subseteq g$.

Thus, in both (1) and (2) $v_{i+1} = v_{m-j+1}$, $v_{m-j} = v_i$, and $m = i + j$. $J(\beta)$ is the concatenation of J_x^i and J_y^j with $e_{i+1} = e_{m-j+1}$. □

Lemma 16. *Suppose vertex v_{i+1} is valid and $e_{i+1} \subseteq J(s_\beta, s_{a_{i+1}})$. Then $R(\alpha_{i+1})$ has not been visited by J_x^i nor J_y^j, i.e., $\alpha_{i+1} \neq \alpha_\ell$ for $\ell \leq i$ and for $m - j < \ell$.*

Proof. Let $e_k, k \leq i$, be a bisector edge of J_x^i. Denote by ∂R_k^1 the portion of $\partial R(\alpha_k)$ from α_k to v_k in a counterclockwise traversal, see the bold red part ∂R_i^1 in Figure 3.25. Analogously for a bisector edge e_{m-j} of J_y^j , where ∂R_{m-j}^1 is defined in a clockwise traversal of $\partial R(\alpha_{m-j})$. Recall that $\partial R_{e_k}(\alpha_k)$, denotes the portion of $\partial R(\alpha_k)$ *cut out* by edge e_k, at opposite side from α_k.

The cut property of Lemma 7 implies that v_{i+1} cannot be on $\partial R_{e_\ell}(\alpha_\ell)$ for any ℓ, $\ell < i$ and $m - j < \ell$ and that v_{i+1} cannot be on ∂R_i^1. This implies that v_{i+1} cannot be on ∂R_ℓ^1 for any $\ell < i$, because we have a plane graph in $D_\mathcal{P}$ and by its layout ∂R_ℓ^1 is not reachable from e_i without first hitting $\partial R_{e_\ell}(\alpha_\ell)$ or ∂R_i^1. See Figure 3.25. Thus, v_{i+1} can not be on $\partial R(\alpha_\ell)$, $\ell < i$. By Lemma 15 v_{i+1} cannot be on ∂R_{m-j}^1. This implies, again by the layout, that v_{i+1} cannot be on ∂R_ℓ^1 for all $\ell > m - j$. Thus, v_{i+1} can not be on $\partial R(\alpha_\ell)$, for any $\ell > m - j$. This implies that $\alpha_{i+1} \neq \alpha_\ell$, for any ℓ, $\ell \leq i$ or $\ell > m - j$. □

By Lemma 16, J_x^{i+1} and J_y^{j+1} always enter a new region of $\mathcal{V}_l(\mathcal{P})$ that has not been visited in any previous step. Thus, conditions (1) or (2) of Lemma 15 must be fulfilled at some point of the induction. Thus, the induction in the proof of Theorem 2 is complete.

Completing the bi-directional induction establishes also the remaining properties for $J(\beta)$. First, $J(\beta)$ can never enter the same region twice (by Lemma 16), except $R(\alpha_1)$, if $R(\alpha_1) = R(\alpha_m)$. This is Observation 1(c) where arc β splits a single arc $\alpha \in \mathcal{P}$. In this case $J(\beta)$ enters $R(\alpha, \mathcal{P})$ exactly twice and $e_1, e_{m-1} \subseteq J(s_\alpha, s_\beta)$. This is because $J(\beta)$ must intersect $\partial R(\alpha, \mathcal{P})$ in this case, i.e., $J(\beta) \not\subseteq R(\alpha, \mathcal{P})$. If $J(\beta) \subseteq R(\alpha, \mathcal{P})$, then $J(\beta) = J(s_\alpha, s_\beta)$ contradicting the labeling of the cut property (Lemma 7) (see Fig 3.11(b)).

Completing the induction for Theorem 2 establishes also that $J(\beta)$ is unique and that the conditions of Lemmas 13 and 14 can never be met. Thus, no vertex of $J(\beta)$, except its endpoints, can be on a boundary arc of $\mathcal{P}$.

3.4 Conclusion

In this chapter we have introduced the *Voronoi-like diagram*. It is defined relative to a Voronoi region $\mathrm{VR}(s, S)$ and is a relaxed version of a Voronoi diagram. Voronoi-like regions are supersets of real Voronoi regions. A Voronoi-like diagram $\mathcal{V}_l(\mathcal{P})$ is defined for a *boundary curve* $\mathcal{P}$, based on the notions of *monotone paths* in the arrangement of abstract bisectors. A boundary curve $\mathcal{P}$ is defined for a subset of *arcs* $\mathcal{S}' \subseteq \mathcal{S}$, which are Voronoi edges of $\partial\mathrm{VR}(s, S)$.

Moreover, we have introduced an *insertion operation* $\oplus$: given a Voronoi-like diagram $\mathcal{V}_l(\mathcal{P})$, where $\mathcal{P}$ is a boundary curve for $\mathcal{S}'$, we can insert an arc $\beta \in \mathcal{S} \setminus \mathcal{S}'$ into $\mathcal{V}_l(\mathcal{P})$ in order to obtain a new Voronoi-like diagram $\mathcal{V}_l(\mathcal{P}) \oplus \beta$. Establishing the correctness of the insertion of a new Voronoi-like region was particularly difficult (Section 3.3.1). Voronoi-like diagrams lack the standard monotonicity property[4] of real Voronoi diagrams, but instead we established the *cut property* (Lemma 7), which gives a powerful tool to derive many of our proofs. The cut property allows also to trace a newly inserted region efficiently as in ordinary Voronoi diagrams (Lemma 9).

[4]The monotonicity property for Voronoi diagrams is the following: If $S_1 \subseteq S_2$, then $\mathrm{VR}(p, S_2) \subseteq \mathrm{VR}(p, S_1)$, $p \in S_1$. In Voronoi-like diagrams a corresponding statement is not true: Let $\mathcal{S}_1 \subset \mathcal{S}_2 \subseteq \mathcal{S}$ and let $\mathcal{P}_1$, $\mathcal{P}_2$ be two arbitrary boundary curves of $\mathcal{S}_1$ and $\mathcal{S}_2$ respectively. Then for $\alpha \in \mathcal{S}_1$, it is possible that $R(\alpha, \mathcal{P}_1) \not\supseteq R(\alpha, \mathcal{P}_2)$. See Section 7.1 for more details and examples.

Chapter 4

Existence and uniqueness of Voronoi-like diagrams

In this chapter we establish that given a boundary curve $\mathcal{P}$, its Voronoi-like diagrams $\mathcal{V}_l(\mathcal{P})$ exists (Section 4.2), and $\mathcal{V}_l(\mathcal{P})$ is unique (Section 4.1) proving Theorem 1 from Section 3.2.

Section 4.1 is based on the following publication:

Kolja Junginger and Evanthia Papadopoulou. Deletion in Abstract Voronoi Diagrams in Expected Linear Time. In *34th International Symposium on Computational Geometry* (SoCG 2018), Dagstuhl, Germany.

The corresponding longer arXiv version contains all proofs:

Kolja Junginger and Evanthia Papadopoulou. *Deletion in abstract Voronoi diagrams in expected linear time*, CoRR abs/1803.05372. URL: `http://arxiv.org/abs/1803.05372`

Section 4.2 is based on the following paper:

Kolja Junginger and Evanthia Papadopoulou. Abstract tree-like Voronoi diagrams in expected linear time. In *CGWeek Young Researchers Forum 2019*, Computational Geometry week, Portland, Oregon, 2019.

4.1 Uniqueness

In this section the notation β refers to some arc that is not included in $\mathcal{P}$.

Consider an arbitrary region $R(\alpha)$ in $\mathcal{V}_l(\mathcal{P})$. The following lemma establishes an essential property of Voronoi-like regions, and completes the *cut property* of Lemma 7. It essentially states that if a bisector $J(s_\alpha, \cdot)$ appears within $R(\alpha)$ (something that can never happen in a real Voronoi region) then we are *missing an arc*

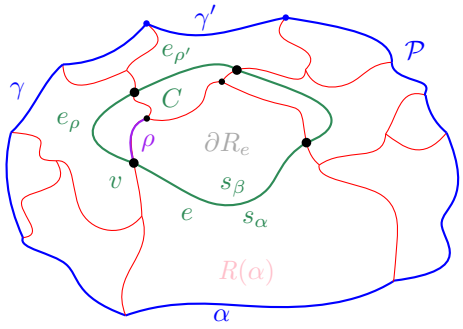

Figure 4.1. A component e of $J(s_\alpha,\cdot)$ in $R(\alpha)$ as in Lemma 17.

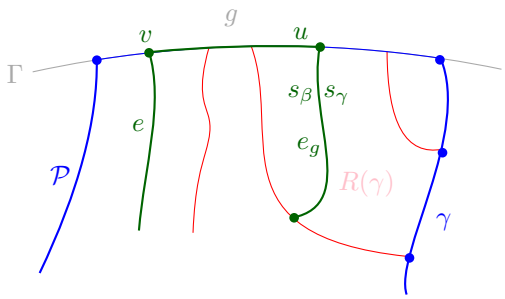

Figure 4.2. A component e of $J(s_\alpha,\cdot)$ in $R(\alpha)$ with its endpoint v on a Γ-arc g as in Lemma 17.

from $\mathcal{P}$. Notation β refers to some arc that is not included in $\mathcal{P}$ and appears in $D_\mathcal{P}$.

Lemma 17. *Suppose there is a non-empty component e of $J(s_\alpha,\cdot)$ intersecting $R(\alpha)$ in $\mathcal{V}_l(\mathcal{P})$. Then $J(s,\cdot)$ must also intersect $D_\mathcal{P}$. Further, there exists a component of $J(s,\cdot)\cap D_\mathcal{P}$, denoted as β, such that the merge curve $J(\beta)$ in $\mathcal{V}_l(\mathcal{P})$ contains e.*

We say that boundary arc β is *missing from* $\mathcal{P}$.

Proof. Suppose there is a non-empty component e of $J(s_\alpha,s_\beta)\cap R(\alpha)$, however, $J(s,s_\beta)\cap D_\mathcal{P}=\emptyset$.

Since $J(s,s_\beta)\cap D_\mathcal{P}=\emptyset$, and thus, $D_\mathcal{P}\subseteq D(s,s_\beta)$, and because of the transitivity of dominance regions (Lemma 3), it follows that for any arc $\chi\in\mathcal{P}$, $\chi\subseteq D(s_\chi,s_\beta)$. Let ∂R_e denote the portion of $\partial R(\alpha)$ cut out by e (at opposite side from α); by Lemma 7, $\partial R_e\subseteq D(s_\beta,s_\alpha)$. Consider an endpoint v of e. We distinguish two cases for v:

1. If v is on an edge ρ incident to regions $R(\alpha)$ and $R(\gamma)$, then $J(s_\beta,s_\gamma)$ intersects $R(\gamma)$ by an edge e_ρ, incident to v, leaving ρ and γ at opposite sides, because $\gamma\subseteq D(s_\gamma,s_\beta)$. See Figure 4.1.

2. If v is on a Γ-arc g, let $R(\gamma)$ be the first region after v (towards $D(s_\beta,s_\alpha)$) with $J(s_\beta,s_\gamma)$ intersecting $g\cap\overline{R(\gamma)}$ at point u (see Figure 4.2). There exists such $R(\gamma)$ because for all boundary arcs $\chi\in\mathcal{P}$, $\chi\subseteq D(s_\chi,s_\beta)$, and this includes the boundary arc that is incident to g. Let e_g be the component of $J(s_\beta,s_\gamma)\cap R(\gamma)$ incident to u.

Thus, given e and v, we derive an edge e', either $e'=e_\rho$ or $e'=e_g$, with the same properties as e, in another region of $\mathcal{V}_l(\mathcal{P})$. This process repeats and there is no

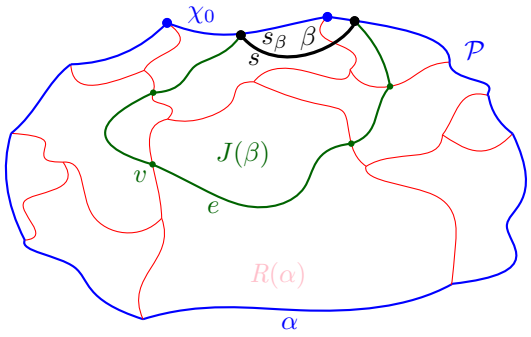

Figure 4.3. Arc $\beta \subseteq J(s, s_\beta)$ in $D_\mathcal{P}$; The merge curve $J(\beta)$ contains e.

way to break it because for any arc $\chi \in \mathcal{P}$, $\chi \subseteq D(s_\chi, s_\beta)$. Thus, we create a closed curve on $\mathcal{V}_l(\mathcal{P})$ consisting of consecutive pieces of $J(s_\beta, .)$, possibly interleaved with Γ-arcs, which has the label s_β in its interior. No two edges of this curve can intersect because otherwise the bisector corresponding to such intersecting edges would not be a Jordan curve. Furthermore, no vertex of this curve can repeat as no three s_β-related bisectors can intersect at the same point, by our general position assumption. Thus, the closed curve must be an s_β-cycle C that is contained in $D_\mathcal{P}$, see Figure 4.1. This is a contradiction by Lemma 6. Thus, our assumption that $J(s, s_\beta) \cap D_\mathcal{P} = \emptyset$ was false, and thus, $J(s, s_\beta)$ must intersect $\mathcal{P}$. The above process must encounter such an intersection as otherwise the forbidden s_β-cycle C would exist. Let $J_e(\beta)$ denote the sequence of encountered edges e_ρ, starting with the initial edge e and ending on the first intersection of an arc χ_0 in $\mathcal{P}$ with $J(s, s_\beta)$. Let β be the component of $J(s, s_\beta) \cap D_\mathcal{P}$ incident to χ_0, see Figure 4.3.

By its definition, the path $J_e(\beta)$ fulfills the definition of the merge curve $J(\beta)$ (Definition 7). Since by Theorem 2 the merge curve $J(\beta)$ on $\mathcal{V}_l(\mathcal{P})$ is unique, it follows that $J(\beta)$ includes $J_e(\beta)$, and thus it includes edge e. □

We can now prove the uniqueness part of Theorem 1 from Section 3.2.

Theorem 4. *Given a boundary curve $\mathcal{P}$ of $\mathcal{S}' \subseteq \mathcal{S}$, $\mathcal{V}_l(\mathcal{P})$ (if it exists) is unique.*

Proof. Let $\mathcal{P}$ be a boundary curve for $\mathcal{S}' \subseteq \mathcal{S}$ such that $\mathcal{P}$ admits a Voronoi-like diagram $\mathcal{V}_l(\mathcal{P})$. Suppose there exist two different Voronoi-like diagrams of $\mathcal{P}$, $\mathcal{V}_l^{(1)} \neq \mathcal{V}_l^{(2)}$. Then there must be an edge $e^{(1)}$ of $\mathcal{V}_l^{(1)}$ bounding regions $R^{(1)}(\alpha)$ and $R^{(1)}(\beta)$ of $\mathcal{V}_l^{(1)}$, where $\alpha, \beta \in \mathcal{P}$, such that $e^{(1)}$ intersects region $R^{(2)}(\alpha)$ of $\mathcal{V}_l^{(2)}$, since α is common to both $R^{(1)}(\alpha)$ and $R^{(2)}(\alpha)$.

Let edge $e \subseteq J(s_\beta, s_\alpha)$ be the component of $R^{(2)}(\alpha) \cap J(s_\beta, s_\alpha)$ overlapping with $e^{(1)}$, see Figure 4.4. From Lemma 17 it follows that there is a non-empty

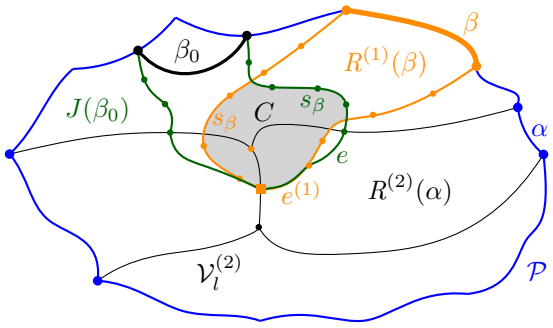

Figure 4.4. Illustrations for the proof of Theorem 1.

component β_0 of $J(s, s_\beta) \cap D_\mathcal{P}$ such that $J(\beta_0)$ on $\mathcal{V}_l^{(2)}$ contains edge e. Since $J(\beta_0)$ and $\partial R^{(1)}(\beta)$ have an overlapping portion $\left(e \cap e^{(1)}\right)$ and they bound the regions of two different arcs $\beta_0 \neq \beta$ of site s_β, they form an s_β-cycle C as shown in Figure 4.4. But C is contained in $D_\mathcal{P}$, deriving a contradiction to Lemma 6. □

4.2 Existence

Consider a Voronoi region $\mathrm{VR}(s, S)$ and let $\mathcal{S}$ denote the (ordered) set of Voronoi edges on its boundary ($\mathcal{S} = \partial \mathrm{VR}(s, S) \cap D_\Gamma$). In this section we prove that, given any boundary curve $\mathcal{P}$, for any $\mathcal{S}' \subseteq \mathcal{S}$, its Voronoi-like diagram $\mathcal{V}_l(\mathcal{P})$ always exists (Theorem 5). Let $S' \subseteq S$ denote the sites that induce the arcs in $\mathcal{S}'$.

Definition 9. *A boundary curve $\mathcal{C}$ for $\mathcal{S}' \subseteq \mathcal{S}$ is called* plain *if it contains no auxiliary arc, i.e., each arc of $\mathcal{C}$ contains a core arc in $\mathcal{S}'$ (see Figure 4.5).*

Lemma 18. *The* plain boundary curve $\mathcal{C}$ *of any $\mathcal{S}' \subseteq \mathcal{S}$ exists and is unique. It can be constructed in time $O(|\mathcal{S}'|)$.*

Proof. Refer to Figure 4.5. For every three consecutive arcs $\alpha, \beta, \gamma \in \mathcal{S}'$, expand β to $\bar{\beta}$ along $J(s, s_\beta)$ until reaching the first intersection with $J(s, s_\alpha)$ (if $s_\alpha \neq s_\beta$) and with $J(s, s_\gamma)$ (if $s_\gamma \neq s_\beta$), or with Γ, respectively. If $s_\alpha = s_\beta$, then expand $\bar{\beta}$ until α. Symmetrically if $s_\gamma = s_\beta$. By construction, $\mathcal{C}$ is an s-monotone path in the arrangement of $\mathcal{J}_{s,S'}$; it contains all the core arcs $\mathcal{S}'$ and contains no auxiliary arcs. The construction is done by going once around the cyclically ordered set $\mathcal{S}'$ and thus, in time $O(|\mathcal{S}'|)$.

If $\mathcal{C}$ contained any additional arc to the above constructed original arcs, it would be auxiliary and the curve would not be plain. Further, $\mathcal{C}$ cannot contain less than these arcs, because a boundary curve must contain all arcs in $\mathcal{S}'$. Thus, $\mathcal{C}$ is unique. □

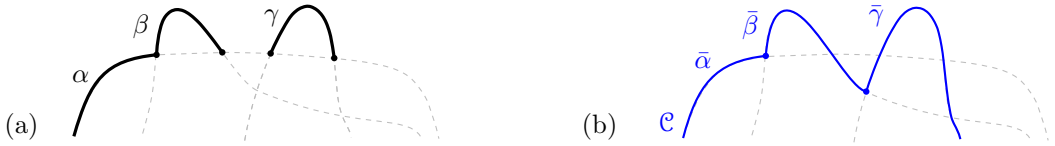

Figure 4.5. (a) Three arcs $\mathcal{S}' = \{\alpha, \beta, \gamma\}$ in black, and (b) their plain boundary curve $\mathcal{C}$ in blue.

Definition 10. *For arcs $\alpha, \beta \in \mathcal{S}$, we say that α* splits *β, if $J(s, s_\beta) \cap D(s, s_\alpha)$ consists of two connected components, see Figure 4.6. This defines a binary* split relation *on $\mathcal{S}$.*

From the fact that related bisectors intersect at most twice we infer that the split relation is anti-symmetric and transitive; thus, it is also acyclic. The split relation induces a partial order on $\mathcal{S}$, where $\alpha < \beta$ iff α splits β.

Lemma 19. *The Voronoi-like diagram of a plain boundary curve $\mathcal{C}$ always exists. It can be constructed in time $O(|\mathcal{S}'|^2)$.*

Proof. Let o be the partial order on $\mathcal{S}'$ inherited from $\mathcal{S}$. Consider any linear extension (topological ordering) of o $(\alpha_1, \dots a_m)$, $m = |\mathcal{S}'|$. We can construct $\mathcal{C}$ and $\mathcal{V}_l(\mathcal{C})$ by inserting the core arcs in $\mathcal{S}'$ in the order of any linear extension (topological ordering) of o, using the insertion operation $\oplus$ of Section 3.3 and applying Theorem 3. In particular, let $\mathcal{P}_1$ be the boundary curve derived by the first core arc in o, i.e., $\mathcal{P}_1 = \partial D(s, s_{\alpha_1}) \cap D_\Gamma$, $\mathcal{V}_l(\mathcal{P}_1) = \emptyset$. Then $\mathcal{P}_i = \mathcal{P}_{i-1} \oplus \alpha_i$, where the arc insertion operation is defined in Section 3.3. $\mathcal{V}_l(\mathcal{P}_i)$ is derived from $\mathcal{V}_l(\mathcal{P}_{i-1})$ by first computing the boundary of the Voronoi-like region of α_i and then inserting this region into $\mathcal{V}_l(\mathcal{P}_{i-1})$ to obtain $\mathcal{V}_l(\mathcal{P}_i)$ Theorem 3. By the definition of the split relation, which induces the insertion order o, no auxiliary arc can be created by this arc insertion process. Thus, $\mathcal{P}_m = \mathcal{C}$ and $\mathcal{V}_l(\mathcal{P}_m) = \mathcal{V}_l(\mathcal{C})$, as the Voronoi-like diagram of a boundary curve is unique (Theorem 4). Since the order o is fixed, the construction time is $O(|\mathcal{S}'|^2)$. □

Given a boundary curve $\mathcal{P}$ and arcs $\alpha, \beta \in \mathcal{P}$, let $\mathcal{P}[\alpha, \beta]$ denote the portion of $\mathcal{P}$ from α to β in a counterclockwise traversal, including α and β.

Lemma 20. *Let $\mathcal{P}$ be a boundary curve of $\mathcal{S}'$ and let $\alpha, \beta \in \mathcal{P}$ be any two* consecutive *original arcs. No s-related bisector in $\mathcal{J}_{s,\mathcal{S}'}$ can split (intersect twice) an arc in $\mathcal{P}[\alpha, \beta]$.*

Proof. Suppose for the sake of contradiction that bisector $J(s, p) \in \mathcal{J}_{s,\mathcal{S}'}$ intersects an arc $\gamma \in \mathcal{P}[\alpha, \beta]$ twice, see Figure 4.7. Then $J(s, p)$ could not appear an the

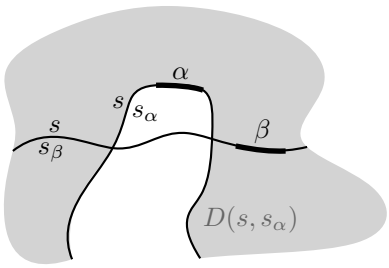

Figure 4.6. An arc α splits arc β. In the induced partial order on $\mathcal{S}$, $\alpha < \beta$.

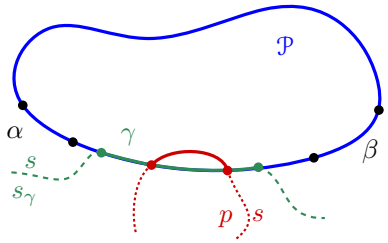

Figure 4.7. A bisector $J(s,p) \in \mathcal{J}_{s,S'}$ splitting an arc γ contradicts the existence of an original arc of site p.

s-envelope $\text{env}(\mathcal{J}_{s,S}) = \mathcal{S}$ as the s-related bisectors $J(s,p), J(s,s_\gamma)$ can intersect at most twice. This contradicts the fact that there must exist an original arc of site p on $\mathcal{P}$. □

Corollary 2. *Any boundary curve $\mathcal{P}$ of $\mathcal{S}'$ is enclosed by the plain boundary curve $\mathcal{C}$. That is, $D_\mathcal{P} \subseteq D_\mathcal{C}$.*

Proof. Let $\alpha, \beta \in \mathcal{P}$ be two consecutive original arcs such that α neighbors an auxiliary arc $\alpha_1 \in \mathcal{P}$. Since $\mathcal{P}$ is an s-monotone path, arc α_1 lies at least partially in $D_\mathcal{C}$. Suppose an auxiliary arc $\gamma \in \mathcal{P}$ exits $D_\mathcal{C}$ at arc α, as shown in Figure 4.9. Then the bisector $J(s,s_\gamma)$ must intersect α a second time, otherwise $\alpha \subset D(s_\gamma, s)$, which contradicts the fact that α is original. However, two intersections with α contradict Lemma 20. □

Corollary 2 implies that any auxiliary arc of any boundary curve $\mathcal{P}$ lies in the interior of $D_\mathcal{C}$. We use this fact to derive an algorithm to construct $\mathcal{V}_l(\mathcal{P})$, given $\mathcal{V}_l(\mathcal{C})$. Thus, we prove the following existence theorem.

Theorem 5. *The Voronoi-like diagram of any boundary curve $\mathcal{P}$ on $\mathcal{S}' \subseteq \mathcal{S}$ always exists.*

Proof. The proof is by construction. Given a boundary curve $\mathcal{P}$, we first construct $\mathcal{V}_l(\mathcal{C})$ using Lemma 19. Given $\mathcal{V}_l(\mathcal{C})$, we incrementally construct $\mathcal{V}_l(\mathcal{P})$ by inserting the remaining arcs of $\mathcal{P}$ (that are not in $\mathcal{C}$), using the arc insertion operation $\oplus$ of Section 3.3. These remaining arcs are all auxiliary, however, by Corollary 2, they all lie in $D_\mathcal{C}$, thus, it is possible to use the insertion operation $\oplus$. Theorem 3 requires that each arc under insertion lies in the interior of the current boundary curve. We show how to maintain this invariant during a sequential insertion process in the following. Then, Lemma 20 ensures that no arc split is possible by the insertion order, and thus, no new auxiliary arc can be created.

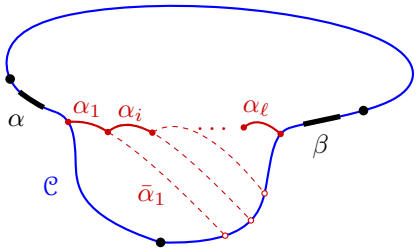

Figure 4.8. Inserting auxiliary arcs $\delta_1, \ldots, \delta_\ell$ between two consecutive original arcs α and β.

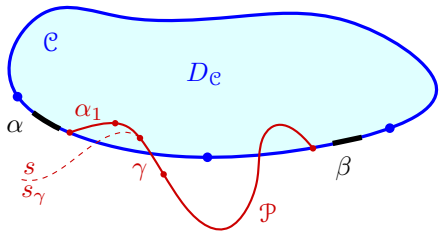

Figure 4.9. The assumption that $\mathcal{P}$ exits $D_{\mathcal{C}}$ yields a contradiction to Lemma 20.

Let $\delta_1, \ldots, \delta_\ell$ denote the auxiliary arcs in $\mathcal{P}$ between two consecutive original arcs α, β in counterclockwise order. Let $\mathcal{P}_0 = \mathcal{C}$. Denote by $\mathcal{P}_i$, $i = 1, \ldots, \ell$, the boundary curve after inserting arcs $\delta_1, \ldots, \delta_i$ to $\mathcal{C}$. We define $\bar{\delta}_i$ as the connected component of $J(s, s_{\delta_i}) \cap D_{\mathcal{P}_{i-1}}$ that contains δ_i. In the following we prove that $\bar{\delta}_i$ on $\mathcal{P}_i$ is well-defined, that it contains δ_i and that it doesn't split any arc. Let $\tilde{\beta}_i$ denote the shrunk arc β on $\mathcal{P}_i$. Let $\bar{\delta}_0 = \alpha$, $\delta_0 = \alpha$ and $\beta_0 = \beta$. Note that between two consecutive boundary arcs on $\mathcal{P}_i$ there can be a Γ-arc.

Following the order $i = 1, \ldots, \ell$, when we insert arc $\bar{\delta}_i$ its first endpoint lies on $\bar{\delta}_{i-1}$, which is then shrunk to δ_{i-1} on $\mathcal{P}_i$ (or its first endpoint lies on a Γ-arc after $\bar{\delta}_{i-1}$). We prove that the second endpoint of $\bar{\delta}_i$ lies on $\tilde{\beta}_{i-1}$ (or on a Γ-arc before $\tilde{\beta}_{i-1}$). If it did not lie on $\tilde{\beta}_{i-1}$ (or on a Γ-arc next to $\tilde{\beta}_{i-1}$), then $\bar{\delta}_i$ would either follow $\tilde{\beta}_{i-1}$ (and thus delete it) contradicting that $\tilde{\beta}_{i-1}$ is original; or $\bar{\delta}_i$ would have a second intersection with $\bar{\delta}_i$ (split it) contradicting Lemma 20. This implies that $\bar{\delta}_i$ contains δ_i. Every time we insert an arc $\bar{\delta}_i$ we also insert its region $R(\bar{\delta}_i)$ and compute the Voronoi-like diagram $\mathcal{V}_l(\mathcal{P}_i)$ which is possible by Theorem 3.

We insert all remaining arcs between each consecutive pair of original arcs and in the end we obtain the Voronoi-like diagram $\mathcal{V}_l(\mathcal{P})$, since we inserted all arcs of $\mathcal{P}$ and did not create any additional arc. □

The plain boundary curve $\mathcal{C}$ of $\mathcal{S}'$ is interesting in its own right. Its Voronoi-like regions are supersets of the corresponding Voronoi-like regions of any boundary curve $\mathcal{P}$ on $\mathcal{S}'$, including the s-envelope $\mathcal{E}$ of $\mathcal{S}'$. This is a direct corollary of the last proof, stated in the following lemma.

Lemma 21. *For any original arc $\alpha \in \mathcal{P}$, for any boundary curve $\mathcal{P}$ of $\mathcal{S}'$, $R(\alpha, \mathcal{P}) \subseteq R(\bar{\alpha}, \mathcal{C})$, where $\bar{\alpha} \supseteq \alpha$ is the corresponding arc in the plain boundary curve $\mathcal{C}$.*

Proof. From the proof of Theorem 5 it follows that we can construct $\mathcal{V}_l(\mathcal{P})$ from $\mathcal{V}_l(\mathcal{C})$ by inserting the auxiliary arcs of $\mathcal{P}$ in a certain order. By Theorem 3, when

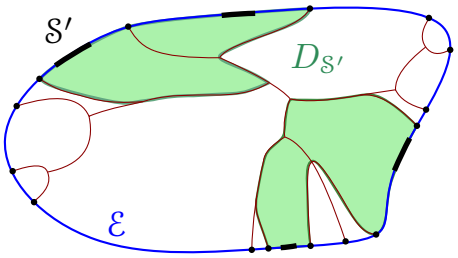

Figure 4.10. The domain $D_{\mathcal{S}'} = D_{\mathcal{E}} \setminus \bigcup_{\text{auxiliary } \alpha \in \mathcal{E}} \overline{R(\alpha, \mathcal{E})}$ is shown shaded (green).

inserting an arc β to a Voronoi-like diagram, its initial regions can only shrink (or remain the same) after inserting β. Thus, while inserting in $\mathcal{V}_l(\mathcal{C})$ the auxiliary arcs of $\mathcal{P}$, a region $R(\bar{\alpha}, \mathcal{C})$ can only become smaller, ending up in $R(\alpha, \mathcal{P}) \subseteq R(\bar{\alpha}, \mathcal{C})$. Since the diagram $\mathcal{V}_l(\mathcal{P})$ is unique the claim follows. □

Consider the real Voronoi diagram $\mathcal{V}(\mathcal{S}') = \mathcal{V}(S' \setminus \{s\}) \cap D_s$, where $D_s = \mathrm{VR}(s, S') \cap D_\Gamma$. It coincides with $\mathcal{V}_l(\mathcal{E})$, where $\mathcal{E}$ is the s-envelope of $\mathcal{S}'$ (more precisely the s-envelope of $\mathcal{J}_{s,\mathcal{S}'} \cup \Gamma$). In contrast to $\mathcal{V}_l(\mathcal{C})$, the regions of $\mathcal{V}_l(\mathcal{E})$, are subsets of Voronoi-like regions in $\mathcal{V}_l(\mathcal{P})$, for any boundary curve $\mathcal{P}$ (Lemma 4).

In contrast, the regions of $\mathcal{V}_l(\mathcal{E})$, which is a real Voronoi diagram, are subsets of Voronoi-like regions in $\mathcal{V}_l(\mathcal{P})$, for any boundary curve $\mathcal{P}$ (Lemma 4). From this subset/superset relation of Voronoi-like regions we can derive a relation between the Voronoi-like diagrams of different boundary curves on $\mathcal{S}'$, which is given in the following lemma. Recall that any boundary curve $\mathcal{P}$ on $\mathcal{S}'$ lies between the plain curve $\mathcal{C}$ and the envelope $\mathcal{E}$ and they all share the same core arcs in $\mathcal{S}'$. Recall also that $D_{\mathcal{E}} = \mathcal{V}(s, S') \cap D_\Gamma$.

The following lemma states that the Voronoi-like diagram always coincides with the real Voronoi diagram within the domain $D_{\mathcal{S}'} = D_{\mathcal{E}} \setminus \bigcup_{\text{auxiliary } \alpha \in \mathcal{E}} \overline{R(\alpha, \mathcal{E})}$, see Figure 4.10.

Lemma 22. *Within domain $D_{\mathcal{S}'}$, $\mathcal{V}_l(\mathcal{P}) \cap D_{\mathcal{S}'} = \mathcal{V}(S') \cap D_{\mathcal{S}'}$, for any boundary curve $\mathcal{P}$ on $\mathcal{S}' \subseteq \mathcal{S}$.*

Proof. By the definitions, $\mathcal{V}(S') \cap D_{\mathcal{S}'} = \mathcal{V}_l(\mathcal{E}) \cap D_{\mathcal{S}'}$. So, it is enough to show that $\mathcal{V}_l(\mathcal{P}) \cap D_{\mathcal{S}'} = \mathcal{V}_l(\mathcal{E}) \cap D_{\mathcal{S}'}$.

Let $\alpha^* \in \mathcal{S}'$ and let $\alpha \supseteq \alpha^*$, $\bar{\alpha} \supseteq \alpha^*$ denote the corresponding original arcs in $\mathcal{P}$ and $\mathcal{E}$ respectively. Then $R(\bar{\alpha}, \mathcal{E}) \subseteq R(\alpha, \mathcal{P})$, by Lemma 4. In addition, $D_{\mathcal{S}'} = \left(\overline{\bigcup_{\alpha \in \mathcal{S}'} R(\alpha, \mathcal{E})}\right)^\circ$, i.e., $D_{\mathcal{S}'}$ contains all Voronoi-like regions of the original arcs in $\mathcal{V}_l(\mathcal{E})$. Since each of these regions has a corresponding region in $\mathcal{V}_l(\mathcal{P})$

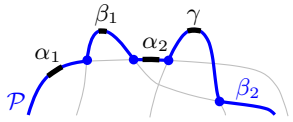

Figure 4.11. A boundary curve that does not form a Davenport-Schinzel sequence of order 2 w.r.t. site occurrences.

that is a superset of it, the two diagrams $\mathcal{V}_l(\mathcal{P})$ and $\mathcal{V}_l(\mathcal{E})$ coincide within $D_{\mathcal{S}'}$. □

The proof of Theorem 5 gives an algorithm to construct $\mathcal{V}_l(\mathcal{P})$ for any boundary curve $\mathcal{P}$ of $\mathcal{S}'$. However, this algorithm may require $\Theta(|\mathcal{P}|^2)$ time and this also applies to the plain boundary curve $\mathcal{C}$ and the envelope $\mathcal{E}$. Note that the latter yields a real Voronoi diagram in time $\Theta(|\mathcal{E}|^2)$.

In this chapter we have established that for any boundary curve $\mathcal{P}$ the Voronoi-like diagram $\mathcal{V}_l(\mathcal{P})$ exists and is unique and thus, it is well-defined. An open problem is to establish the complexity of an arbitrary boundary curve $\mathcal{P}$ for $\mathcal{S}' \subseteq \mathcal{S}$ and the thus, the complexity of $\mathcal{V}_l(\mathcal{P})$. Recall that the combinatorial complexity $|\mathcal{P}|$ of $\mathcal{P}$ is the number of boundary arcs of $\mathcal{P}$.

Remark 1. *We remark that a boundary curve that is not an s-envelope is not necessarily a Davenport-Schinzel sequence of order 2, with respect to site occurrences in $S \setminus \{s\}$. See Figure 4.11 for an example of a boundary curve $\mathcal{P}$, where the arcs $\alpha_1, \beta_1, \alpha_2, \beta_2$ appear in this order as a subsequence along $\mathcal{P}$ and such that the sites repeat as follows: $s_{\alpha_1} = s_{\alpha_2}$ and $s_{\beta_1} = s_{\beta_2}$.*

Question 1. *What is the combinatorial complexity of an arbitrary boundary curve $\mathcal{P}$ for $\mathcal{S}'$. Can it be bounded by $O(|\mathcal{S}'|)$?*

Note that all boundary curves on a subset of arcs $\mathcal{S}'$ that we compute incrementally by *arc insertion* in the following chapter can be easily bounded by $O(|\mathcal{S}'|)$, see Lemma 23.

4.3 Conclusion

In this Chapter we have seen that the Voronoi-like diagram introduced in Chapter 3 is well-defined. For a given boundary curve $\mathcal{P}$, the diagram $\mathcal{V}_l(\mathcal{P})$ always *exists* and there are no two different such diagrams for $\mathcal{P}$, thus, $\mathcal{V}_l(\mathcal{P})$ is also *unique*. These two properties are not directly obvious from the definition and

it needed a long research process to end up with the correct conjectures, where especially the existence property was a surprising result.

Combining *existence* with *uniqueness* this chapter establishes the Voronoi-like structure as a well-defined geometric partitioning tool that can handle disconnected Voronoi regions.

Chapter 5

A randomized incremental algorithm

In this chapter we present an easy randomized incremental algorithm for solving the fundamental problem of site deletion, i.e., problem (1) of the three the tree-like problems listed in Section 1.3.

In Section 5.1 we give a simple randomized algorithm to update a Voronoi diagram after deletion of one site in expected time linear in the number of the neighbors of the deleted site. Given $\mathcal{V}(S)$ and a site $s \in S$, we compute $\mathcal{V}(S \setminus \{s\})$ within $\mathrm{VR}(s,S)$ in expected time linear on the complexity of $\partial\mathrm{VR}(s,S)$ using Voronoi-like diagrams as intermediate structures. The time analysis is given in Section 5.2.

This chapter is based on the publication:

Kolja Junginger and Evanthia Papadopoulou. Deletion in Abstract Voronoi Diagrams in Expected Linear Time. In *34th International Symposium on Computational Geometry* (SoCG 2018), Dagstuhl, Germany.

We also published a corresponding longer version with all proofs on arXiv:

Kolja Junginger and Evanthia Papadopoulou. *Deletion in abstract Voronoi diagrams in expected linear time*, CoRR abs/1803.05372. URL: `http://arxiv.org/abs/1803.05372`

5.1 Updating an abstract Voronoi diagram after deletion of one site

Consider a Voronoi region $\mathrm{VR}(s,S)$ and let $\mathcal{S}$ denote the (ordered) set of Voronoi edges on its boundary ($\mathcal{S} = \partial\mathrm{VR}(s,S) \cap D_\Gamma$).

Consider a random permutation $o = (\alpha_1, \ldots, \alpha_h)$ of the set of core arcs $\mathcal{S}$, where $|\mathcal{S}| = h$. For $1 \leq i \leq h$, define set $\mathcal{S}_i = \{\alpha_1, \ldots, \alpha_i\} \subseteq \mathcal{S}$ to be the subset of

the first i arcs in o, and permutation $o_i = (\alpha_1, \ldots, \alpha_i)$. Let $\mathcal{P}_i$ denote the boundary curve of the set $\mathcal{S}_i$, which is derived by the arc insertion operation $\oplus$ in the order of the permutation o_i; let D_i denote the corresponding domain enclosed by $\mathcal{P}_i$.

Our randomized algorithm is inspired by the randomized, two-phase, approach of Chew Chew [1990] for the Voronoi diagram of points in convex position; however, the sites are boundary arcs, forming boundary curves, and the algorithm constructs Voronoi-like diagrams within a series of shrinking domains $D_i \supseteq D_{i+1}$. The domain D_1 is $D(s, s_{\alpha_1}) \cap D_\Gamma$, and D_h coincides with the Voronoi region $\mathrm{VR}(s, S)$ within Γ, i.e, $D_h = \mathrm{VR}(s, S) \cap D_\Gamma$. The boundary curves are obtained by the insertion operation $\oplus$, one at each step, starting with $\mathcal{P}_1 = J(s, s_{\alpha_1}) \cap D_\Gamma$ and ending with $\mathcal{P}_h = \partial\mathrm{VR}(s, S) \cap D_\Gamma$.

The approach works in two phases. In phase 1, the arcs in $\mathcal{S}$ get deleted one by one, in reverse order of o, while recording the neighbors of each arc at the time of its deletion. Let $\mathcal{P}_1 = J(s, s_{\alpha_1}) \cap D_\Gamma$, $R(\alpha_1, \mathcal{P}_1) = D_1$, and $\mathcal{V}_l(\mathcal{P}_1) = \emptyset$.

In phase 2, we start with $\mathcal{V}_l(\mathcal{P}_1)$ and incrementally compute $\mathcal{V}_l(\mathcal{P}_i)$, $i = 2, \ldots, h$, by inserting arc α_i, where $\mathcal{P}_i = \mathcal{P}_{i-1} \oplus \alpha_i$, and $\mathcal{V}_l(\mathcal{P}_i) = \mathcal{V}_l(\mathcal{P}_{i-1}) \oplus \alpha_i$. When inserting an arc α_i, we use the information of its recorded neighbors from phase 1 to determine a starting point. At the end we obtain $\mathcal{V}_l(\mathcal{P}_h)$, where $\mathcal{P}_h$ is the unique boundary curve corresponding to the envelope of $\mathcal{S}$, i.e., $\mathcal{P}_h = \mathrm{env}(\mathcal{S}) = \partial\mathrm{VR}(s, S) \cap D_\Gamma$.

By Corollary 1, we have already established that $\mathcal{V}_l(\mathrm{env}(\mathcal{S}))$ is the real Voronoi diagram $\mathcal{V}(\mathcal{S})$. Given the correctness of the insertion operation $\oplus$ that was established in Section 3.3, the algorithm correctly computes $\mathcal{V}_l(\mathcal{P}_h)$, where $\mathcal{V}_l(\mathcal{P}_h) = \mathcal{V}(\mathcal{S}) = \mathcal{V}(S \setminus \{s\}) \cap \mathrm{VR}(s, S)$.

Lemma 23. *$\mathcal{P}_i$ contains at most $2i - 1$ arcs. Its auxiliary arcs are less than the original. The complexity of $\mathcal{V}_l(\mathcal{P}_i)$ is $O(i)$.*

Proof. By definition, $|\mathcal{P}_1| = 1$. At each step of phase 2, exactly one original arc is inserted, and at most one additional auxiliary arc is created by a split in case (c) of Observation 1, except from $i = 1$ and $i = h$. Thus, the total number of auxiliary arcs is at most $i - 1$. The number of original arcs is at most i, since an original arc may be merged with its neighbor in case (f) of Observation 1. Since in this case no auxiliary arc is created, the claim follows. Since the complexity of $\mathcal{V}_l(\mathcal{P}_i)$ is $O(|\mathcal{P}_i|)$ and by the above we have $|\mathcal{P}_i| \leq 2i - 1$, the complexity of $\mathcal{V}_l(\mathcal{P}_i)$ is $O(i)$. □

By Lemma 23 the total number of auxiliary arcs that may be created up to step i of the algorithm is less than i, which immediately implies for the term $d_2(\alpha_i, \mathcal{P}_j)$ in the time complexity of step i, as expressed by Lemma 10, that

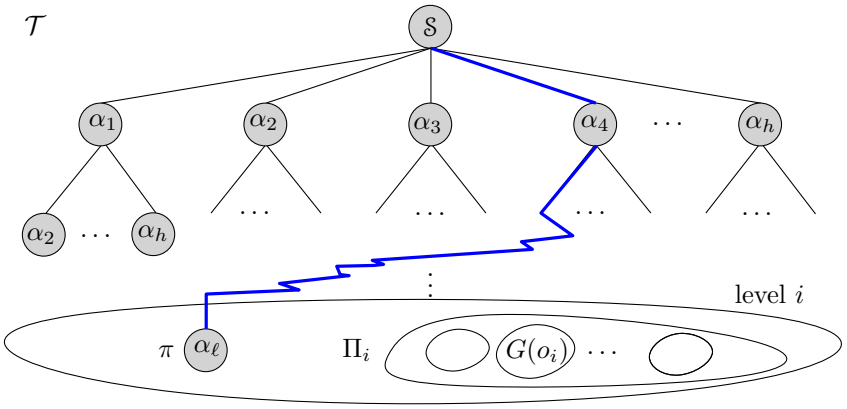

Figure 5.1. The decision tree $\mathcal{T}$ of all possible random choices: Each node at level i corresponds to a unique choice of arcs (e.g. the blue path). There are $N = h(h-1)\cdot\cdots\cdot(h-i+1)$ nodes at level i of $\mathcal{T}$, which are partitioned into $\binom{h}{i}$ many blocks (e.g. Π_i) and each block into $(i-1)!$ groups (e.g. $G(o_i)$).

$\sum_{j=1}^{i} d_2(\alpha_j, \mathcal{P}_j) < i$. Independent of this argument, in the next section we bound the expected values of all terms in Lemma 10.

Remark 2. *Note that even though any boundary curve $\mathcal{P}_i$ that is created by insertion of i arcs has complexity $O(i)$ by Lemma 23, it does not necessarily form a Davenport-Schinzel sequence of order 2, with respect to site occurrences. For an example, see the boundary curve $\mathcal{P}$ of Figure 4.11, which can be created by inserting the core arcs in the order $o = (\beta_1, \gamma, \alpha_1, \alpha_2)$ and $\alpha_1, \beta_1, \alpha_2, \beta_2$ appear in this order as a subsequence along $\mathcal{P}$, where $s_{\alpha_1} = s_{\alpha_2}$ and $s_{\beta_1} = s_{\beta_2}$.*

5.2 Time analysis of the randomized incremental algorithm

Consider the *decision tree* $\mathcal{T}$ of all possible random choices that can be made by our incremental algorithm on the input set of core arcs $\mathcal{S}$, $h = |\mathcal{S}|$, see Figure 5.1. $\mathcal{T}$ has $h!$ leaves each corresponding to a unique permutation of the arcs in $\mathcal{S}$. At level-i, there are $h!/(h-i)!$ nodes, and each node corresponds to a unique permutation of i core arcs. Out of all nodes at level-i, $i!$ nodes are associated with the same set of i core arcs $\mathcal{S}_i$. We call this set of nodes the *block of* $\mathcal{S}_i$ and we have $\binom{h}{i}$ distinct such blocks at level-i. Although all nodes within one block are associated with the same set of core arcs, their corresponding boundary curves can vary considerably depending on the permutation order. Because of

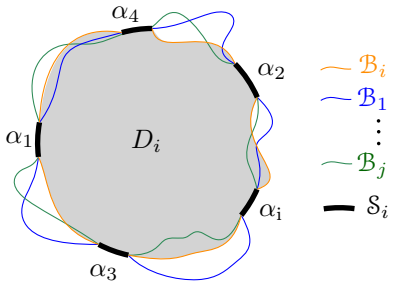

Figure 5.2. Schematic differences between the boundary curves $\mathcal{B}_1, \ldots, \mathcal{B}_i$. The domain D_i is shown shaded.

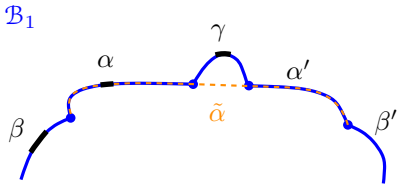

Figure 5.3. Illustration for Definition 11: The core arc $\alpha \in \mathcal{S}_i$ is the source of $\alpha' \in \text{in}_1$. The expanded arc $\tilde{\alpha} \supseteq \alpha'$ was created by inserting α during the construction of $\mathcal{B}_1$, where $o_1 = (\beta, \alpha, \gamma)$ and $o_i = (\gamma, \beta, \alpha)$. The corresponding $\mathcal{B}_i$ is shown in Figure 5.4.

this order-dependent variation, we cannot easily apply *backwards analysis* as in the original randomized incremental construction of Chew [1990]. Instead, we establish the expected linear-time complexity of our algorithm by analyzing the blocks of nodes at level-i of the decision tree[1].

We use the following strategy. We partition the set of nodes in each block of nodes at level-i of $\mathcal{T}$ into disjoint groups of i nodes each, and analyze the total time complexity of the algorithm on the i nodes (permutations) of each group. In particular, we show that step-i of our algorithm requires $O(i)$ time in total on each entire group of our partition. Thus, on average, the algorithm spends $O(1)$ time on each node of $\mathcal{T}$. Since all permutations are equally likely, we obtain the expected linear ($O(h)$) time complexity of our algorithm.

Let $o_i = (\alpha_1, \alpha_2, \ldots, \alpha_i)$ be an arbitrary permutation of $\mathcal{S}_i$. From o_i we define a group $G = G(o_i)$ of i permutations: for each $1 \leq j < i$, remove α_j from its position in o_i and append it to the end of o_i.

$$o_i = (\alpha_1, \alpha_2, \ldots, \alpha_{j-1}, \boxed{\alpha_j}, \alpha_{j+1} \ldots, \alpha_{i-1}, \alpha_i) \tag{5.1}$$

$$o_j = (\alpha_1, \alpha_2, \ldots, \alpha_{j-1}, \quad \alpha_{j+1}, \ldots, \alpha_{i-1}, \alpha_i, \boxed{\alpha_j}), \tag{5.2}$$

Let $\mathcal{B}_j$, $1 \leq j \leq i$, denote the boundary curves derived by inserting the core arcs in $\mathcal{S}_i$ in order o_j, see Figure 5.2. $\mathcal{B}_i$ is the base boundary curve derived from o_i, and its domain is denoted D_i. In the following we establish the relation between these boundary curves so that we can prove our objective regarding the

[1]The analysis outline in our paper Junginger and Papadopoulou [2018a] follows the backwards analysis framework of Chew [1990], however, its direct applicability is questionable as our boundary curves are order-dependent. We revisit the analysis in this section.

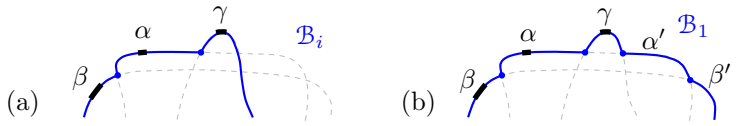

Figure 5.4. Left: Boundary curve $\mathcal{B}_i$, where $o_i = (\gamma, \beta, \alpha)$. Right: $\mathcal{B}_1$, where $o_i = (\beta, \alpha, \gamma)$, containing arcs $\alpha', \beta' \in \text{in}_1$, because γ was inserted last.

time complexity of the ith step of the algorithm on all of them (Lemma 28). We first introduce some terminology.

Definition 11. *Let α' be an auxiliary arc in $\mathcal{B}_j$ and let $\alpha \in \mathcal{S}_i$ be a core arc of the same site. We say that α'* is an auxiliary arc of the core arc *α if there is an expanded arc $\tilde{\alpha} \supseteq \alpha$ and $\tilde{\alpha} \supseteq \alpha'$, such that $\tilde{\alpha}$ was created by inserting α during the construction of $\mathcal{B}_j$ (see Figure 5.3). The core arc α is denoted as $\text{source}_j(\alpha')$ and it is called the* source of *α'.*

If α' appears counterclockwise (resp. clockwise) from its source α along their common s-bisector then α' is called a ccw *(resp.* cw*) auxiliary arc.*

The boundary curves $\mathcal{B}_j$ may get in and out of the domain D_i for $j < i$ (see Figure 5.2). To identify their differences from $\mathcal{B}_i$, let $\text{in}_j = \mathcal{B}_j \cap D_i$ and $\text{out}_j = \mathcal{B}_j \setminus \overline{D_i}$ denote the portion of $\mathcal{B}_j$ inside and outside of D_i, respectively. We partition the auxiliary arcs in in_j into in_j^+ and in_j^-, where in_j^+ (resp. in_j^-) includes the ccw (resp. cw) auxiliary arcs of in_j, see Figure 5.4. In the following we only consider in_j^+ as in_j^- is entirely symmetric.

Observation 2. *The boundary curve $\mathcal{B}_j$, $j \neq i$, contains no auxiliary arcs of the core arc α_j (since α_j appears last in o_j). Furthermore, these are the only auxiliary arcs of $\mathcal{B}_i$ that are missing from $\mathcal{B}_j$ (since the insertion order of all other core arcs is identical). Thus, any auxiliary arc $\alpha' \in \text{out}_j$ must be entirely* below *an auxiliary arc $\alpha'_j \in \mathcal{B}_i$ of the core arc α_j (i.e., $\alpha' \in \mathbb{R}^2 \setminus \overline{D_i}$), see Figure 5.5.*

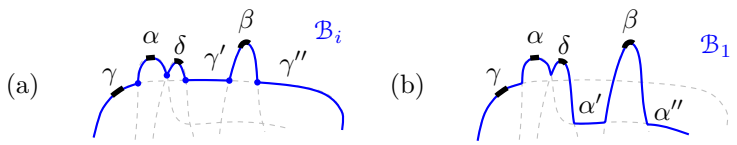

Figure 5.5. Left: Boundary curve $\mathcal{B}_i$, where $o_i = (\gamma, \alpha, \beta, \delta)$. Right: $\mathcal{B}_1$ containing arcs in out_1, where $o_1 = (\alpha, \beta, \delta, \gamma)$.

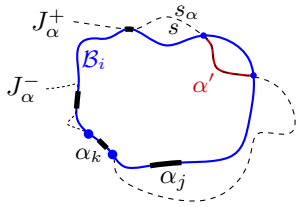

Figure 5.6. The arc α_k is the source of the auxiliary arc $\alpha' \in \text{in}_j$: $(\alpha_k, \alpha_j, \alpha')$ appear in ccw order on $\mathcal{B}_j$.

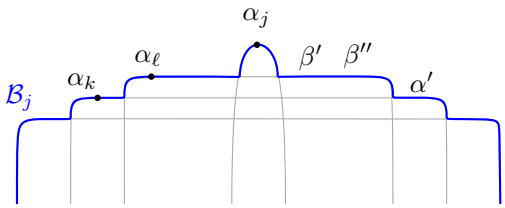

Figure 5.7. If $\alpha', \beta' \in \text{in}^+_j$, then $j < k < \ell$ and $(\alpha_k, \alpha_\ell, \alpha_j, \beta', \alpha')$ appear in ccw order on $\mathcal{B}_j$.

Observation 3. *Consider a ccw arc $\alpha' \in in^+_j$ and let α_k be its source core arc ($\alpha_k = source_j(\alpha')$). Then α_k must follow α_j in o_i, i.e., $k > j$. Further, $(\alpha_k, \alpha_j, \alpha')$ must appear ccw in $\mathcal{B}_j$. Symmetrically for a cw arc in in^-_j.*

Observation 4. *Figure 5.7 indicates the structure of in^+_j. Suppose that core arcs α_k and α_ℓ, $k < \ell$, have auxiliary arcs α' and β' in in^+_j respectively. Then $j < k < \ell$ and $(\alpha_k, \alpha_\ell, \alpha_j, \beta', \alpha')$ appear in ccw order on $\mathcal{B}_j$. Further, all auxiliary arcs of α_ℓ must appear before the auxiliary arcs of α_k as we move counterclockwise from α_j.*

Since many auxiliary arcs of in^+_j can have the same source we define

$$N_j = \{\text{source}_j(\alpha') \in \mathcal{S}_i \mid \alpha' \in \text{in}^+_j\}.$$

All arcs in N_j are of pairwise different sites. It is possible that $\text{in}^+_j \cap \text{in}^+_k \neq \emptyset$ for $k \neq j$. Moreover it is possible that $|\text{in}^+_j| = \Omega(i)$ even if $|N_j| = 1$. However, we have the following disjointness property.

Lemma 24. *$N_j \cap N_k = \emptyset$ for all $k \neq j$. Thus, $\sum_{j=1}^{i} |N_j| = O(i)$.*

Proof. Suppose $\alpha_\ell \in N_j \cap N_k$ and $j < k$, then $\alpha_\ell = \text{source}_j(\alpha')$, where $\alpha' \in \text{in}^+_j$ and $\alpha_\ell = \text{source}_k(\alpha'')$, where $\alpha'' \in \text{in}^+_k$. (The arcs α' and α'' may or may not be equal.) By Observation 3, $j < k < \ell$, and $(\alpha_\ell, \alpha_k, \alpha_j)$ must appear in ccw order on $\mathcal{B}_i$. However, since $k < \ell$, the arc α_k must be inserted before α_ℓ in $\mathcal{B}_j$, and thus, α' cannot exist in $\mathcal{B}_j$, see Figure 5.8. Further, since $j < \ell$, the arc α_j is inserted before α_ℓ in $\mathcal{B}_k$, and thus, α'' cannot exist on $\mathcal{B}_k$, see Figure 5.9. In both cases we derive a contradiction. □

We next establish that all parameters of the time complexity analysis for step i (Lemma 10) sum up to $O(i)$ on all boundary curves $\mathcal{B}_j, j \leq i$.

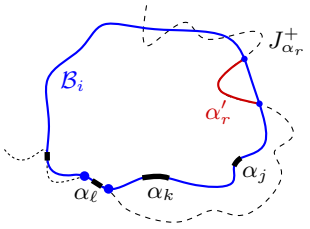

Figure 5.8. Illustration for the proof of Lemma 24. The case $(\alpha_\ell, \alpha_k, \alpha_j)$ appear ccw.

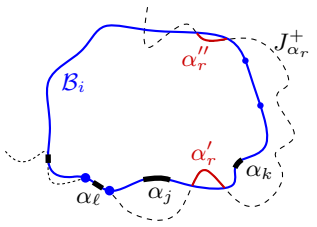

Figure 5.9. Illustration for the proof of Lemma 24. The case $(\alpha_\ell, \alpha_j, \alpha_k)$ appear ccw.

Lemma 25.

$$\sum_{j=1}^{i} d_1(\alpha_j, \mathcal{B}_j) + d_2(\alpha_j, \mathcal{B}_j) + \tilde{d}(\alpha_j, \mathcal{B}_j) = O(i).$$

Proof. Let α and γ denote the original arcs preceding and following α_j respectively in $\mathcal{B}_i$ (equiv. in $\mathcal{B}_j$). Let $d(\alpha_j, \mathcal{B}_k) = d_1(\alpha_j, \mathcal{B}_k) + d_2(\alpha_j, \mathcal{B}_k)$ denote the auxiliary arcs on the boundary curve $\mathcal{B}_k$, $k = i, j$.

We first observe that $d(\alpha_j, \mathcal{B}_j)$ cannot contain any portion of out_j as no auxiliary arc of α_j may appear in $\mathcal{B}_i$ from α to γ (since there is no other core arc from α to γ, except α_j, and a different core arc is necessary to produce an auxiliary arc). Thus, we only need to consider the auxiliary arcs of in_j. Next, we observe that no two auxiliary arcs in $d(\alpha_j, \mathcal{B}_j)$ can have the same source in N_j for the same reason, i.e., there is no core arc from α to γ except α_j. Thus, we can bound $|d(\alpha_j, \mathcal{B}_j)| \leq |d(\alpha_j, \mathcal{B}_i)| + |N_j|$. Then, by Lemma 24, $\sum_{j=1}^{i} |d(\alpha_j, \mathcal{B}_j)| \leq |\mathcal{B}_i| + O(i) = O(i)$.

If $\tilde{d}(\alpha_j, \mathcal{B}_j) > 0$ we have case (d) of Observation 1. In this case both endpoints of α_j are incident to Γ in both $\mathcal{B}_j$ and $\mathcal{B}_i$. Then, by Observations 2 and 4, $\text{in}_j = \emptyset$ and $\text{out}_j = \emptyset$, implying that $\mathcal{B}_j = \mathcal{B}_i$; thus, $\tilde{d}(\alpha_j, \mathcal{B}_j) = \tilde{d}(\alpha_j, \mathcal{B}_i)$. Then, $\sum_{j=1}^{i} |\tilde{d}(\alpha_j, \mathcal{B}_j)| \leq |\tilde{\mathcal{B}}_i| = O(i)$. □

Lemma 26. $|R(\alpha_j, \mathcal{B}_j)| \leq 2|R(\alpha_j, \mathcal{B}_i)| + |N_j|$.

Proof. We compare $R(\alpha_j, \mathcal{B}_j)$ and $R(\alpha_j, \mathcal{B}_i)$ and bound differences in their adjacencies. First, we observe that no arc in out_j can have a region adjacent to $R(\alpha_j, \mathcal{B}_j)$. To see that, note that by Observation 2 any arc $\alpha' \in \text{out}_j$ lies below an auxiliary arc $\alpha'_j \in \mathcal{B}_i$ of the same site as α_j. If the regions of α' and α_j were adjacent, then also $R(\alpha'_j, \mathcal{B}_j \oplus \alpha'_j)$ would be adjacent to α_j, which is not possible in a Voronoi-like diagram, because α_j and α'_j are of the same site. Next, we observe

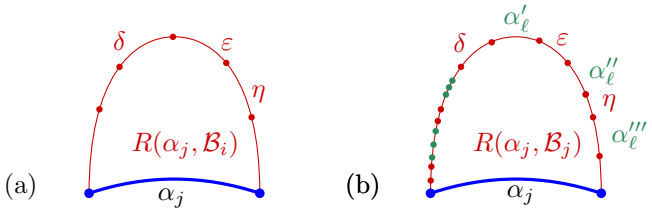

Figure 5.10. Illustration for Lemma 26.

that any arcs common to both $\mathcal{B}_j$ and $\mathcal{B}_i$, whose regions are adjacent to $R(\alpha_j, \mathcal{B}_i)$, they must also be adjacent to $R(\alpha_j, \mathcal{B}_j)$. In particular, if an arc $\varepsilon \in \mathcal{B}_j \cap \mathcal{B}_i$ is adjacent to $R(\alpha_j, \mathcal{B}_j)$ then it must also be adjacent to $R(\alpha_j, \mathcal{B}_i)$. This is clear, because otherwise, their common Voronoi edge e in $\mathcal{V}_l(\mathcal{B}_j)$ (or a portion of it) would be taken in $\mathcal{V}_l(\mathcal{B}_i)$ by an arc that is *missing* from $\mathcal{B}_j$ (Lemma 17), i.e., by an auxiliary arc of α_j, $\alpha'_j \in \mathcal{B}_i$. But if we insert this missing arc α'_j to $\mathcal{V}_l(B_j)$, the region $R(\alpha'_j, B_j \oplus \alpha'_j)$ will be adjacent to $R(\alpha_j, B_j \oplus \alpha'_j)$, deriving a contradiction.

Let $|R(\alpha_j, \mathcal{B}_j)|_x$ denote the number of additional adjacencies that $R(\alpha_j, \mathcal{B}_j)$ may have over $R(\alpha_j, \mathcal{B}_i)$, that is, $|R(\alpha_j, \mathcal{B}_j)| \leq |R(\alpha_j, \mathcal{B}_i)| + |R(\alpha_j, \mathcal{B}_j)|_x$. We show that $|R(\alpha_j, \mathcal{B}_j)|_x \leq |R(\alpha_j, \mathcal{B}_i)| + |N_j|$. Since auxiliary arcs of the same site can never have adjacent regions, it follows that between any two possible new adjacencies of $|R(\alpha_j, \mathcal{B}_j)|_x$ with arcs of in_j that have the same site or source, there must be an adjacency with at least one arc of $\mathcal{B}_i$ (see Figure 5.10(b)). Since by Observation 4 auxiliary arcs of one source in N_j must appear in a certain order along $\mathcal{B}_j$ and they cannot alternate, the bound follows. □

Lemma 27. *Consider case (c) of Observation 1 at the insertion of α_j in $\mathcal{B}_j$. Suppose that the insertion of α_j splits an existing arc ω into two pieces ω_1 and ω_2. Then at least one of these two arcs (say ω_1) must also exist in $\mathcal{B}_i$. Further, $|R(\omega_1, \mathcal{B}_j)| \leq 2|R(\omega_1, \mathcal{B}_i)| + |N_j|$.*

Proof. Suppose $\omega_1 \alpha_j \omega_2$ appear in $\mathcal{B}_j$ in ccw order and $\omega_2 \notin \mathcal{B}_i$. Then $\omega_2 \in \text{in}_j^+$, see Figure 5.11. We claim that ω_1 must belong in $\mathcal{B}_i$.

Let $\alpha_\ell = \text{source}_j(\omega_2)$. Then $\ell > j$ as $\omega_2 \in \text{in}_j^+$. Let $\tilde{\omega} \supset \alpha_\ell$ ($\tilde{\omega} \supset \omega$) denote the expanded arc created at the insertion time of α_ℓ, following o_j. Let $\hat{\omega} \supset \alpha_\ell$ denote the expanded arc created at the insertion time of α_ℓ, following o_i. Since $\ell > j$, it follows that $\hat{\omega}$ can extend ccw at most until α_j and $\hat{\omega} \subset \tilde{\omega}$. Since $\tilde{\omega}$ extends ccw past α_j, it follows that no other core arc α_ρ, with $\rho < \ell$ (except α_j), can prevent $\tilde{\omega}$ from extending. Thus, $\hat{\omega}$ extends ccw to α_j and $\hat{\omega} \supset \omega_2$. In addition, no α_ρ, with $\rho > \ell$, can delete ω_1 at its insertion following either o_j or o_i (since ω_1 exists in $\mathcal{B}_j$). Thus, ω_1 must exist in $\mathcal{B}_i$.

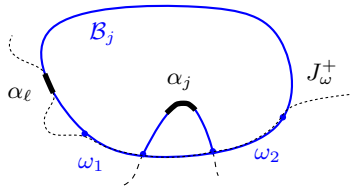

Figure 5.11. Illustration for the proof of Lemma 27. If $\omega_1 \notin \mathcal{B}_i$, then $\omega_2 \in \mathcal{B}_i$.

The only additional argument needed for the fact that no arc in out_j can have a region adjacent to $R(\omega_1, \mathcal{B}_j)$ is the observation that each arc in out_j lies below the s_ω-bisector, because arc α_j splits arc ω (case (c) of Observation 1). □

Let $T(i, o_j)$ denote the time required by the ith step of the algorithm following permutation o_j, i.e., the time required by the last arc insertion.

Lemma 28. *The time the algorithm requires at step-i on the entire group $G(o_i)$ is*

$$T(i, G) = \sum_{o_j \in G} T(i, o_j) = O(i).$$

Proof. Lemmas 26 and 27 establish that $|R(\alpha_j, \mathcal{B}_j)| + |R(\omega_j, \mathcal{B}_j)| \leq 2(|R(\alpha_j, \mathcal{B}_i)| + |R(\omega_j, \mathcal{B}_i)| + |N_j|)$. Here, if case (c) of Observation 1 is concerned, ω_j denotes one of the two arcs that is split and belongs to $\mathcal{B}_i$ (see Lemma 27); otherwise, $|R(\omega_j, \mathcal{B}_j)| = 0$. Since ω_j is always an immediate neighbor of α_j, we count it at most twice and thus, the total complexity $\sum_{j=1}^{i} |R(\omega_j, \mathcal{B}_i)|$ is $O(i)$. Together with Lemma 24 this directly implies that $\sum_{j=1}^{i} |R(\alpha_j, \mathcal{B}_j)| + r(\alpha_j, \mathcal{B}_j) = O(i)$. Lemma 25 establishes that $\sum_{j=1}^{i} d_1(\alpha_j, \mathcal{B}_j) + d_2(\alpha_j, \mathcal{B}_j) + \tilde{d}(\alpha_j, \mathcal{B}_j) = O(i)$. Then by Lemma 10 the claim is derived. □

Before stating the final result, we show that the partitioning of each block of $i!$ nodes (permutations) at level-i of $\mathcal{T}$ into $(i-1)!$ groups of i permutations each, is possible, if we follow the scheme we described in equation (5.2) for $G(o_i)$. Let Π_i denote such a block of all $i!$ permutations of the set $\mathcal{S}_i$. The references for the following lemma were provided by Stefan Felsner [2019].

Lemma 29 (Felsner [2019])**.** *The partitioning of Π_i into groups by the scheme we defined in equation* (5.2) *is possible, i.e.: For all $i \in \mathbb{N}$ and any block Π_i of permutations on $\mathcal{S}_i$ there exists a set $F \subset \Pi_i$ of $(i-1)!$ permutations such that $\Pi_i = \dot{\bigcup}_{o \in F} G(o)$.*

Proof. Following Levenshtein [1992] denote by $\lfloor \pi \rfloor$ the set of all permutations that are obtained from a permutation π by deleting one element. The following

property is clearly an equivalent condition for a set F to satisfy $\Pi_i = \dot{\bigcup}_{o\in F} G(o)$: For each $\pi, \sigma \in F$ the sets $\lfloor \pi \rfloor$ and $\lfloor \sigma \rfloor$ are disjoint. Levenshtein calls a family F of $(i-1)!$ permutations with this disjointness property a *code capable of correcting single deletions* and proves that these codes exist for all $i \in \mathbb{N}$ [Levenshtein, 1992, Theorem 3.1]. □

Since, by Lemma 29, it is possible to partition each block of $i!$ nodes at level-i of the decision tree into $(i-1)!$ groups following our scheme of equation (5.2), and since, by Lemma 28, each group requires total $O(i)$ time to perform step i, we obtain the following theorem. Recall that the decision tree at level-i has $\binom{h}{i}$ distinct blocks of $i!$ nodes each, and all nodes are equally likely.

Theorem 6. *Step-i of the randomized algorithm is performed in expected $O(1)$ time.*

Proof. For a subset $\mathcal{S}_i \subseteq \mathcal{S}, |\mathcal{S}_i| = i$, let $F(\mathcal{S}_i)$ denote a set of $(i-1)!$ permutations $(F(\mathcal{S}_i) \subset \Pi_i)$ satisfying $\Pi_i = \dot{\bigcup}_{o\in F(\mathcal{S}_i)} G(o)$ (which exists by Lemma 29). Let $T(i)$ denote the running time of the randomized algorithm to perform step i.

$$\mathbb{E}(T(i)) = \frac{1}{N} \sum_{\pi \text{ node at level i of } \mathcal{T}} T(i,\pi) \tag{5.3}$$

$$= \frac{1}{N} \sum_{\mathcal{S}_i \subseteq \mathcal{S}: |\mathcal{S}_i| = i} \sum_{\pi \in \Pi_i} T(i,\pi) \tag{5.4}$$

$$= \frac{1}{N} \sum_{\mathcal{S}_i \subseteq \mathcal{S}: |\mathcal{S}_i| = i} \sum_{o \in F(\mathcal{S}_i)} \sum_{o_j \in G(o)} T(i,o_j) \tag{5.5}$$

$$= \frac{1}{N} \sum_{\mathcal{S}_i \subseteq \mathcal{S}: |\mathcal{S}_i| = i} \sum_{o \in F(\mathcal{S}_i)} O(i) \tag{5.6}$$

$$= \frac{1}{N} \binom{h}{i} (i-1)! O(i) = O(1) \tag{5.7}$$

From line (5.4) to (5.5) we use Lemma 29, from (5.5) to (5.6) we use Lemma 28. □

Since there are h arcs to insert, the expected time of the algorithm is $O(h)$. We conclude with the following theorem.

Theorem 7. *Given an abstract Voronoi diagram $\mathcal{V}(S)$, $\mathcal{V}(S \setminus \{s\}) \cap VR(s,S)$ can be computed in expected $O(h)$ time, where h is the complexity of $\partial VR(s,S)$. Thus, $\mathcal{V}(S \setminus \{s\})$ can also be computed in expected time $O(h)$, given $\mathcal{V}(S)$.*

5.2.1 An alternative way to partition the set of all permutations

The rule of (5.2) to partition the block of arcs Π_i into $(i-1)!$ groups of i arcs is only one of many partitionings. The purpose of this section is to present an alternative way to partition the block Π_i into groups. With this way the analysis can be done in a similar way as in the last section. As a side effect the understanding of the differences of boundary curves is strengthened. This section can be seen as a short appendix to the time analysis of the last section and is not necessary for the results in this or other chapters.

Let $o_i = (\alpha_1, \alpha_2, \ldots, \alpha_i)$ be an arbitrary permutation of $\mathcal{S}_i$. This time, we define a group of i permutations from o_i as follows. For each $1 \leq j < i$ we *exchange* α_j with the last element α_i in o_i (instead of appending it as in (5.2)):

$$o_i = (\alpha_1, \alpha_2, \ldots, \alpha_{j-1}, \boxed{\alpha_j}, \alpha_{j+1} \ldots, \alpha_{i-1}, \boxed{\alpha_i}) \tag{5.8}$$

$$o_j = (\alpha_1, \alpha_2, \ldots, \alpha_{j-1}, \boxed{\alpha_i}, \alpha_{j+1}, \ldots, \alpha_{i-1}, \boxed{\alpha_j}) \tag{5.9}$$

Clearly, a partitioning of Π_i into groups as defined above is possible.

Now we analyze the group of permutations $G = G(o_i) = \{o_1, o_2, \ldots, o_{i-1}, o_i\}$, which is created by the above exchange rule. Let $\mathcal{B}_1, \mathcal{B}_2, \ldots, \mathcal{B}_{i-1}, \mathcal{B}_i$ be their corresponding boundary curves.

Any two consecutive core arcs α, δ along the cyclically ordered set $\mathcal{S}_i$ define one *gap* (for any boundary curve on $\mathcal{S}_i$), thus, i gaps in total, see Figure 5.2. Any boundary curve on $\mathcal{S}_i$ may fill out these gaps by auxiliary arcs. Note that all arcs in a gap are auxiliary arcs.

In contrast to the group in the last section, now there are four reasons for the differences between $\mathcal{B}_j$ and $\mathcal{B}_i$ manifesting themselves as in_j and out_j. From now on we set $\beta = \alpha_i$.

1. Auxiliary arcs of $\beta = \alpha_i$, appearing inside D_i, because arc β is considered earlier in o_j than the time it is considered in o_i. Let $\text{in}_j(\beta')$ denote these arcs, $\text{in}_j(\beta') \subseteq \text{in}_j$.

2. Auxiliary arcs α', where $\alpha_k = \text{source}_j(\alpha') \in \mathcal{S}_i$, $s_{\alpha_k} = s_{\alpha'}$, $k > j$. These arcs are included in $\text{in}_j(\alpha_j) = \text{in}_j \setminus \text{in}_j(\beta')$, see Figure 5.4.

3. Arcs in out_j that lie below any *missing* auxiliary arcs of site s_{α_j} in $\mathcal{B}_i$ as $\mathcal{B}_j$ may contain no auxiliary arcs of α_j. Let these arcs be denoted as $\text{out}_j(\alpha_j')$, see Figure 5.4(b).

4. $\mathcal{B}_j$ may be *missing* auxiliary arcs of site $s_{\alpha_k}, k > j$, if β splits α_k. In their place, $\mathcal{B}_j$ contains auxiliary arcs of site $s_{\alpha_k}, k < j$, outside of D_i, denoted

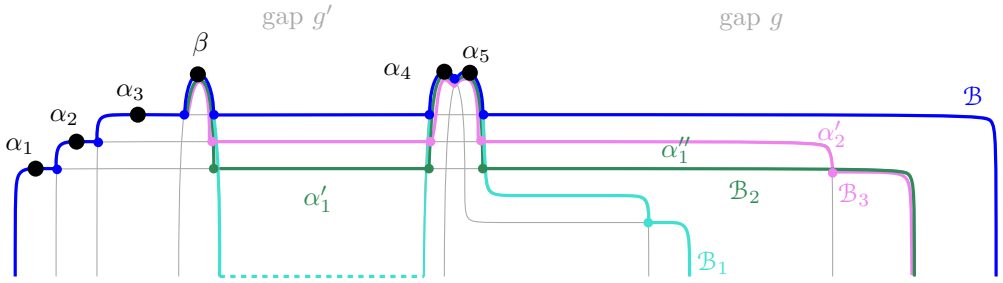

Figure 5.12. In this example, $o_i = (\alpha_1, \alpha_2, \alpha_3, \alpha_4, \alpha_5, \beta)$ and $o_1 = (\beta, \alpha_2, \alpha_3, \alpha_4, \alpha_5, \alpha_1)$. For $j = 1, 2$, $\mathcal{B}_j$ contains arcs of $\text{out}_j(\beta)$ ($\alpha_1 = \text{source}_2(\alpha_1') = \text{source}_2(\alpha_1'')$).

as $\text{out}_j(\beta) = \text{out}_j \setminus \text{out}_j(\alpha_j')$, see Figure 5.12. Note that $\text{out}_{j+1}(\beta) \setminus \text{out}_j(\beta)$ consists of auxiliary arcs of the site α_j (if any), if β splits α_j.

The auxiliary arcs of $\text{in}_j(\alpha_j)$ (2.) and $\text{out}_j(\alpha_j')$ (3.) appear equally for the group of Section 5.2. Observation 2 about $\text{out}_j(\alpha_j')$ holds here as well. Observations 3 and 4 about the arcs in $\text{in}_j(\alpha_j)$ (2.) is exactly the same as in Section 5.2 and Lemma 24 holds for these arcs as well. The arcs $\text{in}_j(\beta)$ (1.) do not cause any problem, because they add at most i arcs to the total sum in the group G. However, the auxiliary arcs of $\text{out}_j(\beta)$ (4.) add two new terms to Lemma 26 that have to be dealt with.

We now characterize the arcs in $\text{out}_j(\beta), 1 \le j < i$ and partition them into *substitution* and *new* arcs. For $2 \le j \le i$, let μ be an arc in $\text{out}_j(\beta) \setminus \text{out}_{j-1}(\beta)$. We call μ a *substitution* arc if the insertion of μ in $\mathcal{B}_{j-1}$ ($\mathcal{B}_{j-1} \oplus \mu$) deletes at least one arc in $\mathcal{B}_{j-1}$ (see Figure 5.13(b,c)), otherwise we call it a *new* arc (see Figure 5.13(a)). For $j = 1$, all arcs in $\text{out}_1(\beta)$ are considered *new* arcs.

Let $\text{count}(j)$ denote the number of all new arcs in out_j, and let $\text{count}(G) = \sum_{j=1}^{i} \text{count}(j)$.

Lemma 30. *$\text{count}(G) = O(i)$.*

Proof. First note that $\text{count}(1) = |\text{out}_1(\beta)| < i$ as $\text{out}_1(\beta)$ is a subset of $\mathcal{B}_1$. As before we only consider auxiliary arcs on $J^+_{\alpha_k}, k > j$, see Figure 5.12.

Since $o_{j-1} = (\alpha_1, \ldots, \alpha_{j-2}, \beta, \alpha_j, \ldots)$ and $o_j = (\alpha_1, \ldots, \alpha_{j-1}, \beta, \alpha_{j+1}, \ldots)$, the only possible difference between $\text{out}_j(\beta)$ and $\text{out}_{j-1}(\beta)$ are auxiliary arcs of the site $s_{\alpha_{j-1}}$. Observe that for any $\alpha_k' \in \text{out}_j(\beta)$ and $\alpha_\ell' \in \text{out}_m(\beta)$, where $k < \ell < j, m$ the arcs $(\alpha_k, \alpha_\ell, \beta)$ appear in ccw order; otherwise the ccw order $(\alpha_\ell, \alpha_k, \beta)$ would prevent α_ℓ' to appear in $\text{out}_m(\beta)$, because α_k is inserted before α_ℓ (in both o_j, o_m). Further, the auxiliary arcs $(\beta, \alpha_\ell', \alpha_k')$ (if any) appear in ccw order,

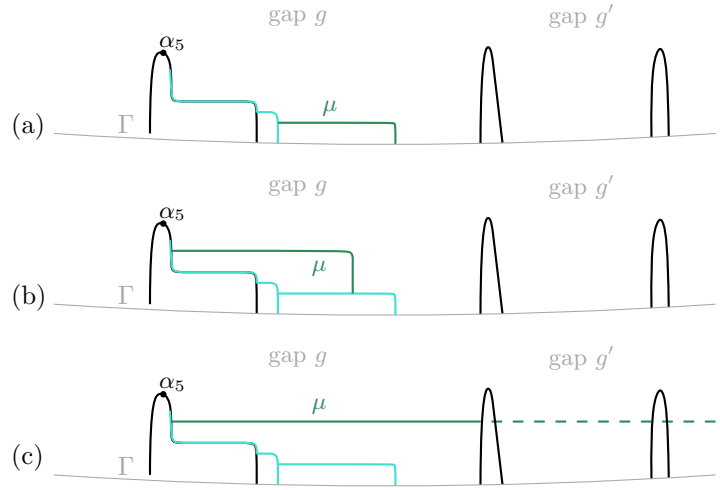

Figure 5.13. In (a) arc μ is a new arc. In (b) and (c) μ a substitution arc, because the $\mathcal{B}_{j-1} \oplus \mu$ deletes at least one arc in $\mathcal{B}_{j-1}$.

because the related bisectors J_{α_k} and J_{α_ℓ} intersect at most twice. Thus, all auxiliary arcs of α_{j-1} along $\mathcal{B}_j$ must appear *before* any auxiliary arc α'_k, $k < j-1$, i.e., $(\beta, \alpha_{j-1}, \alpha'_k)$ appear in ccw order (similarly to Observation 4). As a result, there can be only one gap where both an auxiliary arc α'_{j-1} and α'_k can appear. We conclude that $\alpha'_{j-1} \in \text{out}_j(\beta)$ can be *new* in at most one gap that contains already an arc α'_k for $k < j-1$. Since we only count new arcs in $\text{count}(j)$, this implies that $\sum_{j=2}^{i} \sum_{g \text{ gap}} \text{count}_g(j) = O(i)$, where $\text{count}_g(j)$ are the new arcs of $\text{count}(j)$ in gap g. □

Given any boundary curve $\mathcal{P}$, we define the *contracted* Voronoi-like diagram $\mathcal{V}_l^c(\mathcal{P})$ by uniting the regions of all auxiliary arcs within each gap of $\mathcal{P}$. By $|R_c(\alpha_j, \mathcal{P})|$ we denote the number of contracted arcs that have a region adjacent to $R_c(\alpha_j)$. In the following we analyze the degrees of the core arcs $\mathcal{S}_i$ on the contracted graphs, (i.e., $\deg(\alpha_j, \mathcal{P}) = |R_c(\alpha_j, \mathcal{P})|$) and the degrees of the gaps, $\deg(g, \mathcal{P})$, which is the number of contracted arcs in $\mathcal{V}_l^c(\mathcal{P})$ that have a region adjacent to gap g.

To be able to consider the portions of $\mathcal{B}_j$ in and out of the domain D_i separately we define a series of boundary curves $L_j, 1 \le j < i$, where L_j consists of the pieces of $\mathcal{B}_j$ outside the domain D_i, connected by pieces of $\mathcal{B}_i$, in particular, $L_j = \text{out}_j \cup (\mathcal{P} \setminus D_{\mathcal{B}_j})$ (see Figure 5.14).

Lemma 31. $\sum_{j=1}^{i} |R_c(\alpha_j, L_j)| = O(i)$.

Proof. We show that $\sum_{j=1}^{i} |R_c(\alpha_j, L_j)| \le 2|\mathcal{V}_l^c(\mathcal{B}_i)| = O(i)$.

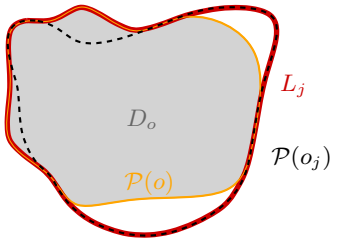

Figure 5.14. The boundary curve L_j (red) consists of the portion of $\mathcal{B}_j$ outside of D_i connected by $\mathcal{P}$.

When going from $j = 1$ to i, in the contracted Voronoi-like diagrams $\mathcal{V}_l^c(\mathcal{B}_j)$, an adjacency of α_j with another core arc α_ℓ can switch to an adjacency with a gap, and this adjacency in turn can switch to an adjacency between this gap and another gap. Thus, we can bound the total sum $\sum_{j=1}^{i} |R_c(\alpha_j, L_j)|$ to be less than the sum of the degrees of all original contracted arcs in $\mathcal{V}_l^c(L_i)$ plus the sum of the degrees of all contracted gaps in $\mathcal{V}_l^c(L_i)$. More formally,

$$\begin{aligned}
\sum_{j=1}^{i} |R_c(\alpha_j, L_j)| &= \sum_{j=1}^{i} \deg(\alpha_j, L_j) \leq \sum_{j=1}^{i} (a_j, L_j) + \sum_{\text{gaps } g} \deg(g, L_1) \\
&= \deg(\alpha_1, L_1) + \sum_{j=2}^{i} \deg(\alpha_j, L_j) + \sum_{\text{gaps } g} \deg(g, L_1) \\
&\leq \deg(\alpha_1, L_2) + \sum_{j=2}^{i} \deg(\alpha_j, L_j) + \sum_{\text{gaps } g} \deg(g, L_2) \\
&\vdots \\
&\leq \deg(\alpha_1, L_i) + \deg(\alpha_2, L_i) + \cdots + \deg(\alpha_i, L_i) + \sum_{\text{gaps } g} \deg(g, L_i) \\
&= 2|\mathcal{V}_l^c(\mathcal{B}_i)|
\end{aligned}$$

□

In addition to the terms in Lemma (26), in the group defined by equation 5.9, we have to take into account the additional auxiliary arcs $\text{out}_j(\beta)$ when bounding $|R(\alpha_j, \mathcal{B}_j)|$. Thus, we have to add the possible additional contracted adjacencies of α_j, which is done by the term $|R_c(\alpha_j, L_j)|$ and the possible (uncontracted) adjacencies within the gaps, which is done by the term count(G) for the entire group. This yields the following bound for the group G defined by the exchange rule of (5.9).

Lemma 32. $|R(\alpha_j, \mathcal{B}_j)| \leq 2(|R_c(\alpha_j, L_j)| + |R_c(\alpha_j, \mathcal{B}_i)| + |N_j|) + count(G)$.

The last lemma together with Lemmas 24, 30 and 31 allows to bound the total adjacencies in the group G similarly as in the last section by $\sum_{j=1}^{i} |R(\alpha_j, \mathcal{B}_j)| \in O(i)$.

5.3 Conclusion

In this chapter we have presented a *randomized incremental algorithm* that uses Voronoi-like diagrams as intermediate structures. Since Voronoi-like diagrams are simpler and can be computed faster than their real Voronoi counterparts, they realize optimal linear running time (in an expected sense) for the fundamental tree-like problem (1) of *site deletion*.

Thus, we have shown the usefulness and practicability of the Voronoi-like structure that we introduce in this dissertation. The algorithm is simple, not more complicated than their counterparts for point sites from Chew [1990]. However, establishing correctness and bounding the running time was a long and challenging process and needed the development of new techniques.

Chapter 6

Extensions of the randomized algorithm

In this chapter we present extensions of the randomized incremental algorithm of Chapter 5 for solving the fundamental tree-like problems (2) and (3) that we have listed in Section 1.3.

In Section 6.1 we design randomized linear-time algorithms for computing the order-(k+1) subdivision within an order-k Voronoi region (problem (2)), i.e., for a face f of $\mathcal{V}_k(S)$ we compute $\mathcal{V}_{k+1}(S) \cap f$.

In Section 6.2 we present an extension of our randomized algorithm to computing the farthest Voronoi diagram in expected linear time, after the sequence of its faces at infinity is known (problem (3)).

Section 6.1 is based on the following publication:

Kolja Junginger and Evanthia Papadopoulou. Abstract tree-like Voronoi diagrams in expected linear time. In *CGWeek Young Researchers Forum 2019*, Computational Geometry week, Portland, Oregon, 2019.

Section 6.2 is based on the publication:

Kolja Junginger and Evanthia Papadopoulou. *Deletion in abstract Voronoi diagrams in expected linear time*, CoRR abs/1803.05372. URL: `http://arxiv.org/abs/1803.05372`

6.1 Computing the order-(k+1) subdivision within an order-k region

In this section we present randomized linear-time algorithms for computing the order-(k+1) subdivision within an order-k Voronoi region (problem (2)). In particular, given a face f of $\mathrm{VR}_k(H,S)$, we compute $\mathcal{V}(S_f) \cap f$ in expected time $O(|\partial f|)$, where $S_f \subseteq S \setminus H$ denotes the set of sites inducing a Voronoi edge on

∂f, see Section 6.1.2. In the following we give the structure of this section.

In Section 6.1.1 we extend the algorithm of Section 5.1 for deletion of a site to compute in expected linear-time a Voronoi-like diagram for any subset of Voronoi edges $\mathcal{S}' \subseteq \partial\mathrm{VR}(s,S)$, and refer to this problem as the *extended deletion problem*. The Voronoi region $\mathrm{VR}(s,S)$ need not be known entirely. Given only a subset $\mathcal{S}'$ of its edges (or portions of them), we can compute in expected time $O(|\mathcal{S}'|)$, the Voronoi-like diagram of some boundary curve $\mathcal{P}'$, which contains the arcs in $\mathcal{S}'$, it has complexity $O(|\mathcal{S}'|)$, and $\mathcal{V}_l(\mathcal{P}') \cap D = \mathcal{V}(S \setminus \{s\}) \cap D$ for some domain $D \subseteq \mathrm{VR}(s,S)$.

The main complication in settling the *order-k problem* ($k > 1$), compared to the *deletion problem* ($k = 1$), is the complexity of ∂f, which depends on $k = |H|$ in addition to $|S_f|$. We show how to deal with $k > 1$ (using the arc insertion operation established in Section 3.3) through a reduction to the *extended deletion problem* (Section 6.1.2).

Our expected linear-time algorithm to compute the order-(k+1) subdivision reduces by a logarithmic factor (in the expected sense) the time complexity of the iterative construction of Lee [1982] (see Bohler, Liu, Papadopoulou and Zavershynskyi [2016] for the adaption to AVDs) to compute the order-k abstract Voronoi diagram i.e., from $O(k^2 n \log n)$ to $O(k^2 n + n \log n)$ steps.

6.1.1 The extended deletion problem

In this section we show that given a boundary curve $\mathcal{P}$ of $\mathcal{S}' \subseteq \mathcal{S}$, in expected time $O(|\mathcal{S}'|)$, we can construct $\mathcal{V}_l(\mathcal{P}')$ for some other boundary curve $\mathcal{P}'$, most likely different from $\mathcal{P}$. Combining it with Lemma 22 from Chapter 4, we derive an expected linear-time algorithm to compute $\mathcal{V}(S')$ within $D_{\mathcal{S}'}$. Note that computing $\mathcal{V}_l(\mathcal{P})$ for any fixed boundary curve $\mathcal{P}$ (including the envelope $\mathcal{E}$ or the plain curve $\mathcal{P}$) may require $\Theta(|\mathcal{P}|^2)$ time.

Recall that given a boundary curve $\mathcal{P}$ and arcs $\alpha, \beta \in \mathcal{P}$, we denote by $\mathcal{P}[\alpha, \beta]$ the portion of $\mathcal{P}$ from α to β in a counterclockwise traversal, including α and β. To establish the next theorem we first need a key property on $\mathcal{P}[\alpha, \beta]$ for any two *consecutive original arcs* α, β that is stated in the following lemma. An arc α partitions $J(s, s_\alpha)$ in two parts (see Figure 6.1); let $J^+(s, s_\alpha)$ (resp. $J^-(s, s_\alpha)$) denote the part incident to the clockwise (resp. counterclockwise) endpoint of α.

Lemma 33. *Let $\mathcal{P}$ be a boundary curve of $\mathcal{S}'$ and let $\alpha, \beta \in \mathcal{P}$ be any two consecutive original arcs. Then $J^+(s, s_\alpha)$ (resp. $J^-(s, s_\beta)$) cannot intersect $\mathcal{P}[\alpha, \beta]$. Thus, no auxiliary arc of s_α on $J^+(s, s_\alpha)$ and no auxiliary arc of s_β on $J^-(s, s_\beta)$ can appear on $\mathcal{P}[\alpha, \beta]$.*

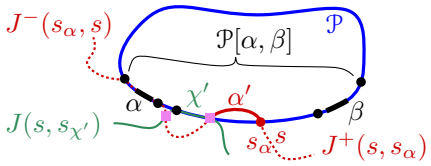

Figure 6.1. Assuming that $J^+(s, s_\alpha)$ intersects $\mathcal{P}[\alpha, \beta]$ contradicts α and β being consecutive original arcs.

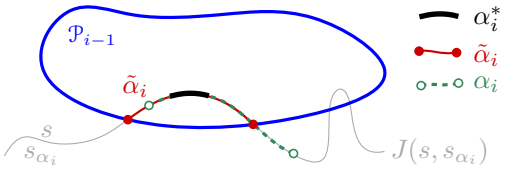

Figure 6.2. When inserting an original arc α_i into $\mathcal{P}_{i-1}$, we need to identify the component $\tilde{\alpha}_i$. Core arc $\alpha_i^* \in \mathcal{S}'$ is contained in $\tilde{\alpha}_i$ and in $\alpha_i \in \mathcal{P}$.

Proof. Suppose $J^+(s, s_\alpha)$ intersects $\mathcal{P}[\alpha, \beta]$ for the first time at arc $\chi' \subseteq J(s, s_{\chi'})$, creating an auxiliary arc α', see Figure 6.1. Since α is an original arc, and thus, $J(s, s_{\chi'})$ cannot *cut out* its core part, $J^+(s, s_\alpha)$ and $J(s, s_{\chi'})$ must intersect twice between α and α', see the pink squares in Figure 6.1. Thus, all core arcs of site $s_{\chi'}$ must appear on $J(s, s_{\chi'})$ between these two intersection points and there must be at least one such arc. This contradicts the fact that α and β are two consecutive original arcs on $\mathcal{P}$. □

We remark that component $J^-(s, s_\alpha)$ may still intersect $P[\alpha, \beta]$. An example is shown in Figure 6.3.

Theorem 8. *Given a boundary curve $\mathcal{P}$ on $\mathcal{S}' \subseteq \mathcal{S}$, in expected time $O(|\mathcal{S}'|)$, we can compute $\mathcal{V}_l(\mathcal{P}')$, for some boundary curve $\mathcal{P}'$ on $\mathcal{S}'$ that depends on the randomization order. Thus, in expected linear time, we can obtain $\mathcal{V}(S')$ within $D_{\mathcal{S}'}$, i.e., $\mathcal{V}(S') \cap D_{\mathcal{S}'} = \mathcal{V}_l(\mathcal{P}') \cap D_{\mathcal{S}'}$.*

$\mathcal{P}$ is assumed to know its original arcs, however, it may not know where their core portions in $\mathcal{S}'$ are. ($\mathcal{P}$ is typically the plain boundary curve $\mathcal{C}$.)

Proof. We apply the randomized approach of Chapter 5. Let π be a random permutation of $\mathcal{S}'$. Delete the original arcs in $\mathcal{S}'$ in reverse order π^{-1}, while registering their neighboring original arcs at the time of deletion. Insert back the arcs one

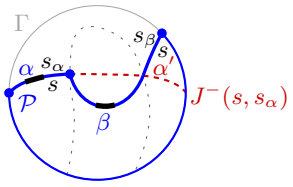

Figure 6.3. In the situation in Lemma 33, in contrast to $J^+(s, s_\alpha)$, this example shows that $J^-(s, s_\alpha)$ intersects $P[\alpha, \beta]$, where α, β are consecutive original arcs.

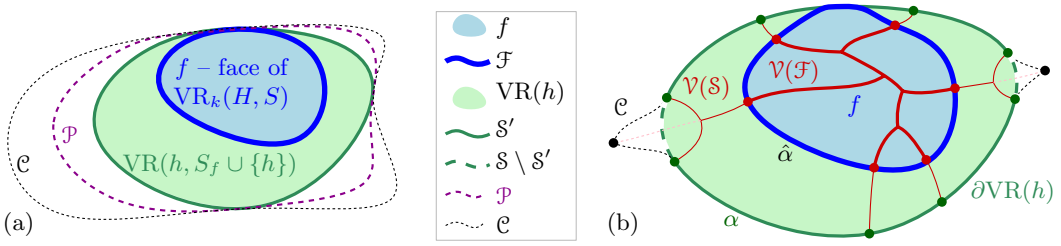

Figure 6.4. Arcs $\mathcal{F}$ are shown in blue, arcs $\mathcal{S}$ in green, $\mathcal{S}'$ as solid lines and $\mathcal{S}\setminus\mathcal{S}'$ as dashed lines.

by one in the order π, while computing $\mathcal{V}_l(\mathcal{P}_i)$, starting at $\mathcal{P}_1 = \partial(D(s,s_{a_1})\cap D_\Gamma)$ and $\mathcal{V}_l(\mathcal{P}_1)=\emptyset$. At the end, we obtain $\mathcal{V}_l(\mathcal{P}')$, which is some boundary curve of $\mathcal{S}'$. Which exactly boundary curve is $\mathcal{P}'$ depends on the insertion order π. In all cases $|\mathcal{P}'|$ is $O(|\mathcal{S}'|)$ as shown in Chapter 5. Thus, $|\mathcal{V}_l(\mathcal{P}')|$ is $O(|\mathcal{S}'|)$.

Lemma 33 resolves a technical issue that is associated with this construction, since the core arcs α_i^* need not be known. When considering arc $\alpha_i \in \mathcal{P}$ we need to correctly identify the component $\tilde{\alpha}_i$ of $J(s,s_{\alpha_i})\cap D_{\mathcal{P}_{i-1}}$ that contains the core arc α_i^*, revealing the corresponding original arc in $\mathcal{P}_i$, see Figure 6.2. By Lemma 33, there can be only one relevant component of $J(s,s_{\alpha_i})\cap D_{\mathcal{P}_{i-1}}$ with endpoints between the stored neighbors of α_i. Thus, we can easily identify this component by scanning $\mathcal{P}_{i-1}$, starting at one of the stored neighbors of α_i until its first intersection with $J(s,s_{\alpha_i})$ is found. Since α_i is original, this component of $J(s,s_{\alpha_i})\cap D_{\mathcal{P}_{i-1}}$ must contain α_i^*. Thus, by Lemma 33 this construction is possible. □

By Lemma 22, $\mathcal{V}_l(\mathcal{P}')$, $\mathcal{V}_l(\mathcal{C})$, and $\mathcal{V}(S')$, they all coincide within domain $D_{\mathcal{S}'}\subseteq D_{\mathcal{E}}$. This fact can lead to very simple, expected-linear time algorithms for various tree-like diagrams such as those considered in the following sections.

6.1.2 The order-k update

Let f be a face of the order-k Voronoi region $\mathrm{VR}_k(H)$, $H\subset S, |H|=k$. Let $S_f\subseteq S\setminus H$ denote the set of sites in $S\setminus H$ that induce the Voronoi edges on the boundary of f, ∂f. Our goal is to compute the order-1 Voronoi diagram of S_f, truncated within f, i.e., $\mathcal{V}(S_f)\cap f$. Clearly, $\mathcal{V}(S_f)\cap f = \mathcal{V}(S\setminus H)\cap f = \mathcal{V}_{k+1}(S)\cap f$.

Lemma 34. *Let f be a face of $VR_k(H)$. Then for any site $h\in H$, $f\subseteq VR(h, S_f\cup\{h\})$ (see Figure 6.4).*

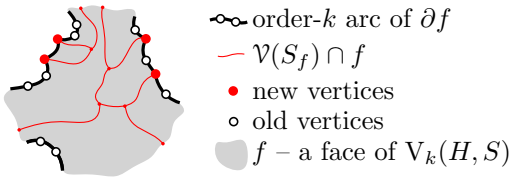

Figure 6.5. $\mathcal{V}(S_f) \cap f$, where f is unbounded. Order-k arcs are depicted as thick black arcs.

Proof.

$$f \subseteq \mathrm{VR}_k(H,S) = \bigcap_{p\in H, s\in S\setminus H} D(p,s) \subseteq \bigcap_{p\in H, s\in S_f} D(p,s) \subseteq \bigcap_{s\in S_f} D(h,s) = \mathrm{VR}(h, S_f \cup \{h\}).$$

□

Corollary 3. *$\mathcal{V}(S_f) \cap f$ is a subgraph of $\mathcal{V}(S_f) \cap \mathrm{VR}(h, S_f \cup \{h\})$.*

The Voronoi vertices along ∂f are classified into *new* and *old*, see Bohler et al. [2015]. Let v be an order-k Voronoi vertex incident to regions $\mathrm{V}_k(H_1,S)$, $\mathrm{V}_k(H_2,S)$, $\mathrm{V}_k(H_3,S)$, for sets $H_1,H_2,H_3 \subseteq S$ of cardinality k each. Vertex v is called *new* if $|H_1 \cap H_2 \cap H_3| = k-1$ and *old* if $|H_1 \cap H_2 \cap H_3| = k-2$. A new vertex in $\mathrm{V}_k(S)$ is an old vertex of $\mathcal{V}_{k+1}(S)$. The new vertices along ∂f (see Section 3.1) partition ∂f into *order-k arcs*, see Figure 6.5. An order-k arc α is a portion of the *Hausdorff bisector* between a site $s_\alpha \in S_f$ and H, which is defined as $J(s_\alpha, H) = \partial \mathrm{FVR}(s_\alpha, H \cup \{s_\alpha\})$. We say that an order-$k$ arc α is *induced* by site s_α.

The following lemma shows that $\mathcal{V}(S_f) \cap f$ is *tree-like*. For a bounded face f, it is a tree and the latter observation has also been stated by Bohler, Klein and Liu [2016]. An alternative proof for it has recently appeared in [Bohler et al., 2019, Lemma 4].[1]

Lemma 35. *$\mathcal{V}(S_f) \cap f$ is tree-like, i.e., it is a forest having exactly one face for each order-k arc of ∂f. Its leaves are the new vertices of ∂f, and points at infinity if f is unbounded. If f is bounded then $\mathcal{V}(S_f) \cap f$ is a tree.*

Proof. By Lemma 34, $f \subseteq \mathrm{VR}(h, S_f \cup \{h\})$ for any $h \in H$. By Lemma 1, every face in $\mathcal{V}(S_f) \cap \mathrm{VR}(h, S_f \cup \{h\})$ touches the boundary $\partial \mathrm{VR}(h, S_f \cup \{h\})$. Thus, every face in $\mathcal{V}(S_f) \cap f$ must touch the boundary ∂f; this implies that the diagram

[1] For more details on the order of the published results, see the discussion in the footnote on page 35.

does not contain cycles. Every order-k arc $\hat{\alpha} \subseteq J(s_{\hat{\alpha}}, H)$ on ∂f must be entirely contained in $\mathrm{VR}(s_{\hat{\alpha}}, S_f)$. Thus, no leaf can lie in the interior of an order-k arc of ∂f. On the other hand, each new vertex of ∂f must be a leaf of the diagram as its incident order-k arcs are induced by different sites.

Now we show that no two order-k arcs of ∂f can be incident to the same face of $\mathcal{V}(S_f) \cap f$. Consider two order-k arcs α, β on ∂f induced by the same site $p \in S_f$. Suppose these two order-k arcs are incident to the same face f_α of $\mathcal{V}(S_f) \cap f$ as shown in Figure 6.6. Let $J(h_1, p)$ and $J(h_2, p)$ be two bisectors of one of the Voronoi edges in α and β, respectively. The bisectors $J(h_1, p)$ and $J(h_2, p)$ must intersect outside of f in a point I_1, otherwise $\mathrm{VR}(p, \{p, h_1, h_2\})$ would be empty. If $h_1 = h_2$, then $J(h_1, p) = J(h_2, p)$.

There exists an order-k arc γ between α and β on ∂f, induced by a site $q \neq p$. Since $\gamma \subset \mathrm{VR}(q, S_f)$ and the face $f_\alpha \subset \mathrm{VR}(p, S_f)$, we know that $\gamma \subset D(q, p)$ and $f_\alpha \subset D(p, q)$. Since the bisector system is admissible, and thus $\mathrm{VR}(p, \{p, q, h_1, h_2\})$ is non-empty, $J(p, q)$ intersects $D(p, h_1) \cap D(p, h_2)$ in a point I_2 as shown in Figure 6.6. W.l.o.g. $I_2 \in J(p, h_1)$. Since the bisector $J(p, q)$ is unbounded it intersects $J(h_1, p)$ in a second point I_3, implying that $\mathrm{VR}(p, \{p, q, h_1\})$ is disconnected, which is a contradiction.

Thus $\mathrm{VR}(p, S_f) \cap f$ cannot be connected and the two order-k arcs of p (α, β) must be incident to different faces of $\mathcal{V}(S_f) \cap f$.

If f is unbounded, two consecutive order-k arcs of ∂f can extend to infinity, in which case there is at least one order-k arc of $\mathcal{V}(S_f) \cap f$ extending to infinity between them; thus, leaves can be points at infinity. If f is bounded, all leaves of $\mathcal{V}(S_f) \cap f$ must lie on ∂f. Since no face is incident to more than one order-k arc of ∂f, in this case $\mathcal{V}(S_f) \cap f$ cannot be disconnected, and thus is a tree. □

We reduce the order-k problem of computing $\mathcal{V}(S_f) \cap f$ to the problem of Theorem 8. Consider the Voronoi region $\mathrm{VR}(h, S_f \cup \{h\}) \cap D_\Gamma$, see Figure 6.4. By Lemma 34 it entirely encloses f. Let $\mathcal{S}$ denote the sequence of the Voronoi edges (order-k arcs) along the boundary $\partial \mathrm{VR}(h, S_f \cup \{h\}) \cap D_\Gamma$. The set $\mathcal{S}$ defines the Voronoi diagram $\mathcal{V}(\mathcal{S})$, which is the portion of $\mathcal{V}(S_f)$ truncated within region $\mathrm{VR}(h, S_f \cup \{h\})$, i.e., $\mathcal{V}(\mathcal{S}) = \mathcal{V}(S_f) \cap \mathrm{VR}(h, S_f \cup \{h\}) \cap D_\Gamma$. The face of this diagram incident to an arc $\alpha \in \mathcal{S}$ is considered its Voronoi region $\mathrm{VR}(\alpha, \mathcal{S})$. Similarly, let $\mathcal{F}$ denote the sequence of the order-k arcs along ∂f; the Voronoi diagram of $\mathcal{F}$ is $\mathcal{V}(\mathcal{F}) = \mathcal{V}(S_f) \cap f \cap D_\Gamma$, see Figure 6.4(b). Let $\mathrm{VR}(\hat{\alpha}, \mathcal{F})$ denote the face of this diagram incident to arc $\hat{\alpha}$.

From Corollary 3, we derive a 1−1 correspondence from the order-k arcs in $\mathcal{F}$ to $\mathcal{S}$. Let $\mathcal{S}' \subseteq \mathcal{S}$ denote the range of this correspondence.

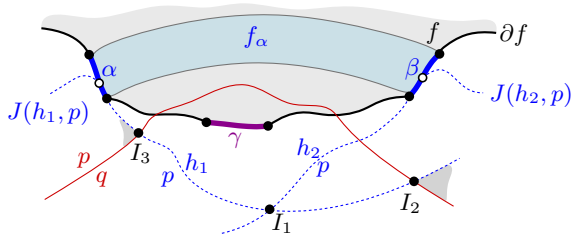

Figure 6.6. The assumption of one face f_α of $\mathcal{V}(S_f)\cap f$ which is incident to two order-k arcs α and β yields a contradiction to axiom (A1).

Lemma 36. *There is a* 1−1 *correspondence from* $\mathcal{F}$ *to* $\mathcal{S}$*, where each order-k arc* $\hat{\alpha}\in\mathcal{F}$ *(*$\hat{\alpha}\subseteq J(H,s_\alpha)$*) corresponds to an arc* $\alpha\in\mathcal{S}$ *(*$\alpha\subseteq J(h,s_\alpha)$*). In addition,* $VR(\alpha,\mathcal{S})\cap f = VR(\hat{\alpha},\mathcal{F})$.

Proof. Each order-k arc $\hat{\alpha}\in\mathcal{F}$ has a face $\mathrm{VR}(\hat{\alpha},\mathcal{F})$ that contains a face $\mathrm{VR}(\alpha,\mathcal{S})$ of $\mathcal{V}(\mathcal{S})$ by Corollary 3. Since $\mathcal{V}(\mathcal{S})$ is tree-like (Lemma 1) each face $\mathrm{VR}(\alpha,\mathcal{S})$ of $\mathcal{V}(\mathcal{S})$ has a unique owner, which is the corresponding arc $\alpha\in\mathcal{S}$. □

By combining Lemma 36, Lemma 22, and because $f\subseteq D_{\mathcal{S}'}$ we establish:

Lemma 37. *For any boundary curve* $\mathcal{P}$ *for* $\mathcal{S}'$*,* $\mathcal{V}_l(\mathcal{P})\cap f = \mathcal{V}(\mathcal{F}) = \mathcal{V}(S_f)\cap f\cap D_\Gamma$.

Proof. Recall that $D_{\mathcal{S}'} = D_{\mathcal{E}}\setminus\bigcup_{\text{auxiliary }\alpha\in\mathcal{E}}\overline{R(\alpha,\mathcal{E})}$. By Lemma 36 there is a core arc $\alpha\in\mathcal{S}'$ for every region $\mathrm{VR}(\hat{\alpha},\mathcal{F})$ such that Voronoi region $R(\alpha,\mathcal{E})\supseteq\mathrm{VR}(\hat{\alpha},\mathcal{F})$. Thus, in the diagram $\mathcal{V}_l(\mathcal{E})$, the union of the (closure of the) regions of the original arcs related to $\mathcal{S}'$ contain the entire face f, and thus, we have $f\subseteq D_{\mathcal{S}'}$. The claim follows by Lemma 22. □

Lemma 38. $\mathcal{C}$ *can be computed from* ∂f *in* $O(|\partial f|)$ *time.*

Proof. Refer to Figure 6.7. For every three consecutive order-k arcs $\hat{\alpha},\hat{\beta},\hat{\gamma}\in\mathcal{F}$ define β to be the portion of $J(h,s_{\hat{\beta}})$ between the first intersection with $J(h,s_{\hat{\alpha}})$ (if $s_{\hat{\alpha}}\neq s_{\hat{\beta}}$) and $J(h,s_{\hat{\gamma}})$ (if $s_{\hat{\gamma}}\neq s_{\hat{\beta}}$), resp., or with Γ in case they don't intersect. If $s_{\hat{\alpha}}=s_{\hat{\beta}}$, and use $\hat{\alpha}$'s predecessor for finding the endpoint of the arc β. In the end, $\mathcal{C}$ is the h-monotone path that we obtain by linking the consecutive arcs that are incident to Γ by a Γ-arc. Clearly the running time is $O(|\partial f|)$. □

Lemma 37 (Lemma 22) and Theorem 8 imply a very simple approach to compute $\mathcal{V}(S_f)\cap f$: Let $\mathcal{C}$ denote the plain boundary curve for $\mathcal{S}'$, which is derived from ∂f in time $O(|\partial f|)$ by applying Lemma 36 (see Lemma 38). Run the randomized algorithm of Section 6.1.1 on $\mathcal{C}$ as modified by Theorem 8. It will compute $\mathcal{V}_l(\mathcal{P}')$ for some boundary curve $\mathcal{P}'$ for $\mathcal{S}'$ (see Theorem 8). By Lemma 37, output $\mathcal{V}_l(\mathcal{P}')\cap f$, which is $\mathcal{V}(S_f)\cap f\cap D_\Gamma$.

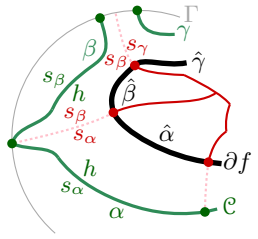

Figure 6.7. Constructing $\mathcal{C}$ from ∂f.

Theorem 9. *Given a face f of the order-k Voronoi region $VR_k(H,S)$, $\mathcal{V}(S_f)\cap f$ can be computed in expected $O(m)$ time, where m is the complexity of ∂f. Thus, we can compute $\mathcal{V}_{k+1}(S)\cap f$ in expected time $O(m)$.*

6.2 The farthest abstract Voronoi diagram

In this section we show how to modify and simplify the algorithm for deletion of a region of the previous sections to compute the *farthest* abstract Voronoi diagram, after the sequence of its faces at infinity is known.

The *farthest Voronoi region* of a site $p \in S$ is $\mathrm{FVR}(p,S) = \bigcap_{q\in S\setminus\{p\}} D(q,p)$ and the *farthest abstract Voronoi diagram* of S is $\mathrm{FVD}(S) = \mathbb{R}^2 \setminus \bigcup_{p\in S} \mathrm{FVR}(p,S)$. $\mathrm{FVD}(S)$ is a tree of complexity $O(n)$, however, regions may be disconnected and a farthest Voronoi region may consist of $\Theta(n)$ disjoint faces, see Mehlhorn et al. [2001]. Let $D^*(p,q) = D(q,p)$; then $\mathrm{FVR}(p,S) = \bigcap_{q\in S\setminus\{p\}} D^*(p,q)$.

Unless otherwise noted, we adopt the following convention: we reverse the labels of bisectors and use $D^*(\cdot,\cdot)$, in the place of $D(\cdot,\cdot)$, in most definitions and constructs of Sections 3.2, 3.3. Under this convention the definition of an e.g., p-monotone path remains the same but it uses $\partial\,\mathrm{FVR}(p,\cdot)$ in the place of $\partial\,\mathrm{VR}(p,\cdot)$. The corresponding arrangement of p-related bisectors $\mathcal{J}_{p,S'}$, $S' \subseteq S$, is considered with the labels of bisectors and their dominance regions reversed from the original system $\mathcal{J}$.

Consider the enclosing curve Γ as defined in Chapter 3, and let $\mathcal{S}$ be the sequence of arcs on Γ derived by $\Gamma \cap \mathrm{FVD}(S)$. $\mathcal{S}$ represents the sequence of the farthest Voronoi faces in $\mathrm{FVD}(S)$ at infinity. The domain of computation is D_Γ. For an arc α of $\mathcal{S}$ let s_α denote the site in S for which $\alpha \subset \mathrm{FVR}(s_\alpha, S)$. With respect to site occurrences, $\mathcal{S}$ is a Davenport-Schinzel sequence of order 2. $\mathcal{S}$ can be computed in time $O(n \log n)$ in a divide and conquer fashion, similarly to computing the *hull* of a farthest segment Voronoi diagram, see e.g., Papadopoulou and Dey [2013].

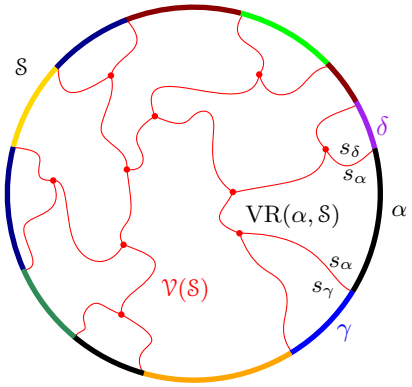

Figure 6.8. The farthest Voronoi diagram $\mathcal{V}(\mathcal{S}) = \mathrm{FVD}(S) \cap D_\Gamma$ and the Voronoi region $\mathrm{VR}(\alpha, \mathcal{S})$. Bisector labels are shown in the farthest (reversed) sense.

We treat the arcs in $\mathcal{S}$ as sites and compute $\mathcal{V}(\mathcal{S}) = \mathrm{FVD}(S) \cap D_\Gamma$. Let $\mathrm{VR}(\alpha, \mathcal{S})$ denote the face of $\mathrm{FVD}(S) \cap D_\Gamma$ incident to $\alpha \in \mathcal{S}$, see Figure 6.8. $\mathcal{V}(\mathcal{S})$ is a tree whose leaves are the endpoints of the arcs in $\mathcal{S}$.

For $\mathcal{S}' \subseteq \mathcal{S}$, let $S' \subseteq S$ be the set of sites that define the arcs in $\mathcal{S}'$. Let $\mathcal{J}(S') = \{J(p,q) \in \mathcal{J} \mid p, q \in S', p \neq q\}$.

Definition 12. *A* boundary curve $\mathcal{P}$ for $\mathcal{S}'$ *is a partitioning of* Γ *into* arcs *whose endpoints are in* $\Gamma \cap \mathcal{J}(S')$ *such that any two consecutive arcs* $\alpha, \beta \in \mathcal{P}$ *are incident to* $J(s_\alpha, s_\beta) \in \mathcal{J}(S')$*, having consistent labels, and* $\mathcal{P}$ *contains an arc* $\alpha \supseteq \alpha^*$*, for every core arc* $\alpha^* \in \mathcal{S}'$*. We say that the labels of* α*,* β *are* consistent*, if there is a neighborhood* $\tilde{\alpha} \subseteq \alpha$ *incident to the common endpoint of* α *and* β *such that* $\tilde{\alpha} \in D^*(s_\alpha, s_\beta)$*, and respectively for* β*.*

There can be several different boundary curves for $\mathcal{S}'$. The arcs in $\mathcal{P}$ that contain a core arc in $\mathcal{S}'$ are called *original* and any remaining arcs are called *auxiliary*. The arcs in $\mathcal{P}$, although they are arcs on Γ, they are all boundary arcs and none is considered a Γ-arc in the sense of the previous sections. The endpoint $J(s_\alpha, s_\beta) \cap \Gamma$ on $\mathcal{P}$ separating two consecutive arcs α, β is denoted by $\nu(\alpha, \beta)$.

The Voronoi-like diagram of a boundary curve $\mathcal{P}$ is defined analogously to Definition 6. Since $\mathcal{P}$ consists only of boundary arcs, $\mathcal{V}_l(\mathcal{P})$ is a tree whose leaves are the vertices of $\mathcal{P}$. The properties of a Voronoi-like diagram in Section 3.2 remain the same (under the conventions of this section).

Given $\mathcal{V}_l(\mathcal{P})$ for a boundary curve $\mathcal{P}$ of $\mathcal{S}' \subset \mathcal{S}$, we can insert a core arc $\beta^* \in \mathcal{S} \setminus \mathcal{S}'$ and obtain $\mathcal{V}_l(\mathcal{P} \oplus \beta^*)$. The insertion is performed analogously to Section 3.3. Let $\beta \supseteq \beta^*$ be defined as follows: let x, y denote the endpoints of β, and let δ be the first arc on $\mathcal{P}$ counterclockwise (resp. clockwise) from β^* such that

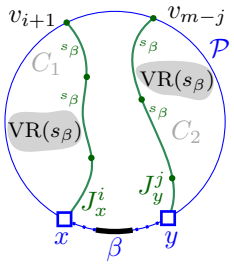

Figure 6.9. Illustration for Lemma 40. Nearest labels are shown.

$J(s_\beta, s_\delta) \cap \delta \neq \emptyset$; let $x = \nu(\delta, \beta)$ (resp. $y = \nu(\beta, \delta)$). Let $\mathcal{P}_\beta = \mathcal{P} \oplus \beta$ be the boundary curve obtained from $\mathcal{P}$ by substituting with β its overlapping piece. No original arc of $\mathcal{P}$ can be deleted in $\mathcal{P}_\beta$. Observation 1 remains the same, except from cases (d),(e) that do not exist.

The *merge curve* $J(\beta)$, given $\mathcal{V}_l(\mathcal{P})$, is defined analogously to Definition 7; it is only simpler as it does not contain Γ-arcs. Theorem 2 remains valid, i.e., $J(\beta)$ is an s_β-monotone path in $\mathcal{J}_{s_\beta, S'}$ connecting the endpoints of β. The proof structure is the same as for Theorem 2, however, Lemma 13 now requires a different proof, which we give in the sequel (see Lemma 40). Lemma 14 is not relevant; while Lemma 15 and Lemma 16 are analogous.

In the following lemma we restore the labeling of bisectors to the original.

Lemma 39. *In an admissible bisector system $\mathcal{J}$ (resp. $\mathcal{J} \cup \Gamma$) there cannot be two p-cycles, $p \in S$, with disjoint interior.*

Proof. By its definition, the nearest Voronoi region $\mathrm{VR}(p, S)$ (resp. $\mathrm{VR}(p, S) \cap D_\Gamma$) must be enclosed in the interior of any p-cycle of the admissible bisector system $\mathcal{J}$ (resp. $\mathcal{J} \cup \Gamma$). But $\mathrm{VR}(p, S)$ (resp. $\mathrm{VR}(p, S) \cap D_\Gamma$) is connected (by axiom (A1)), thus, there cannot be two different p-cycles with disjoint interior. □

Lemma 40. *Consider the merge curve $J(\beta)$. Suppose v_{i+1} is not a valid vertex because $v_{i+1} \in \alpha_i$, i.e., e_i hits arc α_i. Then vertex v_{m-j} can not be on $\mathcal{P}$.*

Proof. Suppose otherwise, i.e., vertex v_{m-j} is on the boundary arc α_{m-j}. Then J_x^i and J_y^j partition D_Γ in three parts: a middle part incident to β, and two parts C_1 and C_2 at either side of J_x^i and J_y^j respectively, whose closures are disjoint, see Figure 6.9. But the boundaries of C_1 and C_2 are s_β-cycles in the admissible bisector system $\mathcal{J} \cup \Gamma$ contradicting Lemma 39. Note that here we use the original labels of bisectors, including $\Gamma = J(s_\beta, s_\infty)$. □

The diagram $\mathcal{V}_l(\mathcal{P}) \oplus \beta$ is defined analogously and the proof that $\mathcal{V}_l(\mathcal{P}) \oplus \beta$ is a Voronoi-like diagram for $\mathcal{P}_\beta = \mathcal{P} \oplus \beta$, $\mathcal{V}_l(\mathcal{P}_\beta)$, is analogous to the proof of Theorem 3.

The randomized algorithm for computing $\mathcal{V}(\mathcal{S}) = \text{FVD}(S) \cap D_\Gamma$ is the same as in Section 5.1. Following the same analysis as in Section 5.2, the (expected) linear time complexity can be established.

The sets in_j and out_j in the time analysis are defined analogously, however, their exact definition is slightly different from Section 5.2. Here out_j are the auxiliary arcs in $\mathcal{B}_j$ that overlap with the auxiliary arcs of α_j in $\mathcal{B}_i$. The set in_j are any remaining auxiliary arcs in $\mathcal{B}_j \setminus \text{out}_j$ that differ from the corresponding auxiliary arcs in $\mathcal{B}_i$. All observations of Section 5.2 remain intact under the updated notion of in_j and out_j. Thus, all counting arguments remain also the same, except from cases (d) and (e) of Observation 1 that do not occur, simply setting their respective parameters to 0.

Theorem 10. *Given the sequence of its faces at infinity (i.e., $\mathcal{S}$),* $\text{FVD}(S)$ *can be computed in expected $O(h)$ time, where $h \in O(n)$ is the number of faces of* $\text{FVD}(S)$ *($h = |\mathcal{S}|$).*

6.3 Conclusion

In this chapter we have extended the algorithm for site deletion of Chapter 5 to solve the fundamental tree-like problems (2) and (3) that we introduced in Chapter 1: *the order-k* and the *farthest* problem.

Thus, we have shown uses of Voronoi-like diagrams as intermediate structures in various randomized incremental algorithms yielding optimal expected linear running time for tree-like Voronoi problems.

Chapter 7

Towards a deterministic linear-time algorithm for Voronoi-like diagrams

Updating an abstract Voronoi diagram, after deletion of one site, in *deterministic* linear time is still a well-known open problem. We have investigated the applicability of Voronoi-like diagrams in the linear-time framework of Aggarwal et al. [1989] in our research. However, results have not been conclusive.

In this chapter we present the idea how Voronoi-like diagrams might be used in the deterministic linear-time framework of Aggarwal et al. [1989] and explain the difficulties. We formulate open problems the solution of which would establish significant progress on the way to a deterministic linear-time algorithm for the problem of deletion in an abstract Voronoi diagram based on Voronoi-like diagrams.

First we summarize the linear-time framework of Aggarwal et al. [1989] for points in convex position (which is also sketched in Chapter 2) on a high level as follows:

Linear-time Algorithm (point sites) by Aggarwal et al. [1989]

Input: The sequence of a set of point-sites S along the convex hull of S, where S is in convex position.
Output: The Voronoi diagram $\mathcal{V}(S)$.

1. Partition S into a *red* set R and a *blue* set B of almost equal size with a *Coloring rule*.

2. Recursively compute the blue Voronoi diagram $\mathcal{V}(B)$.

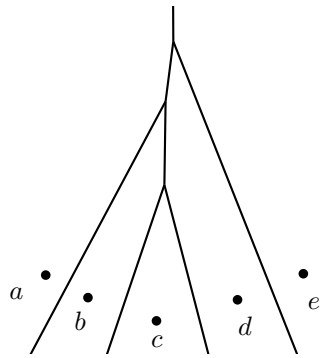

Figure 7.1. According to the coloring rule of step 1 site c is colored red, because the Voronoi regions VR(b) and VR(d) are adjacent in the Voronoi diagram $\mathcal{V}(\{a,b,c,d,e\})$.

3. Using $\mathcal{V}(B)$ apply the *combinatorial lemma*, which outputs a constant fraction of *crimson* sites $C \subset R$ such that no two crimson Voronoi regions intersect the same edge.

4. Insert the (non-adjacent) crimson regions one-by-one to the blue diagram $\mathcal{V}(B)$ in linear time.

5. Recursively compute the *garnet* Voronoi diagram $\mathcal{V}(G), G = S \setminus (B \cup C)$.

6. Merge $\mathcal{V}(B \cup C)$ with $\mathcal{V}(G)$ into $\mathcal{V}(S)$ in linear time $O(|S|)$.

The coloring rule for points in convex position is the following (see Aggarwal et al. [1989]; Klein and Lingas [1994]):

Coloring rule (step 1). Let a,b,c,d,e denote five consecutive sites of a subset $S' \subseteq S$ according to the ordering of the convex hull of S. Site c is colored red if in the Voronoi diagram $\mathcal{V}(\{a,b,c,d,e\})$ there is a Voronoi vertex incident to VR(b), VR(c) and VR(d) or equivalently the regions VR(b) and VR(d) are adjacent in $\mathcal{V}(\{a,b,c,d,e\})$. See Figure 7.1.

Coloring properties. For a subset $S' \subseteq S$, let R be all red colored sites by the above rule and let the remaining sites $B = S' \setminus R$ be blue. Then the following properties can be shown (see Aggarwal et al. [1989]; Klein and Lingas [1994]):

- No two consecutive sites of S' belong in the red set.
- The number of red sites is roughly equal to the number of blue sites, i.e., $R = \Theta(|B|)$.

- Let $a, b \in R$ be two successive red sites. Then the Voronoi regions $\mathrm{VR}(a, B \cup \{a, b\})$ and $\mathrm{VR}(b, B \cup \{a, b\})$ in $\mathcal{V}(B \cup \{a, b\})$ are not adjacent.

Idea for Voronoi-like diagrams

Recall that Voronoi-like diagrams are defined for boundary curves derived from a subset of *arcs* $\mathcal{S}$ instead of sites S. The idea is to use this set of arcs $\mathcal{S}$ as input to the linear-time algorithm and compute Voronoi-like diagrams for boundary curves of subset of arcs.

We have identified three major challenges in order to realize the use of Voronoi-like diagrams in the deterministic linear-time framework.

Challenges

C.1 Define a *colouring rule* for arcs instead of sites, see step 1 of the algorithm, and prove its correctness.

C.2 Generalise the *combinatorial lemma* to be able to deal with the much more complex situation when dealing with Voronoi-like diagrams, see step 3 of the algorithm. This includes tree-like graphs instead of simple trees as in the original case.

C.3 *Merge* two Voronoi-like diagrams in linear time, see step 6 of the algorithm.

In the following we explain the difficulties of these challenges in more detail.

7.1 Challenge C.1: A coloring rule

Generalizing the coloring rule of step 1 of the algorithm to the Voronoi-like setting constitutes the main problem. Neither the coloring rule from Aggarwal et al. [1989] nor from Bohler et al. [2014] can be applied. A new formulation is necessary.

The reason for the particular difficulty of establishing the *Colouring rule* in our setting is the following: Voronoi-like diagrams don't have the standard *monotonicity property* that real Voronoi diagrams have (illustrated in Figure 7.2):

$$S_1 \subseteq S_2 \quad \Rightarrow \quad \mathrm{VR}(p, S_2) \subseteq \mathrm{VR}(p, S_1),\ p \in S_1. \tag{7.1}$$

In contrast for Voronoi-like diagrams a subset relation of the arcs $\mathcal{S}_1 \subseteq \mathcal{S}_2$ does *not* imply a subset relation for the regions $R(\alpha, \mathcal{P}_2)$ and $R(\alpha, \mathcal{P}_1)$ of a common arc

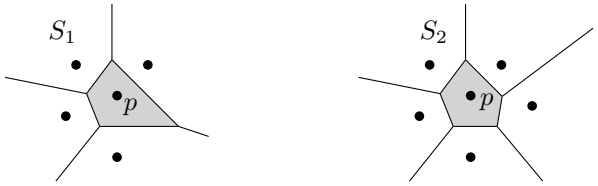

Figure 7.2. In a Voronoi diagram, $S_1 \subseteq S_2$ implies $\mathrm{VR}(p, S_2) \subseteq \mathrm{VR}(p, S_1)$.

α, for two arbitrary boundary curves $\mathcal{P}_1$ and $\mathcal{P}_2$ on $\mathcal{S}_1$ and $\mathcal{S}_2$, respectively:

$$\mathcal{S}_1 \subseteq \mathcal{S}_2 \not\Rightarrow R(\alpha, \mathcal{P}_2) \subseteq R(\alpha, \mathcal{P}_1), \alpha \in \mathcal{S}_1. \tag{7.2}$$

See Figure 7.3 for an example, where $\mathcal{S}_1 \subseteq \mathcal{S}_2$, but $R(\alpha, \mathcal{P}_2) \not\subseteq R(\alpha, \mathcal{P}_1)$.

In the linear-time frame work for real Voronoi diagrams the monotonicity property (7.1) is crucial for proving that the coloring rule implies the desired adjacency properties (see the *coloring properties* on page 98). The lack of monotonicity in the Voronoi-like setting makes the formulation of a coloring rule a non-trivial challenge.

Question 2. *Given a subset of Voronoi edges $\mathcal{S}' \subseteq \mathcal{S}$, is there a way to partition $\mathcal{S}'$ into blue and red arcs B and R in $O(|\mathcal{S}'|)$ time such that the following properties are satisfied:*

- *No two consecutive arcs of $\mathcal{S}'$ belong in the red set R.*
- *$R = \Theta(|B|)$.*
- *Let $\beta, \delta \in R$ be two successive red arcs and $\mathcal{P}$ be a boundary curve for $B \cup \{\beta, \delta\}$. Then the regions $R(\beta, \mathcal{P})$ and $R(\delta, \mathcal{P})$ in $\mathcal{V}_l(\mathcal{P})$ are not adjacent.*

7.2 Challenge C.2: The combinatorial lemma

Generalizing the *combinatorial lemma* from the point-site setting to the abstract Voronoi-like setting is the next challenge.

The combinatorial lemma of Aggarwal et al. [1989] takes as input the Voronoi diagram $\mathcal{V}(B)$, which is an *unrooted binary tree* $\mathcal{T}$ embedded in the plane. Each *leaf* of $\mathcal{T}$ is associated with a *neighborhood*, which is a subtree of $\mathcal{T}$ rooted at that leaf with the property that consecutive leaves of $\mathcal{T}$ have disjoint neighborhoods. A neighborhood of a leaf ℓ corresponds to the Voronoi region of a red site $r \in R$ when inserted in $\mathcal{V}(B)$, i.e., leaf ℓ corresponds to site $r \in R$ if ℓ is contained in

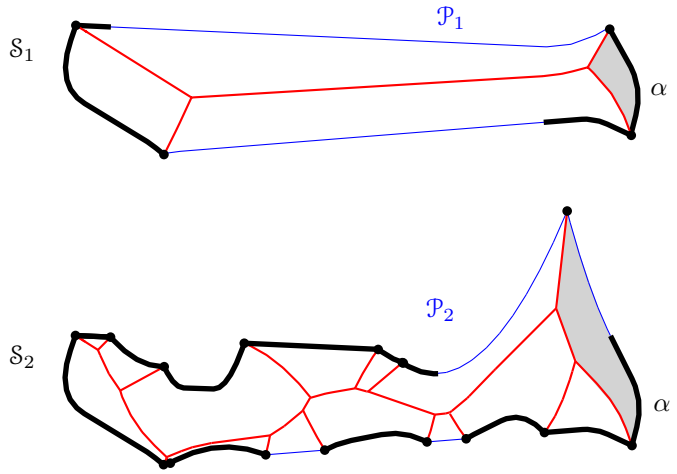

Figure 7.3. Two Voronoi-like diagrams $\mathcal{V}_l(\mathcal{P}_1)$ and $\mathcal{V}_l(\mathcal{P}_2)$ (in red), where $R(\alpha, \mathcal{P}_2) \supset R(\alpha, \mathcal{P}_1)$, even though $\mathcal{S}_1 \subset \mathcal{S}_2$. ($\mathcal{P}_1$ and $\mathcal{P}_2$ are boundary curves for $\mathcal{S}_1$ and $\mathcal{S}_2$, respectively.) The core arcs $\mathcal{S}_1$ and $\mathcal{S}_2$ are black bold.

$\mathrm{VR}(r, B \cup \{r\})$ and the neighborhood of ℓ is $\mathcal{V}(B) \cap \mathrm{VR}(r, B \cup \{r\})$. The combinatorial lemma outputs in linear time a constant fraction of the leaves (respectively a subset of sites R) such that their neighborhoods are pairwise disjoint. These leaves of the output correspond to the *crimson* sites in step 3 in the linear-time Algorithm.

Some progress has been achieved in generalizing the combinatorial lemma from the point-site setting to the abstract Voronoi-like setting, see Junginger, Mantas and Papadopoulou [2019]. One part of the generalization is that not every leaf of the tree has an associated neighborhood. Instead the leaves are partitioned in two parts, the *marked* and the *unmarked* leaves, where only the marked have a neighborhood. The following theorem is the main result of this paper.

Theorem 11. *Let $\mathcal{T}$ be an unrooted binary tree embedded in the plane with n leaves, m of which have been marked. Each marked leaf of $\mathcal{T}$ is associated with a neighborhood, which is a subtree of $\mathcal{T}$ rooted at that leaf; consecutive marked leaves in the topological ordering of $\mathcal{T}$ have disjoint neighborhoods. Then, there exist at least $\frac{1}{10}m$ marked leaves in $\mathcal{T}$, with pairwise disjoint neighborhoods, such that no tree edge has its endpoints in two different neighborhoods. We can select at least p of these $\frac{1}{10}m$ marked leaves in time $O(\frac{1}{1-p}n)$, for any $p \in (0, 1)$.*

In the Voronoi-like setting the *marked* leaves correspond to regions of *original* arcs and the *unmarked* leaves to regions of *auxiliary* arcs. Here a challenge was

to ensure the fraction $\frac{p}{10}$ of the m marked leaves in $\mathcal{T}$ with pairwise disjoint neighborhoods without increasing the time complexity too much.

However, we emphasize that Theorem 11 only is able to deal with trees and not with *tree-like* graphs (see Definition 1) as needed e.g., in case of the problem of deleting a site in an abstract Voronoi diagram using Voronoi-like diagrams as well as in the order-(k+1) subdivision within an order-k abstract Voronoi region. We believe that further generalizing Theorem 11 to *tree-like* graphs is a non-trivial task, even in the more specific case without allowing unmarked leaves, but with marked leaves only.

Question 3. *Can Theorem 11 be generalized for tree-like graphs in order to be applicable to Voronoi-like diagrams as needed e.g., in the problem of site deletion?*

7.3 Challenge C.3: Merging two Voronoi-like diagrams

Merging two Voronoi-like diagrams in linear time is the third challenge, for which we have made the following observation: The property from [Klein, 1989, Lemma 3.4.1.4] is the key that allows using the classic clockwise-counterclockwise scan without *backtracking* and thus allows merging two AVDs in linear time. However, an analogous statement of this lemma does not hold for Voronoi-like diagrams. A new algorithm for merging two Voronoi-like diagrams is needed. In Figure 7.4 we present an example showing that in some cases backtracking is needed if we follow the standard clockwise-counterclockwise technique of merging two abstract Voronoi diagrams as described by Klein [1989].

Figure 7.4 shows two boundary curves $\mathcal{P}_1$ (black) and $\mathcal{P}_2$ (blue) in thick solid lines together with their overlapping Voronoi-like diagrams $\mathcal{V}_l(\mathcal{P}_1)$ (gray) and $\mathcal{V}_l(\mathcal{P}_2)$ (turquoise), thin lines, respectively. The example is based on an admissible bisector system defined by disjoint line segments, which are shown in light gray, and the site s is dashed. The start of the merge curve (red) is m_1. In the classic merging method in each step i we move in $\mathcal{V}_l(\mathcal{P}_1)$ counterclockwise and in $\mathcal{V}_l(\mathcal{P}_2)$ clockwise along the boundaries of the current regions. The intersections m_i and m_i' of the bisector of the current merge curve edge is computed in both diagrams and the one being closer to the last merge curve vertex is the valid next vertex. In the example, in step 4 in order to compute the intersection m_4 one has to *backtrack* on $\mathcal{V}_l(\mathcal{P}_1)$ in *clockwise* direction from m_3'. This example shows that the next question can not be answered easily by using known results for abstract Voronoi diagrams from the literature.

Question 4. *Can two Voronoi-like diagrams $\mathcal{V}_l(\mathcal{P}_1)$ and $\mathcal{V}_l(\mathcal{P}_2)$ for $\mathcal{S}_1$ and $\mathcal{S}_2$, respectively be merged in $O(|\mathcal{V}_l(\mathcal{P}_1)| + |\mathcal{V}_l(\mathcal{P}_2)|)$ time into a Voronoi-like diagram for $\mathcal{S}_1 \cup \mathcal{S}_2$?*

We believe that Challenge C.1 and Challenge C.3 are very much related and the solution of one will help to solve also the other.

We conclude this chapter by classifying the following natural and fundamental problem as highly non-trivial to solve.

Question 5. *Does there exist a deterministic linear-time algorithm for updating an Abstract Voronoi Diagram after deletion of one site?*

The solutions of challenges C.1, C.2 and C.3, i.e., affirmative answers to Questions 2, 3 and 4 will bring significant progress towards its answer.

7.4 Conclusion

In this chapter we investigated the applicability of our Voronoi-like diagrams to the deterministic linear-time framework of Aggarwal et al. [1989]. We gave the idea how to apply them and pointed out three major challenges to overcome. For each of them, we explained the difficulty or gave concrete reasons why their solution will not be trivial.

We still believe that for future research it is worth to continue the investigation, since realizing this application would solve a very long-standing open problem: The design of a deterministic linear-time algorithm for tree-like Voronoi diagram of non point-sites, and in particular, the question: Can an abstract Voronoi region be deleted in linear time?

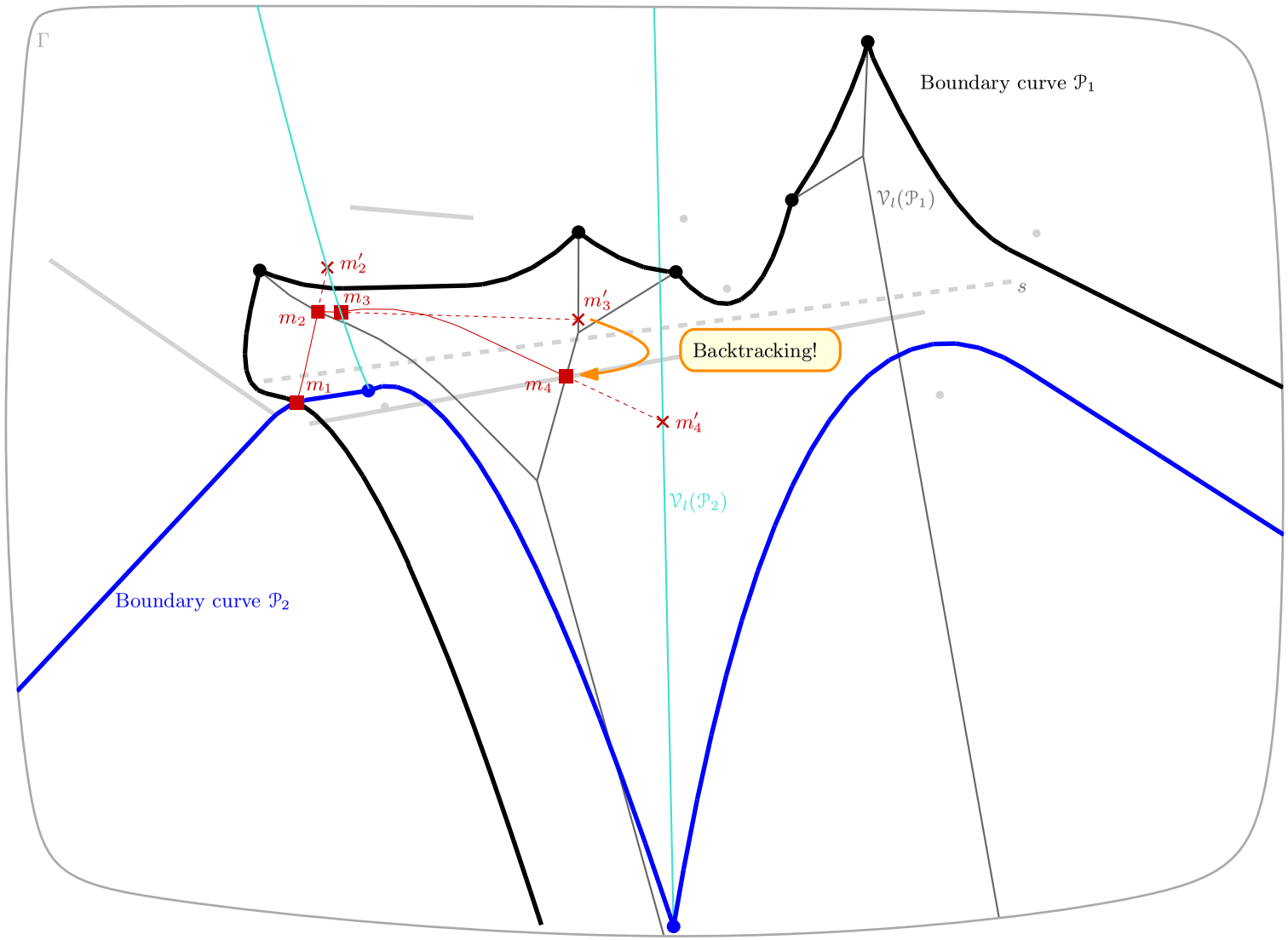

Figure 7.4. Two boundary curves $\mathcal{P}_1$ (black) and $\mathcal{P}_2$ (blue) are shown in thick solid lines together with their overlapping Voronoi-like diagrams $\mathcal{V}_l(\mathcal{P}_1)$ (gray) and $\mathcal{V}_l(\mathcal{P}_2)$ (turquoise), thin lines, respectively. In step 4 in order to compute the intersection m_4 one has to backtrack on $\mathcal{V}_l(\mathcal{P}_1)$ in clockwise direction from m_3'.

Chapter 8

Conclusion

In this dissertation we formalized the notion of a *Voronoi-like diagram*, given a *boundary curve* consisting of *arcs*, which represent the relevant sites and which are derived from the boundary of an (abstract) Voronoi region $\mathrm{VR}(s,S)$. The Voronoi-like diagram is a *relaxed* Voronoi structure and we proved that it is *well-defined*, and shown its usefulness by presenting algorithms, where the Voronoi-like diagram can be much faster computed than the real one. The main part of the research, which is summarized in this dissertation, was to formulate definitions and establish the properties of this structure.

Using Voronoi-like diagrams as intermediate structures, we derived a simple randomized algorithm to compute the subdivision $\mathcal{V}(S \setminus \{s\}) \cap \mathrm{VR}(s,S)$ in time $O(|\partial\mathrm{VR}(s,S)|)$, and thus, to *update* an abstract Voronoi diagram $\mathcal{V}(S)$, *after the deletion of site s*, in expected linear time. The Voronoi-like structure provides the means to efficiently deal with the *disconnected* Voronoi regions that are present in $\mathcal{V}(S \setminus \{s\}) \cap \mathrm{VR}(s,S)$ despite its *tree-like* simple structure. The approach iteratively computes Voronoi-like diagrams of boundary curves that are progressively enclosed within one another, until it reaches $\partial\mathrm{VR}(s,S)$.

We also extended the approach to compute the *order-*$(k{+}1)$ *subdivision* within a face of an order-k Voronoi region. Further, our approach can be adapted, in fact simplified, to compute the *farthest abstract Voronoi diagram* in linear expected time, after the order of its faces at infinity is known. In this case a boundary curve represents an ordering of Voronoi regions at infinity.

A *deterministic* linear-time algorithm for constructing tree-like abstract Voronoi diagrams remains an *open problem*. We expect that the notion of Voronoi-like diagrams will be useful in this direction as well. An interesting, however non-trivial, task for future research is to realize the application of the Voronoi-like structures introduced in this dissertation to the linear-time framework of Aggar-

wal et al. [1989], aiming to a deterministic linear-time algorithm. We pointed out challenges for this goal and formulated subproblems, the solutions of which would yield significant progress towards the linear-time algorithm.

Bibliography

Agarwal, P. K., Sharir, M. and Shor, P. [1989]. Sharp upper and lower bounds on the length of general davenport-schinzel sequences, *Journal of Combinatorial Theory, Series A* **52**(2): 228–274.

Aggarwal, A., Guibas, L. J., Saxe, J. B. and Shor, P. W. [1989]. A linear-time algorithm for computing the voronoi diagram of a convex polygon, *Discrete & Computational Geometry* **4**: 591–604.

Aurenhammer, F., Drysdale, R. and Krasser, H. [2006]. Farthest line segment Voronoi diagrams, *Inform. Process. Lett.* **100**: 220–225.

Aurenhammer, F., Klein, R. and Lee, D.-T. [2013]. *Voronoi Diagrams and Delaunay Triangulations*, World Scientific.

Barba, L. [2019]. Optimal Algorithm for Geodesic Farthest-Point Voronoi Diagrams, *in* G. Barequet and Y. Wang (eds), *35th International Symposium on Computational Geometry (SoCG 2019)*, Vol. 129 of *Leibniz International Proceedings in Informatics (LIPIcs)*, Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, Dagstuhl, Germany, pp. 12:1–12:14.
URL: *http://drops.dagstuhl.de/opus/volltexte/2019/10416*

Bohler, C., Cheilaris, P., Klein, R., Liu, C., Papadopoulou, E. and Zavershynskyi, M. [2015]. On the complexity of higher order abstract Voronoi diagrams, *Computational Geometry: Theory and Applications* **48**(8): 539–551.

Bohler, C., Klein, R., Lingas, A. and Liu, C.-H. [2018]. Forest-like abstract voronoi diagrams in linear time, *Computational Geometry* **68**: 134 – 145.
URL: *http://www.sciencedirect.com/science/article/pii/S0925772117300652*

Bohler, C., Klein, R. and Liu, C. [2014]. Forest-like abstract Voronoi diagrams in linear time, *Proc. 26th Canadian Conference on Computational Geometry (CCCG)*.

Bohler, C., Klein, R. and Liu, C.-H. [2016]. An Efficient Randomized Algorithm for Higher-Order Abstract Voronoi Diagrams, *32nd International Symposium on Computational Geometry (SoCG)*, Vol. 51 of *LIPIcs*, Dagstuhl, Germany, pp. 21:1–21:15.
URL: *http://drops.dagstuhl.de/opus/volltexte/2016/5913*

Bohler, C., Klein, R. and Liu, C.-H. [2019]. An efficient randomized algorithm for higher-order abstract voronoi diagrams, *Algorithmica* **81**(6): 2317–2345.

Bohler, C., Liu, C., Papadopoulou, E. and Zavershynskyi, M. [2016]. A randomized divide and conquer algorithm for higher-order abstract Voronoi diagrams, *Computational Geometry: Theory and Applications* **59**(C): 26–38.

Boissonnat, J.-D., Devillers, O., Schott, R., Teillaud, M. and Yvinec, M. [1992]. Applications of random sampling to on-line algorithms in computational geometry, *Discrete & Computational Geometry* **8**(1): 51–71.

Boissonnat, J.-D., Devillers, O. and Teillaud, M. [1990]. A randomized incremental algorithm for constructing higher order voronoi diagrams.", *Algorithmica* .

Boissonnat, J.-D. and Teillaud, M. [1993]. On the randomized construction of the delaunay tree, *Theoretical Computer Science* **112**(2): 339–354.

Boissonnat, J.-D. and Yvinec, M. [1998]. *Algorithmic geometry*, Cambridge university press.

Brévilliers, M., Chevallier, N. and Schmitt, D. [2007]. Triangulations of line segment sets in the plane, *International Conference on Foundations of Software Technology and Theoretical Computer Science*, Springer, pp. 388–399.

Buchin, K., Devillers, O., Mulzer, W., Schrijvers, O. and Shewchuk, J. [2013]. Vertex deletion for 3D Delaunay triangulations, *Algorithms – ESA 2013*, Vol. 8125 of *LNCS*, Springer Berlin Heidelberg, Berlin, Heidelberg, pp. 253–264.

CGAL [1995]. Computational geometry algorithms library.
URL: *http://www.cgal.org*

Chew, L. P. [1990]. Building Voronoi diagrams for convex polygons in linear expected time, *Technical report*, Dartmouth College, Hanover, USA.

Chew, L. P. and Kedem, K. [1989]. Placing the largest similar copy of a convex polygon among polygonal obstacles, *Proceedings of the fifth annual symposium on Computational geometry*, pp. 167–173.

Chin, F., Snoeyink, J. and Wang, C. A. [1999]. Finding the medial axis of a simple polygon in linear time, *Discrete & Computational Geometry* **21**(3): 405–420.

Clarkson, K. [1985]. A probabilistic algorithm for the post office problem, *Proceedings of the seventeenth annual ACM symposium on Theory of computing*, pp. 175–184.

Clarkson, K. L. [1987]. New applications of random sampling in computational geometry, *Discrete & Computational Geometry* **2**: 195–222.

Clarkson, K. L., Mehlhorn, K. and Seidel, R. [1993]. Four results on randomized incremental constructions, *Computational Geometry* **3**(4): 185–212.

Clarkson, K. L. and Shor, P. W. [1988]. Algorithms for diametral pairs and convex hulls that are optimal, randomized, and incremental, *Proceedings of the fourth annual symposium on Computational geometry*, pp. 12–17.

Clarkson, K. and Shor, P. [1989]. Applications of random sampling in computational geometry, II, *Discrete & Computational Geometry* **4**: 387–421.
URL: *http://dx.doi.org/10.1007/BF02187740*

Davenport, H. and Schinzel, A. [1965]. A combinatorial problem connected with differential equations, *American Journal of Mathematics* **87**(3): 684–694.

de Berg, M., Cheong, O., van Kreveld, M. and Overmars, M. [2008]. *Computational Geometry: Algorithms and Applications*, 3rd edn, Springer-Verlag.

de Berg, M., Schwarzkopf, O., van Kreveld, M. and Overmars, M. [2000]. *Computational Geometry: Algorithms and Applications*, 2nd edn, Springer-Verlag.

Devillers, O. [1996]. An introduction to randomization in computational geometry, *Theoretical computer science* **157**(1): 35–52.

Devillers, O. [2011]. Vertex removal in two-dimensional delaunay triangulation: Speed-up by low degrees optimization, *Computational Geometry* **44**(3): 169–177.

Felsner, S. [2019]. Personal communication.

Fortune, S. J. [1987]. A sweepline algorithm for Voronoi diagrams, *Algorithmica* **2**: 153–174.

Guibas, L. J., Knuth, D. E. and Sharir, M. [1992]. Randomized incremental construction of delaunay and voronoi diagrams, *Algorithmica* **7**(1-6): 381–413.

Hart, S. and Sharir, M. [1986]. Nonlinearity of davenport—schinzel sequences and of generalized path compression schemes, *Combinatorica* **6**(2): 151–177.

Junginger, K., Mantas, I. and Papadopoulou, E. [2019]. On selecting leaves with disjoint neighborhoods in embedded trees, *Conference on Algorithms and Discrete Applied Mathematics*, Springer, pp. 189–200.

Junginger, K. and Papadopoulou, E. [2018a]. Deletion in Abstract Voronoi Diagrams in Expected Linear Time, *34th International Symposium on Computational Geometry (SoCG 2018)*, Vol. 99 of *LIPIcs*, Dagstuhl, Germany, pp. 50:1–50:14.
URL: *http://drops.dagstuhl.de/opus/volltexte/2018/8763*

Junginger, K. and Papadopoulou, E. [2018b]. Deletion in abstract voronoi diagrams in expected linear time, *CoRR* **abs/1803.05372**.
URL: *http://arxiv.org/abs/1803.05372*

Kaplan, H., Mulzer, W., Roditty, L., Seiferth, P. and Sharir, M. [2017]. Dynamic planar voronoi diagrams for general distance functions and their algorithmic applications, *Proceedings of the Twenty-Eighth Annual ACM-SIAM Symposium on Discrete Algorithms*, SIAM, pp. 2495–2504.

Khramtcova, E. and Papadopoulou, E. [2015]. Linear-time algorithms for the farthest segment Voronoi diagram and related tree-structures, *Proc. 26th International Symposium on Algorithms and Computation (ISAAC)*, Vol. 9472 of *LNCS*.

Khramtcova, E. and Papadopoulou, E. [2017]. An expected linear-time algorithm for the farthest-segment Voronoi diagram, arXiv:1411.2816v3 [cs.CG]. Preliminary version in *Proc. 26th Int. Symp. on Algorithms and Computation (ISAAC), LNCS* 9472, 404-414, 2015.

Klein, R. [1989]. *Concrete and Abstract Voronoi Diagrams*, Vol. 400 of *Lecture Notes in Computer Science*, Springer-Verlag.

Klein, R., Langetepe, E. and Nilforoushan, Z. [2009]. Abstract Voronoi diagrams revisited, *Computational Geometry: Theory and Applications* **42**(9): 885–902.

Klein, R. and Lingas, A. [1994]. Hamiltonian abstract Voronoi diagrams in linear time, *Algorithms and Computation, 5th International Symposium, (ISAAC)*, Vol. 834 of *Lecture Notes in Computer Science*, pp. 11–19.

Klein, R., Mehlhorn, K. and Meiser, S. [1993]. Randomized incremental construction of abstract Voronoi diagrams, *Computational geometry: Theory and Applications* **3**: 157–184.

Lawson, C. L. [1977]. Software for c1 surface interpolation, *Mathematical software*, Elsevier, pp. 161–194.

Lee, D. T. [1982]. On k-nearest neighbor Voronoi diagrams in the plane, *IEEE Trans. Comput.* **C-31**(6): 478–487.

Levenshtein, V. [1992]. On perfect codes in deletion and insertion metric, *Discrete Mathematics and Applications* **2**(3): 241–258.

Mehlhorn, K., Meiser, S. and Rasch, R. [2001]. Furthest site abstract Voronoi diagrams, *International Journal of Computational Geometry and Applications* **11**(6): 583–616.

Mulmuley, K. [1994]. Computational geometry: an introduction through randomized algorithms.

Okabe, A., Boots, B., Sugihara, K. and Chiu, S. N. [2000]. *Spatial Tessellations: Concepts and Applications of Voronoi Diagrams*, second edn, John Wiley.

Papadopoulou, E. and Dey, S. K. [2013]. On the farthest line-segment Voronoi diagram, *International Journal of Computational Geometry and Applications* **23**(6): 443–459.

Preparata, F. P. and Shamos, M. I. [1985]. *Computational Geometry*, Texts and Monographs in Computer Science. Springer, New York, NY.

Seidel, R. [1990]. Linear programming and convex hulls made easy, *Proceedings of the sixth annual symposium on Computational geometry*, pp. 211–215.

Seidel, R. [1993]. Backwards analysis of randomized geometric algorithms, *Trends in Discrete and Computational Geometry, Algorithms and Combinatorics*, Vol. 10, Springer-Verlag, pp. 37–68.

Shamos, M. I. and Hoey, D. [1975]. Closest-point problems, *16th Annual Symposium on Foundations of Computer Science (sfcs 1975)*, IEEE, pp. 151–162.

Sharir, M. [2003]. The clarkson–shor technique revisited and extended, *Combinatorics, Probability and Computing* **12**(2): 191–201.

Sharir, M. and Agarwal, P. K. [1995]. *Davenport-Schinzel sequences and their geometric applications*, Cambridge university press.