On Implementing Multiplicatively Weighted Voronoi Diagrams*

Martin Held[1] and Stefan de Lorenzo[1]

1 **Universität Salzburg, FB Computerwissenschaften, Salzburg, Austria**
{held,slorenzo}@cs.sbg.ac.at

Abstract

We present a simple wavefront-like approach for computing multiplicatively weighted Voronoi diagrams of points and straight-line segments in the Euclidean plane. If the input sites may be assumed to be randomly weighted points then the use of a so-called overlay arrangement [Har-Peled&Raichel, Discrete Comput. Geom., 2015] allows to achieve an expected runtime complexity of $\mathcal{O}(n \log^4 n)$, while still maintaining the simplicity of our approach. We implemented the full algorithm for weighted points as input sites, based on CGAL. The results of an experimental evaluation of our implementation suggest $\mathcal{O}(n \log^2 n)$ as a practical bound on the runtime. Our algorithm can be extended to handle also additive weights in addition to multiplicative weights.

1 Introduction and Preliminaries

Aurenhammer and Edelsbrunner [1] present a worst-case optimal algorithm for constructing the multiplicatively weighted Voronoi diagram (MWVD) of a set of n points in $\mathcal{O}(n^2)$ time and space. Har-Peled and Raichel [2] show that a bound of $\mathcal{O}(n \log^2 n)$ holds on the expected combinatorial complexity if the weights of all points are chosen randomly. They also sketch how to compute MWVDs in expected time $\mathcal{O}(n \log^3 n)$, where linear-time triangulation and the algorithm by Aurenhammer and Edelsbrunner [1] are used as subroutines.

Let $S := \{s_1, s_2, \ldots, s_n\}$ denote a set of n distinct weighted points in $\mathbb{R}^2$ that are indexed such that $w(s_i) \leq w(s_j)$ for $1 \leq i < j \leq n$, where $w(s_i) \in \mathbb{R}^+$ is the weight associated with s_i. It is common to regard the weighted distance $d_w(p, s_i)$ from an arbitrary point p in $\mathbb{R}^2$ to s_i as the standard Euclidean distance $d(p, s_i)$ from p to s_i divided by the weight of s_i. The *(weighted) Voronoi region* $\mathcal{VR}_w(s_i, S)$ of s_i relative to S is the set of all points of the plane that are not farther to s_i than to any other site s_j in S. A connected component of a Voronoi region is called a *face*. For two distinct sites s_i and s_j of S, the *bisector* $b_{i,j}$ of s_i and s_j models the set of points of the plane that are at the same weighted distance from s_i and s_j. The MWVD $\mathcal{VD}_w(S)$ of S is the union of the boundaries of the individual Voronoi regions; see Figure 1. Following common terminology, a connected component of such a set is called a *(Voronoi) edge* of $\mathcal{VD}_w(S)$. An end-point of an edge is called a *(Voronoi) node*. It is known that the bisector between two unequally weighted sites forms a circle.

The wavefront $\mathcal{WF}(S, t)$ emanated by S at time $t \geq 0$ is the set of all points p of the plane whose minimal weighted distance from S equals t. The wavefront itself consists of several circular arcs which we call *wavefront arcs*. A common end-point of two consecutive wavefront arcs is called a *wavefront vertex*; see the blue dots in Figure 2. For $t \geq 0$, the *offset circle* $c_i(t)$ of the i-th site s_i is given by a circle centered at s_i with radius $t \cdot w(s_i)$. We specify a point of $c_i(t)$ relative to s_i by its polar angle α and its (weighted) polar radius t and denote it by $p_i(\alpha, t)$. Every pair of offset circles defines exactly two *(moving) vertices* $v^l_{i,j}$

* Work supported by Austrian Science Fund (FWF): Grant P31013-N31.

36th European Workshop on Computational Geometry, Würzburg, Germany, March 16–18, 2020.
This is an extended abstract of a presentation given at EuroCG'20. It has been made public for the benefit of the community and should be considered a preprint rather than a formally reviewed paper. Thus, this work is expected to appear eventually in more final form at a conference with formal proceedings and/or in a journal.

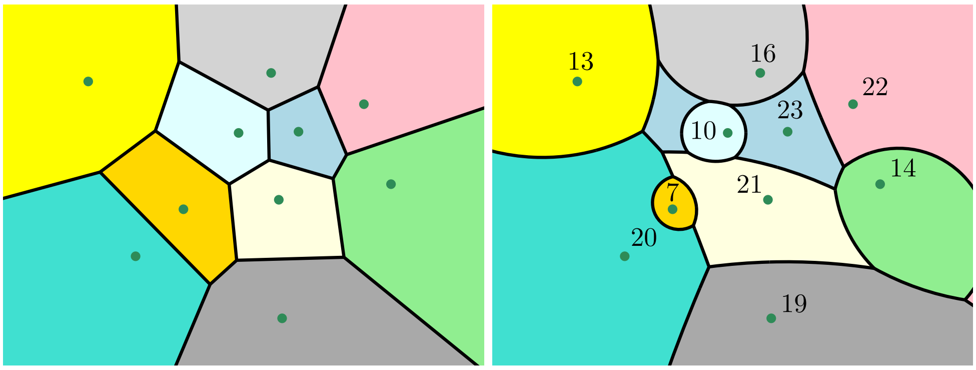


Figure 1 Left: Standard Voronoi diagram of a set of points (depicted by dark-green dots). Right: The numbers next to the points indicate their weights and the corresponding MWVD is shown.

and $v^r_{i,j}$ which can be interpreted as the traces of the intersections of the two offset circles over time. We refer to $v^l_{i,j}(t)$ as the *vertex married to* $v^r_{i,j}(t)$, and vice versa; see Figure 3.

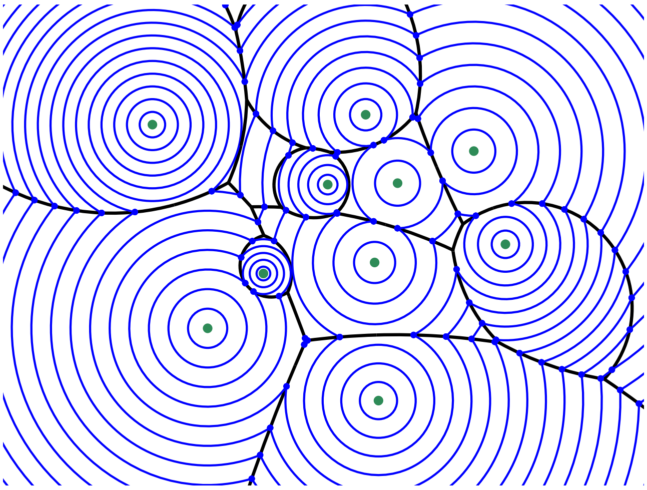


Figure 2 Wavefronts (in blue) for equally-spaced points in time for the input shown in Figure 1.

2 A Simple Event-Based Construction Scheme

In prior work [3], we introduced a wavefront-based strategy to compute $\mathcal{VD}_w(S)$ on the basis of which we made our improvements. Thus, we want to review its main ideas. For the sake of descriptional simplicity, we assume that no point in the plane has the same weighted distance to more than three elements of S. (This restriction can be waived.)

▶ **Definition 2.1** (Active point). A point p on the offset circle $c_i(t)$ is called *inactive* at time t (relative to S) if there exists $j > i$, with $1 \leq i < j \leq n$, such that p lies strictly inside of $c_j(t)$. Otherwise, p is *active* (relative to S) at time t. A vertex $v_{i,j}(t)$ is an *active vertex* if it is an active point on both $c_i(t)$ and $c_j(t)$ at time t; otherwise, it is an *inactive vertex*.

▶ **Lemma 2.2.** *If $p_i(\alpha, t)$ is inactive at time t then $p_i(\alpha, t')$ will be inactive for all $t' \geq t$.*

An inactive point $p_i(\alpha, t)$ cannot be part of the wavefront $\mathcal{WF}(S, t)$. Lemma 2.2 ensures that none of its future incarnations $p_i(\alpha, t')$ can become part of the wavefront $\mathcal{WF}(S, t')$.

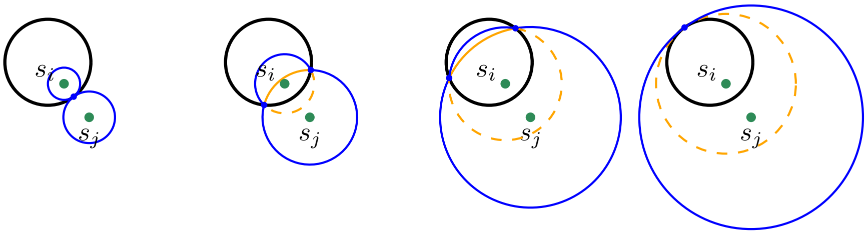


Figure 3 Two married vertices (highlighted by the blue dots) trace out the bisector (in black).

▶ **Definition 2.3** (Active arc). For $1 \leq i \leq n$ and $t \geq 0$, an *active arc* of the offset circle $c_i(t)$ at time t is a maximal connected set of points on $c_i(t)$ that are active at time t. The closure of a maximal connected set of inactive points of $c_i(t)$ forms an *inactive arc* of $c_i(t)$ at time t.

Every end-point of an active arc of $c_i(t)$ is given by the intersection of $c_i(t)$ with some other offset circle $c_j(t)$, i.e., by a moving vertex $v_{i,j}(t)$. This vertex is active, too. If $i < j$ then one active arc of $c_i(t)$ and two active arcs of $c_j(t)$ are incident to $v_{i,j}(t)$. The *arc arrangement (AA)* of S at time t, $\mathcal{A}(S,t)$, is the arrangement induced by all active arcs of all offset circles of S at time t; see Figure 4. As time t increases, the offset circles expand. This causes the vertices of $\mathcal{A}(S,t)$ to move, but it will also result in topological changes of the arc arrangement. Every such topological change can be classified as one of three event types.

▶ **Definition 2.4** (Collision event). Let $p_i(\alpha, t_{ij}^{min}) = p_j(\alpha+\pi, t_{ij}^{min})$ be the point of intersection of the offset circles of s_i and s_j at the collision time t_{ij}^{min}, for some fixed angle α. A *collision event* occurs between these two offset circles at time t_{ij}^{min} if the points $p_i(\alpha, t)$ and $p_j(\alpha+\pi, t)$ have been active for all times $0 \leq t \leq t_{ij}^{min}$.

▶ **Definition 2.5** (Domination event). Let $p_i(\alpha, t_{ij}^{max}) = p_j(\alpha, t_{ij}^{max})$ be the point of intersection of the offset circles of s_i and s_j at the domination time t_{ij}^{max}, for some fixed angle α. A *domination event* occurs between these two offset circles at time t_{ij}^{max} if the points $p_i(\alpha, t)$ and $p_j(\alpha, t)$ have been active for all times $0 \leq t \leq t_{ij}^{max}$.

▶ **Definition 2.6** (Arc event). An *arc event* e occurs at time t_e when an active arc shrinks to zero length as two unmarried vertices $v_{i,j}(t_e)$ and $v_{i,k}(t_e)$ meet in a point p_e on $c_i(t_e)$.

Domination events and arc events are easy to detect. The point and time of a collision is trivial to compute for any pair of offset circles, too. For the rest of this section we assume that all collisions among all pairs of offset circles are computed prior to the actual arc expansion. All events are stored in a priority queue $\mathcal{Q}$. If the maximum weight of all sites is associated with only one site then there will be a time t when the offset circle of this site dominates all other offset circles, i.e., when $\mathcal{WF}(S,t)$ contains only this offset circle as one active arc. Obviously, at this time no further event can occur and the arc expansion stops. If multiple sites have the same maximum weight then $\mathcal{Q}$ can only be empty once $\mathcal{WF}(S,t)$ contains only one loop of active arcs which all lie on offset circles of these sites and if all wavefront vertices move along rays to infinity.

During the arc expansion $\mathcal{O}(n^2)$ collision and domination events are computed. We know that collision events create and domination events remove active vertices (and make them inactive for good). A collapse of an entire active-arc triangle causes two vertices to become inactive. During every other arc event at least one active vertex becomes inactive, but at the same time one inactive vertex may become active again. In order to bound the number of arc events it is essential to determine how many vertices can be active and how often a vertex can undergo a *reactivation*, i.e., change its status from inactive to active.

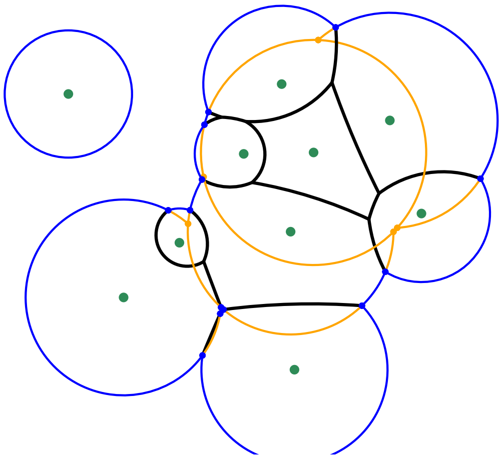


Figure 4 A snapshot of the arc expansion for the input shown in Figure 1. Active arcs that are currently not part of the wavefront are drawn in orange.

▶ **Lemma 2.7.** *Every reactivation of a moving vertex during an arc event forces another moving vertex to become inactive and remain inactive for the rest of the arc expansion.*

▶ **Lemma 2.8.** *Let h be the number of different vertices that ever were active during the arc expansion. Then $\mathcal{O}(h)$ arc events can take place during the arc expansion.*

Our naïve approach computes all potential collision events between all pairs of input sites as preprocessing. Thus, $h \in \Theta(n^2)$, and we get an overall runtime of $\mathcal{O}(n^2 \log n)$.

3 Reducing the Number of Collisions Computed

Experiments quickly indicate that the vast majority of pairwise collisions computed a priori does never end up on pairs of active arcs. Furthermore, the resulting Voronoi diagrams show a quadratic combinatorial complexity only for contrived input data.

This observation is backed by a result by Har-Peled and Raichel [2]: They show that the expected combinatorial complexity of $\mathcal{VD}_w(S)$ for a set S of n randomly weighted point sites is $\mathcal{O}(n \log^2 n)$. In order to keep our paper self-contained, we summarize their key principles.

- *Candidate set*: Consider an arbitrary (but fixed) point $q \in \mathbb{R}^2$, and let s be its nearest neighbor in S under the weighted distance. Let $s' \in S \setminus \{s\}$ be another site. Since s is the nearest neighbor of q we know that either s has a higher weight than s' or a smaller Euclidean distance to q than s'. Thus, one can define a *candidate set* for a weighted nearest neighbor of q which consists of all sites $s \in S$ such that all other sites in S either have a smaller weight or a larger Euclidean distance to q. Now assume that the sites are weighted randomly. Then Har-Peled and Raichel [2] show that this candidate set for q has a cardinality of $\mathcal{O}(\log n)$ with high probability.
- *Gerrymandering the plane*: The plane $\mathbb{R}^2$ is partitioned into a small number of regions such that the candidate set stays the same for all points within a region.
- *Randomized incremental campaigning*: Consider inserting the sites in order of decreasing weight: Then the i-th site is in the candidate set of a point $q \in \mathbb{R}^2$ if and only if it is the (unweighted) nearest neighbor of q among the first i sites. That is, if and only if q lies in the Voronoi region of the i-th site within the Voronoi diagram of the first i sites.

In more formal terms, we make use of the following results in order to determine all collision events among elements of S in near-linear expected time.

▶ **Lemma 3.1** (Har-Peled and Raichel [2]). *For all points $q \in \mathbb{R}^2$, the candidate set for q among S is of size $\mathcal{O}(\log n)$ with high probability.*

▶ **Lemma 3.2** (Har-Peled and Raichel [2]). *Let K_i denote the Voronoi cell of s_i in the unweighted Voronoi diagram of the i-th suffix $S_i := \{s_i, \ldots, s_n\}$. Let $\mathcal{OA}$ denote the arrangement formed by the overlay of the regions $K_1, \ldots, K_n$. Then, for every face f of $\mathcal{OA}$, the candidate set is the same for all points in f.*

Therefore, the overlay arrangement can be generated by incrementally constructing the unweighted Voronoi diagram of S in which the sites are inserted ordered by decreasing weights; see Figure 5. Kaplan et al. [4] prove that this overlay arrangement has an expected complexity of $\mathcal{O}(n \log n)$. Note that their result is applicable since inserting the points in sorted order of their randomly chosen weights corresponds to a randomized insertion. These results allow us to derive better complexity bounds.

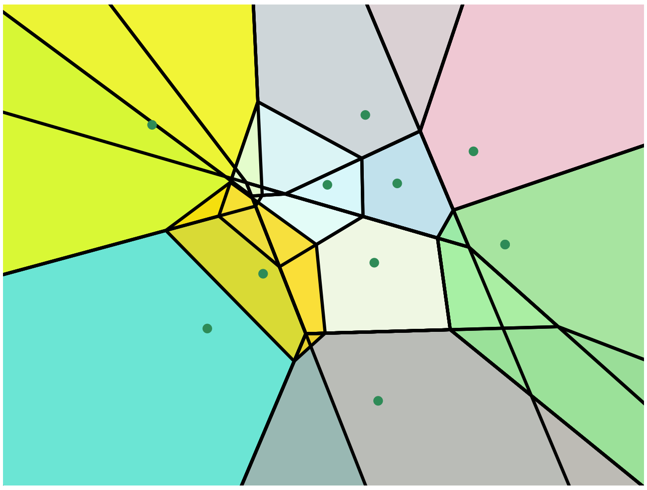


Figure 5 We insert the sites ordered by decreasing weights to generate $\mathcal{OA}$.

▶ **Lemma 3.3.** *If a collision event occurs between the offset circles of two sites $s_i, s_j \in S$ then there exists at least one candidate set which includes both s_i and s_j.*

▶ **Theorem 3.4.** *All collision events can be determined in $\mathcal{O}(n \log^3 n)$ expected time by computing the overlay arrangement $\mathcal{OA}$ of a set S of n input sites.*

Thus, the number h of vertices created during the arc expansion can be expected to be bounded by $\mathcal{O}(n \log^3 n)$. Theorem 2.8 tells us that the number of arc events is in $\mathcal{O}(h)$. Therefore, $\mathcal{O}(n \log^3 n)$ events happen in total.

▶ **Theorem 3.5.** *A wavefront-based approach allows to compute the multiplicatively weighted Voronoi diagram $\mathcal{VD}_w(S)$ of a set S of n (randomly) weighted point sites in expected $\mathcal{O}(n \log^4 n)$ time and expected $\mathcal{O}(n \log^3 n)$ space.*

4 Extensions

Consider a set S' of n disjoint weighted straight-line segments in $\mathbb{R}^2$. A wavefront propagation among weighted line segments requires us to refine our notion of "collision". We call an intersection of two offset circles a *non-piercing collision event* if it marks the initial contact of the two offset circles. That is, it occurs when the first pair of moving vertices appear. We call an intersection of two offset circles a *piercing collision event* if it takes place when two already intersecting offset circles intersect in a third point for the first time; see Figure 6. In this case, a second pair of moving vertices appear.

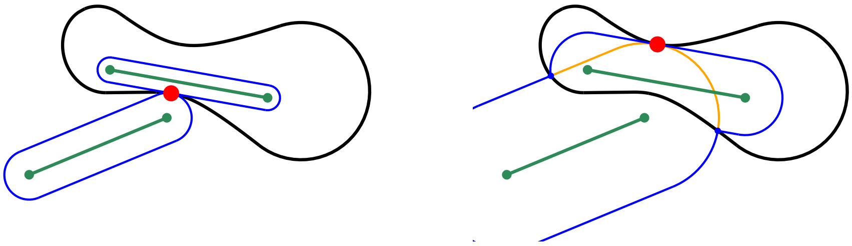


Figure 6 An example of a non-piercing (left) as well as a piercing collision event (right).

Hence, a minor modification of our event-based construction scheme is sufficient to extend it to weighted straight-line segments: We only need to check whether a piercing collision event that happens at a point p_e at time t_e currently is part of $\mathcal{WF}(S', t_e)$. In such a case the two new vertices as well as the corresponding active arc between them need to be flagged as part of $\mathcal{WF}(S', t_e)$. See Figure 7.

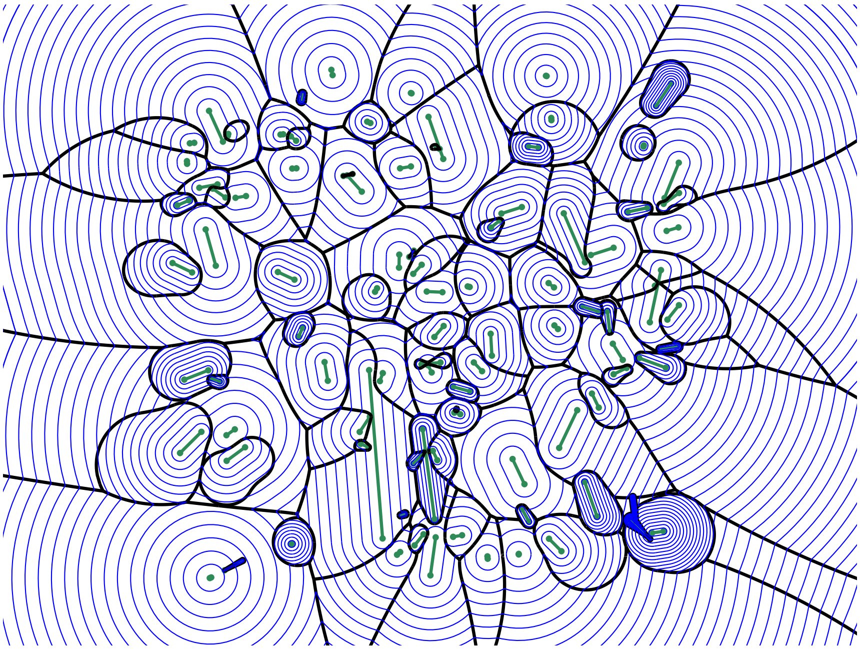
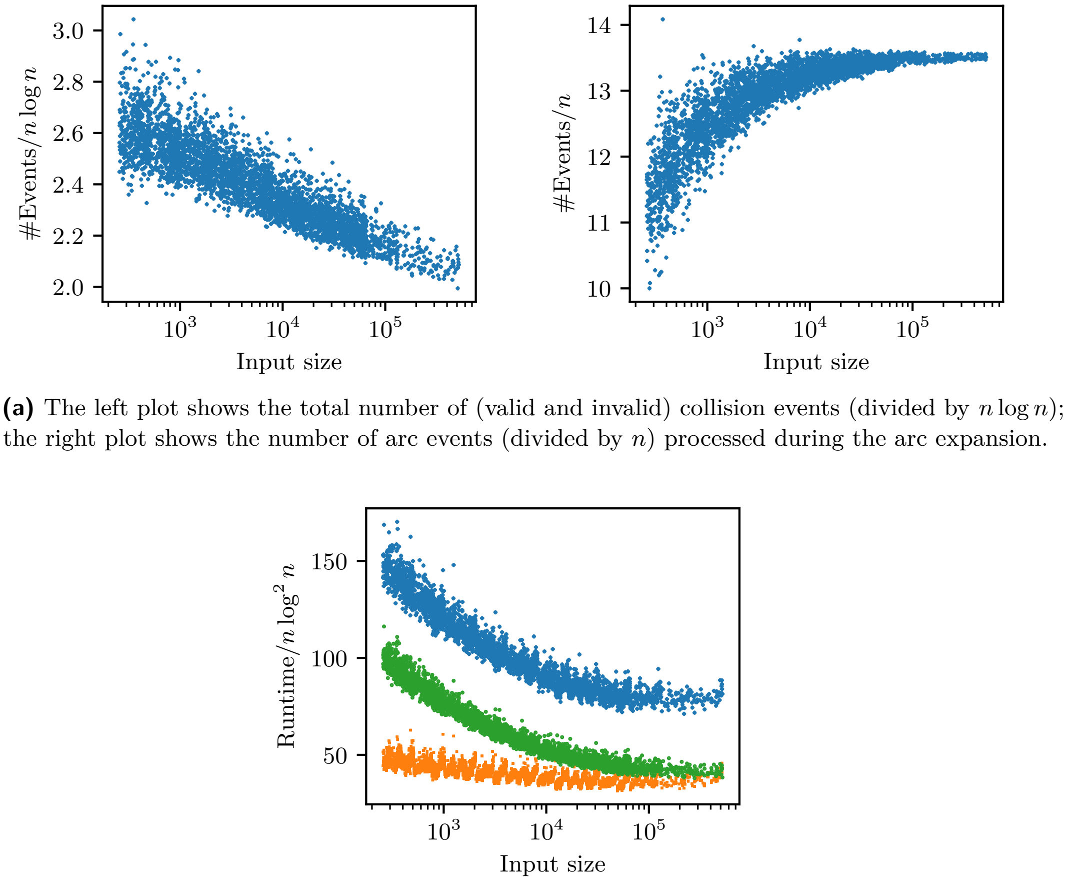


Figure 7 The MWVD of a set of weighted points and weighted straight-line segments together with a family of wavefronts for equally-spaced points in time.

An extension to additive weights can be integrated easily into our scheme by simply giving every offset circle a head-start of $w_a(s_i)$ at time $t = 0$, where $w_a(s_i) \geq 0$ denotes the real-valued additive weight that is associated with s_i.

5 Experimental Evaluation

We implemented our full algorithm for multiplicatively weighted points as input sites[1], based on the Computational Geometry Algorithms Library (CGAL) and exact arithmetic. In particular, we use CGAL's `Arrangement_2` package for computing the overlay arrangement and its `Voronoi_diagram_2` package for computing unweighted Voronoi diagrams. The computation of the MWVD itself utilizes CGAL's `Exact_circular_kernel_2` package.



(a) The left plot shows the total number of (valid and invalid) collision events (divided by $n \log n$); the right plot shows the number of arc events (divided by n) processed during the arc expansion.

(b) The orange plot shows the runtime consumed by the computation of the overlay arrangement. The green plot shows the time which it took to process all events, and the blue plot shows the overall runtime. All runtimes were divided by $n \log^2 n$.

Figure 8 Each marker on the x-axes indicates the number n of input sites for one out of over 3800 test cases. All weights and all point coordinates were chosen randomly.

We used our implementation for an experimental evaluation and ran our code on over 3800 inputs ranging from 256 vertices to 524 288 vertices. For all inputs all weights were chosen uniformly at random, and all point coordinates were chosen according to either a

[1] Our code is publicly available on GitHub under `https://github.com/cgalab/wevo`. We do also have a prototype implementation that handles both weighted points and weighted straight-line segments. It was used to generate the diagram shown in Figure 7.

uniform or a normal distribution. All tests were carried out with CGAL 4.11 on an Intel Xeon E5-2687W v4 processor clocked at 3.0 GHz. (We carried out our tests before CGAL 5.0 was released. Sample runs obtained with CGAL 5.0 indicate that all results would be the same for CGAL 5.0.) The numbers of collision events and arc events that occurred during the arc expansion are plotted in Figure 8a. Our tests suggest that we can expect at most $c \cdot n \log n$ collision events to occur, for some small constant c. (We had $c \leq 3$ in our tests.) Furthermore, we observed at most $14n$ arc events.

In any case, the number of events is smaller than predicted by the theoretical analysis. This is also reflected by our runtime statistics: In Figure 8b the runtime of the generation of the overlay arrangement, the time that was consumed by the computation of the MWVD, and the overall runtime are plotted. Summarizing, our tests suggest an average overall runtime of $\mathcal{O}(n \log^2 n)$ if all weights are chosen randomly.

References

1 Franz Aurenhammer and Herbert Edelsbrunner. An Optimal Algorithm for Constructing the Weighted Voronoi Diagram in the Plane. *Pattern Recogn.*, 17(2):251–257, 1984. doi:10.1016/0031-3203(84)90064-5.

2 Sariel Har-Peled and Benjamin Raichel. On the Complexity of Randomly Weighted Multiplicative Voronoi Diagrams. *Discrete Comput. Geom.*, 53(3):547–568, 2015. doi:10.1007/s00454-015-9675-0.

3 Martin Held and Stefan de Lorenzo. A Wavefront-Like Strategy for Computing Multiplicatively Weighted Voronoi Diagrams. In *Proceedings of the 35th European Workshop on Computational Geometry*, Utrecht, Netherlands, 2019.

4 Haim Kaplan, Edgar Ramos, and Micha Sharir. The Overlay of Minimization Diagrams in a Randomized Incremental Construction. *Discrete Comput. Geom.*, 45(3):371–382, 2011. doi:10.1007/s00454-010-9324-6.