An On-line Construction of Higher Order Voronoï Diagrams and its Randomized Analysis

(Abstract)

Jean-Daniel Boissonnat[†] **Olivier Devillers**[†] **Monique Teillaud**[†]

Key words : Computational geometry, On-line algorithms, Random sampling, d dimensional order k Voronoï diagram.

Previous work

The order k Voronoï diagram has been introduced in [SH75] in order to deal with k-closest points and related distance relationships. Lee [Lee82] gave the first algorithm for constructing the order k Voronoï diagram of a set of n points (or sites) in the plane. This algorithm constructs the order k Voronoï diagram from the order $(k-1)$ Voronoï diagram in time $O(kn \log n)$. Thus the order k Voronoï diagram (in fact, the family of all order j Voronoï diagrams for $1 \leq j \leq k$ — the order $\leq k$ Voronoï diagrams for short) can be constructed in time $O(k^2 n \log n)$. This bound can be tightened to $O(n \log n + k^2 n)$ using the result of [AGSS89].

Chazelle and Edelsbrunner [CE85] developed two versions of an algorithm that is better for large values of k. The first one takes $O(n^2 \log n + k(n-k)\log^2 n)$ time and $O(k(n-k))$ storage while the other takes $O(n^2 + k(n-k)\log^2 n)$ time and $O(n^2)$ storage.

A radically different approach, pioneered by Clarkson [Cla87], uses random sampling. Clarkson's algorithm determines the order k Voronoï diagram of n sites in the plane in time $O(kn^{1+\epsilon})$ with a constant factor that depends on ϵ.

More recently, in order to gain simplicity, several authors have designed algorithms which are incremental and randomized. Such an approach has been applied successfully for constructing Voronoï diagrams in the plane [CS89,GKS90,MMO90] and in d-space [BT89,Mul89]. A common point to all these randomized algorithms is that no distribution assumptions are made as it is the case, for example, in [Dwy89]. Hence the results remain valid for any set of points, provided that the points are inserted at random.

The algorithms in [CS89,MMO90,Mul89] are incremental in the sense that the points are introduced one at a time. But all the points need to be known in advance and maintained in an auxiliary data structure, the so called conflict graph.

The well known relationship between higher order Voronoï diagrams in d dimensions and arrangements of hyperplanes in $d+1$ dimensions can be used for the design of an algorithm that constructs the order $\leq n-1$ Voronoï diagrams in time and storage $O(n^{d+1})$, as shown by Edelsbrunner, O'Rourke and Seidel [EOS86].

In $d > 2$ dimensions, Clarkson [CS89] has shown that the size of the order $\leq k$ Voronoï diagrams is $O(k^{\lceil \frac{d+1}{2} \rceil} n^{\lfloor \frac{d+1}{2} \rfloor})$. Very recently, Mulmuley [Mul89,Mul] has obtained a randomized algorithm whose expected complexity meets this bound for $d > 2$ and whose complexity is $O(nk^2 + n \log n)$ for $d = 2$. This algorithm also uses a conflict graph.

[†]INRIA, 2004 Route des Lucioles, B.P.109, 06561 Valbonne cedex, France, E-mail : boissonn@alcor.inria.fr

* *This work has been supported in part by the ESPRIT Basic Research Action Nr. 3075 (ALCOM).*

Main result

None of the previous algorithms, except [BT89,GKS90] are on-line. If one new site is to be added, the Voronoï diagram has to be entirely reconstructed.

In this abstract, we present an algorithm that is *on-line*. After each insertion of a new site, the algorithm updates a data structure, called the k-Delaunay tree [BDT90]. This structure generalizes the Delaunay tree, introduced in [BT86,BT89] to compute the Delaunay triangulation (and, by duality, the Voronoï diagram) of a set of points. The k-Delaunay tree contains all the successive versions of the order $\leq k$ Voronoï diagrams and allows fast point location.

As any dynamic algorithm constructing the Voronoï diagram, ours cannot be very good in the worst-case. However, a randomized analysis shows that it is very efficient on the average. The analysis of our algorithm has some similarity with the ones in [BT89,CS89,GKS90]. Our main result is stated in the following theorem :

Theorem : *The k-Delaunay tree (and thus the order $\leq k$ Voronoï diagrams) of n sites in the plane* (resp. *in d-space) can be on-line constructed using $O(k^3 n)$* (resp. $O\left(k^{d+1} n^{\lfloor \frac{d+1}{2} \rfloor}\right)$) *expected storage and with $O(k^4 \log n)$* (resp. $O\left(k^{d+2} n^{\lfloor \frac{d+1}{2} \rfloor - 1} \log k\right)$) *expected update time.*

The analysis of the algorithm is randomized. Our results hold when averaging over all possible permutations of the set of inserted points. An important point about our analysis is that it is not amortized. These results are asymptotically optimal for fixed d and k. These bounds are not as good as those of Mulmuley : the increase in the exponent of k is the price to be paid to have an on-line algorithm.

An important point is that these results hold whatever the point distribution may be. The algorithm is simple and, moreover, the numerical computations involved are also quite simple : they consist mostly of comparisons of (squared) distances in order to check if a point lies inside or outside a ball.

Experimental results, for uniform as well as degenerate distributions of points, have provided strong evidence that this algorithm is very effective in practice, for small values of k.

For large values of k, a similar structure, based on the order k furthest neighbours Voronoï diagrams could be derived. It provides results similar to the ones above to construct all order $\geq n - k$ Voronoï diagrams and to find l furthest neighbours for $l \leq k$.

The k-Delaunay tree can also be generalized to construct order $\leq k$ Voronoï diagrams of line segments. Results will be reported in a forthcoming paper [BDS*90].

Overview of the algorithm

We denote by $Vor_k(\mathcal{S})$ the order k Voronoï diagram of a set $\mathcal{S}$ of n sites in the plane. Let T be a triangle whose vertices are sites of $\mathcal{S}$ and let $B(T)$ denote the *open* disk circumscribing triangle T.

A fundamental property of order k Voronoï diagrams is that T corresponds (is *dual*) to a vertex of both $Vor_{k+1}(\mathcal{S})$ and $Vor_{k+2}(\mathcal{S})$ if $B(T)$ contains k sites.

Reciprocally, a vertex of $Vor_k(\mathcal{S})$ is dual to a triangle whose circumscribing disk contains $k-1$ or $k-2$ sites in its interior.

Including and excluding neighbours

For a triangle T, we will define 2 neighbours through each of its 3 edges : one will be called the *including* neighbour and the other one the *excluding* neighbour. This notion of neighbourhood corresponds to the actual notion of adjacency in the higher order Voronoï diagrams.

Let E be an edge of T and p be the third vertex of T. Let us consider a moving disk B whose boundary passes through the end points of E, and whose center moves along the bisecting line of E. Starting from $B = B(T)$, we can move B in two opposite directions : the one such that $p \in B$ is called the including direction and the other such that $p \notin B$ is called the excluding direction. We stop moving B as soon as its boundary encounters a site different from the end points of E. Let q_i (*resp.* q_e) be the first site encountered in the including (*resp.* excluding) direction. The triangle T_i (*resp.* T_e) having E as an edge and q_i (*resp.* q_e) as a vertex will be called the *including neighbour* (*resp. excluding neighbour*) of T through edge E (see Figure 1). Notice that q_i and q_e may be on either side of E.

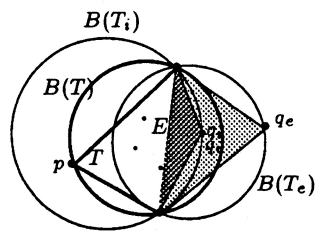

Figure 1: Including and excluding neighbours

Remark : The following property will be useful in the sequel : $B(T) \subset B(T_i) \cup B(T_e)$. Hence, if a site m lies into $B(T)$, we can deduce that m lies into either $B(T_i)$ or $B(T_e)$.

Construction of the k-Delaunay tree

Our algorithm is based on the well known incremental algorithm of [GS78] for constructing the Delaunay triangulation. Each site is introduced one after another in each of the order $\leq k$ Voronoï diagrams and each diagram is subsequently updated. We will describe this algorithm at the same time as the k-Delaunay tree, but it is actually independent of that data structure.

The k-Delaunay tree is not really a tree but a rooted direct acyclic graph. The nodes are associated to triangles. We will use the same word *triangle* for both a triangle and its associated node.

Following our algorithm, each site is introduced one after another and we keep all triangles in a hierarchical manner in the k-Delaunay tree, creating appropriate links between "old" (i.e. created before the introduction of the site) and "new" triangles (i.e. created after the introduction of the site). This will allow to efficiently locate a new site in the current structure.

For the initialization step we choose 3 sites. They generate one finite triangle and six half planes (considered as infinite triangles) limited by the supporting lines of the finite triangle. These 7 triangles will be the sons of the root of the tree.

We define the *current width* of a triangle T dual to a vertex of some higher order Voronoï diagram to be the number of already inserted sites lying inside $B(T)$.

The k-Delaunay tree will be constructed so that it satisfies the following property :

($\mathcal{P}$) all the triangles of current width strictly less than k are present in the k-Delaunay tree.

Hence, the triangles dual to the vertices of the order $\leq k$ Voronoï diagrams are all present in the structure. Moreover, we keep their including and excluding neighbours (or equivalently their adjacency relationships in the corresponding Voronoï diagrams). The k-Delaunay tree thus contains the whole information necessary to construct all the order $\leq k$ Voronoï diagrams.

When a new site m is to be inserted, we traverse the k-Delaunay tree to find all the triangles whose balls contain m and we update the k-Delaunay tree in such a way that property ($\mathcal{P}$) is preserved.

References

[AGSS89] A. Aggarwal, L.J. Guibas, J. Saxe, and P.W. Shor. A linear time algorithm for computing the Voronoï diagram of a convex polygon. *Discrete and Computational Geometry*, 4:591–604, 1989.

[BDS*90] J.D. Boissonnat, O. Devillers, R. Schott, M. Teillaud, and M. Yvinec. *Applications of Random Sampling to On-line Algorithms in Computational Geometry.* Technical Report, Institut National de Recherche en Informatique et Automatique, (France), 1990. To be published.

[BDT90] J.D. Boissonnat, O. Devillers, and M. Teillaud. A randomized incremental algorithm for constructing higher order Voronoï diagrams. In *Second Canadian Conference on Computational Geometry in Ottawa*, August 1990. Full paper available as Technical Report INRIA 1207.

[BT86] J.D. Boissonnat and M. Teillaud. A hierarchical representation of objects: the Delaunay Tree. In *Second ACM Symposium on Computational Geometry in Yorktown Heights*, June 1986.

[BT89] J.D. Boissonnat and M. Teillaud. *On the Randomized Construction of the Delaunay Tree.* Technical Report 1140, Institut National de Recherche en Informatique et Automatique, (France), December 1989.

[CE85] B. Chazelle and H. Edelsbrunner. An improved algorithm for constructing k^{th}-order Voronoï diagrams. In *First ACM Symposium on Computational Geometry in Baltimore*, pages 228–234, June 1985.

[Cla87] K.L. Clarkson. New applications of random sampling in computational geometry. *Discrete and Computational Geometry*, 2:195–222, 1987.

[CS89] K.L. Clarkson and P.W. Shor. Applications of random sampling in computational geometry, II. *Discrete and Computational Geometry*, 4(5), 1989.

[Dwy89] R.A. Dwyer. Higher-dimensional Voronoï diagrams in linear expected time. In *5th ACM Symposium on Computational Geometry in Saarbruchen*, June 1989.

[EOS86] H. Edelsbrunner, J. O'Rourke, and R. Seidel. Constructing arrangements of lines and hyperplanes with applications. *SIAM Journal on Computing*, 15:341–363, 1986.

[GKS90] L.J. Guibas, D.E. Knuth, and M. Sharir. Randomized incremental construction of Delaunay and Voronoï diagrams. In *ICALP*, July 1990.

[GS78] P.J. Green and R. Sibson. Computing Dirichlet tesselations in the plane. *The Computer Journal*, 21, 1978.

[Lee82] D.T. Lee. On k-nearest neighbor Voronoï diagrams in the plane. *IEEE Transactions on Computers*, C-31:478–487, 1982.

[MMO90] K. Mehlhorn, S. Meiser, and C. Ó'Dúnlaing. On the construction of abstract Voronoï diagrams. In C. Choffrut and T. Lengauer, editors, *STACS 90*, pages 227–239, Springer-Verlag, 1990.

[Mul] K. Mulmuley. On levels in arrangements and Voronoï diagrams. *Discrete and Computational Geometry.* To be published.

[Mul89] K. Mulmuley. On obstruction in relation to a fixed viewpoint. In *IEEE Symposium on Foundations of Computer Science*, pages 592–597, 1989.

[SH75] M.I. Shamos and D. Hoey. Closest-point problems. In *IEEE Symposium on Foundations of Computer Science*, pages 151–162, October 1975.