Linear-Time Delaunay Triangulations Simplified[*]

Kevin Buchin[†] Wolfgang Mulzer[‡]

Abstract

Recently it was shown that — under reasonable assumptions — Voronoi diagrams and Delaunay triangulations of planar point sets can be computed in time $o(n \log n)$, beating the classical comparison-based lower bound. A number of increasingly faster randomized algorithms have been proposed, most recently a linear-time algorithm based on a randomized incremental construction that uses a combination of nearest neighbor graphs and the history structure to speed up point location. We present a simpler variant of this approach relying only on nearest neighbor graphs. The algorithm and its analysis generalize to higher dimensions, with an expected performance that is proportional to the structural change of the randomized incremental construction. As a by-product, we analyze an interesting class of insertion orders for randomized incremental constructions.

1 Introduction

A classical lower bound asserts that it takes $\Omega(n \log n)$ time to construct the Delaunay triangulation (DT) of a planar point set in the algebraic decision tree model. However, three years ago Chan and Pătraşcu [5] showed that this lower bound can be broken on a word RAM, giving a randomized algorithm that runs in $O(n \log n / \log \log n)$ time, which they later improved to $n2^{O(\sqrt{\log \log n})}$ [6]. More recently, a randomized linear-time algorithm in a different model was proposed [2]. In this paper, we show how to simplify this approach by using a variant of a randomized incremental construction (RIC) con BRIO.

Our main tool is a reduction from nearest neighbor graphs (NNGs) to Delaunay triangulations: we show that if the NNG for an m-point set can be computed in time $f(m)$, then the Delaunay triangulation for an n-point set P can be computed in time $O(C(n)+f(n))$, where $C(n)$ is the expected structural change for a RIC. In a recent paper, Chan [4] explains a linear-time reduction from quadtrees to NNGs and describes two settings in which quadtrees can be constructed quickly. The first assumes a real-RAM with a constant-time *restricted* floor-function that can be applied only if the resulting integer has $O(\log n)$ bits. Thus, we avoid issues about creating an unreasonably powerful model of computation. Furthermore, we assume that the input P has polynomially bounded *spread* (defined below). Then, the NNG of P, $\mathrm{N}(P)$, can be computed in linear time [4]. In the second setting, we have a word RAM which can compute the *shuffle* of a point in constant time (if the coordinates of p are $(p_{1w} \cdots p_{11}, p_{2w} \cdots p_{21}, \dots, p_{dw} \cdots p_{d1})$, the shuffle is $p_{1w}p_{2w} \cdots p_{dw} \cdots p_{d1}$). This is a reasonable assumption, since the shuffle operation is in AC^0. Here, a NNG can be found in the time needed for integer sorting. We believe that current integer sorting algorithms can be adapted to the shuffle order and that therefore the assumption can be dropped.

Our reduction follows the RIC paradigm, choosing a random permutation of the input and inserting the points into the DT one by one. However, unlike the previous algorithm [2], the choice is not uniform, but we use a sampling strategy that can be seen as a generalization of a biased randomized insertion order (BRIO) [1]. Compared to the classical approach, RIC con BRIO achieves an improved locality of reference as follows: take a *gradation* $P = S_0 \supseteq S_1 \supseteq \cdots \supseteq S_\ell$, where $|S_\ell| = O(1)$ and S_{i+1} is obtained from S_i by sampling each point independently with probability 1/2. When performing the RIC, first insert all points in S_ℓ, then $S_{\ell-1}$, then $S_{\ell-2}$, and so on. The order within the sets S_i is arbitrary; eg, it could be optimized for the underlying hardware. This procedure runs in expected time proportional to what classical RICs achieve [1, 3].

Owing to the flexibility of RICs, our algorithm works in any dimension. For the planar case, we can give a very simple analysis for the reduction. However, this analysis crucially uses the fact that the worst-case complexity of planar DTs is linear. Of course, this is no longer true in higher dimensions, but even then, "typical" point sets often have linear-size DTs. Even stronger, it is not uncommon to encounter inputs in which the size of the DT for a random subset is linear in the subset. For example, this happens for points that are distributed uniformly in a d-dimensional ball. For such point sets the structural change of classical RICs and RICs con BRIO is linear (but they lose a logarithmic factor for updating the

[*]The first author was supported by the Netherlands' Organisation for Scientific Research (NWO) under BRICKS/FOCUS project no. 642.065.503. The second author was supported in part by NSF grant CCF-0634958 and NSF CCF 0832797.

[†]Dept. of Information and Computing Sciences, Utrecht University, Netherlands, buchin@cs.uu.nl

[‡]Department of Computer Science, Princeton University, wmulzer@cs.princeton.edu

conflict information). As for many random distributions, the expected spread is polynomially bounded, which suffices to compute the NNG in linear time in our first model of computation. In a more refined analysis, we will show that the expected running time of our reduction is proportional to the expected structural change of the RIC plus the time for computing the NNG. Thus, on typical inputs we again achieve an improvement in running time over classical RICs.

2 Algorithm & First Analysis

We say that an n-point set $P \subseteq \mathbb{R}^d$ has polynomially bounded *spread*, if the ratio between the maximum and minimum distance between any two points in P is $O(n^c)$, for some constant c. NNGs for point sets with bounded spread or for points with integer coordinates can be computed quickly, as was shown by Chan [4].

Theorem 1 (Chan [4]) *Let $P \subseteq \mathbb{R}^d$ be an n-point set. If P has polynomially bounded spread, $\mathrm{N}(P)$ can be computed in time $O(n)$ on a real-RAM with the restricted floor function. If the coordinates of the points are b-bit integers and we assume a word RAM with the shuffle operation, then $\mathrm{N}(P)$ can be found in time $O(\mathrm{sort}(n))$, where $\mathrm{sort}(n)$ is the time for sorting db-bit integers.*

We now describe our algorithm `BrioDC`. Suppose that NNGs can be computed in $f(m)$ time, such that $f(m)/m$ is increasing. By Theorem 1, we have $f(n) = O(n)$ or $f(n) = O(\mathrm{sort}(n))$, depending on the model. Given a point set P, we first compute $\mathrm{N}(P)$ in $O(f(n))$ time. Next we compute a sample $S \subseteq P$ such that S meets every connected component of $\mathrm{N}(P)$ and such that S contains every point in P with probability 1/2. This is done as follows: we define a (partial) matching $\mathcal{M}(P)$ on P by pairing up two arbitrary points in each component of $\mathrm{N}(P)$. The sample S is obtained by picking one random point from each pair in $\mathcal{M}(P)$ and sampling the points in $P \setminus \mathcal{M}(P)$ independently with probability 1/2 (although they could also be paired up). We recursively compute the Delaunay triangulation $\mathrm{DT}(S)$ of S. To locate $P \setminus S$ in $\mathrm{DT}(S)$, we traverse the components of $\mathrm{N}(P)$ starting in each component at a point of S and inserting points while we walk through the Delaunay triangulation along the edges of $\mathrm{N}(P)$. Since $f(m)/m$ is increasing, it takes $O(f(n))$ time to compute the NNGs for all the samples.

Theorem 2 *Let P be a planar n-point set.* `BrioDC` *computes $\mathrm{DT}(P)$ in expected time $O(f(n)+n)$, where $f(m)$ is the time to compute a m-point NNG and we assume that $f(m)/m$ is increasing.*

Proof. The cost of a straight-line walk along an edge pq of $\mathrm{N}(P)$ with p in the current DT and q to be inserted next can be split into the cost of finding the triangle at p in which the walk starts and the cost of the actual traversal. The first part of the cost can be bounded by the number of triangles at p in the current DT. Summing over all walking steps, any triangle is counted for each of its vertices at most as often as the degree of this vertex in $\mathrm{N}(P)$, which is bounded. The second part of the cost, ie, the cost of the traversal, can be bounded by the structural change since all triangles traversed are destroyed when the next point is inserted.

The structural change can be bounded by comparing it to the structural change of the last round of a RIC con BRIO. Let p_s be the probability that a given triangle with s conflicts appears in the last round of such a construction. We have $p_s = c/2^s$ for a suitable constant c. The probability p'_s of this triangle appearing in `BrioDC` while constructing $\mathrm{DT}(P)$ from $\mathrm{DT}(S)$ is also bounded by $1/2^s$: either the stoppers of the triangle are sampled independently of each other (then we directly get this bound), or not (then S includes a stopper and the simplex cannot occur). Thus, the expected structural change is asymptotically as for RIC con BRIO and therefore linear [1, 3]. Now, the expected size of S is $|P|/2$, and we can apply the argument above to the construction of $DT(S)$, and so on. Overall this yields the desired running time. □

3 Dependent Insertion Orders & Second Analysis

Theorem 2 does not apply to higher dimensions, since it requires that the *worst-case* complexity of a planar DT is linear. We now make the analysis sensitive to the expected structural change of a RIC.

Theorem 3 *Let P be a d-dimensional n-point set. The expected running time of* `BrioDC` *is $O(C(P) + f(n))$, where $C(P)$ denotes the expected structural change incurred by a RIC on P and f is as in Theorem 2. The constant in the O-notation depends exponentially on d.*

Let $P = S_0 \supseteq \cdots \supseteq S_\ell$ be the sequence of samples taken by `BrioDC`. Fix a set $\mathbf{u}$ of $d+1$ distinct points in P. Let Δ be the simplex spanned by $\mathbf{u}$, and let $L_\mathbf{u} \subseteq P$ denote the points in the circumsphere of Δ. The set $\mathbf{u}$ is the *trigger* set, $L_\mathbf{u}$ the *stopper* set for Δ. Consider the event A_α that Δ occurs in one of the DTs in the construction of $\mathrm{DT}(S_\alpha)$ from $\mathrm{DT}(S_{\alpha+1})$, for some α. Clearly, A_α can only happen if $\mathbf{u} \subseteq S_\alpha$ and $L_\mathbf{u} \cap S_{\alpha+1} = \emptyset$.

We will prove Theorem 3 using Lemma 4 which in turn follows from Proposition 6.

Lemma 4 *We have*

$$\Pr[A_\alpha] \leq e^{2d+2}\, 2^{-(d+1)\alpha} \left(1 - 2^{-\alpha-1}\right)^{|L_\mathbf{u}|}.$$

Let us first give some intuition: we think of the sampling as an $(\alpha+1)$-step random walk on a path with $|L_{\mathbf{u}}|$ nodes: a random step represents a new sample, and each node represents the current number of stoppers. The goal is to upperbound the probability of reaching state 0 while retaining all triggers. This model is overly simplistic: the random choices in a step not only depend on the number of stoppers in the current sample S, but also on the matching $\mathcal{M}(S)$. Even worse, the probability distribution in the current state may depend on past states. However, we will show how to avoid these issues through appropriate conditioning, and that the random walk essentially behaves like a Markov process that in each round eliminates $d+1$ stoppers and samples the remaining stoppers independently. The elimination is due to trigger-stopper pairs, since we assume that all triggers survive. The remaining stoppers are not necessarily independent, but matching two stoppers guarantees that one of them survives, which can only help. Eliminating $d+1$ stoppers in the ith round has a similar effect as starting with about $(d+1)2^i$ fewer stoppers: though a given trigger can be matched with only one stopper per round, these parings can vary for different instances of the walk, and since a given stopper survives a round with probability roughly $1/2$, the "amount" of stoppers eliminated by one trigger in all instances roughly doubles per round.

Proof. (of Lemma 4) For a sample $S \subseteq P$, we define the *matching profile* for S as the triple (a,b,c) that counts the number of trigger-stopper, stopper-stopper, and trigger-trigger pairs in $\mathcal{M}(S)$. In order to bound $\Pr[A_\alpha]$, we consider

$$p_{s,k} := \max_{\mathcal{P}_k} \Pr[A_\alpha \mid X_{s,k}, \mathcal{P}_k], \tag{1}$$

where we define

$$X_{s,k} := \{\mathbf{u} \subseteq S_{\alpha-k}\} \cap \{|L_{\mathbf{u}} \cap S_{\alpha-k}| = s\},$$

ie, the event that the sample $S_{\alpha-k}$ contains all the triggers and exactly s stoppers. The maximum in (1) is taken over all possible sequences $\mathcal{P}_k = \mathbf{m}_0, \ldots, \mathbf{m}_{\alpha-k-1}, Y_0, \ldots, Y_{\alpha-k-1}$ of matching profiles $\mathbf{m}_i$ for S_i and events Y_i of the form $X_{t_i,\alpha-i}$ for some t_i. Since $\Pr[A_\alpha] = p_{|L_{\mathbf{u}}|,\alpha}$, it suffices to upperbound $p_{s,k}$. We define a recursion for $p_{s,k}$. To do that, let

$$T_k := \{\mathbf{u} \subseteq S_{\alpha-k}\},$$

ie, the event that $S_{\alpha-k}$ contains all the triggers, and let

$$U_{k,i} := \{|L_{\mathbf{u}} \cap S_{\alpha-k}| = i\},$$

denote the event that $S_{\alpha-k}$ contains exactly i stoppers.

Proposition 5 *We have*

$$p_{s,k} \le \max_{\boldsymbol{m}} \Pr[T_{k-1} \mid X_{s,k}, \boldsymbol{m}] \cdot \sum_{i=0}^{s} p_{i,k-1} \Pr[U_{k-1,i} \mid T_{k-1}, X_{s,k}, \boldsymbol{m}],$$

where the maximum is over all possible matching profiles $\boldsymbol{m} = (a,b,c)$ for $S_{\alpha-k}$.

Proof. Fix a sequence $\mathcal{P}_k$ as in (1). Then, by distinguishing how many stoppers are present in $S_{\alpha-k+1}$,

$$\Pr[A_\alpha \mid X_{s,k}, \mathcal{P}_k] = \sum_{i=0}^{s} \Pr[X_{i,k-1} \mid X_{s,k}, \mathcal{P}_k] \Pr[A_\alpha \mid X_{i,k-1}, X_{s,k}, \mathcal{P}_k].$$

Now if we condition on a matching profile $\mathbf{m}$ for $S_{\alpha-k}$, we get

$$\Pr[X_{i,k-1} \mid X_{s,k}, \mathcal{P}_k, \mathbf{m}] = \Pr[T_{k-1} \mid X_{s,k}, \mathbf{m}] \Pr[U_{k-1,i} \mid T_{k-1}, X_{s,k}, \mathbf{m}],$$

since the distribution of triggers and stoppers in $S_{\alpha-k+1}$ becomes independent of $\mathcal{P}_k$ once we know the matching profile and the number of triggers and stoppers in $S_{\alpha-k}$. Furthermore,

$$\Pr[A_\alpha \mid X_{i,k-1}, \mathbf{m}, X_{s,k}, \mathcal{P}_k] \le \max_{\mathcal{P}_{k+1}} \Pr[A_\alpha \mid X_{i,k-1}, \mathcal{P}_{k+1}] = p_{i,k-1}.$$

The claim follows by taking the maximum over $\mathbf{m}$. $\square$

We use Proposition 5 to bound $p_{s,k}$: if $\mathbf{m} = (a,b,c)$ pairs up two triggers, we get $\mathbf{u} \not\subseteq S_{\alpha-k+1}$ and $\Pr[T_{k-1} \mid X_{s,k}, \mathbf{m}] = 0$. Hence we can assume $c = 0$ and therefore $\Pr[T_{k-1} \mid X_{s,k}, \mathbf{m}] = 1/2^{d+1}$, since all triggers are sampled independently. Furthermore, we know that none of the a stoppers paired with a trigger and half of the $2b$ stoppers paired with a stopper end up in $S_{\alpha-k+1}$, while the remaining $t_{\mathbf{m}} := s - a - 2b$ stoppers are sampled independently. Thus, Proposition 5 gives

$$p_{s,k} \le \max_{\substack{\mathbf{m} \\ c=0}} \sum_{i=b}^{s-a-b} \frac{p_{i,k-1}}{2^{d+1}} \Pr\left[B_{1/2}^{t_{\mathbf{m}}} = i - b\right], \tag{2}$$

where $B_{1/2}^{t_{\mathbf{m}}}$ denotes a binomial distribution with $t_{\mathbf{m}}$ trials and success probability $1/2$.

Proposition 6 *We have*

$$p_{s,k} \le 2^{-(d+1)k} \left(1 - 2^{-k-1}\right)^s \prod_{j=1}^{k} \left(1 - 2^{-j}\right)^{-d-1}.$$

Proof. The proof is by induction on k. For $k = 0$, we have

$$p_{s,0} \le \left(1 - \frac{1}{2}\right)^s,$$

since we require that none of the s stoppers in S_α be present in $S_{\alpha+1}$, and this can only happen if they are sampled independently of each other. Furthermore, by (2),

$$\begin{aligned} & p_{s,k+1} \\ & \le \max_{\substack{\mathbf{m} \\ c=0}} \sum_{i=b}^{s-a-b} \frac{p_{i,k}}{2^{d+1}} \Pr\left[B_{1/2}^{t_\mathbf{m}} = i - b\right] \qquad (3) \\ & = \max_{\substack{\mathbf{m} \\ c=0}} \frac{1}{2^{d+1+t_\mathbf{m}}} \sum_{i=0}^{t_\mathbf{m}} \binom{t_\mathbf{m}}{i} p_{i+b,k}. \end{aligned}$$

Using the inductive hypothesis and the binomial theorem, we bound the sum as

$$\begin{aligned} & \sum_{i=0}^{t_\mathbf{m}} \binom{t_\mathbf{m}}{i} p_{i+b,k} \\ & \le \frac{\sum_{i=0}^{t_\mathbf{m}} \binom{t_\mathbf{m}}{i} \left(1-2^{-k-1}\right)^{i+b}}{2^{(d+1)k}} \prod_{j=1}^{k} \left(1-2^{-j}\right)^{-d-1} \\ & = \frac{\left(2-2^{-k-1}\right)^{t_\mathbf{m}}}{2^{(d+1)k}} \left(1-2^{-k-1}\right)^b \prod_{j=1}^{k} \left(1-2^{-j}\right)^{-d-1} \\ & = \frac{\left(1-2^{-k-2}\right)^{t_\mathbf{m}}}{2^{(d+1)k-t_\mathbf{m}}} \left(1-2^{-k-1}\right)^b \prod_{j=1}^{k} \left(1-2^{-j}\right)^{-d-1}. \end{aligned}$$

Now, since $t_\mathbf{m} = s - a - 2b \ge s - d - 1 - 2b$ and since

$$\frac{\left(1-2^{-k-1}\right)^b}{\left(1-2^{-k-2}\right)^{2b}} = \left(\frac{1-2^{-k-1}}{1-2^{-k-1}+2^{-2k-4}}\right)^b \le 1,$$

it follows that

$$\begin{aligned} & \sum_{i=0}^{t_\mathbf{m}} \binom{t_\mathbf{m}}{i} p_{i+b,k} \\ & \le \frac{\left(1-2^{-k-2}\right)^s}{2^{(d+1)k-t_\mathbf{m}}} \prod_{j=1}^{k+1} \left(1-2^{-j}\right)^{-d-1}, \end{aligned}$$

and hence (3) gives

$$p_{s,k+1} \le \frac{\left(1-2^{-k-2}\right)^s}{2^{(d+1)(k+1)}} \prod_{j=1}^{k+1} \left(1-2^{-j}\right)^{-d-1},$$

which finishes the induction. □

(Proof of Lemma 4 continued) Now, since $(1-x) > \exp(-x/(1-x))$ for $x < 1$ we have

$$\prod_{j=1}^{k} (1-2^{-j})^{-d-1} \le e^{2(d+1)\sum_{j=1}^{\infty} 2^{-j}} = e^{2(d+1)},$$

so we get as claimed

$$\Pr[A_\alpha] \le p_{|L_\mathbf{u}|,\alpha} \le e^{2d+2} 2^{-(d+1)\alpha} (1-2^{-\alpha-1})^{|L_\mathbf{u}|}.$$

□

Proof. (of Theorem 3) The cost for NNG computations and walking can be bounded as in the proof of Theorem 2 (in particular the degree of the NNG is bounded by a constant for any fixed dimension), except that the expected structural change is not necessarily linear. To prove the theorem, it is sufficient to show that the probability that the simplex Δ spanned by $\mathbf{u} \subseteq P$ occurs in `BrioDC` is up to a constant factor upper bounded by the same probability in a RIC. In the case of `BrioDC`, this probability is bounded by $\sum_{\alpha=0}^{\infty} \Pr[A_\alpha]$. Let p denote the corresponding probability in a RIC and let

$$p_\alpha := 2^{-(d+1)\alpha}(1-2^{-\alpha-1})^{|L_\mathbf{u}|}(1-2^{-d-1}).$$

We have

$$\Pr[A_\alpha] \le \exp(2d+2)(1-2^{-d-1})^{-1} p_\alpha.$$

Now, from an analysis of RIC con BRIO [3, Lemma 3.8] we have, $\sum_{\alpha=0}^{\infty} p_\alpha \le 2^{d+1} p$, thus,

$$\sum_{\alpha=0}^{\infty} \Pr[A_\alpha] \le 2^{d+1} \exp(2d+2)(1-2^{-d-1})^{-1} p.$$

□

References

[1] N. Amenta, S. Choi, and G. Rote. Incremental constructions con BRIO. In *Proc. 19th Annu. ACM Sympos. Comput. Geom.*, pages 211–219. ACM Press, 2003.

[2] K. Buchin. Delaunay triangulations in linear time? Submitted, see `arXiv:0812.0387v1 [cs.CG]`.

[3] K. Buchin. *Organizing Point Sets: Space-Filling Curves, Delaunay Tessellations of Random Point Sets, and Flow Complexes.* PhD thesis, Free University Berlin, 2007. http://www.diss.fu-berlin.de/diss/receive/FUDISS_thesis_000000003494.

[4] T. M. Chan. Well-separated pair decomposition in linear time? *Inform. Process. Lett.*, 107(5):138–141, 2008.

[5] T. M. Chan and M. Pătraşcu. Transdichotomous results in computational geometry, I: Point location in sublogarithmic time. *SIAM J. Comput.* To appear. Preliminary versions in: Proc. 47th IEEE Sympos. Found. Comput. Sci. (FOCS), 2006, pp. 325–332, 333–342.

[6] T. M. Chan and M. Pătraşcu. Voronoi diagrams in $n \cdot 2^{O(\sqrt{\lg \lg n})}$ time. In *Proc. 39th Annu. ACM Sympos. Theory Comput.*, pages 31–39, 2007.