Algorithmica (1992) 7: 381–413

Algorithmica

Randomized Incremental Construction of Delaunay and Voronoi Diagrams[1]

Leonidas J. Guibas,[2,3] Donald E. Knuth,[2] and Micha Sharir[4,5]

Abstract. In this paper we give a new randomized incremental algorithm for the construction of planar Voronoi diagrams and Delaunay triangulations. The new algorithm is more "on-line" than earlier similar methods, takes expected time $O(n\log n)$ and space $O(n)$, and is eminently practical to implement. The analysis of the algorithm is also interesting in its own right and can serve as a model for many similar questions in both two and three dimensions. Finally we demonstrate how this approach for constructing Voronoi diagrams obviates the need for building a separate point-location structure for nearest-neighbor queries.

Key Words. Delaunay triangulation, Voronoi diagram, randomized algorithms.

1. Introduction. The *Voronoi diagram* of n sites in the plane and its dual, the *Delaunay tesselation*, are among the most important constructions in two-dimensional computational geometry. Their properties, as well as algorithms for their construction, are extensively covered in the standard textbooks of the field, such as [13] or [26], and in numerous papers, e.g., [20]. The original papers on the subject are [29] and [30] by Voronoi (circa 1907), and [10] and [11] by Delaunay (circa 1932). Because of the practical importance of the Voronoi and Delaunay diagrams, some of the algorithms for constructing them have been carefully implemented and widely tested in practice. These diagrams are important tools in applications ranging from finite element codes to pattern classification in statistics to motion planning in robotics, etc. Analogs of these diagrams exist in higher dimensions as well. Though the higher-dimensional diagrams are still very useful, a lot less is known about efficient algorithms for their construction because their topological structure is significantly harder to analyze.

In the plane it is known that the worst-case complexity of computing such a diagram for n sites is $\Theta(n \log n)$. Interesting divide-and-conquer [20] and plane-sweep [16] algorithms are available that attain this bound. Both have in fact been implemented and used in practice. The implementations of these methods have

[1] Leonidas Guibas and Micha Sharir wish to acknowledge the generous support of the DEC Systems Research Center in Palo Alto, California, where some of this work was carried out. Donald Knuth has been supported by NSF Grant CCR-86-10181. Micha Sharir has been supported by NSF Grant CCR-89-01484, ONR Grant N00014-K-87-0129, the U.S.–Israeli Binational Science Foundation, and the Fund for Basic Research administered by the Israeli Academy of Sciences.
[2] Computer Science Department, Stanford University, Stanford, CA 94305, USA.
[3] DEC Systems Research Center, Palo Alto, CA 94301, USA.
[4] Courant Institute of Mathematical Sciences, New York University, New York, NY 10012, USA.
[5] School of Mathematical Sciences, Tel Aviv University, Tel Aviv, Israel.

Received November 13, 1989; revised May 20, 1990. Communicated by Kurt Mehlhorn.

been gradually simplified over the years, but they still remain codes of significant programming complexity. For this reason a number of people have continued to implement a simpler method that is based on incrementally adding the sites one at at time, and updating the diagram at each step. For a detailed description of this method see [18] or [20]. Unfortunately, as we will see later, there are situations where the complexity of the incremental method can be as high as $\Theta(n^2)$. In practice, however, these bad situations seem to be rare—most people report a linear, or nearly so, running time. This has been formally proven in case the sites are drawn under some uniform distribution assumptions [4], [12]. Our point of departure in this paper is a simple proof that, for *any* collection of n sites (regardless of their distribution), if we randomize over the sequence of their insertions by the incremental algorithm, then the expected total number of structural changes that happen to the diagram is only $O(n)$, and the overall cost of the algorithm is $O(n \log n)$. Thus we show that an easily implemented incremental algorithm works optimally for any data set, as long as we randomize the insertion sequence of the sites.

Similar results have been recently obtained by Clarkson and Shor [9], and also by Chew [7] and by Mehlhorn *et al.* [24]. Using a standard transformation that lifts the given sites to the paraboloid of revolution $z = x^2 + y^2$ (see Section 4.1 and [13] for details), the problem of computing the Delaunay triangulation of the given sites reduces to that of computing the (lower) convex hull of the transformed sites. Clarkson and Shor [9] show that by inserting the sites one by one in random order, and by maintaining a certain *conflict graph* between the edges of the current hull and the sites yet to be inserted, the incremental construction of the hull can be implemented to run in $O(n \log n)$ expected time. Mehlhorn *et al.* [24] adapt this technique to compute "abstract Voronoi diagrams" in an incremental and randomized fashion. Chew [7] uses a slightly different (and simplified) analysis technique, but his algorithm appears to lack one of the crucial steps (namely, point location of newly inserted sites—see below) needed to achieve an expected $O(n \log n)$ time complexity.

Our paper follows the general approach of an incremental randomized construction, but it differs from the above techniques in the following aspects:

(i) The proof of the bounds on the expected performance of the algorithm is considerably simpler than that given in [9]. We also obtain more information concerning the behavior of the algorithm, such as the linear bound mentioned above on the expected number of structural changes that occur during the construction, together with probabilistic estimates on whether a particular triangle or edge ever arises during the incremental construction. As a matter of fact, many ingredients of our proof can be used to obtain simple derivations of similar bounds on the performance of a variety of incremental randomized algorithms.

(ii) The algorithm itself is simpler and more "on line." It requires no auxiliary data structures referencing as yet uninserted sites (such as the conflict graph of [9]). The main reason for maintaining such a structure is to avoid having to search for the current Delaunay triangle containing the next site to be added (see Section 3 and [20] for details). However, combining our incremental algorithm with a recent interesting randomized approach to point location by Seidel [28], we show that, if we keep all the incremental versions of the triangulation "on top" of one another, we obtain a data structure that already facilitates fast point location, at least in an amortized sense, allowing us to locate the next site in the current triangulation in expected $O(\log n)$ amortized time. This yields an overall $O(n \log n)$ expected time for the construction of the triangulation.

Our analysis is based on the notion of the *scope* of the triangle spanned by three sites X, Y, and Z. This is defined to be equal to the number of sites contained in the interior of the circumcircle of the triangle XYZ. (In particular, 0-scope triangles are Delaunay triangles in the final triangulation.) In Section 2 we analyze the probability that a triangle XYZ will arise as a Delaunay triangle at some stage during the incremental construction, and obtain bounds on this probability in terms of the scope of XYZ. Then by an argument based on random sampling, similar to that used by Clarkson and Shor [9], we estimate the number of triples of sites defining triangles whose scope is at most k. By combining these results we can conclude that the expected total number of triangles that ever arise as Delaunay triangles during the incremental construction, and therefore the total number of structural changes in the diagram, is linear in the number of sites n.

We also develop another analysis technique, in which we analyze how the algorithm manipulates edges connecting pairs of sites, rather than triangles, as above. We extend the notion of scope to edges—an edge XY is said to have *symmetric scope* k if there exists a circle passing through X and Y and containing in its interior exactly k sites to the left of XY and exactly k sites to the right. We then derive bounds on the probability that an edge with symmetric scope k ever appears in the triangulation, and on the number of edges with symmetric scope at most k. This analysis is somewhat more complicated than the analysis of triangles, but it is combinatorially quite interesting. Furthermore, it enables us to obtain sharp bounds on certain refined aspects of the construction. For example, suppose that the Delaunay diagram of m sites is already computed and n additional sites are to be added. We show that if these new sites are added in random order, then the total number of structural changes in the diagram has expected value $O(m \log n + n)$, and that this bound is tight in the worst case.

Returning now to the incremental algorithm, we need to address the issue of how such an algorithm performs point location for each site. In other words, when a new site is to be added, the algorithm must locate the triangle containing the new site in the Delaunay triangulation of the already inserted sites. Although fancy structures for doing this are available, implementors who switched to the incremental method because of its simplicity also wanted a comparably simple piece of code to accomplish the point-location task. A number of different heuristic ideas have been tried [7], [20]. Another way of achieving this would be to maintain some sort of conflict graph (such as in [9]), appropriately adapted to the case of Delaunay triangulation. However, we show that none of these techniques is necessary. If we just maintain all versions of the triangulation on top of one another, together with appropriate links between overlapping "old" and "new" triangles, we obtain a structure in which the next site can be located by simply tracing all triangles that contain it, in chronological order of their creation. This is a similar idea to the Delaunay tree proposed by Boissonnat and Teillaud [5]—see Section 3 for more details. We show that the expected number of triangles that need to be traced per site, summed over all sites, in only $O(n \log n)$, so that the expected total cost of point location is $O(n \log n)$, when averaged over all insertion orderings of the sites. From a well-known lower bound for sorting, we can prove that this is best possible, even for a randomized algorithm.

One nice feature of the incremental method and the analysis below is that it can be extended to higher dimensions. In Section 4 we give a randomized incremental algorithm for computing the convex hull of n sites in three dimensions, which is a considerably simplified version of Clarkson and Shor's technique [9]. The algorithm and its analysis are a fairly direct translation of our two-dimensional algorithm for the Delaunay triangulation in the plane. Of much greater interest is the question of computing Voronoi and Delaunay diagrams in three dimensions. We prove that the expected total number of structural changes during a randomized incremental construction of a three-dimensional diagram is $O(n^2)$, i.e., asymptotically the same as the worst-case size of the final diagram. However, the design of an actual efficient algorithm for doing so remains unresolved. In three and higher dimensions the whole issue of output sensitive algorithms arises, and we explore some basic aspects of this subject. Specifically, we obtain a more refined bound on the expected number of structural changes, in terms of the expected size of the diagrams of random subsets of the sites of certain sizes.

We also obtain a number of results in combinatorial geometry about Voronoi edges of higher order in two or three dimensions. For example, we obtain a new proof of the fact that in the plane there is always a pair of sites such that any circle through these two sites always contains at least a fixed fraction of all the sites.

Finally in Section 5 we given another application of the idea of keeping all versions of the incremental structure "one on top of the other." We give a method for building the Voronoi diagram of n sites in the plane incrementally, while maintaining all intermediate results appropriately linked together. The result is a structure whose expected size is $O(n)$ if we randomize over the sequence of insertion of the sites, and which can be used for efficient point location in the final Voronoi diagram. Thus no further preprocessing is necessary to answer nearest-neighbor queries. We prove that, for any fixed point p in the plane, we can trace through the incremental structure and locate the final Voronoi region containing p in $O(\log^2 n)$ expected time. We believe that this query time is actually $O(\log n)$, and that the idea of linking together all versions of a randomized incremental structure will find further applications in the future.

2. The Number of Structural Changes. In this section we analyze the expected number of structural changes that a Delaunay triangulation undergoes during a randomized incremental construction. The first two subsections prove by independent arguments, based on the Delaunay faces and edges respectively, that the expected number of changes is *linear* in the number of sites to be inserted (and the implied constant of proportionality is reasonably small). The third subsection analyzes the situation where some sites have already been inserted and we randomize over the insertion ordering of the additional set of sites only.

In what follows we use the well-known facts that any triangulation of n points in the plane contains $2n - h - 2$ triangles and $3n - h - 3$ edges, where h denotes the number of points on the convex hull. To avoid monotony of language we sometimes refer to the n given points as "sites." In some contexts the word "point" may refer to an arbitrary point of the plane, but the word "site" will always refer to one of the original given points.

2.1. Analysis of the Triangle Probabilities. Consider a set $\mathscr{P}$ of n points in the plane, no three collinear and no four on a circle (these are the usual nondegeneracy conditions for Delaunay/Voronoi computations; they guarantee that the Delaunay diagram is actually a triangulation). The Delaunay triangulation of $\mathscr{P}$ is composed of the set of edges XY ($X, Y \in \mathscr{P}$), such that there is a circle through X and Y containing no other point. Alternatively, it consists of those triangles XYZ ($X, Y, Z \in \mathscr{P}$) whose circumcircle contains no site in its interior. These basic facts are discussed in all standard treatments of the subject (see [13], [26], or [20]). It is possible to show that even for degenerate point sets the upper bounds derived here will hold; a general technique for doing so has been recently introduced by Jaromczyk and Swiatek [21].

Define the *scope* of a triangle XYZ to be the number of sites contained in the interior of its circumcircle. The scope of a triangle can assume values between 0 and $n - 3$, and triangles of zero scope are precisely the Delaunay triangles. As we will see, this turns out to be a key concept in our analysis.

Suppose we insert the sites one at a time in random order, and maintain the Delaunay triangulation as we go. A triangle XYZ with a small scope is likely to appear as a Delaunay triangle at some stage during the incremental process—this occurs when we insert its three vertices before we insert any of the sites in the interior of the circumcircle of XYZ. A triangle with a large scope is less likely to appear as a Delaunay triangle. More precisely, we have:

LEMMA 2.1. *Let XYZ be a triangle spanned by three sites X, Y, Z, and having scope k. Then the probability that XYZ will ever arise as a Delaunay triangle during the incremental insertion of the sites is $6/(k + 1)(k + 2)(k + 3)$.*

PROOF. Triangle XYZ will appear as a Delaunay triangle if and only if we insert X, Y, and Z before we insert any of the k sites lying inside of the circumcircle of XYZ. The probability that this will happen in a random insertion permutation of the sites is $3!\, k!/(k + 3)! = 6/(k + 1)(k + 2)(k + 3)$. □

Let T_k denote the set of triangles XYZ whose scope is k, for $0 \le k \le n - 3$. Let $T_{\le k} = \bigcup_{j=0}^{k} T_j$.

LEMMA 2.2. *The number of triangles XYZ whose scope is at most k is*

$$\|T_{\leq k}\| = O(n(k+1)^2).$$

PROOF. The proof is based on the probabilistic technique of Clarkson and Shor [9]. Draw a random sample $\mathscr{R}$ of r ($r \geq 3$) among the given sites. Let $D(\mathscr{R})$ denote the set of Delaunay triangles XYZ. The expected number of such triangles is

$$\begin{aligned}\mathbf{E}[\|D(\mathscr{R})\|] &= \sum_{XYZ} \mathbf{Prob}[XYZ \in D(\mathscr{R})] \\ &= \sum_{j=0}^{n-3} \sum_{XYZ \in T_j} \mathbf{Prob}[XYZ \in D(\mathscr{R})] \\ &\geq \sum_{j=0}^{k} \sum_{XYZ \in T_j} \mathbf{Prob}[XYZ \in D(\mathscr{R})].\end{aligned}$$

For a fixed triangle $XYZ \in T_j$, the probability that it is a triangle of $D(\mathscr{R})$ is equal to the probability that X, Y, and Z are chosen in $\mathscr{R}$, and that none of the j sites in the interior of the circumcircle of this triangle are chosen. This probability is $\binom{n-j-3}{r-3}\Big/\binom{n}{r}$, which is $\geq \binom{n-k-3}{r-3}\Big/\binom{n}{r}$ when $j \leq k$. Since $\|D(\mathscr{R})\|$ is always at most $2r$, we obtain the inequality

$$2r \geq \mathbf{E}[\|D(\mathscr{R})\|] \geq \sum_{j=0}^{k} \|T_j\| \cdot \frac{\binom{n-k-3}{r-3}}{\binom{n}{r}}$$

or equivalently

$$\|T_{\leq k}\| \leq \frac{2r\binom{n}{r}}{\binom{n-k-3}{r-3}} = \frac{2n\binom{n-1}{r-1}}{\binom{n-k-3}{r-3}} = 2nB(n-1, k+2, r-3, 2),$$

where

$$B(N, M, s, t) = \frac{\binom{N}{s+t}}{\binom{N-M}{s}}.$$

The best upper bound is obtained when we choose r so that $B(n-1, k+2, r-3, 2)$ is minimized.

In general, we have that for successive values of s the ratio $B(N, M, s+1, t)/B(N, M, s, t) = t(N-M+1)/M$. Hence the minimum of $B(N, M, s, t)$ for fixed N, M, and t occurs when $s = \lfloor t(N-M+1)/M \rfloor$.

In our case the best upper bound is therefore obtained when

$$r = \lfloor 2n/(k+2) \rfloor + 1$$

(if this makes $r < 3$ the result of the lemma is obvious and we can stop). Otherwise we get the bound

$$\begin{aligned} B(n-1, k+2, r-3, 2) &= \frac{n-1}{r-1} \cdot \frac{n-2}{r-2} \cdot \frac{n-3}{n-r} \cdot \frac{n-4}{n-r-1} \cdots \frac{n-k-2}{n-r-k+1} \\ &< \left(\frac{n-2}{r-2}\right)^2 \left(\frac{n-k-2}{n-r-k+1}\right)^k \\ &< \left(\frac{(n-2)(k+2)}{2n-2k-4}\right)^2 \left(1 + \frac{2}{k}\right)^k = O((k+1)^2). \end{aligned}$$

□

REMARK. The O-constant we get from this analysis is about $e^2/2 \approx 3.6$, but the actual constant is probably less.

THEOREM 2.3. *The expected number of triangles that appear as Delaunay triangles during the incremental construction of the Delaunay triangulation is* $O(n)$.

PROOF. In what follows we abbreviate "triangle XYZ appears as a Delaunay triangle during the incremental construction" to "XYZ appears." The expected number in question is

$$\begin{aligned} \sum_{XYZ} [\mathbf{Prob}[XYZ \text{ appears}] &= \sum_{j=0}^{n-3} \sum_{XYZ \in T_j} \mathbf{Prob}[XYZ \text{ appears}] \\ &= \sum_{j=0}^{n-3} \sum_{XYZ \in T_j} \frac{6}{(j+1)(j+2)(j+3)} \\ &= 6 \sum_{j=0}^{n-3} \frac{\|T_j\|}{(j+1)(j+2)(j+3)} \\ &\le \|T_0\| + 6 \sum_{j \ge 1} \left[\frac{\|T_{\le j}\|}{(j+1)(j+2)(j+3)} - \frac{\|T_{\le j-1}\|}{(j+1)(j+2)(j+3)}\right] \\ &= 18 \sum_{j \ge 0} \frac{\|T_{\le j}\|}{(j+1)(j+2)(j+3)(j+4)} \\ &= O\left(n \cdot \sum_{j \ge 0} \frac{1}{(j+3)^2}\right) = O(n). \end{aligned}$$

□

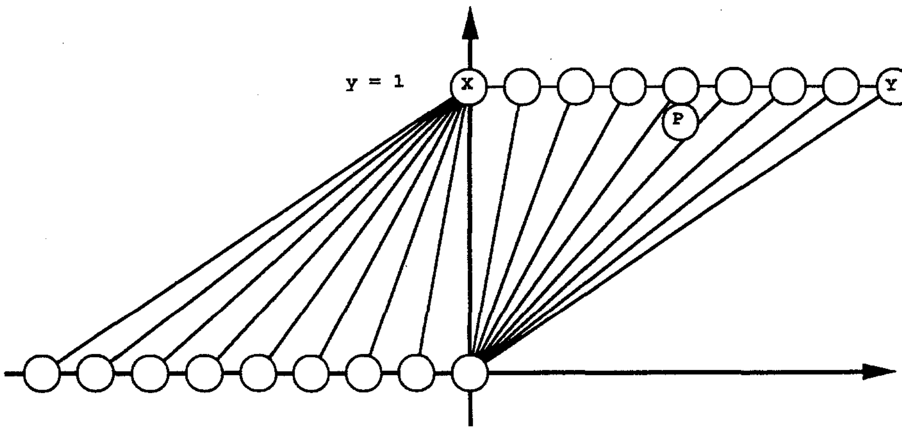

Fig. 1. A difficult case for the incremental algorithm (points X, Y, and P are used in Section 3).

REMARK. The actual O-constant we get from this calculation is about $3e^2 \approx 22$; again this appears to be a significant overestimate of the true constant of proportionality.

REMARK. If the order of insertion of the points is *not* random, it is possible that $\Theta(n^2)$ triangles will appear as Delaunay triangles. To see this, consider the following example, related to an example introduced in [17]: Let $\mathscr{P}$ be a set of $n = 2m$ points, half of which lie on the negative portion of the x-axis, and the other half lie on the positive portion of the line $y = 1$. If we first insert all the lower points, and then insert the upper points from *right to left*, it is easily seen that each new point with $y = 1$ needs to be connected by Delaunay edges to each of the points on the x-axis (see Figure 1). Thus quadratically many Delaunay edges and triangles will arise in this case.

REMARK. It would be interesting also to obtain estimates for the variance of the number of triangles that appear, and for the probability that the number of triangles that arise exceeds the expected value by more than some constant multiple.

REMARK. The analysis presented here applies with minor modifications to Delaunay triangulations under other metrics, by replacing circles with the appropriate balls in the given metric.

2.2. Analysis of the Edge Probabilities. Since the number of triangles that ever appear during the incremental construction is $O(n)$, it follows that the total number of edges that ever appear as Delaunay edges is also linear. Below we give a refined analysis of the edge probabilities. We introduce the concept of the *symmetric scope* of an edge that is strongly related to the probability that the edge will appear at some point during the random incremental process and is analogous to the scope

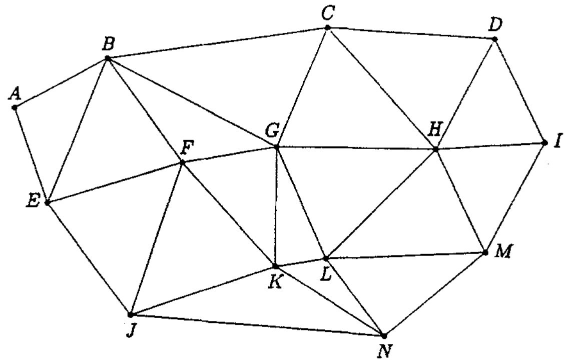

Fig. 2. A Delaunay triangulation of 14 points.

of a triangle defined above. This refined analysis is interesting in its own right; it also allows us to obtain bounds in certain situations for which a triangle-based analysis is not sufficient (see Section 2.3).

For the edge-based analysis, it will be instructive to focus on a particular example of a Delaunay diagram. Figure 2 shows a set of fourteen points in the plane with their Delaunay triangulation.

Given two points X and Y, we can imagine a family of circles with parameter t that pass through X and Y and through the point that lies t units to the right of the line XY, on the perpendicular bisector of X and Y. As t runs from 0 to ∞, these circles sweep past all the other points in a definite order. For example, when $XY = NB$ in the 14-point arrangement above, this order turns out to be

$$L\ G\ A\ C\ E\ H\ J\ D\ M\ I\ K\ F.$$

Points A, E, J, K, F lie to the left of NB, and the other points lie to the right; we can conveniently represent the situation by writing

$$A_3\ E_5\ J_7\ K_{11}\ F_{12} \mid L_1\ G_2\ C_4\ H_6\ D_8\ M_9\ I_{10}, \tag{1}$$

where the subscripts indicate the relative order and the vertical line represents NB.

Each pair of points defines a different ordering. For example, the ordering for NL is

$$B_1\ A_2\ F_3\ E_4\ J_5\ K_6 \mid M_7\ H_8\ I_9\ D_{10}\ C_{11}\ G_{12}.$$

In this case all the subscripts on the left turn out to be less than all the subscripts on the right; this condition is necessary and sufficient for the given edge XY to be part of the final Delaunay triangulation.

Suppose we input the 14 points $A, \ldots, N$ of our example in random order and maintain Delaunay triangulations as we go. What is the probability that NB will be an edge of the triangulation at some time during this process? Using the information in (1), we can express this probability as the sum of the probabilities of the following mutually disjoint events:

$$\begin{array}{ll}
\{N,F\}\,B\,\{L,G,C,H,D,M,I\} & \{B,F\}\;N\;\{L,\,G,\,C,\,H,\,D,\,M,\,I\} \\
\{N,K\}\,B\,\{F,L,G,C,H,D,M,I\} & \{B,K\}\;N\;\{F,\,L,\,G,\,C,\,H,\,D,\,M,\,I\} \\
\{N,J\}\,B\,\{K,F,L,G,C,H\} & \{B,J\}\;N\;\{K,\,F,\,L,\,G,\,C,\,H\} \\
\{N,E\}\,B\,\{J,K,F,L,G,C\} & \{B,E\}\;N\;\{J,\,K,\,F,\,L,\,G,\,C\} \\
\{N,A\}\,B\,\{E,J,K,F,L,G\} & \{B,A\}\;N\;\{E,\,J,\,K,\,F,\,L,\,G\} \\
\{N\}\,B\,\{A,E,J,K,F\} & \{B\}\;N\;\{A,\,E,\,J,\,K,\,F\}.
\end{array}$$

Here "$\{N,J\}\ B\ \{K,\ F,\ L,\ G,\ C,\ H\}$" means, for example, that points N and J appear before B, which appears before K, F, L, G, C, and H. (In that case, the edge NB will be in the Delaunay triangulation when B first appears.) The probability of the event $\{X_1,\ldots,X_k\}\ Y\ \{Z_1,\ldots,Z_l\}$ is $k!\ l!/(k+l+1)!$. Hence, if we write

$$a_k = 2/(k+1)(k+2), \tag{2}$$

$$b_k = 4/(k+1)(k+2)(k+3) = a_k - a_{k+1}, \tag{3}$$

the probabilities of the pairs of events on the six lines above are respectively b_7, b_8, b_6, b_6, b_6, a_5.

The pattern of subscripts in (1) can be represented by a zig-zag path that runs from (0, 5) to (7, 0) (see Figure 3). Here the kth step of the path goes downward if subscript k appears on the left of the vertical line; otherwise the kth step goes to the right. The successive coordinates of the path count the number of points to the right and left of NB that lie inside the circle at time t, as t as t increases from 0 to ∞.

In general, if there are l points $\{p_1,\ldots,p_l\}$ to the left of XY and r points $\{q_1,\ldots,q_r\}$ to the right, we can construct a similar zig-zag path from $(0, l)$ to $(r, 0)$. The probability that XY lies in a Delaunay triangulation at some time, when the $l+r+2$ points are input sequentially in random order, will be the sum of $l+1$ pairs of mutually disjoint events:

$$\begin{array}{cc}
\{X,p_l\}\ Y\ \{q_1,\ldots,q_{x_1}\} & \{Y,p_l\}\ X\ \{q_1,\ldots,q_{x_1}\} \\
\{X,p_{l-1}\}\ Y\ \{p_l,q_1,\ldots,q_{x_2}\} & \{Y,p_{l-1}\}\ X\ \{p_l,q_1,\ldots,q_{x_2}\} \\
\vdots & \vdots \\
\{X,p_l\}\ Y\ \{p_2,\ldots,p_l,q_1,\ldots,q_{x_l}\} & \{Y,p_1\}\ X\ \{p_2,\ldots,p_l,q_1,\ldots,q_{x_l}\} \\
\{X\}\ Y\ \{p_1,\ldots,p_l\} & \{Y\}\ X\ \{p_1,\ldots,p_l\},
\end{array} \tag{4}$$

where x_k is the x-coordinate of the kth-from-last vertical step of the path. (The values of $x_1,\ldots,x_5$ in the example above are respectively 7, 7, 4, 3, 2.) The final pair of events will occur with probability a_l; the kth pair, for $1 \le k \le l$, will occur with probability b_{x_k+k-1}.

In particular, when edge XY is part of the overall Delaunay triangulation, the path runs from $(0, l)$ to $(0, 0)$ to $(r, 0)$; hence $x_k = 0$ for all k, and the total probability

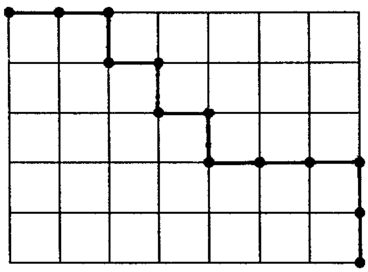

Fig. 3. A zig-zag path corresponding to an edge.

is

$$b_0 + b_1 + \cdots + b_{l-1} + a_l = (a_0 - a_1) + (a_1 - a_2) + \cdots + (a_{l-1} - a_l) + a_l$$
$$= a_0 = 1.$$

The minimum probability, given l and r, occurs when $x_k = r$ for all k; it is

(5) $$b_r + b_{1+r} + \cdots + b_{l+r-1} + a_l = a_l + a_r - a_{l+r}.$$

We can express the probability in a more intuitive and more symmetric way by putting the "weight" c_{i+j} into the unit square bounded by (i, j), $(i + 1, j)$, $(i, j + 1)$, and $(i + 1, j + 1)$, where

$$c_k = 12/(k + 1)(k + 2)(k + 3)(k + 4) = b_k - b_{k+1}.$$

The probability is then 1 minus the sum of all the weights under the zig-zag path. For instance, the weights under our example path (see Figure 4) sum to

$$(c_0 + c_1 + \cdots + c_6) + (c_1 + c_2 + \cdots + c_7) + (c_2 + \cdots + c_5)$$
$$+ (c_3 + c_4 + c_5) + (c_4 + c_5)$$
$$= (b_0 - b_7) + (b_1 - b_8) + (b_2 - b_6) + (b_3 - b_6) + (b_4 - b_6)$$
$$= 1 - (b_7 + b_8 + b_6 + b_6 + b_6 + a_5).$$

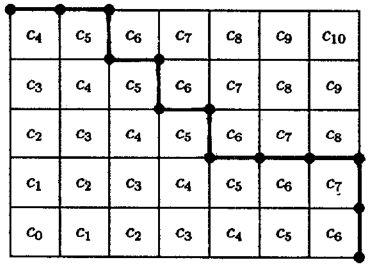

Fig. 4. The weights under a path.

An equivalent formula for the probability is $a_l + a_r - a_{l+r}$ plus the sum of all weights *above* the zig-zag path, in the $l \times r$ rectangle that encloses all possible paths.

Notice that the area under the zig-zag path is the number of inversions of the permuation represented by the subscripts of (1). This property holds in general; the path defines a partition of the inversions. Our example $XY = NB$ has 23 inversions, and the partition is $7 + 7 + 4 + 3 + 2$. A Delaunay edge is an edge that has no inversions. If there are comparatively few inversions, the probability is high that XY will be included as a Delaunay edge during the process; if there are comparatively many inversions, the probability is low. However, we will see that another measure of "non-Delaunayhood" is more relevant than the total number of inversions.

Define the *symmetric scope* of edge XY to be the unique integer w such that (w, w) lies on the zig-zag path associated with XY. In geometric terms, the symmetric scope is w if and only if there exists a circle passing through XY whose interior contains exactly w points to the right of XY and exactly w points to the left. For example, the symmetric scope of NB in our 14-point illustration is 3; there is a circle through NB that contains J, K, F on the left and L, G, C on the right.

Notice that $0 \le w \le \min(l, r) \le (l + r)/2 = n/2 - 1 < \lfloor n/2 \rfloor$, and that edges of zero scope are true Delaunay edges. The concept of symmetric scope turns out to be the key tool in our edge-based analysis, because it provides a canonical classification of edges in which most edges have a fairly large scope, and because a symmetric scope of w implies that all inversions in the square bounded by $(0, 0)$, $(0, w)$, $(w, 0)$, and (w, w) are present. (The same idea has proved to be important in the theory of partitions, where w is called the width of the "Durfee square;" see p. 28 of [2].)

LEMMA 2.4. *The probability that edge XY appears as a Delaunay edge during a random incremental procedure is at most* $4/(w + 1)(w + 2)$, *when w is the symmetric scope of XY.*

PROOF. We know that the probability is $1 - c(XY)$, where $c(XY)$ is $\sum c_{i+j}$ summed over all pairs (i, j) lying below the zig-zag path corresponding to XY. The smallest possible $c(XY)$, given w, corresponds to the path from $(0, l)$ to $(0, w)$ to (w, w) to $(w, 0)$ to (w, r). This sums to $1 - 2a_w + a_{2w}$ by (5), because it corresponds to the minimum probability when $l = r = w$. Hence the probability for XY is at most

$$2a_w - a_{2w} \le 2a_w = 4/(w + 1)(w + 2). \qquad \square$$

REMARK. The tight upper bound $2a_w - a_{2w} = 3.5w^{-2} + O(w^{-3})$ derived in this proof has a corresponding lower bound $a_{2w} = 0.5w^{-2} + O(w^{-3})$, obtained by subtracting from the upper bound $\sum\{c_{i+j} \mid \min(i, j) < w \text{ and } \max(i, j) \ge w\} = 2\sum_{i=0}^{w-1}\sum_{j=w}^{\infty} c_{i+j} = 2\sum_{i=0}^{w-1} b_{i+w} = 2(a_w - a_{2w})$. Thus the probability for XY is $\Theta(w^{-2})$.

Let E_w denote the set of edges XY whose symmetric scope is w, and let $E_{\le k} = \bigcup_{w=0}^{k} E_w$. Thus, $E_0 = E_{\le 0}$ is the set of Delaunay edges. When $w \ge \lfloor n/2 \rfloor$, the set E_w is empty and $E_{\le w}$ contains all $\binom{n}{2}$ edges.

LEMMA 2.5. *The number $\|E_{\le k}\|$ of edges XY whose symmetric scope is at most k is $O(n(k+1))$.*

PROOF. The proof is analogous to that of Lemma 2.2. We draw a random sample $\mathscr{R}$ by choosing r of the given points, where r is a parameter to be chosen later. We now let $D(\mathscr{R})$ denote the set of Delaunay edges of the sample $\mathscr{R}$. The expected number of such edges is

$$\mathbf{E}[\|D(\mathscr{R})\|] \ge \sum_{w=0}^{k} \sum_{XY \in E_w} \mathbf{Prob}[XY \in D(\mathscr{R})], \tag{6}$$

arguing as before. We also know that the total number of Delaunay edges in the sample is always at most $3r$.

Suppose XY is an edge of E_w. If w is fairly small, the edge XY is pretty likely to be Delaunay, provided that points X and Y are in the sample and the sample is not too large, because the absence of only a few points can guarantee that XY will be Delaunay. Indeed, the definition of symmetric scope tells us that there is a set $\mathscr{P}_{XY}$ of $2w$ points, w to the left of XY and w to the right, for which some circle through X and Y encloses precisely these $2w$ points. Thus we have the lower bound

$$\mathbf{Prob}[XY \in D(\mathscr{R})] \ge \mathbf{Prob}[\{X, Y\} \subseteq \mathscr{R} \subseteq \mathscr{P} \backslash \mathscr{P}_{XY}] = \frac{\binom{n-2w-2}{r-2}}{\binom{n}{r}}.$$

Putting these estimates together yields

$$3 \ge \mathbf{E}[\|D(\mathscr{R})\|] \ge \sum_{w=0}^{k} \frac{\binom{n-2w-2}{r-2}\|E_w\|}{\binom{n}{r}}$$

$$\ge \sum_{w=0}^{k} \frac{\binom{n-2k-2}{r-2}\|E_w\|}{\binom{n}{r}} = \frac{\binom{n-2k-2}{r-2}\|E_{\le k}\|}{\binom{n}{r}}.$$

In other words, we have derived an upper bound on $\|E_{\le k}\|$ that depends on the sample size r:

$$\|E_{\le k}\| \le \frac{3r\binom{n}{r}}{\binom{n-2k-2}{r-2}} = \frac{3n\binom{n-1}{r-1}}{\binom{n-2k-2}{r-2}} = 3nB(n-1, 2k+1, r-2, 1).$$

The general analysis of $B(N, M, s, t)$ in Lemma 2.2 tells us that the best bound is obtained by setting $r = \lfloor n/(2k+1) \rfloor + 1$; for that choice we obtain $\|E_{\le k}\| = O(n(k+1))$. The actual implied O-constant we get from this calculation is about $6e \approx 16$. □

An analog of Theorem 2.3 for the total number of edges can now easily be proven. The overall O-constant thus obtained is about 65. This constant could be significantly reduced by slight refinements of the analysis. Incidentally, the 14-point example introduced above has $\|E_0\|, \|E_1\|, \ldots, \|E_5\| = 30, 29, 17, 8, 5, 2$, respectively; the two edges with symmetric scope 5 are AM and EI. The expected number of Delaunay edges in a randomized incremental procedure turns out to be 44.559 in that example, suggesting that the actual proportionality constant is much smaller than what we derived. (The 14-point example also has $\|T_0\|, \|T_1\|, \ldots, \|T_{11}\| = 17, 32, 42, 44, 47, 44, 40, 33, 28, 18, 12, 7$; hence its expected number of Delaunay triangles is 34.592.)

2.3. Adding Points to an Already Existing Set. Suppose we have a set of m points, $\mathcal{Q}$, whose Delaunay triangulation has already been constructed. We are then given another set, $\mathcal{P}$, of n points, and wish to insert these points one by one in random order, and maintain the Delaunay triangulation of the combined set, until we finally obtain the triangulation of $\mathcal{Q} \cup \mathcal{P}$. What is the expected number of edges that will appear as Delaunay edges during this process? An appropriate modification of the proof technique used in Section 2.2 yields:

THEOREM 2.6. *In the above situation, the expected number of edges (or triangles) that appear in the Delaunay triangulation is $O(m \log n + n)$. This bound is tight in the worst case.*

PROOF. Let us redo Lemmas 2.4 and 2.5; as before, we can dispose of degenerate configurations by a special argument. First, we redefine the scope of an edge XY as follows:

(i) If there exists a circle that passes through XY, contains exactly k of the points of $\mathcal{P}$ on the left of XY and exactly k points on the right, and does not contain any point of $\mathcal{Q}$, then the scope of XY is k.
(ii) If there does not exist a circle passing through XY and free of points of $\mathcal{Q}$, then the scope is $+\infty$.
(iii) If none of the first two cases arises, then there has to exist a unique circle that passes through X, Y and a point q of $\mathcal{Q}$, contains l points of $\mathcal{P}$ on the same

side of XY as q, contains $k \geq l$ points of $\mathcal{P}$ on the other side, and does not contain any point of $\mathcal{Q}$ in its interior. In this case we define the scope of XY to be k.

Note that the new definition of scope is no longer symmetric. Note also that the definition makes sense also in case either X or Y belongs to $\mathcal{Q}$.

LEMMA 2.7. *For a fixed edge XY, where both X and Y belong to $\mathcal{P}$, the probability that XY appears in the Delaunay triangulation during the incremental process is $O(1/(k+1)^2)$, where k is the new scope of XY.*

PROOF. Let the permutation of points in $\mathcal{Q} \cup \mathcal{P}$ ordered by the circle-sweeping order associated with XY be

$$W_1, \ldots, W_s | Z_1, \ldots, Z_t,$$

where $W_1, \ldots, W_s$ lie to the left of XY and $Z_1, \ldots, Z_t$ lie to the right. Let μ be the largest index such that $W_\mu \in \mathcal{Q}$ ($\mu = 0$ if no such point exists), and let ν be the smallest index such that $Z_\nu \in \mathcal{Q}$ ($\nu = t + 1$ if no such point exists).

It is easily seen that the event of XY appearing as a Delaunay edge is now the union of only some of the pairs of events considered previously. Specifically, if r_j denotes the length of the jth row from the bottom below the zig-zag path associated with XY, then the only pairs of events that have nonzero probability are of the form

$$\{X, W_{s-j+1}\}\ Y\ \{W_{s-j+2}, \ldots, W_s, Z_1, \ldots, Z_{r_j}\},$$
$$\{Y, W_{s-j+1}\}\ X\ \{W_{s-j+2}, \ldots, W_s, Z_1, \ldots, Z_{r_j}\}$$

provided $j < s - \mu + 1$ and $r_j < \nu$ (and $\{X\}\ Y\ \{W_1, \ldots, W_s\}$ or $\{Y\}\ X\ \{W_1, \ldots, W_s\}$, if $\mu = 0$).

Let us consider separately each of the three possible cases in the definition of the scope of XY:

- In case (i) we have $k < s - \mu + 1$ and $r_j > k$ for all $j \leq k$. Arguing as in the proof of Lemma 2.4, it is easily checked that the probability that XY appears as an edge is $O(1/(k+1)^2)$.
- In case (ii) the probability that XY appears as an edge is clearly 0.
- In case (iii), let us assume, without loss of generality, that the point q lies to the left of XY. Then only the first l pairs of events in which XY appears as an edge can have nonzero probability, and for each of these pairs we have $r_j \geq k$. Thus the probability that XY appears as an edge can be bounded by

$$\sum_{j=1}^{l} b_{r_j+j-1} \leq \sum_{j=1}^{l} b_k = O\left(\frac{l}{(k+1)^3}\right) = O\left(\frac{1}{(k+1)^2}\right).$$

This completes the proof of the lemma. □

LEMMA 2.8. *For a fixed edge XY, where $X \in \mathscr{P}$ and $Y \in \mathscr{Q}$, the probability that XY appears in the Delaunay triangulation during the incremental process is $O(1/(k+1))$, where k is the new scope of XY.*

PROOF. The proof proceeds in a manner analogous to the proof of the preceding lemma. The only difference is that in each pair of events that make XY appear as an edge, one event, namely that in which X is chosen before Y, is impossible, and in the other event the choice of Y before X is already guaranteed. This makes the probability of the jth event $\frac{1}{2}a_{r_j+j-1}$ instead of b_{r_j+j-1} (here we think of the events ordered in the same sequence as in the previous subsection). The assertion of the lemma now follows in much the same way as above; we leave it to the reader to work out the details. □

Next we need to extend Lemma 2.5 to the current case:

LEMMA 2.9. *Let k be a finite integer. The number of edges XY of scope at most k, where both X and Y belong to $\mathscr{P}$, is $O(m(k+1)^2 + n(k+1))$. In case one of X, Y belongs to $\mathscr{P}$ and one to $\mathscr{Q}$, the number of such edges is $O(m(k+1)+n)$.*

PROOF. Let $E_j^{(2)}$ (resp. $E_{\le j}^{(2)}$) denote the set of edges of scope j (resp. at most j) connecting two points in $\mathscr{P}$, and let $E_j^{(1)}$ (resp. $E_{\le j}^{(1)}$) denote the set of edges of scope j (resp. at most j) connecting a point of $\mathscr{P}$ to a point of $\mathscr{Q}$.

Consider first the case where both X and Y belong to $\mathscr{P}$. The argument is very similar to that used in the proof of Lemma 2.5, except for the following technical differences:

1. We draw a random sample $\mathscr{R}$ of r points of $\mathscr{P}$, but consider the Delaunay triangulation of $\mathscr{Q} \cup \mathscr{R}$. This triangulation has at most $3(m+r)$ edges.
2. In the expression for $\mathbf{E}[\|D(\mathscr{Q} \cup \mathscr{R})\|]$ we can ignore edges XY having infinite scope, because they can never show up as edges of this triangulation.
3. For an edge $XY \in E_j^{(2)}$ (with $j < +\infty$), there exists a circle passing through XY and containing in its interior ar most $2j$ points of $\mathscr{P}$ and no point of $\mathscr{Q}$. Thus if we manage to choose X, Y in $\mathscr{R}$, but not to choose any of these points, the edge XY will appear as a Delaunay edge. This enables us to obtain the same lower bound on the probability that XY appears as an edge of $D(\mathscr{Q} \cup \mathscr{R})$.

Using these observations, the analysis in the proof of Lemma 2.5 now yields

$$\|E_{\le k}^{(2)}\| = O\left(\frac{n^2}{r^2}\cdot(m+r)\right),$$

where $r = \lfloor n/k \rfloor$. This implies immediately the bound asserted in the first part of the lemma.

As to the proof of the second part of the lemma, let us suppose that $X \in \mathscr{P}$ and $Y \in \mathscr{Q}$. The analysis is similar to the one just outlined, except that now, to make an edge XY of scope j appear as an edge of $D(\mathscr{Q} \cup \mathscr{R})$, it suffices to choose X in

$\mathscr{R}$ and not to choose any of the at most $2j$ points of $\mathscr{P}$ lying in the corresponding circle (that is, we do not have to worry about choosing Y—it has already been "chosen" in $\mathscr{Q}$). It is easy to show that the modified analysis gives

$$\|E^{(1)}_{\leq k}\| = O\left(\frac{n}{r}\cdot(m + r)\right)$$

for $r = \lfloor n/k \rfloor$. Again, the asserted bound follows. □

Now we can complete the proof of the theorem. Arguing in an analogous manner to the proof of Theorem 2.3, we show that the expected number of edges, adjacent to at least one point of $\mathscr{P}$, that appear as Delaunay edges during the incremental process is

$$O\left(\sum_j \frac{E^{(2)}_j}{(j+1)^2} + \sum_j \frac{E^{(1)}_j}{j+1}\right) = O\left(\sum_j \frac{E^{(2)}_{\leq j}}{(j+1)^3} + \sum_j \frac{E^{(1)}_{\leq j}}{(j+1)^2}\right).$$

The preceding lemma implies that this bound is

$$O\left(\sum_j \frac{m(j+1)^2 + n(j+1)}{(j+1)^3} + \sum_j \frac{m(j+1) + n}{(j+1)^2}\right) = O(m \log n + n).$$

REMARK. The same result can be obtained by a "backward analysis" similar to that given by Chew in [7], or in the proof of Theorem 5.4 of Section 5.

We next show that this bound is tight in the worst case. Similar to the example given after Theorem 2.3, let $\mathscr{Q}$ be a set of m points lying on the negative portion of the x-axis, and let $\mathscr{P}$ be a set of n points lying on the positive portion of the line $y = 1$. If we add the points of $\mathscr{P}$ one by one in any order, then every time we add a point that is currently leftmost, it will have to be connected to each point of $\mathscr{Q}$ by a Delaunay edge. Since the expected number of right-to-left maxima in a random permutation of n points is $\Theta(\log n)$, it follows that the expected number of edges that will appear in the Delaunay triangulation during insertion of the points of $\mathscr{P}$ in random order is $\Theta(m \log n + n)$. This completes the proof of the theorem. □

3. An Efficient Incremental Algorithm. In this section we present a randomized incremental algorithm for calculating the Delaunay triangulation of a set of n points in the plane. The analysis of the algorithm makes use of and extends the probabilistic techniques developed in the preceding section.

The idea of the algorithm is quite simple. Suppose we have already inserted j points of $\mathscr{P}$, and are about to insert the $(j + 1)$st point, P. Suppose we already know the Delaunay triangle Δ (in the current triangulation) containing P. Then updating the triangulation in the presence of P is straightforward: we connect P

to the three vertices of Δ, and test each edge e of Δ as to whether it is still a valid Delaunay edge. For this it suffices to test whether P lies in the circumcircle of the triangle Δ' lying on the other side of e. If not, no update of the triangulation beyond e is necessary. Otherwise we delete e, connect P to the third vertex of the triangle Δ', and examine the two other edges of Δ' for validity. We continue this way until all edges that we encounter are valid, and then we stop. We remark that if P lies outside the convex hull of the first j points, the above triangle-flipping procedure requires some modifications. We can handle this issue by adding to $\mathscr{P}$ three dummy points "at infinity," Ω_1, Ω_2, Ω_3, whose spanning triangle contains all sites in $\mathscr{P}$, and by starting the incremental construction with the triangle $\Omega_1\Omega_2\Omega_3$. It is easily verified that the bounds derived in the preceding section do not change asymptotically when the construction is modified in this manner.

Thus we can write a high-level description of the algorithm as follows:

```
⟨Initialize the triangulation to the single triangle Ω₁Ω₂Ω₃⟩;
for k ← 1 to n do
  begin ⟨Select a random point P that has not previously been selected⟩;
  ⟨Find the triangle ABC containing P⟩;
  ⟨Replace ABC by the three triangles PAB, PBC, PCA⟩;
  X ← A;
  repeat ⟨Let Y be the third vertex of the triangle to the right of PX, in
          the current triangulation⟩;
    progress ← true;
    if X and Y not both infinite then
      begin ⟨Find Z ≠ P such that ZYX is in the current triangula-
            tion⟩;
      if in(P, X, Y, Z) then
        begin ⟨Flip triangles PXY and ZYX to obtain PXZ and
              PZY⟩;
        progress ← false;
        end;
      end;
    if progress then X ← Y;
  until X = A and progress;
  end.
```

Here the test in(A, B, C, D) is true if D is on the left of the oriented circle that goes from A to B to C to A. In terms of coordinates,

$$\mathrm{in}(A, B, C, D) \quad \Leftrightarrow \quad \det\begin{pmatrix} x_A & y_A & x_A^2 + y_A^2 & 1 \\ x_B & y_B & x_B^2 + y_B^2 & 1 \\ x_C & y_C & x_C^2 + y_C^2 & 1 \\ x_D & y_D & x_D^2 + y_D^2 & 1 \end{pmatrix} > 0.$$

If one or two of A, B, C, D is infinite, an appropriate limiting determinant should be calculated. A discussion of this and similar tests, as well as a proof of the validity of this algorithm appears in [20]. The triangulation can be maintained conveniently by using a simple form of the quad-edge data structure described in Section 4.1 of that paper.

The triangle-flipping procedure takes constant time for each edge it manipulates, including newly inserted edges (incident to P) and old edges that are deleted. Since an edge, once deleted, will never be re-introduced into the triangulation, the total expected work in updating the triangulation in this manner is $O(n)$, as follows from the previous section.

However, a major difficulty remains—how do we determine the Delaunay triangle containing P? We cannot afford to use point-location techniques, because they are not that simple to implement, and, worse still, we need to use *dynamic* point location, since we constantly update the triangulation. Although such techniques have recently been developed [27], they are complicated, and their update time, $O(\log^2 n)$, is too high. The other techniques mentioned in the introduction are also not ideal, because they require the maintenance of auxiliary data structures referencing points not yet inserted, which tends to complicate the algorithm.

Instead, we use the following simple approach. We maintain all versions of the triangulation on top of one another. More precisely, whenever we replace some triangle YZW by new triangles, we leave YZW as part of the structure, mark it as "old," and maintain a pointer from YZW to each of the newly generated triangles that partially overlaps it. Notice that the number of these new triangles is either two (when the deletion of YZW is caused by an edge flip) or three (when a new site X falls inside YZW, splitting it into three new triangles). It follows that the expected total number of pointers, and thus the expected size of the data structure, is only $O(n)$. (Note that this structure also contains "intermediate" triangles that have been generated during the insertion of a site and were then removed during the same insertion step; however, it is easily checked that the number of such triangles is also only linear in n—we can charge each of these triangles to the edge whose flipping has eliminated the triangle, and observe that no edge is charged more than once.)

When a new site P is to be added, we locate it in the current triangulation by tracing all triangles containing P in the chronological order of their creation. We start at the first enclosing triangle $\Omega_1\Omega_2\Omega_3$, and at each step check the two or three pointers from the present triangle to find the next newer triangle containing P, until we reach a triangle belonging to the current Delaunay triangulation.[6]

---

[6] This will require the use of another geometric predicate discussed in [20], the cc(A, B, C) test, defined to be true if the triangular path from A to B to C to A is oriented counterclockwise. In terms of coordinates this can be implemented as

$$\mathrm{cc}(A, B, C) \Leftrightarrow \det\begin{pmatrix} x_A & y_A & 1 \\ x_B & y_B & 1 \\ x_C & y_C & 1 \end{pmatrix} > 0,$$

similar to the in(A, B, C, D) test above.

What is the expected cost of this procedure? Consider the process of locating the site P as we insert it into the triangulation. Let XYZ be a triangle that is traced during the location of P. There is only one such triangle in the current triangulation, so we may assume that XYZ is "old." By construction, XYZ was removed from the triangulation at some earlier stage, due to one of the following events:

(i) A new site W has been inserted inside XYZ, causing it to be split into three subtriangles.
(ii) An edge flip has replaced XYZ and an adjacent triangle, say ZYW, by a pair of new triangles XYW, XWZ.

In case (i) the triangle XYZ was Delaunay before W was inserted. In case (ii) either XYZ was Delaunay, or ZYW was Delaunay, in which case X must have been the newly inserted site, and both X and P lie within the circumcircle of ZYW. Thus we can charge the tracing of an old triangle XYZ, during the insertion of P, either to the (earlier) removal of XYZ from the triangulation, or to the removal of an adjacent triangle. In either case, the charged triangle was Delaunay prior to the insertion step that has caused it to be removed, and P lies within the circumcircle of that triangle.

It follows that a triangle XYZ of scope k can be charged by at most k sites. It is also plain that no triangle can be charged more than once by the same site. Moreover, a necessary condition that a triangle XYZ is charged at all is that it arises as a Delaunay triangle at some stage during the incremental construction. Hence, the expected cost of locating all sites as they are inserted incrementally is at most

$$O\left(n + \sum_{j=0}^{n-3} \sum_{XYZ \in T_j} j \cdot \mathbf{Prob}[XYZ \text{ arises as a Delaunay triangle}]\right)$$
$$= O\left(n + 6 \sum_{j=0}^{n-3} \frac{j \cdot \|T_j\|}{(j+1)(j+2)(j+3)}\right)$$
$$= O\left(n + 12 \sum_{j \ge 0} \frac{\|T_{\le j}\|}{(j+2)(j+3)(j+4)}\right) = O\left(n + n \cdot \sum_{j=0}^{n} \frac{1}{j+1}\right) = O(n \log n).$$

We thus obtain the summary result:

THEOREM 3.1. *The incremental algorithm described above calculates the Delaunay triangulation of a set of* n *points in the plane in randomized expected time* $O(n \log n)$ *and linear expected storage.*

REMARK. Note that the bound in the preceding theorem is best possible, in the algebraic decision-tree model of computation, because we can reduce sorting to the problem of constructing the Delaunay triangulation of n points lying on some convex curve, and it is well known that the expected time complexity of any randomized sorting algorithm, in this model of computation, is $\Omega(n \log n)$. It is interesting to carry out the reduction from sorting and apply the above incremental algorithm to obtain a new randomized expected $O(n \log n)$ sorting algorithm. Specifically, given the real numbers $x_1, \ldots, x_n$, we transform them to the points (x_i, x_i^2), $i = 1, \ldots, n$, and apply the incremental algorithm to this set of points. The reader may find it interesting to work out the details of the resulting sorting algorithm, which is strongly related to Quicksort.

REMARK. What about an actual, comparably efficient, randomized algorithm for computing the Delaunay triangulation of $\mathscr{Q} \cup \mathscr{P}$, as in Section 2.3? There is of course an obvious and naïve solution—dismantle the triangulation of $\mathscr{Q}$ and construct the desired diagram from scratch. This will take expected time $O((m + n) \log(m + n))$. Curiously enough, unless m is much larger than n, this is the best asymptotic bound we can achieve. Indeed, the lower bound in Theorem 2.6 and the previous remark imply a lower bound of $\Omega((m + n) \log n)$ on the expected complexity of any randomized algorithm for an incremental construction of the desired triangulation, even assuming that the triangulation of $\mathscr{Q}$ is already available.

REMARK. The fact that the incremental algorithm needs quadratically many structural updates in the worst case has led some researchers to couple the incremental method with certain simple point-location algorithms that also take quadratic time in the worst case; see, for instance, [18]. The unfortunate thing about those point location methods is that they take $\Omega(n^{3/2})$ time even if we randomize over the sequence of sites *and* assume uniform distribution.

REMARK. The above analysis shows that the average expected cost of locating a site is $O(\log n)$. Unfortunately, there are situations where the expected number of triangles that are traced when locating a specific site P is linear in n. To see such an example, consider the following configuration, involving $n - 3$ points lying on the negative part of the x-axis, and three additional points X at (0, 1), Y at (1, 1), and P at $(0.5, 1 - \varepsilon)$, for a sufficiently small positive ε; see again Figure 1 (but now ignore all sites on or near $y = 1$ except X, Y, and P). Suppose the three top sites are inserted in the order Y, X, P (this event occurs with some constant positive probability that does not depend on n), and suppose further that m of the bottom sites have already been inserted before X is inserted. Then the insertion of X will create m new Delaunay edges, connecting X to all bottom sites; moreover, during this insertion step all the triangles XYb, where b ranges over all bottom sites, will be generated and, with the exception of the rightmost such triangle, immediately removed from the triangulation. Then, when P is inserted, the algorithm will need to trace all these triangles in order to locate P. Since the expected value of m is $\Theta(n)$, the claim follows.

REMARK. The preceding example raises the issue of modifying our point-location precedure so that it takes expected $O(\log n)$ time to locate any newly inserted site, or, for that matter, to locate any point q. One possible approach toward this goal might be not to keep around intermediate triangles in our data structure, but only those that are Delaunay in some version of the triangulation. Then, when tracing the triangles containing a newly inserted site P, we can move from one version of the triangulation to another by performing binary search through the new triangles in the latter version (all of which are incident to the same previously inserted site). In our current strategy, we may in effect carry out a sequential search through these triangles (this is the case in the preceding example). However, we do not know whether this improved strategy will always take only $O(\log n)$ expected time. We will see some similar ideas used in Section 5.

REMARK. As remarked in the introduction, the first instance of the idea of linking together all the Delaunay triangles that ever arise in the incremental construction is the *Delaunay tree*, proposed by Boissonnat and Teillaud [5]. In the details their construction differs substantially from ours—they need to maintain several more pointers.

4. Generalizations and Applications

4.1. Incremental Construction of Convex Hulls in Three Dimensions. The Delaunay triangulation of a set $\mathscr{P}$ of n points in the plane is a special case of the convex hull of a set of n points in three dimensions. This is shown by the following transformation: Lift each point $(x, y) \in \mathscr{P}$ to the point $(x, y, x^2 + y^2)$, that is onto the convex surface $z = x^2 + y^2$, defining a paraboloid of revolution. Let $\hat{\mathscr{P}}$ denote the resulting set, and let $LH(\hat{\mathscr{P}})$ denote the *lower convex hull* of $\hat{\mathscr{P}}$; this is the portion of the convex hull of $\hat{\mathscr{P}}$ consisting of all faces whose outward normals point downward. It is well known (see [20]) that the vertical projection of $LH(\hat{\mathscr{P}})$ onto the xy-plane gives the Delaunay triangulation of $\mathscr{P}$.

Hence a natural extension of our problem is: given a set $\mathscr{P}$ of n points in three dimensions, no three collinear and no four coplanar, we want to construct the lower convex hull of $\mathscr{P}$, starting with the empty set and adding the points of $\mathscr{P}$ one by one in random order. We show that this can be done in randomized expected time $O(n \log n)$, by providing an algorithm that is simpler (and has a considerably simpler analysis) than the recent algorithm of Clarkson and Shor [9] mentioned in the introduction. As above, our first goal is to estimate the number of triangles XYZ that appear as lower hull faces at some point during the process.

It is easy to modify the analysis of Section 2.1 so that it applies to this case as well. For a fixed triangle XYZ spanned by three points of $\mathscr{P}$, we define its *scope* to be k if there are k points of $\mathscr{P}$ lying below the plane containing XYZ. Clearly, triangles of scope 0 are exactly the faces of the lower hull. It is now easy to extend Lemma 2.1 to this set-up—the proof is essentially identical. Lemma 2.2 is also proved in a similar fashion, making use of the fact that the number of faces bounding the convex hull of r points in 3-space is $O(r)$. We thus obtain:

THEOREM 4.1. *During an incremental construction of the lower convex hull of n points in 3-space, where the points are inserted one by one in random order, the expected overall number of faces that appear on the lower hull at any stage during the process is $O(n)$.*

Next we describe an actual incremental randomized algorithm of constructing the lower convex hull of a set of n points. As above, we require no auxiliary structures. When a point P is inserted, we first locate the face Δ of the current lower hull lying directly above it. (If P lies above the hull, no action is needed and P is simply discarded; also, as before, we start the process with a dummy triangle $\Omega_1\Omega_2\Omega_3$ lying sufficiently high above all the points in $\mathscr{P}$.) We then add P to the hull using a propagation step that is similar to that in the construction of the Delaunay triangulation. That is, we remove Δ from the hull, and connect P to the three vertices of Δ. We now inspect each newly generated triangle together with its adjacent triangle that lies further from P. If this pair of triangles "bend downward," i.e., the lower dihedral angle between them is less than 180°, then we replace this pair by the opposite pair, and continue to propagate further, until no downward bends are found. Special treatment is required in case a pair of triangles that bend downward is such that their xy-projection forms a nonconvex quadrilateral. In this case we replace this pair by only a single triangle incident to P (this is the case when a site is being removed from the lower hull). We omit further details.

The point-location procedure is implemented as in the case of Delaunay triangulation. That is, we maintain all faces that are generated during the incremental process on top of one another. Whenever a face is removed from the hull, we store in it pointers to the (one, two, or three) new faces whose xy-projections partially overlap that of the old face. Theorem 4.1 implies that the expected size of this structure is $O(n)$. When a new point P is added to the hull, we locate it by tracing all faces that lie above or below P in chronological order of their creation, starting with the virtual face $\Omega_1\Omega_2\Omega_3$, and following the appropriate pointers. (If P is found to lie above any of these faces, we stop the tracing and discard P.) The analysis of the expected cost of locating all points as they are inserted is very similar to that given in Section 3, although more care is needed to handle some of the structural changes that can arise here but were not possible in the case of Delaunay triangulation.

In an entirely symmetric fashion we can also build the upper convex hull of the n points in this fashion. Putting the two together, we conclude that:

THEOREM 4.2. *The randomized incremental procedure given above for the construction of the convex hull of n points in 3-space takes expected time $O(n \log n)$ and expected space $O(n)$.*

REMARK. Our result is somewhat stronger than that of Clarkson and Shor [9], because we obtain in addition a linear bound on the expected number of hull updates that are required by the algorithm. However, it is possible to use their method to derive such a linear bound as well [8]. In addition, our method is more "on line" than theirs, since we only require a random permutation of the points and no other precomputation or auxiliary structures referencing sites not yet inserted (such as their conflict graph).

REMARK. As above, a similar analysis can be given based on edges.

REMARK. Of course all this implies that we can compute the *furthest-point* Voronoi diagram [20] (and its dual triangulation) within the same time and space bounds. This is the upper hull of the lifted points $(x, y, x^2 + y^2)$. We again omit further details.

4.2. Combinatorial Applications. Going back to the case of Delaunay triangulations, we can make a few observations concerning our proof techniques. Regarding Lemma 2.5, let us apply it with $k = \alpha n$, for some sufficiently small constant fraction α. The lemma implies that there are at most $O(\alpha n^2)$ edges whose symmetric scope is at most k. Hence, if α is sufficiently small, there has to exist at least one edge whose scope is greater than k. In other words, we have shown that for any set of n points in the plane there exist a pair of these points so that any circle passing through them contains at least αn points in its interior. This result is already known, and the constant that we get is not the best possible—Edelsbrunner *et al.* [15] have obtained a better estimate, using related techniques.

Lemma 2.5 also implies the following result, obtained independently in [1]. The j-order Voronoi diagram of the sites is the partition of the plane according to the j nearest neighbors of each point [13].

LEMMA 4.3. *The overall number of pairs X, Y such that the j-order Voronoi diagram contains a region whose set of sites includes both X and Y, for $j = 1, \ldots, k$, is $O(nk)$.*

PROOF. Let X, Y be a pair of sites that are both associated with one region of the j-order Voronoi diagram, but not with any region of any lower-order Voronoi diagram. From the definition of such a diagram it follows that there exists a circle passing through XY which contains $j - 1$ other points in its interior. Suppose that a of these points lie to the right of XY and $b = j - 1 - a$ points lie to the left, and suppose without loss of generality that $a \geq b$. If $a = b$, then we have established that $XY \in E_{(j-1)/2}$. Otherwise push the circle to the left, while maintaining its contact with X, Y. In this process, some points on the right may leave the interior of the circle and some additional points on the left may enter this interior. Eventually we will obtain a circle containing the same number of points on the left and on the right, and this number clearly cannot exceed j.

We have just shown that the edge XY defined by the pair X, Y must have symmetric scope at most k in the final Delaunay triangulation, so the total number of such pairs is $O(kn)$, by Lemma 2.5. □

REMARK. It is interesting to contrast this result with the bound $\Theta(nk^2)$ on the maximum total size of all j-order Voronoi diagrams, for $j = 1, \ldots, k$ (see [13]). The apparent contradiction is resolved by noting that a pair X, Y can be associated with a single region in many (actually linearly many) Voronoi diagrams.

REMARK. Extending Lemma 2.5 to the case of convex hulls in 3-space, we obtain the corollary that the number of edges XY connecting pairs among n given points in 3-space, such that there exists a plane passing through XY and containing at most k points below it, is $O(kn)$. Again, this has to be contrasted with the bound

$\Theta(nk^2)$ on the maximum number of triangles XYZ spanned by triples of the given points and having the property that the plane containing XYZ has at most k of the points below it (see [9]).

4.3. Delaunay Triangulations in Three Dimensions. Finally, consider the extension of our results to Delaunay triangulations in 3-space. We have a set $\mathscr{P}$ of n sites in 3-space, and we wish to maintain its Delaunay triangulation as we add the points one by one in random order. The problem is to estimate the number of tetrahedra $XYZW$ spanned by quadruples of the sites in $\mathscr{P}$ and appearing during the process in the Delaunay triangulation. The condition for this to happen is that, by the time and last of X, Y, Z, W is added, none of the sites lying in the interior of the circumsphere of $XYZW$ has yet been inserted. (As before, if the circumsphere contains no site in its interior, then $XYZW$ appears in the final Delaunay triangulation.)

As above, define the *scope* of a tetrahedron $XYZW$ spanned by four of the sites to be equal to the number of sites contained in the interior of the circumsphere of $XYZW$. An easy extension of the analysis presented in Section 2 yields:

LEMMA 4.4. *The probability that a tetrahedron $XYZW$ appears as a Delaunay tetrahedron during the incremental process is* $24/(k+1)(k+2)(k+3)(k+4)$, *where k is the scope of $XYZW$.*

LEMMA 4.5. *The number of tetrahedra $XYZW$ spanned by quadruples of sites and having scope at most k is* $O(n^2(k+1)^2)$.

PROOF. Apply an analysis similar to that of Lemma 2.2, and use the fact that the number of tetrahedra appearing in the Delaunay triangulation of r sites is $O(r^2)$. □

Now arguing as in the proof of Theorem 2.3, we obtain:

THEOREM 4.6. *The expected number of tetrahedra that appear in the Delaunay triangulation of a set of n points in 3-space during the incremental construction is* $O(n^2)$.

REMARK. Similar results were obtained in [9].

REMARK. In three dimensions, the bound $O(r^2)$ on the size of $D(\mathscr{R})$, although attainable in the worst case, is often too high. A challenging open problem is whether we can extend the preceding theorem to assert that the expected number of tetrahedra appearing in the triangulation during the process is proportional, or at least very close, to the actual size of the final Delaunay triangulation. Unfortunately, this is not possible in general. As a counterexample, consider a set $\mathscr{P}$ consisting of $2n+1$ points, n of which lie on the x-axis, n others lie on the line $x=0$, $z=1$, and the last point ζ lies at $(0, 0, \frac{1}{2})$. It is easy to see that, without the point ζ, the Delaunay triangulation is indeed of size $\Theta(n^2)$ (just pair up pairs of consecutive points, one pair on each of the two lines to obtain Delaunay tetrahedra). Moreover, any subset of the points of size r, not containing ζ and having at least a fraction of its points lying on each of the two skewed lines, will have a Delaunay triangulation of size $\Theta(r^2)$. However, if we space the points on the two lines sufficiently far apart we can guarantee that the circumspheres of all these Delaunay tertrahedra contain ζ and therefore, as soon as ζ is added, the complexity of the triangulation goes down to linear size. Thus, even though the output size (or the final triangulation) is linear, the expected number of tetrahedra appearing during the process will be quadratic.

Is there any hope for improvement? It appears that the crucial parameter is not the final output size, but the expected size of the Delaunay triangulation of samples of large cardinality. If we denote by $F(r)$ this expected value for samples of size r, then a variant of Lemma 2.2 can be used to show that the number of tetrahedra with scope at most k is $O(k^4 F(n/k))$. The proof technique of Theorem 2.3 then yields:

THEOREM 4.7. *The expected number of tetrahedra that appear in the Delaunay triangulation of a point set in 3-space during the incremental construction, is*

$$O\left(\sum_{k \geq 1} \frac{F(n/k)}{k}\right),$$

where $F(r)$ is the expected number of tetrahedra in the Delaunay triangulation of a sample whose size is r.

REMARK. A challenging task is to extend the techniques in Section 3 to obtain a simple randomized incremental algorithm for constructing three-dimensional Delaunay triangulations. Two main challenges arise:

(i) Obtain a tetrahedron-flipping procedure to update the diagram as a new point is inserted.[7]
(ii) Obtain an efficient point-location mechanism to accompany these flips, such as storing pointers from tetrahedra being removed to newly generated tetrahedra overlapping them, and locate a new site by tracing these pointers.

[7] An elementary flip in three dimensions is as follows: we are given two disjoint tetrahedra $ABCD$ and $ABCE$ sharing a common face ABC. Furthermore, the line segment DE cuts the face ABC. Then we replace $ABCD$ and $ABCE$ by the *three* tetrahedra $DEAB$, $DEBC$, and $DECA$ (or we do the reverse of this transformation). Now in two dimensions, if ABC and BCD are two disjoint triangles sharing edge BC, and D lies inside the circumcircle of ABC, *then AD must cut BC*, so the flip $(ABC, BCD) \to (ABD, ADC)$ is topologically legal. Unfortunately in three dimensions we can have a situation where E is inside the circumsphere of $ABCD$, but the flip $(ABCD, ABCE) \to (DEAB, DEBC, DECA)$ is not valid, since *DE does not cut* the triangle ABC.

Very recently, Barry Joe proved that the incremental tetrahedron-flipping procedure always works to produce the Delaunay triangulation in three dimensions (in particular, the topologically illegal situation described above cannot arise). Edelsbrunner and Shah have generalized this result to arbitrary regular triangulations in three dimensions (unpublished manuscripts).

REMARK. An intriguing open problem raised in view of the preceding theorem is: Given a set $\mathscr{P}$ of n points in 3-space, whose Delaunay triangulation is large, can we add to it a small number of extra points so that the new set has a small (say, linear size) Delaunay triangulation? Some results in this direction have been obtained very recently [6]. An even further challenging goal is to add points to $\mathscr{P}$ so as to obtain a set for which the expected size of the Delaunay triangulation of a sample of size r is small, say linear in r, for all "sufficiently large" values of r.

REMARK. Similar to the results in two dimensions, we can show that, given a set $\mathscr{P}$ of n points in 3-space, there always exists a triple of these points so that all spheres passing through three points contain at least a positive fraction of the points of $\mathscr{P}$. This result has already been obtained by Bárány *et al.* [3].

5. Randomized Incremental Construction of Voronoi Diagrams. In Section 3 we discussed an incremental randomized technique for computing the Delaunay triangulation of n sites in the plane. Because of the duality between Delaunay triangulations and Voronoi diagrams, a simple adaptation of our technique can also be used to compute the Voronoi diagram of n sites in the plane, within the same time and space bounds. (Under the duality Delaunay vertices correspond to Voronoi regions and vice versa, and Delaunay edges to Voronoi edges.) We can either build the Voronoi diagram only at the end, once we have the final Delaunay triangulation, or we can build the Voronoi diagram itself incrementally, updating it as the sites are added one at a time. The updating operations are all easily derived from the corresponding operations on the Delaunay triangulations. In fact, such incremental methods for Voronoi diagrams are commonly implemented.

We often construct Voronoi diagrams in order to use them for further processing. For example, the Voronoi diagram is usually computed as a means of solving the nearest-neighbor problem for n sites in the plane. However, once we have the Voronoi diagram we are not done when it comes to this application. In order to get efficient nearest-neighbor query time, we need to build on top of the Voronoi diagram a point-location structure. Several such structures are known that take $O(n)$ preprocessing time, use $O(n)$ storage, and allow point-location queries to be made in $O(\log n)$ time. The structures of [22] or [14] are well-known examples, though neither is especially simple to implement.

In this section our goal is to show that if we keep around all the intermediate Voronoi structures arising during the incremental construction process, and appropriately link them together, then we have an (expected) linear-size data structure that can be used for efficient point location in the final Voronoi diagram. Specifically, we show that the expected size of our structure is $O(n)$, and that, for any point p in the plane, the expected cost of locating the Voronoi region containing p is $O(\log^2 n)$. The structure is similar to that used to do point location for incremental Delaunay in Section 3. Now, however, we derive our query bound for *any* fixed point p in the plane. Similar constructions can be given for point location in arrangements of lines or planes built by randomized incremental techniques. These are discussed elsewhere [19].

The basic idea of our technique is to subdivide the Voronoi regions into smaller cells, so that during the incremental construction the number of different cells covering any particular point p is small. Such a finer subdivision is necessary. If the final region of site s has $\Omega(n)$ sides and p is very close to s, then the full Voronoi region in which p lies will change $\Omega(n)$ times, no matter in what order the sites are inserted.

We use a *radial triangulation* of each Voronoi region, obtained by connecting the defining site by line segments to each of the vertices of the region. Thus all of our cells are triangles. We say that such a triangle is *based* on the corresponding Voronoi edge. An example is shown in Figure 5(a). A crucial feature of this finer subdivision is that each cell depends on only a constant number of the sites (four, to be exact).

We now describe how our point-location structure $\mathcal{S}$ is built by connecting together such radial triangles as they arise during the incremental Voronoi construction. It is easiest to visualize $\mathcal{S}$ if we think of a series of n parallel and "aligned" planes in space, each plane containing one of the sites, in the (random) order in which the sites were inserted. For convenience we label the sites as 1, 2, ..., n, in their insertion order. On plane i we store all the new triangles created when the site i was first inserted. These include not only the triangles triangulating the Voronoi region of i at that time, but also new triangles created in neighboring regions that were affected by the addition of site i. Furthermore, all triangles destroyed by the insertion of site i are made to point to the plane containing i. We write $V_i(j)$, for $j \leq i$, to denote the Voronoi region of site j immediately after the insertion of site i and abbreviate $V_i(i)$ to $V(i)$. The following lemmas clarify the structure of $\mathcal{S}$.

LEMMA 5.1. *When site i is inserted, the Voronoi region $V_i(j)$ corresponding to site j, for $j < i$, can be affected at most by being intersected with a half-plane.*

PROOF. The new region of site j is the old region intersected with the half-plane consisting of points closer to site j than site i (see Figure 5(b)). □

Thus the insertion of site i creates a new Voronoi region for i by carving off portions from the regions of other sites. A subregion of each such unlucky site is sliced off by a line segment. These segments are portions of bisectors involving site i and proceed circularly around i to form the boundary of $V(i)$, while the boundaries of the sliced-off pieces of old regions form a tree partitioning the interior of $V(i)$.

With this understanding we can now examine more carefully the process of triangle destruction and creation (see Figure 5(b)). The triangles destroyed during the insertion of site i are exactly the triangles cut by the bisector segments discussed above (no old triangle can be completely covered by the new region $V(i)$, as each such triangle has a vertex on an old site j, $j < i$). Notice that all destroyed triangles, except for at most two per region, are based on Voronoi edges corresponding to a pair of Voronoi regions that were adjacent before the insertion of site i, but cease to be adjacent afterward. The exceptional triangles in a region are the ones where the bisector segment slicing that region terminates.

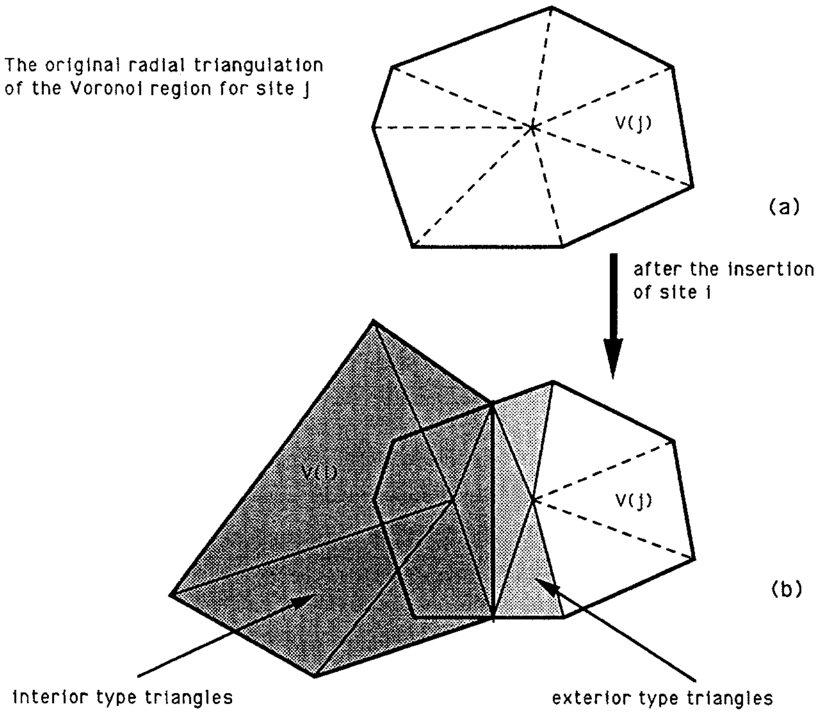

Fig. 5. Updating the radial triangulation of a Voronoi cell.

The newly created triangles when site i is inserted are of two types (see Figure 5(b)). For each site j whose Voronoi region is carved up by some bisector segment arising from i there will be exactly three new triangles that need to be created, as in Figure 5(b) (except for degeneracies—then only one or two new triangles might be created). This is the *exterior* type of triangle: there are three triangles of this type per neighbor of i in the Voronoi diagram of the first i sites. The remaining new triangles, those of the *interior* type, are those corresponding to the radial triangulation of $V(i)$. Conceptually we place all of these triangles on the parallel plane corresponding to site i. Notice that their number is proportional to the number of sides t in $V(i)$ (it is at most $4t$, in fact, by our analysis).

So to summarize, our point-location structure $\mathcal{S}$ consists of the planes of newly created triangles associated with the insertion of each site. On each plane i the triangles of the interior type on that plane are stored in an array ordered polarly around the site i. The triangles of the exterior type are just linked with the corresponding (by sharing a base) triangles of the interior type. With each triangle we also store the name of the site whose insertion destroys that triangle, or a special marker if that triangle partly survies all the way to the final Voronoi diagram.

THEOREM 5.2. *The expected size of the structure $\mathcal{S}$ for any group of n sites (under random ordering of the insertions) is $O(n)$.*

PROOF. When site i is inserted we can pay for all the triangle destruction and creation out of the structural changes that happen in the corresponding Delaunay triangulation.

A destroyed triangle of region $V_i(j)$ whose Voronoi edge is fully covered up by $V(i)$ corresponds to a Delaunay edge that must be deleted. A Delaunay edge cannot be charged more than twice in this fashion—it corresponds to the base for two triangles.

A destroyed triangle of region $V_i(j)$ whose Voronoi edge survives in the new diagram will give rise to a new triangle. Then we can charge the destruction of such a triangle to the new Delaunay edge $\{i, j\}$ that will be created. In fact, we can charge each new Delaunay edge $\{i, j\}$ at most six times and pay for all the triangle creation as well. We charge the edge $\{i, j\}$ from triangles based on it as follows:

- Two charges from destroyed triangles, portions of whose Voronoi edges survive.
- Three charges from the new triangles in the Voronoi region $V_i(j)$ (exterior type).
- One charge from the new radial triangle created in the region $V(i)$ (interior type).

Thus the destruction and creation can be paid out of the total number of structural changes in the Delaunay triangulation during the insertion process, which we know to be $O(n)$. The linking of triangles, etc., clearly also takes only $O(n)$ space. □

THEOREM 5.3. *The above structure can be built in $O(n \log n)$ expected time.*

PROOF. Every operation we need for the incremental construction of $\mathscr{S}$ parallels one of the steps we go through during the incremental construction of the Delaunay triangulation or the Voronoi diagram. We omit the easy details. Therefore the same time bound applies. □

THEOREM 5.4. *Let p be any fixed point of the plane. If the point-location structure $\mathscr{S}$ is constructed by inserting the sites in random order, then the expected cost of searching through $\mathscr{S}$ to locate the radial triangle containing p in the final Voronoi diagram is $O(\log^2 n)$.*

PROOF. We first show that the expected number of triangles covering p in $\mathscr{S}$ is $O(\log n)$. To see this it is best to think of the insertion algorithm as running backward in time. In the final Voronoi diagram the point p lies in some triangle. This triangle is determined by four of the sites. Thus, unless one of these sites was the last one to be inserted, the same triangle will already have existed in the Voronoi diagram of the first $n - 1$ sites. This implies that the probability that the last site to be inserted caused the triangle containing p to change is at most $4/n$. From this it follows that the expected number of triangles containing p during the whole process is $O(1 + \frac{1}{2} + \frac{1}{3} + \cdots + 1/n) = O(\log n)$.[8]

[8] In fact, in [5] it is shown that the expected number of Delaunay triangles whose circumcircle contains P is $O(\log n)$.

In the structure $\mathscr{S}$ we need to trace through all the triangles containing p. Say we are at a triangle containing p on plane j. This triangle refers us to the insertion that destroys it, say that of site i. On the plane of site i we locate the triangle containing p as follows. Since a triangle of site j is destroyed by the insertion of site i, it follows that i and j will be neighbors when i is inserted. We first do a radial binary search among all the interior triangles around i in $V(i)$ in order to find the sector containing p. Then we compare p with the bisector line of sites i and j. If i is closer to p than j, then the earlier binary search has identified the (interior) triangle containing p in $V(i)$ and we are done. Otherwise we sequentially compare p with each of the (at most) three new (exterior) triangles of $V_i(j)$ situated on the other side of the $\{i, j\}$ bisector. The base of the interior triangle determined by the radial binary search points to these exterior triangles. One of them must contain p, and we are again done.

Since we trace through $O(\log n)$ triangles, visit $O(\log n)$ planes, and we have a worst-case cost of $O(\log n)$ to locate a point in a plane, the overall expected cost is $O(\log^2 n)$. □

REMARK. It is tempting to try to reduce this to $O(\log n)$. Notice that instead of storing with a triangle on plane j just the name of the vertex i whose insertion destroys it, we can store instead a pointer to the interior triangle on plane i bordered by the bisector of i and j. Then we can test p against this bisector first and, if j is still closer to p, locate the new exterior triangle containing p in constant time. The difficulty is in the other case. When i is closer to p the interior triangle containing p in $V(i)$ need not be well correlated with the interior triangle containing p in $V(j)$ that we started from. Figure 6 shows an example.

REMARK. We might hope that a sequential walk through interior triangles will amortize to constant time per triangle, but this is not so. A counterexample can

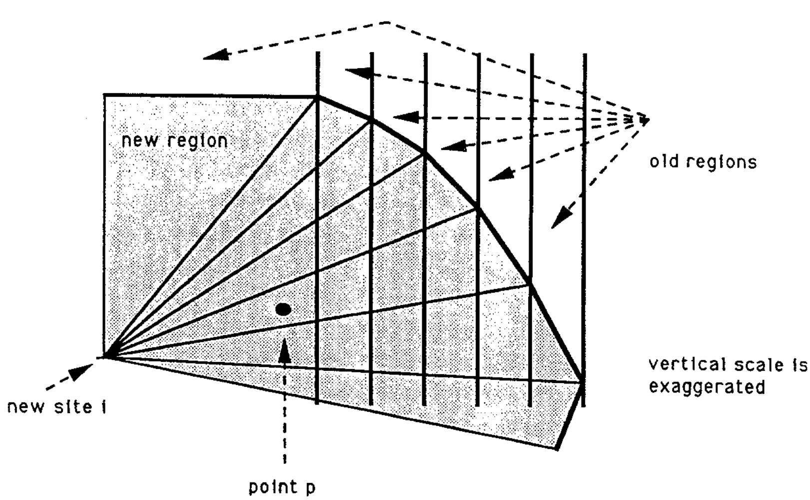

Fig. 6. The new triangle containing p can be hard to find.

be given using a point p close to a site whose region has $\Omega(n)$ sides in the final diagram, as before.

A three-dimensional analog of this method can be used for constructing the convex hull of n points in space so that in the end we have a structure that can be used to answer queries about (other) points lying inside or outside the hull. Again, if we randomize over the sequence of insertions the overall expected size of our structure will be $O(n)$, and the expected query cost for any particular point will be $O(\log^2 n)$.

Acknowledgments. This work has profited from valuable discussions with Ken Clarkson, Rajeev Motwani, Raimund Seidel, Peter Shor, and Jorge Stolfi.

References

[1] A. Aggarwal, H. Imai, N. Katoh, and S. Suri, Finding k points with minimum diameter and related problems, *Proc. 5th ACM Symp. on Computational Geometry*, 1989, pp. 283–291.
[2] G. Andrews, *The Theory of Partitions*, Addison-Wesley, Reading, MA, 1976. (*Encyclopedia of Mathematics and Its Applications*, Volume 2.)
[3] I. Bárány, J. Schmerl, S. Sidney, and J. Urrutia, A combinatorial result about points and balls in Euclidean space, *Discrete Comput. Geom.*, **3** (1988), 259–262.
[4] J. Bentley, B. Weide, and A. Yao, Optimal expected time algorithms for closest point problems, *ACM Trans. Math. Software*, **6** (1980), 563–580.
[5] J.-D. Boissonnat, and M. Teillaud, A hierarchical representation of objects: the Delaunay tree, *Proc. 2nd ACM Symp. on Computational Geometry*, 1986, pp. 260–268.
[6] B. Chazelle, H. Edelsbrunner, L. Guibas, R. Seidel, and M. Sharir, Selecting multiply covered points and reducing the size of Delaunay triangulations, Manuscript, 1989.
[7] P. Chew, The simplest Voronoi diagram algorithm takes linear expected time, Manuscript, 1988.
[8] K. Clarkson, Personal communication, 1989.
[9] K. Clarkson and P. Shor, Applications of random sampling in computational geometry, II, *Discrete Comput. Geom.*, **4** (1989), 387–421.
[10] B. Delaunay, Neue Darstellung der geometrischen Krystallographie, *Z. Kryst.*, **84** (1932), 109–149.
[11] B. Delaunay, Sur la sphère vide, *Izv. Akad. Nauk SSSR. Otdel. Mat. Estestv. Nauk*, **7** (1934), 793–800.
[12] R. Dwyer, Higher dimensional Voronoi diagrams in linear expected time, *Proc. 5th ACM Symp. on Computational Geometry*, 1989, pp. 326–333.
[13] H. Edelsbrunner, *Algorithms in Combinatorial Geometry*, Springer-Verlag, Heidelberg, 1987.
[14] H. Edelsbrunner, L. Guibas, and J. Stolfi, Optimal point location in a monotone subdivision, *SIAM J. Comput.*, **15** (1986), 317–340.
[15] H. Edelsbrunner, N. Hasan, R. Seidel, and X. Shen, Circles through two points that always enclose many points, Technical Report UIUCDCS-R-88-1400, University of Illinois at Urbana, January 1988.
[16] S. Fortune, A sweepline algorithm for Voronoi diagrams, *Algorithmica*, **2** (1987), 153–174.
[17] S. Fortune, A note on Delaunay diagonal flips, Unpublished manuscript.
[18] P. Green and R. Sibson, Computing Dirichlet tesselation in the plane, *Comput. J.*, **21** (1977), 168–173.
[19] L. Guibas and M. Sharir, History helps queries, In preparation.
[20] L. Guibas and J. Stolfi, Primitives for the manipulation of general subdivisions and the computation of Voronoi diagrams, *ACM Trans. Graphics*, **4** (1985), 74–123.

[21] J. Jaromczyk and G. Swiatek, Degenerate cases do not require more memory, Manuscript, 1989.

[22] D. Kirkpatrick, Optimal search in planar subdivisions, *SIAM J. Comput.*, **12** (1983), 28–35.

[23] D. E. Knuth, *The Art of Computer Programming*, Vol. 3, Addison-Wesley, Reading, MA, 1973.

[24] K. Mehlhorn, S. Meiser, and C. Ó'Dúnlaing, On the construction of abstract Voronoi diagrams, Manuscript, 1989.

[25] K. Mulmuley, A fast planar partition algorithm, Technical Report 88–107, Department of Computer Science, University of Chicago, May 1988.

[26] F. Preparata and M. Shamos, *Computational Geometry—An Introduction*, Springer-Verlag, Berlin, 1985.

[27] F. Preparata and R. Tamassia, Fully dynamic techniques for point location and transitive closure in planar structures, *Proc. 29th IEEE Symp. on Foundations of Computer Science*, 1988, pp. 558–567.

[28] R. Seidel, Private communication.

[29] G. Voronoi, Nouvelles applications des paramètres continus à la théorie des formes quadratiques. Premier Mémoire: Sur quelques proprieteés des formes quadratiques positives parfaites, *J. Reine Angew. Math.*, **133** (1907), 97–178.

[30] G. Voronoi, Nouvelles applications des paramètres continus à la théorie des formes quadratiques. Deuxième Mémoire: Recherches sur les parallélloèdres primitifs, *J. Reine Angew. Math.*, **134** (1908), 198–287; **136** (1909), 67–181.