Applications of Random Sampling in Computational Geometry, II

extended abstract

Kenneth L. Clarkson
AT&T Bell Laboratories
Murray Hill, New Jersey 07974

Abstract

Random sampling is used for several new geometric algorithms. The algorithms are "Las Vegas," and their expected bounds are with respect to the random behavior of the algorithms. One algorithm reports all the intersecting pairs of a set of line segments in the plane, and requires $O(A + n \log n)$ expected time, where A is the size of the answer, the number of intersecting pairs reported. The algorithm requires $O(n)$ space in the worst case. Another algorithm computes the convex hull of a point set in E^3 in $O(n \log A)$ expected time, where n is the number of points and A is the number of points on the surface of the hull. A simple Las Vegas algorithm triangulates simple polygons in $O(n \log \log n)$ expected time. Algorithms for halfspace range reporting are also given. In addition, this paper gives asymptotically tight bounds for a combinatorial quantity of interest in discrete and computational geometry, related to halfspace partitions of point sets.

1 Introduction

In recent years, random sampling has seen increasing use in discrete and computational geometry, with applications in proximity problems, point location, and range queries [Cla85, Cla87, HW87]. These applications have largely used random sampling for divide-and-conquer, to split up problems into subproblems each guaranteed to be of small size. In this paper, random sampling will be used in a similar way, with the additional observation that the total of the sizes of the subproblems is small on the average. This fact will allow improvement in bounds for certain randomized algorithms, and for a combinatorial quantity.

The results in this paper have been used in algorithms for computing diametral pairs[CS88], triangulating simple polygons[CTVW88], and in various combinatorial results on arrangements[CEG⁺88]. These ideas are used in [CS88] to show that a simple, general technique for computing geometric structures is asymptotically fast. This technique is applied in this paper, replacing earlier and more complicated versions of the algorithms.

The dimension d is considered to be a constant. The expected resource bounds indicated are "Las Vegas," and the expectations are with respect to the random behavior of the algorithms. The parameter A is generally used to denote the size of the Answer to a computation. The inputs to the algorithms will be assumed nondegenerate, so an input set of line segments has no three intersecting at the same point, an input set of points in E^d has no $d+1$ coplanar, and so on. This is no loss of generality, as small random perturbations assure the condition with probability 1, and the answer sizes A as defined are unchanged. (See also [EM88, Yap88].)

1.1 The problems and results

An algorithm for line segment intersections. Let S be a set of n line segments in the plane. The problem of determining the pairs of segments in S that intersect has received much attention, culminating in the recent algorithm of Chazelle and Edelsbrunner requiring $O(A + n \log n)$ time in the worst case to report the A intersecting pairs. Their algorithm requires (moderately) sophisticated data structures and many sophisticated algorithmic techniques, and $\Omega(n + A)$ space. This paper gives a Las Vegas algorithm for this problem. The algorithm requires

$O(A + n \log n)$ expected time and $O(n)$ space in the worst case. The only data structure used by the algorithm is a subdivision of the plane into simple regions induced by the line segments.

Convex hulls. A Las Vegas algorithm is given for computing the convex hull of n points in E^3. The algorithm requires $O(n \log A)$ expected time for any set of points in E^3, where A is the number of points of S on the surface of the hull. Kirkpatrick and Seidel obtained a deterministic algorithm for planar convex hulls with the same time bound [KS86]

An algorithm for triangulation of simple polygons. Triangulation of a simple polygon is a preliminary step for many algorithms on such polygons. For many problems, triangulation is the only step not known to require $O(n)$ time. Therefore, triangulation is a particularly intriguing problem. Tarjan and Van Wyk [TVW88] have given an algorithm for this problem requiring $O(n \log\log n)$ time in the worst case. Their algorithm is quite complex. This paper gives a simple Las Vegas algorithm that requires $O(n \log\log n)$ expected time. This algorithm has recently been refined to expected $O(n \log^* n)$. [CTVW88]

Tight bounds for k-sets. Let $S \subset E^d$ contain n points. A set $S' \subset S$ with $|S'| = j$ is a *j-set* of S if there is a hyperplane that separates S' from the rest of S. A j-set is a $(\leq k)$-set if $j \leq k$. Let $g_k(S)$ be the number of $(\leq k)$-sets, and let $g_{k,d}(n)$ be the maximum value of $g_k(S)$ over all n-point sets $S \subset E^d$.

This paper shows that

$$g_{k,d}(n) = \Theta(n^{\lfloor d/2 \rfloor} k^{\lceil d/2 \rceil}),$$

as $n/k \to \infty$, for fixed d. The proof technique for the combinatorial bound can also be applied to give $(\leq k)$-set bounds for independently identically distributed points. For example, if the convex hull of such a set of points has $O(f(n))$ expected facets, then the expected number of $(\leq k)$-sets is $O(k^d f(n/k))$. The proof technique employed for the improved bounds is an instance of a "probabilistic method" [ES74].

The concept of a k-set is a generalization of the concept of a convex hull facet, which can be viewed as a d-set. The new bound is a generalization of the known upper bound $O(n^{\lfloor d/2 \rfloor})$ for the number of facets of a convex polytope with n vertices. Indeed, the new bound follows from this polytope upper bound.

Goodman and Pollack showed [GP84] that for $k < n/2$, the bound $g_{k,2}(n) \leq 2nk - k^2 - k$ holds. This was improved to $g_{k,2}(n) \leq nk$ in [AG86], and that bound was proven tight in the same paper. (See also [Wel86] for a related result for the plane.)

Cole and others [CSY84] showed that $g_{k,3}(n) = O(n^2 k)$, and Chazelle and Preparata [CP86] showed that $g_{k,3}(n) = O(nk^5)$. In [Cla87], it was shown that $g_{k,3}(n) = O(nk^2 \log^8 n / (\log\log n)^6)$. The new bound $O(nk^2)$ is thus an improvement over all previous results for $d = 3$; no bounds were known before for higher dimensions. The proof of the bound is also considerably simpler than those given for these earlier, weaker bounds.

Improved bounds for halfspace range reporting. The halfspace range reporting problem is to build a data structure for a set S of points, so that given a query halfspace the points of S in the halfspace can be reported quickly. The new bound for k-sets is applied in this paper to sharpen the analysis of the algorithm of [CP86] for halfspace range reporting. It is also used to analyze two new algorithms for that problem. One algorithm is shown to require expected $O(n^{\lfloor d/2 \rfloor + \epsilon})$ preprocessing time, and in the worst case $O(n^{\lfloor d/2 \rfloor + \epsilon})$ storage. The resulting query time is $O(A + \log n)$, where A is the size of the answer to the query. These resource bounds apply for any fixed $\epsilon > 0$, and the constant factors in the bounds depend on d and ϵ. Another algorithm requires $O(n)$ storage, $O(n \log n)$ expected preprocessing time, and allows queries to be answered in $O(A + n^{1+\epsilon-\gamma})$ time, where $\gamma = 1/(1 + (d-1)\lfloor d/2 \rfloor)$. The algorithm is a variant of Haussler and Welzl's [HW87]. Their query time is $O(n^{1+\epsilon-\gamma'})$, where $\gamma' = 1/(1 + d(d-1))$. (This is independent of the answer size, however.)

1.2 Outline of the paper

The remainder of this section gives an informal discussion of the ideas in this paper, followed by a description of the formal framework used in the theorems. The next section gives a proof of a general theorem that implies the asymptotically tight bound for k-sets. Some applications of the k-set bound to halfspace range reporting are shown. The proof technique for the k-set bound is closely related to ideas used in §3 for convex hulls, line segment intersection, and triangulation of simple polygons.

1.3 The ideas

To give an impression of the ideas used in this work, versions of them will be given for some simple cases. Consider a set S of n real numbers, and random $R \subset S$ of size r. The subset R gives a lot of information about S. For example, let $S_>$ be the set of numbers in S that are greater than all the numbers in R. Then with high probability, the number of values in $S_>$ is small, that is, no more than $O(\log r)n/r$. Similarly, with high probability, the number of values in a corresponding set $S_<$ is $O(\log r)n/r$. Indeed, with high probability, both sets are small. These statements are "tail estimates," bounds on the probability that a random variable will exceed a certain value. Another kind of statement about a random variable is its expectation. For example, the sizes of these sets are expected $O(n/r)$.

Both of these kinds of statements can be generalized to higher dimensions. The convex hull of a set in E^1 is just the interval between the largest and smallest values in that set, so the above statements are claims about the convex hull of R. Now suppose S is a set of points in the plane. Let e be an edge of the convex hull of R, and let l be the straight line containing e. The points in the halfspace bounded by l, and not containing R, will be said to be *beyond* the edge e. An analog to the first claim above is this: with high probability, for every edge e of the convex hull of R, the number of points of S that are beyond e is $O(\log r)n/r$. This kind of tail estimate has been the basis of several previous applications of random sampling to computational geometry.

It is also possible to generalize the second kind of claim about R and S, as follows: if S_e is the set of points in S that are beyond e, then the expected value of the sum of $|S_e|$, over all edges e of the hull of R, is $O(n)$. Thus a tighter bound is obtainable about the sizes $|S_e|$ on average than can be made about any particular one. This observation is a key new idea in this paper.

Why is this kind of bound useful in computing convex hulls? The idea is to take a large random subset $R \subset S$, recursively compute the convex hull of R, and then determine the result of adding the remaining points $S \setminus R$. Roughly speaking, the changes owing to the remaining points can be expected to be "local," and require a small amount of work per point. Indeed, for a given point $p \in S \setminus R$, the number of such changes is proportional to the number of edges of the hull of R that p is beyond. The total of these changes is therefore expected $O(n)$, as in the above discussion.

The convex hull algorithm will be given as an algorithm for determining the intersection of a set of halfspaces. This problem is linear-time equivalent to the convex hull problem via well-known duality mappings. The above statements translate as follows to halfspace intersections: suppose S is a set of n halfspaces, and $R \subset S$ is a random subset of S of size r. Suppose v is a vertex of the convex polytope that is the intersection of the halfspaces in R. Then the expected number of halfspaces of S that do not contain v is $O(n/r)$. Intuitively, because all the halfspaces of R contain v, it is likely that most of the halfspaces of S do as well.

Similar observations are also useful in other intersection problems, such as the problem of determining the number of intersecting pairs in a set of n line segments in the plane. Suppose S is a set of n line segments in the plane, and $R \subset S$ is a random subset of S of size r. The set R induces a subdivision of the plane that will be called the "vertical visibility map," or $\mathcal{V}(R)$. This subdivision is defined as follows: for every point p that is either an endpoint of a segment in R, or an intersection point of two segments in R, extend a vertical segment from p to the first segment of R above p, and to the first segment of R below p. If no such segment is "visible" to p above it, then extend a vertical ray above p, and similarly below. The resulting vertical segments, together with the segments in R, form a subdivision of the plane into simple regions that are generally trapezoids. (So $\mathcal{V}(R)$ is also known as the "trapezoidal diagram" of R.) No segment of R intersects the interior of any such region Q. This implies (intuitively) that few segments of S intersect the interior of Q. This gives a way to "divide-and-conquer" the line segment intersection problem. Note that the number of regions in the subdivision is proportional to $r + A'$, where A' is the number of intersecting pairs of segments in R. It is easy to show that the expected value of A' is $AO(r^2/n^2)$.

For the combinatorial problem of bounding the number of k-sets, it is helpful to use a kind of converse to the above relation between the convex hulls of point sets $R \subset S \subset E^2$. That is, let two given points in R define some line l, that divides S into j and $n-2-j$ points. The number of such pairs of points of S is the same, up to a constant, as the number of j-sets of S. If $j < n/r$, then the probability is not very small that none of the other points of R will be chosen from the j points on one side. In such a case, the two given points of R will define an edge of the hull of R. Roughly speaking, if the number of $(\leq n/r)$-sets of S is too large, then the number of edges of the hull of R will be greater than r, which of course cannot happen. That is, a bound on the complexity of the convex hull of R implies a bound on the number of $(\leq n/r)$-sets of S. It is this relationship that is exploited in this paper.

1.4 The formal framework

The ideas in this paper can be applied to a variety of geometric structures. To aid and to show this generality, a formal and abstract framework for geometric computations is useful. This framework is much the same as that in [Cla87].

Let S be a set of n subsets of E^d, which we will term *objects*. The set S corresponds to the input to the computation, and could be a set of points (singleton sets), line segments in the plane, halfspaces, or balls. Let $\mathcal{F}$ be a set of subsets of E^d, which we will term *regions*. The regions will be defined by the objects in some way for an application. For convex hulls in the plane, the objects are points, and the regions are halfplanes, so the regions defined by the objects are those open halfplanes that are bounded by lines through pairs of the points. The notion of "defined" is formalized as follows: for integer b, let δ be a relation between $\mathcal{F}$ and $\cup_{i\leq b}S^i$. A region F is "defined" by $X \in S^i$ if $F\delta X$, that is, if the δ relation holds between X and F. The set of regions $\mathcal{F}_S$ defined by the objects in S is formally

$$\mathcal{F}_S = \{F \in \mathcal{F} \mid F\delta X, X \in S^i, i \leq b\}.$$

In problems of construction, the desired computation is this: determine all $F \in \mathcal{F}_S$ such that $F \cap s$ is empty for all $s \in S$. For convex hulls, this is the set of all open halfplanes that contain none of the input points. For Voronoi diagrams in the plane, the regions are open disks or halfspaces, and the empty members of $\mathcal{F}_S$ are Delaunay disks or empty halfspaces as for the convex hull. The relation δ has a 3-tuple of points defining the open disk bounded by the circle circumscribing the points, and a 2-tuple of points defining the two open halfspaces bounded by the line through the points. For trapezoidal diagrams of line segments in the plane, the objects are line segments, the regions are open "trapezoids," and the parameter b is four. The relation δ would define the trapezoids using various configurations of four or fewer segments, as such trapezoids would appear in a trapezoidal diagram.

While the desired regions in construction problems have no intersections with S, it also will be useful to consider the number of objects of S that have nonempty intersection with a region $F \in \mathcal{F}_S$. Let $C_F(S)$ denote this number, and similarly $C_F(R)$ denotes the number of objects of R that have nonempty intersection with F. For a given integer j, let $\mathcal{F}_S^j$ denote the set of $F \in \mathcal{F}$ with $C_F(S) = j$. Thus in construction problems, the set $\mathcal{F}_S^0$ is desired. Note that with S and $\mathcal{F}$ as for convex hulls, as mentioned above, the set $\mathcal{F}_S^j$ is closely related to the set of j-sets of S, and $|\mathcal{F}_S^j|$ is the number of such sets.

For $R \subset S$, the sets $\mathcal{F}_R$ and $\mathcal{F}_R^j$ are defined analogously: $\mathcal{F}_R$ is the collection of all regions F such that $F\delta X$ for some $X \in R^b$, and $F \in \mathcal{F}_R^j$ has $C_F(R) = j$.

We may have some other requirements on the δ relation, but the essential condition on this relation is that $\mathcal{F}_S^0$ contains all the regions of interest in the given problem.

2 Improved bounds for k-sets, and applications

2.1 The bound

Rather than prove an upper bound for k-sets only, a much more general result will be proven, from which the k-set bound will follow as a corollary.

Before the main theorem, a technical lemma involving the function $p_j(n,r,b) = \binom{n-j-b}{r-b}/\binom{n}{r}$. This function is a lower bound on the probability that $F \in \mathcal{F}_R^0$, given that $F \in \mathcal{F}_S^j$, when R is a random subset of S: $F \in \mathcal{F}_R^0$ if and only if some b or fewer objects defining F are in R, and the $C_F(S) = j$ objects meeting F are not in R. There are at least $\binom{n-j-b}{r-b}$ subsets of S with these properties, out of $\binom{n}{r}$ subsets.

Lemma 2.1 *For nonnegative integers n, b, $k > 1$, $r = \lfloor n/k \rfloor$, and $j \leq k$, the bound*

$$p_j(n,r,b) \geq \frac{1}{e^{3/2}k^b}(1 + O(k/n))$$

holds as $k/n \to 0$.

Proof. Omitted in this abstract. □

Theorem 2.2 *With the notation of §1.4, let $R \subset S$ be a random subset of size $r = \lfloor n/k \rfloor$, with $k > 1$. Then*

$$\sum_{0\leq j\leq k} |\mathcal{F}_S^j| \leq k^b e^{3/2} E[|\mathcal{F}_R^0|](1 + O(k/n)),$$

as $k/n \to 0$.

Proof. For $F \in \mathcal{F}_S$, let I_F be the indicator function for the event $F \in \mathcal{F}_R^0$, so $I_F = 1$ when $F \in \mathcal{F}_R^0$ and 0 otherwise. Then $|\mathcal{F}_R^0| = \sum_{F\in\mathcal{F}_S} I_F$, so

$$\begin{aligned} E[|\mathcal{F}_R^0|] &= E[\sum_{F\in\mathcal{F}_S} I_F] = \sum_{F\in\mathcal{F}_S} E[I_F] \\ &= \sum_{\substack{j\geq 0 \\ F\in\mathcal{F}_S^j}} \mathrm{Prob}\{F \in \mathcal{F}_R^0\} \\ &\geq \sum_{j\geq 0} p_j(n,r,b)|\mathcal{F}_S^j| \\ &\geq \sum_{0\leq j\leq k} |\mathcal{F}_S^j|/k^b e^{3/2}(1 + O(k/n)), \end{aligned}$$

The first inequality is from the discussion above of $p_j(n,r,b)$. The second inequality uses the technical lemma. □

The theorem gives a bound on the quantity $\hat{g}_{k,d}(n)$, defined as follows: suppose $S \subset E^d$ is a set of n

points, $\mathcal{F}$ is the set of halfspaces in E^d, and $\mathcal{F}_S$ is the set of halfspaces bounded by hyperplanes that are affine hulls of points in S. Then $\hat{g}_{k,d}(n)$ is the maximum, over all such S, of $\sum_{0\le j\le k}|\mathcal{F}_S^j|$. A member of $\mathcal{F}_S^j$ will be called a *j-facet* of S.

Corollary 2.3

$$g_{k,d}(n) = O(n^{\lfloor d/2\rfloor}k^{\lceil d/2\rceil}),$$

as $n \to \infty$.

Proof. It is easy to show, using the results of [Ede87, §3.2], that $\hat{g}_{k,d}(n) = \Theta(g_{k,d}(n))$ as $n \to \infty$.

To apply the theorem to $\hat{g}_{k,d}(n)$, S is a set of points in E^d (or more precisely, a collection of singleton sets of points of E^d). The set of regions $\mathcal{F}$ is the set of open halfspaces of E^d, and with $b = d$, the δ relation is defined as follows: for $X \in S^b$, let $F\delta X$ when F is bounded by the affine hull of the points in X. The upper bound for $\hat{g}_{k,d}(n)$ follows, using the upper bound $O(r^{\lfloor d/2\rfloor})$ for $|\mathcal{F}_R^0|$, here the number of facets of a polytope with r vertices [Ede87, §6.2.4] □

Lemma 2.4

$$g_{k,d}(n) = \Omega(n^{\lfloor d/2\rfloor}k^{\lceil d/2\rceil}),$$

as $n \to \infty$.

Proof. Omitted. Cyclic polytopes realize the bound, as can be shown using the techniques of the theorem, or constructively [Ede]. □

2.2 Algorithms for Half-Space Range Queries

In this section, the new k-set bound gives an improved storage bound for a deterministic algorithm for halfspace range search in E^3. Two new randomized algorithms for range search in E^d are given and analyzed.

Given n points $S \subset E^d$, the halfspace range search problem is to build a data structure for S so that given a query halfspace h^*, the set of points $h^* \cap S$ can be reported quickly. In [CP86], Chazelle and Preparata show that in the case $d = 3$, a data structure requiring $O(n(\log n)^8(\log\log n)^4)$ storage can be constructed that allows queries to be answered in $O(A + \log n)$ time, where A is the number of points in the answer to the query. Theorem 1 of that paper implies that if $\hat{g}_{k,3}(n) = O(nk^\beta)$, then the storage required by their data structure is $O(n(\log n)^{2(\beta-1)}(\log\log n)^{\beta-1})$. The bound given here implies that $O(n(\log n)^2\log\log n)$ storage is sufficient.

The upper bound on $\hat{g}_{k,d}(n)$ gives a bound on a randomized algorithm for halfspace range search in E^d. This algorithm is conveniently described by putting the range query problem into a dual form, using the transform $\mathcal{D}$ described in [Ede87, §3.1], to which the reader is referred for background. The transform $\mathcal{D}$ maps non-vertical hyperplanes in E^d to points in E^d, and vice versa. (Here "non-vertical" means that the hyperplane does not contain a vertical line. A vertical line is a translate of the x_d-axis.) Under this duality, incidence and order are preserved, so that point p is on plane h if and only if $\mathcal{D}(h) \in \mathcal{D}(p)$, and also p is in the halfspace h^+ if and only if $\mathcal{D}(H) \in \mathcal{D}(p)^+$. In this setting, a point set S gives rise to an arrangement of hyperplanes $\mathcal{A}$ by the duality transform. Given a query halfspace h^-, the answer to the query is the set of all hyperplanes $\mathcal{D}(p)$ in $\mathcal{A}$ such that $\mathcal{D}(h)$ is below them, that is, $\mathcal{D}(h) \in \mathcal{D}(p)^-$.

We will be interested in the set of all points that are below no more than k hyperplanes of $\mathcal{A}$, for some k. These points correspond to the set of all query halfspaces h^- whose answer set has no more than k points. The lower surface of this set of points is called a k-level. It is not too hard to see that the number of vertices of cells above a k-level is bounded by $\hat{g}_{k,d}(S)$. This value asymptotically bounds the total complexity of the cells of $\mathcal{A}$ above the k-level.

The main idea for the algorithm range search is the following generalization and restatement of Lemmas 5.3 and 5.4 of [Cla87].

Lemma 2.5 *Suppose $S \subset E^d$ in general position, with $|S| = n$. Let $R \subset S$ be a random subset of size r. Then there is an integer $j_* = O(\log r/\log\log r)$ and a value $\alpha_* = O(\log r/r)$, such that with probability at least $1/2$, the following holds: the j_*-level of R is below the $n/(r-d^2)$-level of S, and strictly above the $\alpha_* n$-level of S.*

Proof. Omitted. □

Let us assume that r is constant (though "sufficiently large"). A given random subset can be tested for satisfying these conditions in $O(\hat{g}_{j_*,d}(r)n)$ time, which is $O(n)$ for fixed r. Thus, by repeatedly sampling, a suitable sample can be found in two trials, on the average.

Suppose that for a query halfspace h^-, the point $\mathcal{D}(h)$ is below the j_*-level of R, where R is now a suitable sample. (We will assume that the query halfspace contains the $-\infty$ point of the x_d-axis. Symmetric processing must be done for positive halfspaces.) Then $\mathcal{D}(h)$ is below the $n/(r-d^2)$-level of $\mathcal{A}$, and the query has answer size $A = \Omega(n)$. Here sophistication doesn't pay, and linear search through S determines the answer in $O(n) = O(A)$ time. On the other hand, suppose $\mathcal{D}(h)$ is above or on the j_*-level of R. Then we will recursively search a data structure for S, which is built as follows: triangulate the cells of the arrangement of R that are above the j_*-level of R. By the results of [Cla87], this yields $O(\hat{g}_{j_*,d}(r))$ simplices, as $r \to \infty$. For any simplex T, let $S(T)$ be the set of hyperplanes of $\mathcal{A}$ that are above any point of T. Recursively build a search structure for $S(T)$, for all such simplices T. To answer a query when $\mathcal{D}(h)$ is above the j_*-level of R, search the data structure for the simplex containing $\mathcal{D}(h)$.

The resulting data structure has a query time of $O(A + \log n)$, and a storage bound $B(n)$ satisfying

$$\begin{aligned} B(n) &\leq O(n) + O(\hat{g}_{j_*,d}(r))B(\alpha_* n) \\ &\leq O(n) + O(r^{\lfloor d/2 \rfloor} j_*^{\lceil d/2 \rceil})B(O(\log r)n/r), \end{aligned}$$

as $r \to \infty$. The given bound $O(n^{\lfloor d/2 \rfloor + \epsilon})$ follows, using the k-set results above. The expected time required by the algorithm to build the data structure satisfies the same bound.

Lemma 2.5 and the upper bound on $\hat{g}_{k,d}(n)$ can be applied in another way to obtain an algorithm for halfspace range queries that requires less storage and preprocessing, at the cost of a longer query time. Consider the arrangement $\mathcal{A}'_{j_*}$ of hyperplanes defined by the $(\leq j_*)$-facets of R. These hyperplanes are the duals of the vertices of the cells of $\mathcal{A}$ that are above or on the j_*-level of $\mathcal{A}$. The cells of this arrangement induce a partition of S. For each cell C, we recursively build a data structure for the points $C \cap S$.

In answering a query, as above we can tell quickly when the size of the answer is more than $n/(r-d^2)$, in which case a naive algorithm should be used. If the answer is smaller, then the data structure must be used. The cells of the arrangement $\mathcal{A}'_{j_*}$ are examined. Some do not intersect the query hyperplane, and so contribute either all or none of the points they contain to the answer. The remaining cells, that do meet the query hyperplane, must be examined recursively.

From previous analysis [HW87, AHW87], there are two key properties of this algorithm that imply a bound on the query time. The first is that the number of cells cut by a given query hyperplane is $O(\binom{g_{j_*,d}(r)}{d-1})$, the complexity of the subdivision of the query hyperplane by the hyperplanes of $\mathcal{A}_{j_*}$. The second property is that the total number of points in the cells examined for a query is no more than $(d+1)\alpha_* n$. Observe that when the dual point of a query hyperplane is above the j_*-level of R, that point is a convex combination of $d+1$ vertices of $\mathcal{A}$ above the j_*-level. This implies that the query halfspace h^- is contained in the union of $d+1$ halfspaces bounded by $(\leq j_*)$-facets of R. This union of halfspaces is also a union of cells of $\mathcal{A}_{j_*}$. Hence the total number of points in the cells examined for the query is no more than $(d+1)\alpha_* n$.

The resulting query time is [HW87, AHW87] $O(n^\beta)$, where $\beta = 1 - 1/(1+B)$, and

$$B = \frac{\log \binom{r^{\lfloor d/2 \rfloor} j_*^{\lceil d/2 \rceil}}{d-1}}{-\log(d+1)\alpha_*},$$

so $\beta = 1 - \gamma + \epsilon$, where $\gamma = 1/(1 + (d-1)\lfloor d/2 \rfloor)$, and $\epsilon > 0$ is independent of n and decreasing in r. The preprocessing time is expected $O(n \log n)$ and the storage is $O(n)$, as is easy to verify.

3 Line Segment Intersections, Convex Hulls, and Triangulation

3.1 The probabilistic theorem

This section gives a theorem showing that random sampling can be used effectively for divide and conquer. For an introduction to closely related ideas, see [Cla87]. A corollary is also given that combines the results of [Cla87] with the theorem.

A technical lemma will be needed regarding a function related to the hypergeometric probability distribution. This function is $p_j(n, r, b) = \binom{n-b-j}{r-b} / \binom{n}{r}$, as in Lemma 2.1.

Lemma 3.1 *For integers i, j, r, n, b with $r \leq n/2$, $i \geq 0$, and $j \geq in/r$, let $r_i = \lfloor (r-b)/(i+1) \rfloor + b$. Then*

$$p_j(n, r, b)/p_j(n, r_i, b) \leq (i+1)^b e^{-i/2} e^{2/3}(1 + O(i/r)).$$

Proof. Omitted in this abstract. □

To state the main theorem needed, some terminology and notation in addition to that in §1.4 is useful:

Call the relation δ a *function up to permutation* if for every $F \in \mathcal{F}_S$, the set $\{X \in \cup_{i \leq b} S^i \mid F \delta X\}$ is either empty, or has elements that are i-tuples from the same i-element $T_F \subset S$, for some $i \leq b$. With this condition on δ, any $F \in \mathcal{F}_S$ is in $\mathcal{F}_R$ if and only if $T_F \subset R$. This requirement on δ makes necessary the nondegeneracy assumption for the inputs in the algorithmic applications.

A function $w(j)$ on the integers will be called *submultiplicative* when $w(ij) \leq w(i)w(j)$ for sufficiently large ij. Note that $w(j) = j \log j$ and $w(j) = j^m$ are submultiplicative.

Let $\mathcal{P} \subset \mathcal{F}$, a predicate on the regions. For $R \subset S$, let $T_w(R)$ denote

$$\sum_{F \in \mathcal{F}_R^0 \cap \mathcal{P}} w(C_F(S)).$$

That is, $T_w(R)$ is the total work done for $\mathcal{F}_R^0$, when work is done only for regions in $\mathcal{P}$, and $w(C_F(S))$ work is done for the $C_F(S)$ objects of S meeting $F \in \mathcal{F}_R^0 \cap \mathcal{P}$. Note that if $w(j)$ is the function $1(j) = 1$ for all j, then $T_1(R) = |\mathcal{F}_R^0 \cap \mathcal{P}|$.

Theorem 3.2 *With the terminology of §1.4 and above, suppose the relation δ is a function up to permutation. Let $m(r) \geq E[|\mathcal{F}_R^0 \cap \mathcal{P}|]$ for random R with $|R| = r$. Suppose $m(r)$ and $w(j)$ are nondecreasing, $w(j)$ is submultiplicative and $w(j) = j^{O(1)}$ as $j \to \infty$. Then*

$$E[T_w(R)] \leq O(w(n/r)m(r))$$

as $n \to \infty$, and the constant factor does not depend on r.

That is, the average work done for a member of $\mathcal{F}_R^0$ is $O(w(n/r))$. In particular, when $w(j) = j$, the theorem implies that the average number of objects of S meeting $F \in \mathcal{F}_R^0$ is $O(n/r)$.

Proof. It will be convenient to put $\mathcal{F}_S$ and $\mathcal{F}_R$ relative to $\mathcal{P}$, that is, to change the definitions to require inclusion in $\mathcal{P}$. It will also be convenient to assume that δ is defined only for S^b and not for $i < b$. If the resulting restricted theorem is then applied b times, the general theorem is proven, with a term of b in the constant factor.

As in the proof of Theorem 2.2, let $I_F = 1$ when $F \in \mathcal{F}_R^0$, and 0 otherwise. Then

$$T_w(R) = \sum_{j \geq 0, F \in \mathcal{F}_S^j} w(j) I_F,$$

so

$$\begin{aligned} E[T_w(R)] &= \sum_{j \geq 0, F \in \mathcal{F}_S^j} E[w(j) I_F] \\ &= \sum_{j \geq 0, F \in \mathcal{F}_S^j} w(j) \mathrm{Prob}\{F \in \mathcal{F}_R^0\}. \end{aligned}$$

Since δ is a function up to permutation, the probability that $F \in \mathcal{F}_R^0$ given that $F \in \mathcal{F}_S^j$ is $p_j(n, r, b) = \binom{n-b-j}{r-b} / \binom{n}{r}$. Letting $n' = (n-b)/(r-b)$, we have

$$\begin{aligned} E[T_w(R)] &= \sum_{j \geq 0} w(j) p_j(n, r, b) |\mathcal{F}_S^j| \qquad (1) \\ &= \sum_{\substack{i \geq 0 \\ in' \leq j < (i+1)n'}} w(j) p_j(n, r, b) |\mathcal{F}_S^j| \\ &\leq O(w(n/r)) \sum_{\substack{i \geq 0 \\ in' \leq j < (i+1)n'}} w(i+1) \frac{(i+1)^b}{e^{i/2}} p_j(n, r_i, b) |\mathcal{F}_S^j| \\ &\leq O(w(n/r)m(r)) \sum_{i \geq 0} w(i+1) \frac{(i+1)^b}{e^{i/2}}. \end{aligned}$$

The first inequality uses the technical lemma and the condition that $w(j)$ is nondecreasing, and the second inequality uses Equation 1, the definition of $m(r)$, and the condition that $m(r)$ is nondecreasing. The theorem follows from the observation that the sum converges. □

The following is a combination of the above theorem and Corollary 4.2 of [Cla87].

Corollary 3.3 *Under the conditions of Theorem 3.2 and for any q, there exists $z_{r,q} = O(q + b \ln r)$ as $r \to \infty$ such that with probability $3/4 - 2^{-q}$, both of these conditions hold:*

$$\sum_{F \in \mathcal{F}_R^0} C_F(S) \leq O(n/r) g(r)$$

and

$$\max_{F \in \mathcal{F}_R^0} C_F(S) \leq z_{r,q} n/r.$$

Proof. By Markov's inequality, the probability that $T_1(R)$ exceeds four times its mean is no more than 1/4. From Corollary 4.2 of [Cla87], the probability that the second condition fails is at most 2^{-q}. □

3.2 Line segment intersections

Given a set S of n line segments in the plane, the problem of determining all intersecting pairs of segments in S is reducible to the problem of constructing $\mathcal{V}(S)$, the vertical visibility map of S. (This map was defined in §1.3.) In this section, two algorithms are given for the problem of constructing $\mathcal{V}(S)$. The first algorithm will be called the "basic" algorithm, and requires $O(A + n \log n)$ expected time. The second algorithm is a refinement of the basic algorithm that has the same time bound and requires $O(n)$ space in the worst case.

The basic algorithm is an instance of the "randomized incremental construction" procedure of Clarkson and Shor[CS88]. The line segments are added in random order, one by one, to a set R. The visibility graph $\mathcal{V}(R)$ is maintained as R grows. In addition, a *conflict graph* is maintained. The nodes of the conflict graph correspond to the line segments in $S \setminus R$, and the interiors of the visibility cells of $\mathcal{V}(R)$. There is a graph edge between a segment and a cell interior when they intersect. That is, there are edges between each segment s and all the cells of $\mathcal{V}(R)$ that not in $\mathcal{V}(R \cup \{s\})$. Equivalently, for each segment s there is a list L_s of conflicting cells, and for each cell Q, there is a list L_Q of conflicting segments. When adding a segment s, the cells that must be deleted are found by examining L_s. These cells are then split up by s, each into at most four pieces. The edges in the conflict graph to the new cells that result from s can be found by examining the lists L_Q, for the cells $Q \in L_s$.

From the Randomized Incremental Construction theorem of [CS88], the time required by this procedure is proportional to $n + A + n\sum_{1\le r\le n/2} m(r)/r^2$, where $m(r)$ is the expected number of cells in $\mathcal{V}(R)$, for a random $R \subset S$ of size r. The following lemma implies the time bound.

Lemma 3.4 *Suppose S is a set of n nondegenerate line segments in the plane, with A intersections. Let R be a random subset of S of size r. The number of regions in $\mathcal{V}(R)$ is no more than $O(r + A')$, where A' is the number of intersecting pairs of segments of R. The expected value of A' is $Ar(r-1)/n(n-1)$. Therefore $m(r) = O(r) + O(Ar^2/n^2)$.*

Proof. The first statement is an obvious consequence of the fact that $\mathcal{V}(R)$ is a planar map.

Let X_S be the set of intersection points of S, and X_R the set of intersection points of R. For $x \in X_S$, let $I_x = 1$ when $x \in X_R$, and 0 otherwise. Then $A' = \sum_{x\in X_S} I_x$, so

$$\begin{aligned} E[A'] &= \sum_{x\in X_S} E[I_x] = \sum_{x\in X_S} \mathrm{Prob}\{x \in X_R\} \\ &= A\binom{r}{2}\Big/\binom{n}{2}, \end{aligned}$$

since $x \in X_R$ if and only if the two line segments that meet at x are both in R. □

The time bound for the construction and the expression for $m(r)$ give the following theorem.

Theorem 3.5 *For a set S of n line segments having A intersecting pairs, the vertical visibility map $\mathcal{V}(S)$ can be computed in $O(A + n\log n)$ expected time.*

The space bound for this algorithm is certainly $O(n\log n + A)$ on the average. To do better in the space bound, we use the above "basic" algorithm as a major step in a "global" algorithm. The global algorithm is recursive, but has a recursion depth of 3. The idea is this: the algorithm takes a random subset R of S of size $r = \sqrt{n}$, and determines $\mathcal{V}(R)$ using the basic algorithm. The algorithm next determines Q_S for all $Q \in \mathcal{V}(R)$, where Q_S is the set of segments of S that intersect Q. The algorithm also checks that these sets satisfy certain cardinality conditions. (The determination of Q_S is discussed below.) The algorithm then recursively determines the intersections in each set Q_S in turn. If on some recursive step, the size of the input is less than $r = n^{1/2}$, then the basic algorithm is used to determine the intersections of that input.

The following lemma describes the negations of the desired conditions on R and the sets Q_S, and implies that these desired conditions hold with high probability.

Corollary 3.6 *Suppose S is a set of n line segments in the plane, with A pairs of these segments intersecting. Let R be a random subset of S of size r. There exist constants $K_{\max}$ and K_{tot} such that, with probability at most $1/4$,*

$$K_{\mathrm{tot}}(n + Ar/n) \le \sum_{Q\in\mathcal{V}(R)} |Q_S|,$$

and with probability at most $1/n^{10}$,

$$K_{\max}(n/r)\log n \le \max_{Q\in\mathcal{V}(R)} |Q_S|.$$

Proof. The proof follows that of Corollary 3.3. □

If the second inequality does not hold, the sample R will be said to be "good." The global algorithm will repeatedly take random subsets $R \subset S$ of size $r = n^{1/2}$ until a good subset is found. Since the subset is good, all the sets Q_S have at most $O(n^{1/2}\log n)$ segments. At the next recursive level, all the input sets will have $O(\log^2 n)$ segments. At this level, the basic algorithm is used. Thus, no intersections are determined using the basic algorithm for sets larger than $n^{1/2}$, and a space bound of $O(n)$ holds.

It remains to describe the means by which the global algorithm determines the sets Q_S, having computed $\mathcal{V}(R)$. Two methods are used to do this, and are run concurrently until one is done. Both require $O(n)$ space in the worst case. Method I requires $O(A + n\log n)$ expected time when $A \le n^{3/2}$. Method II requires $O(A + n^{3/2})$ expected time, which is $O(A + n\log n)$ for $A > n^{3/2}$. Thus this step requires $O(A + n\log n)$ expected time, regardless of the value of A.

Method I requires that additional information be obtained while the basic algorithm is computing $\mathcal{V}(R)$. This information is the location of the endpoints of the segments of S in $\mathcal{V}(R)$. Since such information is maintained in computing $\mathcal{V}(S)$ using the basic algorithm, the $O(A + n\log n)$ time bound applies, and the space needed is $O(n)$.

With this additional information, Method I walks along each line segment of S, determining the regions meeting that segment. (It is assumed that each region has no more than 4 neighbors, which holds if no three line segment endpoints of S are on the same vertical line. If this does not hold, a random rotation of the coordinate system ensures it with probability 1.) If at any time, the number of line segments of S meeting a region of $\mathcal{V}(R)$ exceeds $K_{\max}n/r \log r$, or the total over all regions of $\mathcal{V}(R)$ of the number of line segments meeting that region exceeds $K_{\text{tot}}n$, then the method restarts with another sample. When $A \leq n^{3/2}$, so that $Ar/n \leq n$, the probability of a restart is less than $1/2$ by the above lemma, so that two attempts on the average give the desired subset and information. Each attempt requires $O(n \log n)$ expected time, independent of whether the attempt succeeds, so the desired time bound holds.

Method II simply checks every line segment in S against every region in $\mathcal{V}(R)$, restarting if $\mathcal{V}(R)$ is shown to have a region meeting more than $K_{\max}n/r \log r$ segments of S. (That is, if R is not good.) If R is good, then the sets Q_S and the intersections therein are determined for each $Q \in \mathcal{V}(R)$ in turn. In at most two samples, on the average, this strategy will succeed. Method II requires $O(nr+nA')$ time, where A' is the number of intersections of segments of R. The expected value of A' is $O(A/n)$. However, we are interested in the expected value of A', under two conditions: that R is not good, and that R is good. The probability that R is not good is so small that even if the expected value of A' is n^2, we can ignore this possibility. Since for nonnegative random variable X and event B, we know that $E[X|B] \leq E[X]/\text{Prob}\{B\}$, we have that the expected value of A', given that R is good, is within a constant factor of A/n. Thus method II requires $O(nr + A)$ expected time to obtain the sets Q_S.

Similar reasoning shows that the expected value of $\sum_{Q\in\mathcal{V}(R)} |Q_S| \log |Q_S|$ differs by only a constant factor from the expected value of that quantity, conditioned on the "success" requirements for R of either Method I or Method II. Suppose we assume inductively that the global algorithm requires $O(|Q_S| \log |Q_S| + A_Q)$ expected time for region $Q \in \mathcal{V}(R)$, where A_Q is the number of intersections in Q. This assumption, together with Theorem 3.2, readily yield the following theorem. Note the "induction" has only 2 steps, so multiplication by constant factors at each step gives only a larger constant factor.)

Theorem 3.7 *Suppose S is a set of n nondegenerate line segments in the plane, with A pairs of these segments intersecting. The vertical visibility map $\mathcal{V}(S)$ can be computed in $O(n)$ space and $O(A + n \log n)$ expected time.*

3.3 Triangulation of simple polygons

It is enough to determine vertical vertex-edge visibility relations, that is, the map $\mathcal{V}(S)$ where S is the set of edges of the simple polygon P. The algorithm is: pick random $R \subset S$ of size $r = n/\log n$, and compute $\mathcal{V}(R)$ in $O(r \log r) = O(n)$ time. Determine Q_S for every $Q \in \mathcal{V}(R)$ by walking along P, maintaining the current cell Q along the walk. Since each cell Q has $O(1)$ edges, this can be done in time proportional to $\sum_{Q\in\mathcal{V}(R)} |Q_S|$. (This assumes that no three vertices of P fall on the same vertical line, which can be assured with probability 1 by a random rotation of the coordinate system.) On the average, each set Q_S contains $O(\log n)$ edges, and it is enough to determine the visibility relations among the edges in each Q_S. This can be done in $O(\log n \log\log n)$ time on average for each Q, so the total time needed is $O(n/\log n)O(\log n \log\log n)$, or $O(n \log\log n)$.

4 A convex hull algorithm

The algorithm will be stated in terms of an equivalent problem, that of determining the intersection $\mathcal{P}(S)$ of a set S of n halfspaces. It is assumed, without loss of generality, that a point p_* in the intersection is known.

The main idea of the algorithm is to quickly filter out those halfspaces in S that contain $\mathcal{P}(S)$ in their interiors. Such halfspaces are *redundant*, and removing them gives a set of *irredundant* halfspaces S' with $\mathcal{P}(S) = \mathcal{P}(S')$. Furthermore, if the descriptive complexity of $\mathcal{P}(S)$ is A, then there are certainly no more than A halfspaces in S'. (Dually, we are quickly removing points that are inside the hull.)

The algorithm is based on an expected relation between the intersection $\mathcal{P}(R)$ of random $R \subset S$, and the intersection $\mathcal{P}(S)$. Suppose $\mathcal{P}(R)$ is split up into simple pieces as follows: take some arbitrary (fixed) plane H, and cut each face of $\mathcal{P}(R)$ into trapezoids using the translates of H that pass through the vertices of the face. Decompose $\mathcal{P}(R)$ into a set of simple regions $\Delta(R)$, each region consisting of the convex closure of p_* with a trapezoid from the cutting of the faces. For $Q \in \Delta(R)$, let Q_S denote the set of halfspaces of S that do not contain Q entirely in their interiors. The following lemma is a corollary of Theorem 3.2.

Lemma 4.1 *The expected value of $\sum_{Q\in\Delta(R)} |Q_S|$ is $O(n)$.*

Proof. (Sketch) The objects are halfspaces. The parameter b is five: one halfspace determines the face containing a trapezoid, two more halfspaces determine the two edges on that face that determine the trapezoid, and two more determine the vertices of the trapezoid. The regions are five-sided polyhedra with p_* as one vertex. □

Note that $\mathcal{P}(S)$ consists of the union of the regions $Q \cap \mathcal{P}(S)$, over all $Q \in \Delta(R)$. The halfspaces of S that contribute to a nontrivial region $Q \cap \mathcal{P}(S)$ are contained in Q_S.

Since every irredundant halfspace determines a vertex of $\mathcal{P}(S)$, we need not consider all the regions $Q \in \Delta(R)$, only those that contain at least one vertex of $\mathcal{P}(S)$. Call this set of regions $\Delta^*(R)$. Certainly $\Delta^*(R)$ contains at most A regions. The following lemma is a corollary of Theorem 3.2.

Lemma 4.2 *The expected value of $\sum_{Q \in \Delta^*(R)} |Q_S|$ is $AO(n/r)$, for sample size r.*

Proof. As in the previous lemma. The subset $\mathcal{P}$ of Theorem 3.2 is here the subset of regions that contain a vertex of $\mathcal{P}(S)$. □

Now suppose the sample size r is $\Omega(A^2)$. Then the total number of halfspaces in the sets Q_S, for $Q \in \Delta^*(R)$, is $O(n/A)$. This observation provides a fast means of filtering out redundant halfspaces, making two assumptions: we have a fast means of determining the regions $\Delta^*(R)$ and the corresponding halfspaces Q_S, and we can obtain an estimate of A. We next consider these two problems.

The regions $Q \in \Delta(R)$ and the corresponding Q_S can be readily obtained in $O(n \log r)$ expected time using the randomized incremental construction technique discussed in [CS88]. In this technique, we add halfspaces one by one, maintaining the intersection of the halfspaces that have been added, and also maintaining, for each edge of the intersection, the set of (not yet included) halfspaces that do not contain that edge in their interior. For the present application, we apply this technique by including r halfspaces, and then stop. The sets Q_S are readily obtained from the sets maintained for the intersection edges. The time bound follows readily from the analysis in [CS88].

To determine $\Delta^*(R)$ from $\Delta(R)$, we must have a fast means of determining the regions Q that contain no vertices of $\mathcal{P}(S)$, or conversely, the regions that contain only parts of faces or edges. This is done as follows: let t be a face of a region $Q \in \Delta(R)$, with p_* incident to t. Then the polygon $P_t = t \cap \mathcal{P}(S)$ can be determined using the algorithm of Kirkpatrick and Seidel [KS86] in time on the order of $|Q_S| \log A_t$, where A_t is the number of sides of P_t. All but two of the sides of P_t correspond to faces of $\mathcal{P}(S)$, so that the total time to compute all such polygons is expected $O(n \log A')$, where A' is the total number of faces of $\mathcal{P}(S)$ identified. If a region $Q \in \Delta(R)$ contains no vertices of $\mathcal{P}(S)$, the polygons corresponding to faces of Q completely determine the structure of $Q \cap \mathcal{P}(S)$, and this can be verified or disproven in $O(|Q_S|)$ time. Thus the regions in $\Delta^*(R)$ and their corresponding halfspaces can be found in $O(n)(\log r + \log A')$ time.

Now to consider the problem of estimating A, or rather, of using only lower bounds for A. To do this, we determine $\Delta^*(R)$ for a sequence of sample sizes, using at each step an estimate A^* of A. At each step, only some of the halfspaces may remain, since some may be eliminated by not being in some Q_S for $Q \in \Delta^*(R)$. Suppose that at a given step, the total number of halfspaces in the sets Q_S, for $Q \in \Delta^*(R)$, is greater than half the number of halfspaces from the previous step. Then the estimate A^* is taken as the square of the maximum of the A^* estimate and the A' value (as above), of the previous step. (If this new estimate of A^* is greater than the number of halfspaces remaining, compute the intersection of the halfspaces by the algorithm of [CS88].) On the other hand, if at least half the halfspaces are eliminated, the value of A^* will remain the same.

It is easy to verify that the total amount of work performed up to and during the period that A^* is a given value is $O(n \log A^*)$. Since $\log A^* = O(\log A)$, the total work performed by the algorithm is $O(n \log A)$.

5 Concluding Remarks

As one more application of random sampling, the ideas of this paper can readily be used to obtain an algorithm for point location in planar subdivisions that requires $O(n \log n)$ expected preprocessing, $O(n)$ space, and $O(\log n)$ query time.

Acknowledgements. It is a pleasure to thank Peter Shor for helpful discussions. This work and [CS88] began separately and converged, and it wasn't easy to allot results to papers. Thanks also to John Hershberger for pointing out that the hypergeometric is not binomial, and for the finding an error in an ancestor of Theorem 3.2. Thanks to Micha Sharir for helpful comments, and for observing the need for nondegeneracy in Theorem 3.2. Thanks to John Hobby and Ellen Feinberg (nè Silverberg) for helpful comments.

References

[AG86] N. Alon and E. Győri. The number of small semispaces of a finite set of points in the plane. *J. Combin. Theory Ser. A*, 41:154–157, 1986.

[AHW87] N. Alon, D. Haussler, and E. Welzl. Partitioning and geometric embedding of range spaces of finite Vapnik-Chervonenkis dimension. In *Proc. Third Symp. on Comp. Geometry*, pages 331–340, 1987.

[CEG+88] K. L. Clarkson, H. Edelsbrunner, L. Guibas, M. Sharir, and E. Welzl. The complexity of many faces in an arrangement of lines in the plane. To appear, 1988.

[Cla85] K. L. Clarkson. A probabilistic algorithm for the post office problem. In *Proc. 17th Annual SIGACT Symp.*, pages 75–184, 1985.

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

[CP86] B. Chazelle and F. P. Preparata. Halfspace range search: an algorithmic application of k-sets. In *Discrete & Comp. Geom.*, pages 83–94, 1986.

[CS88] K. L. Clarkson and P. W. Shor. Algorithms for diametral pairs and convex hulls that are optimal, randomized, and incremental. *This proceedings*, 1988.

[CSY84] R. Cole, M. Sharir, and C. Yap. On k-hulls and related problems. In *Proc. 16th Annual SIGACT Symp.*, pages 154–166, 1984.

[CTVW88] K. L. Clarkson, R. E. Tarjan, and C. J. Van Wyk. A fast Las Vegas algorithm for triangulating a simple polygon. *This proceedings*, 1988.

[Ede] H. Edelsbrunner. Personal communication.

[Ede87] H. Edelsbrunner. *Algorithms in Combinatorial Geometry.* Springer-Verlag, New York, 1987.

[EM88] H. Edelsbrunner and E. P. Muecke. Simulation of simplicity: a technique to cope with degenerate cases in geometric algorithms. *This proceedings*, 1988.

[ES74] P. Erdős and J. Spencer. *Probabilistic Methods in Combinatorics.* Academic Press, New York, 1974.

[GP84] J. E. Goodman and R. E. Pollack. On the number of k-subsets of a set of n points in the plane. *J. Combin. Theory Ser. A*, 36:101–104, 1984.

[HW87] D. Haussler and E. Welzl. Epsilon-nets and simplex range queries. *Discrete & Comp. Geom.*, 2:127–151, 1987.

[KS86] D. G. Kirkpatrick and R. Seideal. The ultimate planar convex hull algorithm? *SIAM Journal on Computing*, 15:287–299, 1986.

[TVW88] R. E. Tarjan and C. J. Van Wyk. An $O(n \log \log n)$ algorithm for triangulating a simple polygon. *SIAM Journal on Computing*, 17:143–178, 1988.

[Wel86] E. Welzl. More on k-sets of finite sets in the plane. *Discrete & Comp. Geom.*, 1:95–100, 1986.

[Yap88] C. K. Yap. A geometric consistency theorem for a symbolic perturbation scheme. *This proceedings*, 1988.