Discrete Comput Geom 6:385–406 (1991)

Discrete & Computational
Geometry
© 1991 Springer-Verlag New York Inc

Cutting Hyperplane Arrangements*

Jiří Matoušek

Department of Applied Mathematics, Charles University,
Malostranské nám. 25, 118 00 Praha 1, Czechoslovakia

Abstract. We consider a collection H of n hyperplanes in E^d (where the dimension d is fixed). An *ε-cutting* for H is a collection of (possibly unbounded) d-dimensional simplices with disjoint interiors, which cover all E^d and such that the interior of any simplex is intersected by at most εn hyperplanes of H. We give a deterministic algorithm for finding a $(1/r)$-cutting with $O(r^d)$ simplices (which is asymptotically optimal). For $r \leq n^{1-\delta}$, where $\delta > 0$ is arbitrary but fixed, the running time of this algorithm is $O(n(\log n)^{O(1)}r^{d-1})$. In the plane we achieve a time bound $O(nr)$ for $r \leq n^{1-\delta}$, which is optimal if we also want to compute the collection of lines intersecting each simplex of the cutting. This improves a result of Agarwal, and gives a conceptually simpler algorithm.

For an n point set $X \subseteq E^d$ and a parameter r, we can deterministically compute a $(1/r)$-net of size $O(r \log r)$ for the range space

$$(X, \{X \cap R;\ R \text{ is a simplex}\}),$$

in time $O(n(\log n)^{O(1)}r^{d-1} + r^{O(1)})$. The size of the $(1/r)$-net matches the best known existence result. By a simple transformation, this allows us to find ε-nets for other range spaces usually encountered in computational geometry.

These results have numerous applications for derandomizing algorithms in computational geometry without affecting their running time significantly.

1. Introduction and Statement of Results

Algorithmic and proof techniques based on random sampling have gained a central position in computational geometry during last few years. These techniques, pioneered by Clarkson (e.g., [C1]) and Haussler and Welzl [HW] yield nearly

* A preliminary version of this paper appeared in *Proceedings of the Sixth ACM Symposium on Computational Geometry*, Berkeley, 1990, pp. 1–9. Work on this paper was supported by DIMACS Center.

optimal randomized algorithms for an enormous range of geometric problems. In a significant portion of these results, the following statement is used:

Let H be a collection of n hyperplanes in E^d, and let R be a random sample of r hyperplanes of H. When we triangulate the regions of the arrangement of R (yielding $O(r^d)$ simplices), then with high probability each simplex in this triangulation is intersected only by $O((n/r)\log r)$ hyperplanes of H.

This statement is proved by an elementary probabilistic consideration. Similar statements can also be proved for other types of hypersurfaces; the triangulation can be replaced by another suitable subdivision of the arrangement into constant complexity cells. In geometric algorithms, such results usually serve as an efficient geometric divide-and-conquer strategy, allowing us to divide a problem involving the hypersurfaces into smaller subproblems defined by the cells.

For some reasons, people usually consider deterministic algorithms more satisfactory than randomized ones. In this paper we give an efficient deterministic analogy of the above probabilistic technique, which allows us to remove randomization from many algorithms. We develop it for the case of hyperplanes but, as we indicate, it can also be applied for other hypersurfaces.

Before we state the results we give some definitions. A *cutting* is a collection of (possibly unbounded[1]) d-dimensional simplices with disjoint interiors, which cover all E^d. The *size* of a cutting is the number of its simplices. The total number of faces of all dimensions is proportional to the size. Let H be a collection of n hyperplanes and let Ξ be a cutting. For a simplex $s \in \Xi$, let H_s denote the collection of hyperplanes intersecting the interior of s. A cutting Ξ is an *ε-cutting* for H provided that $|H_s| \leq \varepsilon n$ for every $s \in \Xi$. The number $(1/\varepsilon)$ is called the *cutting factor* of an ε-cutting.

Several previous papers use other names for cutting, such as, e.g., *partitioning* [A1] or *simplicial packing* [CF]. The way the parameter ε defines how "fine" a cutting is, perhaps is not the most natural one, but it has been chosen in analogy with ε-nets.

Note that the definition does not require that a cutting is a simplicial complex (after including faces of all dimensions), e.g., a vertex of one simplex may lie in the interior of a face of another simplex. Later we put additional restrictions on the cuttings used in our algorithms.

A construction of an ε-cutting for a collection of hyperplanes is usually applied as a "divide" step in a divide-and-conquer strategy. Thus the size of the cutting is very important for the efficiency of such algorithms, and we would like to have it as small as possible.

The best size of a $(1/r)$-cutting we can hope for is of order r^d. We can see this as follows: a collection of n hyperplanes in general position determines $\Omega(n^d)$ intersections (vertices of the arrangement), and a single simplex of a $(1/r)$-cutting may contain only $O((n/r)^d)$ of them.

[1] To be rigorous, we should work in the projective space; bearing this in mind, we freely use the Euclidean space with more intuitive notions.

Chazelle and Friedman [CF] proved that for every H there exists a $(1/r)$-cutting of asymptotically optimal size $O(r^d)$. They also gave a deterministic algorithm which computes such a cutting, with time complexity $O(n^{d(d+3)/2+1}r)$. Their proof shows that if we permit randomization, we can find such a cutting in expected time $O(nr^{d-1})$. We explain their proof (with some technical simplifications) in the Appendix, and we apply their ideas at several other places in this paper.

For the two-dimensional case, Matoušek [M2] independently gave an existence proof for $(1/r)$-cutting of an asymptotically optimal size, and also an $O(nr^2 \log r)$ deterministic algorithm computing it. The time complexity has been improved by Agarwal [A1] to $O(nr \log n \log^{\omega} r)$ ($\omega < 3.3$ is a constant), and the companion paper [A2] gives an extensive survey of applications.

The time bound $O(nr^{d-1})$ is optimal in the following sense: if we also want to compute the collection of hyperplanes intersecting each simplex of the cutting (as is the case in many applications), then the output size generally is already of order $\Omega(nr^{d-1})$. However, there are applications where this additional information is not required (such as, e.g., the construction of a spanning tree with a low crossing number), and then the above argument for optimality cannot be used.

The main result of this paper is the following:

Theorem 1.1. *Given a collection H of n hyperplanes in E^d and a number $r \le n^{1-\delta}$ (for an arbitrary but fixed $\delta > 0$), we can deterministically compute a $(1/r)$-cutting of size $O(r^d)$ for H, in time $O(n(\log n)^A r^{d-1})$ (A is a constant depending on dimension, and the constant of proportionality in the bound on the size of the cutting increases as δ approaches to 0). For dimension 2, the running time can be made $O(nr)$, which is optimal if we also want to compute the collection of lines intersecting each simplex of the cutting.*

The proof of this theorem applies the techniques of [CF], [M1], and [M2], and adds some new ingredients. For the sake of clarity we do not try to achieve the best value of A. The two-dimensional result is similar to the result of [A1], but the algorithm is conceptually simpler and more efficient. Our techniques also give some results for a general value of r without the limitation $r \le n^{1-\delta}$; see the discussion in Section 3.2.

Now let us recall the notions of a range space and an ε-net, introduced in [HW]. A *range space* is a pair (X, R), where X is a set (the *points*) and R is a set of subsets of X (the *ranges*). Let ε be a real number, $O \le \varepsilon < 1$. A subset $N \subseteq X$ is called an ε-net for (X, R) if N intersects every range $r \in R$ with $|r| > \varepsilon|X|$ (this definition makes sense for a finite X only).

Range spaces defined by simple geometric objects in Euclidean space (e.g., half-spaces, wedges, simplices) have a speical significance for computational geometry. By an *H^k-range* we mean an intersection of k half-spaces (in E^d). If X is a subset of E^d, we denote by $H^k(X)$ the following range space: the point set will be just X, and the ranges will be all subsets of X, which can be obtained as the intersection of X with an H^k-range. The most interesting cases are $k = 1$ (ranges defined by half-spaces) and $k = d + 1$ (ranges defined by simplices). In these special range spaces we have the following constructive analogue of the general existence result of [HW]:

Theorem 1.2. *Let $k \leq d + 1$ be a fixed integer. Given a set X of n points in E^d and a number r, we can deterministically find a $(1/r)$-net of size $O(r \log r)$ for the range space $H^k(X)$, in time $O(n(\log n)^A r^{d-1} + r^B)$($A$, B are constants dependent on k and d).*

Let us remark that our bound on the size of the ε-net for a general dimension d matches the best known upper bound (gained by probabilistic methods). At the same time it is known that this upper bound cannot be improved for general range spaces (see [PW]), but for range spaces $H^k(X)$ this is an open problem. For a partial result in this direction see [SWM].

A simple transformation allows us to extend the above result to a deterministic computation of ε-nets in all range spaces usually encountered in computational geometry; this observation is due to Yao and Yao [YY]. We explain this in Section 5. This in turn can be used for cutting collections of other hypersurfaces.

The above results allow us to remove randomization from many algorithms without affecting their asymptotic time complexity significantly. From two-dimensional applications (where our result only improves some logarithmic factors compared with the result of Agarwal) let us mention, e.g., [EGH*] and the examples in [A2]. Higher-dimensional applications can be found, e.g., in [CEG] and [AESW], and we believe that many other applications will be found.

The paper has two main parts. In one part (Section 2) we investigate "approximating structures" for a collection of hyperplanes. In another part (Section 3) we build the cutting algorithm in a recursive fashion, using tools developed in the previous section as primitives. Section 4 describes how to compute ε-nets (using cuttings).

In the conference version of this paper, a somewhat worse size of the resulting cutting was attained—$O(r^d(\log r)^{O(1)})$. The present version differs mainly by incorporating the technique of Chazelle and Friedman [CF], thus attaining asymptotically an optimal size of the cuttings. Moreover, one of the intermediate products of the algorithm is singled out as a notion deserving attention (the so-called ε-approximation), which might shed more light on the interplay of various "approximating structures" for a collection of hyperplanes.

The results of this paper were further improved in two recent papers: [M4] and [M5]. The first one improves the running time in Theorem 1.1 to $O(nr^{d-1})$ for every dimension, and the second one shows that a $(1/r)$-cutting can be computed in time $O(n \log r)$ if r is not too large ($r \leq n^{1/(2d-1)}$).

2. Approximating Structures for Collections of Hyperplanes

2.1. Preliminaries on Arrangements, Canonical Triangulation

We consider an arrangement H of n hyperplanes in E^d. For terminology about arrangements see [E]. Throughout this paper we assume that the hyperplanes are in a general position, which simplifies many proofs and allows us to concentrate on the essence. Similarly, when considering a collection of hyperplanes and an ε-cutting for it, we assume that the vertices of the cutting are not incident to the hyperplanes. There is no loss of generality in the algorithms, since we may apply the technique known as *simulation of simplicity* (see [EM] and [E]). This technique formally introduces infinitesimal perturbations of the hyperplanes, which always put them into a general position. It costs a constant multiplicative factor in the running time.

The arrangement (as a cell complex) can be triangulated in various ways; we use the so-called *canonical triangulation*. This triangulation is constructed as follows: For every face of the arrangement (of any dimension), one vertex is defined as the *apex* of that face; it is the vertex with the lexicographically smallest coordinate vector. Then the triangulation is defined by induction on the dimension. The triangulation of one-dimensional arrangements is unique. For an arrangement of dimension $d \geq 2$, we first triangulate the n arrangements in $d - 1$ space formed by intersecting every hyperplane with the $n - 1$ others. Then for a d-dimensional face f, we triangulate it by the simplices arising as cones from the apex to the simplices forming the triangulation of the lower-dimensional faces of f not incident to the apex of f.

A more detailed discussion of the canonical triangulation appears in [C2] or [CF]. For us it is important that this triangulation determines a simplicial complex with $O(n^d)$ simplices. Given the hyperplanes, both the arrangement and its canonical triangulation can be constructed in time $O(n^d)$ [EOS].

We call any cutting arising as a canonical triangulation of some arrangement a *standard* cutting. For some purposes, the manipulations with standard cuttings are slightly more efficient than for general ones.

2.2. Computing Subproblems Defined by Cutting

Lemma 2.1. *Given a collection H of n hyperplanes and a standard $(1/r)$-cutting Ξ for H, we can compute the collection H_s for every simplex $s \in \Xi$ in total time proportional to $\sum_{s \in \Xi} |H_s|$, i.e., in time $O(nk/r)$ for a $(1/r)$-cutting of size k.*

Proof. We determine the collection of simplices intersected by a hyperplane h, for every $h \in H$. First suppose that we already know one simplex intersected by a hyperplane h. The remaining simplices can be determined by "walking along" the hyperplane, which amounts to a searching in a graph of bounded degree (since we assume that Ξ is standard, thus it determines a simplicial complex). The time needed for this is proportional to the number of intersected edges. This is the only point where we need the "standardness" requirement, and obviously we might relax it in various ways.

A starting simplex intersected by a hyperplane can be determined as follows: At the beginning, we fix a vertical line λ, we compute all simplices of Ξ intersecting it, and we sort them in the order of their occurrence along λ. Then a starting simplex for a hyperplane h can be determined by a binary search (we search for the simplex containing the point $h \cap \lambda$). The time for this search is dominated by the number of simplex/hyperplane incidences. □

2.3. Refining a Cutting

In our algorithm we first show how to compute a $(1/r)$-cutting for small (e.g., constant) values of r. The following observation allows us also to deal with larger r: Suppose that we are given a $(1/r_1)$-cutting Ξ for H. For every $s \in \Xi$, let Ξ_s be a $(1/r_2)$-cutting for the collection H_s of hyperplanes intersecting the interior of the simplex s. Then we can obtain a $(1/r_1 r_2)$-cutting for H as follows: For every simplex s of Ξ, consider all nonempty cells of the form $s \cap s'$, where s' is a simplex of Ξ_s. Each such cell is defined as the intersection of two simplices, and thus can be triangulated using a constant number of simplices. It is easily seen that each of the resulting simplices is intersected by at most $n/r_1 r_2$ hyperplanes, and thus these simplices form a $(1/r_1 r_2)$-cutting for H.

Let us look what happens to the size of the cutting in the above refinement step. Assume for a moment that we have an algorithm computing a $(1/r_0)$-cutting of size at most Kr_0^d for any collection H (where r_0, K are certain constants). If we want to compute a $(1/r)$-cutting for H, where $r = r_0^m$, we could use the refining step m times. In the first step we get a $(1/r_0)$-cutting of size at most Kr_0^d, in the second step we obtain a $(1/r_0^2)$-cutting of size $CK^2(r_0^2)^d$ (C is some additional constant factor arising from the triangulation of constant complexity cells in the refining step), and in mth step we get a $(1/r)$-cutting of size $C^{m-1}K^m r^d$. Now if r is really big ($\sqrt{n}$, say), m will be of order $\log n$ and thus the size of our cutting will be very far from optimal. To remedy this, our algorithm contains a *simplification step*, which again improves the size of the cutting.

The cuttings we encounter in our cutting algorithm usually have the following structure: they arise by refining a standard cutting using a standard cutting for each of the collections H_s. Sometimes this refinement is repeated more than once, but a constant number of times. It is not difficult to see that the claim of Lemma 2.1 remains valid for this kind of a cutting: we first trace the hyperplanes in the first-level simplices and then in the refining simplices—for every first-level simplex separately; similarly for more levels of refinement. We call cuttings satisfying the claim of Lemma 2.1 *walkable*.

2.4. ε-Approximations

The notion of ε-approximation, which we introduce in this section, appears already in [VC] and [HW], but so far it has not been applied in computational geometry. Here it is an essential tool in our algorithm.

We say that a collection A of hyperplanes is an *ε-approximation* for H provided that, for every segment e, it is

$$\left| \frac{|A_e|}{|A|} - \frac{|H_e|}{|H|} \right| < \varepsilon,$$

where A_e (resp. H_e) denotes the set of all hyperplanes of A (resp. of H) intersecting the segment e.

Haussler and Welzl [HW] use the notion of ε-approximation for a general range space; it is such a subset A of the point set that the relative fraction of elements of A in every range approximates the relative size of that range with accuracy ε. From this point of view, we should call our notion a weak ε-approximation, since we do not require that $A \subseteq H$. However, we use the shorter term, since we do not deal with any other ε-approximations.

It is sometimes convenient to work with weighted collections of hyperplanes. A weighted collection of hyperplanes is a pair H, w, where H is a collection of hyperplanes, and w: $H \to R^+$ is a weight function on H. If $X \subseteq H$, we write just $w(X)$ for $\sum_{h \in X} w(h)$. The notions introduced for unweighted collections of hyperplanes can usually be generalized for weighted collections in an obvious way, e.g., a cutting Ξ is an *ε-cutting for H, w* provided that, for every simplex s of Ξ, the collection H_s of hyperplanes of H intersecting the interior of s has total weight at most $\varepsilon w(H)$. Similarly, a weighted collection A, w is an ε-approximation for H provided that $|w(A_e)/w(A) - |H_e|/|H|| \leq \varepsilon$ for every segment e.

The collection H itself is, of course, an ε-approximation for H for every ε. But the point is that the cardinality of ε-approximations can be much smaller that the cardinality of H, even depending on ε only. A theorem due to Vapnik and Chervonenkis [VC] implies the following: for every fixed dimension d, every H, and every r, there exists an (unweighted) $(1/r)$-approximation of size $O(r^2 \log r)$ for H.

In our algorithm we use a special type of $(1/r)$-approximation, which is larger but still of polynomial size in r. First we introduce an equivalence relation on the hyperplanes: we call two hyperplanes *equivalent* with respect to a point set P if they separate the points of P in the same manner (or, in other words, there is no point of P in one of the double wedges defined by these hyperplanes). Let Ξ be a $(1/r)$-cutting for H. We define a weighted collection of hyperplanes A, w as follows: We pick in A one hyperplane h from every equivalence class of the hyperplanes with respect to the set of vertices of Ξ, and we define its weight $w(h)$ to be the number of hyperplanes in that equivalence class. Let us call this collection a *description of H relative to Ξ*. If Ξ has k vertices, then there are no more than k^d equivalence classes there and thus $|A| \leq k^d$.

Lemma 2.2. *A description of H relative to a $(1/r)$-cutting Ξ is a $(2/r)$-approximation for H.*

Proof. Consider an arbitrary segment e and let s and s' be the simplices of Ξ containing the endpoints of e. Call an equivalence class of hyperplanes *homogeneous for e* if all its members intersect e or none does. The number of hyperplanes of H from homogeneous classes intersecting e is exactly reflected in the description; the only differences might arise for classes which are not homogeneous. But all hyperplanes from such classes must intersect s or s', and thus their number is at most $2|H|/r$. □

Lemma 2.3. *Given a walkable $(1/r)$-cutting of size k for H, we can compute the description of H relative to Ξ in time proportional to the number of simplex/hyperplane incidences, i.e., in time $O(nk/r)$.*

Proof. We can compute the collection of simplices intersected by every hyperplane of H in the above time bound, and thus we can also obtain the collection of edges of Ξ intersected by that hyperplane. Now the collection of intersected edges uniquely determines the equivalence class of a hyperplane. Having the set of intersected edges for every hyperplane (as a list of integers not exceeding k), it suffices to determine the classes of equal lists. To this end, we may sort each list (in linear time) and then sort the lists lexicographically, which can also be done in time proportional to the total size of the lists (see [AHU]). □

A very straightforward operation for ε-approximations is the merging of several collections of hyperplanes:

Observation 2.4. *Let $H_1, \ldots, H_m$ be collections of hyperplanes and let A_i, w_i be an ε-approximation for H_i with $w_i(A_i) = |H_i|$. If we set $A = A_1 \cup \cdots \cup A_m$ and $w = w_1 \cup \cdots \cup w_m$, then A, w is an ε-approximation for $H = H_1 \cup \cdots \cup H_m$.* □

2.5. *From Approximations to Cuttings and Back: The Simplification Step*

We begin with a lemma observed by many researchers:

Lemma 2.5. *Let Ξ be a cutting such that no segment contained in the interior of a simplex of Ξ intersects more than εn hyperplanes of H. Then Ξ is a $(d\varepsilon)$-cutting for H.*

Proof. Consider a simplex $s \in \Xi$ and choose a set E of d of its edges, forming a connected graph on the vertices of s. For every $e \in E$, consider a segment e' in the interior of s, intersecting the same subset of hyperplanes (meeting the interior of s) as e does. No e' is intersected by more than εn hyperplanes, and thus the interior of s is intersected by no more than $d\varepsilon n$ hyperplanes. □

This implies the following observation, which serves as a base for a quick computation of cuttings:

Lemma 2.6. *Let A, w be an ε-approximation for H and let Ξ be a δ-cutting for A, w. Then Ξ is a $d(\varepsilon + \delta)$-cutting for H.* □

Now if we have a $(1/r)$-approximation for H of small size (compared with the size of H), we can afford to use a relatively inefficient algorithm to compute a $(1/r)$-cutting for this approximation, which gives us a $(2d/r)$-cutting for H itšelf. A polynomial-time algorithm for finding a cutting of asymptotically optimal size for a collection of hyperplanes is due to Chazelle and Friedman; we only need to extend it to a weighted case:

Theorem 2.7. *Let H, w be a weighted collection of n hyperplanes and let $r \leq n$ be a number. We can deterministically compute a walkable $(1/r)$-cutting for H, w of size $O(r^d)$, in time bounded by a polynomial in n and r.*

For the unweighted case, this is just what Chazelle and Friedman [CF] proved. As shown in [M3], the weighted case is no more difficult than the unweighted one:

Theorem 2.8 [M3]. *Let H, w be a weighted collection of n hyperplanes and let $r \leq n$ be a number. Suppose that we can find a $(1/2r)$-cutting of size at most S for any collection of $\leq 2n$ hyperplanes, in time at most T. Then we can find a $(1/r)$-cutting for H, w of size at most S, in time $O(T + n)$.*

Proof. We reproduce the (simple) proof here: First we may norm the weight function in such a way that $w(H) = n$ (in linear time). Then we construct a multiset H' of hyperplanes: it will contain just the hyperplanes of H, and a hyperplane $h \in H$ will be contained in H with multiplicity $\lceil w(h) \rceil$.

The cardinality of H' (counted with the multiplicities) is at most $2n$: we have $|H'| = \sum_{h \in H} \lceil w(h) \rceil \leq \sum_{h \in H} (w(h) + 1) \leq w(H) + |H| = 2n$.

Now we use the algorithm for unweighted cutting on the multiset H' with parameter $2r$. It may be that the algorithm does not admit multiple occurrences of hyperplanes, but in any case we may deceive it by simulation of simplicity (see [EM]), i.e., by introducing infinitesimal perturbations and thus yielding (formally) objects in general position. Simulation of simplicity increases the running time at most by a constant factor. The resulting cutting may contain some simplices with empty interior, but such simplices can be simply left out.

Finally it is easy to check that a $(1/2r)$-cutting for H' is a $(1/r)$-cutting for H, w (and also a $(1/r)$-cutting for H). □

This finishes the proof of Theorem 2.7. For the sake of completeness, the method of [CF] is explained in the Appendix (in a less general setting and with slightly simplified calculations).

Summarizing, we have the following proposition:

Proposition 2.9. *Given a $(1/r)$-approximation A, w for H, consisting of k hyperplanes, we can compute a walkable (K/r)-cutting for H of size $O(r^d)$, in time polynomial in k (K is a constant).*

This shows how to compute a "simplified" cutting from an ε-approximation. In our algorithm we need to complement this by the computation of a "simplified" ε-approximation.

Lemma 2.10. *Given a $(1/r)$-approximation $\tilde{A}$, $\tilde{w}$ of size k for H and a $(1/r)$-cutting Ξ of size $O(r^d)$ for $\tilde{A}$, $\tilde{w}$, we can compute a $(3/r)$-approximation A, w of size $O(r^{d^2})$ for H in time polynomial in k.*

Proof. We let A, w be the description of $\tilde{A}$, $\tilde{w}$ relative to Ξ. By Lemma 2.2, A, w is a (2/r)-approximation for $\tilde{A}$, $\tilde{w}$ and thus also a $(3/r)$-approximation for H. The cardinality of A is of order $O(r^{d^2})$. □

3. The Cutting Algorithm

3.1. Recursion in n

In this section we give our first algorithm for finding a $(1/r)$-cutting. It will also simultaneously find a $(1/r)$-approximation.

Algorithm CUT1
Input: H, a collection of n hyperplanes, and $r \leq n$, a parameter.
Output: Ξ, a walkable $(1/r)$-cutting for H of size $O(r^d)$, and A, w, a $(3/r)$-approximation for H of size $O(r^{d^2})$.
Method:

1. (Base case) If r is sufficiently large (greater than n^α, where α is a suitable positive constant), we output H itself as the $(3/r)$-approximation A, w and we use the algorithm of Chazelle and Friedman (see Theorem 2.8) to compute the $(1/r)$ cutting Ξ, in time polynomial in r. If the above condition does not hold, we continue with the next step.
2. (Divide step) We choose a number m (which is specified later) and divide the hyperplanes of H into m groups $H_1, \ldots, H_m$ of approximately equal sizes. For every H_i we compute a $(1/Kr)$-approximation A_i, w_i by a recursive application of Algorithm CUT1 (K is the constant appearing in Proposition 2.9).
3. (Merge step) We set $\tilde{A} = A_1 \cup \cdots \cup A_m$ and $\tilde{w} = w_1 \cup \cdots \cup w_m$; by Observation 2.4, this is a $(1/Kr)$-approximation for H, and its size is $O(m \cdot r^{d^2})$.
4. (Simplification step) We apply the method of Proposition 2.9 to compute a $(1/r)$-cutting Ξ of size $O(r^d)$ for H using $\tilde{A}$, $\tilde{w}$. Then we use this cutting Ξ and $\tilde{A}$, $\tilde{w}$ to compute a $(3/r)$-approximation A, w of size $O(r^{d^2})$ for H according to Lemma 2.10. We then output Ξ and A, w.

This completes the description of Algorithm CUT1. Let $T_1(n, r)$ denote the worst-case running time of Algorithm CUT1 applied for n hyperplanes and a parameter r. We want to show that

$$T_1(n, r) = O(n(\log n)^A r^D) \tag{1}$$

for some constants A, D.

If the base case occurs, then the execution time is polynomial in n and thus also in r, so (1) is satisfied. Otherwise the recursive calls in the divide step take time at most $m \cdot T(n/m, Kr)$, and the simplification step takes time $O((mr^{d^2})^{c_1})$ for some constant c_1. We get a recurrence

$$T_1(n, r) \leq m \cdot T_1(n/m, Kr) + O(m^{c_1} r^{d^2 c_1}).$$

Choosing $m = n^{1/c_1}$, it is not difficult to verify that a function $T_1(n, r)$ satisfying this recurrence is bounded as in (1) for $D = d^2 c_1$ and for any A with $(1 - 1/c_1)^A > K^D$.

3.2. *Recursion in r*

In this section we improve the complexity of the algorithm from the previous section, namely its dependence on r. The tool for this will be the refinement of a cutting, introduced in Section 2.3. The starting observation is that if r is bounded by a constant, then Algorithm CUT1 is already good enough. We arrange the recursion in such a way that Algorithm CUT1 will always work in this favorable situation.

First let us assume that r is not too big; precisely that $r < n^{\alpha}$, where α is a suitable positive constant. Then we use the following algorithm:

Algorithm CUT2
Input: H, a collection of n hyperplanes, and $r \le n^{\alpha}$, a parameter.
Output: Ξ, a walkable $(1/r)$-cutting for H of size $O(r^d)$.
Method:

1. (Base case) If $r < r_0$ (where r_0 is a suitable constant), we use Algorithm CUT1 directly. If $r \ge r_0$, we continue with the next step.
2. (Recursion) We use Algorithm CUT2 recursively to compute a walkable $(2/r)$-cutting Ξ_1 of size $k_1 = O(r^d)$ for H.
3. (Computing subproblems) For every simplex s of Ξ_1 we compute the collection H_s of hyperplanes intersecting its interior.
4. (Refinement) For every s, we use Algorithm CUT1 to compute a $(1/4K)$-cutting Ξ_s of size at most $k_2 = O(1)$ for H_s (K is the constant from Proposition 2.9). Then we use the method of Section 2.3 to compute a $(1/2Kr)$-cutting $\tilde{\tilde{\Xi}}$ of size $O(k_1 k_2) = O(r^d)$ for H. This cutting is again walkable.
5. (Simplication step) We compute the description A, w of H relative to $\tilde{\tilde{\Xi}}$, which is a $(1/Kr)$-approximation for H. Then we apply the method of Proposition 2.9 to compute a $(1/r)$-cutting Ξ of size $O(r^d)$ for H from A, w, and we output this Ξ. Note that the size of Ξ compared with the size of $\tilde{\tilde{\Xi}}$ decreased by a constant factor only, but the constant of proportionality for the size now does not increase during the recursion (see the remark in Section 2.3).

Let us denote the worst-case running time of Algorithm CUT2 by $T_2(n, r)$. From the base case we get

$$T_2(n, r) = O(n(\log n)^A) \qquad \text{for} \quad r < r_0.$$

The recursion step requires time $T_2(n, r/2)$, the computing subproblems time is $O(nr^{d-1})$, the refinement time is $O(r^d) \cdot T_1(n/r, 2K) = O(nr^{d-1}(\log n)^A)$ and, finally, the simplification step needs time $O(nr^{d-1})$ plus time which only depends polynomially on r. We get the following recurrence relation:

$$T_2(n, r) \le T_2(n, r/2) + O(n(\log n)^A r^{d-1}) + O(nr^{d-1} + r^B)$$

(for a certain constant B). The solution of this recurrence is

$$T_2(n, r) \leq O(n(\log n)^A r^{d-1} + r^B).$$

Since we have assumed that $r \leq n^\alpha$, the first term dominates the term r^B.

For a bigger value of r (but smaller than $n^{1-\delta}$ for a fixed $\delta > 0$), the idea is to use refinement of the cutting a constant number of times, this time without a simplication step (which is prohibitively expensive for bigger values of r). The complete Algorithm CUT3 computing a $(1/r)$-cutting for a collection H of n hyperplanes can be formulated as follows:

> If $r \leq n^\alpha$ (where α is as in Algorithm CUT2), use Algorithm CUT2 to compute the answer. Otherwise set $r_1 = n^\alpha$ and $r_2 = r/r_1$, compute a $(1/r_1)$-cutting Ξ_1 for H using Algorithm CUT2 and for every collection H_s ($s \in \Xi_1$) compute a $(1/r_2)$-cutting Ξ_s by a recursive application of Algorithm CUT3. Use these cuttings to refine Ξ_1 as in Section 2.3.

Suppose that Algorithm CUT3 is called for some n and r; let us express this by writing CUT3(n, r). Then in the first level of recursion we have a call of the form CUT3($n^{1-\alpha}$, r/n^α), in the second level of recursion we have CUT3($n^{(1-\alpha)^2}$, $r/n^{\alpha+(1-\alpha)\alpha}$), and in the kth recursion level we get CUT3($n^{(1-\alpha)^k}$, r/n^{c_k}), where $c_k = \sum_{j=1}^{k}(1-\alpha)^{j-1}\alpha$. Since c_k tends to 1 with growing k, in a constant number of recursion levels c_k exceeds $1-\delta$ (for a fixed $\delta > 0$), and the recursion in Algorithm CUT3 terminates. The bound for the size of the resulting cutting is now obvious, and the analysis of the running time is also straightforward and we omit it. This finishes the proof of Theorem 1.1 for a general dimension.

Let us remark that the difference $1 - c_k$ decreases exponentially with k, and thus that for any $r \leq n$ the recursion Algorithm CUT3 terminates within $O(\log \log n)$ levels. This in turn implies that we may compute a $(1/r)$-cutting of size $O(r^d(\log r)^C)$ (C a constant) by Algorithm CUT3, with running time $O(n(\log n)^A r^{d-1}(\log r)^C)$. An alternative way for a value of r near to n is to let Algorithm CUT3 go only into a constant depth of recursion, and then solve the "sufficiently small" subproblems by an application of the procedure of Chazelle and Friedman. A straightforward analysis of this approach shows that we get a cutting of asymptotically optimal size, but the running time increases by a factor $O(n^\delta)$ ($\delta > 0$ is again arbitrary but fixed).

Let us turn to the planar case. Here we may replace the application of Algorithm CUT1 in Algorithm CUT2 by a more efficient procedure, which computes a $(1/r)$-cutting in linear time for a value of r bounded by a constant:

Theorem 3.1 [M2]. *Given a collection H of n lines, we can compute a $(1/r)$-cutting of size $O(r^2)$ for H in time $O(nr^2 \log r)$.*

For the running time of the modified version of Algorithm CUT2, we now obtain a recurrence

$$T_2(n, r) \leq T_2(n, r/2) + O(nr + r^B),$$

whose solution is immediately seen to satisfy $T_2(n, r) = O(nr + r^B)$. Further, we proceed exactly as for the case of a general dimension.

Let us remark that the algorithm from Theorem 3.1 can also replace the calls to the procedure of Chazelle and Friedman in Algorithm CUT2: according to Theorem 2.8, we may also apply it for a weighted collection of lines. We also note that an arbitrary planar cutting can be easily refined into a cutting which is a triangulation (i.e., a simplicial complex), and hence the computation of subproblems for any given cutting or a description relative to a given cutting poses no problem. Simulation of simplicity can be used to avoid degenerate cases.

4. Computing ε-Nets

4.1. A Special Case of the Greedy Algorithm for the Covering Problem

As subroutine of our algorithm for ε-nets, we need to solve the following combinatorial problem:

Let $G = (A, B, E)$ be a bipartite graph with vertex sets A and B and edge set $E \subseteq A \times B$. We want to find a subset $X \subseteq A$ as small as possible, such that every vertex $b \in B$ has a neighbor $a \in X$, i.e., such that the whole B is covered by the vertices of X. This problem can also be viewed as *set-covering problem* (if B corresponds to points, A to sets, and E to the incidence relation) or *hypergraph-transversal problem* (if the role of sets and points is interchanged); we find the bipartite graph view the most intuitive one.

It is well known that this problem is NP-complete and also well known is an approximate algorithm solving this problem, the so-called *greedy algorithm* (usually attributed to Lovász; see, e.g., [CF] for references). This algorithm proceeds as follows:

We put $A_1 = A$, $B_1 = B$. In the ith step we select a vertex $a_i \in A_i$, which has the maximum number of neighbors in B_i, and we set $A_{i+1} = A_i \backslash \{a_i\}$, $B_{i+1} = B_i \backslash Nbh(a_i)$, where $Nbh(a)$ denotes the set of all neighbors of a in G. We continue in this manner until B_{i+1} becomes empty. Then $X = \{a_1, \ldots, a_i\}$ is a covering subset. Obviously, the greedy algorithm can be implemented to run in time $O(|A| + |B| + |E|)$. We need the following result about the size of the solution found by the greedy algorithm:

Lemma 4.1 [CF]. *Suppose that every vertex $b \in B$ has at least $\varepsilon|A|$ neighbors in A ($\varepsilon > 0$ is a parameter). Then the greedy algorithm finds a covering subset $X \subseteq A$ of size at most $O((1/\varepsilon) \log|B|)$.*

Proof outline. We know that every vertex of B_i has degree greater than $\varepsilon|A|$, thus the total number of edges joining B_i to A_i is at least $\varepsilon|A| \cdot |B_i|$, and since $|A_i| \leq |A|$, there exists a vertex $a_i \in A_i$ with at least $\varepsilon|B_i|$ neighbors in B_i. We get that $|B_{i+1}| < |B_i| \cdot (1 - \varepsilon)$, hence the number of steps of the greedy algorithm we have to execute is at most $\log_{1-\varepsilon}|B| = O((1/\varepsilon)\log|B|)$. □

It is amusing that the greedy algorithm for our special case can also be viewed as a special case of the method of conditional probabilities of Raghavan and Spencer (see [R], [S], and also the Appendix). Namely, we can easily show that when choosing a random r-tuple of vertices of A, the expected number of uncovered vertices of B will be smaller than one for $r \geq (C/\varepsilon)\log|B|$ (C is a suitable constant). If we now assume that every vertex of B has *exactly* $\varepsilon|A|$ neighbors (which we may do, since we can remove the superfluous edges), then the algorithm given by the method of conditional probabilities and the greedy algorithm are exactly the same.

4.2. *Applying the Greedy Algorithm*

In this section we consider the computation of ε-nets for the range spaces of the form $H^k(X)$. Our procedure is almost identical to the one given in [M2] for the two-dimensional case (with one improvement).

First we define a dual counterpart of the range space $H^k(X)$; we need a slightly more general definition:

Let $\mathscr{C}$ be a collection of subsets of E^d and let k be a natural number (the interesting cases are $k \leq d+1$). A *k-combination* of $\mathscr{C}$ is an ordered pair $K = (\{X_1, X_2, \ldots, X_m\}, \{X_{m+1}, X_{m+2}, \ldots, X_k\})$, where $0 \leq m \leq k$ and the X_i are elements of $\mathscr{C}$. We say that a nonvertical hyperplane h realizes the k-combination K if $X_1, \ldots, X_m$ lie above h and $X_{m+1}, \ldots, X_k$ lie below h. Let H be a set of hyperplanes; we define a range space

$$DH^k(H) = (H, \{\{h \in H;\ h \text{ realizes } K\};\ K \text{ a } k\text{-combination of points of } \mathrm{E}^d\}).$$

Let X be a point set in general position. Then all ranges of $H^k(X)$ arise as intersections of X with H^k-ranges determined only by half-spaces with nonvertical bounding hyperplanes, and by the properties of duality transform, the range spaces $H^k(X)$ and $DH^k(D(X))$ are isomorphic, in particular, ε-nets are preserved by the transform.

The proof of the following lemma is identical to the proof of Lemma 5.2 of [M2]:

Lemma 4.2. *Let H be a set of n hyperplanes and let Ξ be an ε-cutting for H. Suppose that $N \subseteq H$ is a subset with the following property: Every k-combination of simplices of Ξ, which is realized by more than δn hyperplanes of H, is also realized by a hyperplane of N (let us call this a δ-covering property of N). Then N is a $(k\varepsilon + \delta)$-net for the range space $DH^k(H)$.* □

This lemma is a base of our construction of ε-nets for $DH^k(H)$. Suppose that we are given a $(1/r)$-cutting Ξ of size $O(r^d)$ for H. We define a bipartite graph (A, B, E) as follows: We put $A = H$, we let B be the set of all k-combinations of the simplices of Ξ realized by more than n/r hyperplanes of H and we put an edge (h, K) into E whenever the hyperplane $h \in H$ realizes the k-combination K. The size of B is $O(r^{kd})$ and every vertex of B has degree at least n/r, hence by Lemma 4.1 the greedy algorithm applied on this bipartite graph computes a covering subset of A (which is just a subset of hyperplanes with the $(1/r)$-covering property) of size $O(r \log r)$. By Lemma 4.2, this covering subset is a $((k+1)/r)$-net for $DH^k(H)$.

The above bipartite graph is unnecessarily large for our computation. We observe that when two hyperplanes h, h' are equivalent with respect to the set of vertices of Ξ, then they have the same neighborhood in our bipartite graph. If we leave only one hyperplane (vertex) of every equivalence class in the bipartite graph, then obviously the greedy algorithm attains a covering subset of exactly the same size as for the full graph. This reduced graph has a size polynomial in r. We thus first compute the description of H relative to Ξ (in time $O(nr^{d-1})$), and then we construct the reduced bipartite graph and apply the greedy algorithm on it. We summarize our considerations as follows:

Proposition 4.3. *Let $k \leq d+1$ be fixed. Given a walkable $(1/r)$-cutting of size $O(r^d)$ for H, we can compute a $((k+1)/r)$-net of size $O(r \log r)$ for $DH^k(H)$, in time $O(nr^{d-1} + r^B)$ ($B \leq kd + d^2$ is a constant).*

Using this proposition, we see that Theorem 1.2 can be deduced from Theorem 1.1. For a value of r close to n, which is prohibited in Theorem 1.1, we may use the $(1/n)$-cutting arising by canonical triangulation of the arrangment of H (computed in time $O(n^d)$) as a starting point in the previous proposition. The factor r^B will be much larger than n^d, and so this will not increase the bound on the running time.

5. Applications for Other Geometric Objects

Yao and Yao [YY] observed that an algorithm for half-space range queries in a general fixed dimension can also be used for answering other types of geometric queries, using a simple "lifting" transformation on the problem in question. As noted by Welzl, a similar argument shows that questions about ε-nets in range spaces arising in computational geometry can be reduced to the case of range spaces of the form $H^k(X)$ (with k and the dimension d fixed).

Let us say that a range space (X, R) is *embeddable* in a range space (Y, S) if there exists an injective mapping $\varphi: X \to Y$, such that, for every $r \in R$, $\varphi(r)$ can be expressed as $s \cap \varphi(X)$ for some $s \in S$ (this definition appears in [AHWW]). Obviously if we can determine ε-nets for subspaces of the space (Y, S), then ε-nets for subspaces of (X, R) can be determined as inverse images under φ.

In range spaces encountered in computational geometry, the ranges can usually be described by conjunctions and disjunctions of fixed-degree polynomial inequalities. Disjunctions are easy to handle in ε-net construction—we can just form a union of ε-nets. It is not difficult to see that range spaces with ranges contained in E^d and defined by a conjunction of k fixed-degree polynomial inequalities are embeddable in range spaces of the form $H^k(X)$, where $X \subseteq E^{d'}$ for some d' bounded by a constant. The idea is to assign one coordinate in the image space to every monomial occurring in the defining polynomials. For example, the range space in E^3 with ranges defined by balls of the form

$$\{(x, y, z); (x - a)^2 + (y - b)^2 + (z - c)^2 \leq r^2\} \tag{2}$$

can be embedded as follows: the mapping φ: $E^3 \rightarrow E^6$ will be defined by

$$\varphi(x, y, z) = (x, x^2, y, y^2, z, z^2),$$

and then the points in the sphere (2) will be those which are mapped into the half-space

$$x_2 - 2ax_1 + x_4 - 2bx_3 + x_6 - 2cx_5 \leq r^2 - a^2 - b^2 - c^2.$$

This example also shows that the general method does not always give the most efficient embedding; the range space defined by balls can be represented by half-spaces in dimension 4, using the well-known lifting to the paraboloid.

Finding ε-nets can also be applied for cutting arrangements of more complicated hypersurfaces (than hyperplanes). The size of the cuttings obtained depends crucially on the ability to divide the cells of an arrangement of the hypersurfaces considered into constant-complexity cells. The most general known results in this direction are given in [CEGS].

Let us again give a concrete example for a collection of circles in the plane (where no problems with triangulations arise). The geometric fact we need is that an arrangement of n circles can be vertically subdivided into $O(n^2)$ "circular trapezoids" (figures bounded by two vertical segments and two circular arcs). Given a collection H of n circles, we define a range space

$$(H, \{H_R;\ R \text{ is a circular trapezoid}\}),$$

where $H_R = \{h \in H;\ h \cap R \neq \emptyset\}$. Let us construct a $(1/r)$-net N for this range space using Theorem 1.2 (and the embedding method described above). When we construct the vertical subdivision of the arrangement of N, then every cell will be intersected by at most n/r circles of H (note that it is not sufficient to compute a $(1/r)$-net for ranges defined on H by *segments* only, as is the case for a collection of *lines*). In this way, we get something like a $(1/r)$-cutting of size $O((r \log r)^2)$ for H. We can afford to refine this cutting a constant number of times (similarly as in Section 2.3). The important fact here is that an intersection of two circular trapezoids can be subdivided into a constant number of circular trapezoids. Therefore we can always work with small values of r in the ε-net algorithm and get an overall running time of order $O(nr^{1+\delta})$ (for arbitrarily small fixed $\delta > 0$). The price we pay for this is a high-degree polylogarithmic factor in the size of the resulting refined cutting.

Let us remark that in some cases, our algorithm for finding cuttings for hyperplanes can be directly modified for curved surfaces (obtaining a better result than by the above method with ε-nets), but the approach via ε-nets seems to be simpler and more general.

6. Conclusion

In this paper we give deterministic algorithms for finding asymptotically optimal cuttings and ε-nets (of the best size guaranteed by known existence proofs) for "geometric" range spaces, all this in theoretically reasonable time. An important open problem is to improve the $O(r^B)$ factor in the computation of $(1/r)$-nets in Theorem 1.2. Our result is satisfactory for small values of r only and it is quite inadequate, e.g., for $r = \sqrt{n}$. Also, the computation of $(1/r)$-cuttings is unsatisfactory for r approaching n (e.g., $r = n/\log n$), which is sometimes required for applications.

We have avoided the issue of parallel implementations of our algorithms. The only nontrivial part in this respect is the method of conditional probabilities (or the greedy algorithm in the case of ε-nets). It seems that here we should be able to use the results of [BRS] and [BR] without much difficulty, but this has not been elaborated in detail.

Appendix. Computing Optimum-Sized Cutting in Polynomial Time

Here we give the proof of Theorem 2.7 for the unweighted case, essentially following Chazelle and Friedman [CF]. The proof also goes through for the weighted case with trivial modifications (replacing cardinalities by weights everywhere); we prefer to give it for the unweighted case.

First we have to say more about canonical triangulations; for the proofs we refer to [CF]. If S is a collection of hyperplanes, let $\mathrm{CT}(S)$ denote the set of full-dimensional simplices of the canonical triangulation of the arrangement of S The canonical triangulation has the following properties:

Lemma A.1 [CF]. *Let H be a collection of hyperplanes in E^d.*

(i) *For every simplex s of $\mathrm{CT}(H)$, there exists a unique inclusion-minimal collection $S(s) \subseteq H$, such that $s \in \mathrm{CT}(S)$. This collection $S(s)$ has at most a constant number $D = d(d+3)/2$ of hyperplanes.*

(ii) *If S is a subcollection of H and s is a simplex $\mathrm{CT}(S)$, then s belongs to the canonical triangulation of H iff its interior is intersected by no hyperplane of H.*

Part (i) of this lemma implies that there are only polynomially many (in $|H|$) simplices which can ever appear in the canonical triangulation of a sample drawn from H—namely, those belonging to the canonical triangulation of some subset of at most D hyperplanes of H. Let us denote the set of these "candidate simplices" by $\mathscr{T}(H)$.

Part (ii) gives us a criterion when a candidate simplex $s \in \mathscr{T}(H)$ appears in the canonical triangulation of a sample $R \subseteq H$: this is iff

(a) $S(s) \subseteq R$ and

(b) $H_s \cap R = \emptyset$.

Now we can begin with the construction of the desired small $(1/r)$-cutting for H. The proof is probabilistic; the derandomization comes in the end.

Let us draw a random sample R from H in such a way that every hyperplane $h \in H$ is drawn with probability r/n and the choices are independent (thus the expected size of R is just r).

Let us consider the canonical triangulation $\mathrm{CT}(R)$ (call its simplices the *first-generation simplices*). The sample R is too rough to guarantee a cutting factor of order r, at least using the usual probabilistic argument. We refine this cutting similarly as in Section 2.3, but we are more careful about the cutting factors of the secondary cuttings. Namely, for every first-generation simplex s, we compute a $(1/t_s)$-cutting Ξ_s for H_s, where t_s is the factor by which $|H_s|$ exceeds the quantity n/r, thus we set $t_s = |H_s| \cdot r/n$.

Let us assume for this moment that we can make the size of each of the secondary cuttings Ξ_s polynomial in t_s, say at most t_s^c (this is a much milder requirement than the asymptotically optimal size t_s^d). Then the total size $\Phi(R)$ of the resulting cutting $\Xi(R)$ is

$$\Phi(R) \leq \sum_{s \in \mathrm{CT}(R)} t_s^c,$$

and our goal is to show that the expected value of this quantity is of the same order as the expected size of $\mathrm{CT}(R)$, i.e., $O(r^d)$.

The reason why this is true is that the existence of many simplices s with large t_s is extremely improbable, more exactly the expected number of simplices with $t_s \geq t$ decreases exponentially with t.

Let us prove this. We put $\mathscr{T} = \mathscr{T}(H)$ and $\mathscr{T}_t = \{s \in \mathscr{T};\, |H_s| \geq tn/r\}$ (these are those of the candidate simplices for which the number of hyperplanes intersecting it exceeds the number allowed for a $(1/r)$-cutting by a factor at least t).

Let t be a parameter and let a random variable n_t be the number of simplices s of the canonical triangulation of R, whose interior intersects more than tn/r hyperplanes of H. We bound the expectation $\mathrm{E}n_t$. This expectation can be expressed as

$$\sum_{s \in \mathscr{T}_t} p(s),$$

where $p(s)$ denotes the probability that a simplex s belongs to the canonical triangulation of R. Now, using Lemma A.1, $p(s)$ is the probability that each of the hyperplanes of $S(s)$ falls into R and none of the hyperplanes of H_s does, thus

$$\mathrm{E}n_t = \sum_{s \in \mathscr{T}_t} \left(\frac{r}{n}\right)^{|S(s)|} \left(1 - \frac{r}{n}\right)^{|H_s|}. \tag{3}$$

In order to estimate this sum from above, we consider another random sample $R' \subseteq H$, where a hyperplane $h \in H$ is chosen with probability $(r/n)/t$, thus we pick approximately r/t hyperplanes. Let n' denote the number of simplices in the canonical triangulation of R'. Since the expected size of R' is r/t, intuitively the expectation of n' should be at most of order $O((r/t)^d)$, and this is indeed true (this is shown by estimating the expectation of $|R'|^d$). On the other hand, we can write

$$\begin{aligned} \mathrm{E}n' &= \sum_{s \in \mathscr{T}} \left(\frac{r}{tn}\right)^{|S(s)|} \left(1 - \frac{r}{tn}\right)^{|H_s|} \\ &\leq \sum_{s \in \mathscr{T}_t} t^{-D} \left(\frac{r}{n}\right)^{|S(s)|} \left(1 - \frac{r}{n}\right)^{|H_s|} \cdot \left(\frac{1 - r/(tn)}{1 - r/n}\right)^{|H_s|} \end{aligned}$$

Every term in the last sum is just the corresponding term in (3) multiplied by the factor

$$f(s) = t^{-D} \left(\frac{1 - r/(tn)}{1 - r/n}\right)^{|H_s|}$$

Now we may assume that $r/n \leq \frac{1}{2}$ (for larger r, the canonical triangulation of the arrangement of H will do as the desired $(1/r)$-cutting). Using the inequalities $1 - x \leq e^{-x}$ (valid for all x) and $1 - x \geq e^{-2x}$ (valid for $x \leq \frac{1}{2}$), we estimate the above factor by

$$f(s) \geq t^{-D} \exp\left(|H_s|\left(-\frac{2r}{tn} + \frac{r}{n}\right)\right).$$

Since we consider only simplices $s \in \mathscr{T}_t$, i.e., with $|H_s| \geq tn/r$, we get

$$f(s) \geq t^{-D} e^{t-2},$$

and thus we have

$$\mathrm{E}n_t \leq t^D e^{-(t-2)} \mathrm{E}n' \leq O(r^d) t^{D-d} e^{-(t-2)}.$$

We may now return to the expected value of the size $\Phi(R)$ of the cutting $\Xi(R)$ constructed by the above two-step process (triangulating the arrangement of R and refining the cutting for every simplex separately). This expectation will be bounded by

$$\mathrm{E}\Phi(R) \leq \sum_{t=1}^{n} t^c \cdot \mathrm{E}n_t \leq O(r^d) \sum_{t=1}^{n} t^{D-d+c} e^{-(t-2)} = O(r^d).$$

Now we have a randomized precedure which allows us to compute a $(1/r)$-cutting of an asymptotically optimal size. Implementing it carefully, we can achieve an expected running time $O(nr^{d-1})$. The method of conditional probabilities of Raghavan [R] and Spencer [S] allows us to make the computations deterministically in polynomial (although much longer) time.

In our setting the method is applied as follows. We want to compute a sample $\bar{R}$, for which the value of $\Phi(\bar{R})$ is not bigger than the expectation $\mathrm{E}\Phi(R)$ (or, at least, of the same order). We order the hyperplanes of H into a sequence $h_1, h_2, \ldots, h_n$ and we note that the random selection of R can be viewed as a choice of the characteristic vector $(\chi_R(h_1), \ldots, \chi_R(h_n))$ (where $\chi_R(h) = 1$ if $h \in R$, $\chi_R(h) = 0$ otherwise), whose components are independent random 0/1 variables, each of them having the value 1 with probability r/n.

For a 0/1 vector $(\rho_1, \rho_2, \ldots, \rho_i)$, we set

$$E(\rho_1, \ldots, \rho_i) = \mathrm{E}(\Phi(R) | \chi_R(h_1) = \rho_1, \ldots, \chi_R(h_i) = \rho_i).$$

In other words, $E(\rho_1, \ldots, \rho_i)$ is the expected value of $\Phi(R)$ if the first i entries of χ_R are fixed as $(\rho_1, \ldots, \rho_i)$ and the remaining ones are chosen randomly. Thus $E(\chi_R(h_1), \ldots, \chi_R(h_n))$ for a specific R is nothing but $\Phi(R)$, while E (without parameters, i.e., for $i = 0$) is just the expected value of $\Phi(R)$ for a random choice of R.

Suppose that $\rho_1, \ldots, \rho_{i-1}$ have been fixed in such a way that $E(\rho_1, \ldots, \rho_{i-1}) \leq E$ (this is trivially true for $i = 1$). We compute values $V_0 = E(\rho_1, \ldots, \rho_{i-1}, 0)$ and $V_1 = E(\rho_1, \ldots, \rho_{i-1}, 1)$. The properties of conditional expectation guarantee that $\min(V_0, V_1) \leq E(\rho_1, \ldots, \rho_{i-1})$, and we can thus set ρ_i so that $E(\rho_1, \ldots, \rho_i) = V_{\rho_i} \leq E$. In this way we finally compute a vector $(\rho_1, \ldots, \rho_n)$, which determines a sample R with $\Phi(R) \leq E$.

We have to assure that the conditional expectations $E(\rho_1, \ldots, \rho_i)$ can be evaluated in polynomial time. We observe that $E(\rho_1, \ldots, \rho_i)$ can be expressed as a sum similar to (3), where the summand for a gives simplex s means the probability that s will occur in the canonical triangulation of R (when the first i components of the characteristic vector of R are fixed to $(\rho_1, \ldots, \rho_i)$). Moreover, it suffices to evaluate the expectations with accuracy $O(n^{-2})$ (say), since we can afford to lose this quantity in the size of the resulting cutting in each of the n steps of the choice. It is not difficult to see that these computations can be done in polynomial time.

We have left aside the question of how to compute the secondary cuttings inside the first-generation simplices. Recall that the problem here is to compute a $(1/r)$-cutting of size polynomial in r for a given collection of hyperplanes. Here we may proceed as follows: picking a larger random sample R of the given collection of hyperplanes (of size $Cr \log r$, C a sufficiently large constant), the expected number of simplices intersected by more than n/r hyperplanes will be smaller than 1, and thus the method of conditional probabilities allows us to construct a $(1/r)$-cutting of size $O((r \log r)^d)$. Another (but actually quite similar) approach is via ε-nets; this has been elaborated on in the conference version of this paper. In both these cases these secondary cuttings are standard and thus the whole cutting is walkable. This finishes the proof.

References

[AESW] P. K. Agarwal, H. Edelsbrunner, O. Schwarzkopf, and E. Welzl. Euclidean minimum spanning trees and bichromatic closest pairs. In *Proc. 6th ACM Symposium on Computational Geometry*, pages 203–210, 1990.

[A1] P. K. Agarwal. Partitioning arrangments of lines, I: An efficient deterministic algorithm. *Discrete & Computational Geometry*, **5**:449–483, 1990.

[A2] P. K. Agarwal. Partitioning arrangements of lines, II: Applications. *Discrete & Computational Geometry*, **5**:533–573, 1990.

[AHU] A. V. Aho, J. E. Hopcroft, and J. D. Ulman. *The Design and Analysis of Computer Algorithms*. Addison-Wesley, Reading, MA, 1983.

[AHWW] N. Alon, D. Haussler, E. Welzl, and G. Wöginger. Partitioning and geometric embedding of range spaces of finite Vapnik–Chervonenkis dimension. In *Proc. 3rd ACM Symposium on Computational Geometry*, pages 331–340, 1987.

[BR] B. Berger and J. Rompel. Simulating $(\log n)^c$-wise independence in NC. In *Proc. 30th IEEE Symposium on Foundations of Computer Science*, pages 2–7, 1989.

[BRS] B. Berger, J. Rompel, and P. W. Shor. Efficient NC algorithms for set cover with applications to learning and geometry. In *Proc. 30th IEEE Symposium on Foundations of Computer Science*, pages 54–59, 1989.

[CEG] B. Chazelle, H. Edelsbrunner, and L. Guibas. Lines in space: Combinatorics, algorithms and applications. In *Proc. 21st ACM Symposium on Theory of Computing*, pages 389–392, 1989.

[CEGS] B. Chazelle, H. Edelsbrunner, L. Guibas, and M. Sharir. A singly-exponential stratification scheme for real semi-algebraic varieties and its applications. In *Proc. 16th International Colloquium on Automata, Languages and Programming*, pages 179–192, 1989.

[CF] B. Chazelle and J. Friedman. A deterministic view of random sampling and its use in geometry. *Combinatorica*, **10**:229–249, 1990.

[C1] K. L. Clarkson. Applications of random sampling in computational geometry, II. In *Proc. 4th ACM Symposium on Computational Geometry*, pages 1–11, 1988.

[C2] K. L. Clarkson. A randomized algorithm for closest-point queries. *SIAM Journal on Computing*, **17**:830–847, 1988.

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

[EGH*] H. Edelsbrunner, L. Guibas, J. Herschberger, R. Seidel, M. Sharir, J. Snoeyink, and E. Welzl. Implicitly representing arrangements of lines or segments. *Discrete & Computational Geometry*, **4**:433–466, 1989.

[EM] H. Edelsbrunner and P. Mücke. Simulation of simplicity: A technique to cope with degenerate cases in geometric algorithms. *ACM Transactions on Computer Graphics*, **9**:66–104, 1990.

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

[HW] D. Haussler and E. Welzl. ε-nets and simplex range queries. *Discrete & Computational Geometry*, **2**:127–151, 1987.

[M1] J. Matoušek. Approximate half-planar range counting. KAM Series (technical report) 59–87, Charles University, Prague, 1987. A revised version (Approximate levels in line arrangements) in *SIAM Journal on Computing*, **20**(2):222–227, 1991.

[M2] J. Matoušek. Construction of ε-nets. *Discrete & Computational Geometry*, **5**:427–448, 1990.

[M3] J. Matoušek. More on cutting arrangements and spanning trees with low crossing number. Tech. Report B-90-2, FB Mathematik, FU Berlin, 1990.

[M4] J. Matoušek. Approximations and optimal geometric divide-and-conquer. In *Proc. 23rd ACM Symposium on Theory of Computing*, pages 506–511, 1991.

[M5] J. Matoušek. Efficient partition trees. In *Proc. 7th ACM Symposium on Computational Geometry*, pages 1–9, 1991.

[PW] J. Pach and G. Wöginger. Some new bounds for epsilon-nets. In *Proc. 6th ACM Symposium on Computational Geometry*, pages 10–15, 1990.

[R] P. Raghavan. Probabilistic construction of deterministic algorithms: approximating packing integer programs. In *Proc. 27th IEEE Symposium on Foundations of Computer Science*, pages 10–18, 1986.

[S] R. Spencer. *Ten Lectures on the Probabilistic Method*. CBMS–NSF, SIAM, Philadelphia, PA, 1987.

[SWM] R. Seidel, E. Welzl, and J. Matoušek. Netting a lot with a little: Small ε-nets for disks and half-spaces. In *Proc. 6th ACM Symposium on Computational Geometry*, pages 16–22, 1990.

[VC] V. N. Vapnik and A. Ya. Chervonenkis. On the uniform convergence of relative frequencies of events to their probabilities. *Theory of Probability and Its Applications*, **16**:264-280, 1971.

[YY] F. F. Yao and A. C. Yao. A general approach to geometric queries. In *Proc. 17th ACM Symposium on Theory of Computing*, pages 163–168, 1985.

Received July 5, 1990, and in revised form February 19, 1991.