Cutting hyperplane arrangements*

JIŘÍ MATOUŠEK
Department of Computer Science,
Charles University
Malostranské nám. 25, 118 00 Praha 1, Czechoslovakia

Abstract

We will consider an arrangement H of n hyperplanes in E^d (where the dimension d is fixed). An *ε-cutting* for H will be 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, finding a $(1/r)$-cutting with $O(r^d(\log r)^C)$ simplices in time $O(n(\log n)^A r^{d-1}(\log r)^B)$ (A,B,C are constants dependent on dimension). In a similar time bound (with an additional $O(r^{O(1)})$ overhead) we can also find a $(1/r)$-net for the range space $(X, H(X))$, where X is a n-point set in E^d and $H(X)$ denotes the set of all subsets of X which can be cut by a halfspace. This $(1/r)$-net has size $O(r \log r)$, which matches the best known existence result; in fact, the method gives a constructive existence proof. In the plane, we can obtain a $(1/r)$-cutting of optimal size $O(r^2)$ in time $O(nr)$ (which is optimal if we want to compute also the collection of lines intersecting each simplex of the cutting). This improves the result of Agarwal, and our algorithm is conceptually simpler.

*This research has been performed while the author was visiting DIMACS at Princeton University

1 Introduction and statement of results

Algorithmic and proof techniques based on random sampling gained a central position in computational geometry during last few years. These techniques, pioneered by Clarkson (e.g.,[Cla]) and Haussler and Welzl [HW] yield nearly 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 usually provides an efficient geometric divide and conquer strategy. For two-dimensional applications see e.g. [C&al], [E&al] and many others.

This motivates the following definition: 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 will be the number of its simplices (the total number of faces of all dimensions is proportional to the size). An *ε-cutting* for an arrangement H of n hyperplanes is a cutting such that the interior of any its simplex is intersected by at most εn hyperplanes of H. The number ε is called the *cutting factor* of an ε-cutting (several previous papers use other names for cutting, as e.g. *partitioning* [Aga] or *simplicial packing*

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

[CF]). Note that we did not require that a cutting were a simplicial complex (after including faces of all dimensions).

Chazelle and Friedman [CF] proved that for every H there exists a $(1/r)$-cutting of asymptotically optimal size, namely $O(r^d)$. They also gave a deterministic algorithm finding 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 $O(nr^{d-1})$.

For dimension 2, Matoušek [Ma] 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 [Aga] to $O(nr \log n \log^\omega r)$ ($\omega < 3.3$ is a constant), and the companion paper [Aga1] gives an extensive survey of applications.

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

In this paper we prove the following result:

Theorem 1.1 *Given a collection H of n hyperplanes in E^d and a number $r \leq n$, we can deterministically compute a $(1/r)$-cutting of size $O(r^d(\log r)^C)$ for H, in time $O(n(\log n)^A r^{d-1}(\log r)^B)$ (A,B,C are constants depending on dimension).*

The proof of this theorem extends the techniques of [Ma1], [Ma], and adds some new ingredients. For the sake of readability, we will not try to achieve the best values of A,B,C.

When the technique is applied in dimension 2 together with some other results of [Ma] and a few other tricks, we obtain the following:

Theorem 1.2 *Given a collection H of n lines in the plane and a number $r \leq n$, we can deterministically compute a $(1/r)$-cutting of size $O(r^2)$ for H, in time $O(nr)$.*

This is similar to the result of [Aga], but the algorithm is conceptually simpler and more efficient.

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*). A subset $N \subseteq X$ is called an ε-net for (X, R) (ε is a nonnegative real number), 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 halfspaces in Euclidean space have a special significance for computational geometry. If X is a subset of E^d, we denote by $H(X)$ the set of all subsets of X, which can be obtained as the intersection of X with a halfspace. The range space $(X, H(X))$ will be denoted by $Halfsp(X)$. In these special range spaces, we have the following constructive analogue of the general existence result of Haussler and Welzl [HW]:

Theorem 1.3 *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 $Halfsp(X)$, in time $O(n(\log n)^A r^{d-1}(\log r)^B + r^C)$ (A,B,C are constants dependent on dimension).*

(The constants A,B,C in the above theorem are not necessarily the same as in Theorem 1.1, and a similar remark will apply for other constants occurring in this paper, since we do not want to introduce myriads of them.)

Let us remark that our bound on the size of 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 $Halfsp(X)$ this is an open problem.

The above result may seem to apply to very special range spaces only, but a simple transformation allows to extend it to all range spaces usually encountered in computational geometry (this observation is due to Yao and Yao [YY]). Let us say that a range space (X, R) is *representable* 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 $s \in S$. Now the observation says that any range space which is representable in a space of the form (m,d are

constants)

$$(E^d, \{\{(x_1,\ldots,x_d) \in E^d;\ p(x_1,\ldots,x_d) \geq 0\};$$

$$p \text{ a } d\text{-variate polynomial of degree} \leq m\})$$

(which is true for "usual" geometric range spaces) is also representable in a range space of the form $Halfsp(E^{d'})$, where d' depends on d and m only. To see this, if suffices to assign to every possible monomial in $x_1,\ldots,x_d$ of degree $\leq m$ one coordinate in the space $E^{d'}$. This shows that we can use Theorem 1.3 e.g. for range spaces defined by circles, spheres, simplices etc.

The above results allow to remove randomization from many algorithms without a significant loss of efficiency; nice examples of such algorithms are e.g. in [CF], [CEG], [HW].

In section 2 we give some definitions and auxiliary results. Section 3 describes some operations computing new cuttings from old ones, the most important one being an algorithm allowing to decrease the size of a too large cutting, paying some price in its cutting factor. In this section we also deduce Theorem 1.3 from Theorem 1.1. Section 4 then uses the primitives from the previous section in a recursive fashion, and builds the algorithm of Theorem 1.1 (first an algorithm efficient for small values of r only, then the algorithm for a general case). In section 5 we sketch the proof of Theorem 1.2.

2 Preliminaries on arrangements and cuttings

We will consider an arrangement H of n hyperplanes in E^d. We will assume that the hyperplanes are in a general position where convenient. For terminology about arrangements as well as about the general position assumption see [Ede].

By a *distance* of two points in an arrangement we mean the number of hyperplanes separating these points (this is a pseudometrics).

The arrangement (as a cell complex) can be triangulated in various ways; we will use the so-called *canonical triangulation* (see e.g. [Cla1] or [CF] for definition; roughly speaking, each face is triangulated from its vertex with the smallest x_1-coordinate). This triangulation determines a simplicial complex with $O(n^d)$ simplices; these are the only properties we will use. Given the hyperplanes, both the arrangement and the triangulation can be constructed in time $O(n^d)$ [EOS].

Let $Seg(H)$ denote the range space (H, S), where S is the set of all subsets of H, which are defined as those hyperplanes of H intersecting some segment. A *weak ε-net* for $Seg(H)$ is a set R of hyperplanes (not necessarily belonging to H), such that any segment intersecting more than εn hyperplanes of H also intersects a hyperplane of R. The following is a well-known observation and we omit the proof here:

Lemma 2.1 *Let R be a weak $(1/r)$-net for $Seg(H)$. Then any triangulation of the arrangement of R is a (d/r)-cutting for H.* □

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

For the manipulations with a cutting Ξ, we will need some information about the position of the hyperplanes of H relative to the vertices of the cutting. However, the size of Ξ will often be much smaller than n, and we would like to have a description whose size depends on the size of Ξ only.

We call two hyperplanes *equivalent* with respect to a point set P, if they separate the points of P in the same manner. A *description of H relative to a cutting Ξ* will be the collection of all nonempty equivalence classes of the hyperplanes of H with respect to the set of vertices of Ξ, each class represented by one its member and by the number of members.

If Ξ has k vertices, then there are no more than k^d equivalence classes there.

A specific application for which a description of H relative to Ξ can be used is an approximate distance calculation: given two points x,y, we can determine their distance with error no greater than $2n/r$ in time depending only polynomially on k (the size of Ξ) and independent on n (we determine the simplices s, resp. s' into which x, resp. y belong to, and we sum up the sizes of all equivalence classes of hyperplanes separating s and s'). Of course, we can do this distance measurement with various degree of sophistication and also we

actually need much less information than the description of H relative to Ξ, but for our purposes the important thing is solely the polynomial dependence on k.

Lemma 2.2 *Given a collection H of n hyperplanes and a standard $(1/r)$-cutting Ξ for H of size k, we can compute the description of H relative to Ξ in time $O(nk/r)$ (for a general cutting of size k, the description is easily computed in $O(nk)$ time).*

Proof: The equivalence class of a hyperplane is uniquely determined by the set of edges of Ξ it intersects, so it suffices to determine this set for every hyperplane of H.

First suppose that we already know one simplex intersected by a hyperplane h. The remaining simplices (and thus edges) 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 be 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: choose a path of edges of Ξ going from the wertex with minimum x_1 coordinate to the vertex with maximum x_1 coordinate. Then if a given hyperplane separates the end-vertices of this path, some edge of intersection can be determined by a binary search along the path, otherwise the hyperplane must intersect one of a constant number of rays of the cutting emanating from the end-vertices of the path.

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 [AHU]. Since Ξ is a $(1/r)$-cutting, the total number of edge/hyperplane incidences is $O(nk/r)$. □

3 Operations with cuttings

3.1 Refining a cutting

This section contains a simple observation. Suppose that we are given a $(1/r_1)$-cutting Ξ for H of size k_1. Let H_s be the collection of hyperplanes intersecting a simplex s of Ξ, and let Ξ_s be a $(1/r_2)$-cutting for H_s of size at most k_2. Then we can obtain a $(1/r_1r_2)$-cutting of size $O(k_1k_2)$ 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 (obviously there are at most k_1k_2 such cells). Each cell is defined as the intersection of two simplices, and thus it can be triangulated using a constant number of simplices. Taking the simplices of such triangulations of all cells, we obtain the desired $(1/r_1r_2)$-cutting.

3.2 Simplifying a cutting

Theorem 3.1 *Let Ξ be a $(1/r)$-cutting of size k for H, and suppose that we are able to determine the distance of any given two points with error at most $2n/r$ in time polynomial in k. Then the following holds (K, K' denote certain constants dependent on dimension):*

(i) One can find a weak (K/r)-net of size $O(r \log k)$ for the range space $Seg(H)$, in time polynomial in k (and independent of n).

(ii) If Ξ is standard, then with additional $O(nk/r)$ time one can find a (K/r)-net of size $O(r \log k)$ for the range space $Seg(H)$.

(iii) One can find a standard (K'/r)-cutting of size $O((r \log k)^d)$ for H, in time polynomial in k.

The most interesting part for our application is (iii), which gives a means to simplify a cutting to almost optimal size (while the cutting factor is blown up by a constant factor K'). Let us note that (iii) easily follows from (i): We use (i) to get a weak $(1/r)$-net, we triangulate the arrangement of its hyperplanes and we get the desired standard cutting by Lemma 2.1.

Any ε-net in the range space $Seg(H)$ obviously determines an ε-net in the range space $Halfsp(D(H))$ (where $D(H)$ denotes the set of points dual to the set H of hyperplanes). Applying Theorem 3.1(ii), we get that Theorem 1.1 implies Theorem 1.3: First we find a $(1/r)$-cutting

(by Theorem 1.1) of size $k = O(r^d(\log r)^C)$, and then we use Theorem 3.1(ii) to get a (K/r)-net.

Let us begin the proof of Theorem 3.1(i) and (ii). First we reformulate the problem as the search for a covering subsystem in a set system, or (perhaps more intuitively) a covering set of vertices in a bipartite graph.

The following lemma is a trivial generalization of Lemma 5.2(i) of [Ma] and we omit the easy proof:

Lemma 3.2 *Let Ξ be a $(1/r)$-cutting for H and let R be a set of hyperplanes such that every pair of simplices $s, s' \in \Xi$ separated by more than cn/r hyperplanes of H is also separated by a hyperplane of R. Then R is a weak $((c+2)/r)$-net for $Seg(H)$.* □

For a collection H of hyperplanes and a cutting Ξ, we define a bipartite graph G as follows: One set of vertices of G will be H (the set of hyperplanes), and the other one will be the set P of all pairs of simplices s,s' of the cutting Ξ, such that s and s' are separated by at least n/r hyperplanes of H. A vertex $h \in H$ is joined to a pair $(s, s') \in P$ iff h separates s from s'. Now we want to cover all vertices of P by (the neighborhood of) a small subset of vertices of H. We use the greedy algorithm of Lovász, in a similar spirit as it is used in [CF]:

We put $H_1 = H$, $P_1 = P$. In i-th step, we select a vertex $h_i \in H_i$, which has the maximum number of neighbors in P_i, and we set $H_{i+1} = H_i \setminus \{h_i\}$, $P_{i+1} = P_i \setminus Nbh(h_i)$, where $Nbh(h)$ denotes the set of all neighbors of h in G. We continue in this manner until P_{i+1} becomes empty.

We recall the argument bounding the size of a solution gained by the greedy algorithm ([CF]): We know that every vertex of P has degree greater than n/r, thus the total number of edges joining P_i to H_i is at least $n|P_i|/r$, and since $|H_i| \leq n$, there exists a vertex $v \in H_i$ with at least $|P_i|/r$ neighbors in P_i. We get that $|P_{i+1}| < |P_i|.(1 - 1/r)$, hence the number of steps of the greedy algorithm we have to execute is $O(r \log |P|) = O(r \log k)$.

Since by the assumption of Theorem 3.1 we can only measure distances with error $2n/r$, we cannot exactly tell which pairs of simplices should belong to P. However, if we take e.g. all pairs for which the approximate measurement gave distance $> (5n/r)$, then on one hand they are really separated by $> n/r$ hyperplanes (thus the argument bounding the solution size works), and on the other hand any pair separated by $> (7n/r)$ hyperplanes has been taken into account, so any covering set of vertices in our bipartite graph is surely a $(9/r)$-net.

We have almost achieved our goal (finding a small (K/r)-net for $Seg(H)$), but the running time still depends of n. First we note that it suffices to work with the description of H relative to Ξ, which by Lemma 2.2 can be obtained in time $O(nk/r)$, and the complexity of the greedy algorithm then only depends polynomially on k, thus Theorem 3.1(ii) is proved.

To achieve running time independent of n (which will be crucial for our application), we cannot afford to consider every hyperplane in H separately. What we can do, however, is to compute a set H' containing one hyperplane of every *possible* equivalence class with respect to the vertices of the cutting Ξ. Using duality, this amounts to the construction of an arrangement of k hyperplanes (dual to the vertices) and choosing a point in some of its cells (corresponding to the separation of appropriate sets of vertices), which can be done in time polynomial in k.

Now we run the greedy algorithm on the bipartite graph G', where H is replaced by H'. Since for every $h \in H$ there exists a $h' \in H'$ with the same set of neighbors in P, the greedy algorithm in i-th step always selects a vertex of H'_i of degree at least $|P_i|/r$ and so in this case $O(r \log k)$ steps also suffice (however, we obtain a weak (K/r)-net only). This proves Theorem 3.1(i). □

Let us close this section by a remark on the existence of ε-nets. In the discussion of the algorithm finding ε-nets for range spaces $Seg(H)$, we will once make use of the general existence result for ε-nets, namely in Section 3.3. However, Theorem 3.1 can be easily used to deduce the existence of $(1/r)$-nets of size $O(r \log r)$ in range spaces $Seg(H)$ and thus also in other "geometric" range spaces (but we will not do this here), so our presentation might be quite self-contained in this.

3.3 Merging cuttings

Theorem 3.3 *Let $H_1, \ldots, H_m$ be disjoint collections of hyperplanes and let Ξ_i be a $(1/r)$-cutting of size at most k for H_i. We can compute a standard (K/r)-cutting Ξ (K a certain constant*

dependent on dimension) of size $O((r\log r)^d)$ for $H = H_1 \cup \ldots \cup H_m$, in time $O(nk + m^{O(1)}k^{O(1)})$, where $n = |H|$.

Proof: We consider the cell complex arising by superimposing all the cuttings $\Xi_1, \ldots \Xi_m$ (the regions are the maximal subsets of E^d intersected by no facet of simplices of the Ξ_i's), and we triangulate the full-dimensional regions of this complex. This cutting Θ has a polynomial number (in k and m) of simplices and it is easy to see that it is a $(1/r)$-cutting for H. Now it remains to apply the simplification algorithm from Theorem 3.1 in a suitable way to achieve the desired smaller size.

As for the distance measurement with respect to H, required by Theorem 3.1, it suffices to compute the description of each H_i relative to Ξ_i (which takes time at most $O(|H_i|k)$ for each i, thus $O(nk)$ in total). Then the distance of two points in H is computed as the sum of the distances of these points in every H_i, and the total error is at most $2n/r$.

The first application of the simplification algorithm gives a cutting of size $O((r\log(mk))^d)$, which is fine if mk is polynomial in r. If it is not the case, we may use a trick appearing in [CF]: we repeat the simplification step once more, obtaining size $O((r(\log r + \log\log(mk)))^d)$, etc. If we do not obtain a cutting of the appropriate size after a few more repetitions of the simplification step, it means that the quantity mk is enormous compared to r (more than doubly exponential), and also compared to the size of the bipartite graph we have obtained by the reductions in Theorem 3.1. Then instead of applying the greedy algorithm, we can simply find the optimal solution for the covering problem in this bipartite graph by an (exponentially long) exhaustive search. The theory of ε-nets guarantees that a solution of the right size ($O(r\log r)$ for the covering set) exists (see [Ma], Sect. 5 for details on this).

The whole procedure takes time $O(nk)$ (preprocessing for distance measurements) plus a time polynomial in mk as claimed, and since we use a constant number of simplification steps only, the cutting factor increases by a constant. □

4 The algorithm

4.1 Recursion in n

In this section we give our first algorithm finding a $(1/r)$-cutting.

Lemma 4.1 *There exists a deterministic algorithm finding a (standard) $(1/r)$-cutting of size $O((r\log r)^d)$ for H in time $O(n(\log n)^A r^D)$ (where A,D are constants dependent on dimension).*

Proof: We use the following recursive algorithm, whose parameters are H (a collection of n hyperplanes) and a number $r \le n$:

Algorithm CUT1

1. (Base case) If $n^d \le nr^D$ (thus $r \ge n^{(d-1)/D}$, so in particular $\log n = O(\log r)$), we compute a $(1/n)$-cutting directly by triangulating the arrangement of H and then we apply the simplification step from Theorem 3.1(iii) to achieve the appropriate size of the cutting. This step has complexity $O(r^{O(1)})$. If the above condition does not hold, we continue by the next step.

2. We choose a number m (which will be 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 $(1/Kr)$-cutting Ξ_i by a recursive application of algorithm CUT1 (K is the constant from Theorem 3.3).

3. We use the algorithm of Theorem 3.3 to compute a $(1/r)$-cutting for H, merging the cuttings Ξ_i.

If we denote by $T_1(n,r)$ the worst-case complexity of algorithm CUT1 applied for n hyperplanes and a parameter r, we get the recurrence

$$\begin{aligned} T_1(n,r) &= O(r^{O(1)}) \qquad \text{for } r \ge n^{D/(d-1)}, \\ T_1(n,r) &\le O(nr^{c_1} + m^{c_2}r^{c_3}) + m.T_1(n/m, Kr) \end{aligned}$$

(K,c_1,c_2,c_3 constants). Choosing $m = n^{1/c_2}$, it is not difficult to verify that this recurrence is satisfied by a function $T_1(n,r)$ of order $O(n(\log n)^A r^D)$ for suitable constants A,D (we may take $D = \max(c_1, c_3)$ and any A with $(1 - 1/c_2)^A > K^D$). □

4.2 Recursion in r

In this section we will 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 3.1. The starting observation is that if r is bounded by a constant, then algorithm CUT1 is already good enough. We will arrange the recursion in such a way that 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, whose input are H and r and whose output is a standard $(1/r)$-cutting of size at most $K(r \log r)^d$ for H (K a certain constant).

Algorithm CUT2

1. If $r < r_0$ (where r_0 is a suitable constant), we use algorithm CUT1 directly; the time complexity of this step is $O(n(\log n)^A)$, which is what we need. If $r \geq r_0$, we continue by the next step.

2. We choose parameters $r_1 = r/2$ and $r_2 = 2K'$, where K' is the constant emerging in the "simplification step" (Theorem 3.1(iii)), thus $r_1 r_2 = K'r$. We use algorithm CUT2 recursively to compute a $(1/r_1)$-cutting Ξ_1 of size $k_1 \leq K(r_1 \log r_1)^d = O((r \log r)^d)$ for H.

3. For every simplex s of Ξ_1 we compute the collection H_s of hyperplanes intersecting its interior (in total time $O(nk_1/r_1) = O(nr^{d-1}(\log r)^d)$).

4. For every s, we use algorithm CUT1 to compute a $(1/r_2)$-cutting Ξ_s of size at most $k_2 = O(1)$ for H_s. The total time needed for this is at most $k_1.O((n/r_1)(\log n)^A r_2^D)$, which is bounded by $O(n(\log n)^A r^{d-1}(\log r)^d)$.

5. We use the method of section 3.1 to compute a $(1/K'r)$-cutting of size $O(k_1 k_2)$ for H.

6. We use the algorithm of Theorem 3.1(iii) to simplify the cutting obtained in the previous step, yielding a $(1/r)$-cutting of size at most $K(r \log r)^d$. In our setting, the size has decreased by a constant factor only, but the proportionality constant is now absolute and does not increase in the recursion. This step requires time $O(nr^{d-1}(\log r)^d + r^{O(1)})$, and the second addend can be neglected compared to the first one (this is where our assumption that r was not too big comes into play).

Let us denote the worst-case running time of algorithm CUT2 by $T_2(n, r)$. We get the following recurrence relations:

$$\begin{aligned} T_2(n,r) &= O(n(\log n)^A) \qquad \text{for } r < r_0, \\ T_2(n,r) &\leq O(n \log^A nr^{d-1} \log^d r) + T_2(n, r/2). \end{aligned}$$

The solution is $T_2(n, r) \leq O(n(\log n)^A r^{d-1}(\log r)^d)$.

Now it remains to remove the restriction $r \leq n^\alpha$, introduced in the above algorithm. Let us consider a general value of r, and choose a value $\bar{r}$, which is a permissible value of r for algorithm CUT2 (i.e. $\bar{r} < n^\alpha$) and such that $r = \bar{r}^p$ for a suitable constant p. We proceed in p steps:

In the first step, we compute a $(1/\bar{r})$-cutting for H by algorithm CUT2.

In the i-th step, we already have a $1/\bar{r}^{i-1}$)-cutting for H. For every simplex s of this cutting, we compute the collection of hyperplanes intersectig the interior of s, we use algorithm CUT2 to compute a $(1/\bar{r})$-cutting for this set of hyperplanes, and we use these cuttings as in section 3.1 to yield a $(1/\bar{r}^i)$-cutting for H.

An easy calculation shows that in this way we get a $(1/r)$-cutting of size $O(r^d(\log r)^{pd})$ for H, and also that the running time of this algorithm does not exceed $O(n(\log n)^A r^{d-1}(\log r)^{pd})$; we omit the details. This proves Theorem 1.1. □

5 Cutting in the plane

In this section we sketch the proof of Theorem 1.2. Our algorithm will be analogous to Algorithm CUT2, but may use a more efficient algorithm for cutting with a constant value of r (replacing the call to Algorithm CUT1), and a more efficient simplification procedure. These subroutines are described in the following two theorems:

Theorem 5.1 [Ma] *Given a collection H of n lines, we can compute a $(1/r)$-cutting of size $O(r^2)$ for H in time $O(nr^2 \log r)$.* □

Lemma 5.2 *Let Ξ_0 be a $(1/r)$-cutting of size k for a collection H of n lines in the plane. Then we can compute a (K/r)-cutting Ξ of size $O(r^2)$ for H, in time $O(nk/r + (rk)^c)$ (K, c constants).*

In our application, the value of k in this simplification lemma will be of order $O(r^2)$ (only with a larger constant than for the simplified cutting). Now if r is small enough (at most $n^{1/3c}$), the term $O(nk/r) = O(nr)$ will dominate the time bound in Lemma 5.2, and we obtain the total running time of the cutting algorithm of order $O(nr)$ by a straightforward analysis. For larger values of r, we may use the same trick as we did with Algorithm CUT2 in the end of Section 4.2.

Let us sketch the proof of Lemma 5.2 First we observe that a general cutting in the plane can be in linear time converted into a cutting which is a triangulation, with size increased only by a constant factor. Hence we may assume that Ξ_0 is a triangulation and so in particular we may compute a description of H relative to Ξ_0 in time $O(nk/r)$.

The basic strategy for the simplification is the following: (we assume that the reader is familiar with [Aga] or [Ma] at this point): Using the original $(1/r)$-cutting Ξ_0, we compute a (n/r)-approximate leveling for H (this procedure is described in [Aga], and it takes time $O(nk/r)$). In time $O(k)$, we then compute a $(3n/r)$-approximate leveling with $O(r^2)$ edges in total, as described in [Ma]. Now we want to use this leveling to obtain a new (K/r)-cutting of complexity $O(r^2)$. Applying the method of [Ma] straightforwardly, we could do it in time $O(nr \log r)$. The factor $\log r$ appears when we need to subdivide potentially "long" quadrilaterals into smaller ones, which may require an approximate sorting of intersections of lines with the sides of such quadrilaterals.

In order to do better, we use the original cutting Ξ_0 again. Namely, we first compute a description of H relative to Ξ_0 and also all intersections of the simplified approximate leveling with all edges of Ξ_0. Now it is easy to see that one suffices with these intersections as the subdividing points on the approximate levels, and using an approximate distance measurement in H, the new cutting Ξ can be computed from the simplified approximate leveling in time depending polynomially on r and k (and not on n). □

6 Conclusion

In this paper we gave deterministic algorithms finding nearly optimal cuttings and ϵ-nets (of the best size guaranteed by known existence proofs) for "geometric" range spaces, all this in (theoretically) reasonable time. There is, of course, much room for improvement of the powers of logarithms in our bounds, which can be achieved e.g. by a more careful implementaion of the operations with cuttings. The most challenging problem in this area now is perhaps to compute cuttings of asymptotically optimal size in reasonable time, maybe adapting the method of [CF].

References

[AHU] A.V.Aho, J.E.Hopcroft, J.D.Ulman: *The design and analysis of computer alqorithms*, Addison-Wesley 1983

[Aga] P.K.Agarwal: A deterministic algorithm for partitioning arrangements of lines and its applications, Proc. 5. Ann. ACM Symposium on Comput. Geometry (1989), pp. 11-21

[Aga1] P.K.Agarwal: Ray shooting and other applications of spanning trees with low stabbing number, Proc. 5. Ann. ACM Symposium on Comput. Geometry (1989), pp. 315-325

[CEG] B.Chazelle, H.Edelsbrunner, L.Guibas: Lines in space: Combinatorics, algorithms and applications, STOC 1989, pp. 389-392

[CF] B.Chazelle, J.Friedman: A deterministic view of random sampling and its use in geometry, Report CS-TR-181-88, extended abstract FOCS 1988, pp.539-549

[Cla] K.L.Clarkson: Applications of random sampling in computational geometry II, Proc. 4th Ann. ACM Symposium on Comput. Geometry (1988), pp. 1-11

[Cla1] K.L.Clarkson: A randomized algorithm for closest-point queries, SIAM J. on Computing 17(1988), pp. 830-847

[C&al] K.Clarkson, H.Edelsbrunner, L.Guibas, M.Sharir, E.Welzl: Combinatorial complexity for arrangements of curves and surfaces, FOCS 1988, pp. 568-579

[Ede] H.Edelsbrunner: *Algorithms in combinatorial geometry*, Springer 1987

[E&al] H.Edelsbrunner, L.Guibas, J.Herschberger, R.Seidel, M.Sharir, J.Snoeyink, E.Welzl: Implicitly representing arrangements of lines or segments, Discr & Comput. Geometry 4(1989), pp. 433-466

[EOS] H.Edelsbrunner, J.O'Rourke, R.Seidel: Constructing arrangements of hyperplanes with applications, SIAM J. on Computing 15(1984) pp. 341-363

[HW] D.Haussler, E.Welzl: ϵ-nets and simplex range queries, Discr. & Comput. Geometry 2(1987) pp. 127-151

[Ma] J.Matoušek: Construction of ϵ-nets, Proc. 5. Ann. ACM Symposium on Comput. Geometry (1989), pp. 1-10

[Ma1] J.Matoušek: Approximate halfplanar range counting, KAM Series 59-87, Charles University 1987

[PW] J.Pach, G. Woeginger: Some new bounds for epsilon-nets, Report B-89-08 Serie B – Informatik, FU Berlin 1989

[YY] F.F.Yao, A.C.Yao: A general approach to geometric queries, Proc. 17th STOC(1985), pp.163-168