# Cuttings and applications

# Cuttings and Applications

Mark de Berg, Otfried Schwarzkopf

RUU-CS-92-26
August 1992


Utrecht University

**Department of Computer Science**

Padualaan 14, P.O. Box 80.089,
3508 TB Utrecht, The Netherlands,
Tel. : ... + 31 - 30 - 531454

# Cuttings and Applications

Mark de Berg, Otfried Schwarzkopf

Technical Report RUU-CS-92-26
August 1992

Department of Computer Science
Utrecht University
P.O.Box 80.089
3508 TB Utrecht
The Netherlands

ISSN: 0924–3275

# Cuttings and Applications*

Mark de Berg Otfried Schwarzkopf
Utrecht University, Department of Computer Science,
P.O. Box 80.089, 3508 TB Utrecht, the Netherlands

**Abstract**

We prove a general lemma on the existence of $(1/r)$-cuttings of geometric objects in $\mathsf{E}^d$ that satisfy certain properties. We use this lemma to construct $(1/r)$-cuttings of (asymptotically) optimal size for arrangements of line segments in the plane and arrangements of triangles in 3-space; for line segments in the plane we obtain a cutting of size $O(r + Ar^2/n^2)$, and for triangles in 3-space our cutting has size $O(r^2\alpha(r) + Ar^3/n^3)$. Here $A$ is the combinatorial complexity of the arrangement. Finally, we use these results to obtain new results for several problems concerning line segments in the plane and triangles in 3-space.

## 1 Introduction

A $(1/r)$-cutting for a set $H$ of hyperplanes in $\mathsf{E}^d$ is partitioning of $\mathsf{E}^d$ into simplices with disjoint interiors, such that each simplex is intersected by at most $n/r$ of the hyperplanes. Cuttings can be used to solve a variety of problems on sets of hyperplanes, using a divide-and-conquer approach. The efficiency of the resulting algorithms and data structures depends heavily on the size of the cutting, that is, its number of simplices. Therefore, much research in the past few years has been devoted to constructing cuttings of small size [2, 5, 7, 8, 9, 16, 17, 18]. These days there are several methods for constructing cuttings of optimal size $O(r^d)$ [5, 7, 18].

The concept of cuttings is readily generalized to sets of other geometric objects than hyperplanes: a $(1/r)$-cutting for a set $H$ of geometric objects in $\mathsf{E}^d$ is a partitioning of $\mathsf{E}^d$ into 'elementary shapes'—which we call boxes—such that each box is intersected by at most $n/r$ objects. (See Section 2 for a more precise definition of a box.) In this paper we prove a general lemma on the existence of cuttings for sets of geometric objects in $\mathsf{E}^d$ that satisfy certain properties. More precisely, we show that a $(1/r)$-cutting of size $O(\tau(r))$ exists, where $\tau(r)$ is the expected number of boxes in a so-called canonical triangulation of a random subset $R \subset H$, where each object in $H$ is drawn with probability $r/n$. (Note that $\varepsilon$-net theory [14] proves the existence of an $O(\log r/r)$-cutting of size $O(\tau(r))$.)

We apply this General Cutting Lemma to construct cuttings of (asymptotically) optimal size for arrangements of line segments in the plane and for arrangements of triangles in 3-space. For line segments this yields a cutting of size $O(r + Ar^2/n^2)$, and for triangles in 3-space our cutting has size $O(r^2\alpha(r) + Ar^3/n^3)$. Here $A$ is the combinatorial complexity of the arrangement.

*This research was supported by the Netherlands' Organization for Scientific Research (NWO).

## 2.1 The General Cutting Lemma

Let $\mathcal{U}$ be a set of geometric *objects* in $\mathsf{E}^d$. A *box* is a closed subset of $\mathsf{E}^d$ which has constant description (that is, it can be represented in a computer with $O(1)$ space, and it can be checked in constant time whether a point lies in a box or whether an object intersects (the interior of) a box). Let $H \subset \mathcal{U}$ with $|H| = n$. For a box $s$, let $H_s$ denote the set of objects in $H$ intersecting the interior of $s$. A $(1/r)$-*cutting* $\Xi(H)$ *for* $H$ is a family of boxes with disjoint interiors that cover $\mathsf{E}^d$ and such that $|H_s| \leqslant n/r$ for all boxes $s \in \Xi(H)$. The *size* of a cutting is the number of its boxes. We prove a result on the existence of $(1/r)$-cuttings of small size under some conditions on the universe $\mathcal{U}$ of objects.

We require that we can define for every $H \subset \mathcal{U}$ a *canonical triangulation* $\mathcal{T}(H)$ of $H$, defined as a set of boxes with disjoint interiors that cover $\mathsf{E}^d$ such that $H_s = \emptyset$ for every $s \in \mathcal{T}(H)$. We need the following properties:

(C1) If $s \in \mathcal{T}(H)$ then there exists a unique inclusion-minimal subset $S(s) \subset H$ with $s \in \mathcal{T}(S(s))$. The cardinality $|S(s)|$ is bounded by a constant $D$.

(C2) For $s \in \mathcal{T}(H)$, there is a unique set $K_s$ such that for any $R \supset H$, $s$ is in $\mathcal{T}(R)$ exactly if $K_s \cap R = \emptyset$.

(C3) For a set $H$ of objects and a parameter $t \geqslant 1$, there exists a $(1/t)$-cutting for $H$ of size $O(t^C)$, for some constant $C$. (This condition is easy to fulfill in general, since $C$ can be chosen arbitrarily large).

Let a set $H$ of $n$ objects and a parameter $r \leqslant n/2$ be chosen. For a box $s$, let the *excess* $t_s$ be the number $\lceil |H_s|(r/n) \rceil$. Furthermore, let $N(p,t)$ be the expected number of boxes with excess at least $t$ in the canonical triangulation $\mathcal{T}(R)$ of a random sample $R \subset H$, where each object in $H$ is drawn independently with probability $p = r/n$. The following lemma essentially follows from results by Chazelle and Friedman [7]. A short proof is given by Matoušek [15].

**Lemma 2.1** $N(p,t) = O(2^{-t}) \cdot N(p/t, 0)$.

(Actually, [7] and [15] define the excess in terms of $|K_s|$ instead of $|H_s|$. Observing that $H_s \subset K_s$, it is not difficult to verify that the lemma as stated here follows from their result.) The following algorithm computes a $(1/r)$-cutting for a given set $H$. It is based on Lemma 2.1 and follows the lines of [7] and [15].

**Algorithm 2.1**
{ Computes a $(1/r)$-cutting $\Xi(H)$ for a set $H$ of $n$ objects. }
1. Take a sample $R \subset H$ by drawing every object in $H$ with probability $r/n$.
2. Construct the canonical triangulation $\mathcal{T}(R)$.
3. **for** each box $s \in \mathcal{T}(R)$
4. **do** Compute the set $H_s$ of objects in $H$ intersecting $s$.
5. Compute a $(1/t)$-cutting $\Xi_s$ for $H_s$ of size $O(t^C)$, where $t = t_s$ is the excess of $s$.
6. Let $\Xi(H) = \{s' \cap s : s \in \mathcal{T}(R), s' \in \Xi_s\}$.

**Lemma 2.2 [General Cutting Lemma]** *Let $H$ be a set of $n$ objects in $\mathsf{E}^d$ and $r \leqslant n$ a parameter. There exists a $(1/r)$-cutting for $H$, consisting of $O(\tau(r))$ boxes, where $\tau(r)$ is the expected number of boxes in the canonical triangulation of a random subset $R \subset H$, where every object in $H$ is drawn with probability $r/n$.*

**Proof:** Consider Algorithm 2.1. It is easy to check that this algorithm produces a $(1/r)$-cutting for $H$. Next we prove that the expected size of the cutting is $O(\tau(r))$. This amounts to bounding the expected value $S$ of the sum $\sum_{s\in\mathcal{T}(R)} t_s^C$. We can (crudely) estimate $S$ by

$$\sum_{t=1}^{\infty} N(p,t)t^C,$$

and using Lemma 2.1 we can bound this by

$$\sum_{t=1}^{\infty} 2^{-t}t^C O(N(p/t,0)).$$

Since $N(p/t,0)$ is the expected number of boxes in $\mathcal{T}(R')$ where $R'$ is obtained by randomly drawing every object in $H$ with probability $p/t = r/nt$, we can bound $N(p/t,0)$ by $\tau(r/t) \leqslant \tau(r)$. We thus get

$$S \leqslant \sum_{t=1}^{\infty} 2^{-t}t^C O(\tau(r)) \leqslant O(\tau(r)).$$

Note that the algorithm is not guaranteed to give us a cutting of size $O(\tau(r))$, but that this is only the expected size of the cutting. So what we should do is try a number of samples until we have found a good one. The expected number of trials is constant. Note that the time taken by the algorithm depends on the size of the cutting that is produced. But as soon as we spend too much time—that is, more time than is predicted by the expected size of the cutting—we may stop the algorithm and try the next sample. □

## 2.2 Cuttings for Segments in the Plane

In this subsection we use the general result on cuttings stated above to prove a theorem on cuttings for arrangements of line segments in the plane.

**Theorem 2.1** *Let $H$ be a set of $n$ line segments in the plane with a total of $A$ intersections. It is possible to construct a $(1/r)$-cutting $\Xi(H)$ for $H$ of size $O(r + Ar^2/n^2)$ in randomized time $O(n \log r + Ar/n)$.*

**Proof:** To be able to apply the results of the previous subsection, we have to define a canonical triangulation $\mathcal{T}(H)$ for a set $H$ of line segments in the plane. We take $\mathcal{T}(H)$ to be the trapezoidal decomposition of the arrangement $\mathcal{A}(H)$. This decomposition is obtained by adding a vertical segment through every vertex of $\mathcal{A}(H)$ (this can be an endpoint of a segment in $H$ or an intersection between two segments) of maximal length that does not cross any segment in $H$. The resulting planar map consists of (possibly degenerated) trapezoids with disjoint interiors that cover the plane. One easily checks that the decomposition satisfies conditions (C1) and (C2). (In this case the set $K_s$ is identical to the set $H_s$ of segments intersecting the interior of $s$.) Also condition (C3) can be fulfilled, because a $(1/t)$-cutting for the set of lines through the segments—which is obviously a $(1/t)$-cutting for the segments—of size $O(t^2)$ exists. Such a cutting can be constructed in time $O(nt)$, see [5]. (Actually, a much simpler algorithm can be used, since any polynomial sized cutting is sufficient).

Next we bound the size of $\mathcal{T}(R)$ for a random subset $R \subset H$ that is obtained by picking every element in $H$ with probability $p = r/n$. Clearly, the size of $\mathcal{T}(R)$ is linear in the complexity of $\mathcal{A}(R)$. The probability that a certain intersection between two segments in $H$ is present in $\mathcal{A}(R)$ is $r^2/n^2$. Hence, the expected number of trapezoids in $\mathcal{T}(R)$ is $O(r + Ar^2/n^2)$. We may conclude from the General Cutting Lemma that Algorithm 2.1 constructs a $(1/r)$-cutting for $H$ whose expected size is $O(r + Ar^2/n^2)$.

It remains to prove the bound on the preprocessing time. The trapezoidal decomposition $\mathcal{T}(R)$ can be constructed in time $O(r \log r + |\mathcal{T}(R)|)$, see [6]. Next we have to compute the sets $H_s$. We do this for all trapezoids $s$ simultaneously, as follows. We build a point location structure for $\mathcal{T}(R)$. For each segment $e \in H - R$, we locate one endpoint in $\mathcal{T}(R)$ and we traverse $\mathcal{T}(R)$. We can step from one trapezoid intersected by $e$ to the next by considering all adjacent trapezoids, similar to the incremental arrangement construction algorithm [11].

The time needed to trace one line segment $e$ in this fashion can be bounded by considering $\mathcal{T}(R \cup \{e\})$. The intersection points cut the segments of $R \cup \{e\}$ into pieces, and the time for $e$ is linear in the number of trapezoids in $\mathcal{T}(R)$ adjacent to a piece of $e$, or to a piece of a segment in $R$ which is in turn incident to $e$. This allows the application of Seidel's backwards analysis [21], and we can prove that the expected time to trace all the segments in $H - R$ through $\mathcal{T}(R)$ is bounded by $O(n \log r + Ar/n)$.

Finally, we compute for each trapezoid $s$ a $(1/t)$-cutting $\Xi_s$, where $t = t_s$ is the excess of $s$, and we intersect the resulting boxes in $\Xi_s$ with $s$. As mentioned above, we can compute such a cutting of size $O(t^2)$ in time $O(|H_s|t)$. The total time for this is bounded by

$$\sum_s |H_s| t_s \leqslant \sum_s (n/r) t_s^2 \leqslant (n/r) \sum_{t=1}^{\infty} t^2 N(p,t)$$

Following the analysis given in the proof of the General Cutting Lemma, we see that this sum is bounded by $O((n/r)\tau(r))$, where $\tau(r) = O(r + Ar^2/n^2)$ is the expected size of $\mathcal{T}(R)$. The time bound follows. □

## 2.3 Cuttings for Triangles in 3-Space

As a second application of the general cutting lemma, we consider cuttings for arrangements of triangles in 3-space. We denote the arrangement of 3-space induced by a set $H$ of triangles by $\mathcal{A}(H)$. The combinatorial complexity of $\mathcal{A}(H)$ is the total number of faces of various dimension (vertices, edges, facets and cells). The complexity of a subset of the cells is defined analogously. The cutting is based on the so-called Slicing Theorem, proved by Aronov and Sharir [4].

**Theorem 2.2 (Slicing Theorem [4])** *Let $K$ be a collection of cells in an arrangement of $n$ triangles in 3-space, with total complexity $A$. Then $K$ can be triangulated using $O(n^2\alpha(n) + A)$ tetrahedra.*

This theorem implies the following:

**Theorem 2.3** *Let $H$ be a set of triangles in three-dimensional space, and let $A$ denote the complexity of the arrangement they induce. There exists a $(1/r)$-cutting for $H$ of size $O(r^2\alpha(r) + Ar^3/n^3)$.*

**Proof:** Let $R \subset H$. It is fairly easy to check that applying the Slicing Theorem to the collection of all cells in $\mathcal{A}(R)$ produces a canonical triangulation satisfying conditions (C1)–(C2). There is one small subtlety, though: The decomposition produced by the Slicing Theorem depends on the order in which the triangles are treated. However, by fixing an (arbitrary) order on the triangles in $H$ beforehand, the canonical triangulation can be made unique. Note that the set $K_s$ can now be strictly larger than the set $H_s$ of triangles intersecting a box (simplex of the slicing). Also (C3) is easily fulfilled: we just take a $(1/r)$-cutting of size $O(r^3)$ for the set of planes containing the triangles, which can be computed in $O(nr^2)$ time [5].

We are thus in the position to apply the General Cutting Lemma. So let us bound the expected number of tetrahedra if the Slicing Theorem is applied to a random subset $R \subset H$. Let $V_H$ be the set of vertices in $\mathcal{A}(H)$ that are the intersection of three triangles in $H$. Similarly, $V_R$ is the set of vertices in $\mathcal{A}(R)$ that are the intersection of three triangles in $R$. Observe that the complexity of $\mathcal{A}(R)$ is $O(r^2 + V_R)$. Recall that the random subset $R \subset H$ is obtained by picking every triangle in $H$ with probability $r/n$. Hence, the expected size of $V_R$ is $O(|V_H|(r/n)^3))$. Since $|V_H| \leqslant A$, a random sample $R$ induces an arrangement of expected size $O(r^2 + Ar^3/n^3)$. The Slicing Theorem thus gives us a canonical triangulation of the desired size. □

**Remark 2.1** Though it is not difficult to give a polynomial time construction algorithm for Theorem 2.3, we do not know how to construct the cutting in time close to $O(nr\alpha(r) + Ar^2/n^2)$, which is the total size of all sets $H_s$. The main problem is to give an efficient algorithm for the Slicing Theorem. In our applications, however, $r$ will be constant and a brute-force construction method suffices.

# 3 Applications in the Plane

## 3.1 Counting Intersections

In the previous section we have seen that it is possible to construct a $(1/r)$-cutting for a set $H$ of line segments whose size depends on $A$, the complexity of $\mathcal{A}(H)$. In some cases it may thus be advantageous to choose $r$ dependent upon $A$. But then we need to know $A$, of course. Note that $A = O(n + I)$, where $I$ is the number of intersections in $H$. The fastest algorithm to count the number of intersections in a set $H$ of $n$ segments in the plane runs in time $O(n^{4/3}\log^{1/3} n)$, see Chazelle [5]; we will present an algorithm that counts the number of intersections faster when this number is $o(n^2)$.

**Theorem 3.1** *Let $H$ be a set of $n$ line segments in the plane. The number of intersections in $H$ can be counted in $O(n\log n + A^{1/3}n^{2/3}\log^{1/3} n)$ randomized time, where $A$ is the complexity of $\mathcal{A}(H)$.*

**Proof:** According to Theorem 2.1, we can construct a $(1/r)$-cutting for $H$ of size $O(r + Ar^2/n^2)$ in randomized time $O(n\log r(1 + Ar/n^2))$. We would like to construct such a cutting for $r = n^2/(n + A)$, thus obtaining a cutting of size $O(r) = O(n^2/(n + A))$ such that each triangle in the cutting is intersected by at most $n/r = (n + A)/n$ segments. Inside each triangle of the cutting, we then count the number of intersections in time

be bounded as follows

$$\begin{aligned}
&\sum_s \{\, (|H_s| + |P_s|)\log(|H_s| + |P_s|) \;+\; |H_s|^{2/3}|P_s|^{2/3}2^{O(\log^*(|H_s|+|P_s|))} \,\} \\
&\quad\leqslant\; (n+m)\log(n+m) \;+\; ((n+A)/n)^{2/3}2^{O(\log^*(n+m))}\sum_s |P_s|^{2/3} \\
&\quad\leqslant\; (n+m)\log(n+m) \;+\; ((n+A)/n)^{2/3}2^{O(\log^*(n+m))}[(n^2/(n+A))((m+A)/n)^{2/3}] \\
&\quad\leqslant\; (n+m)\log(n+m) \;+\; A^{1/3}m^{2/3}2^{O(\log^*(n+m))}
\end{aligned}$$

□

# 4 Point Location in 3-Space

In this section we study the following point location problem: store a set $H$ of $n$ triangles in 3-space such that one can determine efficiently the cell of $\mathcal{A}(H)$ that contains a query point.

## 4.1 The structure

Our point location method for arrangements of triangles is almost the same as the 'standard method' for point location in arrangements of hyperplanes. We compute a cutting of constant size, which we associate with the root of a tree $\mathcal{T}$. Each tetrahedron $s$ in the cutting is associated with a unique child $\nu_s$ of the root of $\mathcal{T}$. Node $\nu_s$ is the root of a recursively defined structure on the set $H(\nu_s)$ of triangles that intersect $s$. When the number of triangles at some node $\mu$ in $\mathcal{T}$ drops below some constant—we call $\mu$ a *small node*—we proceed as follows. Let $s_\mu$ be the tetrahedron corresponding to node $\mu$, and let $F(s_\mu)$ be the set of facets of $s_\mu$. We construct the cells of the arrangement $\mathcal{A}(H(\mu)\cup F(s_\mu))$ that lie inside $s_\mu$ explicitly. Every cell is then stored in a separate leaf, which becomes a child of $\mu$. In other words, a leaf which is a child of $\mu$ corresponds to the region of points inside $s_\mu$ which can be connected by a path that stays inside $s_\mu$ and does not intersect any triangle.

To locate a point $q$ in the arrangement $\mathcal{A}(H)$, we we can now proceed in a a standard manner. We start in the root, and whenever we are at a node $\nu$, we continue in the child $\mu$ of $\nu$ where the tetrahedron associated with $\mu$ in the cutting of $\nu$ contains $q$. In this fashion, we can walk down $\mathcal{T}$ in $O(\log n)$ time.

The following lemma proves the correctness of our point location scheme.

**Lemma 4.1** *The cell of $\mathcal{A}(H)$ that contains a query point $q$ is uniquely determined by the leaf where the search path of $q$ in $\mathcal{T}$ ends.*

**Proof:** From the way $\mathcal{T}$ is defined it follows that a triangle in $H$ intersects the tetrahedron $s_\mu$ if and only if the triangle is in $H(\mu)$. But this means that if we can connect two points with a path inside $s_\mu$ that does not intersect any triangle in $H(\mu)$, then the path does not intersect any triangle in $H$. Hence, two points whose search paths end in the same leaf must be in the same cell of $\mathcal{A}(H)$. □

Note that the reverse is not true: For two points lying in the same cell of $\mathcal{A}(H)$, we

might wind up in two different leaves of $\mathcal{T}$. This implies that in order to use the above structure for point location, we still have to associate leaves in $\mathcal{T}$ with the corresponding cells in $\mathcal{A}(H)$. Below we show how to do this, and we analyze the amount of storage and preprocessing time that we need. Anticipating these results, we state the main theorem of this section.

**Theorem 4.1** *For any $\varepsilon > 0$, there exists a structure for point location in an arrangement of $n$ triangles in 3-space with total complexity $A$ whose query time is $O(\log n)$ and that uses $O(n^{2+\varepsilon} + A^{1+\varepsilon})$ storage. The structure can be built in $O(n^{2+\varepsilon} + A^{1+\varepsilon})$ randomized time.*

## 4.2 Preprocessing

The preprocessing of the structure consists of two tasks: the tree $\mathcal{T}$ has to be built, and we have to associate the leaves of $\mathcal{T}$ with the corresponding cells of $\mathcal{A}(H)$. The first task is a fairly standard exercise, so let us concentrate on the second task, which is to associate leaves with cells. Thus we have to determine which leaves of $\mathcal{T}$ correspond to the same cell. We will do this in a bottom-up manner. Each leaf in the tree will be an element in a Union-Find structure. The initial sets in the Union-Find structure are the singleton sets corresponding to these elements. At the end of the algorithm the sets will correspond to the cells in $\mathcal{A}(H)$. To this end we perform certain unions as we work our way up the tree. These unions are such that the following invariant is maintained: when we have handled a node $\nu$ then the leaves in the subtree $\mathcal{T}_\nu$ rooted at $\nu$ are in the same set if and only if they correspond to the same cell of $\mathcal{A}(H(\nu) \cup F(s_\nu))$. This means that we are ready when we have restored the invariant at the root of $\mathcal{T}$, where we have $s_{root} = \mathbb{E}^3$.

The invariant is trivially true at the small nodes of the structure, so now consider a non-small node $\nu$ in $\mathcal{T}$. Every child $\mu$ of $\nu$ corresponds to a certain tetrahedron $s_\mu$ in the cutting for $H(\nu)$. Consider the part of $\mathcal{A}(H)$ inside $s_\mu$. There are cells of $\mathcal{A}(H)$ that are fully contained in $s_\mu$ and cells that intersect the boundary of $s_\mu$. The first type of cells is of no concern any more at this stage of the algorithm: all the leaves that correspond to such cells are in the subtree rooted at $\mu$, and therefore they are already in the same set. The latter type of cells need further treatment, since they induce leaves not only in the subtree rooted at $\mu$ but also in other subtrees. In other words, we have to union the set that contains the leaves in $\mathcal{T}_\mu$ corresponding to such a cell with the sets corresponding to the same cell that contain the leaves in other subtrees $\mathcal{T}_{\mu'}$. This is done with the following straightforward procedure.

1. Let $M$ be the set of tetrahedra corresponding to the children of $\nu$. Compute all pairs of facets of tetrahedra in $M$ that intersect.

2. Let $f, f'$ be a pair of intersecting facets, and let $\mu$ and $\mu'$ be the children of $\nu$ that correspond to the tetrahedra with facets $f, f'$.
   (i) Compute the arrangement $\mathcal{A}(H(\nu) \cap f \cap f')$.
   (ii) For each region $g$ of $\mathcal{A}(H(\nu) \cap f \cap f')$ that is not contained in a triangle of $H(\nu)$, do the following: Search with a point $p \in g$ in $\mathcal{T}_\mu$ and $\mathcal{T}_{\mu'}$. Let $\gamma, \gamma'$ be the leaves where the search paths end. Compute the union of the sets containing $\gamma$ and $\gamma'$ by performing the following operation on the Union-Find-structure: UNION( FIND($\gamma$), FIND($\gamma'$) ).

It is not difficult to prove that this procedure correctly collects all leaves of $\mathcal{T}$ belonging to the same cell.

## 4.3 Analysis

Above we have described our point location data structure and shown how to build it. Next, we analyze the amount of storage and the preprocessing time. Let us start with the amount of storage.

By Theorem 2.3, we can construct a $(1/r)$-cutting of size $O(r^2\alpha(r) + Ar^3/n^3)$. Hence, if $A \leqslant n^3/r$, then we obtain a cutting of size $O(r^2\alpha(r))$. Let $n_\nu$ denote the size of the set $H(\nu)$ of triangles that are considered at node $\nu$, and let $A_\nu$ denote the complexity of $\mathcal{A}(H(\nu))$. By the above argument, we can construct a cutting of size $O(r^2\alpha(r))$ if $A_\nu \leqslant n_\nu^3/r$. Otherwise, we obtain a cutting of size $O(r^3)$; in that case it follows in a standard manner that we can achieve, for any $\varepsilon > 0$, a total amount of storage of $O(n_\nu^{3+\varepsilon})$. Let $\varepsilon > 0$. If $M_\nu$ denotes the amount of storage used by the subtree rooted at node $\nu$, then there exists a constant $c$ such that

$$M_\nu \leqslant \begin{cases} cr^2\alpha(r) + \sum_{\mu=child(\nu)} M_\mu & \text{if } A_\nu \leqslant n_\nu^3/r \\ cn_\nu^{3+\varepsilon} & \text{otherwise} \end{cases}$$

Note that at each child $\mu$ of a node $\nu$ we have $n_\mu \leqslant n_\nu/r$. Furthermore, we know that $\sum_{\mu=child(\nu)} A_\mu \leqslant A_\nu$. Using the above facts, we will show by induction that $M_\nu \leqslant dn_\nu^{2+\varepsilon} + A_\nu^{1+\varepsilon}$ for some constant $d$:

- case (i): $A_\nu \leqslant n_\nu^3/r$

$$\begin{aligned} M_\nu &\leqslant cr^2\alpha(r) + \textstyle\sum_{\mu=child(\nu)} M_\mu \\ &\leqslant cr^2\alpha(r) + \textstyle\sum_{\mu=child(\nu)} [dn_\mu^{2+\varepsilon} + A_\mu^{1+\varepsilon}] \\ &\leqslant cr^2\alpha(r) + \textstyle\sum_{\mu=child(\nu)} d[n_\nu/r]^{2+\varepsilon} + \sum_{\mu=child(\nu)} A_\mu^{1+\varepsilon} \\ &\leqslant cr^2\alpha(r) + cr^2\alpha(r)d[n_\nu/r]^{2+\varepsilon} + A_\nu^{1+\varepsilon} \\ &\leqslant cr^2\alpha(r) + dn_\nu^{2+\varepsilon}[\tfrac{c\alpha(r)}{r^\varepsilon}] + A_\nu^{1+\varepsilon} \\ &\leqslant dn_\nu^{2+\varepsilon} + A_\nu^{1+\varepsilon} \qquad \text{for } r \text{ and } n_\nu \text{ sufficiently large} \end{aligned}$$

- case (ii): $A_\nu \geqslant n_\nu^3/r$

$$\begin{aligned} M_\nu &\leqslant cn_\nu^{3+\varepsilon} \\ &\leqslant cn_\nu^{3+\varepsilon}[\tfrac{n_\nu^{2\varepsilon}}{cr^{1+\varepsilon}}] \\ &\leqslant A_\nu^{1+\varepsilon} \qquad \text{for } n_\nu \text{ sufficiently large} \end{aligned}$$

Note that the above calculation holds for any fixed $\varepsilon > 0$, provided we choose $r$ appropriately.

Finally, let us analyze the time taken by the preprocessing procedure. Recall that preprocessing was done in two stages: first, the tree was built and then the leaves were grouped into sets that correspond to the same cell of the arrangement. The time that we need at a node $\nu$ during the first stage is dominated by the time needed to construct

a cutting of constant size $r$; this is linear in $\nu$ (the constant of linearity clearly depends on $r$). As for the second stage, we note that the time needed for one pair of tetrahedra is bounded by $O(n_\nu^2 \log n)$, with a constant of proportionality depending on $r$. Thus the time spent at node $\nu$ is $O(n_\nu)$. Using this fact one can prove that the preprocessing time is $O(n^{2+\epsilon} + A^{1+\epsilon})$. (In fact, the above calculation holds almost verbatim.)

# 5 Concluding Remarks

We have proved a general lemma on the existence of cuttings of small size. We applied this General Cutting Lemma to obtain cuttings for arrangements of line segments in the plane and arrangements of triangles in 3-space of (asymptotically) optimal size that depends on the complexity of the arrangement. It seems that cuttings provide a useful tool for solving problems on such arrangements. As an example, we applied our results to obtain new solutions for intersection counting and a variant of Hopcroft's problem in the plane, and for point location in 3-space.

Our algorithms for computing these cuttings are based on random sampling. While they can all be made deterministic using Chazelle and Friedman's conformal samples [7], it would be interesting to find more efficient deterministic algorithms for computing these cuttings. Another direction for further research is to find more applications of our General Cutting Lemma.

# References

[1] P.K. Agarwal, Ray Shooting and Other Applications of Spanning Trees with Low Stabbing Number, *Proc. 5th ACM Symp. on Computational Geometry*, 1989, pp. 315–325.

[2] P.K. Agarwal, Partitioning Arrangements of Lines I: An Efficient Deterministic Algorithm, *Discr. & Computational Geometry* **5** (1990), pp. 449–483.

[3] P.K. Agarwal, Partitioning Arrangements of Lines II: Applications, *Discr. & Computational Geometry* **5** (1990), pp. 533–573.

[4] B. Aronov and M. Sharir, Triangles in Space, or Building (and Analyzing) Castles in the Air, *Combinatorica* **10** (1990), pp. 137–173.

[5] B. Chazelle, An Optimal Convex Hull Algorithm and New Results on Cuttings, *Proc. 32nd IEEE Symp. on Foundations of Computer Science*, 1991, pp. 29–38.

[6] B. Chazelle and H. Edelsbrunner, An Optimal Algorithm for Intersecting Line Segments in the Plane, *J. ACM* **39** (1992), pp. 1–54.

[7] B. Chazelle and J. Friedman, A Deterministic View of Random Sampling and its Use in Computational Geometry, *Combinatorica* **10** (1990), pp. 229–249.

[8] K. Clarkson, New Applications of Random Sampling in Computational Geometry, *Discr. & Computational Geometry* **2** (1987), pp. 195–222.