Discrete Comput Geom 2:99-111 (1987)

Discrete & Computational Geometry

Triangulating Point Sets in Space

David Avis[1]* and Hossam ElGindy[2]

[1] School of Computer Science, McGill University, 805 Sherbrooke St. W., Montreal, Canada, H3A 2K6

[2] Department of Computer and Information Science, University of Pennsylvania, Philadelphia, PA 19104, USA

Communicated by David Dobkin

Abstract. A set P of n points in R^d is called simplicial if it has dimension d and contains exactly $d+1$ extreme points. We show that when P contains n' interior points, there is always one point, called a splitter, that partitions P into $d+1$ simplices, none of which contain more than $dn'/(d+1)$ points. A splitter can be found in $O(d^4+nd^2)$ time. Using this result, we give an $O(nd^4 \log_{1+1/d} n)$ algorithm for triangulating simplicial point sets that are in general position. In R^3 we give an $O(n \log n+k)$ algorithm for triangulating arbitrary point sets, where k is the number of simplices produced. We exhibit sets of $2n+1$ points in R^3 for which the number of simplices produced may vary between $(n-1)^2+1$ and $2n-2$. We also exhibit point sets for which every triangulation contains a quadratic number of simplices.

1. Introduction

Unless otherwise stated we let P denote a set of n points in R^d which has dimension d. P is in *general position* if each subset of P containing $d+1$ points has dimension d. A *simplex* is a set of $d+1$ points in general position. A *triangulation* of P is a partition of the interior of the convex hull of P into simplices, the vertices of which are points of P. We say that P is *simplicial* if P has exactly $d+1$ extreme points. Two extreme edges of P are *disjoint* if they have no common endpoints. For terms not defined here the reader is referred to the book by Grünbaum [3].

* Research supported by the Natural Science and Engineering Research Council grant A3013 and the F.C.A.R. grant EQ1678.

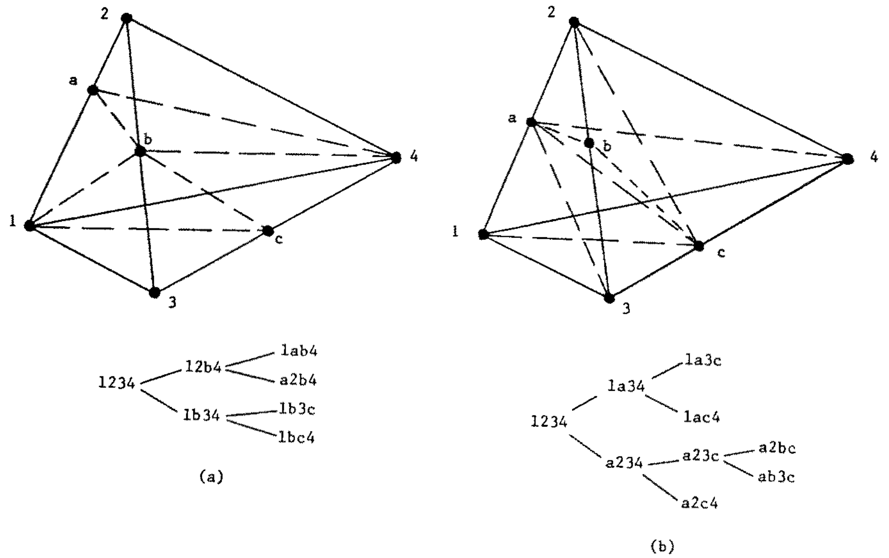


Fig. 1

Triangulations of planar point sets are well behaved and well understood. For example, every triangulation of a given planar point set of n points has the same number of triangles, and this number depends only on the number of points on the convex hull. Furthermore, such a point set can be triangulated in $O(n \log n)$ time and this is optimal [6]. The preceding reference gives a concise survey of various kinds of planar triangulations.

The triangulation problem in higher dimensions has been much less studied. Even in three dimensions, the situation becomes much more complicated. As shown in Fig. 1(a) and (b) the same set of points may have triangulations using a different number of simplices. In fact, as will be shown in the next section, the same set of points P may have triangulations in which the number of simplices varies from a linear to quadratic number in the size of P. The example can be modified so that the points are in general position with the same property. One way of triangulating a three-dimensional set is by using the space sweep technique [6]. Roughly, the idea is to sweep through the points joining each new point to all "visible" points "below" it. Even when the points are in general position, this may give a quadratic number of simplices. In case no more than a constant number of the points are collinear, it will be shown that P can be triangulated with a linear number of simplices. We will present an efficient algorithm that achieves this, and which generalizes to higher dimensions. When the points are not in general position, however, it may be that a quadratic number of simplices are required, as illustrated by the point set in Fig. 4.

The algorithms to be presented in this paper are based on the following geometric fact that will be proved in Section 2:

Every simplicial set P of n points in d dimensions with $n'>0$ interior points can be partitioned into $d+1$ simplices, none of which contains more than $dn'/(d+1)$ points in its interior. The partitioning point is contained in P and is called a $d/(d+1)$-*splitter*. It can be found in $O(d^4+nd^2)$ time.

Using the above fact, Section 3 describes an algorithm for triangulating three-dimensional simplicial point sets in $O(n \log n+k)$ time, where k is the number of simplices produced. In case the vertices are in general position, the algorithm is particularly simple and generalizes to all dimensions. In this case it is shown that $k=O(n)$. In R^3 it is shown that $k=O(n)$ even under the weaker assumption that no more than a constant number of points are collinear (coplanar points are allowed). Finally, in three dimensions, we show that the assumption that the point sets are simplicial can be dropped and give an algorithm for triangulating arbitrary point sets in the same time bound.

2. Geometric Results

Unless stated otherwise, we assume that P is a simplicial d-dimensional n point set in R^d with n' interior points. Let $\{p_1, \ldots, p_{d+1}\}$ be the vertices of P. Let x be any vertex in the interior of P. Then P can be partitioned into $d+1$ simplices

$$S_i=\{p_1, \ldots, p_{i-1}, x, p_{i+1}, \ldots, p_{d+1}\}$$

for $i=1, \ldots, d+1$. Let $f(d)$ be a function defined on the integers. We call x an $f(d)$-*splitter* if each simplex S_i contains at most $f(d)n'$ points of P in its interior. Figure 2 shows a planar point set and a 2/3-splitter. We call a splitter x *optimal* if it minimizes the maximum number of points of P contained in the interior of any of its new simplices. It is easy to verify that the splitter shown in Fig. 2 is optimal. We will prove the following theorem.

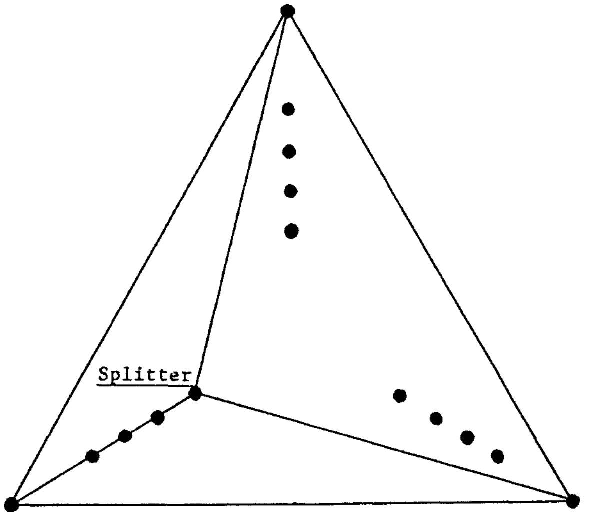


Fig. 2. $(2n/3)$-splitter is best possible!

Theorem 2.1. *Let P be an n point simplicial set in R^d with $n'>0$ interior points. Then P contains a $d/(d+1)$-splitter which can be found in $O(d^4+nd^2)$ time.*

Proof. Let P^I denote the n' interior points of P. The proof involves successively deleting points of P^I that are "close" to the vertices of P. Any of the remaining points can act as a splitter for P.

For each $i=1,\dots,d+1$, let h_i be the unit normal of the facet of P not containing vertex p_i, pointing away from p_i. That is, h_i points to the half-space, bounded by this facet, that does not contain p_i. Set $Q_0=P^I$. We formalize the deletion procedure mentioned above as follows. For each $i=1,\dots,d+1$:

(a) For each $x\in Q_{i-1}$ let $s_x=(x-p_i)h_i$. Observe that s_x is the distance of x from p_i in the direction h_i.
(b) Let y_i be the $\lceil n'/(d+1)\rceil$st order statistic in the ordering induced by h_i.
(c) Set

$$P_i=\{x\in Q_{i-1}\colon s_x<s_{y_i}\},$$

$$\bar{P}_i=\{x\in Q_{i-1}\colon s_x\le s_{y_i}\},$$

$$Q_i=\{x\in Q_{i-1}\colon s_x\ge s_{y_i}\}.$$

We first show that Q_{d+1} is nonempty. By construction, for $i=1,\dots,d+1$,

$$|P_i|<\frac{n'}{d+1},$$

and so

$$\left|\bigcup_{i=1}^{d+1} P_i\right|<n'.$$

Therefore Q_{d+1} is nonempty. Also by construction we have

$$|\bar{P}_i|\ge\frac{n'}{d+1}.$$

We claim that any z contained in Q_{d+1} is a $d/(d+1)$-splitter for P. Indeed, for $i=1,\dots,d+1$, consider the simplices

$$S_i=\{p_1,\dots,p_{i-1},z,p_{i+1},\dots,p_{d+1}\},$$

created by the splitter z. Draw a hyperplane H_i with normal h_i through z. Let H_i^+ be the closed half-space bounded by H_i that contains p_i, and let H_i^- be the opposite open half-space containing the interior of S_i. We have by construction that

$$\bar{P}_i\subset H_i^+,$$

since

$$z\in Q_{d+1}\subseteq Q_i,$$

and this implies that z is at least as far from p_i in the direction h_i as y_i. We may now conclude that

$$|H_i^+ \cap P^I| \geq |H_i^+ \cap \bar{P}_i| = |\bar{P}_i| \geq \frac{n'}{d+1}.$$

Therefore the number of interior points of P contained in the simplex S_i is bounded above by

$$|H_i^- \cap P^I| \leq n' - \frac{n'}{d+1} = \frac{dn'}{d+1}.$$

It remains to show that the time bound can be achieved. Finding a splitter for P involves for each $i = 1, \ldots, d+1$:

(a) Finding a normal vector h_i pointing away from p_i.
(b) Finding the $\lceil n'/(d+1) \rceil$st order statistic in the direction h_i from p_i.
(c) Constructing the sets P_i, $\bar{P}_i$, and Q_i.

Consider some i in the range $1, \ldots, d$. The normal required in step (a) can be calculated $O(d^3)$ time. For step (b), consider any point x in P. Its distance from p_i in the direction h_i is given by the dot product $(x - p_i)h_i$. This requires $O(d)$ operations per point. Finding a one-dimensional order statistic requires $O(n)$ time. Therefore this step takes $O(nd)$ time for each i. In step (c) each of the sets can be constructed in $O(nd)$ time if they are constructed explicitly, or in $O(n)$ time using pointers. The total time complexity of the procedure is thus seen to be $O(d^4 + nd^2)$. □

We gave a different proof of the existence of a $d/(d+1)$-splitter in [1]. This proof was based on an induction on d. A suitably chosen subset of points are recursively projected onto a face of the simplex of one lower dimension. The basic for the recursion is $d = 1$, and at this point the algorithm returns the median of the one-dimensional set. Because of the need to project points, the algorithm has complexity $O(nd^4)$, and so the procedure given above is preferred.

Corollary 2.1. *If P is also in general position, P can be triangulated into at most $n'd+1$ simplices in $O(nd^4 \log_{1+1/d} n)$ time.*

Proof. The algorithm is as follows:

(a) Find a $d/(d+1)$-splitter for P and sets P_i, $i = 1, \ldots, d+1$, as described in the proof of Theorem 2.1.
(b) Recursively apply (a) to each P_i, $i = 1, \ldots, d+1$, that is nonempty.

It follows from Theorem 1 that the recursion can have depth at most $\log_{1+1/d} n$. It thus suffices to bound the total amount of work done at any level in the recursion by $O(nd^4)$. Indeed, suppose at some level there are t nonempty simplices, with, respectively, $m_1, \ldots, m_t$ vertices in their interiors. Then the total amount of work at this level is

$$\sum_{i=0}^{t} (d^4 + m_i d^2) \le td^4 + d^2 n.$$

The result follows since t is bounded above by n. □

Theorem 2.2. *For any integer t, there exist simplicial sets P with $n' = t(d+1)$ interior points in R^d for which the optimal splitters is a $d/(d+1)$-splitter.*

Proof. We generalize the example of Fig. 2. Let $d \ge 2$ be fixed. For convenience in notation, we in fact construct an example in R^{d+1} that has dimension d. Indeed, for $i = 1, \ldots, d+1$, let $p_i = (x_1, \ldots, x_{d+1})$ where $x_i = d+1$ and $x_j = 0$ whenever j is different from i. Then $S = \{p_1, \ldots, p_{d+1}\}$ is a d-dimensional simplex containing the point $c = (1, \ldots, 1)$. We generate t points on each segment joining a vertex of S to c. Indeed, let

$$p_{ij} = 2^{-j}c + (1 - 2^{-j})p_i$$

for $i = 1, \ldots, d+1$, and $j = 1, \ldots, t$. Finally, set

$$P = \{p_{ij} \colon i = 1, \ldots, d+1; j = 1, \ldots, t\} \cup S.$$

Now consider any splitter for P. By symmetry, we may assume that it has the label p_{1j} for some j. It is easy to check that the simplex $\{p_{1j}, p_2, \ldots, p_{d+1}\}$ contains the simplex $\{c, p_2, \ldots, p_{d+1}\}$. However, this latter simplex contains all points p_{ij} with $i \ge 2$. There are precisely $dt = dn'/(d+1)$ such points. □

Corollary 2.1 shows, for fixed dimension d, that a triangulation of a simplicial point set in general postion can be constructed with a linear number of simplices. Intuitively, one can image inserting the interior points one at a time. Each point lands in exactly one simplex if the point set is in general position. This simplex can be retriangulated creating $d+1$ new simplices. Suppose now that we relax the assumption of general position. We consider the case $d = 3$. It is possible that a point is inserted into a face of the existing triangulation. This face bounds two simplices (unless it is an external face, in which case it lies in one simplex and there is no difficulty). We may now treat both simplices independently and retriangulate each with the new point. This partitions each of the old simplices into three new simplices for a net gain of four simplices.

The problems caused by collinearity are more serious. Repeating our earlier point insertion process, suppose we insert a point into an edge of the existing triangulation. As shown in Fig. 3, in a set of $n+2$ points, such an edge may lie in as many as $n-1$ existing simplices. Inserting the point x into edge $\overline{12}$ in the figure creates an additional $n-1$ simplices. If we insert a further $n-3$ points into edge $\overline{12}$, we create a triangulation with $2n$ points and $(n-1)^2$ simplices. It can be shown that the given triangulation is unique for this point set.

Consider the preceding example of $2n$ points with an additional point y on edge $\overline{23}$, as shown in Fig. 4. First consider a partition of the simplex $\{1, 2, 3, 4\}$

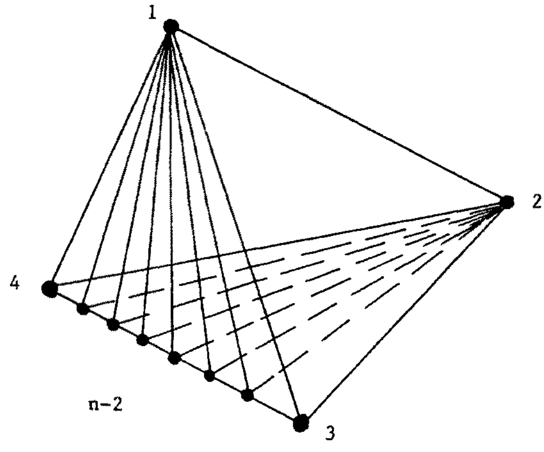


Fig. 3

into simplices $\{1, y, 3, 4\}$ and $\{1, 2, y, 4\}$. Each of these simplices is similar to the example of Fig. 3 and can be triangulated with only $2n-2$ simplices. On the other hand, if we ignore the point y and triangulate the remaining points, we obtain, as before, $(n-1)^2$ simplices. Inserting point y creates one additional simplex. Summarizing, we have proved the following:

Theorem 2.3.

(a) *A simplicial point set P in R^3 with at least $n \geq 5$ points and no 3 points collinear can be triangulated using at most $4(n-4)$ simplices.*
(b) *There exist $2n+1$ point sets P in R^3 that can be triangulated with as few as $2n-2$ and as many as $(n-1)^2+1$ simplices.*

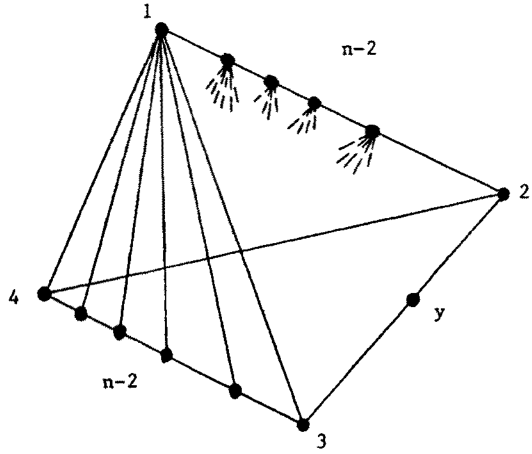


Fig. 4

(c) *There exist* $2n$ *point sets* P *in* R^3 *for which the unique triangulation requires* $(n-1)^2$ *simplices.*

In a very recent paper, Rothschild and Straus [7] study the problem of which n point sets (not necessarily simplicial) in d space produce the minimum and maximum number of simplices. Firstly they characterize point sets which produce the minimum number $n-d$ of simplices. Then they consider which point sets give the maximum number of simplices. Using the Upper Bound Theorem they give bounds on the maximum number T_n of simplices that can be constructed. For $d=3$ they show that

$$\frac{(n-3)(n-2)}{2} \leq T_n \leq \frac{(n+1)(n-2)}{2} - 4.$$

The examples achieving the minimum number of simplices are quite different from those that obtain the maximum number. In our case, we are interested in those point sets that simultaneously allow both a "good" triangulation (linear) and a "bad" triangulation (quadratic).

3. Algorithms for Three Dimensions

We will now describe an $O(n \log n + k)$ algorithm for triangulating sets in three dimensions. Initially we consider simplicial point sets. The heart of the algorithm involves finding a 3/4-splitter for P. Let $\{r, s, t, u\}$ be the extreme points of P. The procedure *SPLIT*(P, r, s, t, u) returns a 3/4-splitter for P. This algorithm is described implicitly in the proof of Theorem 2.1 and will not be given in detail here.

We are now ready to describe our triangulation algorithm, *TRIANGULATE*(P, r, s, t, u). This algorithm takes as input a set P and four points $\{r, s, t, u\}$ which are assumed in general position. The algorithm produces a triangulation of the points of P contained in the convex hull generated by these four points, $CH(r, s, t, u)$. For convenience in describing the recurisve part of the algorithm we allow points of P not in $CH(r, s, t, u)$. The excess points are simply discarded in the first step. The algorithm is a standard "divide and conquer" procedure based on *SPLIT*. A second procedure *EDGE-GEN*(r, s, t, u) is used to generate the edges of the simplex, specified by its arguments, which is assumed to have *empty* interior. In the case where the points of P are in general position, this procedure is trivial and produces the obvious four edges. In the general case, additional points may lie on the boundary of the simplex. These "degenerate" points are detected in the divide phase: points lying on a dividing edge ij are stored in a list E_{ij}; points lying in the interior of a dividing fact i, j, k are stored in a list F_{ijk}. The details of *EDGE-GEN* are given later.

TRIANGULATE(P, r, s, t, u).
1. Let $S = \{r, s, t, u\}$.
   For all $x \in P$ if $x \notin$ interior of S then
      begin

remove x from P
if x lies on edge i, j of S let $E_{ij} = E_{ij} \cup \{x\}$
else if x lies on face (i, j, k) of S let $F_{ijk} = F_{ijk} \cup \{x\}$
end.
2. If $|P| \geq 5$ then
begin
$y = SPLIT(P, r, s, t, u)$; $TRIANGULATE(P, r, s, t, y)$;
$TRIANGULATE(P, r, s, y, u)$; $TRIANGULATE(P, r, y, t, u)$;
$TRIANGULATE(P, y, s, t, u)$;
end
else $EDGE\text{-}GEN(r, s, t, u)$.

Finally we describe the procedure *EDGE-GEN* which triangulates a simplex with no points in its interior and possibly many points on its boundary. Along with the simplex, the procedure receives a list F_{ijk}, for each face $\{i, j, k\}$, of all points interior to that face and a list E_{ij}, for each edge i, j, of all points lying in the interior of that edge.

If each of the lists F_{ijk} is empty, we proceed to the next case. Otherwise we triangulatè the points in each face, excluding points lying on its edges. This process consists of four two-dimensional triangulation problems. Assume that F_{rst} is nonempty, and let x, y, z be the not necessarily distinct points in the triangulation of $F_{rst} \cup \{r, s, t\}$ that are adjacent, respectively, to edges rs, st, tr (as illustrated in Fig. 5). We then join vertex u to all points in F_{rst}, vertex x to all points in F_{rsu}, vertex y to all points in F_{stu}, and vertex z is then joined to all points in F_{tru}. These edges together with the edges generated in the four two-dimensional triangulation problems partition the simplex $\{r, s, t, u\}$ into a set of simplices whose number if linearly proportional to the number of points in the interior of its faces. Simplices with many points on their edges are then triangulated separately as described in the following paragraph.

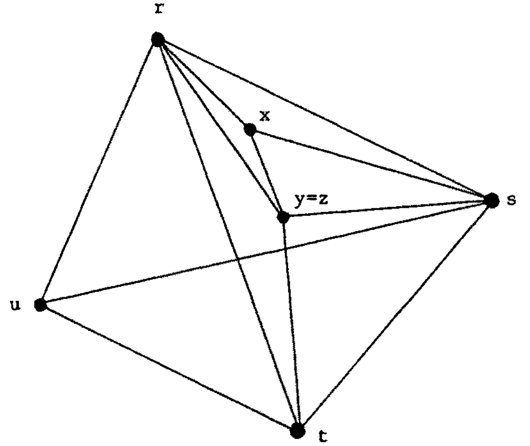


Fig. 5. Only triangulation of F_{rst} is shown.

In this case we consider simplices with no points in their interiors, no points interior to their faces, and many points on their edges. The first step is to sort all the edge lists. For concreteness we use the alphabetic order of the vertices to determine the direction of sorting, so that on edge ru the points are sorted from r to u. Consider a simplex $\{r, s, t, u\}$ and let $ij_1, ij_2, \ldots, ij_{k_{ij}}$ be the sorted points along some edge ij. If all the points lie on two disjoint edges, say rs and tu, it can be shown that the only possible triangulation is:

$$\begin{array}{l}\{r, rs_1, t, tu_1\}, \{rs_1, rs_2, t, tu_1\}, \ldots, \{rs_{k_{rs}}, s, t, tu_1\},\\ \{r, rs_1, tu_1, tu_2\}, \{rs_1, rs_2, tu_1, tu_2\}, \ldots, \{rs_{k_{rs}}, s, tu_1, tu_2\},\\ \vdots \\ \{r, rs_1, tu_{k_{tu}}, u\}, \{rs_1, rs_2, tu_{k_{tu}}, u\}, \ldots, \{rs_{k_{rs}}, s, tu_{k_{tu}}, u\}\end{array}$$

which has $(|E_{rs}|+1)(|E_{tu}|+1)$ simplices. Otherwise there exists a face, say $\{r, s, t\}$, with at least two nonempty edge lists. We triangulate the face $\{r, s, t\}$ as shown in Fig. 6, and then join the opposite vertex u to all points in $E_{rs} \cup E_{st} \cup E_{tr}$. These edges partition the simplex $\{r, s, t, u\}$ into a set of simplices whose number is linearly proportional to the number of points in the edge lists of face $\{r, s, t\}$. Each of the simplices that contain edges ru, su, and tu may have nonempty edge lists. Since these edges cannot be disjoint, we can partition the simplices into a linear number of simplices.

For a simplex $S = \{r, s, t, u\}$ with empty interior, let m_e be the number of points lying on the interior of edges of S and let m_f denotes the number of points lying in the interior of faces of S. We say that S is *degenerate* if $m_f = 0$ and if the extreme points can be labeled so that $|E_{rs}| > 0$, $|E_{tu}| > 0$ and for all other edges ij, $|E_{ij}| = 0$.

Theorem 3.1. *The number of edges produced by* EDGE-GEN *is either*

(i) $O(m_e + m_f)$ *if S is nondegenerate, or*
(ii) $(|E_{rs}|+1)(|E_{tu}|+1)$ *if S is degenerate.*

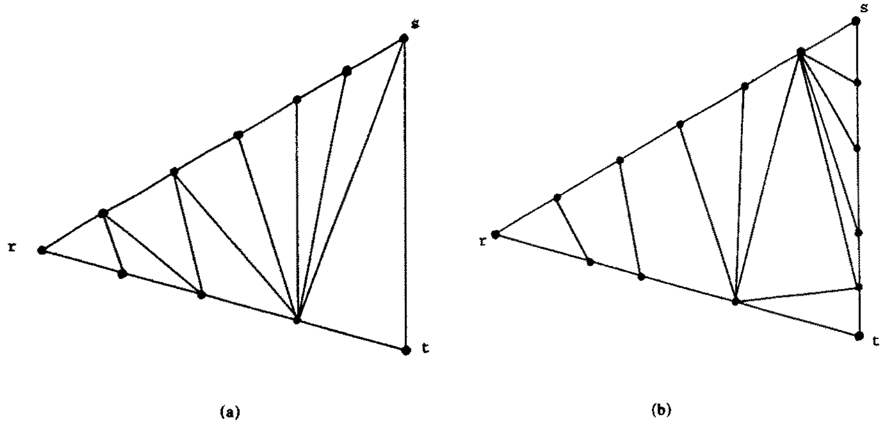


Fig. 6. (a) Two nonempty edge lists. (b) Three nonempty edge lists.

Triangulating points interior to the faces of an empty simplex requires $O(m_f \log m_f)$ time. After sorting the edge lists, we can triangulate an empty simplex with points on its edges in $O(k)$ time, where k is the number of output simplices. Since a point interior to a face is processed at most twice and the points on each edge list are ordered once, the total time spent in *EDGE-GEN* during the triangulation of P is $O(n \log n + k)$. As the rest of the algorithm *TRIANGULATE* runs in time $O(n \log n)$, the overall running time is thus $O(n \log n + k)$ time.

We conclude this section by describing how the assumption that the point sets are simplicial can be dropped. The main idea is to partition the point set into a collection of simplicial sets which are then processed separately by *TRIANGULATE.*

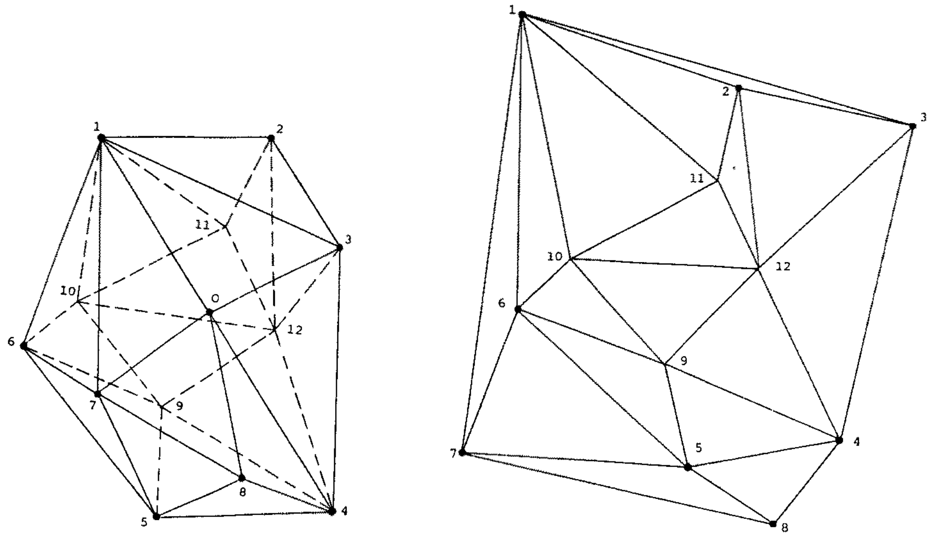
Given an arbitrary set P of n points in R^3, we begin by computing the convex hull using the $O(n \log n)$ time algorithm of Preparata and Hong [5], [6]. Let $CH(P)$ denote the subset of P consisting of those points on the convex hull. A vertex x of $CH(P)$ is chosen arbitrarily. The faces of the convex hull not containing x, *excluding* points of the set that may lie interior to these faces, are triangulated. Vertex x is then joined to all other points in $CH(P)$. This results in a decomposition of the convex hull into simplices, in $O(n)$ time. Now we describe how to distribute the nonconvex hull vertices into the interior, faces and edges of those simplices.

Let H and H' be two nonidentical parallel planes of support of the convex hull such that H intersects the convex hull at x. Let $l(y, x)$ denote the line through nonidentical points x and y. We project the convex hull vertices onto H', giving a set C^*, as follows:

$$C^* = \{y' \colon y' = l(y, x) \cap H', y \in CH(P), \text{and } y \neq x\}$$

so that y' denotes the projection of a convex hull point y onto the plane H'. Points in C^* are joined by an edge whenever they arise from convex hull vertices that were joined by an edge on the triangulated convex hull of P. C^* forms a planar subdivision of a convex polygon whose interior regions are all triangles. A triangle on a face of the convex hull not containing x is mapped into an interior triangle of the planar subdivision, and a triangle on a face of the convex hull containing x is mapped onto an edge of the exterior face of C^*. This is illustrated in Fig. 7(a) and (b). Figure 7(a) shows a three-dimensional convex polyhedron with hidden lines dashed. The planar subdivision C^* obtained by projecting from vertex 0 is shown in Fig. 7(b).

We now construct a subdivision hierarchy of $O(\log n)$ height using the $O(n \log n)$ time and $O(n)$ space algorithm of Kirkpatrick [4], [6]. For each nonconvex hull vertex, z, we search the subdivision hierarchy for the location of its projection, $z' = l(z, x) \cap H'$, in the planar subdivision C^*. In the case where the points of the set are in general position, z' must lie in the *interior* of some triangle and is directly associated with the corresponding simplex. In the general case, additional tests must be performed to check whether z lies on an edge, on a face, or in the interior of a simplex. Details of these tests are as follows:



(a) (b)

Fig. 7

if z' lies on vertex i' of C^* *then*
  let $E_{ix} = E_{ix} \cup \{z\}$
if z' lies on edge $i'j'$ of C^* *then*
  if z lies on edge ij of the convex hull *then*
    let $E_{ij} = E_{ij} \cup \{z\}$
  else
    let $F_{ijx} = F_{ijx} \cup \{z\}$
if z' lies inside triangle $i'j'k'$ of C^* *then*
  if z lies on face $\{i, j, k\}$ of the convex hull *then*
    let $F_{ijk} = F_{ijk} \cup \{z\}$
  else
    let $P_{ijkx} = P_{ijkx} \cup \{z\}$.

The simplices are then processed separately by *TRIANGULATE*. Along with the points interior to each simplex, the algorithm receives as input lists of the points interior to each face and lists of points lying on each edge.

Our splitting algorithms generalize in a straightforward way to d-dimensional space. Assuming the point sets are simplicial and in general position Corollary 2.1 shows that they may be triangulated efficiently. The triangulation algorithm for general point sets does not, however, readily generalize due to two difficulties: (i) convex hulls in higher dimensions are more difficult to compute and may contain $O(n^{\lfloor (d-1)/2 \rfloor})$ faces; (ii) no general methods for point location in higher dimensions are known and so distributing the points into simplices is computationally difficult.

4. Note

Some of the results described in this paper have since been independently rediscovered by Edelsbrunner *et al.* [2]. In particular, for points sets in general position, they have also found an $O(n \log n)$ triangulation algorithm in three dimensions based on the idea of splitting. This paper also contains several interesting combinatorial results on extremum problems concerning triangulations, that are not covered in our paper. Warren Smith has informed the authors that he can improve the time complexity in Theorem 2.1 to $O(ns(d)/d + d^3)$, where $s(d)$ is the time required to multiply two $d \times d$ matrices (private communication).

References

1. D. Avis and H. ElGindy, Triangulating simplicial point sets in space, *Proceedings of the Second ACM Conference on Computational Geometry*, IBM, Yorktown Heights, New York, 1986.
2. H. Edelsbrunner, F. P. Preparata, and D. B. West, Tetrahedrizing point sets in three dimensions, Report No. UIUC DCS-R-86-1310, Department of Computer Science, University of Illinois at Urbana-Champaign, 1986.
3. B. Grünbaum, *Convex Polytopes*, Wiley, New York, 1967.
4. D. Kirkpatrick, Optimal search in planar subdivisions, *SIAM J. Comput.* **12** (1983), 28-35.
5. F. P. Preparata and S. J. Hong, Convex hulls of finite sets of points in two and three dimensions, *Comm. ACM* **22** (1977), 87-93.
6. F. P. Preparata and M. I. Shamos, *Computational Geometry*, Springer-Verlag, New York, 1985.

Received April, 1986, and in revised form September 22, 1986.