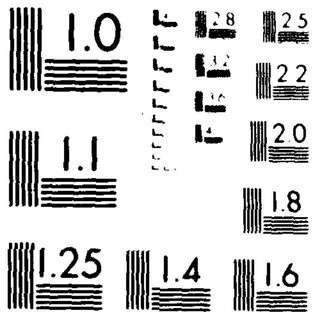
AD-A085 016 ILLINOIS UNIV AT URBANA-CHAMPAIGN APPLIED COMPUTATION--ETC F/G 12/1
NEW RESULTS ON PLANAR TRIANGULATIONS.(U)
JUL 79 P D GILBERT N00014-79-C-0424
UNCLASSIFIED ATC-15 NL

1 of 1
AD
A085016

END
DATE
FILMED
7-80
DTIC


1.0
2.8
2.5
3.2
2.2
2.0
1.1
1.8
1.25
1.4
1.6

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE

READ INSTRUCTIONS BEFORE COMPLETING FORM

1. REPORT NUMBER

2. GOVT ACCESSION NO. AD-A085 016

3. RECIPIENT'S CATALOG NUMBER

4. TITLE (and Subtitle)
NEW RESULTS ON PLANAR TRIANGULATIONS

5. TYPE OF REPORT & PERIOD COVERED
Technical Report

6. PERFORMING ORG. REPORT NUMBER
R-850 (ACT-15); UILU-ENG-78-2243

7. AUTHOR(s)
PETER D. GILBERT

8. CONTRACT OR GRANT NUMBER(s)
NSF MCS 76-17321
N00014-79-C-0424

9. PERFORMING ORGANIZATION NAME AND ADDRESS
Coordinated Science Laboratory
University of Illinois at Urbana-Champaign
Urbana, Illinois 61801

10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS

11. CONTROLLING OFFICE NAME AND ADDRESS
Joint Services Electronics Program

12. REPORT DATE
July, 1979

13. NUMBER OF PAGES
42

14. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office)

15. SECURITY CLASS. (of this report)
UNCLASSIFIED

15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)

Approved for public release; distribution unlimited

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

18. SUPPLEMENTARY NOTES

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

Computational Geometry
Triangulations
Greedy Triangulation
Segment Trees
Efficient Algorithms

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

Several new results in planar triangulations are presented. In particular, an $O(N^2 \log N)$ time worst-case algorithm for the construction of the 'greedy' triangulation is presented, an improvement over the previously best-known construction in time $O(N^3)$. The minimum weight triangulation (MWT) of a simple N-vertex polygon is shown to be constructible in time $O(N^3)$. A local property of MWTs is proved, a side result of which is that the shortest edge between a pair of points must be in a MWT. Also, it is shown that the Delaunay

20. ABSTRACT (continued)

Triangulation is not 'approximately optimal', and that the ratio of the total length of the Delaunay Triangulation of N points to that of the MWT can be as large as roughly N/2.

NEW RESULTS ON PLANAR TRIANGULATIONS

BY

PETER D. GILBERT

B.S., University of Illinois, 1976

THESIS

Submitted in partial fulfillment of the requirements
for the degree of Master of Science in Computer Science
in the Graduate College of the
University of Illinois at Urbana-Champaign, 1979

Thesis Advisor: Professor F. P. Preparata

Urbana, Illinois

ABSTRACT

Several new results in planar triangulations are presented. In particular, an $O(N^2 \log N)$ time worst-case algorithm for the construction of the 'greedy' triangulation is presented, an improvement over the previously best-known construction in time $O(N^3)$. The minimum weight triangulation (MWT) of a simple N-vertex polygon is shown to be constructible in time $O(N^3)$. A local property of MWTs is proved, a side result of which is that the shortest edge between a pair of points must be in a MWT. Also, it is shown that the Delaunay Triangulation is not 'approximately optimal', and that the ratio of the total length of the Delaunay Triangulation of N points to that of the MWT can be as large as roughly N/2.

ACKNOWLEDGMENT

My sincerest thanks and appreciation are reserved for Franco P. Preparata. Hopefully, his help and insights are reflected well in this thesis.

This research was supported by the National Science Foundation under Grant MCS 76-17321 and by the Joint Electronic Services Program under Contracts DAAB-07-72-C-0259 and DAAG-29-78-C-0016.

TABLE OF CONTENTS

CHAPTER 1

INTRODUCTION

1.1 Computational Geometry

Computational geometry is a relatively new field of study, dealing with time and space complexities of geometric problems. Many important problems, both theoretically and from the point of view of applications, are geometric in nature, including, for example, wire-routing, the Traveling Salesman Problem, and the Steiner Tree Problem. Solving these efficiently necessitates developing new algorithms and new tools that take advantage of the geometric nature of the problems. It is currently unclear to what degree geometry affects the complexity of problems, and because of the relative youth of the subject, there remains much work to be done and a large number of open problems.

Much of the early work in computational geometry has either been done or surveyed by M. I. Shamos [1,2], who has given efficient algorithms for many fundamental problems in the field. For a good survey of work in the field, see reference [1].

This thesis is concerned with some algorithms, results, and techniques which could prove very useful in this rapidly expanding area of research.

1.2 Preliminary Definitions

Before proceeding, it is necessary to define some terms.

Let V be a set of N distinct points in the plane, and let L be the set of $\frac{N(N-1)}{2}$ edges (or line segments) between points in V. Two distinct edges e_1 and e_2 properly intersect if e_1 and e_2 intersect at a single point which is not an endpoint of both e_1 and e_2.

A triangulation of V is a maximal subset, T, of L such that no two edges of T properly intersect. The term convex hull is used to refer to the convex polygon on the boundary of a triangulation of a set of points. More precisely, the points on the convex hull of a set of N points are all those points that are not in the interior of any triangle formed by any three of the N points. The edges on the convex hull are all the edges with endpoints on the convex hull, and which do not properly intersect any of the edges connecting pairs of points. Thus, the edges on the convex hull must be in any triangulation of the set of points.

Some properties of triangulations are immediately apparent or easily derived:

1. Each interior region of a triangulation determined by T and V is triangular in shape.
2. Each edge in L is either in T or properly intersects an edge in T.
3. The number of edges in a triangulation, $|T|$, is equal to $3N - |CH| - 3$, where $|CH|$ is the number of points on the convex hull of V.

Figure 1 should help in the visualization of these concepts.

A triangulation of a set of points

A convex hull.

Figure 1. $N = 10$; $|T| = 20$; $|CH| = 7$.

1.3 The Minimum Weight Triangulation Problem

For any given set of points, there is in general more than one possible way in which they can be triangulated. One particular triangulation of a set of points, the Minimum Weight Triangulation (also called the Optimum Triangulation), is defined as follows. Of all possible triangulations of a set of points, a triangulation for which the sum of the lengths of the edges is a minimum is called a Minimum Weight Triangulation (MWT) of the set of points.

Previously, very little has been known about MWTs besides the definition, most results being counter-examples of propositions about MWTs, and thus illustrating what a MWT is not. In Chapter 3 we present a polynomial-time algorithm for constructing the MWT of a simple polygon, and prove a local property of MWTs, a side result of which is that the shortest edge must be in the MWT.

E. L. Lloyd [3] has conjectured that the problem of constructing the MWT of a set of N points is NP-complete, and has presented counter-examples to several propositions regarding MWTs. Other work on the MWT problem has been done by Düppe and Gottschalk [4], and Shamos and Hoey [1].

1.4 The Greedy Triangulation Problem

One algorithm purported to solve the MWT problem (in time polynomial on the number of points) was published by Düppe and Gottschalk [4]. Unfortunately, the algorithm does not necessarily yield a MWT [3], but is of interest, nevertheless. The algorithm, which defines the Greedy Triangulation (GT), is as follows:

1. Construct the $\binom{N}{2}$ possible line segments between pairs of points.

2. Order these line segments by their lengths.

3. Include the shortest of these line segments in the triangulation, and eliminate all line segments which intersect this shortest one.

4. Continue constructing the triangulation as in step 3, using the remaining line segments.

An alternate description of the construction of the Greedy Triangulation is the following:

1. Set GT to the empty set. Sort the $\frac{N(N-1)}{2}$ edges of L by length.

2. Consider each edge e in L in order, from shortest to longest. If e does not properly intersect any edge already in GT, then add e to GT.

3. GT is the set of edges in the Greedy Triangulation.

It is clear that a straightforward implementation of this algorithm uses $O(N^3)$ time and $O(N^2)$ space. G. K. Manacher and A. L. Zobrist [5] have presented an $O(N^3)$ time (worst case) and O(N)-space algorithm to construct the GT, with average running time of $O(N^2 \log N)$. They also show that, contrary to what had been thought, the GT need not be a good approximation to the MWT; that the sum of the lengths of the edges of the GT may be greater than that of the MWT by a factor of $O(N^{1/3})$. Chapter 2 presents two $O(N^2)$ space algorithms for the construction of the GT, with worst-case running times of $O(N^{5/2})$ and $O(N^2 \log N)$, respectively. It is also shown that the total length of the Delaunay Triangulation [2] can be N/2 times as large as that of the MWT.

We note also that it has been shown [2] that $\Omega(N\log N)$ is a lower bound on the time required to construct <u>any</u> triangulation in 'good' form, i.e., all points which are edge-adjacent to a particular point are sorted by polar angle around that point, and that this bound is achievable.

CHAPTER 2

GREEDY TRIANGULATIONS

2.1 Overview

In this chapter a "greedy" algorithm for triangulating a set of points is discussed, and two algorithms for the construction of this triangulation are presented. Each of these algorithms is an improvement over the previously best known $O(N^3)$ time worst-case construction of the Greedy Triangulation.

A brute force construction of the Greedy Triangulation (GT) can be achieved in $O(N^3)$ time and $O(N^2)$ space (see section 1.3). One would naturally expect lower time and space bounds for two basic reasons:

1. An O(NlogN) time triangulation algorithm exists (for the Delaunay Triangulation), this being the lower bound for producing <u>any</u> triangulation in 'good' form [2], and

2. The GT problem is a relatively simple and straight-forward problem in computational geometry.

Manacher and Zobrist have called triangulations of the following form "conformal" [5]:

1. $T \leftarrow \emptyset$; $L \leftarrow \{$all edges connecting pairs of points$\}$.

2. Determine an edge $e \in L$, to be added to the triangulation, and set $T \leftarrow T \cup \{e\}$.

3. Repeat step 2 until T is a triangulation.

In a conformal triangulation algorithm, an edge which is added to the triangulation is not deleted from the triangulation later. In this respect, the two algorithms presented in this chapter are conformal.

2.2 An $O(N^{5/2})$ Time Greedy Triangulation Algorithm

We shall now present an algorithm for the construction of a GT of a set of N points in the plane. It is similar in structure to the algorithm of section 1.3, but "batches" updates to the data structures in order to achieve an $O(N^{5/2})$ time worst-case construction.

At the heart of any algorithm are its data structures. In this proposed algorithm for constructing the GT of N points, we have L, which is an ordered set of the $\frac{N(N-1)}{2}$ edges between pairs of points, sorted in order of increasing length, from which we consider edges one at a time. Also, we have an $N \times N$ adjacency matrix ADJ, the elements of which are Boolean values ADJ(i,j), initialized to 1 (iff $i \neq j$), and set to 0 when the edge $p_i p_j$ is determined to not be in the GT, so that at the end of the algorithm ADJ is a standard adjacency matrix representation of the edges in the GT.

Associated with each point p_s is a star-structure, $STAR_s$, which is an ordered subset of $V-\{p_s\}$, the points being ordered by polar angle around p_s. Initially, $STAR_s = V-\{p_s\}$, ordered by polar angle. If a point p_i is deleted from the star-structures $STAR_s$, this corresponds to the determination that the edge $p_s p_i$ is not in the GT. In the course of the algorithm, the elements ADJ(s,i) which equal 1 correspond exactly with those points p_i in $STAR_s$. Initializing these data structures can be done in time $O(N^2 \log N)$.

To achieve a "low" upper bound on the running time, we "batch" updates to these data structures, so that we process a(N) edges at a time. These edges have been determined to be in the GT, and are kept as an unordered set of edges H, with $|H| \leq a(N)$. In the subsequent analysis, the value of a(N) will be chosen for optimality.

To test an edge p_ip_j for inclusion in GT, we must check ADJ(i,j), which was current as of the last batched update, but we must also check for proper intersection between p_ip_j and each of the edges currently in the set H, which have been added to GT since the last update of the star-structures and the adjacency matrix.

We are now ready to sketch the algorithm (see Figure 2):

1. (Initialization)

   Let V be a set of N points in the plane: $p_1, p_2, \ldots, p_N$.

   $L \leftarrow \{$all edges whose endpoints are in V$\}$, sorted by length.

   $STAR_s \leftarrow V-\{p_s\}$, sorted by polar angle around p_s; for $s = 1,2,\ldots,N$.

   $ADJ(i,j) \leftarrow 1$ for $i,j = 1,2,\ldots,N$, $i \neq j$. $GT \leftarrow \emptyset$; $H \leftarrow \emptyset$.

2. (Should a considered edge be added?)

   Consider each edge p_ip_j in L, in order of increasing length. If all edges have already been considered, then perform step 4 (update data structures) a final time and stop. If ADJ(i,j) = 1, and, if p_ip_j does not properly intersect any edge in H, then perform step 3 (add p_ip_j to GT and H), and possibly step 4 (update STARs); otherwise, continue considering edges.

3. (Insert edge)

   $GT \leftarrow GT \cup \{p_ip_j\}$; $H \leftarrow H \cup \{p_ip_j\}$.

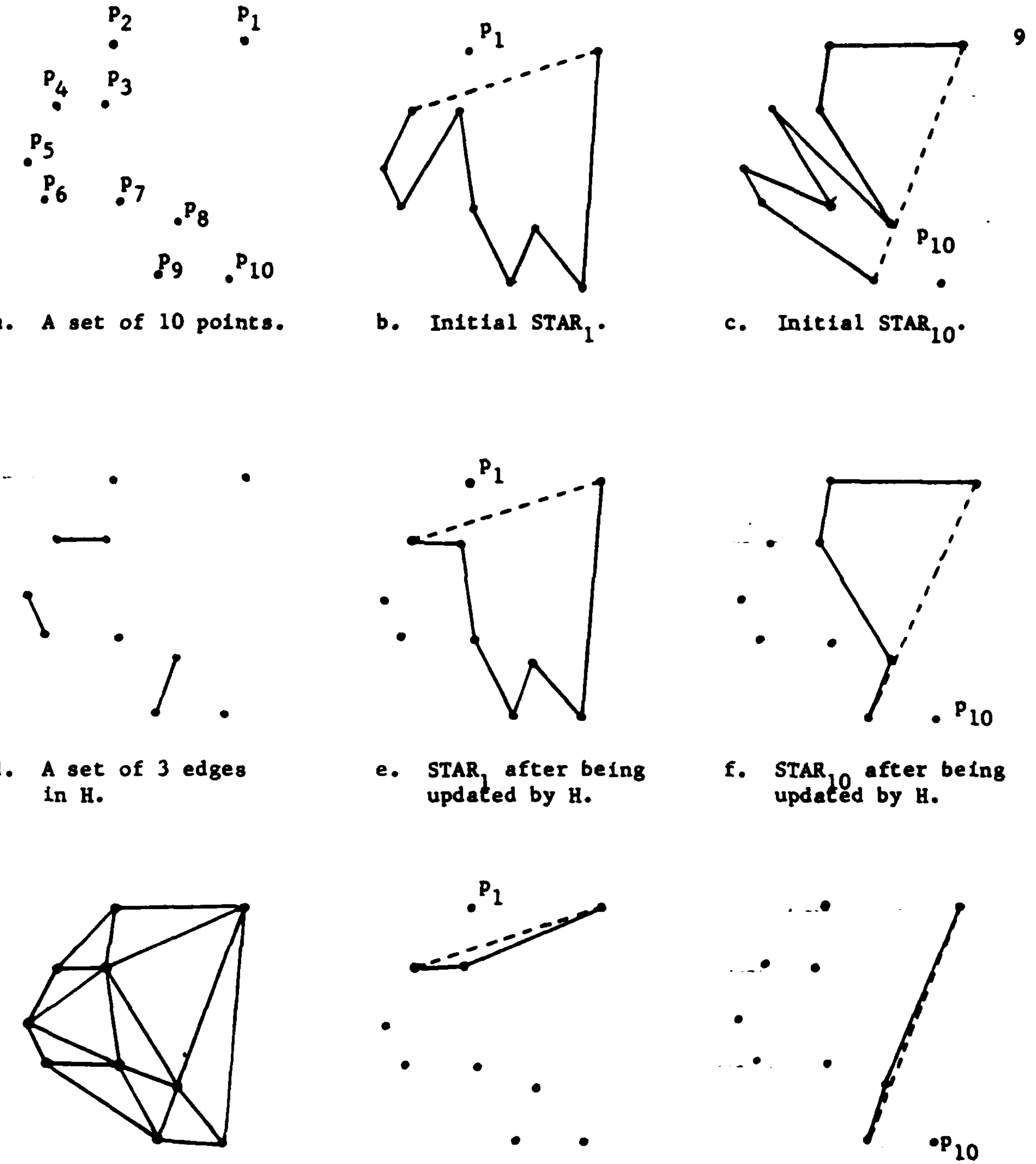
   If $|H| = a(N)$ then perform step 4, and continue with step 2, else just continue with step 2.

4. (Update Stars and ADJ)

   For all $s \in \{1,2,\ldots,N\}$, set $STAR_s \leftarrow STAR_s - \{p_i \mid$ edge p_sp_i properly intersects an edge in H$\}$.

   If p_sp_i properly intersects an edge in H, then set ADJ(s,i) to 0

   Set $H \leftarrow \emptyset$ and go to step 2.



a. A set of 10 points.

b. Initial $STAR_1$.

c. Initial $STAR_{10}$.

d. A set of 3 edges in H.

e. $STAR_1$ after being updated by H.

f. $STAR_{10}$ after being updated by H.

g. Greedy Triangulation.

h. Final $STAR_1$.

i. Final $STAR_{10}$.

Figures 2(a-i). Examples of the updating of STAR-structures.

Most of the work in the algorithm is done in step 4, updating the data structures, which we now analyze in more detail.

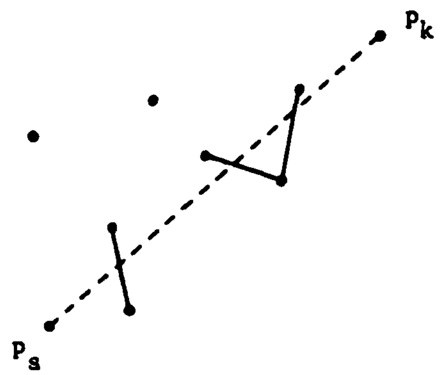
Given a ray r from a point p, and a set E of nonintersecting edges, we let $E_r = \{e | e \in E$, e and r properly intersect$\}$. Then, for two edges $e_1, e_2 \in E_r$ we say e_1 is closer than e_2 ($e_1 \underset{r}{\leq} e_2$) if the intersection of e_1 and r is nearer to p than is the intersection of e_2 and r. This is a proper ordering, since, if r and r' are rays from point p which both intersect e_1 and e_2, then $e_1 \underset{r}{\leq} e_2$ and $e_2 \underset{r'}{\leq} e_1$ would imply a proper intersection between e_1 and e_2, which we know to be impossible by the definition of E.

Lemma 2.2.1: Let $h = |H|$. The step "$STAR_s \leftarrow STAR_s - \{p_i |$ edge $p_s p_i$ properly intersects an edge in H$\}$" can be done in time O(h log h + N), as can the corresponding update of ADJ.

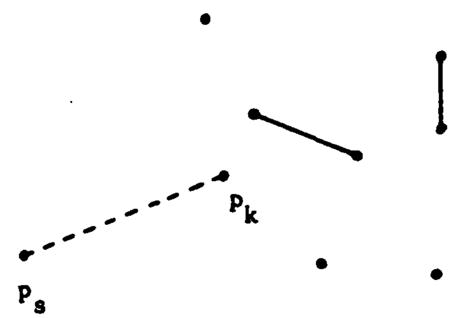
Proof: Consider the following method of performing the step.

Sort the 2h endpoints of edges in H by polar angle around p_s, associating with each endpoint information saying which edge it is from (if any endpoint coincides with p_s, then its corresponding edge can be ignored in the updating of this star-structure, since it will not properly intersect any other $p_s p_k$). Let H_{r_0} be the sequence of edges from H which are intersected by a chosen ray r_0, emanating from p_s, ordered according to their closeness to p_s; H_{r_0} is organized as a height-balanced binary tree (AVL priority queue). Initially we set $U \leftarrow H_{r_0}$, where r_0 is a ray from p_s at polar angle 0. Then we scan the points of $HUSTAR_s$ sorted according to their polar angle around point p_s.

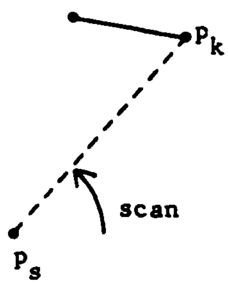
Let p_k be the point being scanned. Either $p_k \in STAR_s$ (Fig. 3a,b) or p_k is an endpoint of one or more edges in H, (Fig. 3c,d,e) or both. If



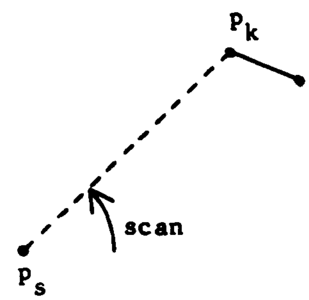
a. Edge $p_s p_k$ properly intersects the closest edge in U (solid edges are in U).



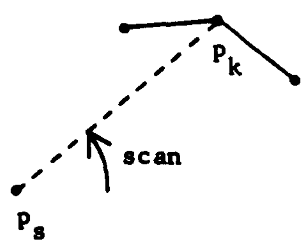
b. Edge $p_s p_k$ does not properly intersect the closest edge in U (solid edges are in U).



c. Point p_k is a 'leading' endpoint of an edge in H.



d. Point p_k is a 'trailing' endpoint of an edge in H.



e. Point p_k is both a 'leading' and a 'trailing' endpoint of two edges in H.

Figure 3(a-e). Scanning points from STARs and endpoints of edges in H (see lemma 2.2.1).

$p_k \in STAR_s$ and $p_s p_k$ properly intersects the closest edge in U (Fig. 3a), then set $STAR_s \leftarrow STAR_s - \{p_k\}$ (and ADJ(s,k) is set to 0).

Additionally, if p_k is an endpoint of an edge in H, then U must be updated. If p_k is the 'leading' endpoint (Fig. 3c), in the given ordering of polar angles, of an edge $p_k p_j$ in H, then $p_k p_j$ must be added to U; if p_k is the 'trailing' endpoint (Fig. 3d), then $p_j p_k$ must be removed from U. Since p_k is a trailing endpoint of the edge $p_j p_k$ in H if and only if $p_j p_k$ is already in U, this update to U can be written:

$$U \leftarrow U \cup \{p_j p_k \mid p_j p_k \in H,\ p_j p_k \notin U\}$$
$$-\{p_j p_k \mid p_j p_k \in H,\ p_j p_k \in U\}$$

We continue scanning until all points in $STAR_s$ have been scanned. Since an edge is added to and deleted from U only once, the time bound follows immediately. ❑

Notice that with this GT algorithm, at the algorithm's completion, the Greedy Triangulation is available in three forms: (1) As a set of edges (GT): (2) as an adjacency matrix (ADJ); (3) as a collection of star-structures, where each point's star-structure contains that point's neighbors, in polar order.

Analysis of GT algorithm:

Initializing L (sorting $\frac{1}{2}N(N-1)$ lengths): $O(N^2 \log N)$ time.

Initializing the N star-structures: $O(N \cdot N \log N)$ time.

Initializing the adjacency matrix: $O(N^2)$ time.

Checking an edge for inclusion in GT (checking the edge against all current members of H): $O(a(N))$ time.

Checking $\frac{1}{2}N(N-1)$ edges: $O(N^2 a(N))$ time.

Updating a star-structure with a(N) edges (by lemma 2.2.1):

$O(a(N) \log a(N) + N)$ time.

Performing $O(3N/a(N))$ batched updates (a triangulation contains no more than 3N edges), of N star-structures each:

$O(N^2 (a(N) \log a(N) + N)/a(N))$ time.

Overall: $O(N^2 \log N + N^2 a(N) + N^3/a(N))$ time.

By choosing $a(N) = O(N^{\frac{1}{2}})$, where the constant of proportionality is independent of N, we obtain the following:

Theorem 2.2.2: The Greedy Triangulation of a set of N points can be constructed in $O(N^{5/2})$ time and $O(N^2)$ space.

2.3 An $O(N^2 \log N)$-time Greedy Triangulation Algorithm

In this section we describe another algorithm, similar in structure to that of section 2.2, for constructing the GT of a set of N points. A new data structure is used which will permit checking an edge for inclusion in the GT in time $O(\log N)$ and the updating for a single edge added to the GT in time $O(N \log N)$, thereby achieving an $O(N^2 \log N)$ running time for the algorithm.

The data structure is an array of Range-trees. For integers $a,b \in \{1,2,\ldots,N\}$ a Range tree of the range (a,b) (RT(a,b)) is a binary tree whose nodes contain a subrange (of (a,b)) field, such that:

1. the root has subrange (a,b).
2. if the subrange in a node has width 1 $(a+1=b)$, then the node has no subtrees.
3. otherwise, the node has left and right subtrees, which are

RT(a, $\lfloor(a+b)/2\rfloor$) and RT($\lfloor(a+b)/2\rfloor$, b), respectively,

It is simple to show that RT(a,b): is a binary tree, each node having 0 or 2 sons; has (b-a) leaves; has 2(b-a)-1 nodes (inner & leaves); is balanced; and has maximum depth of $\lceil \log_2(b-a)\rceil + 1$ (see Figure 4). We call basis ranges the subranges specified in the nodes of a Range-tree. An exact covering of a range (a_0,b_0) is a set of basis ranges (a_i,b_i) such that the union of the basis ranges is equal to (a_0,b_0).

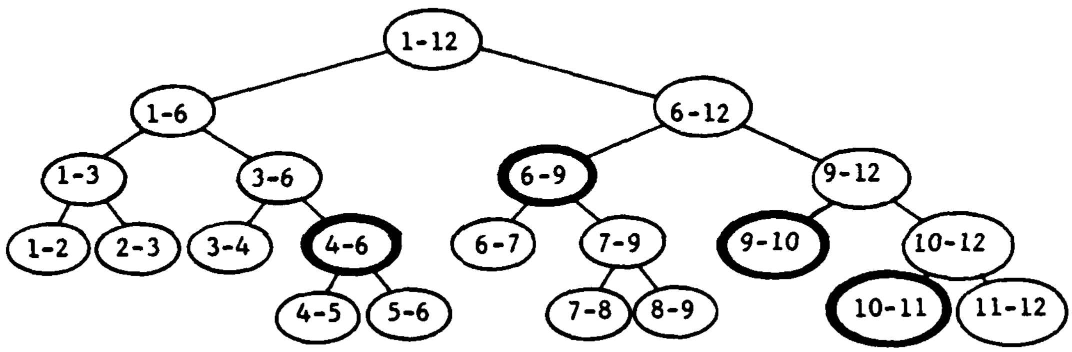


Figure 4. A Range-tree, RT(1,12), with the minimal exact cover of the range 4..11 marked.

Claim: Any subrange (a_0,b_0) of the range (a,b) can be exactly covered using no more than $2\lceil \log_2(b-a)\rceil + 2$ basis ranges from RT(a,b).

Proof: Consider all basis ranges (a_i,b_i) of RT(a,b) such that $a_0 \le a_i < b_i \le b_0$. Let this be the initial covering, which we now prune to the claimed number of basis ranges.

All of the basis range nodes which are leaves of the Range-tree have brothers which are also in the initial covering, except possibly two such nodes at the extremes of the range. A pair of brothers is exactly covered by their father. Remove these pairs of brothers from the covering set, leaving at most two basis ranges at this level. Proceeding similarly level-by-level to the root, it is clear that any subrange of (a,b) can be exactly covered using basis ranges in number no more than twice the depth of the Range-tree, which completes the proof. □

Also, it should be realized that all these nodes can be marked or visited in time O(log(b-a)).

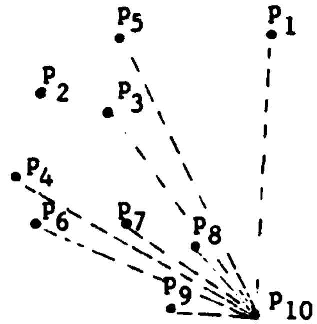
Each p_s will have associated with it a RT(1,N), the nodes of which also contain a line-segment field, described below. A number k as the end of a basis range in this Range-tree corresponds to the direction from p_s to another point. If the N-1 points $V-\{p_s\}$ are sorted by polar angle around p_s, and p_i is the k^{th} point in this ordering, then the number k as the end of a basis range is associated with point p_i in the numbering scheme around p_s. An $N \times N$ array, Global-to-local, is used to convert indices from the global numbering to the local numbering scheme around p_s. In the above case, we have Global-to-local $(s,i) = k$. This array can be initialized in $O(N^2 \log N)$ time (N sorting operations, of N-1 points each). Direction N (around p_s) is considered to be the same as direction 1, for circularity.

The line-segment field of a node with basis range (a_i,b_i), if not empty, will contain the name of the edge which has been added to the GT, spans the range, and is the closest to p_s. Thus, suppose that edge $p_s p_i$ is being considered for inclusion in the GT; we first refer to the

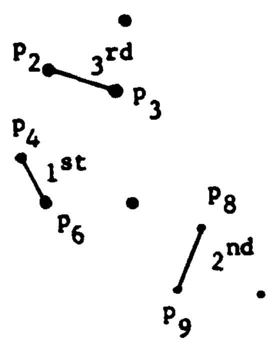
Global-to-local array and obtain an integer $k =$ Global-to-local (s,i), and then traverse a path of the Range-tree for point p_s from the root to a leaf corresponding to the integer k. If p_sp_i properly intersects any of the edges in the line-segment field of the nodes of this path, then p_sp_i cannot be added to the GT: otherwise (since edges are considered in order of increasing length), the edge p_sp_i should be added to the GT. Because the depth of the tree is O(logN), testing an edge for inclusion in GT requires O(logN) time.

When an edge is determined to be in the GT, the line-segment fields of the Range-trees must be updated. Consider updating the Range-tree for point p_s with the edge p_ip_j. If $i = s$ or $j = s$, then no update of this RT need be done. Otherwise, we look up Global-to-local (s,i) and Global-to-local (s,j), and let f and g be the smaller and larger of these, respectively. Then, visiting each node of the RT in the minimal exact cover of (f,g), if the line-segment field is empty, or p_ip_j is closer to p_s than the edge named in that field, set the line-segment field to p_ip_j (actually, if a ray from p_s at polar angle 0 properly intersects p_ip_j, then we must break (f,g) into the two parts $(1,f)$ and (g,N), and perform a similar update with each of these). This can be done in time O(logN).

To analyze this proposed GT algorithm, we note that the initialization of L, the N Range-trees, and the array Global-to-local can be done in $O(N^2 \log N)$ time. Checking an edge for inclusion in GT takes O(logN) time, and hence, checking $\frac{1}{2}N(N-1)$ edges for inclusion in GT takes $O(N^2 \log N)$ time. Since updating a single Range-tree with a single edge can be done in time O(logN), updating N Range-trees with the O(N) edges added to the GT can be



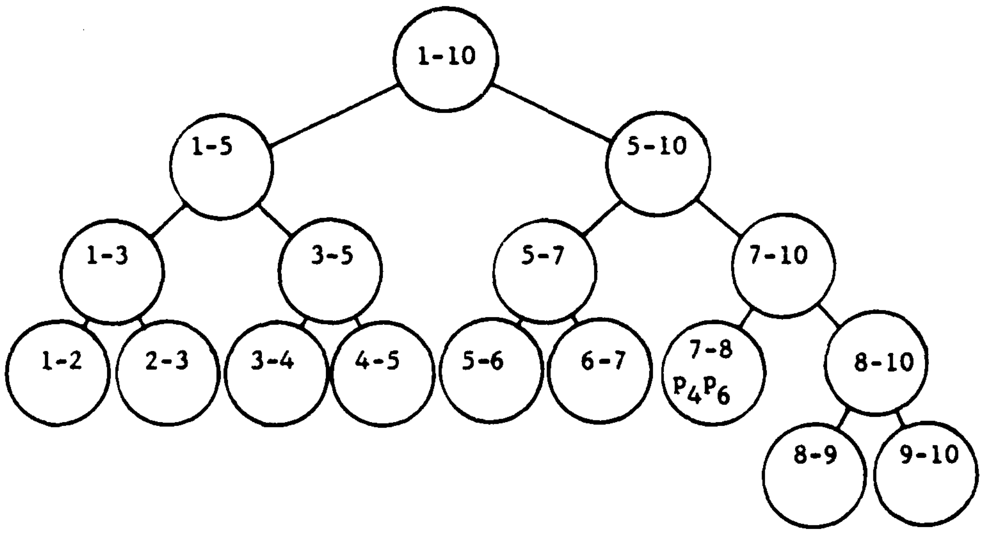
a. A set of 10 points.



b. First 3 edges added to GT.

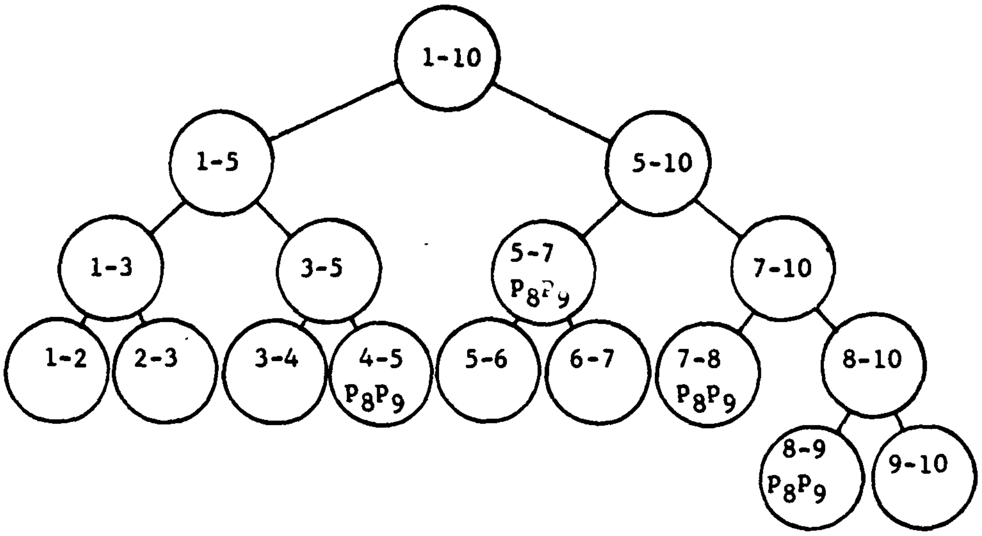
i	1	2	3	4	5	6	7	8	9	10
Global-to-local (10,i)	1	5	3	7	2	8	6	4	9	-

c. Part of the Global-to-local array.

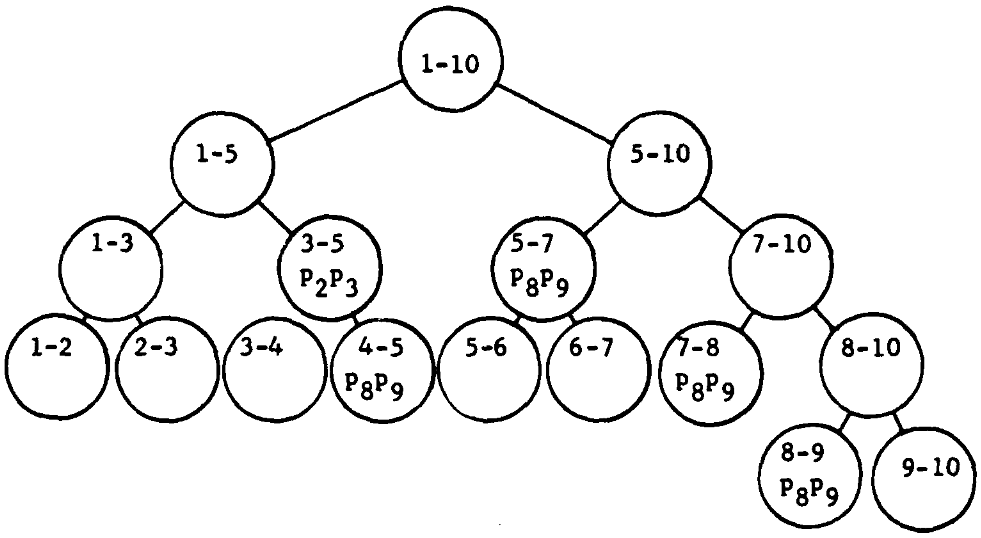


d. Range-tree of point 10 after edge p_4p_6 has been added to GT.

Figure 5(a-d) Examples of updating a Range-tree.



e. Range-tree of point 10 after edge p_8p_9 has been added to GT.



f. Range-tree of point 10 after edge p_2p_3 has been added to GT.

Figure 5(e-f) Examples of updating a Range-tree.

done in $O(N^2 \log N)$ time. Thus, the overall running time of the algorithm is $O(N^2 \log N)$, and we have proved:

Theorem 2.3: The GT of a set of N points can be constructed in $O(N^2 \log N)$ time and $O(N^2)$ space.

By being able to essentially occlude (possibly) many points from being visible from a point p_s at once, without sacrificing fast checking of edges for inclusion in the GT, we achieve a "nicely balanced" $O(N^2 \log N)$ time construction of the Greedy Triangulation for N points.

We note here that a Range-tree structure has been useful in several other computational geometry algorithms [8,9], such as computing the area of the union of N aligned rectangles (sides parallel to the x and y axes) in time $O(N \log N)$ [10].

CHAPTER 3

MINIMUM WEIGHT TRIANGULATIONS

3.1 Introduction

Presented in this chapter are several results concerning Minimum Weight Triangulations (MWT).

A construction of the MWT of a simple polygon in $O(N^3)$ time and $O(N^2)$ space is demonstrated in section 3.2.

The major result of this chapter involves a proof of a local property of MWTs, which may be very useful in further work on MWTs. A side result of this theorem answers the previously open question of whether the shortest edge is in a MWT of a set of vertices; it is. We now informally describe this theorem.

Given two simple closed curves in the plane, one entirely enclosing the other, the plane is divided into three regions; inner, middle, and outer. We consider MWTs of a set of points which lie in the inner and outer regions only; that is, there are no points in the middle region. Intuitively, if

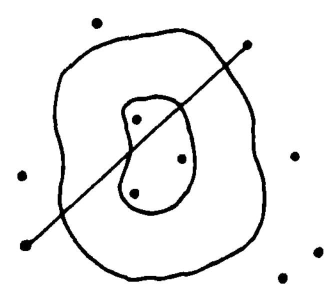


Figure 6. Illustration of a separator segment.

inner region is correctly placed, and is sufficiently "small", we would expect there to be no edge in the MWT connecting two points in the outer region, passing through the inner region, and separating the points in the inner region into two nonempty sets. We call such an edge a separator segment.

The theorem is primarily concerned with the case when the two curves are a pair of concentric circles, and on sufficient conditions which guarantee that there be no separator segment in the MWT of these points. The shortest edge between a pair of points is easily shown to meet these requirements, from which the fact that this shortest edge must be in a MWT of the points follows.

3.2 The MWT of a Polygon

We state the problem of triangulating a polygon as follows: "Given an N-vertex simple polygon P, subdivide its interior into triangles whose vertices are also vertices of the polygon." Correspondingly, of all possible triangulations of a simple polygon, a triangulation for which the sum of the lengths of the edges is a minimum is called a Minimum Weight Triangulation of the polygon. This problem is simpler than that of constructing a MWT of an arbitrary set of points because there are no points in the interior of the region to be triangulated.

For convenience, we consider at first the problem of constructing the MWT of the vertices of a convex polygon. For simplicity, we call such sets of points void hulls (Figure 7).

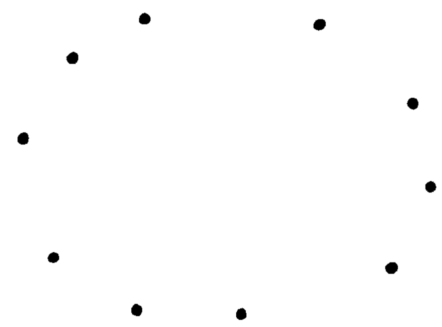


Figure 7. A void hull of 10 points.

Proposition: The MWT of a void hull of N points can be constructed in $O(N^3)$-time and $O(N^2)$-space.

Proof: The proof is algorithmic and is a dynamic programming argument. We label the points $p_0, p_1, \ldots, p_{N-1}$, ordered clockwise around the void hull, and consider the MWT of the s+1 points $p_i, p_{i+1}, \ldots, p_{i+s}$, where the addition in the subscripts is modulo N. The edge $p_i p_{i+s}$ must be in the MWT of these points, since it is on the convex hull of these points, and if s+1 is no less than 3, then $p_i p_{i+s}$ must be a side of a triangle, the third vertex of which must be one of the points $p_{i+1}, p_{i+2}, \ldots, p_{i+s-1}$ (Figure 8).

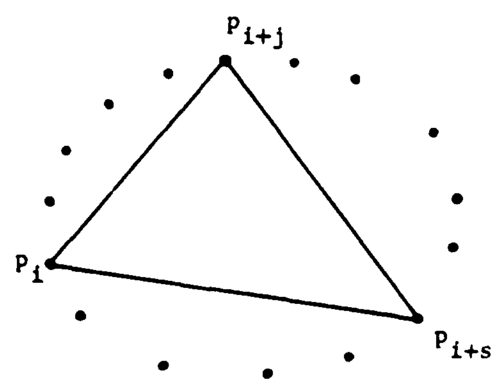


Figure 8. Illustration of the dynamic programming argument for the MWT of a void hull.

Assume that the third vertex of this triangle in the MWT is p_{i+j}, where j may range from 1 to s-1. By the optimality principle of dynamic programming [7], the triangulations of the j+1 points $p_i, p_{i+1}, \ldots, p_{i+j}$, and of the s+1-j points $p_{i+j}, \ldots, p_{i+s}$ must also be of minimum weight. Thus, if the third vertex of this triangle is p_{i+j}, then the length of the MWT of the s+1 point is the sum of

the length of the MWT of the j+1 points $p_i, p_{i+1}, \ldots, p_{i+j}$,

the length of the MWT of the s+1-j points $p_{i+j}, \ldots, p_{i+s}$,

and the length of the edge p_i, p_{i+s}.

We notice that each of these MWT problems is smaller than the (s+1)-point problem. By considering each of $j = 1,2,\ldots,s-1$ in turn, calculating the MWT length under each such value of j, and taking the minimum of these, the MWT problem for the s+1 points $p_i, p_{i+1}, \ldots, p_{i+s}$ is reduced to smaller MWT problems. Reconstruction of the MWT is facilitated by storing along with the length of this MWT, the value of j for which the minimum is achieved.

By solving all the 2-consecutive point MWT subproblems (s=1), then all the 3-consecutive point MWT subproblems (s=2), ..., and finally the N-consecutive point MWT problem, we obtain an $O(N^3)$-time and $O(N^2)$-space algorithm for the construction of the MWT of a void hull of N points.

It should be clear that this method is easily extended to any simple polygonal region; when constructing the MWT of the s+1 points $p_i, p_{i+1}, \ldots, p_{i+s}$, if the edge $p_i p_{i+s}$ is not completely within the polygon, then we simply set the MWT length of this (s+1)-point subproblem to infinity. This augmented algorithm has the *same time and space bounds*.

Unfortunately no dynamic programming approach has so far been found to yield a polynomial algorithm for more general MWTs.

We note in passing that this discussion makes clear the following recurrence on the number of ways to triangulate a void hull of N points (P_N); for any k ($2 \leq k \leq N-1$), a void hull of N-points can be triangulated in a number of ways which is the product of the numbers of triangulations of void hulls with k and N+1-k points, respectively; thus:

$$P_N = P_2 P_{N-1} + P_3 P_{N-2} + \ldots + P_{N-1} P_2, \quad N > 2, \text{ and}$$
$$P_2 = 1.$$

The solution of this recurrence [6] is

$$P_N = \binom{2N-4}{N-2} / (N-1) \approx 4^N / 16N(\pi N)^{1/2}.$$

The values of P_3 through P_7 are 1,2,5,14,42, the well known sequence of Catalan numbers.

3.3 Annular Decision Criteria

Consider a set of points in the plane, with the property that a subset of these points is enclosed in two concentric circles with radii r_1 and r_2, where $r_1 < r_2$, and there are no points in the annulus formed by the two circles. The annulus divides the plane into three regions (inner, middle and outer), and no points are in the middle region. Suppose that in the MWT of the set of points there is an edge connecting a pair of points in the outer region, passing through the inner region, and dividing the points in the inner region into two non-empty sets. As mentioned in the introduction such an edge is called a separator segment (SS). If the outer radius (r_2) were sufficiently large as compared with the inner radius (r_1), one might intuitively expect that there be no separator segment in the MWT of the full set of points (Figure 9).

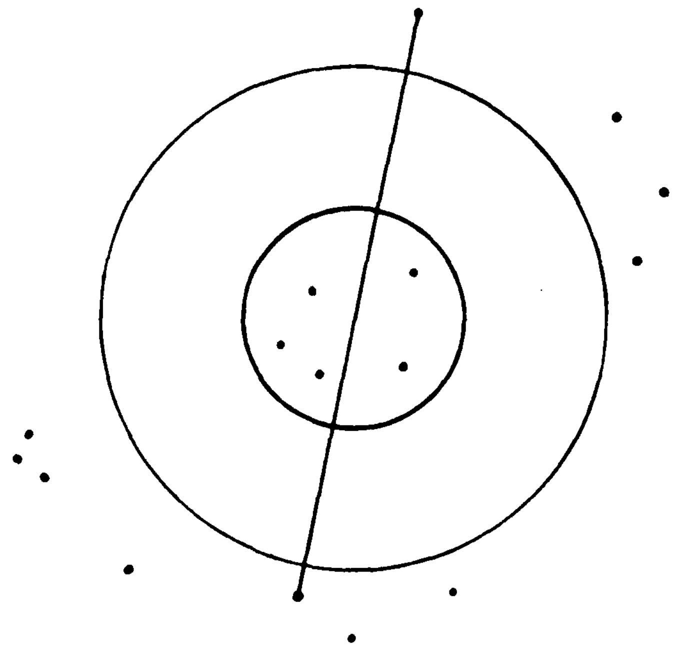
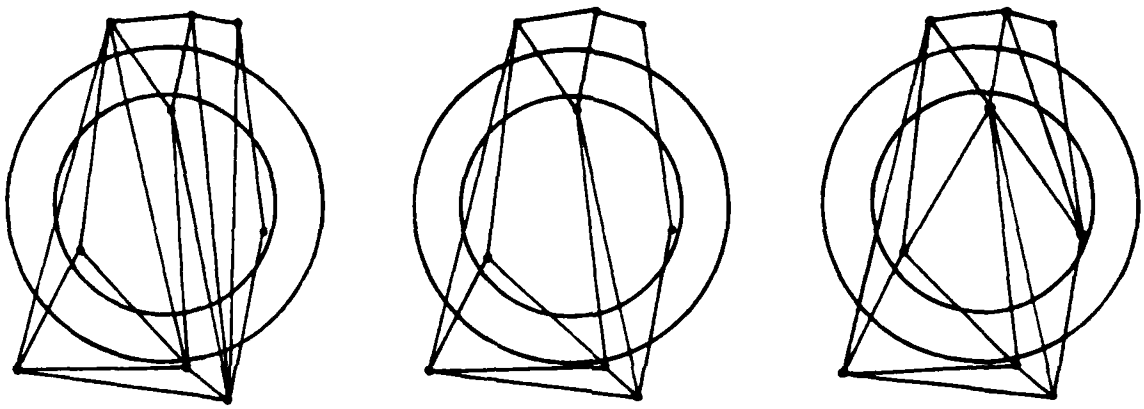


Figure 9. Example of a separator segment.

We now consider the question of how large the ratio of the radii must be in order to guarantee that there by no separator segment in the MWT. We assume without loss of generality that $r_1 = 1$ and let $r_2/r_1 \triangleq r$.

Our approach is by contradition. We assume that a given triangulation be a MWT with separator segments, and then show by construction that there is a shorter length triangulation with no separator segments. Specifically, the separator segments are removed, thereby obtaining some untriangulated polygonal regions; next, these polygonal regions are re-triangulated by inserting edges whose total length is smaller than that of the removed separator segments (Figure 10).



Example of separator segments

Separator segments removed

A shorter length triangulation with no separator segments

Figure 10. Example of Re-triangulation.

It should be clear that each of these polygonal regions may be re-triangulated individually, and that each of them contains exactly two vertices in the inner region, which we call p_A and p_B. Calling p_{A-1} and p_{A+1} the vertices adjacent to p_A on the polygon, we also note that the edge $p_{A-1}p_{A+1}$ was in the alleged MWT (i.e. $p_{A-1}p_{A+1}$ was an SS).

Some preliminary lemmas are required.

Lemma 3.4.1: If $r > \sqrt{2}$ then the length of any separator segment is greater than the length of p_Ap_B.

Proof: Clearly, length $p_Ap_B \leq 2$, and length (SS) $> 2\sqrt{r^2-1} > 2$ (Figure 11). □

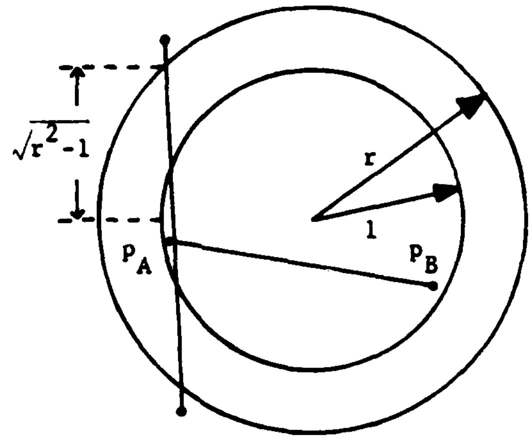


Figure 11. Length (SS) > length ($p_A p_B$) if $r > \sqrt{2}$.

<u>Lemma 3.4.2:</u> If $r \geq 5/3$, then there exists a polygon vertex p_C outside the annulus such that for any separator segment, we have length (SS) > length ($p_A p_B$), length (SS) > length ($p_B p_C$); moreover, each of the segments $p_A p_C$ and $p_B p_C$ is either an edge of the polygon or lies in its interior.

<u>Proof</u>: We select p_C as a polygon vertex outside the annulus for which the distance to the center of the annulus is a minimum. Let this distance be d (Figure 12).

From Figure 12 we see that length (SS) > $2\sqrt{d^2-1}$ for any separator segment, and that by the triangle inequality, length ($p_A p_C$) < $d+1$ and length ($p_B p_C$) < $d+1$. By simple algebra, $d > r \geq 5/3$ implies $2\sqrt{d^2-1} > d+1$, from which we have the two inequalities.

The argument that $p_A p_C$ is either an edge of the polygon or lies in its interior is also by contradiction. Assume that segment $p_A p_C$ either lies outside the polygon, or properly intersects it, and consider the path P_{AC} from p_A to p_C formed by edges of the polygon, and not passing through p_B.

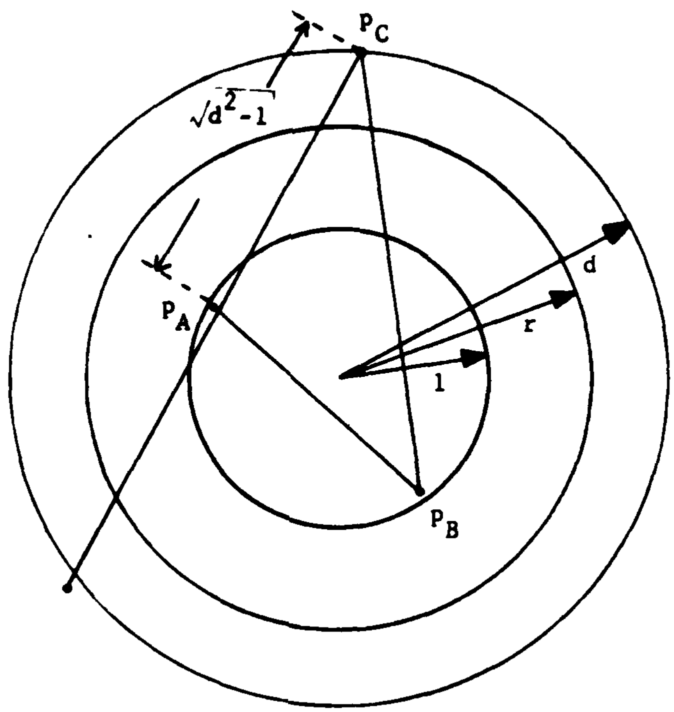
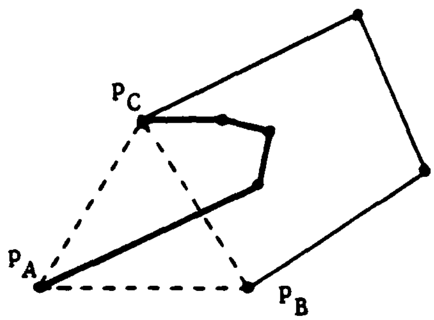
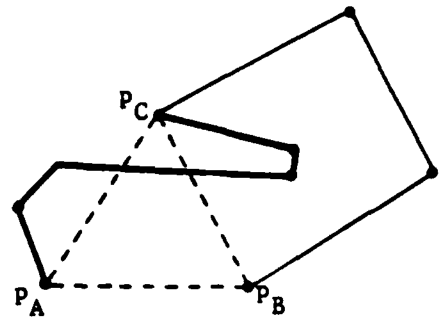


Figure 12. Selecting p_C.

Since the vertices p_A and p_B lie within the circle of radius d, and p_C lies on this circle, the triangle $p_Ap_Bp_C$ is contained in this circle. Recall that any vertex of P_{AC}, other than p_A, and p_C, must lie outside the circle of radius r. If any vertex on path P_{AC} lies within this triangle, it must also lie within the circle of radius d, and its distance to the center of the annulus is less than d. This is a contradiction, because p_C was chosen as the vertex outside the annulus and closest to the center of the annulus, and it is at distance d. Thus we are left with two cases which are typified by the following examples.



Segment p_Ap_B lies outside the polygon.



Segment p_Ap_C intersects the polygon.

Figure 13. Paths from p_A to p_C.

If p_Ap_C lies outside the polygon then the first edge of the path from p_A to p_C on the polygon must cross segment p_Bp_C. If p_Ap_C intersects the polygon, then the edge it intersects also crosses segment p_Bp_C, since no vertices are within triangle $p_Ap_Bp_C$. In either case, since p_C is an endpoint of some separator segment, this separator segment must properly intersect the polygon, but this is also a contradiction.

Therefore, segment p_Ap_C must be an edge of the polygon, or lie within the polygon, and similarly for segment p_Bp_C. $\square$

<u>Lemma 3.4.3:</u> Given a polygon $P = (p_1, p_2, \ldots, p_n)$, and a set T of the n-3 interior edges of a triangulation of P, and three distinguished vertices of the polygon, p_a, p_b, and p_c, such that neither p_a nor p_b is an endpoint of any edge in T, and p_c is any other vertex, there is a one-to-one onto mapping of the edges in T to the points $P - \{p_a, p_b, p_c\}$ such that each edge mapped to one of its endpoints (Figure 14).

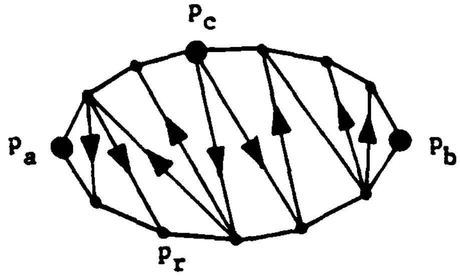


Figure 14. Mapping of edges to vertices (mapping is denoted by arrows).

<u>Proof</u>: (By induction). The lemma is trivial for $n = 4$ (triangulation of a quadralateral). Thus, assume the lemma valid for all $k < n$. Let t_i be the number of edges in T which have p_i as an endpoint. Then

$$\sum_{p_i \in P} t_i = 2|T| = 2n-6, \qquad t_a = t_b = 0,$$

and, by the pigeon-hole property, at least two of the n-2 points $P-\{p_a, p_b\}$ must have $t_i = 1$. Choose one of these which is not p_c, call it p_r, and let $p_r p_s$ be the unique edge of T incident to p_r. Edge $p_r p_s$ is mapped to p_r.

Consider now the polygon p', whose vertex set is $P-\{p_r\}$: obviously $T' = T-\{p_r p_s\}$ is a triangulation of P'. Since $|P'| = n-1 < n$, then by the inductive hypothesis, with the same distinguished vertices p_a, p_b, and p_c as before, there is a one-to-one onto mapping of the edges of T' to the vertices $P'-\{p_a, p_b, p_c\}$, with each edge being mapped to one of its endpoints, and the lemma is proved. ☐

We can now describe the re-triangulation of a polygon P. Assuming $r \geq 5/3$, we find a vertex p_C as in lemma 3.4.2, and construct a one-to-one correspondence of the interior edges of the alleged MWT to the vertices

$P-\{p_A, p_B, p_C\}$, as in lemma 3.4.3. Thus, any separator segment (interior edge) is longer than each of p_Ap_B, p_Bp_C, and p_Ap_C, by lemma 3.4.2, and these three edges divide the original polygonal region into three smaller polygonal regions and the triangle $p_Ap_Bp_C$.

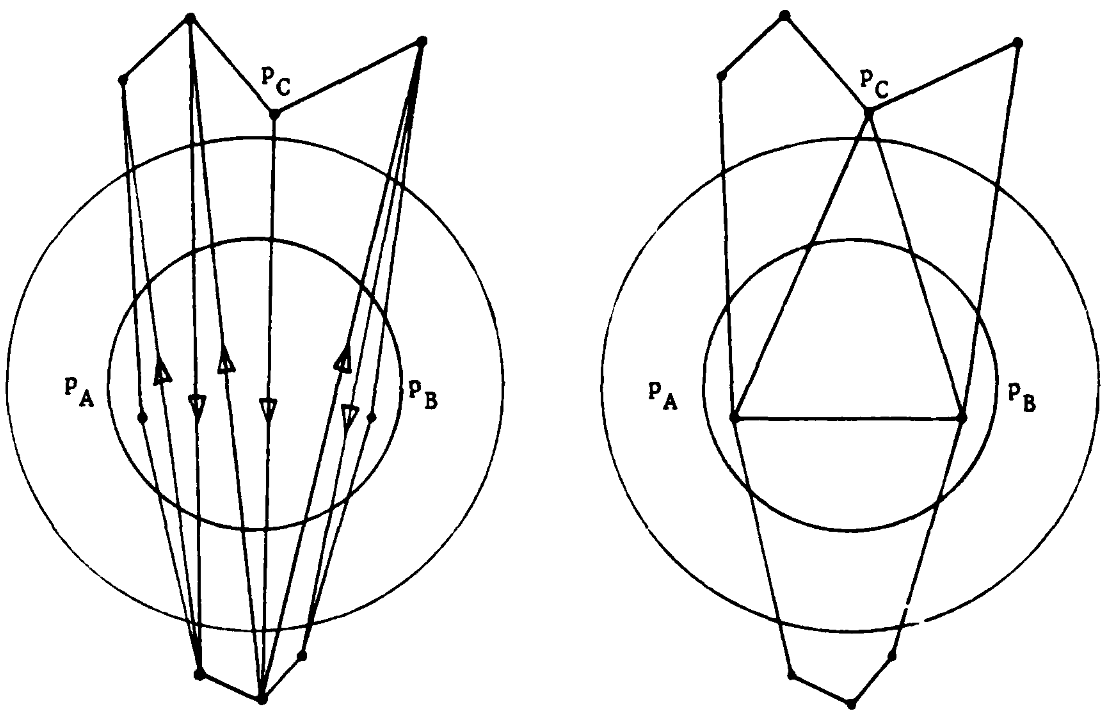


Figure 15. Allocation of edges and division of polygon.

Consider, for example, re-triangulating the smaller polygonal region which has p_Ap_C as an edge (the other two polygonal regions may be handled similarly), and let t be the number of vertices of this polygon. If this polygon is degenerate, i.e. it consists of the single edge p_Ap_C, then no re-triangulation occurs: assume therefore that the polygon be nondegenerate and consider three consecutive vertices p_{j-1}, p_j, p_{j+1} on this smaller polygon such tha $p_j \notin \{p_A, p_C\}$. Then p_j must have an assigned edge. If the length of the segment $p_{j-1}p_{j+1}$ is less than the length of the edge assigned to p_j, and $p_{j-1}p_{j+1}$ lies within this smaller polygon, then $p_{j-1}p_{j+1}$ may be added to the new triangulation, vertex p_j and its associated edge may be removed from further consideration, and the process may be continued. As the final step, edge $p_A p_C$, which is shorter than any separator segment, is added to the new triangulation. The problem is to find such points p_j.

Figure 16. The separator segment assigned to p_j is replaced by a shorter edge.

We now show that if $r \geq 2$, then a vertex of the original polygon for which the distance d to the center of the annulus is a maximum may be chosen as a suitable p_j.

The edge $p_{j-1}p_{j+1}$ lies within the polygon, since the vertices p_{j-1} and p_{j+1} must lie within the circle of radius d and centered at the center of the annulus, and the angle $p_{j-1}p_jp_{j+1}$ is less than 180°. We now show that the length of $p_{j-1}p_{j+1}$ is less than the length of the edge assigned to p_j.

The edge assigned to print p_j must pass through the innermost circle (of radius 1), thus its length must be greater than or equal to $\sqrt{d^2-1} + \sqrt{3}$ (Figure 17). By our choice of p_j, points p_{j-1} and p_{j+1} must lie within the circle of radius d and centered at the center of the annulus. We now distinguish three cases.

(1) If both these points lies within the circle of radius 1, then length $(p_{j-1}p_{j+1}) \leq 2 < 2\sqrt{3} \leq \sqrt{d^2-1} + \sqrt{3}$, since $d \geq 2$.

(2) If one of these points lies within the circle of radius 1, then by the triangle inequality length $(p_{j-1}p_{j+1}) \leq d+1$, and $d \geq 2$ implies that $d+1 < \sqrt{d^2-1} + \sqrt{3}$.

(3) If both p_{j-1} and p_{j+1} lie outside the circle of radius 1, they must also lie outside the circle of radius 2 (because the annulus is by hypothesis empty). In Figure 17 we construct the line $p_{j-1}q$ tangent to the circle of radius 1, with p_j and the center of the annulus on the same side of this line, and with q on the circle of radius 2 so that the tangency point t is between p_{j-1} and q. The separator segment assigned to the other endpoint of p_{j-1} must lie within the half-plane delimited by this line which includes the center of the annulus. Constructing the other tangent through q to the circle of radius 1, we claim that p_{j+1} must lie within the wedge formed by these two tangents and including the center of the annulus (Fig. 18), indeed, by constructing the tangent $p_{j+1}q'$, as $p_{j-1}q$ was constructed, the separator segments assigned to p_{j-1} and p_{j+1} would properly intersect, which is impossible. Thus the largest value of the distance between p_{j-1} and p_{j+1} occurs when they are on the circle with radius d, i.e. when $p_{j-1} \equiv t_1$ and $p_{j+1} \equiv t_2$. By noting that the angle $p_{j-1}\widehat{q}\,p_{j+1}$ is

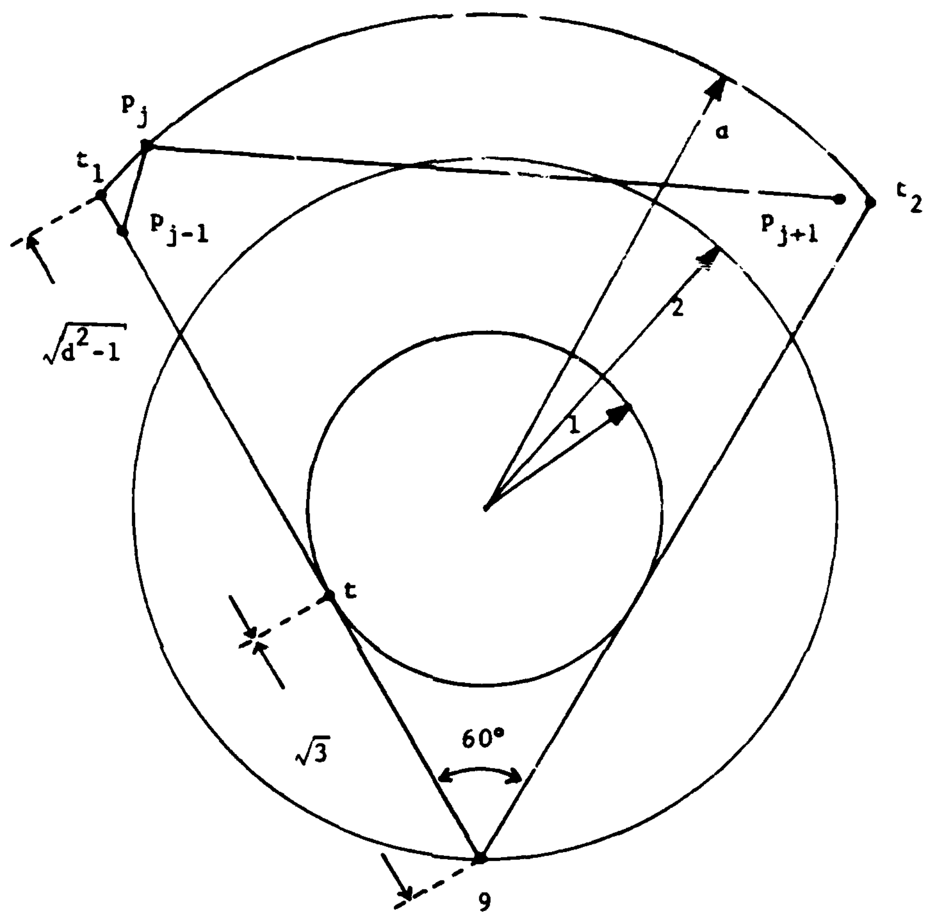


Figure 17. Length (separator segment) > length $(p_{j-1}p_{j+1})$.

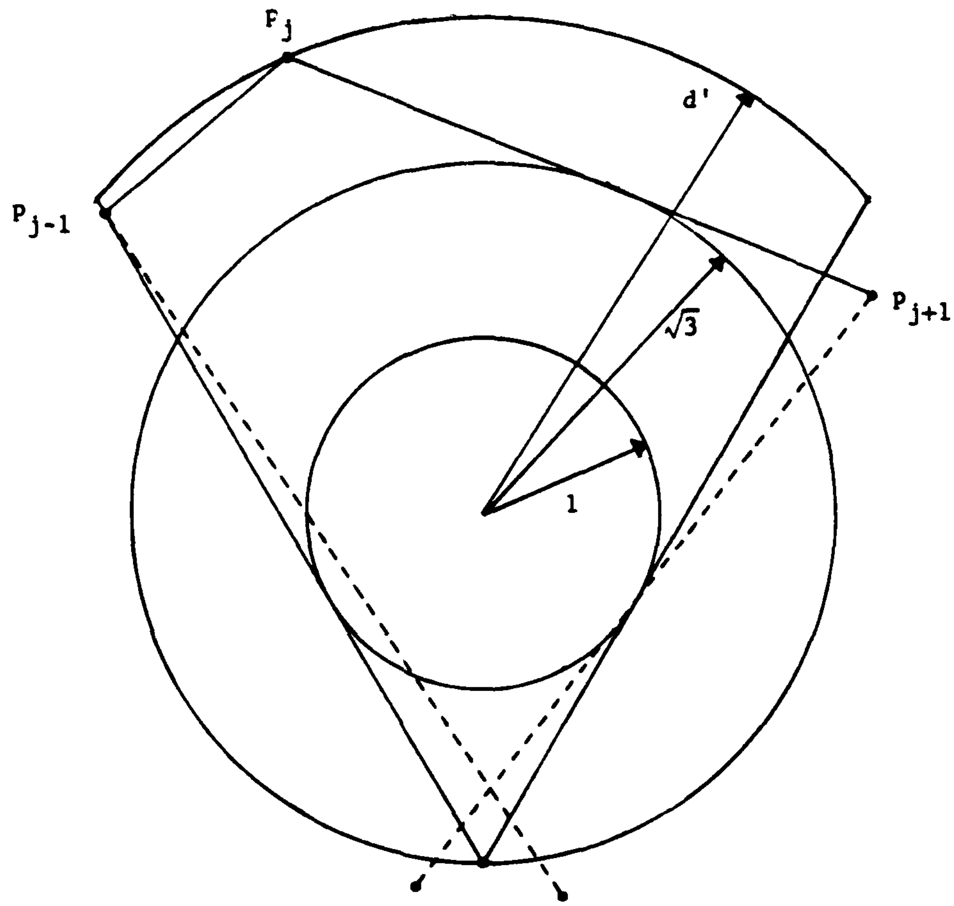


Figure 18. The edges assigned to p_{j-1} and p_{j+1} properly intersect.

60° and the triangle $t_1 q t_2$ is equilateral, length $(p_{j-1}p_{j+1}) \leq \sqrt{d^2-1} + \sqrt{3} \leq$ length (separator segment assigned to p_j).

Thus, regardless of the positions of vertices p_{j-1} and p_{j+1}, length $(p_{j-1}p_{j+1}) \leq$ length (separator segment assigned to p_j), and the vertex on the polygon farthest from the center of the annulus is a suitable p_j.

We include edge $p_{j-1}p_{j+1}$ in the new triangulation, no longer consider vertex p_j and its associated separator segment, and find another suitable point (the next-farthest vertex). Continuing this process, the original polygon is re-triangulated using a shorter total length, which implies that the alleged triangulation was not a MWT, and we may state the following theorems.

Given a set of points in the plane V, and a subset S of V with the properties that the points of S may be enclosed by a circle C_1 of radius r_1 and the points of V-S are outside the circle C_2 concentric with C_1 and with radius $r_2 > r_1$, we have:

Theorem 3.4.4: If $r_2/r_1 \geq 2$, then in the MWT of V there is no separator segment connecting vertices of V-S, passing through the inner circle, and dividing the vertices of S into two nonempty sets.

If two of the vertices in the inner region are diametrically opposite on the inner circle, then a similar analysis can be done, with the ratio of radii reduced to $\sqrt{3}$. Figure 19 corresponds to Figure 17 of the previous analysis. Figure 20 illustrates the case in which the separator segment does not pass through these two vertices.

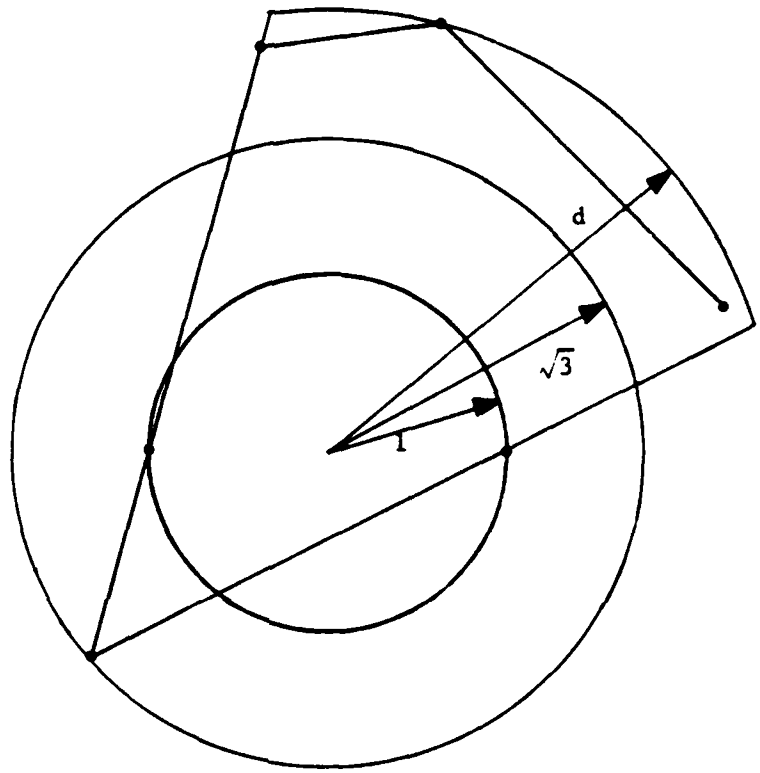


Figure 19. Theorem 3.4.5.

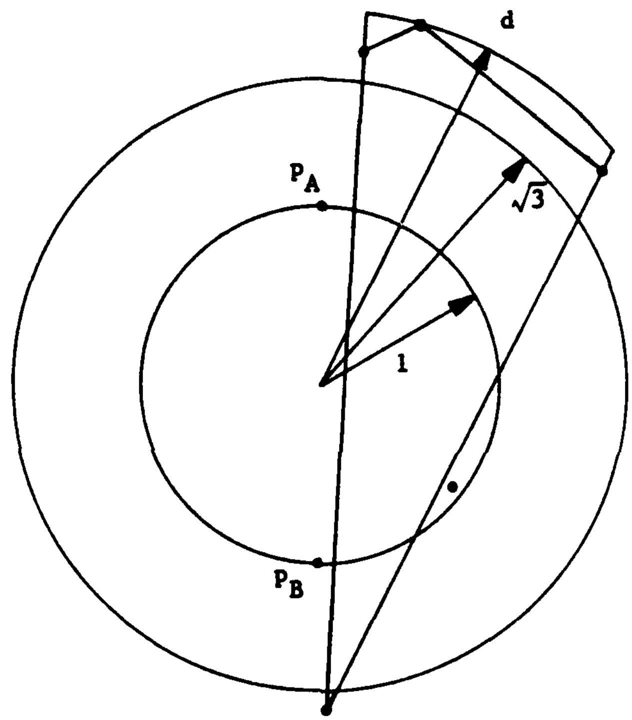


Figure 20. Separator segments not passing between p_A and p_B.

Theorem 3.4.5: If $r_2/r_1 \geq \sqrt{3}$ and two vertices of S are on a diameter of the inner circle, and on the inner circle then in the MWT of V there is no separator segment connecting vertices of V-S, passing through the inner circle and dividing the vertices of S into two nonempty sets. XXX

As a side-result, we have:

Corollary 3.4.6: The shortest edge is in a MWT of any set of points.

Proof: Let $p_s p_t$ be the shortest edge, and let 2d be its length, as in Figure 21. Since $p_s p_t$ is shortest, this implies that there are no other vertices in the circle centered at p_s with radius 2d, and similarly for the circle of radius 2d centered at p_t. The annulus of theorem 3.4.5 fits within the union of these two excluded regions. Since there can be no separator segment through $p_s p_t$, by theorem 3.4.5, edge $p_s p_t$, the shortest edge, must be in a MWT.

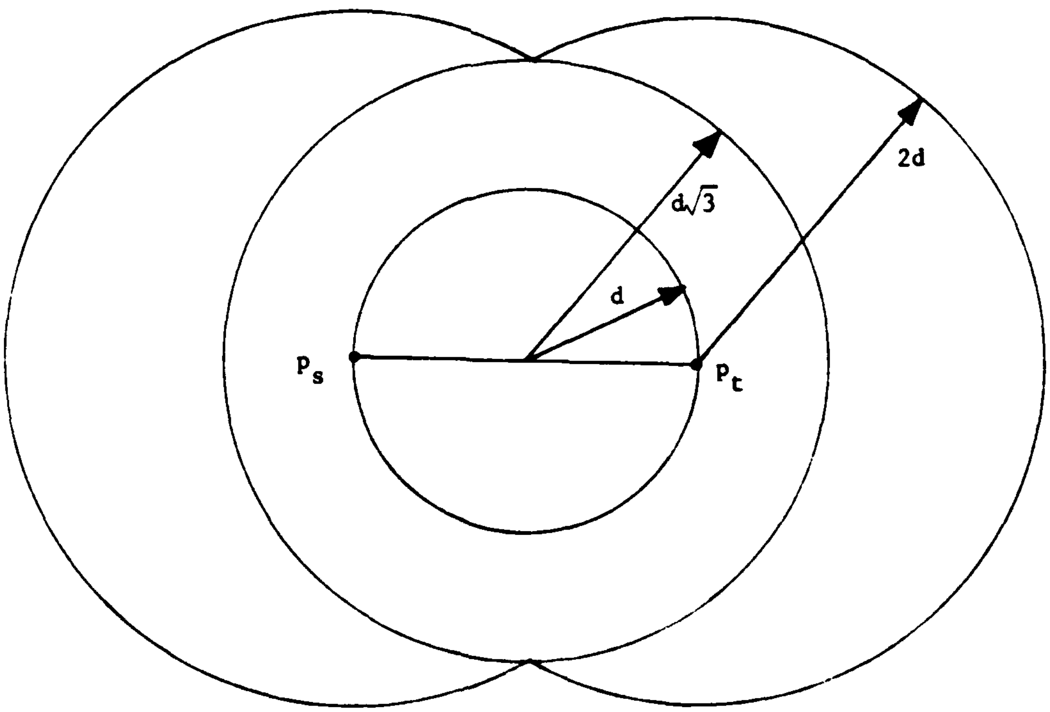


Figure 21. The shortest edge is in a MWT.

3.4 Delaunay Triangulation vs MWT

The Greedy Triangulation has been shown to be a poor approximation to the Minimum Weight Triangulation, in that the ratio of the total length of the Greedy Triangulation of a set of N points to that of the MWT can be as large as $O(N^{1/3})$ [5]. The Delaunay Triangulation (DT) is the straight-line dual of the Voronoi diagram [1]; we present now an example for which the ratio of the length of the DT to that of the MWT of a set of N points is roughly N/2, thus showing that the Delaunay Triangulation is not "approximately optimal".

In Figure 22, five points are closely packed on an arc of a circle with radius r, and a sixth point is at the center of the circle. The Voronoi diagram and Delanunay Triangulation are shown as dashed and solid lines, respectively. If the five points were so closely packed that the distance between the furthest two were negligible as compared to r, then the total length of the triangulation would be roughly 5r. In general, for N points, the total length could be arbitrarily close to (N-1)r.

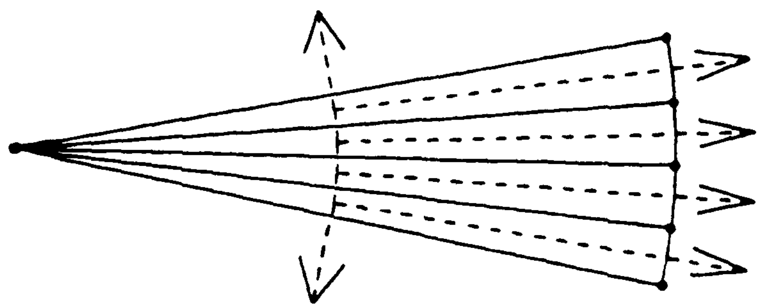


Figure 22. Voronoi diagram (dashed lines), and Delaunay triangulation (solid lines).

Figure 23 illustrates the Minimum Weight Triangulation for the same set of points. With the points on the arc closely packed, the total length can be arbitrarily close to 2r, regardless of the total number of points.

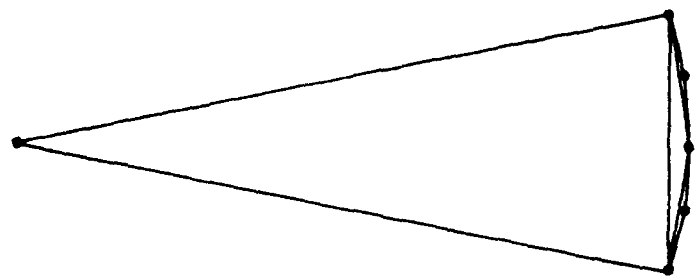


Figure 23. Minimum Weight Triangulation.

Thus, the preceding illustrates a construction for which the total length ratio between DT and MWT can be as large as roughly (N-1)/2, for the $N \geq 3$, and the Delaunay Triangulation need not be a good approximation to the Minimum Weight Triangulation.

REFERENCES

1. M. I. Shamos, Computational Geometry. Department of Computer Science, Yale University (1977). To be published by Springer Verlag.

2. M. I. Shamos and D. Hoey, "Closest Point Problems," Proceedings of the Sixteenth Annual Symposium on the Foundations of Computer Science. IEEE (1975), 151-162.

3. E. L. Lloyd, "On Triangulations of a Set of Points in a Plane," Eighteenth Annual IEEE Conference, Proceedings on the Foundations of Computer Science. IEEE (1977), 228-240.

4. R. D. Duppe and H. J. Gottschalk, "Automatische Interpolation von Isolinien bei willkurlich verteilten Stutzpunkten," Allgemeine Vermessungsnachrichten, 77 (1970), 423-426.

5. G. K. Manacher and A. L. Zobrist, "A Fast, Space-Efficient Average-case Algorithm for the 'Greedy' Triangulation of a Point Set, and a Proof that the Greedy Triangulation is not approximately optimal," Proceedings of the Sixteenth Annual Allerton Conference on Communication, Control, and Computing. Allerton, Illinois, 1978.

6. D. E. Knuth, The Art of Computer Programming, Volume 1, Addison-Wesley (1972), 388-389.

7. R. E. Bellman and S. E. Dreyfus, Applied Dynamic Programming, Princeton University Press (1962).

8. J. L. Bentley and H. A. Maurer, "A note on Euclidean near neighbor searching in the plane," submitted to Information Processing Letters (1978).

9. F. P. Preparata, "A New Approach to Planar Point Location," submitted for publication.

10. J. L. Bentley, unpublished.