MIT/LCS/TM-88

ON TRIANGULATIONS OF A SET OF POINTS IN THE PLANE

Errol Lynn Lloyd

July 1977

MIT/LCS/TM-88

ON TRIANGULATIONS OF A SET OF POINTS IN THE PLANE

Errol Lynn Lloyd

MASSACHUSETTS INSTITUTE OF TECHNOLOGY
LABORATORY FOR COMPUTER SCIENCE
(formerly Project MAC)

CAMBRIDGE MASSACHUSETTS 02139

ON TRIANGULATIONS OF A SET OF POINTS IN THE PLANE

by

Errol Lynn Lloyd

Submitted to the Department of Electrical Engineering and Computer Science

on May 12, 1977 in partial fulfillment of the requirements

for the Degree of Master of Science

ABSTRACT

A set, V, of points in the plane is triangulated by a subset, T, of the straight line segments whose endpoints are in V, if T is a maximal subset such that the line segments in T intersect only at their endpoints. The weight of any triangulation is the sum of the Euclidean lengths of the line segments in the triangulation. We examine two problems involving triangulations. We discuss several aspects of the problem of finding a minimum weight triangulation among all triangulations of a set of points and give counterexamples to two published solutions to this problem. Secondly, we show that the problem of determining the existence of a triangulation in a given subset of the straight line segments whose endpoints are in V is NP-Complete.

Thesis Supervisor: Ronald L. Rivest
Title: Assistant Professor of Computer Science and Engineering

Acknowledgements

I would like to thank my thesis advisor, Ron Rivest, for suggesting the topic and for his guidance throughout the course of the research. Also, I would like to thank Ernst Mayr for his assistance in translating from German. Paul Bayer has been especially helpful in reading and commenting on an earlier version of this thesis.

Finally, I would like to thank my wife Isabel for her love.

This research was supported by National Science Foundation Grants MCS76-14294 and MCS74-12997-A04.

TABLE OF CONTENTS

Chapter 1 - Introduction

1.1 Geometric Complexity

Computational geometry problems frequently arise in many real-world and theoretical circumstances. Solutions to many of these problems have been known for centuries. Only recently, however, have the time and space complexities of geometric problems begun to be examined. A large portion of this work has been done by M. Shamos [12,13,14], who has given efficient algorithms for a number of the fundamental geometric problems.

The complexity of geometric problems is important not only because of the real nature of many of the problems, but also because of the insights provided on the intrinsic nature of computation. For instance, consider the problem of finding the minimum weight spanning tree of a set of points in the Euclidean plane and the corresponding graph-theoretic problem of finding a minimum weight spanning tree in an arbitrary graph. It has been shown that the geometric problem, for n points in the plane, can be solved in time $\mathcal{O}(n \log n)$, whereas the best algorithm presently known for the graph-theoretic version requires time $\mathcal{O}(n^2)$ for a graph with n vertices [1,12]. This suggests that the algebraic and geometric versions of a problem may have substantially different complexities. In contrast, we note that both the algebraic and geometric versions of the Traveling Salesperson Problem and the Steiner Tree Problem have been shown to be NP-Complete [7,8,9,11]. At this time there remains a large amount of mystery about what geometry contributes to a problem in terms of its complexity. Due to the recent emergence of the field there is a large class of problems which remain open. The primary concern of this thesis will be with several related problems in geometric complexity.

1.2 The Triangulation Concept

The concept of a set of points in the plane being triangulated may be formulated as follows: Let V be a set of n distinct points in the plane. We assume that these points are not all collinear and that $n \geq 3$. The points in V will be called vertices. Let L be the set of $\binom{n}{2}$ straight line segments between vertices in V. The elements of L are edges. Two edges, e_1 and e_2 properly intersect if e_1 is not equal to e_2 and if e_1 and e_2 intersect at a 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. There are several useful properties of triangulations which follow directly from this definition:

1. Each edge in the convex hull of V is in T.
2. Each interior face of the straight-line planar graph, as determined by V and T, is a triangle.
3. Each edge in L is either in T or properly intersects an edge in T.
4. If y_1y_2 is an edge in T with endpoints y_1 and y_2 in V and y_1y_2 is not an edge on the convex hull of V, then in each half-plane as determined by a line passing through y_1 and y_2, there must exist a vertex w in V such that edges y_1w and y_2w are in T and there does not exist a fourth vertex u in V which lies on, or interior to, triangle y_1y_2w.

We will make use of these properties throughout this paper.

Triangulations have an application in the approximation of function values for a function of two variables when the value of the function is known at some number of arbitrary points. One method involves finding a triangulation of the set of points where the function values are known [5]. To approximate the value of the function at another point, say p, we find the triangle in which p lies with respect to the triangulation and then approximate the function value at p by linear interpolation of the function values at the vertices of the triangle in which p lies.

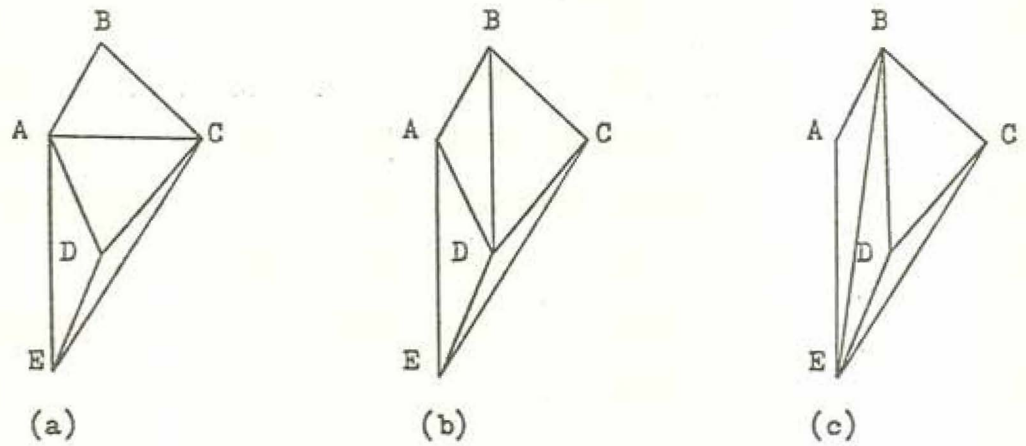
The remaining two sections of this chapter describe the two problems concerning triangulations in which we are interested.

1.3 The Minimum Weight Triangulation Problem

The minimum weight triangulation problem is as follows: Given a set of points in the plane, V, and the set of edges, L, whose endpoints are in V, a weight can be assigned to each edge in L, the weight of an edge being equal to the Euclidean distance between its endpoints. The weight of a triangulation, T, is then defined to be the sum of the weights of all of the edges in T. We are interested in discovering an efficient algorithm for finding a triangulation of minimum weight among all of the triangulations of V. This problem will be referred to as MWT throughout this paper.

An example is shown in Figure 1. There are five vertices to be triangulated. The three triangulations shown are the only triangulations of those vertices. The minimum weight triangulation is given in 1a, because $|AC| + |AD| < |BD| + |AD|$ and $|AC| + |AD| < |BD| + |BE|$, and the three triangulations agree on all other edges.

Figure 1: Three triangulations of a set of points in the plane.



The minimum weight triangulation problem has been studied previously by Duppe and Gottschalk [6] and Shamos and Hoey [12]. We note that other criteria for the "goodness" of a triangulation might be better suited to certain applications and may be easier to find. Criteria concerning the size of the maximum or minimum angles in a triangulation and how they apply to the finite element method are discussed by Babuska and Aziz [2] and Bramble and Zlamal [4].

In chapter 3 we present counterexamples to two algorithms proposed for solving MWT and give counterexamples to several conjectures concerning minimum weight triangulations. A discussion of a dynamic programming approach to this problem is also presented.

1.4 The Triangulation Existence Problem

In this problem we are concerned with determining when a triangulation of V exists in some given subset of L. That is, given a set of vertices, V, and a subset E of L, does there exist a subset T of E such that T is a triangulation of V? This problem will be referred to as TRI.

An efficient algorithm for solving this problem might be useful in attacking other problems involving triangulations. For instance, in our work on MWT, we considered a matroid approach to the problem. A desirable property was to be able to tell efficiently if a subset of L contained a triangulation of V. It appears reasonable that other applications of triangulations may also have cause to use such an algorithm.

In chapter 4, we show that TRI is NP-Complete, hence it is probable that it is not possible to find an efficient algorithm for determining triangulation existence.

Chapter 2 - Preliminaries

2.1 NP-Completeness

A recurring theme throughout this paper is the notion of a problem being NP-Complete. We will give an informal discussion of this subject here and refer the reader to Aho, Hopcroft and Ullman [1] for specifics.

The set NP is defined to consist of all languages which can be recognized by a nondeterministic Turing machine of polynomial time complexity. Similarly, the set P consists of all languages which can be recognized by a deterministic Turing machine of polynomial time complexity. It is not presently known if P is properly contained in NP.

A language M_0 in NP is defined to be NP-Complete if the following condition is satisfied: If we have a deterministic algorithm for recognizing M_0 of time complexity $T(n) \geq n$, then for each language M in NP, we can effectively find a deterministic algorithm for recognizing M of time complexity $T(p(n))$ where p is a polynomial depending on M [1]. Thus, if any NP-Complete language is in P then the sets P and NP are equal.

A language M over alphabet Δ is polynomially reducible to a language M_0 over alphabet Σ if there is a deterministic algorithm which, when given a string w over Δ produces a string w_0 over Σ in time polynomial in the length of w such that w is in M if and only if w_0 is in M_0.

The method that we will use to show that a language M_0 is NP-Complete is to show that:

1. M_0 is in NP
2. There exists a language M which is NP-Complete which is polynomially reducible to M_0.

A language for which the second condition can be shown, but not necessarily the first, is said to be NP-Hard. A large number of combinatorial and optimization problems have been shown to be NP-Complete [1,8].

2.2 Definitions

A brief description of Voronoi diagrams is given here. The interested reader is directed to Shamos [12,13,14] and Rogers [10] for formal definitions and results. Consider a finite set of vertices, V, in the plane. Surrounding each vertex, w, there is a maximal convex polygon called the Voronoi polygon associated with w. This polygon is defined to consist of each point, p, in the plane such that no vertex of V is closer to p than is w. The Voronoi polygons for each vertex in V partition the plane, forming a network of convex polygons called the Voronoi diagram of V. The straight-line dual of the Voronoi diagram of V is a planar graph with vertices V, where a line segment (an edge) exists between two vertices if and only if the Voronoi polygons of those two vertices share an edge. We will refer to these concepts in chapter 3.

There are several other useful definitions.

Suppose V is a set of vertices and T is a set of edges whose endpoints are in V. A path Q in T is defined to be a list of vertices of V, $(p_1, p_2, \ldots, p_k)$ such that each edge $p_i p_{i+1}$ of the path, for $1 \le i \le k-1$, is in T. A circuit is a path in which $k > 3$ and p_1 and p_k are the same. An elementary circuit is a circuit in which each of the vertices $p_1, p_2, \ldots, p_{k-1}$ is distinct.

2.3 Notations

Throughout this paper either of the notations AB or [A, B] will be used to refer to an edge whose endpoints are vertices A and B. Which notation is used will depend upon which is clearer in the given situation.

The coordinates of a point in the plane will be enclosed in parentheses. For example, the origin is (0,0).

When applicable a set of edges between vertices of V will be denoted as follows: If Q and R are sets of vertices of V, then $Q \times R$ represents the set of all edges [q, r] such that q is in Q and r is in R.

The symbol V denotes a set of vertices to be triangulated throughout this paper.

Chapter 3 - The MWT Problem

This chapter provides counterexamples to two algorithms conjectured to solve MWT. These lead to several observations about MWT. A dynamic programming approach to MWT is discussed. Throughout this chapter L is the set of all edges whose endpoints are in V.

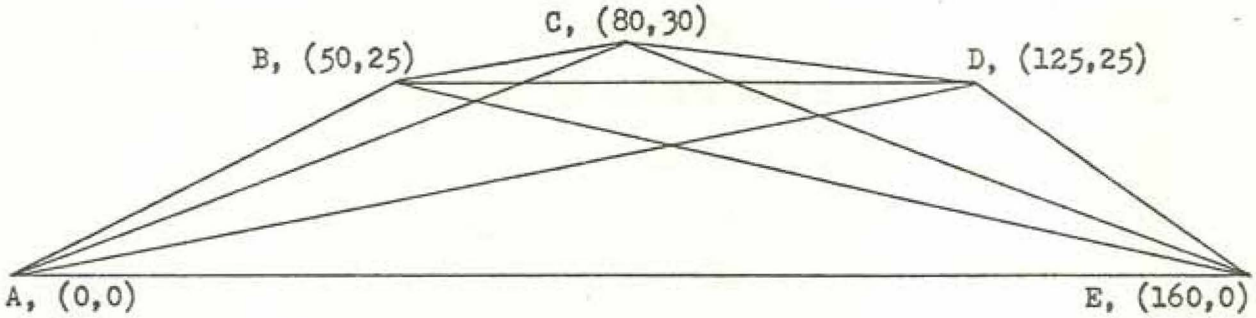
3.1 The Duppe-Gottschalk Algorithm

The first of the algorithms purported to solve MWT was published by Duppe and Gottschalk [6]. Unfortunately, their paper was written in a very informal manner and the explanation they give of their algorithm is ambiguous. For these reasons we have two different versions of their algorithm. The first version is as follows:

1. Set $L_0 \leftarrow L$, $T_0 \leftarrow \emptyset$ and $i \leftarrow 0$.
2. While $L_i \neq \emptyset$ do
   21. Let w be an edge of least weight in L_i
   22. $T_{i+1} \leftarrow T_i \cup \{w\}$
   23. $L_{i+1} \leftarrow L_i - \{w\} - \{m \in L_i \mid m \text{ and } w \text{ properly intersect}\}$
   24. $i \leftarrow i + 1$
3. $T \leftarrow T_i$

The claim is that T is a minimum weight triangulation of V. In Figure 2 we give a set of vertices which shows that the triangulation produced is not necessarily a minimum weight triangulation. In that example, we are concerned only with the edges not on the convex hull of the vertices since the convex hull is in every triangulation. The edges not on the convex hull in the triangulation produced by this algorithm are BD and BE which have a combined weight over 187 units. However, the interior edges in the

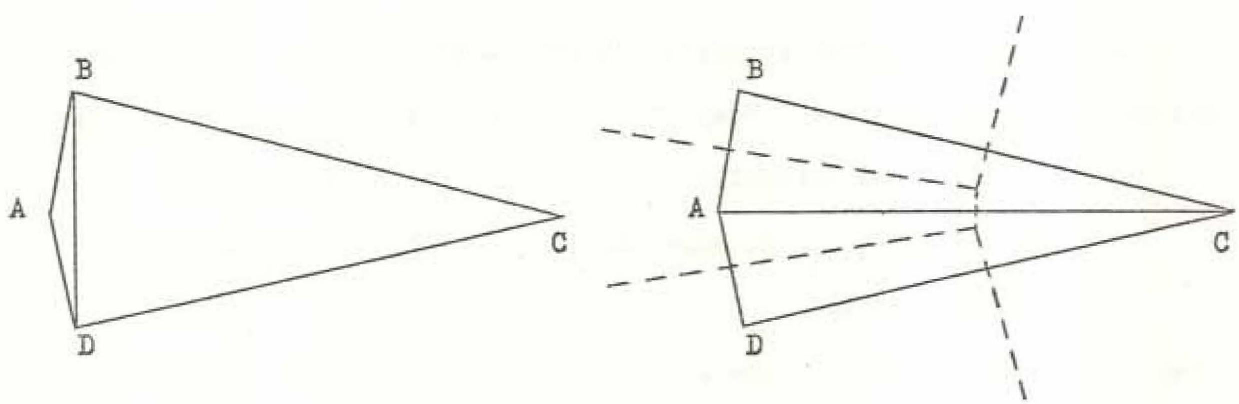
Figure 2: Counterexample to the Duppe-Gottschalk Algorithm.



Edge lengths of interior edges (relative to the given coordinates)

Edge BD: 75 units
Edge CE: < 86 units
Edge AC: < 86 units
Edge BE: > 112 units
Edge AD: > 127 units

Figure 3: Counterexample to the Shamos-Hoey algorithm.



(a) (b)

minimum weight triangulation are CE and AC which have a combined weight of under 172 units. This algorithm was independently proposed by R. Rivest.

The second version of their algorithm is simply a modification of the first version. The following statements are added to the algorithm given above between statements 23 and 24:

231. Let y be an edge of least weight in L_{i+1} which has a common endpoint with w. If no such y exists then jump to step 24.

232. $L_{i+1} \leftarrow L_{i+1} - \{m \in L_{i+1} \mid$ m and y properly intersect and the weight of m exceeds the weight of y $\}$

The example in Figure 2 is also a counterexample to this version. In that example the version two algorithm produces the same triangulation as the first version. It may be that the two versions are equivalent, which would remove much of the ambiguity from the Duppe-Gottschalk paper. However, such an equivalence was not apparent to us. In any case, neither version of the algorithm always produces a minimum weight triangulation.

3.2 The Shamos-Hoey Algorithm

The second algorithm purported to find minimum weight triangulations was published by Shamos and Hoey [12]. Their algorithm is as follows:

1. Construct the Voronoi diagram for the set of vertices V.
2. Let T consist of the edges in the straight-line dual of the Voronoi diagram.

They claim that T is a minimum weight triangulation of V. We note that if more than three Voronoi edges meet in a single point then the dual of the Voronoi diagram is not a triangulation, but only a network of convex polygons which must then be triangulated by another method. Ignoring this detail, we note the correctness of this algorithm, as far as producing a minimum weight triangulation is concerned, is partially based on the work of Duppe and Gottschalk. That this algorithm does not always produce a minimum weight triangulation is shown in Figure 3. The minimum weight triangulation is shown in 3a and the triangulation produced by the Shamos-Hoey algorithm is given in 3b. The Voronoi edges are given as broken lines in 3b. This example also shows that the Shamos-Hoey algorithm is not equivalent to the Duppe-Gottschalk algorithm since, in this example, the Duppe-Gottschalk algorithm does produce the minimum weight triangulation. Such an equivalence was implied in the paper by Shamos and Hoey [12].

As an interesting observation, we note that Shamos and Hoey give an $\mathcal{O}(n \log n)$ lower bound for finding any triangulation of a set of n points in the plane [12]. This bound follows from the reduction of a one-dimensional sorting problem to the problem of finding any triangulation of a given set of points. The Shamos-Hoey algorithm, slightly modified to handle the case of greater than three Voronoi edges meeting at a single point, achieves this lower bound.

3.3 Observations about MWT

There are several observations about minimum weight triangulations which follow from the counterexamples for the two proposed algorithms.

The first observation is that the shortest edge not on the convex hull of V is not always in a minimum weight triangulation of V. The counterexample to the Duppe-Gottschalk algorithm shows this. In that example - Figure 2 - edge BD is the shortest edge not on the convex hull and it is not in the minimum weight triangulation.

A second observation is that a minimum weight triangulation does not always contain a minimum weight spanning tree. The example in Figure 4 illustrates this. The minimum weight triangulation of the four vertices is given in 4a and the minimum weight spanning tree is given in 4b. We note that this observation alone is sufficient to show that the Shamos-Hoey algorithm does not always produce a minimum weight triangulation since a minimum weight spanning tree of a set of vertices is always a subgraph in the dual of the Voronoi diagram of those vertices [12].

In addition to these observations we had conjectured that every triangulation contains a Hamiltonian circuit and in fact, that every minimum weight triangulation contains a minimum weight Hamiltonian circuit. However, in Figure 5, we give a minimum weight triangulation which does not even include a Hamiltonian circuit, much less one of minimum weight.

3.4 The Dynamic Programming Approach

One possible algorithmic approach to finding minimum weight triangulations is dynamic programming [3]. We have examined this possibility in some detail. This section discusses such an approach and what we perceive to be the major difficulty with it in terms of obtaining a polynomial time algorithm.

Before proceeding, we need to develop the notion of a restricted minimum weight triangulation. Consider a planar region, R, which is bordered by an elementary circuit whose edges are in L. We require that no two edges of this circuit properly intersect. R need not be a convex

Figure 4: The minimum weight spanning tree is not always in the minimum weight triangulation.

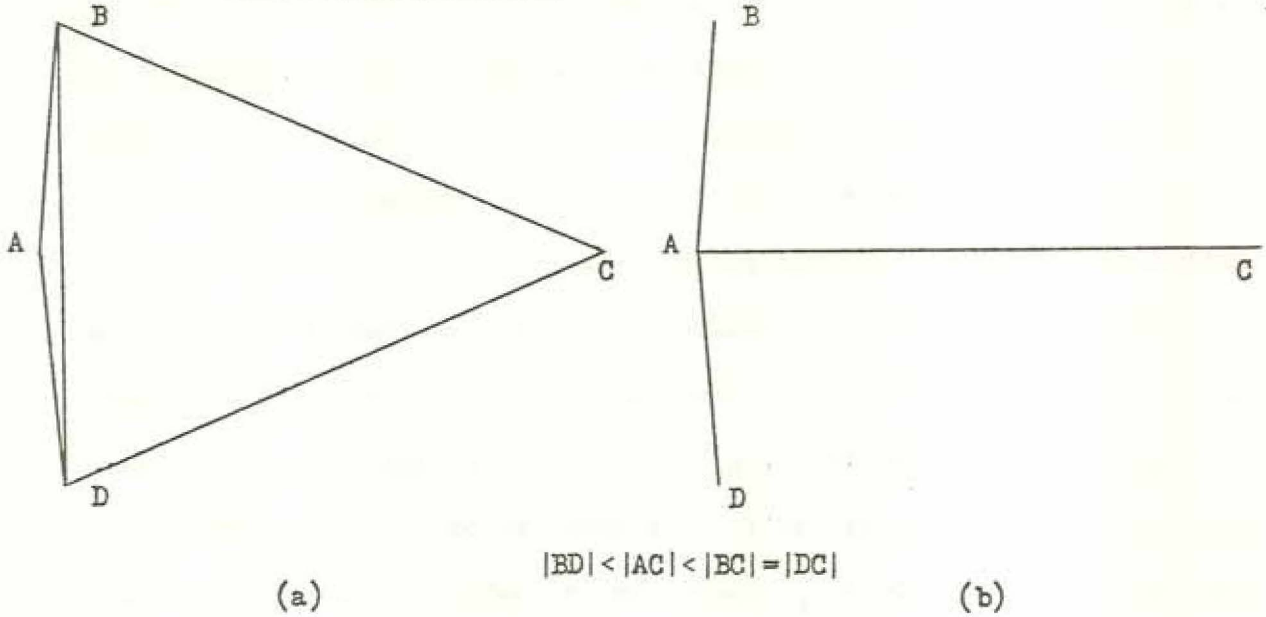
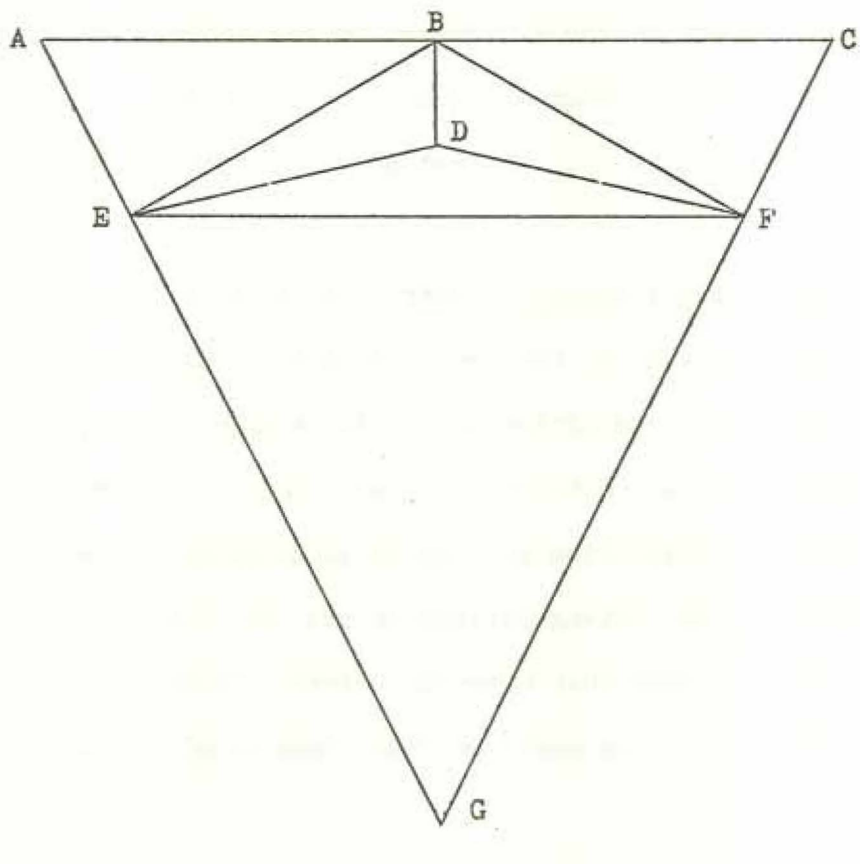


Figure 5: The minimum weight triangulation does not always contain a Hamiltonian circuit.



region. Let V_R be the set of vertices of V that lie in R and let L_R be the set of edges of L which lie entirely in R. A restricted triangulation of V is defined to be a maximal subset, T_R, of L_R such that no two edges of T_R properly intersect. A restricted minimum weight triangulation of V_R is the restricted triangulation of V_R of least weight. We note that if R is a convex region then the definitions of minimum weight triangulation and restricted minimum weight triangulation coincide for V_R.

We may now formulate a dynamic programming approach to the problem as follows. We know that the convex hull of V must be included in any triangulation of V. Define R to be the planar region interior to (and including) the convex hull of V. Now consider paths of the form $(p_1, p_2, \ldots, p_k)$ with $k \geq 2$, where the following five conditions hold:

1. Each p_i is in V.
2. p_1 and p_k lie on the convex hull of V.
3. Each p_i, except p_1 and p_k, is not on the convex hull of V.
4. The path does not intersect itself in any way.
5. If $k = 2$ then p_1 and p_k are not adjacent vertices on the convex hull of V.

We will call such a path a splitting path of R because it splits R into two strictly smaller, although not necessarily convex, regions. Now, let T be a minimum weight triangulation of V. As long as $|V| \geq 3$, there exists at least one splitting path Q whose edges are in T. If we knew Q then the minimum weight triangulation of V could be calculated from the restricted minimum weight triangulations of the two subsets of vertices of V in each of the regions that Q breaks R into. These restricted minimum weight triangulations could be found recursively in a similar manner. The difficulty lies in finding a splitting path Q whose edges are in T.

One possibility is to consider each possible path which splits R into two strictly smaller regions. However, with $\mathcal{O}(|V|)$ vertices interior to the convex hull of V, this would mean examining $\mathcal{O}(|V|!)$ sequences of vertices of V to find each possible splitting path. Hence, in order to obtain a polynomial time algorithm using this approach the number of splitting paths that need to be considered must be limited.

Let us examine splitting paths in T more closely. Let z be the vertex in V with smallest x-coordinate. Note that z is on the convex hull of V. There are two cases to consider:

<u>Case 1</u>: The only edges in T with z as an endpoint are the edges connecting z to the vertices adjacent to it on the convex hull. Let w_1 and w_2 be the vertices adjacent to z on the convex hull. Then, by the definition of a triangulation, edge w_1w_2 must be in T and path $Q = (w_1, w_2)$ is a splitting path of R.

<u>Case 2</u>: There is an edge zy_1 in T such that y_1 is not w_1 or w_2. If y_1 lies on the convex hull of V then $Q = (z, y_1)$ is a splitting path of R as desired. Hence, suppose y_1 does not lie on the convex hull of V. The x-coordinate of y_1 is larger than the x-coordinate of z. Since T is a triangulation of V there is an edge y_1y_2 in T where the x-coordinate of y_2 is larger than the x-coordinate of y_1. And so on. Thus, there is a splitting path $Q = (z, y_1, y_2, \ldots, y_k)$ in T where the x-coordinate of y_i is less than the

x-coordinate of y_{i+1} for each i, $1 \leq i \leq k-1$.

Putting cases 1 and 2 together we find that it is sufficient to consider splitting paths of the following forms:

1. $Q = (w_1, w_2)$ where w_1 and w_2 are the vertices adjacent to z on the convex hull of V.
2. $Q = (z, y_1, y_2, \ldots, y_k)$ where for each i, $1 \leq i \leq k-1$, the x-coordinate of y_i is less than the x-coordinate of y_{i+1}.

Using the above observation, if there are $\mathcal{O}(|V|)$ vertices not on the convex hull of V, then only $\mathcal{O}(2^{|V|})$ splitting paths for R need to be considered. While this is certainly an improvement over $\mathcal{O}(|V|!)$, it is still exponential as opposed to polynomial.

Unfortunately, we have not been able to further reduce the number of splitting paths that need to be considered. The major obstacle is a lack of specific knowledge about the structure of minimum weight triangulations. We have had little success in discovering specific properties of minimum weight triangulations. Without such properties providing a good characterization of minimum weight triangulations the dynamic programming approach appears limited as far as obtaining a polynomial time algorithm is concerned.

Chapter 4 - TRI is NP-Complete

In this chapter we show that TRI is NP-Complete. The major portion of the chapter is devoted to showing that the problem of conjunctive normal form satisfiability (CNF-Satisfiability) is polynomially reducible to TRI. CNF-Satisfiability is an NP-Complete problem [1].

4.1 Intuition and Overview

Assume that we have an instance of the CNF-Satisfiability problem. That is, we have clauses $C_1, C_2, \ldots, C_k$ each of which is a sum of literals drawn from the variables $x_1, x_2, \ldots, x_n$. The problem is to determine if there is a truth assignment to the n variables such that each clause is satisfied. From the k clauses we will construct a set of vertices, V, and a set of edges, E, whose endpoints are in V such that there is a subset T of E triangulating V if and only if the set of clauses is satisfiable. Throughout this chapter a triangulation of V will refer to a subset T of E whose edges are a triangulation of V.

The building block in our construction will be a set of vertices and edges which we will refer to as a switch. A rectangular array of these switches will be employed, with one switch for each variable-clause pair. This array of switches will also be referred to as the network. We let S_{ij} represent the switch for variable x_i and clause C_j. Switch S_{ij} will be one of three types depending on whether x_i is in C_j or $\overline{x_i}$ is in C_j or neither is in C_j. We note that the switches are numbered in an x-y fashion as opposed to standard matrix numbering. That is, switch S_{ij} is is in the i^{th} column of switches going from left to right and in the j^{th} row of switches going from bottom to top.

In any triangulation of this array of switches we may regard two streams of information to be flowing through each switch, one stream flowing vertically and the other from left to right horizontally. The vertical stream of information flowing through S_{ij} carries a truth value for variable x_i. For each variable, x_i, the same truth value must be flowing vertically through each switch S_{ij}, where $1 \leq j \leq k$. The horizontal stream of information leaving switch S_{ij} on the right indicates whether or not clause C_j is satisfied by the assignment of the truth values (as determined by vertically flowing information for each variable) to the variables $x_1, x_2, \ldots, x_i$. This information may then flow into the left side of switch $S_{i+1,j}$. Our construction forces the information flowing into the left side of each switch S_{1j} to be "not satisfied" and the information flowing out of the right side of each switch S_{nj} to be "satisfied". What information is flowing through a switch depends on how the switch is triangulated.

Now consider a truth assignment, H, to the variables such that each clause is satisfied. Then, there exists a triangulation of the switches such that the vertical flowing information supports H and, for each clause C_j, there is a switch, S_{ij}, such that the truth assignment to x_i satisfies C_j, causing the horizontal flowing information about C_j to change from "not satisfied" to "satisfied".

Conversely, consider a truth assignment, H, which does not satisfy every clause. Then there is no triangulation of the switches such that the vertical flowing information supports H and yet for each clause C_j

the horizontal flowing information changes from "not satisfied" to "satisfied" in some switch S_{ij}.

The construction is such that the array of switches may be triangulated if and only if there is a truth assignment to the variables which satisfies each of the clauses.

4.2 Description of a Switch

Before giving a formal specification of the sets V and E we will describe the structure of a switch. Each switch will consist of the vertices and edges given in Figure 6. Note that the coordinates of the vertices are given relative to E_1 as the origin. In Figure 7 is a pictorial representation of a switch. An enlarged view of the center portion of a switch is shown in Figure 8.

Various vertices of each switch are classified as follows:

Frame vertices: $E_1,E_2,E_3,E_4,F,G,H,I,J,L,M,N,P,Q,R,S$

Terminals: $A_1,A_2,B_1,B_2,C_1,C_2,D_1,D_2$

Matched pair of terminals: A_1 and A_2, B_1 and B_2, C_1 and C_2, D_1 and D_2

When it is appropriate we will superscript the vertices of a switch. For example, N^{ij} is vertex N in switch S_{ij}. Note that each switch is symmetric in structure with respect to the lines $x = 50$ and $y = 50$ (the lines relative to E_1).

Figure 6: Switch Specifications

Each switch consists of the following vertices. The coordinates of each vertex are given relative to E_1.

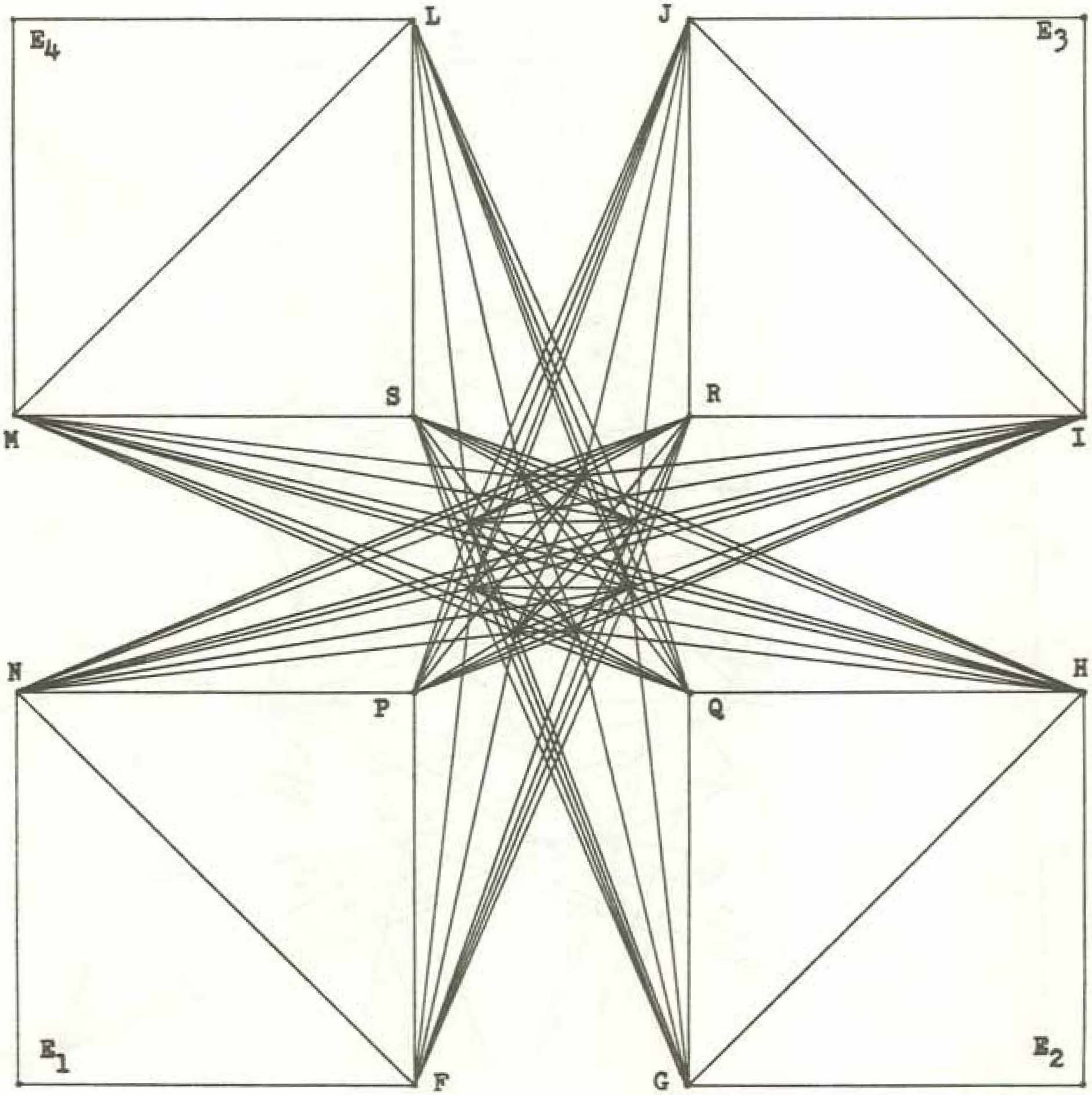
E_4 (0,100)	L (37,100)					J (63,100)	E_3 (100,100)
M (0,63)	S (37,63)					R (63,63)	I (100,63)
			A_2 (47,57)	D_1 (53,57)			
		B_2 (43,53)			C_1 (57,53)		
		C_2 (43,47)			B_1 (57,47)		
			D_2 (47,43)	A_1 (53,43)			
N (0,37)	P (37,37)					Q (63,37)	H (100,37)
E_1 (0,0)	F (37,0)					G (63,0)	E_2 (100,0)

Each switch consists of the following edges:

Frame Edges: E_1F, E_1N, FP, FN, NP, E_2G, E_2H, GH, GQ, HQ, E_3I, E_3J, IJ, IR, JR, E_4L, E_4M, LM, LS, MS

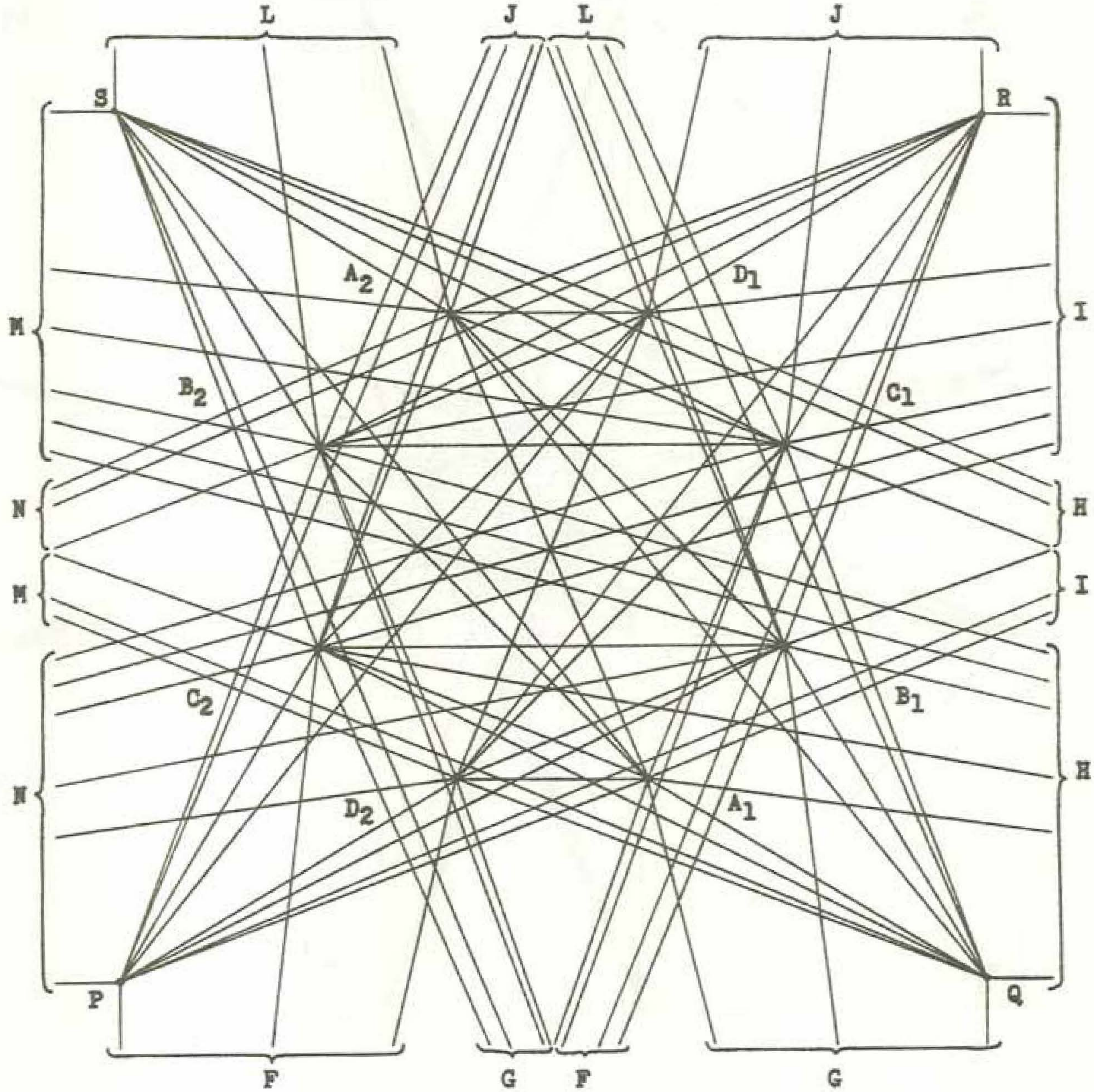
Non-frame Edges: FR, GS, HM, HS, IN, IP, JP, LQ, MQ, NR,
A_1G, A_1Q, A_1H, A_1I, A_1C_1, A_1A_2, A_1S, A_1B_2, A_1C_2, A_1D_2, A_1P, A_1F,
B_1G, B_1Q, B_1H, B_1I, B_1R, B_1L, B_1D_1, B_1A_2, B_1M, B_1C_2, B_1N, B_1D_2, B_1P, B_1F,
C_1Q, C_1H, C_1I, C_1R, C_1J, C_1L, C_1S, C_1A_2, C_1M, C_1B_2, C_1N, C_1D_2, C_1F,
D_1H, D_1I, D_1R, D_1J, D_1L, D_1S, D_1A_2, D_1B_2, D_1C_2, D_1P, D_1D_2,
A_2Q, A_2R, A_2J, A_2L, A_2S, A_2M, A_2N, A_2C_2,
B_2H, B_2I, B_2R, B_2J, B_2L, B_2S, B_2M, B_2N, B_2P, B_2G, B_2D_2,
C_2Q, C_2H, C_2I, C_2J, C_2S, C_2M, C_2N, C_2P, C_2F, C_2G,
D_2G, D_2Q, D_2R, D_2M, D_2N, D_2P, D_2F

Figure 7: A Switch



The eight unlabeled vertices in the center portion of the switch are the terminals. Figures 6 and 8 show the labels of these vertices.

Figure 8: An enlarged view of the center portion of a switch



4.3 Specification of V and E

4.3.1 The Basics

As previously stated our construction consists of a rectangular array of switches with one switch S_{ij} for each variable x_i, clause C_j pair. Adjacent switches in this network will coincide on appropriate frame vertices. Such frame vertices will thus have two labels. For instance, E_2^{11} and E_1^{21} refer to the same vertex. Vertex E_1 of switch S_{ij} will have coordinates $(100\cdot(i-1), 100\cdot(j-1))$.

To fulfill the definition of a triangulation we need to modify the switches in the outermost rows and columns of the network. These switches will be identical to regular switches except they will have one additional vertex (called a special vertex) and several additional edges. These special switches are specified as follows:

1. Each switch S_{1j}, for $1 \le j \le k$, contains a special vertex, T^{1j}, with coordinates $(0, 100\cdot(j-1) + 50)$ and the edges $\{T^{1j}\} \times \{M^{1j}, N^{1j}, A_2^{1j}, B_2^{1j}\}$
2. Each switch S_{i1}, for $1 \le i \le n$, contains a special vertex, U^{i1}, with coordinates $(100\cdot(i-1) + 50, 0)$ and the edges $\{U^{i1}\} \times \{F^{i1}, G^{i1}, A_1^{i1}, B_1^{i1}, C_2^{i1}, D_2^{i1}\}$
3. Each switch S_{nj}, for $1 \le j \le k$, contains a special vertex, V^{nj}, with coordinates $(100\cdot n, 100\cdot(j-1) + 50)$ and the edges $\{V^{nj}\} \times \{H^{nj}, I^{nj}, C_1^{nj}, D_1^{nj}\}$
4. Each switch S_{ik}, for $1 \le i \le n$, contains a special vertex, W^{ik}, with coordinates $(100\cdot(i-1) + 50, 100\cdot k)$ and the edges $\{W^{ik}\} \times \{J^{ik}, L^{ik}, A_2^{ik}, B_2^{ik}, C_1^{ik}, D_1^{ik}\}$

The frame is defined to be a set consisting of the frame edges of each switch in the network and each edge of the network which has a frame vertex as one endpoint and a special vertex as the other endpoint. We note that no edge with a terminal as an endpoint is included in the frame.

4.3.2 The Interswitch Edges

In addition to the edges within each switch there need to be edges in E whose endpoints lie in different switches. These edges will be called interswitch edges. Only terminals will be endpoints of interswitch edges and these edges will lie only between adjacent switches. It will be shown later that between any (horizontally or vertically) adjacent pair of switches, exactly one interswitch edge will be present in any triangulation. Intuitively, the chosen edge will carry information from one switch to the other.

Vertical interswitch edges may be specified as follows:

For each i and j pair, with $1 \leq i \leq n$ and $1 \leq j < k$, the following edges are placed in E:

$\{A_2^{ij}, C_1^{ij}\} \times \{A_1^{i,j+1}, C_2^{i,j+1}\}$ and $\{B_2^{ij}, D_1^{ij}\} \times \{B_1^{i,j+1}, D_2^{i,j+1}\}$

Intuitively, these edges will carry the vertical flowing information about the truth values of the variables, with the A-C edges carrying "false" and the B-D edges carrying "true".

The horizontal interswitch edges between two adjacent switches S_{ij} and $S_{i+1,j}$ will vary depending on the nature of switch S_{ij}. For this reason we classify each switch as being one of three possible types:

A switch S_{ij} is a neutral switch if and only if $x_i \notin C_j$ and $\overline{x_i} \notin C_j$

A switch S_{ij} is a positive switch if and only if $x_i \in C_j$

A switch S_{ij} is a negative switch if and only if $\overline{x_i} \in C_j$

Horizontal interswitch edges may be specified as follows:

1. For each i and j pair, with $1 \le i < n$ and $1 \le j \le k$, such that switch S_{ij} is a neutral switch the following edges are placed in E: $\{A_1^{ij}, B_1^{ij}\} \times \{A_2^{i+1,j}, B_2^{i+1,j}\}$ and $\{C_1^{ij}, D_1^{ij}\} \times \{C_2^{i+1,j}, D_2^{i+1,j}\}$ We define terminals A_1 and B_1 to be Clause-false and terminals C_1 and D_1 to be Clause-true in a neutral switch. Intuitively, these interswitch edges and those specified in 2, 3, 4 and 5, will carry the horizontal flowing information about the clauses, with edges with a Clause-false endpoint carrying "not satisfied" and edges with a Clause-true endpoint carrying "satisfied".
2. For each i and j pair, with $1 \le i < n$ and $1 \le j \le k$, such that switch S_{ij} is a positive switch the following edges are placed in E: $\{A_1^{ij}\} \times \{A_2^{i+1,j}, B_2^{i+1,j}\}$ and $\{B_1^{ij}, C_1^{ij}, D_1^{ij}\} \times \{C_2^{i+1,j}, D_2^{i+1,j}\}$ We define terminal A_1 to be Clause-false and terminals B_1, C_1 and D_1 to be Clause-true in a positive switch.
3. For each i and j pair, with $1 \le i < n$ and $1 \le j \le k$, such that switch S_{ij} is a negative switch the following edges are placed in E: $\{B_1^{ij}\} \times \{A_2^{i+1,j}, B_2^{i+1,j}\}$ and $\{A_1^{ij}, C_1^{ij}, D_1^{ij}\} \times \{C_2^{i+1,j}, D_2^{i+1,j}\}$ We define terminal B_1 to be Clause-false and terminals A_1, C_1 and D_1 to be Clause-true in a negative switch.
4. For each j with $1 \le j \le k$, such that switch S_{nj} is a positive switch, edge $[V^{nj}, B_1^{nj}]$ is placed in E.

5. For each j with $1 \le j \le k$, such that switch S_{nj} is a negative switch, edge $[V^{nj}, A_1^{nj}]$ is placed in E.

4.3.3 The Sets V and E

Set V contains all frame vertices, terminals and special vertices of each switch in the network.

Set E contains all of the edges of each switch in the network, as well as the interswitch edges as specified in the previous section. We note that the frame is included in E and that no edge in E properly intersects any edge of the frame. This means that any triangulation of V must contain all of the edges in the frame.

Finally, we note that the construction can be done in time polynomial in n and k. There are $n \cdot k$ switches in the network. Each switch may be constructed in a constant amount of time. Interswitch edges exist only between adjacent pairs of switches. There are $\mathcal{O}(n \cdot k)$ such pairs. The vertical interswitch edges are the same for each adjacent pair of switches, hence, they can be constructed in constant time for any given pair. The horizontal interswitch edges for any pair of adjacent switches depend only on the type of the left switch in the pair and, hence, can be constructed in constant time for any given pair of switches. Thus, the sets V and E can be constructed in time $\mathcal{O}(n \cdot k)$.

4.4 Proof that a solution to TRI yields a solution to CNF-Satisfiability

In this section we assume that T is a subset of E and is a triangulation of V. We show that there is a truth assignment to the variables $x_1, \ldots, x_n$ such that each clause $C_1, \ldots, C_k$ is satisfied. This truth assignment will be obtained from T.

4.4.1 Preliminaries

As stated earlier the frame must be included in T. This means that the non-frame edges in T must:

1. Complete the triangulation of each switch in the network.
2. Connect the switches together in a manner which yields a triangulation of V.

As we shall show, the triangulation T must fulfill these conditions with a very particular structure.

A terminal, α, in switch S_{ij} is defined to be East-connected in triangulation T if and only if there exists an edge $\alpha\beta$ in T such that $\alpha\beta$ properly intersects edge $[I^{ij}, H^{ij}]$. Now consider edge $[I^{ij}, H^{ij}]$. Since this edge is not in T there must be an edge in T which properly intersects $[I^{ij}, H^{ij}]$. By our construction, each such edge has a terminal of S_{ij} as an endpoint. This means that there must be at least one East-connected terminal per switch in any triangulation of V. Similarly, we can define and imply the existence of at least one West-connected, one North-connected and one South-connected terminal per switch in any triangulation of V. A connected terminal is a terminal that is at least one of East-connected, West-connected, North-connected or South-connected.

In chapter 1 we stated the following property of triangulations: If edge y_1y_2 is in triangulation T and is not on the convex hull of V, then in each half-plane, as determined by a line passing through y_1 and y_2, there must exist a vertex w in V such that edges y_1w and y_2w are in T and there does not exist a fourth vertex in V which lies on, or interior to, triangle y_1y_2w. That is, y_1y_2 is an edge in the boundary of two of the triangular faces of the straight-line planar graph determined by V and T (one face in each half-plane as determined by the line through y_1 and y_2).

This property will be used in the following proof as follows: In general, there will be an edge y_1y_2 in T and a specified half-plane. Consider the set of vertices, P, such that for each vertex w in P:

1. w lies in the specified half-plane.
2. Edges y_1w and y_2w are in E.
3. No other vertex of V lies on, or interior to, triangle y_1y_2w.

If there is only one vertex w in P, then edge y_1y_2 in T forces edges y_1w and y_2w to be in T by the property of triangulations stated above. This is denoted by $y_1y_2 \longrightarrow y_1y_2w$.

If there are two vertices, z_1 and z_2, in P then we will use the following notation: $y_1y_2 \longrightarrow$ choice

1. $y_1y_2z_1$
$\vdots$
2. $y_1y_2z_2$
$\vdots$

Typically, the first choice of $y_1y_2z_1$ will lead to a situation where an edge r is forced to be in T and yet there is already an edge s in T such that r and s properly intersect. Such a contradiction will be denoted " # to s ". In the proof in the next section an edge is said to be finally enumerated if it doesn't lead to a contradiction if placed in T. It may be that $|P| \geq 2$ and no vertex in P leads to a contradiction, but, that there exists a vertex y_3 in V such that for each w in P, $y_1w \longrightarrow y_1wy_3$. Intuitively, edge y_1y_2 in T forces edge y_1y_3 into T but the "force" requires two steps. In this case we write $y_1y_2 \xrightarrow{2} y_1y_3$. A typical example is that edge $[A_2^{ij}, N^{ij}]$ is in T. Then $P \subset \{A_1^{i-1,j}, B_1^{i-1,j}, C_1^{i-1,j}, D_1^{i-1,j}\}$ if $i > 1$ or $P = \{T^{1j}\}$ if $i = 1$. In either case, for any w in P, edge $[A_2^{ij}, w]$ in T forces edge $[A_2^{ij}, M^{ij}]$ to be in T. Hence, we write $A_2N \xrightarrow{2} A_2M$.

4.4.2 The Switch Triangulation Theorem

Theorem 1: Given any triangulation of V there are exactly two connected terminals in each switch and, furthermore, for each switch those two terminals are a matched pair of terminals.

Proof

Consider any triangulation T of V and any switch S_{ij} in the network. At least one terminal of S_{ij} is East-connected. Only terminals A_1, B_1, C_1 and D_1 in S_{ij} may be East-connected.

Case 1: Suppose terminal A_1 is East-connected in S_{ij}. Then there is a vertex Z in V such that A_1Z is in T and A_1Z properly intersects line segment IH of S_{ij}. Because of our construction Z is one of $A_2^{i+1,j}$, $B_2^{i+1,j}$, $C_2^{i+1,j}$ or $D_2^{i+1,j}$ if $i < n$ or is V^{nj} if $i = n$. Then, in S_{ij},

$A_1Z \longrightarrow A_1ZH$

$A_1H \longrightarrow A_1HQ$

$A_1Z \longrightarrow A_1ZI$

$A_1I \longrightarrow A_1IP$

$A_1P \longrightarrow A_1PF$

A_1Q and A_1F force A_1G

$IP \longrightarrow IPB_1$

$PB_1 \longrightarrow PB_1D_2$

$IB_1 \longrightarrow$ choice

1. IB_1R

   $B_1R \longrightarrow B_1RF$ # to PB_1

2. IB_1C_2

   $B_1C_2 \longrightarrow$ choice

   1. B_1C_2H # to PB_1

   2. B_1C_2N

      $B_1N \longrightarrow B_1ND_2$

      $ND_2 \longrightarrow ND_2P$

      $IC_2 \longrightarrow IC_2N$

      $IN \longrightarrow INC_1$

      $NC_1 \longrightarrow NC_1B_2$

      $C_1B_2 \longrightarrow$ choice

      1. C_1B_2M

         $C_1M \longrightarrow C_1MA_2$

         $C_1A_2 \longrightarrow C_1A_2S$

         $C_1S \longrightarrow C_1SH$ # to A_1I

      2. C_1B_2I

         $B_2I \longrightarrow B_2ID_1$

         $B_2D_1 \longrightarrow B_2D_1R$

         $B_2R \longrightarrow B_2RN$

$$RN \longrightarrow RNA_2$$

$$RA_2 \longrightarrow RA_2J$$

$$A_2N \xrightarrow{2} A_2M$$

$$A_2M \longrightarrow A_2MS$$

A_2S and A_2J force A_2L

$\therefore$ If A_1 is East-connected then A_1 is South-connected and A_2 is North-connected and West-connected.

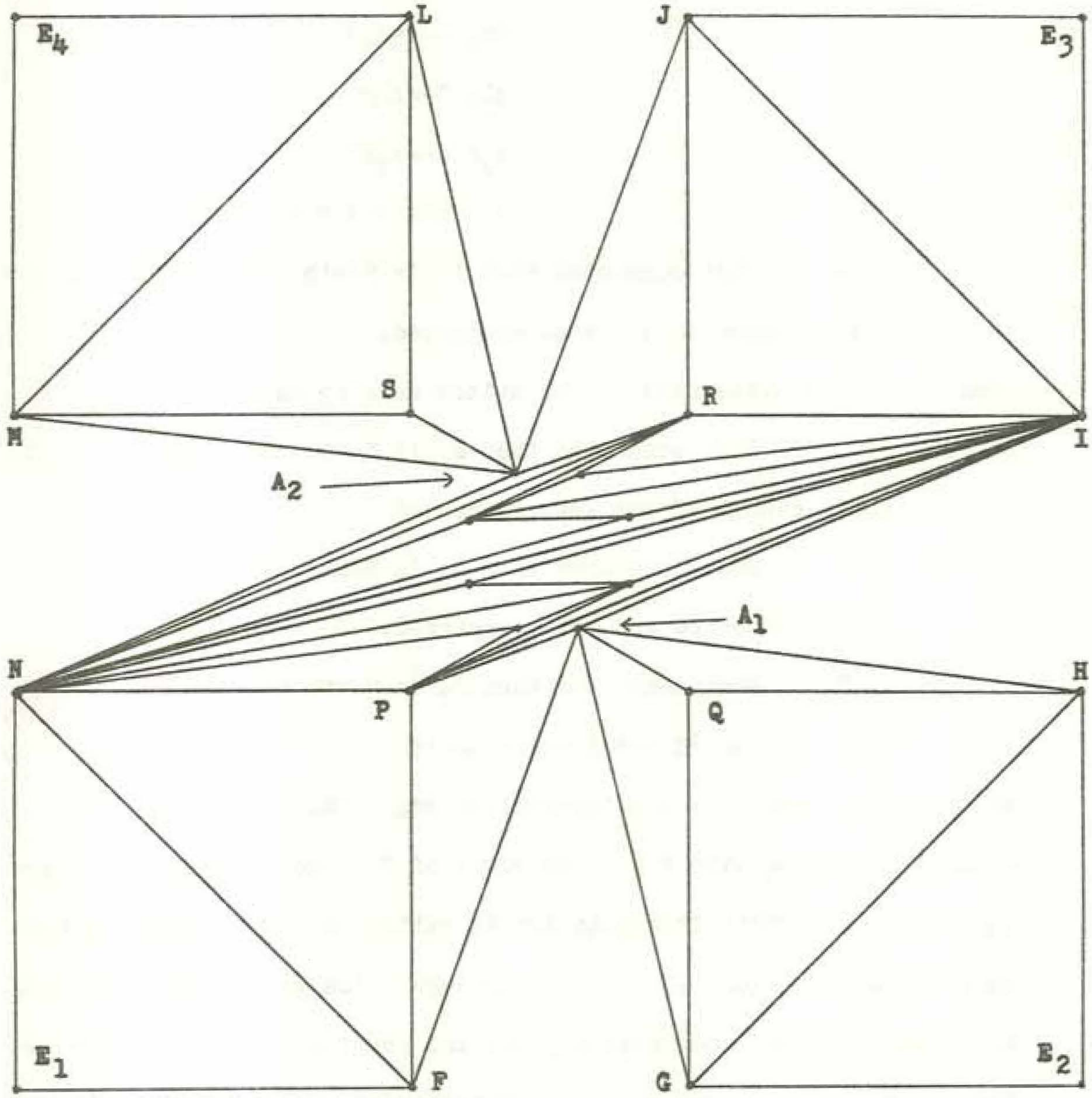
Because of the symmetries of the switch we also have:

1. If D_1 is East-connected then D_1 is North-connected and D_2 is South-connected and West-connected.
2. If A_2 is West-connected then A_2 is North-connected and A_1 is South-connected and East-connected.
3. If D_2 is West-connected then D_2 is South-connected and D_1 is North-connected and East-connected.

In the above proof the non-interswitch edges which are finally enumerated (along with the frame edges of S_{ij}) constitute a triangulation of S_{ij}. This triangulation is called an <u>A-triangulation</u> and is pictured in Figure 9. In an A-triangulation we say that terminal A_1 is <u>East-exposed</u> and <u>South-exposed</u> and terminal A_2 is <u>West-exposed</u> and <u>North-exposed</u>. Analogously, corresponding to D_1 and D_2 being the connected terminals of S_{ij}, there is a set of non-interswitch edges called a <u>D-triangulation</u>. This triangulation is shown in Figure 10. In a D-triangulation terminal D_1 is East-exposed and North-exposed and terminal D_2 is West-exposed and South-exposed.

<u>Case 2</u>: Suppose terminal B_1 is East-connected in S_{ij}. Then there is a vertex Z in V such that B_1Z is in T and B_1Z properly intersects line

Figure 9: The A-triangulation

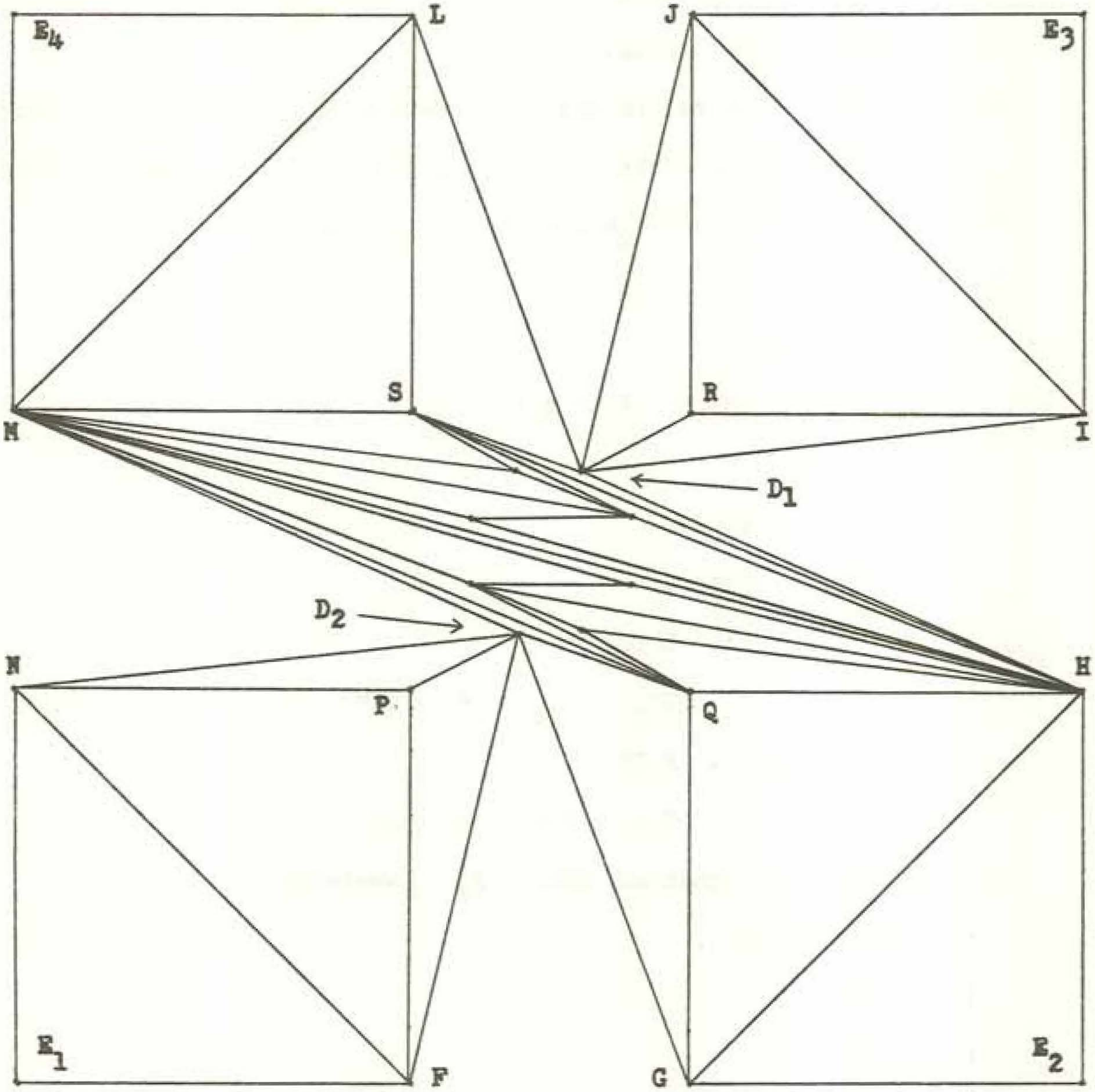


The following edges are in an A-triangulation:

Each frame edge,

A_1P, A_1F, A_1G, A_1Q, A_1H, A_1I, B_1I, B_1C_2, B_1N, B_1D_2, B_1P, C_1I, C_1B_2, C_1N, D_1I, D_1R, D_1B_2, A_2R, A_2J, A_2L, A_2S, A_2M, A_2N, B_2R, B_2N, B_2I, C_2I, C_2N, D_2N, D_2P, IP, NR, IN.

Figure 10: The D-triangulation



The following edges are in a D-triangulation:

Each frame edge,

A_1Q, A_1H, A_1C_2, B_1H, B_1M, B_1C_2, C_1H, C_1S, C_1M, C_1A_2, C_1B_2,

D_1H, D_1I, D_1R, D_1J, D_1L, D_1S, A_2S, A_2M, B_2H, B_2M,

C_2M, C_2Q, C_2H, D_2M, D_2N, D_2P, D_2F, D_2G, D_2Q, HS, MQ, HM.

segment IH in S_{ij}. Because of our construction Z is one of $A_2^{i+1,j}$, $B_2^{i+1,j}$, $C_2^{i+1,j}$ or $D_2^{i+1,j}$ unless $i = n$ in which case Z is V^{nj}. Then, in switch S_{ij}, $B_1Z \longrightarrow B_1ZI$.

Consider which terminal is West-connected in S_{ij}. From case 1, since B_1 is East-connected we know that it is not A_2 or D_2. Hence, suppose it is C_2. Then C_2M and C_2N must be in T. Then,

$C_2M \longrightarrow$ choice

1. C_2MB_1

   $MB_1 \longrightarrow MB_1H$ # to B_1I

2. C_2MS

   $C_2S \longrightarrow C_2SG$

   $B_1I \longrightarrow$ choice

   1. B_1IC_2

      $IC_2 \longrightarrow IC_2N$ # to C_2M

   2. B_1IR

      $B_1R \longrightarrow B_1RF$ # to SG

$\therefore$ C_2 is not West-connected, hence, B_2 is West-connected.

Now, in switch S_{ij},

$B_1Z \longrightarrow B_1ZI$

$B_1Z \longrightarrow B_1ZH$

$B_1H \longrightarrow B_1HQ$

$B_1I \longrightarrow$ choice

1. B_1IC_2

   $IC_2 \longrightarrow IC_2N$

   $C_2N \longrightarrow$ choice

1. C_2NP

$C_2P \longrightarrow$ choice

1. C_2PD_1 # to IC_2

2. C_2PF

$C_2F \xrightarrow{2} C_2G$

$C_2G \longrightarrow C_2GS$ # to IC_2

2. C_2NB_1

$NB_1 \longrightarrow NB_1D_2$

$B_1D_2 \longrightarrow B_1D_2P$

$B_1P \longrightarrow B_1PI$ # to B_1H

2. B_1IR

$B_1R \longrightarrow B_1RF$

B_1Q and B_1F force B_1G

$\therefore$ B_1 is the only East-connected and the only South-connected terminal. Furthermore, since B_2 is West-connected, by the symmetries of the switch, analogously to the above, we can show that B_2 is the only West-connected and the only North-connected terminal. This shows that non-frame edges B_2M, B_2S, B_2L, B_2J, B_2N, B_2P and PJ are all in T. All that remains is to show that the region bordered by the vertices P, J, R and F can indeed be triangulated. This can be done with edges JR, JC_2, C_2P, C_2A_2, A_2J, A_2D_1, D_1J, D_1R, D_1D_2, D_2R, D_2C_1, C_1R, C_1F, C_1A_1, A_1F, A_1D_2, D_2F, PF, D_2P, D_1P and C_2D_1.

$\therefore$ If B_1 is East-connected then B_1 is South-connected and B_2 is North-connected and West-connected.

Because of the symmetries of the switch we also have:

If C_1 is East-connected then C_1 is North-connected and C_2 is South-connected and West-connected.

In the above proof the non-interswitch edges which are finally enumerated (along with the frame edges of S_{ij}) constitute a triangulation of S_{ij}. This triangulation is called a B-triangulation and is pictured in Figure 11. In a B-triangulation terminal B_1 is East-exposed and South-exposed and terminal B_2 is West-exposed and North-exposed. Analogously, corresponding to C_1 and C_2 being the connected terminals of S_{ij}, there is a set of non-interswitch edges called a C-triangulation. This triangulation is shown in Figure 12. In a C-triangulation terminal C_1 is East-exposed and North-exposed and terminal C_2 is West-exposed and South-exposed.

$\therefore$ Given any triangulation of V there are exactly two connected terminals per switch and they are a matched pair of terminals.

□

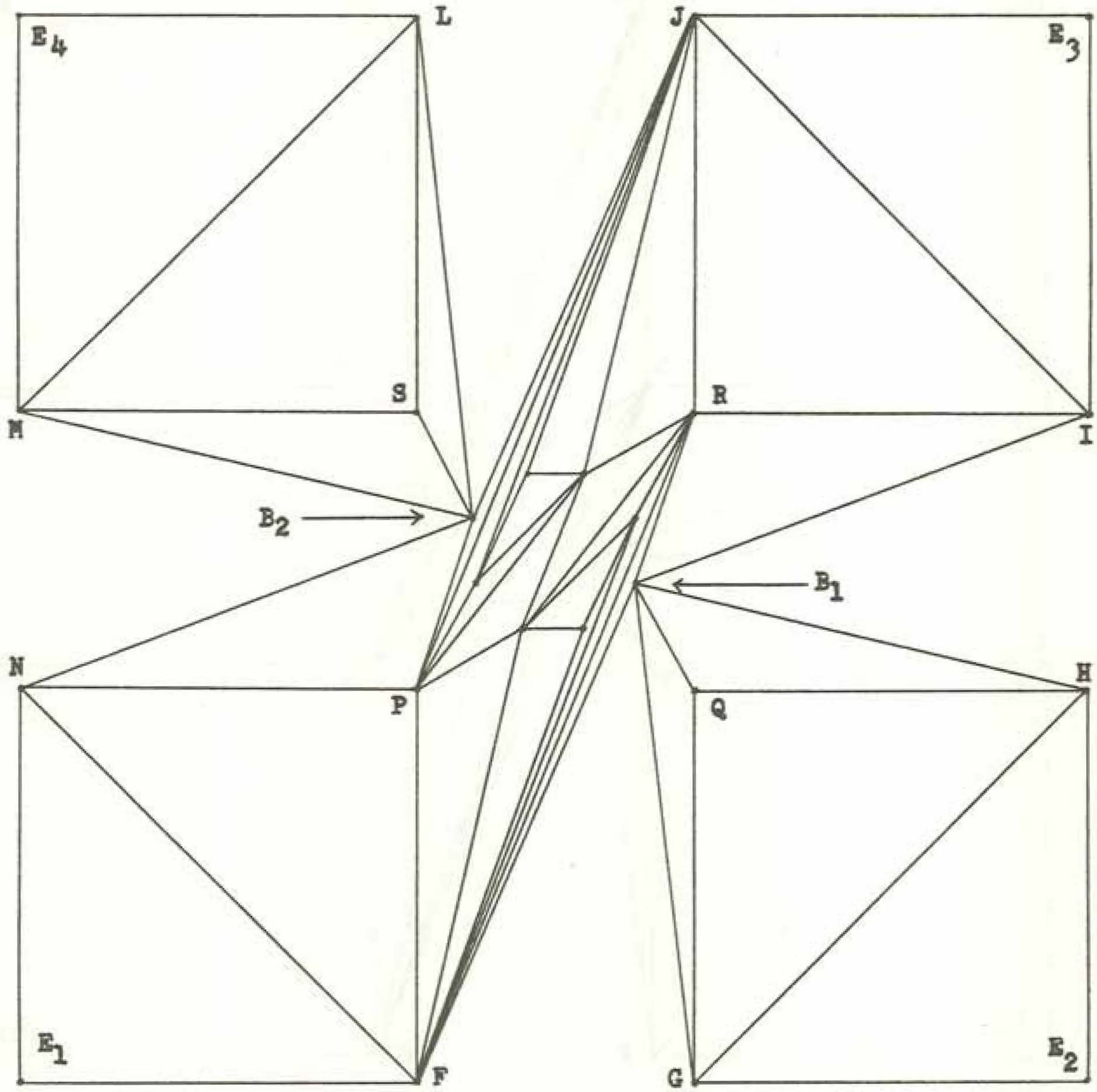
The following corollary follows immediately from the above theorem and our earlier remarks about the non-frame edges in T:

Corollary 1: If S_1 and S_2 are adjacent switches in the network and T is a triangulation of V, then there is exactly one interswitch edge in T whose endpoints are a terminal in S_1 and a terminal in S_2.

4.4.3 The Main Result

In the specifications of interswitch edges we defined various terminals to be Clause-true and Clause-false. For convenience, those definitions are restated here:

Figure 11: The B-triangulation



The following edges are in a B-triangulation:

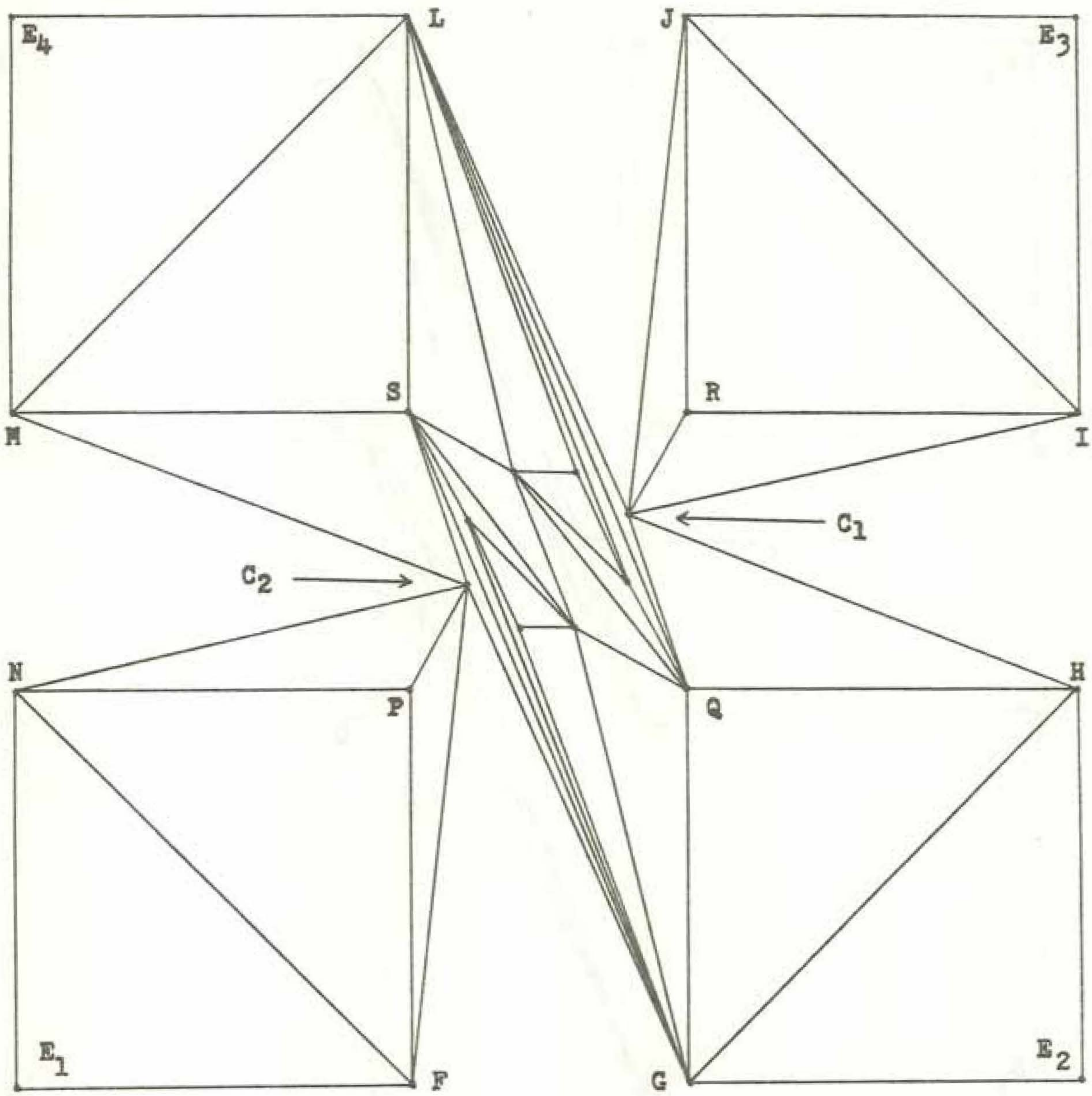
Each frame edge,

A_1F, A_1C_1, A_1D_2, B_1F, B_1G, B_1Q, B_1H, B_1I, B_1R,

C_1F, C_1R, C_1D_2, D_1R, D_1J, D_1A_2, D_1C_2, D_1P, D_1D_2, A_2J, A_2C_2,

B_2J, B_2L, B_2S, B_2M, B_2N, B_2P, C_2J, C_2P, D_2R, D_2P, D_2F, FR, JP.

Figure 12: The C-triangulation



The following edges are in a C-triangulation:

Each frame edge,

A_1G, A_1Q, A_1A_2, A_1S, A_1B_2, A_1D_2, B_1Q, B_1L, B_1D_1, B_1A_2,

C_1Q, C_1H, C_1I, C_1R, C_1J, C_1L, D_1L, D_1A_2, A_2L, A_2S, A_2Q,

B_2S, B_2G, B_2D_2, C_2S, C_2M, C_2N, C_2P, C_2F, C_2G, D_2G, GS, LQ.

In a neutral switch, terminals A_1 and B_1 are Clause-false and terminals C_1 and D_1 are Clause-true.

In a positive switch, terminal A_1 is Clause-false and terminals B_1, C_1 and D_1 are Clause-true.

In a negative switch, terminal B_1 is Clause-false and terminals A_1, C_1 and D_1 are Clause-true.

The following three lemmas are useful in proving the main result:

Lemma 1: In any given triangulation of V, for each i, $1 \le i \le n$, either the connected terminals are B's and D's for all S_{ij}, or the connected terminals are A's or C's for all S_{ij}, $1 \le j \le k$.

Proof

The result follows immediately from our construction of vertical interswitch edges, theorem 1 and corollary 1. □

Lemma 2: In any given triangulation of V, the West-connected terminal in each switch S_{1j} is A_2^{1j} or B_2^{1j} and the East-connected terminal in each switch S_{nj} is Clause-true, for $1 \le j \le k$.

Proof

The result follows immediately from our construction of special switches and interswitch edges. □

Lemma 3: In any given triangulation of V, for each j, $1 \le j \le k$, there exists an i, with $1 \le i \le n$, such that the East-connected terminal of S_{ij} is either A_1 or B_1 and it is Clause-true.

Proof

Consider any j such that $1 \le j \le k$, and suppose the lemma doesn't hold. By lemma 2, the West-connected terminal in S_{1j} is A_2 or B_2. Then the East-connected terminal is A_1 or B_1. By assumption it is Clause-false.

Then, by our construction and corollary 1, the West-connected terminal in S_{2j} is A_2 or B_2. Inductively then, the East-connected terminal in S_{nj} is either A_1 or B_1. By assumption, it is Clause-false. This contradicts lemma 2.

□

Now consider the following truth assignments to the variables $x_1, \ldots, x_n$:

x_i is true if the South-connected terminal in S_{i1} is B_1 or D_2.

x_i is false if the South-connected terminal in S_{i1} is A_1 or C_2.

Theorem 2: For each j, $1 \le j \le k$, the clause C_j is satisfied by this truth assignment to the variables.

Proof

Consider any j such that $1 \le j \le k$. By lemma 3, there is an i such that the East-connected terminal of S_{ij} is either A_1 or B_1 and it is Clause-true.

Case 1: The connected terminal is B_1. Since it is Clause-true this must be a positive switch, so x_i is in C_j. But then B_1 is the South-connected terminal and by lemma 1, the South-connected terminal of S_{i1} is B_1 or D_2. Then, by our assignment x_i is true and C_j is satisfied.

Case 2: The connected terminal is A_1. Since it is Clause-true this must be a negative switch, so $\overline{x_i}$ is in C_j. But then A_1 is the South-connected terminal and by lemma 1, the South-connected terminal of S_{i1} is A_1 or C_2. Then, by our assignment x_i is false and C_j is satisfied.

□

Therefore, from a triangulation T of V, with T a subset of E, we have obtained a truth assignment to the variables $x_1, \ldots, x_n$ such that each of the clauses $C_1, \ldots, C_k$ is satisfied.

4.5 Proof that a solution to CNF-Satisfiability yields a solution to TRI

Assume that $H_1, \ldots, H_n$ is a truth assignment to $x_1, \ldots, x_n$ such that each of the clauses $C_1, \ldots, C_k$ is satisfied. We will show that there is a subset, T, of E, such that the edges in T triangulate V. Initially we note that a set, T, consisting of edges meeting the following requirements will suffice as a triangulation of V. It is clear that T need only include:

1. The edges in the frame.
2. The edges in a triangulation of each switch in the network. That is, for each switch, the edges in either an A, B, C or D-triangulation.
3. For each adjacent pair of switches an edge whose endpoints are the appropriate exposed terminals of those switches. (The exposed terminals having been determined by the triangulations specified in 2.)
4. For each special vertex in V, an edge whose endpoints are the special vertex and the appropriate exposed terminal of the switch in which the special vertex is located.

The remainder of this section is devoted to specifying a set of edges which meets the above requirements. Initially we place the frame in T and again note that no edge in E properly intersects any edge in the frame. The frame edges thus present no further difficulty.

4.5.1 The Triangulation of Each Switch

For each clause, C_j, we define W_j to be the least i such that x_i is in C_j or $\overline{x_i}$ is in C_j and the truth assignment of H_i to x_i causes C_j to be satisfied. Then, switch S_{ij} is triangulated in T as follows:

For $i \le W_j$, if H_i is true then S_{ij} is B-triangulated
else S_{ij} is A-triangulated.

For $i > W_j$, if H_i is true then S_{ij} is D-triangulated
else S_{ij} is C-triangulated.

The exposed terminals of each switch are determined by the triangulation specified for each switch.

4.5.2 Interswitch Edges in T

Theorem 3: For each i and j pair, with $1 \le i \le n$ and $1 \le j \le k-1$, there is an edge in E whose endpoints are the North-exposed terminal of S_{ij} and the South-exposed terminal of $S_{i,j+1}$.

Proof

Consider any i and j pair such that $1 \le i \le n$ and $1 \le j \le k-1$.

Case 1: The North-exposed terminal of S_{ij} is B_2^{ij} or D_1^{ij}. This implies that H_i is true, hence, the South-exposed terminal of $S_{i,j+1}$ is $B_1^{i,j+1}$ or $D_2^{i,j+1}$. But, by our interswitch edge specifications, each of the four edges: $[B_2^{ij}, B_1^{i,j+1}]$, $[B_2^{ij}, D_2^{i,j+1}]$, $[D_1^{ij}, B_1^{i,j+1}]$, and $[D_1^{ij}, D_2^{i,j+1}]$ is in E.

Case 2: The North-exposed terminal of S_{ij} is A_2^{ij} or C_1^{ij}. The proof is completely analogous to the one for case 1. □

Theorem 4: For each i and j pair, with $1 \le i \le n-1$ and $1 \le j \le k$, there is an edge in E whose endpoints are the East-exposed terminal of S_{ij} and the West-exposed terminal of $S_{i+1,j}$.

Proof

Consider any i and j pair such that $1 \le i \le n-1$ and $1 \le j \le k$.

Case 1: $i > W_j$

Because $i > W_j$, the East-exposed terminal of S_{ij} is either C_1^{ij} or D_1^{ij} and the West-exposed terminal of $S_{i+1,j}$ is either $C_2^{i+1,j}$ or $D_2^{i+1,j}$. But, by our interswitch edge specifications each of the four edges: $[C_1^{ij}, C_2^{i+1,j}]$, $[C_1^{ij}, D_2^{i+1,j}]$, $[D_1^{ij}, C_2^{i+1,j}]$, and $[D_1^{ij}, D_2^{i+1,j}]$ is in E.

Case 2: $i = W_j$

Subcase 1: The East-exposed terminal of S_{ij} is B_1^{ij}. By the definition of W_j this switch is either a positive or negative switch. Assume that it is a negative switch, hence $\overline{x_i}$ is in C_j. But since B_1^{ij} is the East-exposed terminal, H_i is true. But this contradicts the definition of W_j. Therefore, this is a positive switch. Since $i+1 > W_j$ the West-exposed terminal of $S_{i+1,j}$ is either $C_2^{i+1,j}$ or $D_2^{i+1,j}$. But, by our interswitch edge specifications both of the edges $[B_1^{ij}, C_2^{i+1,j}]$ and $[B_1^{ij}, D_2^{i+1,j}]$ are in E.

Subcase 2: The East-exposed terminal of S_{ij} is A_1^{ij}. Similarly to subcase 1 we can show that this is a negative switch and that the desired edge exists in E.

Case 3: $i < W_j$

Subcase 1: The East-exposed terminal of S_{ij} is B_1^{ij}.

Subcase a: Switch S_{ij} is a neutral switch. Because $i+1 \le W_j$, the West-exposed terminal of $S_{i+1,j}$ is either $A_2^{i+1,j}$ or $B_2^{i+1,j}$. By the interswitch specifications both of the edges $[B_1^{ij}, A_2^{i+1,j}]$ and $[B_1^{ij}, B_2^{i+1,j}]$ are in E.

Subcase b: Switch S_{ij} is a positive switch. This means that x_i is in C_j. Because B_1^{ij} is the East-exposed terminal of S_{ij}, the truth value of H_i is true. But this means that C_j is satisfied by the assignment of H_i to x_i. This is a contradiction of the definition of W_j. Hence, S_{ij} is not a positive switch.

Subcase c: Switch S_{ij} is a negative switch. Because $i+1 \leq W_j$, the West-exposed terminal of $S_{i+1,j}$ is either $A_2^{i+1,j}$ or $B_2^{i+1,j}$. But, by our interswitch edge specifications for S_{ij}, a negative switch, each of the edges: $[B_1^{ij}, A_2^{i+1,j}]$, $[B_1^{ij}, B_2^{i+1,j}]$ is in E.

Subcase 2: The East-exposed terminal of S_{ij} is A_1^{ij}.

The proof is analogous to that for subcase 1, with the roles of subcases b and c reversed. □

Hence, for each pair of adjacent switches there is an edge in E whose endpoints are the appropriate exposed terminals of those switches. Each of these edges is placed into T.

4.5.3 Additional Special Switch Edges in T

Theorem 5: For each special vertex in V there is an edge in E whose endpoints are the special vertex and the appropriate exposed terminal of the switch in which the special vertex is located.

Proof

Case 1: The special vertex is U^{i1} with $1 \leq i \leq n$. By our basic specifications of special switches each of the edges $[U^{i1}, A_1^{i1}]$, $[U^{i1}, B_1^{i1}]$, $[U^{i1}, C_2^{i1}]$ and $[U^{i1}, D_2^{i1}]$ is in E. Thus, whichever terminal is exposed in S_{i1} the desired edge is in E.

<u>Case 2</u>: The special vertex is W^{ik} with $1 \le i \le n$. By our basic specifications of special switches each of the edges $[W^{ik}, A_2^{ik}]$, $[W^{ik}, B_2^{ik}]$, $[W^{ik}, C_1^{ik}]$ and $[W^{ik}, D_1^{ik}]$ is in E. Thus, whichever terminal is exposed in S_{ik} the desired edge is in E.

<u>Case 3</u>: The special vertex is T^{1j} with $1 \le j \le k$. Because $1 \le W_j$, the West-exposed terminal is either A_2^{1j} or B_2^{1j}. But, by our basic specifications of special switches, both of the edges $[T^{1j}, A_2^{1j}]$, and $[T^{1j}, B_2^{1j}]$ are in E. Thus, whichever terminal is exposed in S_{1j} the desired edge is in E.

<u>Case 4</u>: The special vertex is V^{nj} with $1 \le j \le k$.

<u>Subcase 1</u>: $n > W_j$

Because $n > W_j$, the East-exposed terminal of S_{nj} is either C_1^{nj} or D_1^{nj}. By our basic specifications of special switches each of the edges $[V^{nj}, C_1^{nj}]$ and $[V^{nj}, D_1^{nj}]$ is in E. Thus, whichever terminal is exposed in S_{nj} the desired edge is in E.

<u>Subcase 2</u>: $n = W_j$

<u>Subcase a</u>: The East-exposed terminal of switch S_{nj} is B_1^{nj}. Then, from case 2 of the proof of theorem 4, this is a positive switch. But by our interswitch edge specifications (part 4) the edge $[V^{nj}, B_1^{nj}]$ is in E.

<u>Subcase b</u>: The East-exposed terminal of switch S_{nj} is A_1^{nj}. Then, from case 2 of the proof of theorem 4, this is a negative switch. But by our interswitch edge specifications (part 5) the edge $[V^{nj}, A_1^{nj}]$ is in E.

□

Hence, for each special vertex in V there is an edge in E whose endpoints are the special vertex and the appropriate exposed terminal of the switch that the special vertex is a part of. Each of these edges is placed in T.

We have now specified a set of edges T which is a subset of E and which satisfies the four requirements given as being sufficient for a triangulation of V. Hence, the set T is a triangulation of V.

This completes the proof that CNF-Satisfiability is polynomially reducible to TRI.

4.6 Finishing Up

Theorem 6: TRI is NP-Complete.

Proof

In the first five sections of this chapter we have shown that CNF-Satisfiability, a known NP-Complete problem, is polynomially reducible to TRI. All that remains is to show that TRI is in NP. Consider an instance of TRI as specified by the sets V and E. We know that a set T is a triangulation of V if and only if the following two properties hold for T:

1. No two edges in T properly intersect.
2. For every edge, e, whose endpoints are vertices of V, either e is in T or e properly intersects some edge in T.

Hence, given the sets V and E, we nondeterministically choose the set T and then verify that these two properties hold. To test for property 1 requires time $\mathcal{O}(|T|^2)$ and testing for property 2 may be done in time $\mathcal{O}(|V|^2 \cdot |T|)$. Therefore, TRI is in NP and hence, TRI is NP-Complete.

□

Chapter 5 - Conclusion

5.1 Summary

This thesis has examined two problems involving triangulations of a set of points in the plane: the problem of finding a minimum weight triangulation given all of the edges between the points and the problem of determining the existence of a triangulation in a given subset of the edges. We discussed several aspects of the MWT problem and gave counterexamples to two published algorithms for it. We have shown that TRI is NP-Complete. We conjecture that MWT is also NP-Complete. This is based on a comparison of these two triangulation problems with the corresponding Hamiltonian circuit problems and the corresponding spanning tree problems. Both of the corresponding Hamiltonian circuit problems (that is, the problem of existence given some of the edges and the problem of minimum weight given all of the edges) are NP-Complete. In comparison, there are efficient algorithms for both of the spanning tree problems. Therefore, because TRI is NP-Complete, we would find it very surprising if MWT was not also NP-Complete.

We should note that as we have stated it, we would expect only that MWT be NP-Hard. However, we can change the problem slightly to ask if there is a triangulation of V with weight $\leq m$. We would then expect that this problem is NP-Complete. The difficulty with the original version of MWT which is not present in the new version lies in showing that the problem is in NP. The same comment can of course be made about the Hamiltonian circuit problems mentioned above.

5.2 Open Problems

In addition to the need to resolve the status of MWT there are several other open problems involving minimum weight triangulations.

The first of these problems is to show that a shortest edge between points in V is in a minimum weight triangulation of V. If a shortest edge lies on the convex hull of V then it is in each minimum weight triangulation by the definition of a triangulation. But what if a shortest edge does not lie on the convex hull? We conjecture that in this case also a shortest edge must be in a minimum weight triangulation. This problem should not be confused with the example given earlier which showed that the shortest edge not on the convex hull was not necessarily in a minimum weight triangulation. In that example the shortest edge among all of the edges was on the convex hull.

A second problem is to bound the weights of the triangulations produced by the Duppe-Gottschalk and Shamos-Hoey algorithms with respect to a minimum weight triangulation. We know that for arbitrary triangulations this ratio may be as large as $\mathcal{O}(|V|)$. A further problem is to determine under what conditions either of the two algorithms does produce a minimum weight triangulation. These questions will be especially important if MWT is indeed NP-Complete.

Another problem would be to determine the accuracy of the functional approximations which are obtained from a minimum weight triangulation as opposed to other triangulations. For instance, the Shamos-Hoey algorithm produces a triangulation with the property that the circumcircle of each triangle contains no points of V except the vertices of that triangle. We would like to know if this property makes the triangulation produced by their algorithm especially good for approximations.

References

1. Aho, A.V., Hopcroft, J.E. and Ullman, J.D., The Design and Analysis of Computer Algorithms. Addison-Wesley (1974), 470 pp.

2. Babuska, I. and Aziz, A.K., "On the Angle Condition in the Finite Element Method", SIAM Journal on Numerical Analysis. 13:2 (1976) 214-226.

3. Bellman, R.E. and Dreyfus, S.E., Applied Dynamic Programming. Princeton University Press (1962), 363 pp.

4. Bramble, J.H. and Zlamal, M., "Triangular Elements in the Finite Element Method", Mathematics of Computation. 24 (1970), 809-820.

5. Davis, J.C. and McCullagh, M.J., eds. Display and Analysis of Spatial Data. Wiley (1975), 378 pp.

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

7. Garey, M.R., Graham, R.L. and Johnson, D.S., "Some NP-Complete Geometric Problems", Proceedings of the Eighth ACM Symposium on the Theory of Computing. (1976), 10-22.

8. Karp, R.M., "Reducibility Among Combinatorial Problems", in Complexity of Computer Computations, R.E. Miller and J.W. Thatcher, eds., Plenum Press (1972), 85-104.

9. Papadimitriou, C.H., "The Euclidean Traveling Salesman Problem is NP-Complete", Princeton University Computer Science TR No. 191 (1975).

10. Rogers, C.A., Packing and Covering. Cambridge University Press (1964) 111 pp.

11. Rosenkrantz, D.J., Stearns, R.E. and Lewis, P.M., "Approximate Algorithms for the Traveling Salesperson Problem", Proceedings of the Fifteenth Annual Symposium on Switching and Automata Theory. IEEE (1974), 33-42.

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

13. Shamos, M.I., "Geometric Complexity", Proceedings of the Seventh ACM Symposium on the Theory of Computing. (1975), 224-233.

14. Shamos, M.I., Problems in Computational Geometry. (1974) unpublished manuscript.