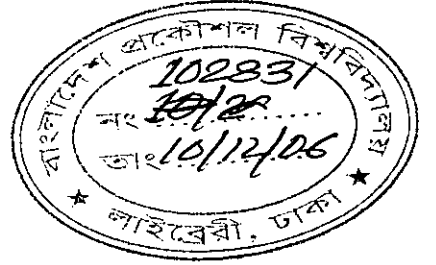


M.Sc. Engg. Thesis

Hardness of Finding  
Minimum Face-Spanning Subgraphs  
of Plane Graphs

by  
Md. Mostofa Ali Patwary

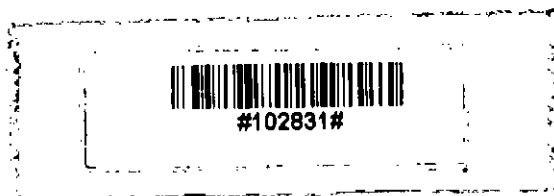
Submitted to



Department of Computer Science and Engineering  
in partial fulfilment of the requirements for the degree of  
Master of Science in Computer Science and Engineering


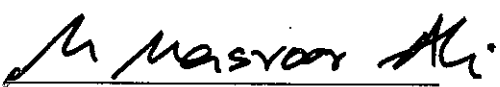
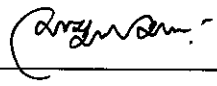

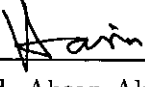
Department of Computer Science and Engineering  
Bangladesh University of Engineering and Technology (BUET)  
Dhaka 1000

November 2006



The thesis titled “**Hardness of Finding Minimum Face-Spanning Subgraphs of Plane Graphs,**” submitted by Md. Mostofa Ali Patwary, Roll No. 040405010P, Session April 2004, to the Department of Computer Science and Engineering, Bangladesh University of Engineering and Technology, has been accepted as satisfactory in partial fulfillment of the requirements for the degree of Master of Science in Computer Science and Engineering and approved as to its style and contents. Examination held on November 21, 2006.

### Board of Examiners

1.   
\_\_\_\_\_  
Dr. Md. Saidur Rahman  
Professor  
Department of CSE  
BUET, Dhaka 1000  
Chairman  
(Supervisor)
2.   
\_\_\_\_\_  
Dr. Muhammad Masroor Ali  
Professor & Head  
Department of CSE  
BUET, Dhaka 1000  
Member  
(Ex-officio)
3.   
\_\_\_\_\_  
Dr. Md. Mostofa Akbar  
Associate Professor  
Department of CSE  
BUET, Dhaka 1000  
Member
4.   
\_\_\_\_\_  
Dr. Masud Hasan  
Assistant Professor  
Department of CSE  
BUET, Dhaka 1000  
Member
5.   
\_\_\_\_\_  
Dr. Md. Ahsan Akhtar Hasin  
Professor  
Department of IPE  
BUET, Dhaka 1000  
Member  
(External)

## Candidate's Declaration

It is hereby declared that this thesis or any part of it has not been submitted elsewhere for the award of any degree or diploma.



---

Md. Mostofa Ali Patwary  
Candidate

# Contents

<i>Board of Examiners</i>	i
<i>Candidate's Declaration</i>	ii
Acknowledgements	viii
Abstract	ix
<b>1 Introduction</b>	<b>1</b>
1.1 Problem Statement . . . . .	2
1.2 Applications . . . . .	2
1.2.1 Planning Gas Pipelines in a Locality . . . . .	2
1.2.2 Planning Canal Networks in Irrigation Systems . . . . .	4
1.3 Literature Review . . . . .	6
1.3.1 Vertex Cover Problem . . . . .	6
1.3.2 Face Cover Problem . . . . .	6
1.4 Scope of Thesis . . . . .	7
1.4.1 Face-Spanning Subgraph Problem . . . . .	8
1.4.2 Minimum-Vertex Face-Spanning Subgraph Problem . . . . .	9
1.4.3 Approximation Algorithms . . . . .	9
1.5 Summary . . . . .	10

<b>2 Preliminaries</b>	<b>12</b>
2.1 Basic Terminology	12
2.1.1 Graphs	13
2.1.2 Connectivity	14
2.1.3 Trees and Subgraphs	14
2.1.4 Planar Graphs and Plane Graphs	15
2.2 Complexity Theory	16
2.2.1 Decision Problems and Optimization Problems	16
2.2.2 Complexity Classes	17
2.3 Approximation Algorithm and Approximation Ratio	20
2.4 Weighted Tree Cover Problem	21
2.5 Connected Vertex Cover Problem	22
2.6 Summary	23
<b>3 Face-Spanning Subgraph Problem</b>	<b>24</b>
3.1 Problem Definition	24
3.2 <i>NP</i> -completeness of FSSP	26
3.2.1 FSSP is in <i>NP</i>	26
3.2.2 FSSP is <i>NP</i> -hard	28
3.3 Summary	34
<b>4 Minimum-Vertex Face-Spanning Subgraph Problem</b>	<b>35</b>
4.1 Problem Definition	36
4.2 <i>NP</i> -completeness of MVFSSP	37
4.2.1 MVFSSP is in <i>NP</i>	37
4.2.2 MVFSSP is <i>NP</i> -hard	38
4.3 Summary	43

<b>5</b>	<b>Approximation Algorithms</b>	<b>45</b>
5.1	Minimal Face-Spanning Subgraph . . . . .	46
5.2	<i>Find-Minimal-Subgraph</i> Algorithm . . . . .	49
5.3	Approximation Algorithms . . . . .	51
5.3.1	Minimum-Vertex Face-Spanning Subgraph Problem . . . . .	51
5.3.2	Face-Spanning Subgraph Problem . . . . .	52
5.4	Summary . . . . .	52
<b>6</b>	<b>Conclusion</b>	<b>54</b>

# List of Figures

1.1	A road-network of a locality along with gas pipelines . . . . .	4
1.2	A canal network to supply water in all plots of a region . . . . .	5
1.3	Vertex cover and face-spanning subgraphs . . . . .	7
1.4	Face cover and face-spanning subgraphs . . . . .	8
2.1	A simple graph with five vertices and eight edges. . . . .	13
2.2	Example of a connected graph and a disconnected graph . . . . .	14
2.3	Example of an edge-induced subgraph . . . . .	15
2.4	The set of all problems that contains the $P$ and $NP$ problems. . . . .	18
2.5	Illustration of the relationship among different complexity classes . . . . .	19
2.6	An example of a connected vertex cover . . . . .	22
3.1	Example of face-spanning subgraphs of a plane graph . . . . .	25
3.2	Data structures to verify face-spanning subgraphs of a plane graph . . . . .	27
3.3	Illustration for the construction of $G'$ from $G$ . . . . .	30
3.4	Illustration for the construction of $T$ from $H$ . . . . .	32
4.1	Example of minimum-vertex face-spanning subgraphs of a plane graph . . . . .	36
4.2	Illustration for the construction of $G'$ from $G$ . . . . .	40
4.3	Illustration for the construction of $H$ from $H'$ . . . . .	41
5.1	Illustration of minimal face-spanning subgraph of a plane graph . . . . .	47
5.2	Minimal face-spanning subgraph and minimum face-spanning subgraph . . . . .	48

5.3 Minimal face-spanning subgraph and minimum-vertex face-spanning subgraph . . . . . 49

5.4 Tight lower bound for a minimal face-spanning subgraph of a plane graph . 49

5.5 An example to illustrate Algorithm *Find-Minimal-Subgraph* . . . . . 50

5.6 Tight upper bound for a minimal face-spanning subgraph of a plane graph 51



# Acknowledgments

All praises due to Allah, Lord of the Worlds, who granted me the ability to finish this thesis.

I would like to thank my supervisor Professor Dr. Md. Saidur Rahman for introducing me to the field of face-spanning subgraph problems and the hardness of the problems. I have learned from him how to write, speak and present well. I thank him for his patience in reviewing my so many inferior drafts, for correcting my proofs and language, suggesting new ways of thinking and encouraging me to continue my work.

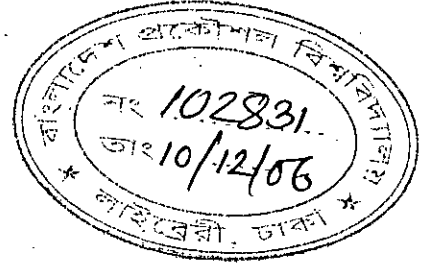
I would also want to thank the members of my thesis committee for their valuable suggestions. I thank Professor Dr. Muhammad Masroor Ali, Dr. Md. Mostofa Akbar, Dr. Masud Hasan and Professor Dr. Md. Ahsan Akhtar Hasin.

It would be inappropriate if I do not mention the members of our research group. They gave me valuable suggestions and listened to all of my presentations. I just want to say them: Thank You!

# Abstract

In this research work, we have introduced a new graph problem namely “minimum face-spanning subgraph problem”. Let  $G$  be an edge weighted connected plane graph, let  $V(G)$  and  $E(G)$  be the set of vertices and edges, respectively. Let  $F$  be the set of faces of graph  $G$ . For each edge  $e \in E$ ,  $w(e) \geq 0$  is the weight of the edge  $e$  of  $G$ . A face-spanning subgraph of  $G$  is a connected subgraph  $H$  induced by a set of edges  $S \subseteq E$  such that  $S$  contains at least one vertex from the boundary of each face  $f \in F$  of  $G$ . The minimum face-spanning subgraph problem asks to find a face-spanning subgraph  $H$  induced by  $S$  of  $G$  such that cost of  $H$  is minimum. Finding a minimum face-spanning subgraph has practical applications in planning irrigation canal networks in irrigation systems, planning gas pipelines in a locality, layout of power supply lines in a printed circuit board etc.

Efficient algorithms are necessary to solve these kinds of problems which arise from numerous practical applications. However developing efficient algorithms is not always possible. In this thesis, we show that finding an efficient algorithm for solving the minimum face-spanning subgraph problem is unlikely by showing that this problem belongs to the infamous class of  $NP$ -complete problems. We also prove that a variation of the minimum face-spanning subgraph problem called “minimum-vertex face-spanning subgraph problem” is  $NP$ -complete. Since it is unlikely to have efficient algorithms for both the problems, design of approximation algorithms is an urgent need. We present approximation algorithms for both the problems in this thesis. Approximation ratio and complexity analysis of the developed approximation algorithms are also presented in this thesis.



# Chapter 1

## Introduction

How can we supply water in all plots of a region from a single water pump at a minimum establishment cost of canal networks given that the canal networks can pass along the boundaries of the plots only? Or, how can we supply gas in all regions of a locality from a gas-field at a minimum establishment cost of gas pipelines with the restriction that the gas pipelines can pass along the road network of the locality only? These and many other practical problems involve graph theory. In this thesis we deal with a newly introduced graph theoretical problem namely “minimum face-spanning subgraph problem” to solve these kinds of problems.

In this chapter we discuss the applications and motivations of the problem. We also review the literature about the problem and present the objectives of the thesis. We start with Section 1.1 by giving a precise description of the face-spanning subgraph problem. Section 1.2 describes some practical applications of the problem. Section 1.3 reviews the literature. Section 1.4 addresses the scope of this thesis. In Section 1.5 we present the summary of the thesis.

## 1.1 Problem Statement

In this section we define the face-spanning subgraph problem. Let  $G = (V, E)$  be an edge weighted connected plane graph, where  $V$  and  $E$  are the set of vertices and edges, respectively. Let  $F$  be the set of faces of graph  $G$ . For each edge  $e \in E$ ,  $w(e) \geq 0$  is the weight of the edge  $e$  of  $G$ . A *face-spanning subgraph* of  $G$  is a connected subgraph  $H$  induced by a set of edges  $S \subseteq E$  such that the vertex set of  $H$  contains at least one vertex from the boundary of each face  $f \in F$  of  $G$ . A *minimum face-spanning subgraph*  $H$  of  $G$  is a face-spanning subgraph of  $G$ , where  $\sum_{e \in S} w(e)$  is minimum, and a *minimum face-spanning subgraph problem* asks to find a minimum face-spanning subgraph of a plane graph. In the following section we discuss on some of practical applications of the face-spanning subgraph problem defined above.

## 1.2 Applications

A minimum face-spanning subgraph problem often arises in applications like planning irrigation canal networks for irrigation systems, establishing gas pipelines in a locality, establishing power transmission lines in a city, power wires layout in a complex circuit etc. Finding a minimum face-spanning subgraph is same as finding minimum canal networks for irrigation systems or finding road networks to establish gas pipelines in a locality. Below we describes some details on different applications of the face-spanning subgraph problem and also discuss on how those applications can be modeled using the face-spanning subgraph problem.

### 1.2.1 Planning Gas Pipelines in a Locality

Let a gas company wants to supply gas to a locality from a single gas source. They are allowed to pass the underground gas lines along the road network only, because no one

allows to pass gas lines through the bottom of his building. The road network divides the locality into many regions as illustrated in Figure 1.1(a), where each road is represented by a line segment and a point at which two or more roads meet is represented by a (black or white) small circle. A point at which two or more roads meet is called an intersection point. Each region is bounded by some line segments and intersection points. These regions need to be supplied gas. If a gas line reaches an intersection point on the boundary of a region, then the region may receive gas from the line at that intersection point. Thus the gas lines should reach the boundaries of all the regions of the locality. Gas will be supplied from a gas field which is located outside of the locality and a single pipe line will be used to supply gas from the gas field to an intersection point on the outer boundary of the locality. The gas company wants to minimize the establishment cost of gas lines by selecting the roads for laying gas lines such that the total length of the selected roads is minimum. Since gas will be supplied from the gas field using a single line to the locality, the selected road network should be connected and contains an intersection point on the outer boundary of the locality. Thus the gas company needs to find a set of roads that induces a connected road network, supply gas in all the regions of the locality and the length of the induced road network is minimum. Such a set of roads is illustrated by thick lines in Figure 1.1(b).

The problem mentioned above can be modeled using a face-spanning subgraph as follows. Let  $G = (V, E)$  be an edge weighted connected plane graph, where  $V$  and  $E$  are the set of vertices and edges, respectively. Let  $F$  be the set of faces of graph  $G$ . For each edge  $e \in E$ ,  $w(e) \geq 0$  is the weight of the edge  $e$  of  $G$ . If we represent each road of the road network by an edge of  $G$ , each intersection point by a vertex of  $G$ , each region by a face of  $G$  and assign the length of a road to the weight of the corresponding edge, then the problem of finding a minimum face-spanning subgraph of  $G$  is the same as the problem of finding gas lines in the gas company problem mentioned above.

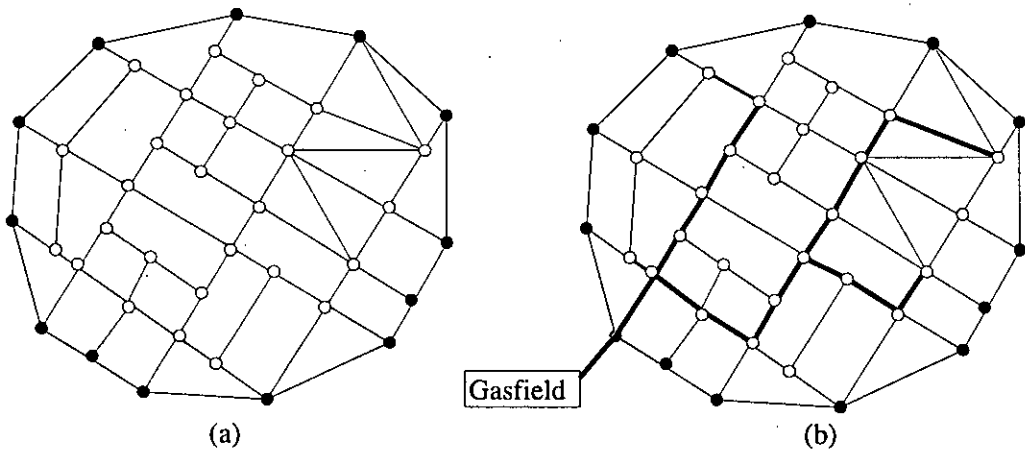


Figure 1.1: (a) A road-network of a locality and (b) a sample setup of gas pipelines drawn by thick lines for supplying gas in all the regions from a gas field.

### 1.2.2 Planning Canal Networks in Irrigation Systems

Let us establish canal networks to irrigate a region from a single water pump. Let the region consists of many plots and each plot has its boundaries. We are allowed to pass the canal networks along the boundaries of the plots only, because no one allows to pass canal networks through the middle of his plots. The boundaries divides the region into many plots as illustrated in Figure 1.2(a), where each boundary is represented by a line segment and a point at which two or more boundaries meet is represented by a small white circle. A point at which two or more boundaries meet is called an intersection point. Each plot is bounded by some line segments and intersection points. These plots of the region need to be supplied water. If a canal network reaches an intersection point on the boundary of a plot, then the plot may receive water from the line at that intersection point. Thus the canal network should reach the boundaries of all the plots of the region. Water will be supplied from a water pump which can be set up in any position on the canal network to supply water in all the plots. We want to minimize the establishment cost of the canal network by selecting the boundaries for laying canals such that the total length of the

selected boundaries is minimum. Since water will be supplied from a water pump to the region, the selected plot boundaries should be connected. Thus we have to find a set of boundaries that induces a connected canal network, supply water in all the plots of the region and the length of the induced canal network is minimum. Such a set of boundaries is illustrated by thick lines in Figure 1.2(b).

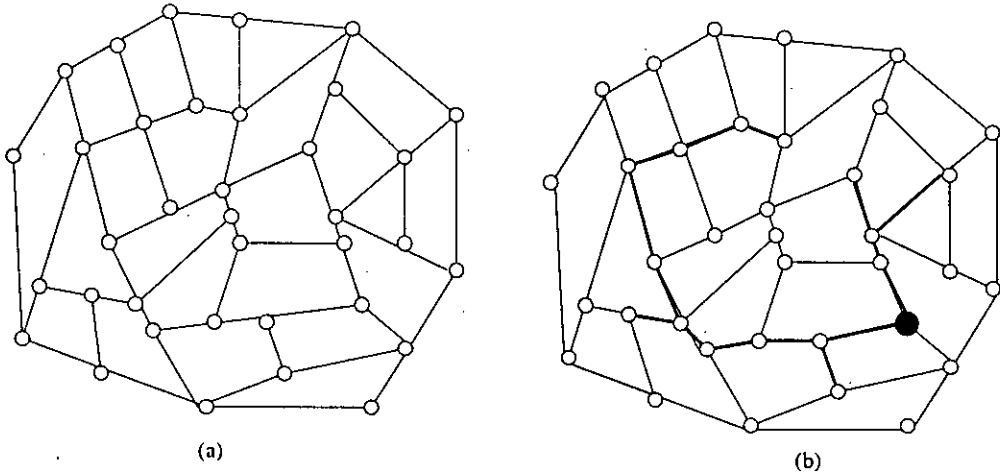


Figure 1.2: (a) Plot boundaries of a region. (b) a sample setup of canal network drawn by thick lines for supplying water in all the plots from a water pump drawn by black circle.

The problem mentioned above can be modeled using a face-spanning subgraph as follows. Let  $G = (V, E)$  be an edge weighted connected plane graph, where  $V$  and  $E$  are the set of vertices and edges, respectively. Let  $F$  be the set of faces of graph  $G$ . For each edge  $e \in E$ ,  $w(e) \geq 0$  is the weight of the edge  $e$  of  $G$ . If we represent each boundary of the plots by an edge of  $G$ , each intersection point by a vertex of  $G$ , each plot by a face of  $G$  and assign the length of a boundary to the weight of the corresponding edge, then the problem of finding a minimum face-spanning subgraph of  $G$  is the same as the problem of finding canal network in irrigation systems mentioned above.

## 1.3 Literature Review

Since the face-spanning subgraph problem is newly introduced, there is no mentionable references on this problem. Anyway, one may think of the “vertex cover problem” [FD04] or the “face cover problem” [AL04] while considering the minimum face-spanning subgraph problem. Unfortunately, the minimum face-spanning subgraph problem is quite different from those two problems. Below we discuss on the differences.

### 1.3.1 Vertex Cover Problem

A vertex set  $C \subseteq V$  is called a *vertex cover* if every edge of  $G$  is incident to some vertex in  $C$  and the *vertex cover problem* asks to compute a minimum vertex cover in given  $G$ . Thus the vertex cover problem asks to find a vertex set which contains at least one vertex from the end vertices of each edge of  $G$  whereas the minimum face-spanning subgraph problem asks to find an edge set which contains at least one vertex from the boundaries of each face of  $G$ . Hence, the vertex cover problem and the minimum face-spanning subgraph problem are different. A simple graph  $G$  is drawn in Figure 1.3. We see that the vertex set  $\{v_4, v_{10}, v_8, v_{11}\}$  of the face-spanning subgraph of  $G$  drawn by thick lines in Figure 1.3 does not contain at least one vertex from each edge of  $G$ . For example, the vertex set of the face-spanning subgraph in Figure 1.3 does not contain any vertex from the edges like  $(v_1, v_2)$ ,  $(v_5, v_6)$  etc. Hence the face-spanning subgraph does not provide any solution to the vertex cover problem. Thus, the vertex cover problem and the minimum face-spanning subgraph problem are different.

### 1.3.2 Face Cover Problem

A set of faces whose boundaries contain all the vertices in a plane graph  $G$  is said to be a *face cover* for  $G$  and the *face cover problem* asks to compute a minimum face cover in given  $G$ . Thus the face cover problem asks to find a face set which contains all the vertices





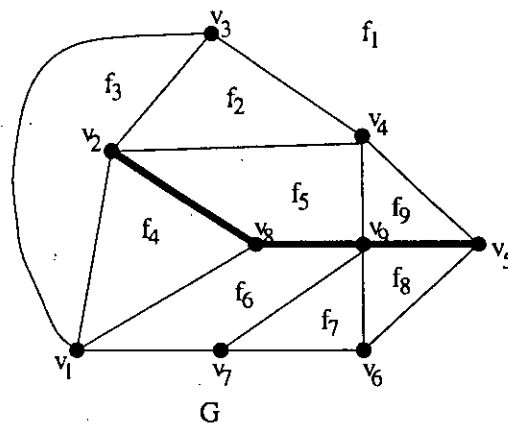


Figure 1.4: A simple graph  $G$  with a face-spanning subgraph drawn by thick lines.

it is unlikely to have a polynomial-time algorithm for finding a minimum face-spanning subgraph of a plane graph. We also show the same scenario for a variation of the face-spanning subgraph problem called “minimum-vertex face-spanning problem”. In such a case, design of approximation algorithms is needed for practical applications.

### 1.4.1 Face-Spanning Subgraph Problem

We show the hardness of the face-spanning subgraph problem in this thesis, that means, we show that the face-spanning subgraph problem belongs to the infamous class of  $NP$ -complete problems. First, we show that the face-spanning subgraph problem is in  $NP$ . For this, we prove that any candidate solution of the face-spanning subgraph problem can be verified in polynomial time. To prove the face-spanning subgraph problem is  $NP$ -hard, “weighted tree cover problem” [AHH93] will be polynomially transformed into the face-spanning subgraph problem. These above two steps will immediately prove that the face-spanning subgraph problem is  $NP$ -complete.

### 1.4.2 Minimum-Vertex Face-Spanning Subgraph Problem

In this thesis we also introduce a variation of the face-spanning subgraph problem namely “minimum-vertex face-spanning subgraph problem”. Let  $G = (V, E)$  be a connected plane graph, where  $V$  and  $E$  are the set of vertices and edges, respectively, and let  $F$  be the set of faces of graph  $G$ . A *minimum-vertex face-spanning subgraph*  $H$  of  $G$  is a face-spanning subgraph of  $G$  where  $|V(H)|$  is minimum. A *minimum-vertex face-spanning subgraph problem* asks to find a minimum-vertex face-spanning subgraph of a plane graph. The minimum-vertex face-spanning subgraph problem often arises in applications like establishing base transceiver stations in wireless networks, establishing power distribution centers in a city etc, where the setup cost for each establishment is huge. In these cases the objective is to minimize the number of vertices instead of edge cost.

In this thesis we show that the minimum-vertex face-spanning subgraph problem is *NP*-complete. First, to show the minimum-vertex face-spanning subgraph problem is in *NP*, we prove that any candidate solution of the minimum-vertex face-spanning subgraph problem can be verified in polynomial time. To prove the minimum-vertex face-spanning subgraph problem is *NP*-hard, “connected vertex cover problem” [GJ77] will be polynomially transformed into the minimum-vertex face-spanning subgraph problem. These above two steps will immediately prove that the minimum-vertex face-spanning subgraph problem is *NP*-complete.

### 1.4.3 Approximation Algorithms

Since the face-spanning subgraph problem and the minimum-vertex face-spanning subgraph problem arises from numerous practical applications and both the problems are *NP*-complete, design of efficient approximation algorithms is essential. In this thesis we present the approximation algorithms for the face-spanning subgraph problem and the

minimum-vertex face-spanning subgraph problem. We show that the time complexities of the presented approximation algorithms is linear. We also present the approximation ratios of the designed approximation algorithms.

## 1.5 Summary

The main result of this thesis are as follows.

1. The face-spanning subgraph problem is *NP*-complete.
2. The minimum-vertex face-spanning subgraph problem is *NP*-complete.
3. We have developed a linear time algorithm for finding a minimal face-spanning subgraph.
4. We have calculated the upper bound and lower bound on the number of vertices for a minimal face-spanning subgraph. Let  $G$  be a plane graph of  $n$  vertices with maximum degree  $\Delta$ . Let  $n_0$  be the number of outer vertices and  $f$  be the number of faces of  $G$ . Then a minimal face-spanning subgraph of  $G$  contains at most  $n - n_0 + 1$  vertices and at least  $(f - 2)/(\Delta - 2)$  vertices. We show that the upper bound and the lower bound are tight.
5. We have developed approximation algorithms for solving the face-spanning subgraph problem and the minimum-vertex face-spanning subgraph problem. The approximation ratio of the developed approximation algorithms for the face-spanning subgraph problem and the minimum-vertex face-spanning subgraph problem are  $\{(n - n_0)(\Delta - 2)e_{max}\}/\{(f - \Delta)e_{min}\}$  and  $2(\Delta - 2)$ , respectively, where  $e_{max}$  and  $e_{min}$  denote the maximum and minimum weight of the edges of  $G$ . The time complexities of the approximation algorithms are linear.

The thesis is organized as follows. Chapter 2 describes preliminary definitions on graph, complexity theory and approximation algorithms. Chapter 3 proves that the face-spanning subgraph problem is *NP*-complete. Chapter 4 shows that the minimum-vertex face-spanning subgraph problem is also *NP*-complete. Chapter 5 discusses a linear time algorithm for finding a minimal face-spanning subgraph of a plane graph. Approximation algorithms for finding solutions of the face-spanning subgraph problem and the minimum-vertex face-spanning subgraph problem along with the approximation ratio and complexity analysis are presented in Chapter 5. Finally Chapter 6 gives the conclusion.

# Chapter 2

## Preliminaries

In this chapter, we define some basic definitions of graphs. Some discussion on complexity theory and approximation algorithms will also be presented in this chapter. Definitions that are not given here are discussed as they are needed. In Section 2.1, we start by giving the definitions of some basic terms of graph which are related to and used through out this thesis. Section 2.2 describes the terms related to the computational complexities. In Section 2.3 we discuss on approximation algorithm and the approximation ratio. Section 2.5 illustrates on the “connected vertex cover problem” that will be used to prove the hardness of the face-spanning subgraph problem. Section 2.4 defines the “weighted tree cover problem” that will be used to prove the hardness of the minimum-vertex face-spanning subgraph problem. Finally, Section 2.6 summarizes this chapter.

### 2.1 Basic Terminology

In this section we give some definitions of standard graph-theoretical terms used throughout the remainder of this thesis.

### 2.1.1 Graphs

Let  $G = (V, E)$  be a graph with vertex set  $V$  and edge set  $E$ . The number of vertices of  $G$  is denoted by  $n$ , that is,  $n = |V|$ , and the number of edges of  $G$  is denoted by  $m$ , that is,  $m = |E|$ . We often denote the set of vertices of  $G$  by  $V(G)$  and the set of edges of  $G$  by  $E(G)$ . We denote an edge joining vertices  $v_i, v_j$  of  $G$  by  $(v_i, v_j)$ . If  $(v_i, v_j) \in E$ , then two vertices  $v_i, v_j$  are said to be *adjacent* in  $G$ ; edge  $(v_i, v_j)$  is then said to be *incident* to vertices  $v_i$  and  $v_j$ ;  $v_i$  is a *neighbor* of  $v_j$ . A *loop* is an edge whose endpoints are equal. *Parallel edges* or *multiple edges* are edges that have the same pair of endpoints. A *simple graph* is a graph having no loops or multiple edges. The graph in which loops and multiple edges are allowed is called a *multi graph*. Figure 2.1 depicts a simple graph  $G$ , where each vertex in  $V(G) = \{v_1, v_2, v_3, v_4, v_5\}$  is drawn by small black circle and each edge in  $E(G) = \{(v_1, v_2), (v_2, v_3), (v_3, v_4), (v_4, v_1), (v_1, v_5), (v_2, v_5), (v_3, v_5), (v_4, v_5)\}$  is drawn by a line segment.

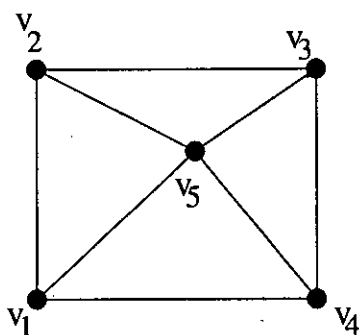


Figure 2.1: A simple graph with five vertices and eight edges.

The *degree* of a vertex  $v$  in a graph  $G$  is the number of edges incident to  $v$  in  $G$ . The degree of a vertex  $v$  is denoted by  $d(v)$  and the *maximum degree* of  $G$  is denoted by  $\Delta(G)$  or simply by  $\Delta$ . In Figure 2.1, the degree  $d(v_1)$  of vertex  $v_1$  is 3 and the maximum degree  $\Delta$  of  $G$  is 4.

A *path* in  $G$  is an ordered list of distinct vertices  $(v_1, v_2, \dots, v_{q-1}, v_q) \in V$  such that  $(v_{i-1}, v_i) \in E$  for all  $2 \leq i \leq q$  [W01]. A path is closed if  $v_1 = v_q$ . A *closed path* containing at least one edge is called a *cycle*. In Figure 2.1,  $(v_1, v_2, v_3)$  is a path and  $(v_1, v_2, v_5, v_1)$  is a cycle.

### 2.1.2 Connectivity

A graph  $G$  is *connected* if for any two distinct vertices  $v_i, v_j$  of  $G$  there is path between  $v_i$  and  $v_j$  in  $G$ . A graph which is not connected is called a *disconnected graph*. A *connected component* of a graph is a maximal connected subgraph. The graph in Figure 2.2(a) is connected since there is path in any two distinct vertices of the graph. On the other hand, the graph in Figure 2.2(b) is disconnected since there is no path between  $v_1$  and  $v_2$ . The graph in Figure 2.2(b) has two connected components  $G_1$  and  $G_2$  indicated by dotted lines.

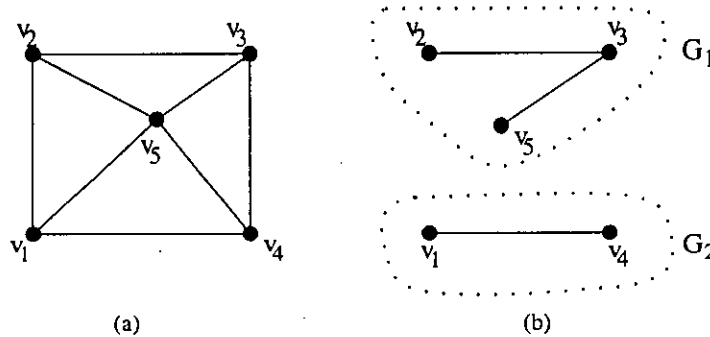


Figure 2.2: (a) A connected graph, and (b) a disconnected graph with two connected components:

### 2.1.3 Trees and Subgraphs

$G$  is a *tree* if  $G$  is connected and has no cycle.  $H = (V', E')$  is called a *subgraph* of  $G$  if  $V' \subseteq V$  and  $E' \subseteq E$ . A subgraph  $H = (V', E')$  of  $G$  is called a *spanning subgraph* of  $G$



if  $H$  contains all the vertices  $V'$  of  $G$ . A spanning subgraph  $H$  of  $G$  is called a *spanning tree* of  $G$  if  $H$  contains no cycle. A subgraph  $H = (V', E')$  of  $G$  is called the *edge induced subgraph* of  $G$  induced by the edge set  $E'$  if  $V'$  contains only the vertices of  $G$  which are end vertices of the edges in  $E'$ . Figure 2.3(b) illustrates an edge induced subgraph induced by the edge set  $E' = \{(v_1, v_2), (v_1, v_5), (v_5, v_6)\}$  of the graph in Figure 2.3(a). For a set of edges  $S \subseteq E$ , we denote by  $V(S)$  the set of vertices consisting of the end vertices of the edges in  $S$ , that means,  $V(S)$  is the set of vertices of the edge induced subgraph of  $G$  induced by  $S$ . Figure 2.3(b) illustrates that the edge induced subgraph of  $G$  induced by  $S = \{(v_1, v_2), (v_1, v_5), (v_5, v_6)\}$  contains the vertex set  $V(S) = \{v_1, v_2, v_5, v_6\}$ .

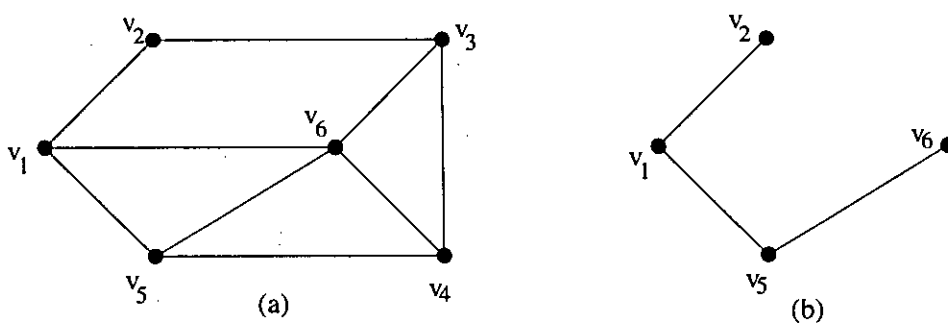


Figure 2.3: (a) A graph with six vertices and nine edges, and (b) an edge-induced subgraph induced by edge set  $\{(v_1, v_2), (v_1, v_5), (v_5, v_6)\}$ .

### 2.1.4 Planar Graphs and Plane Graphs

A graph is *planar* if it can be embedded in the plane so that no two edges intersect geometrically except at a vertex to which they are both incident. A *plane graph* is a planar graph with a fixed embedding. A plane graph divides the plane into connected regions called *faces*. Below we emphasize on some definitions usually used in the following chapters.

Let  $G = (V, E)$  be an edge weighted connected plane graph, where  $V$  and  $E$  are the

set of vertices and edges, respectively. Let  $F$  be the set of faces of plane graph  $G$ . We say a set of edges  $S \subseteq E$  covers a face  $f \in F$  if  $V(S)$  contains at least one vertex from the vertices on the boundary of  $f$ . *Edge weighted connected plane graph* is a connected plane graph where each edge  $e$  has a weight  $w(e) \geq 0$ . Let  $G$  be an edge weighted connected plane graph. Then for a subgraph  $H$  of  $G$ , the *cost of  $H$*  is computed as  $\sum_{e \in E(H)} w(e)$ . We say a vertex set  $V' \subseteq V$  covers all the edges of  $G$  if  $V'$  contains at least one vertex from the end vertices of each edge of  $G$ .  $H$  is a *tree cover* of  $G$  if  $H$  is a tree in  $G$  and  $V(H)$  covers all the edges of  $G$ .

## 2.2 Complexity Theory

In computer science, computational complexity theory is the branch of the theory of computation that studies the resources, or cost, of the computation required to solve a given computational problem. This cost is usually measured in terms of abstract parameters such as time and space, called computational resources. Time represents the number of steps it takes to solve a problem and space represents the quantity of information storage required or how much memory it takes. There are often trade offs between time and space that have to be considered when trying to solve a computational problem. It often turns out that an alternative algorithm will require less time but more space (or vice versa) to solve a given problem. Hence in the complexity theory we can classify problems based on how difficult they are to solve. In this section we discuss on different classes of problems.

### 2.2.1 Decision Problems and Optimization Problems

Much of complexity theory deals with decision problems. A *decision problem* is a problem where the answer is always YES or NO. One example of a decision problem is “given two numbers  $x$  and  $y$ , does  $x$  evenly divide  $y$ ?”. This is a yes or no question, and its answer depends on the values of  $x$  and  $y$ . An algorithm for this decision problem would tell how

to determine whether  $x$  evenly divides  $y$ , given  $x$  and  $y$ . Another example of a decision problem related to the shortest-path problem is, “Given a graph  $G = (V, E)$ , two vertices  $u, v \in V$ , and a nonnegative integer  $K$ , does a path exist in  $G$  between  $u$  and  $v$  whose length is at most  $K$ ?” Here the answer is obviously either yes or no.

In computer science, an *optimization problem* is the problem to find among all feasible solutions for some problem the best one [CCPS98]. One example of an optimization problem related to the shortest-path problem is, “Given a graph  $G = (V, E)$  and two vertices  $u, v \in V$ . Find the shortest path between  $u$  and  $v$  in  $G$ ”. In fact, the decisive version of an optimization problem is a decision problem.

### 2.2.2 Complexity Classes

A *complexity class* is the set of all of the computational problems which can be solved using a certain amount of a certain computational resource. There are different complexity classes in complexity theory. Below we briefly discuss on the commonly used different complexity classes in the computational complexity theory.

#### The Complexity Class $P$

A problem is assigned to the  $P$  (*polynomial time*) class if there exists at least one algorithm to solve that problem, such that the number of steps of the algorithm is bounded by a polynomial in  $n$ , where  $n$  is the length of the input. This complexity class includes the set of decision problems that can be solved by a deterministic machine in polynomial time. More explicitly, a problem is polynomial-time solvable if there exists an algorithm to solve it in time  $O(n^k)$  for some constant  $k$ , where  $n$  is the length of the input.

#### The Complexity Class $NP$

In computational complexity theory,  $NP$  (*Non-deterministic polynomial time*) is the set of decision problems solvable in polynomial time on a non-deterministic turing machine. We

assign a problem to the  $NP$  class if it is solvable in polynomial time by a non-deterministic turing machine. We can identify that whether a problem is in  $NP$  or not, by verifying a candidate solution of the problem in polynomial-time by deterministic turing machine [GJ79].

Since the polynomial problems are solvable by deterministic turing machine, they are also solvable by non-deterministic turing machine. Hence  $P \subset NP$ . Figure 2.4 illustrates the relation of  $P$  and  $NP$ . The question which arises now is whether  $P = NP$  or not. The question of whether  $P$  is the same set as  $NP$  is the most important open question in theoretical computer science. Questions like this motivate the concepts of  $NP$ -hard and  $NP$ -complete which we discuss below.

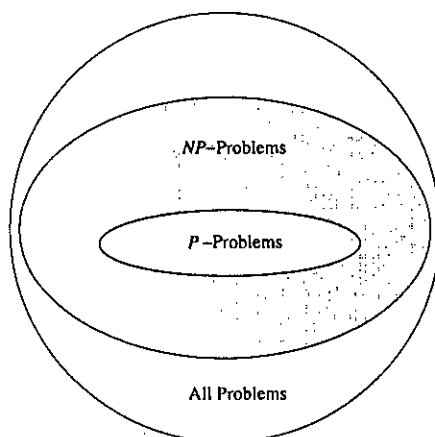


Figure 2.4: The set of all problems that contains the  $P$  and  $NP$  problems.

### The Complexity Class $NP$ -hard

In computational complexity theory, a *reduction* is a transformation of one problem into another problem. Intuitively, if problem  $A$  is *reducible* to problem  $B$ , then a solution to  $B$  gives a solution to  $A$ . Thus, solving  $A$  cannot be harder than solving  $B$ . A problem  $X$  is  $NP$ -hard if all the problems in  $NP$  are polynomially reducible to  $X$ . Figure 2.5 illustrates the relation of different classes of problems.

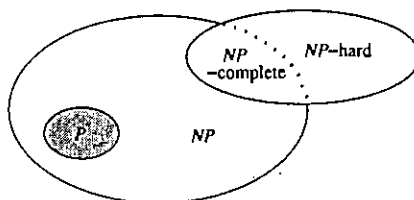


Figure 2.5: Illustration of the relationship among  $P$ ,  $NP$ ,  $NP$ -hard and  $NP$ -complete problems.

### The Complexity Class $NP$ -complete

A problem  $X$  is  $NP$ -complete if  $X$  is in  $NP$  and  $X$  is  $NP$ -hard. Figure 2.5 illustrates the relationship among  $P$ ,  $NP$ ,  $NP$ -hard and  $NP$ -complete problems where we see that the part of the  $NP$ -hard problems that are in  $NP$ , are the  $NP$ -complete problems.  $NP$ -complete problems are computationally intractable based on the following properties of this class [GJ77]:

1. There is no polynomial-time algorithm that can solve any problem in the class.
2. Despite the wide variety and large number of problems in the class, the existence of a polynomial-time algorithm for any one of them would imply that every problem in the class could be solved with a polynomial-time algorithm.

Detailed discussion on the class of  $NP$ -complete problems and its members are described in many references [AHU74, K72, GJ79]. The reader is referred to [AHU74, GJ79] for a thorough description of the formal requirements for a proof of  $NP$ -completeness. In short, to prove a particular problem  $X$  is  $NP$ -complete, the following two steps are required:

1. Prove that  $X$  is in  $NP$ , that is,  $X$  can be solved in polynomial time by a non-deterministic turing machine. In other words, it can be construed as any solution of  $X$  can be verified in polynomial time.

2. Prove that  $X$  is  $NP$ -hard. For this we have to prove that, some known  $NP$ -complete problem  $X'$  can be polynomially transformed into  $X$  in such a way that any polynomial-time algorithm for solving  $X$  could be used to solve  $X'$  in polynomial time.

## 2.3 Approximation Algorithm and Approximation Ratio

It is unlikely to have an efficient algorithm for a  $NP$ -complete problem, that is, it is unlikely to have an optimal solution of the  $NP$ -complete problem in polynomial time. But some  $NP$ -complete problem arises from numerous practical applications like the face-spanning subgraph problem. Hence an algorithm is required that can quickly provide a sub-optimal solution of the problem. An algorithm that finds a sub-optimal solution of a problem in polynomial time within a certain range of the optimal solution is known as *approximation algorithm* [CLR90]. An *approximation ratio* is the measure of goodness of an approximation algorithm. Approximation ratio is generally denoted by  $\rho(n)$  where  $n$  is the input size of the problem. An approximation algorithm that achieves approximation ratio  $\rho(n)$  is known as  $\rho(n)$ -approximation algorithm of the problem. If  $C$  is the cost of the suboptimal solution of a problem produced by an approximation algorithm and  $C^*$  is the optimal cost of the optimal solution of the problem, then

$$\max\left\{\frac{C}{C^*}, \frac{C^*}{C}\right\} \leq \rho(n) \quad (2.1)$$

This definition applies for both the minimization and maximization problems. For a maximization problem,  $0 < C \leq C^*$ , and the ratio  $C/C^*$  gives the ratio by which the cost of an optimal solution is larger than the cost of the approximate solution. Similarly, for a minimization problem,  $0 < C^* \leq C$ , and the ratio  $C^*/C$  gives the ratio by which the cost of an approximate solution is larger than the cost of the optimal solution. Since all

solutions are assumed to have positive cost, these ratios are always well defined. The ratio of an approximation algorithm is never less than a 1, since  $C/C^* < 1$  implies  $C^*/C > 1$ . An optimal algorithm has a ratio 1, and an approximation algorithm with a large ratio may return a solution that is very much worse than optimal [AHU74].

## 2.4 Weighted Tree Cover Problem

In this section we discuss another *NP*-complete problem called “weighted tree cover problem” [AHH93]. We use this problem to prove that the face-spanning subgraph problem is *NP*-complete.

Let  $G = (V, E)$  be an edge weighted connected plane graph, where  $V$  and  $E$  be the set of vertices and edges respectively. Each edge  $e \in E$  has a weight  $w(e) \geq 0$ . A subgraph  $H$  of  $G$  is a *tree cover* of  $G$  if  $H$  is a tree in  $G$  and  $V(H)$  cover all the edges of  $G$ . We say  $H$  be *weighted tree cover* of  $G$  if we compute  $H$  based on the weight of the edges of  $G$ .

A weighted tree cover  $H$  of a plane graph  $G$  is a *minimum weighted tree cover* if  $\sum_{e \in E(H)} w(e)$  is minimum among all the weighted tree cover in  $G$ . It has already been proved that it is unlikely to design an efficient algorithm to find a minimum weighted tree cover of a plane graph. A *weighted tree cover problem* asks to find a minimum weighted tree cover of a plane graph. The formal definition of weighted tree cover problem is as follows [AHH93]:

**Definition 2.4.1** (Weighted Tree Cover Problem) *Given a plane graph  $G = (V, E)$  and weight on the edges  $w(e) \geq 0$  for all  $e \in E$  and a positive real number  $B$ , does there exist any weighted tree,  $T \subseteq (V', E')$  induced by  $E' \subseteq E$ , whose vertices  $V' \subseteq V$  cover all the edges in  $E$  and  $\sum_{e \in E'} w(e) \leq B$ ?*

## 2.5 Connected Vertex Cover Problem

In this section we discuss a known *NP* complete problem called “connected vertex cover problem” [GJ77]. We use this problem to prove that the minimum-vertex face-spanning subgraph problem is *NP*-complete.

Let  $G = (V, E)$  be a connected plane graph, where  $V$  and  $E$  be the set of vertices and edges respectively. We say a vertex set  $V' \subseteq V$  is a *vertex cover* of  $G$  if  $V'$  contains at least one vertex from the end vertices of each edge  $e \in E$  of  $G$ . Figure 2.6(a) shows a connected plane graph where the vertex set  $V' = \{v_1, v_3, v_5, v_6\}$  is a vertex cover of that graph. A vertex cover  $V'$  is *connected* if the subgraph induced by  $V'$  is connected. A subgraph  $H$  of  $G$  be a *connected vertex cover* of  $G$  if  $V(H)$  is a vertex cover of  $G$  and the subgraph  $H$  is connected. In Figure 2.6(b), the subgraph drawn by thick lines is a connected vertex cover where the vertex set of the subgraph is  $\{v_2, v_4, v_5\}$ . In some cases, though a vertex set  $V' \subseteq V$  be a vertex cover of graph  $G$ , the subgraph induced by  $V'$  may not be a connected vertex cover of  $G$ . For example, the vertex set  $V' = \{v_1, v_3, v_5, v_6\}$  in Figure 2.6(a) is a vertex cover, but the subgraph induced by  $V'$  is not a connected vertex cover since  $V'$  induces a disconnected subgraph.

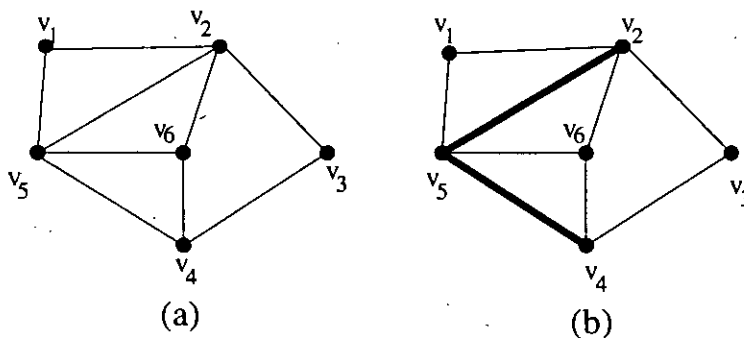


Figure 2.6: (a) A connected plane graph, and (b) a connected plane graph with a connected vertex cover drawn by thick lines.



A connected vertex cover  $H$  of a plane graph  $G$  is a *minimum connected vertex cover* if  $|V(H)|$  is minimum in  $G$ . A *connected vertex cover problem* asks to find a minimum connected vertex cover of a plane graph. The formal definition of connected vertex cover problem is as follows [GJ77].

**Definition 2.5.1** (Connected Vertex Cover Problem) *Given a plane graph  $G = (V, E)$  and an integer  $K$ , does there exist a vertex cover  $V' \subseteq V$  satisfying  $|V'| \leq K$  and the subgraph induced by  $V'$  is connected?*

## 2.6 Summary

In this chapter we discussed on some definitions of standard graph-theoretical terms. We have discussed on different complexity classes. Basic concepts on approximation algorithms, approximation ratios are given in this chapter. We have also discussed on two *NP*-complete problems connected vertex cover problem and weighted tree cover problem that will be used in the following chapters.

## Chapter 3

# Face-Spanning Subgraph Problem

In this chapter we present our main result on the face-spanning subgraph problem. We show that the face-spanning subgraph problem belongs to the *NP*-complete class. The organization of this chapter is as follows. Section 3.1 formally defines the face-spanning subgraph problem. In Section 3.2 we prove the *NP*-completeness of the face-spanning subgraph problem. Finally, Section 3.3 summarizes this chapter. The approximation algorithm to find a minimum face-spanning subgraph of a plane graph is discussed in Chapter 5.

### 3.1 Problem Definition

In this section we formally define the face-spanning subgraph problem. Let  $G = (V, E)$  be an edge weighted connected plane graph, where  $V$  and  $E$  are the set of vertices and edges, respectively. Let  $F$  be the set of faces of graph  $G$ . For each edge  $e \in E$ ,  $w(e) \geq 0$  be the weight of the edge  $e$  of  $G$ . A face-spanning subgraph of  $G$  is a connected subgraph  $H$  induced by a set of edges  $S \subseteq E$  such that the vertex set of  $H$  contains at least one vertex from the boundary of each face  $f \in F$  of  $G$ . Figure 3.1 shows two face-spanning subgraphs drawn by thick lines.

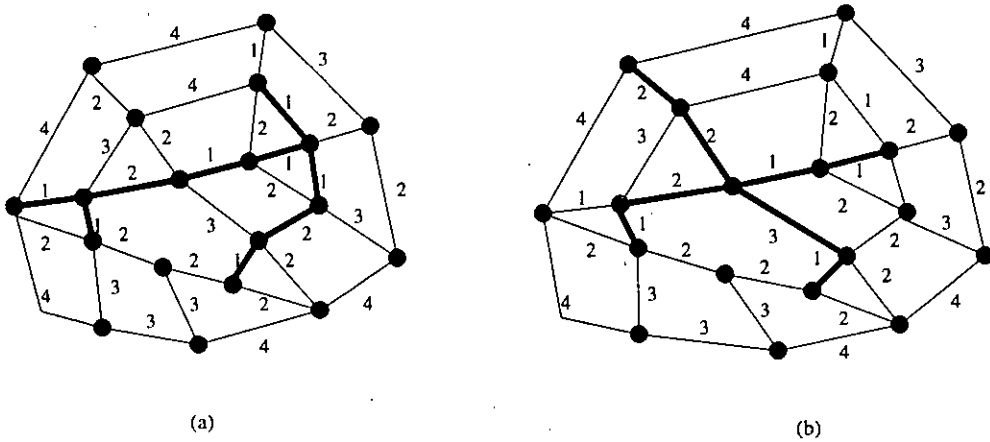


Figure 3.1: A simple graph with (a) a face-spanning subgraph of cost 11 and (b) a face-spanning subgraph of cost 13.

A minimum face-spanning subgraph  $H$  of  $G$  is a face-spanning subgraph of  $G$ , where  $\sum_{e \in S} w(e)$  is minimum, and a face-spanning subgraph problem asks to find a minimum face-spanning subgraph of a plane graph. We say the face-spanning subgraph problem as “FSSP” in short in many places rest of this chapter. The formal definition of the face-spanning subgraph problem is as follows:

**Definition 3.1.1** (Face-Spanning Subgraph Problem) *Let  $G = (V, E)$  be a connected plane graph, where  $V$  and  $E$  are the set of vertices and edges, respectively, and let  $F$  be the set of faces of graph  $G$ . Let  $w(e) \geq 0$  be a positive real number assigned to edge  $e$  as weight for every edge  $e \in E$ . Then is there any set  $S \subseteq E$  such that the subgraph  $H$  induced by  $S$  is connected, cover all faces of  $G$  and the cost of  $H$  is  $\leq B$ , for a given positive real number  $B$ ?*

## 3.2 *NP*-completeness of FSSP

In this section we prove that it is unlikely to design an efficient algorithm for finding a minimum face-spanning subgraph of a plane graph by showing that the face-spanning subgraph problem belongs to the infamous class of *NP*-complete problems. For this, we prove the following theorem.

**Theorem 3.2.1** *The face-spanning subgraph problem is NP-complete.*

To prove the Theorem 3.2.1, we have to show that (i) the face-spanning subgraph problem is in *NP* and (ii) the face-spanning subgraph problem is *NP*-hard. In Subsection 3.2.1, we prove that the face-spanning subgraph problem is in *NP* and in Subsection 3.2.2, we prove that the face-spanning subgraph problem is *NP*-hard. These two steps immediately prove the Theorem 3.2.1. We use the well known *NP*-complete problem weighted tree cover problem to prove the hardness of the face-spanning subgraph problem.

### 3.2.1 FSSP is in *NP*

*NP* is the set of decision problems solvable in polynomial time on a non-deterministic turing machine. We can identify that whether a problem is in *NP* or not, by verifying a candidate solution of the problem in polynomial-time by deterministic turing machine.

In this subsection we prove the first step of the proof of *NP*-completeness of the face-spanning subgraph problem, that is, the face-spanning subgraph problem is in *NP*. Let  $G = (V, E)$  be an edge weighted connected plane graph, where  $V$  and  $E$  are the set of vertices and edges, respectively. Let  $F$  be the set of faces of graph  $G$ . For each edge  $e \in E$ ,  $w(e) \geq 0$  is the weight of the edge  $e$  of  $G$ . Thus we have the following lemma.

**Lemma 3.2.2** *The face-spanning subgraph problem is in NP.*

**Proof.** To prove that the face-spanning subgraph problem is in *NP*, it is sufficient to prove that for a given edge set  $S \subseteq E$  of  $G$ , we can verify in polynomial-time that the

subgraph  $H$  induced by  $S$  (i) is connected, (ii) cover all faces of  $G$  and (iii) the cost of  $H$  is  $\leq B$ .

(i) Connectivity of the subgraph  $H$  induced by  $S$  can be checked using DFS in linear time.

(ii) We can verify whether  $S$  covers all faces of  $G$  or not in linear time by the following method.

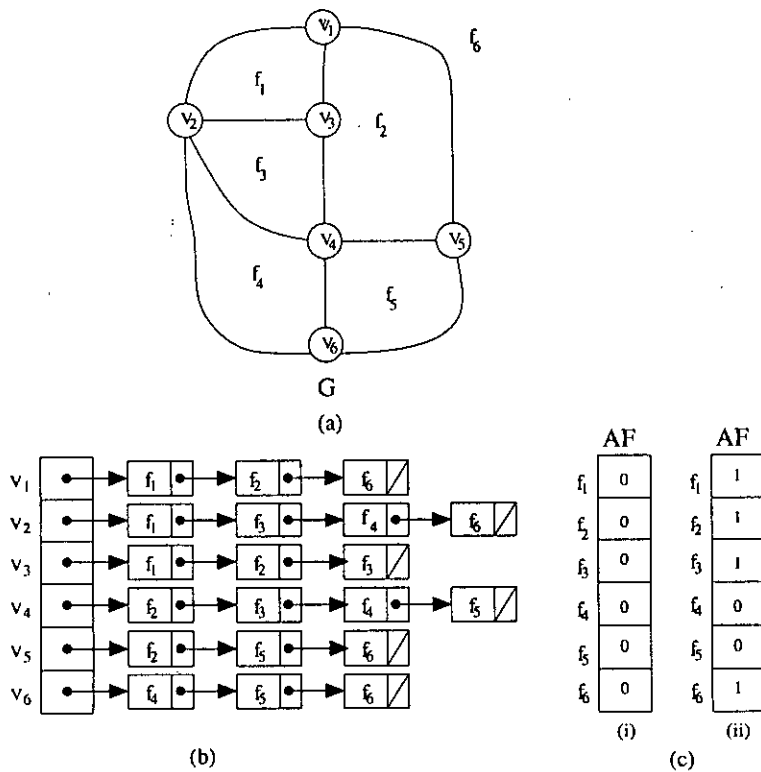


Figure 3.2: (a) A graph  $G$ , (b) face-list for each vertex and (c) the boolean array  $AF$  (i) after initialization and (ii) after checking the vertices of set  $V(S) = \{v_1, v_3\}$

Let  $F(v)$  be the set of faces of  $G$  such that each face in  $F(v)$  contains the vertex  $v$ . We maintain a face-list for each vertex  $v$  as illustrated in Figure 3.2(b), where the face-list for  $v$  contains the faces in  $F(v)$ . In the graph in Figure 3.2(a),  $F(v_1)$  contains

the faces  $f_1, f_2$  and  $f_6$  and the faces  $f_1, f_2$  and  $f_6$  have appeared in the face-list for the vertex  $v_1$  as illustrated in Figure 3.2(b). We also maintain a boolean array  $AF$  of length  $|F|$  to indicate whether the faces of  $G$  are covered by the vertices in  $V(S)$  or not. For all  $j \in \{1, 2, \dots, |F|\}$ ,  $AF[j]$  corresponds to the face  $f_j$  of graph  $G$ . Initially all elements of  $AF$  are set to 0 as shown in Figure 3.2(c)(i) to indicate that no face is covered by the vertices in  $V(S)$  initially. We traverse the face-list for each vertex  $v$  in  $V(S)$  and for each face  $f_j$  in the face-list, we change the value of  $AF[j]$  to 1 to indicate that the face  $f_j$  is covered by the vertices in  $V(S)$ . As an example let us consider a set  $V(S) = \{v_1, v_3\}$ . Figure 3.2(b) shows that  $F(v_1)$  contains the faces  $f_1, f_2$  and  $f_6$ . Hence we set value 1 to  $AF[1], AF[2]$  and  $AF[6]$  as shown in Figure 3.2(c)(ii). Similarly, since  $F(v_3)$  contains the faces  $f_1, f_2$  and  $f_3$ ,  $AF[1], AF[2]$  and  $AF[3]$  are set to 1 as shown in Figure 3.2(c)(ii). After traversing the face-lists for all vertex in  $V(S)$ , we check the array  $AF$  to know that whether all faces of  $G$  are covered or not.

We now calculate the complexity of the method described above. Since  $|F(v)|$  is equal to the degree of  $v$  and  $|V(S)|$  is at most  $|V|$ , then, to check all the vertices of  $V(S)$ , we have to consider at most  $\sum_{v \in V(S)} d(v) = 2m = O(m) = O(n)$  entries in total. Since the length of array  $AF$  is equal to  $|F|$ , the traversing time of  $AF$  is  $O(|F|) = O(n)$ . Thus, the overall time complexity to verify that whether the vertices in  $V(S)$  cover all faces of  $G$  or not is  $O(n)$ .

(iii) It can be verified easily in  $O(n)$  time that the cost of  $H$  is  $\leq B$ .

Since it is possible to verify (i), (ii) and (iii) in polynomial time, the face-spanning subgraph problem is in  $NP$ . □

### 3.2.2 FSSP is $NP$ -hard

A reduction is a transformation of one problem into another problem. Intuitively, if problem  $A$  is reducible to problem  $B$ , then a solution to  $B$  gives a solution to  $A$ . Thus, solving  $A$  cannot be harder than solving  $B$ . We say a problem  $X$  is  $NP$ -hard if all the

problems in  $NP$  are polynomially reducible to  $X$ .

We now prove the second part of the proof of  $NP$ -completeness of the face-spanning subgraph problem, that is, the face-spanning subgraph problem is  $NP$ -hard. Thus we have the following lemma.

**Lemma 3.2.3** *The face-spanning subgraph problem is  $NP$ -hard.*

To prove lemma 3.2.3 we prove that, the  $NP$ -complete problem weighted tree cover problem defined in Section 2.4 can be polynomially transformed into the face-spanning subgraph problem in such a way that any polynomial-time algorithm for solving the face-spanning subgraph problem could be used to solve the weighted tree cover problem in polynomial time.

Let  $G = (V, E)$  be a connected plane graph, where  $V$  and  $E$  are the set of vertices and edges, respectively. Let  $F$  be the set of faces of graph  $G$ . We obtain a graph  $G'$  from  $G$  as follows. For each edge  $e = (v_k, v_l) \in E$  we add a vertex  $v_e$  and two edges  $(v_k, v_e)$  and  $(v_l, v_e)$  to  $G$ . More formally,  $G' = (V', E')$  where  $V' = V \cup V_e$ ,  $V_e = \{v_e | e \in E\}$  and  $E' = E \cup E_e$  where  $E_e = \{(v_e, v_k), (v_e, v_l) | \{e = (v_k, v_l)\} \in E\}$ . In  $G'$  we call a vertex in  $V_e$  a *new vertex*, a vertex in  $V$  an *original vertex*, an edge in  $E_e$  a *new edge* and an edge in  $E$  an *original edge*. Note that original vertices and original edges of  $G'$  are also the vertices and edges of  $G$ . We assign the cost  $w(e)/2$  to each of the edges  $(v_e, v_k)$  and  $(v_e, v_l)$  for all  $e = (v_k, v_l) \in E$ . Figure 3.3 illustrates the construction of  $G'$  where the vertices drawn by white small circles are new vertices, the edges drawn by dashed lines are new edges, the vertices drawn by black circles are original vertices and the edges drawn by solid lines are the original edges of  $G'$ . If  $G$  has  $n$  vertices and  $m$  edges, then  $G'$  has  $n + m$  vertices and  $3m$  edges. Clearly  $G'$  can be constructed in  $O(n)$  time. One can easily observe that the graph  $G'$  is planar as illustrated in Figure 3.3, where a plane embedding of  $G'$  is shown. Throughout the paper we consider  $G'$  as a plane embedding of the graph  $G'$ . For each edge  $e = (v_k, v_l) \in E$ , we call the face  $(v_k, v_l, v_e)$  of  $G'$  a  $\alpha$ -face. We call each of the

remaining faces of  $G'$  a  $\beta$ -face. Figure 3.3 illustrates  $\alpha$ -faces and  $\beta$ -faces. We now have the following lemma.

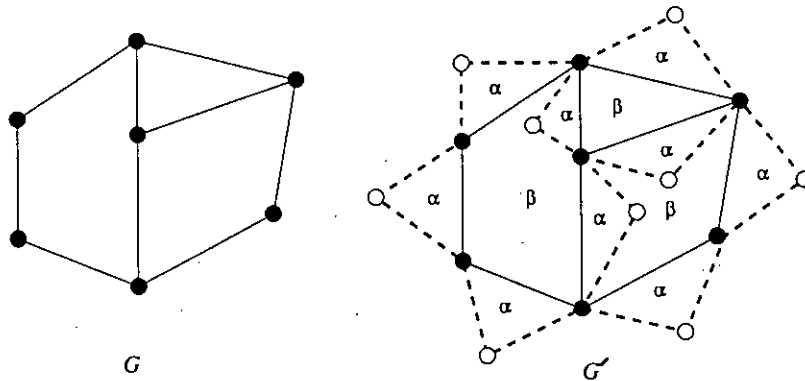


Figure 3.3: Illustration for the construction of  $G'$  from  $G$ .

**Lemma 3.2.4**  $G'$  has a face-spanning subgraph  $H$  of cost  $\leq B'$  if and only if  $G$  has a weighted tree cover  $T$  of cost  $\leq B$ , where  $B$  and  $B'$  are two positive real numbers.

**Proof. Necessity.** Assume that  $G'$  has a face-spanning subgraph of cost  $\leq B'$ , that is, there is an edge set  $S' \subseteq E'$  of graph  $G'$  such that the subgraph  $H$  induced by  $S'$  is connected, cover all faces of  $G'$  and the cost of  $H$  is  $\leq B'$ . We now prove that  $G$  has a weighted tree cover  $T$  of cost  $\leq B$ , for a positive real number  $B$ .

From the construction of  $G'$  it is obvious that the degree of each new vertex  $v$  is two in  $G'$ . Each new vertex has exactly two neighbors  $v_i, v_j$  among the original vertices and there is an original edge  $(v_i, v_j)$  between the two original vertices as illustrated in Figure 3.4(a). Modifying the subgraph  $H$  we construct a subgraph  $T$  of  $G'$  such that  $T$  contains only the original vertices and original edges as follows. Since  $H$  is a subgraph of  $G'$ , degree of each new vertex  $v$  in  $H$  is either one or two. For each new vertex  $v$  of  $H$  we perform one of the two operations described in Case 1 and Case 2 below to obtain  $T$  from  $H$ .

*Case 1:*  $v$  has degree two in  $H$

In this case  $v$  has two neighbors  $v_i, v_j$  among the original vertices such that  $(v_i, v_j)$  is



an original edge. If  $(v_i, v_j) \in E(H)$  then we delete  $v$  from  $H$  to obtain  $T$  as illustrated in Figure 3.4(b) and 3.4(e). Otherwise we replace the path  $(v_i, v, v_j)$  of  $H$  by the edge  $(v_i, v_j)$  to construct  $T$  as illustrated in Figure 3.4(c) and 3.4(e).

*Case 2:  $v$  has degree one in  $H$*

In this case  $v$  has exactly one neighbor  $v_i$  among the original vertices. We simply remove the new vertex  $v$  of  $H$  to construct  $T$ . Figure 3.4(d) and 3.4(f) illustrates this case.

If  $T$  contains cycles, we delete an edge from each cycle until the resulting subgraph has no cycle and we regard the resulting subgraph as  $T$ , and take the set of all edges in  $T$  as  $S$ .

We now prove that  $T$  is a tree in  $G$  of cost  $\leq B$ . Since  $H$  is connected, if we delete the new vertex  $v$  or we replace the path  $(v_i, v, v_j)$  by edge  $(v_i, v_j)$  in Case 1,  $T$  remains connected. Again, the cost of the path  $(v_i, v, v_j)$  is  $w(e)/2 + w(e)/2 = w(e)$  in total, which is equal to the cost of the edge  $(v_i, v_j)$ . Hence in Case 1, the cost of the modified subgraph is decreased (if we delete the new vertex  $v$ ) or unchanged (if the path  $(v_i, v, v_j)$  is replaced by edge  $(v_i, v_j)$ ). In Case 2, the new vertex has degree one and it is omitted, hence  $T$  remains connected after considering Case 2 for all such new vertices. In this case, edge  $(v, v_i)$  is removed, hence the cost of the modified subgraph decreases. Thus  $T$  is a connected subgraph of cost  $\leq B'$  in  $G'$ . Note that we have destroyed cycles to construct  $T$  and  $T$  is a tree of cost  $\leq B'$  in  $G'$ . If we take  $B = B'$ , then the cost of tree  $T$  is  $\leq B$ . Since the edges of  $T$  in  $G'$  are original edges and the vertices of  $T$  in  $G'$  are original vertices,  $G$  contains  $T$ . Hence  $T$  is a tree of cost  $\leq B$  in  $G$ .

Note that  $T$  and  $H$  are induced by  $S$  and  $S'$  respectively. To prove that  $T$  is a weighted tree cover in  $G$  of cost  $\leq B$ , it is now remained to show that the set of vertices  $V(S)$  of subgraph  $T$  is a vertex cover in  $G$ . Since  $H$  is a face-spanning subgraph of  $G'$ , the set of vertices  $V(S')$  of  $H$  covers all faces of  $G'$ . Hence  $V(S')$  contains at least one vertex (either black or white) from the boundary of each face of  $G'$ . Since  $H$  is connected,  $V(S')$  can contain a new vertex  $v$  only if  $V(S')$  contains at least one neighbor  $v_i$  of  $v$  among the

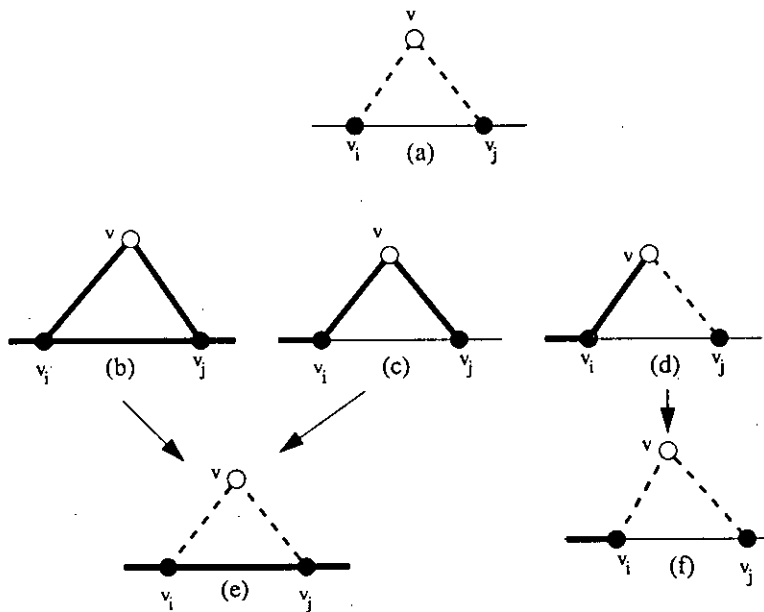


Figure 3.4: Illustration for the construction of  $T$  from  $H$ .

original vertices in  $G'$ . Since a new vertex  $v$  has degree 2, the two faces covered by a new vertex  $v$  are also covered by an original vertex  $v_i$  which is neighbor to the new vertex  $v$ . Thus  $V(S')$  contains at least one original vertex from the boundary of each face of  $G'$ . Since  $V(S)$  contains all the original vertices of  $V(S')$ ,  $V(S)$  also contains at least one original vertex from the boundary of each face of  $G'$ . Since we create an  $\alpha$ -face in  $G'$  for each edge of  $G$  while constructing  $G'$ , there is a face of  $G'$  for each edge in  $G$ . Since each face of  $G'$  is covered by  $V(S)$ , each edge of  $G$  is covered by  $V(S)$ . Hence  $V(S)$  contains at least one vertex from each edge of  $G$ . Thus  $V(S)$  is a vertex cover of  $G$ .

Since,  $T$  is a tree in  $G$  of cost  $\leq B$  and  $V(S)$  is a vertex cover of graph  $G$ ,  $T$  is a weighted tree cover of cost  $\leq B$  of  $G$ .

**Sufficiency.** Assume that  $G$  has a weighted tree cover of cost  $\leq B$ , that is, there is tree  $T$  in  $G$  of cost  $\leq B$  and the vertex set  $V(S)$  that  $T$  contains is a vertex cover of graph  $G$ . We now prove that  $G'$  has a face-spanning subgraph  $H$  of cost  $\leq B'$ , for a positive

real number  $B'$ . We take  $B = B'$ .

From the construction of  $G'$  it is clear that all the vertices in  $V(S)$  and the edges of  $T$  in  $G$  are also in  $G'$ . We take  $S'$  as the set of edges of  $G'$  which are in  $S$  and let  $H$  be the subgraph induced by  $S'$ . Then  $H$  contains all the edges of  $T$  and  $V(S')$  contains all the vertices in  $V(S)$ . We now show that the subgraph  $H$  of  $G'$  (i) is connected, (ii) cover all faces of  $G'$  and (iii) the cost of  $H$  is  $\leq B'$ .

(i) From the construction it is obvious that all the vertices and edges of  $G$  are also in  $G'$ . Since  $T$  is a tree in  $G$  and  $T = H$ ,  $H$  is a tree in  $G'$ . Hence the subgraph  $H$  induced by  $S'$  in  $G'$  is connected.

(ii) Since the subgraph  $T$  induced by  $S$  is a weighted tree cover of  $G$ , then for each edge  $e = (v_k, v_l) \in E$  of  $G$ ,  $V(S)$  contains either  $v_k$  or  $v_l$  or both. By the construction of  $G'$  from  $G$ ,  $G'$  has an  $\alpha$ -face for each edge  $e \in E$  of  $G$ . Thus  $V(S')$  contains  $v_k$  or  $v_l$  or both for each  $\alpha$ -face of graph  $G'$ . Since each edge of  $G$  is covered by  $V(S)$ , each  $\alpha$ -face of graph  $G'$  is covered by  $V(S')$ . We now need to show that the  $\beta$ -faces of  $G'$  are also covered by  $V(S')$ . Since each  $\beta$ -face of  $G'$  contains the original vertices of at least three  $\alpha$ -faces and  $V(S')$  contains at least one original vertex from each  $\alpha$ -face,  $V(S')$  contains at least two original vertices. Hence each  $\beta$ -face of  $G'$  is covered by  $V(S')$ . Thus  $V(S')$  covers all the faces of  $G'$ , that means, the subgraph  $H$  induced by  $S'$  in  $G'$  cover all faces of  $G'$ .

(iii) The cost of  $T$  is  $\leq B$ . Since  $T = H$  and  $B = B'$ , the cost of  $H$  is  $\leq B'$  in  $G'$ .  $\square$

**Proof. of Lemma 3.2.3:** Since the construction of  $G'$  from  $G$  takes polynomial time, Lemma 3.2.4 implies that the face-spanning subgraph problem is  $NP$ -hard.  $\square$

After the long discussion on  $NP$  and  $NP$ -hard of the face-spanning subgraph problem, we are now in a position to prove the Theorem 3.2.1. Below we prove the Theorem 3.2.1.

**Proof. of Theorem 3.2.1:** By Lemma 3.2.2, the face-spanning subgraph problem is in  $NP$  and by Lemma 3.2.3, the face-spanning subgraph problem is  $NP$ -hard. Hence the

face-spanning subgraph problem is *NP*-complete.  $\square$

### 3.3 Summary

In this chapter we have proved that the face-spanning subgraph problem is *NP*-complete. We use two steps to prove the *NP*-completeness of the face-spanning subgraph problem. First, we have shown that the face-spanning subgraph problem is in *NP*. For this, we have shown that any candidate solution of the face-spanning subgraph problem can be verified in polynomial time. As a second step, we have proved that the face-spanning subgraph problem is *NP*-hard. For this, we have transformed the weighted tree cover problem to the face-spanning subgraph problem in polynomial time. These above two steps immediately prove that the face-spanning subgraph problem is *NP*-complete. This is the first time in literature that the face-spanning subgraph problem is introduced and the *NP*-completeness of the face-spanning subgraph problem is proved.

## Chapter 4

# Minimum-Vertex Face-Spanning Subgraph Problem

In this chapter we consider a variation of the face-spanning subgraph problem, which we call the “minimum-vertex face-spanning subgraph problem”. The minimum-vertex face-spanning subgraph problem often arises in applications like establishing base transceiver stations in wireless networks, establishing power distribution centers in a city etc where the setup cost for each establishment is huge. In these cases the objective is to minimize the number of vertices instead of edge cost. In this chapter we show that the minimum-vertex face-spanning subgraph problem belongs to the *NP*-complete class. The organization of this chapter is as follows. Section 4.1 formally defines the minimum-vertex face-spanning subgraph problem. In Section 4.2 we prove the *NP*-completeness of the minimum-vertex face-spanning subgraph problem. Section 4.3 summarizes this chapter. The approximation algorithm to find a minimum-vertex face-spanning subgraph of a plane graph is discussed in Chapter 5.

## 4.1 Problem Definition

In this section we define the minimum-vertex face-spanning subgraph problem. Let  $G = (V, E)$  be a connected plane graph, where  $V$  and  $E$  are the set of vertices and edges, respectively, and let  $F$  be the set of faces of graph  $G$ . A minimum-vertex face-spanning subgraph  $H$  of  $G$  is a face-spanning subgraph of  $G$  where  $|V(H)|$  is minimum. Figure 4.1(a) shows a face-spanning subgraph drawn by thick lines with 6 vertices whereas Figure 4.1(b) shows a face-spanning subgraph drawn by thick lines with 7 vertices.

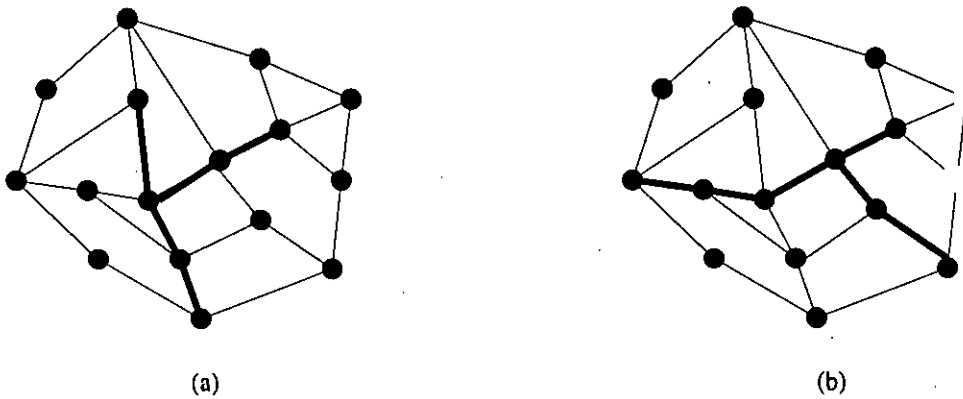


Figure 4.1: A simple graph with (a) a minimum-vertex face-spanning subgraph of 6 vertices and (b) a minimum-vertex face-spanning subgraph of 7 vertices.

A minimum-vertex face-spanning subgraph problem asks to find a minimum-vertex face-spanning subgraph of a plane graph. We say the minimum-vertex face-spanning subgraph problem as “MVFSSP” in short in many places rest of this chapter. The formal definition of the minimum-vertex face-spanning subgraph problem is as follows:

**Definition 4.1.1** (Minimum-Vertex Face-Spanning Subgraph Problem) *Let  $G = (V, E)$  be a connected plane graph, where  $V$  and  $E$  are the set of vertices and edges, respectively, and let  $F$  be the set of faces of graph  $G$ . Then is there any set  $S \subseteq E$  such that the subgraph  $H$  induced by  $S$  is connected, cover all faces of  $G$  and  $|V(H)| \leq K$ , for a given positive integer  $K \leq |V|$ ?*

## 4.2 *NP*-completeness of MVFSSP

It is unlikely to design an efficient algorithm for finding a minimum-vertex face-spanning subgraph of a plane graph. For this, we show that the minimum-vertex face-spanning subgraph problem belongs to the infamous class of *NP*-complete problems. We have the following theorem.

**Theorem 4.2.1** *The minimum-vertex face-spanning subgraph problem is NP-complete.*

To prove the Theorem 4.2.1, we have to show that (i) the minimum-vertex face-spanning subgraph problem is in *NP* and (ii) the minimum-vertex face-spanning subgraph problem is *NP*-hard. In Subsection 4.2.1, we prove that the minimum-vertex face-spanning subgraph problem is in *NP* and in Subsection 4.2.2, we prove that the minimum-vertex face-spanning subgraph problem is *NP*-hard. These two steps immediately prove the Theorem 4.2.1. We use the well known *NP*-complete problem connected vertex cover problem to prove the hardness of the minimum-vertex face-spanning subgraph problem.

### 4.2.1 MVFSSP is in *NP*

As discussed before, *NP* is the set of decision problems solvable in polynomial time on a non-deterministic turing machine. We can identify that whether a problem is in *NP* or not, by verifying a candidate solution of the problem in polynomial-time by deterministic turing machine.

In this subsection we prove the first step of the proof of *NP*-completeness of the minimum-vertex face-spanning subgraph problem, that is, the minimum-vertex face-spanning subgraph problem is in *NP*. Let  $G = (V, E)$  be a connected plane graph, where  $V$  and  $E$  are the set of vertices and edges, respectively. Let  $F$  be the set of faces of graph  $G$ . Thus we have the following lemma.

**Lemma 4.2.2** *The minimum-vertex face-spanning subgraph problem is in NP.*

**Proof.** To prove that the minimum-vertex face-spanning subgraph problem is in  $NP$ , it is sufficient to prove that for a given set  $S \subseteq E$ , we can verify in polynomial-time that the subgraph  $H$  induced by  $S$  (i) is connected, (ii) cover all faces of  $G$  and (iii)  $|V(H)| \leq K$ , for a positive integer  $K$ .

(i) Connectivity of the subgraph  $H$  induced by  $S$  can be checked using DFS in linear time.

(ii) We can verify whether  $S$  covers all faces of  $G$  or not in linear time using a method similar to one in the proof of Lemma 3.2.2.

(iii) It can be verified in  $O(n)$  time that  $|V(H)| \leq K$ .

Since it is possible to verify (i), (ii) and (iii) in polynomial time, the minimum-vertex face-spanning subgraph problem is in  $NP$ .  $\square$

#### 4.2.2 MVFSSP is $NP$ -hard

A reduction is a transformation of one problem into another problem. Intuitively, if problem  $A$  is reducible to problem  $B$ , then a solution to  $B$  gives a solution to  $A$ . Thus, solving  $A$  cannot be harder than solving  $B$ . We say a problem  $X$  is  $NP$ -hard if all the problems in  $NP$  are polynomially reducible to  $X$ .

We now prove the second part of the proof of  $NP$ -completeness of the minimum-vertex face-spanning subgraph problem, that is, the minimum-vertex face-spanning subgraph problem is  $NP$ -hard. Thus we have the following lemma.

**Lemma 4.2.3** *The minimum-vertex face-spanning subgraph problem is  $NP$ -hard.*

To prove lemma 4.2.3 we will prove that, the  $NP$ -complete problem connected vertex cover problem can be polynomially transformed into the minimum-vertex face-spanning subgraph problem in such a way that any polynomial-time algorithm for solving the minimum-vertex face-spanning subgraph problem could be used to solve the connected vertex cover problem in polynomial time.



Let  $G = (V, E)$  be a connected plane graph, where  $V$  and  $E$  are the set of vertices and edges, respectively. Let  $F$  be the set of faces of graph  $G$ . We obtain a graph  $G'$  from  $G$  as follows. For each edge  $e = (v_k, v_l) \in E$  we add a vertex  $v_e$  and two edges  $(v_k, v_e)$  and  $(v_l, v_e)$  to  $G$ . More formally,  $G' = (V', E')$  where  $V' = V \cup V_e$ ,  $V_e = \{v_e | e \in E\}$  and  $E' = E \cup E_e$  where  $E_e = \{(v_e, v_k), (v_e, v_l) | \{e = (v_k, v_l)\} \in E\}$ . In  $G'$  we call a vertex in  $V_e$  a *new vertex*, a vertex in  $V$  an *original vertex*, an edge in  $E_e$  a *new edge* and an edge in  $E$  an *original edge*. Note that original vertices and original edges of  $G'$  are also the vertices and edges of  $G$ . We assign the zero cost to each of the edges  $(v_e, v_k)$  and  $(v_e, v_l)$  for all  $e \in E$ . Figure 4.2 illustrates the construction of  $G'$  where the vertices drawn by white small circles are new vertices, the edges drawn by dashed lines are new edges, the vertices drawn by black circles are original vertices and the edges drawn by solid lines are the original edges of  $G'$ . If  $G$  has  $n$  vertices and  $m$  edges, then  $G'$  has  $n + m$  vertices and  $3m$  edges. Clearly  $G'$  can be constructed in  $O(n)$  time. One can easily observe that the graph  $G'$  is planar as illustrated in Figure 4.2, where a plane embedding of  $G'$  is shown. Throughout the paper we consider  $G'$  as a plane embedding of the graph  $G'$ . For each edge  $e = (v_k, v_l) \in E$ , we call the face  $(v_k, v_l, v_e)$  of  $G'$  a  $\alpha$ -face. We call each of the remaining faces of  $G'$  a  $\beta$ -face. Figure 4.2 illustrates  $\alpha$ -faces and  $\beta$ -faces. We now have the following lemma.

**Lemma 4.2.4**  *$G'$  has a minimum-vertex face-spanning subgraph  $H'$  with  $|V(H')| \leq K'$  if and only if  $G$  has a connected vertex cover  $H$  with  $|V(H)| \leq K$ , where  $K$  and  $K'$  are two positive integers.*

**Proof.** **Necessity.** Assume that  $G'$  has a minimum-vertex face-spanning subgraph  $H'$  with  $|V(H')| \leq K'$ , that is, there is an edge set  $S' \subseteq E'$  of graph  $G'$  such that the subgraph  $H'$  induced by  $S'$  is connected, cover all faces of  $G'$  and  $|V(H')| \leq K'$ . We now prove that  $G$  has a connected vertex cover  $H$  with  $|V(H)| \leq K$ , for a positive integer  $K$ .

From the construction of  $G'$  it is obvious that the degree of each new vertex  $v$  is two in  $G'$ . Each new vertex has exactly two neighbors  $v_i, v_j$  among the original vertices and there

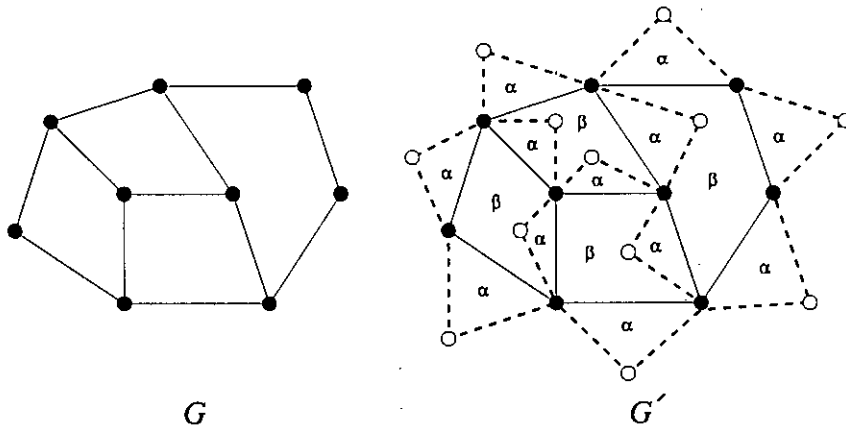


Figure 4.2: Illustration for the construction of  $G'$  from  $G$ .

is an original edge  $(v_i, v_j)$  between the two original vertices as shown in Figure 4.3(a). Modifying the subgraph  $H'$  we construct a subgraph  $H$  of  $G'$  such that  $H$  contains only the original vertices and original edges as follows. Since  $H'$  is a subgraph of  $G'$ , degree of each new vertex  $v$  in  $H'$  is either one or two. For each new vertex  $v$  of  $H'$  we perform one of the two operations described in Case 1 and Case 2 below to obtain  $H$  from  $H'$ .

*Case 1:  $v$  has degree two in  $H'$*

In this case  $v$  has two neighbors  $v_i, v_j$  among the original vertices such that  $(v_i, v_j)$  is an original edge. If  $(v_i, v_j) \in E(H')$  then we delete  $v$  from  $H'$  to obtain  $H$  as illustrated in Figure 4.3(b) and 4.3(e). Otherwise we replace the path  $(v_i, v, v_j)$  of  $H'$  by the edge  $(v_i, v_j)$  to construct  $H$  as illustrated in Figure 4.3(c) and 4.3(e).

*Case 2:  $v$  has degree one in  $T'$*

In this case  $v$  has one neighbor  $v_i$  among the original vertices. We simply remove the new vertex  $v$  of  $H'$  to construct  $H$ . Figure 4.3(d) and 4.3(f) illustrates this case.

We regard the resulting subgraph as  $H$  and take the set of all edges in  $H$  as  $S$ .

We now prove that  $H$  is a connected subgraph in  $G$  with  $|V(H)| \leq K$ . Since  $H'$  is connected, if we delete the new vertex  $v$  or we replace the path  $(v_i, v, v_j)$  by edge  $(v_i, v_j)$  in Case 1,  $H$  remains connected. Again, number of vertices in path  $(v_i, v, v_j)$  is 3 and an

edge  $(v_i, v_j)$  contains 2 vertices. Hence in Case 1, the number of vertices in the modified subgraph  $H$  is decreased by 1 whether we delete the new vertex  $v$  or replace the path  $(v_i, v, v_j)$  by edge  $(v_i, v_j)$ . In Case 2, the new vertex has degree one and it is omitted, hence  $H$  remains connected after considering Case 2 for all such new vertices. In this case, the edge  $(v, v_i)$  is removed, hence the size of the modified subgraph decreases. Thus  $H$  is a connected subgraph in  $G'$  with  $|V(H)| \leq K'$ . If we take  $K = K'$ ,  $|V(H)| \leq K'$  implies  $|V(H)| \leq K$ . Since the edges of  $H$  in  $G'$  are original edges and the vertices of  $H$  in  $G'$  are original vertices,  $G$  contains  $H$ . Hence  $H$  is a connected subgraph in  $G$  with  $|V(H)| \leq K$ .

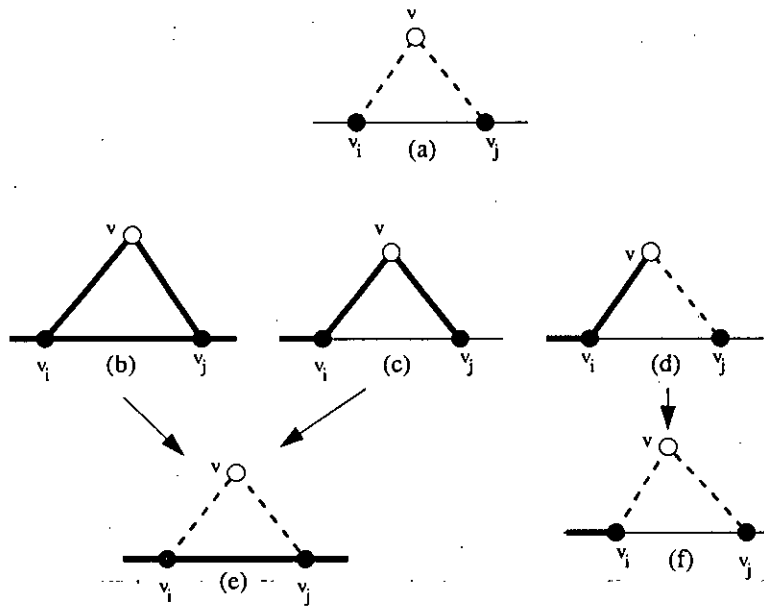


Figure 4.3: Illustration for the construction of  $H$  from  $H'$ .

Note that  $H$  and  $H'$  are induced by  $S$  and  $S'$  respectively. To prove that  $H$  is a connected vertex cover in  $G$  with  $|V(H)| \leq K$ , it is now remained to show that the set of vertices  $V(S)$  of subgraph  $H$  is a vertex cover in  $G$ . Since  $H'$  is a minimum-vertex face spanning subgraph of  $G'$ , the set of vertices  $V(S')$  of  $H'$  covers all faces of  $G'$ . Hence

$V(S')$  contains at least one vertex (either black or white) from the boundary of each face of  $G'$ . Since  $H'$  is connected,  $V(S')$  can contain a new vertex  $v$  only if  $V(S')$  contains at least one neighbor vertex  $v_i$  of  $v$  among the original vertices in  $G'$ . Since a new vertex  $v$  has degree 2, the two faces covered by a new vertex  $v$  are also covered by an original vertex  $v_i$  which is neighbor to the new vertex  $v$ . Thus  $V(S')$  contains at least one original vertex from the boundary of each face of  $G'$ . Since  $V(S)$  contains all the original vertices of  $V(S')$ ,  $V(S)$  also contains at least one original vertex from the boundary of each face of  $G'$ . Since we create an  $\alpha$ -face in  $G'$  for each edge of  $G$  while constructing  $G'$ , there is a face of  $G'$  for each edge in  $G$ . Since each face of  $G'$  is covered by  $V(S)$ , each edge of  $G$  is covered by  $V(S)$ . Hence  $V(S)$  contains at least one vertex from each edge of  $G$ . Thus  $V(S)$  is a vertex cover of  $G$ .

Since,  $H$  is a connected subgraph in  $G$  with  $|V(H)| \leq K$  and  $V(S)$  is a vertex cover of graph  $G$ ,  $H$  is a connected vertex cover of  $G$  with  $|V(H)| \leq K$ .

**Sufficiency.** Assume that  $G$  has a connected vertex cover of size  $\leq K$ , that is, there is a connected subgraph  $H$  in  $G$  with  $|V(H)| \leq K$  and the vertex set  $V(S)$  that  $H$  contains is a vertex cover of graph  $G$ . We now prove that  $G'$  has a minimum-vertex face-spanning subgraph  $H'$  with  $|V(H')| \leq K'$ , for a positive integer  $K'$ . We assume  $K = K'$ .

From the construction of  $G'$  it is clear that all the vertices in  $V(S)$  and the edges of  $H$  in  $G$  are also in  $G'$ . We take  $S'$  as the set of edges of  $G'$  which are contained in  $S$  and let  $H'$  be the subgraph induced by  $S'$ . Then  $H'$  contains all the edges of  $H$  and  $V(S')$  contains all the vertices in  $V(S)$ . We now show that the subgraph  $H'$  of  $G'$  (i) is connected, (ii) cover all faces of  $G'$  and (iii)  $|V(H')| \leq K'$ .

(i) From the construction it is obvious that all the vertices and edges of  $G$  are also in  $G'$ . Since  $H$  is a connected subgraph in  $G$  and  $H = H'$ ,  $H'$  is a connected subgraph in  $G'$ . Hence the subgraph  $H'$  induced by  $S'$  in  $G'$  is connected.

(ii) Since the subgraph  $H$  induced by  $S$  is a connected vertex cover of  $G$ , then for each edge  $e = (v_k, v_l) \in E$  of  $G$ ,  $V(S)$  contains either  $v_k$  or  $v_l$  or both. By the construction of

$G'$  from  $G$ ,  $G'$  has an  $\alpha$ -face for each edge  $e \in E$  of  $G$ . Thus,  $V(S')$  contains  $v_k$  or  $v_l$  or both for each  $\alpha$ -face of graph  $G'$ . Since each edge of  $G$  is covered by  $V(S)$ , each  $\alpha$ -face of graph  $G'$  is covered by  $V(S')$ . We now need to show whether the  $\beta$ -faces of  $G'$  are also covered by  $V(S')$ . Since each  $\beta$ -face of  $G'$  contains the original vertices of at least three  $\alpha$ -faces and  $V(S')$  contains at least one original vertex from each  $\alpha$ -face,  $V(S')$  contains at least two original vertices. Hence each  $\beta$ -face of  $G'$  is covered by  $V(S')$ . Thus  $V(S')$  covers all the faces of  $G'$ , that means, the subgraph  $H'$  induced by  $S'$  in  $G'$  cover all faces of  $G'$ .

(iii) The size of  $H$  is  $\leq K$ , that is,  $|V(H)| \leq K$ . Since  $H = H'$  and  $K = K'$ ,  $|V(H')| \leq K'$  in  $G'$ . □

**Proof.** of Lemma 4.2.3: Since the construction of  $G'$  from  $G$  takes polynomial time, Lemma 4.2.4 implies that the minimum-vertex face-spanning subgraph problem is *NP*-hard. □

After the long discussion on *NP* and *NP*-hard of the minimum-vertex face-spanning subgraph problem, we are now in a position to prove the Theorem 4.2.1. Below we prove the Theorem 4.2.1.

**Proof.** of Theorem 4.2.1: By Lemma 4.2.2, the minimum-vertex face-spanning subgraph problem is in *NP* and by Lemma 4.2.3, the minimum-vertex face-spanning subgraph problem is *NP*-hard. Hence the minimum-vertex face-spanning subgraph problem is *NP*-complete. □

### 4.3 Summary

In this chapter we have proved that the minimum-vertex face-spanning subgraph problem is *NP*-complete. We use two steps to prove the *NP*-completeness of the minimum-vertex face-spanning subgraph problem. First, we have shown that the minimum-vertex face-

spanning subgraph problem is in  $NP$ . For this, we have shown that any candidate solution of the minimum-vertex face-spanning subgraph problem can be verified in polynomial time. As a second step, we have proved that the minimum-vertex face-spanning subgraph problem is  $NP$ -hard. For this, we have transformed the connected vertex cover problem to the minimum-vertex face-spanning subgraph problem in polynomial-time. These above two steps immediately prove that the minimum-vertex face-spanning subgraph problem is  $NP$ -complete. This is the first time in literature that the minimum-vertex face-spanning subgraph problem is introduced and the  $NP$ -completeness of the minimum-vertex face-spanning subgraph problem is proved.

## Chapter 5

# Approximation Algorithms

It is unlikely to have efficient algorithms for finding optimal solutions for the *NP*-complete problems. But there exists numerous practical applications which unfortunately fall in the infamous *NP*-complete class. Hence design of approximation algorithms is an urgent need. Since the face-spanning subgraph problem and the minimum-vertex face-spanning subgraph problem are *NP*-complete, we design approximation algorithms for finding minimum face-spanning subgraph and minimum-vertex face-spanning subgraph of a plane graph in this chapter.

In practical applications of the face-spanning subgraph problem like the gas pipelines planning problem discussed in Section 1, an input is often a plane graph  $G$  such that each vertex of  $G$  has degree three or more. We thus consider those plane graphs where the minimum degree three is at least three in this chapter for designing approximation algorithms.

In this chapter we introduce a new terminology called “minimal face-spanning subgraph”. This minimal face-spanning subgraph is used to find approximate solution of the the face-spanning subgraph problem and the minimum-vertex face-spanning subgraph problem. This chapter is organized as follows. Section 5.1 formally defines the minimal face-spanning subgraph and presents a lower tight bound on the number of vertices of a

face-spanning subgraph of a plane graph. Section 5.2 gives a linear-time algorithm to find a minimal face-spanning subgraph of a plane graph and the upper bound of the algorithm is calculated in this section and it is shown that the upper bound is tight. Section 5.3 illustrates approximation algorithms to find face-spanning subgraph and minimum-vertex face-spanning subgraph of a plane graph. The approximation ratio and complexity of the presented approximation algorithms have also been calculated in Section 5.3. Finally Section 5.4 discuss the findings of this chapter.

## 5.1 Minimal Face-Spanning Subgraph

In this section we define a minimal face-spanning subgraph of a plane graph. Let  $H$  be a face-spanning subgraph of  $G$  induced by edge set  $S \subseteq E$ . We call  $H$  a *minimal face-spanning subgraph* of  $G$  if there is no edge set  $S' \subseteq S$  such that the subgraph induced by  $S'$  is a face-spanning subgraph of  $G$ .

Clearly a minimal face-spanning subgraph is a tree. Figure 5.1 illustrates an example of minimal face-spanning subgraph. The thick lines in Figure 5.1(a) is a minimal face-spanning subgraph. The thick lines in Figure 5.1(b) is not a minimal face-spanning subgraph since the subset of this thick lines can induce a face-spanning subgraph.

A plane graph may have many minimal face-spanning subgraphs. In Figure 5.2(a), a minimal face-spanning subgraph of cost 4 is drawn by thick lines and in Figure 5.2(b) another minimal face-spanning subgraph of cost 2 is drawn by thick lines for the same graph. Note that a minimum face-spanning subgraph of  $G$  defined in Chapter 2 is one of the minimal face-spanning subgraphs of  $G$  whose total edge weight is minimum among all the minimal face-spanning subgraphs. Thus we can find a minimum face-spanning subgraph of a plane graph  $G$  by finding all minimal face-spanning subgraph and choosing one of the minimal face-spanning subgraphs of  $G$  whose total edge weight is minimum among all the minimal face-spanning subgraphs of  $G$ .



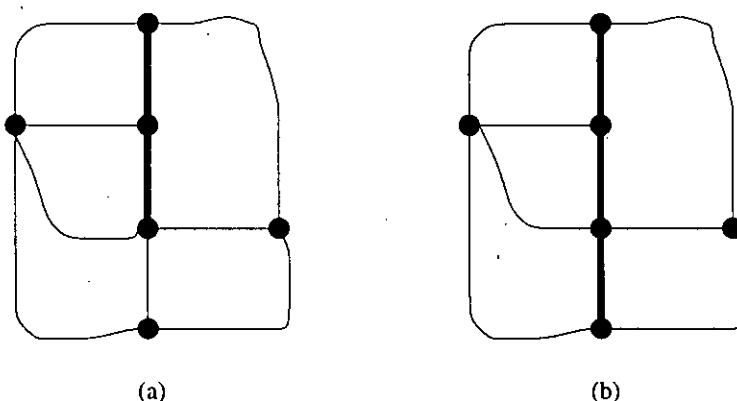


Figure 5.1: Illustration of (a) a minimal face-spanning subgraph, and (b) a non-minimal face-spanning subgraph.

A minimum-vertex face-spanning subgraph  $H$  of  $G$  defined in Chapter 4 may not be a minimal face-spanning subgraph of  $G$ , since the definition of a minimum-vertex face-spanning subgraph allows cycles in  $H$ . However, there exists a minimal face-spanning subgraph  $H'$  of  $G$  with the vertex set  $V(H)$ , and  $H'$  can be obtained by removing an edge from each cycle in  $H$  if  $H$  has any cycle. Figure 5.3 illustrates an example how to find a minimal face-spanning subgraph from a minimum-vertex face-spanning subgraph. Figure 5.3(a) shows a simple graph with a minimum-vertex face-spanning subgraph  $H$  drawn by thick lines and Figure 5.3(b) shows a minimal face-spanning subgraph  $H'$  drawn by thick lines obtained by removing an edge  $e$  from each cycle of  $H$  in Figure 5.3(a).

We now establish a *lower bound* on the number of vertices of a minimal face-spanning subgraph of a plane graph. We have the following lemma.

**Lemma 5.1.1** *Let  $G = (V, E)$  be a connected plane graph. Assume that each vertex of  $G$  has degree three or more. Let  $H$  be a minimal face-spanning subgraph induced by  $S \subseteq E$  of  $G$ . Then  $|V(S)| \geq (f - 2)/(\Delta - 2)$ , where  $f$  is the number of faces of  $G$ .*

**Proof.** Let  $G$  be a connected plane graph with maximum degree  $\Delta$  and  $f$  faces. Let  $H$  be a minimal face-spanning subgraph of  $G$  induced by  $S \subseteq E$  and  $S$  contains  $k$  edges.

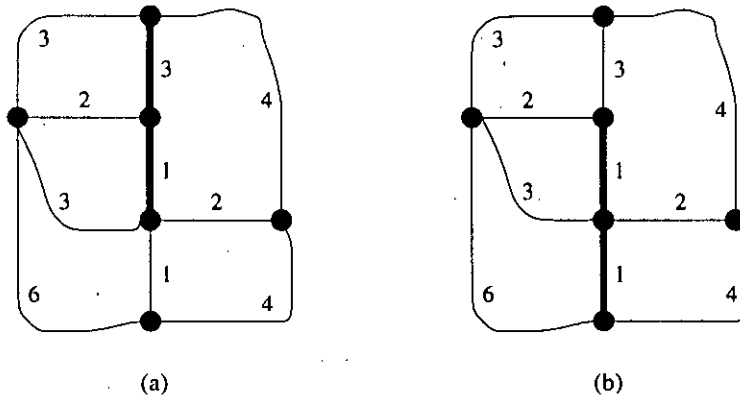


Figure 5.2: (a) A minimal face-spanning subgraph of cost 4, and (b) a minimal face-spanning subgraph of cost 2.

We prove the above claim using induction on  $k$ .

If  $k = 1$ , then  $S$  contains exactly one edge and hence  $|V(S)| = 2$ . In this case  $S$  covers at most  $2\Delta - 2$  faces. This implies  $(f - 2)/(\Delta - 2) \leq 2 = |V(S)|$ . Therefore the claim holds.

Assume that  $k \geq 2$  and the claim holds for all connected plane graphs each of which has a minimal face-spanning subgraph of fewer than  $k$  edges, and suppose that  $G$  has a face-spanning subgraph of  $K$  edges. We remove an edge  $e$  from  $S$  such that the graph  $H'$  induced by  $S' = S - \{e\}$  is connected. Since  $H'$  is connected,  $|V(S')| = |V(S)| - 1$ . Let  $G'$  be the subgraph of  $G$  such that  $G'$  contains all faces of  $G$  covered by  $S'$  and  $H'$  is a minimal face-spanning subgraph of  $G'$ . Let  $f'$  and  $\Delta'$  be the number of faces and the maximum degree of  $G'$ , respectively. Then  $f' \geq f - (\Delta - 2)$ , since  $S$  can cover at most  $(\Delta - 2)$  faces more than the faces covered by  $S'$ . Furthermore,  $\Delta \geq \Delta'$ . Since  $S'$  has less than  $k$  edges, by induction hypothesis  $|V(S')| \geq (f' - 2)/(\Delta' - 2)$ . Since  $f' \geq f - (\Delta - 2)$ ,  $\Delta \geq \Delta'$ , and  $|V(S')| = |V(S)| - 1$ , the claim immediately follows from induction hypothesis.  $\square$

We have a graph of 9 faces with  $\Delta = 3$  as illustrated in Figure 5.4, for which the

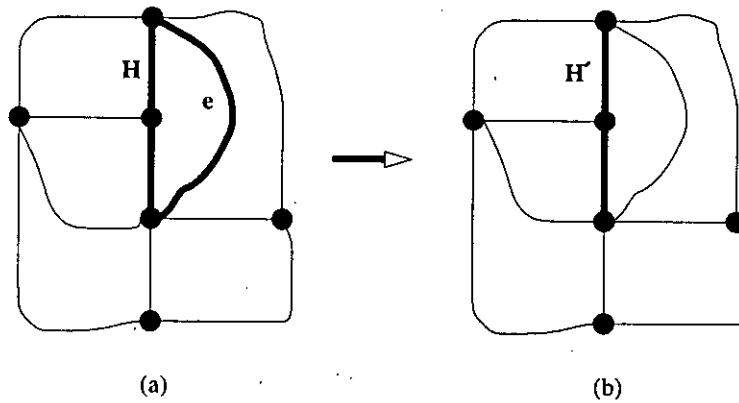


Figure 5.3: (a) A minimum-vertex face-spanning subgraph  $H$ , and (b) a minimal face-spanning subgraph  $H'$  obtained from  $H$ .

minimum number of vertices required for a minimal face-spanning subgraph is 7. Thus the example in Figure 5.4 attains the lower bound, and hence the bound is tight.

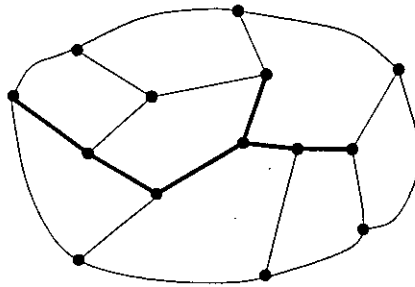


Figure 5.4: A graph of 9 faces with  $\Delta = 3$  for which the minimum number of vertices required for a face-spanning subgraph drawn by thick lines is 7.

## 5.2 *Find-Minimal-Subgraph* Algorithm

We now give an algorithm for finding a minimal face-spanning subgraph based on a spanning tree [HSR98]. Let  $G = (V, E)$  be a connected plane graph, where  $V$  and  $E$  are the set of vertices and edges, respectively, and let  $F$  be the set of faces of  $G$ . Let  $v_0$  be an

outer vertex of  $G$ . Let  $G'$  be the graph obtained from  $G$  by deleting all outer vertices of  $G$  except  $v_0$ . Let  $T$  be a spanning tree of  $G'$ . One can observe that  $T$  is a face-spanning subgraph of  $G$ . We traverse the tree  $T$  and delete each leaf vertex  $v$  of  $T$  if each of the faces of  $G$  which contains  $v$  is covered by any other vertex in  $T$ . Deletion of  $v$  from  $T$  may generate a new leaf vertex of  $T$ . We repeat the operation above for all the leaf vertices of  $T$  including the newly generated leaf vertices. The resulting tree  $H$  is our desired minimal face-spanning subgraph. Using a data structure similar to that described in Lemma 3.2.2 we can obtain a minimal face-spanning subgraph mentioned above in linear time. We call the algorithm described above *Find-Minimal-Subgraph*. Figure 5.5 illustrates the transformation of  $G$  to  $G'$  along with a minimal face-spanning subgraph  $H$  of graph  $G$  drawn by thick lines.

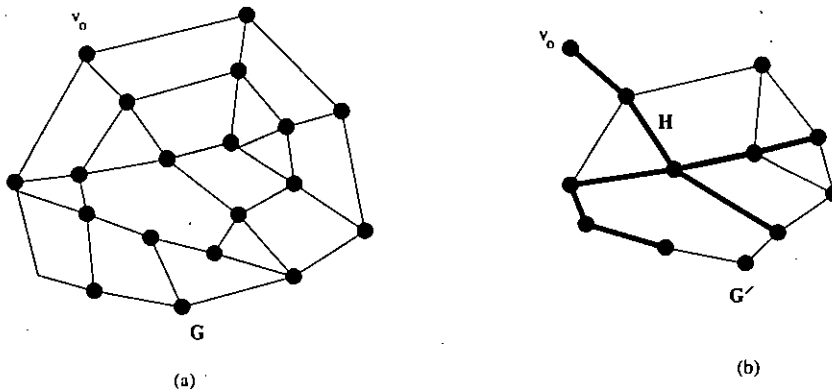


Figure 5.5: Illustration of the transformation of  $G$  to  $G'$ . (a) a simple graph  $G$  and (b)  $G'$  obtained from  $G$  with a minimal face-spanning subgraph  $H$  of  $G$  drawn by thick lines.

Clearly the following lemma holds on the *upper bound* of the number of vertices of a minimal face-spanning subgraph produced by Algorithm *Find-Minimal-Subgraph*.

**Lemma 5.2.1** *Let  $G$  be a plane graph of  $n$  vertices, and let  $n_0$  be the number of outer vertices of  $G$ . Assume that each vertex of  $G$  has degree three or more. Then Algorithm *Find-Minimal-Subgraph* produce a minimal face-spanning subgraph with at most  $n - n_0 + 1$*

vertices in linear time.

The upper bound in Lemma 5.2.1 is also tight, since we have an infinite number of examples attaining the bound; one example is shown in Figure 5.6.

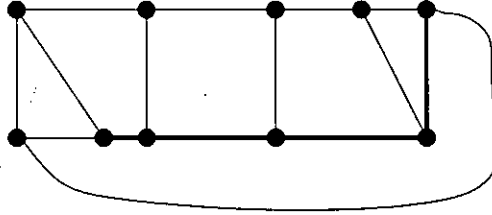


Figure 5.6: A graph of 7 faces with  $n = 10$  and  $n_0 = 6$  for which the minimum number of vertices required for a face-spanning subgraph is 5.

### 5.3 Approximation Algorithms

We can use a minimal face-spanning subgraph of a plane graph to find an approximate solution for the face-spanning subgraph problem and the minimum-vertex face-spanning subgraph problem. In this section we present approximation algorithms for the face-spanning subgraph problem and the minimum-vertex face-spanning subgraph problem along with the approximation ratios and complexity analysis.

#### 5.3.1 Minimum-Vertex Face-Spanning Subgraph Problem

We can take a minimal face-spanning subgraph of a connected plane graph  $G$  produced by Algorithm *Find-Minimal-Subgraph* as an approximate solution of the minimum-vertex face-spanning subgraph problem. Then we have the following theorem.

**Theorem 5.3.1** *Let  $G = (V, E)$  be a connected plane graph. Then the approximation ratio of Algorithm *Find-Minimal-Subgraph* for finding minimum-vertex face-spanning subgraph is  $2(\Delta - 2)$ .*

**Proof.** Algorithm *Find-Minimal-Subgraph* constructs a minimum-vertex face-spanning subgraph of  $G$  with at most  $n - n_0 + 1$  vertices. By Lemma 5.1.1 a face-spanning subgraph of  $G$  has at least  $(f - 2)/(\Delta - 2)$  vertices. Hence approximation ratio is  $(n - n_0 + 1)/\{(f - 2)/(\Delta - 2)\}$ . From Euler's Formula for planar graphs, we have  $f - 2 = m - n$ . Since degree of any vertex in  $G$  is  $\geq 3$ ,  $2m \geq 3n$ . This implies  $(m - n) \geq n/2$  and hence  $(f - 2) \geq n/2$ . Therefore the approximation ratio is  $(n - n_0 + 1)/\{(f - 2)/(\Delta - 2)\} \leq (n - n_0 + 1)/\{(n/2)/(\Delta - 2)\} = 2(n - n_0 + 1)(\Delta - 2)/n \leq 2(\Delta - 2)$ .  $\square$

Since we have used Algorithm *Find-Minimal-Subgraph* to find approximate solution of the minimum-vertex face-spanning subgraph problem and the time complexity of Algorithm *Find-Minimal-Subgraph* is linear, the time complexity of the approximation algorithm to find approximate solution of the minimum-vertex face-spanning subgraph problem is linear.

### 5.3.2 Face-Spanning Subgraph Problem

A minimal face-spanning subgraph produced by Algorithm *Find-Minimal-Subgraph* can also be taken as an approximate solution of the minimum face-spanning subgraph problem. One can easily observe that approximation ratio of Algorithm *Find-Minimal-Subgraph* for finding minimum face-spanning subgraph is  $\{(n - n_0)e_{max}\}/\{(f - 2)/(\Delta - 2) - 1\}e_{min} = \{(n - n_0)(\Delta - 2)e_{max}\}/\{(f - \Delta)e_{min}\}$ , where  $e_{max}$  and  $e_{min}$  denote the maximum and minimum weight of the edges of  $G$ . The time complexity of this approximation algorithm is also linear.

## 5.4 Summary

In this chapter we have introduced minimal face-spanning subgraph of a plane graph. This minimal face-spanning subgraph has been used to find approximate solution of the the face-spanning subgraph problem and the minimum-vertex face-spanning subgraph

problem. We have established a tight lower bound on the number of vertices of a minimal face-spanning subgraph of a plane graph. We have also designed a linear-time algorithm to find a minimal face-spanning subgraph of a plane graph. We have also calculated the upper bound of the algorithm which is also tight.

We have designed approximation algorithms for finding minimum face-spanning subgraph and minimum-vertex face-spanning subgraph of a plane graph. We see that the time complexities for both the approximation algorithms are linear. The approximation ratio of the face-spanning subgraph problem and the minimum-vertex face-spanning subgraph problem are  $\{(n - n_0)(\Delta - 2)e_{max}\}/\{(f - \Delta)e_{min}\}$  and  $2(\Delta - 2)$  respectively.

# Chapter 6

## Conclusion

In this thesis we deal with a newly introduced graph problem called the face-spanning subgraph problem. We have proved the hardness of the face-spanning subgraph problem. We also proved a variation of the face-spanning subgraph problem called the minimum-vertex face-spanning subgraph problem is *NP*-complete. We have introduced minimal face-spanning subgraph of a plane graph in this thesis. We have established a tight lower bound on the number of vertices of a minimal face-spanning subgraph of a plane graph. We have also designed a linear time algorithm to find a minimal face-spanning subgraph of a plane graph. We have calculated the upper bound of the algorithm which is also tight. We have used minimal face-spanning subgraph to find approximate solution of the the face-spanning subgraph problem and the minimum-vertex face-spanning subgraph problem. The approximation ratios and complexities have also been analyzed for the approximation algorithms. Below we summarize each chapter and its contribution.

In Chapter 1 we have a brief description of the problems we have addressed in this thesis and discussed our motivation behind solving these problems. We also reviewed the literature about these problems in the chapter. We have presented that how the practical applications like establishing gas pipelines in a locality, establishing power transmission lines in a city, power wires layout in a complex circuit, planning irrigation canal networks



for irrigation systems etc can be modeled using graph theoretical terms. We have also described our main result of the this in this chapter. We have described the scope of this thesis in this chapter too.

In Chapter 2 we have introduced basic graph theoretical terminologies that have been used throughout the thesis. We have presented a brief description of two known  $NP$ -complete problems, connected vertex cover problem and weighted tree cover problem in this chapter.

In Chapter 3 we proved that the face-spanning subgraph problem is  $NP$ -complete. To prove that the face-spanning subgraph problem is  $NP$ -complete, we use two steps. First, we have shown that the face-spanning subgraph problem is in  $NP$ . As a second step, we prove that the face-spanning subgraph problem is  $NP$ -hard. These steps immediately proved that the face-spanning subgraph problem is  $NP$ -complete.

In Chapter 4 we proved a variation of the face-spanning subgraph problem, which we call the minimum-vertex face-spanning subgraph problem is  $NP$ -complete. The problem often arises in applications like establishing base transceiver stations in wireless networks, establishing power distribution centers in a city etc where the setup cost for each establishment is huge. We have proved that the minimum-vertex face-spanning subgraph problem is  $NP$ -complete. First, we have shown that the minimum-vertex face-spanning subgraph problem is in  $NP$ . As a second step, we have proved that the minimum-vertex face-spanning subgraph problem is  $NP$ -hard. These two steps immediately proved that the minimum-vertex face-spanning subgraph problem is  $NP$ -complete.

In Chapter 5 we have introduced minimal face-spanning subgraph of a plane graph. This minimal face-spanning subgraph is used to find approximate solution of the the face-spanning subgraph problem and the minimum-vertex face-spanning subgraph problem. We have established a tight lower bound on the number of vertices of a face-spanning subgraph of a plane graph. We have designed a linear time algorithm to find a minimal face-spanning subgraph of a plane graph. We have calculated the upper bound of the

algorithm which is also tight. We have presented an approximation algorithm for finding minimum face-spanning subgraph of a plane graph. The approximation ratio of the presented approximation algorithm is  $\{(n - n_0)(\Delta - 2)e_{max}\}/\{(f - \Delta)e_{min}\}$  and the time complexity of the algorithm is linear. We have also designed an approximation algorithm for finding a minimum-vertex face-spanning subgraph of a plane graph. The approximation ratio of the presented approximation algorithm is  $2(\Delta - 2)$  and the time complexity of the algorithm is linear.

This is the first time that the face-spanning subgraph problem and the minimum-vertex face-spanning subgraph problem are introduced in literature and the  $NP$ -completeness of both the problems are proved. Linear-time approximation algorithms have also been presented in this thesis. However, the following problems related to the face-spanning subgraph problem and the minimum-vertex face-spanning subgraph problem are still *open*.

1. Develop an approximation algorithm for finding a minimum face-spanning subgraph of a plane graph with better approximation ratio.
2. Develop an approximation algorithm for finding a minimum-vertex face-spanning subgraph of a plane graph with better approximation ratio.
3. Design an algorithm to find all minimal face-spanning subgraphs of a plane graph.

# Bibliography

- [AFL05] F. N. Abu-Khzam , H. Fernau and M. A. Langston, *Asymptotically Faster Algorithms for Parameterized face cover*, International Workshop on Algorithms and Complexity in Durham, 4, pp. 43-58, 2005.
- [AHH93] E. M. Arkin, M. M. Halldorsson and R. Hassin, *Approximating the tree and tour covers of a graph*, Information Processing Letters, 47(6), pp. 275-282, 1993.
- [AHU74] A. V. Aho, J. E. Hopcroft and J. D. Ullman, *The design and analysis of computer algorithms*, Addison-Wesley, Reading, MA, 1974.
- [AL04] F. N. Abu-Khzam and M. A. Langston, *A direct algorithm for the parameterized face cover problem*, Proceedings of IWPEC 2004, LNCS 3162, pp. 213-222, 2004.
- [CCPS98] W. J. Cook, W. H. Cunningham, W. R. Pulleyblank and A. Schrijver, *Computer Optimization*, John Wiley & Sons, Inc., New York, 1998.
- [CLR90] T. M. Cormen, C. E. Leiserson and R. L. Rivest, *Introduction to Algorithms*, MIT Press, 1990.
- [FD04] T. Fujito and T. Doi, *A 2-approximation NC algorithm for connected vertex cover and tree cover*, Information Processing Letters, 90(2), pp. 59-63, 2004.
- [GJ77] M. R. Garey and D. S. Johnson, *The rectilinear steiner tree problem is NP-complete*, SIAM Journal on Applied Mathematics, 32(4), pp. 826-834, 1977.

- [GJ79] M. R. Garey and D. S. Johnson, *Computers and Intractability: A guide to the theory of NP-completeness*, W. H. Freeman, San Francisco, New York, 1979.
- [GK98] S. Guha and S. Khuller, *Approximation algorithms for connected dominating sets*, *Algorithmica*, 20(4), pp.374 - 387, 1998.
- [HSR98] E. Horowitz, S. Sahni and S. Rajasekaran, *Fundamentals of Computer Algorithms*, Galgotia Publications Pvt. Ltd., New Delhi, 1998.
- [K72] R. M. Karp, *Reducibility among combinatorial problems*, *Complexity of Computer Computations*, R. E. Miller and J. W. Thatcher(eds.), Plenum Press, New York, pp. 85-104, 1972.
- [KKPS03] J. Konemann, G. Konjevod, O. Parekh and A. Sinha, *Improved approximations for tour and tree covers*, *Algorithmica*, 38(3), pp. 441-449, 2003.
- [NR04] T. Nishizeki and M. S. Rahman, *Planar Graph Drawing*, World Scientific, Singapore, 2004.
- [V01] V. V. Vazirani, *Approximation Algorithms*, Springer-Verlog, Berlin, 2001.
- [W01] D. B. West, *Introduction to Graph Theory*, Prentice-Hall, Inc., New Jersey, 2001.
- [YE81] R. B. Yehuda and S. Even, *A Linear-Time Approximation Algorithm for the Weighted Vertex Cover Problem*, *Journal of Algorithms*, 2, pp. 198-203, 1981.

# Index

- adjacent, 13
- approximation
  - algorithm, 20
  - ratio, 20
- complexity class, 17
  - NP*, 17
  - NP*-complete, 19
  - NP*-hard, 18
  - P*, 17
- component
  - connected, 14
- connected vertex cover, 22
  - minimum, 23
  - problem, 23
- cost of-subgraph, 16
- covers a face, 16
- covers all the edges, 16
- cycle, 14
- decision problem, 16
- degree, 13
  - maximum, 13
- edge weighted connected plane graph, 16
- face, 15
- face cover, 6
  - problem, 6
- face-spanning subgraph, 2
  - NP*, 26
  - NP*-complete, 26, 33
  - NP*-hard, 28, 33
  - applications, 2
  - approximation algorithm, 52
  - minimum, 2
  - problem, 2
  - problem definition, 24
- graph
  - connected, 14
  - disconnected, 14
  - multi, 13
  - planar, 15
  - plane, 15
  - simple, 13
- incident, 13

- loop, 13
- minimal face-spanning subgraph, 46  
 algorithm, 49  
 lower bound, 47  
 upper bound, 50
- minimum-vertex face-spanning subgraph,  
 9  
*NP*, 37  
*NP*-complete, 37, 43  
*NP*-hard, 38, 43  
 applications, 9  
 approximation algorithm, 51  
 problem, 9  
 problem definition, 36
- neighbor, 13
- open problems, 56
- optimization problem, 17
- path, 14  
 closed, 14
- reduction, 18
- subgraph, 14  
 edge induced, 15  
 spanning, 14
- tree, 14  
 spanning, 15  
 tree cover, 16, 21  
 vertex cover, 6  
 problem, 6  
 weighted tree cover, 21  
 minimum, 21  
 problem, 21

