M.Sc. Engg. Thesis

SPANNING PATHS, CYCLES AND TREES: PROBLEMS AND RESULTS

By

Mohammad Sohel Rahman

Submitted to

Department of Computer Science & Engineering

In partial fulfillment of the requirements for the degree of

M.Sc. Engg. (Computer Science & Engineering)

Department of Computer Science & Engineering

Bangladesh University of Engineering & Technology

Dhaka-1000, Bangladesh

January 24, 2004

The thesis, SPANNING PATHS, CYCLES AND TREES: PROBLEMS AND RESULTS, submitted by MOHAMMAD SOHEL RAHMAN, ROLL No. 040205030P, Session April, 2002, Registration No. 95389 to the Department of Computer Science and Engineering of Bangladesh University of Engineering and Technology has been accepted as satisfactory for partial fulfillment of the requirements for the degree of M.Sc. Engg. in Computer Science and Engineering and approved as to its style and contents. Examination held on January 24, 2004.

BOARD OF EXAMINERS

1. ______________________________ **Chairman (Supervisor)**

(Dr. M. Kaykobad)
Professor
Department of CSE
BUET, Dhaka-1000

2. ______________________________ **Member**

(Dr. Md. Shamsul Alam)
Professor and Head
Department of CSE
BUET, Dhaka-1000

3. ______________________________ **Member**

(Dr. Md. Mostofa Akbar)
Assistant Professor
Department of CSE
BUET, Dhaka-1000

4. M. M. Islam ______________________________ **Member**

(Dr. Md. Monirul Islam)
Assistant Professor
Department of CSE
BUET, Dhaka-1000

5. ______________________________ **Member (External)**

(Dr. Md. Mozammel Huq Azad Khan)
Professor
Department of CSE
East West University, Dhaka

DECLARATION

I, hereby, declare that the work presented in this thesis is done by me under the supervision of Dr. M. Kaykobad, Professor, Department of Computer Science and Engineering, Bangladesh University of Engineering and Technology, Dhaka-1000. I also declare that neither this thesis nor any part thereof has been submitted elsewhere for the award of any degree or diploma.

Countersigned

Dr. M. Kaykobad
(Supervisor)

(Mohammad Sohel Rahman)

ACKNOWLEDGEMENTS

All praises are for the Almighty ALLAH who is the most beneficent and the most merciful.

First of all, I would like to thank my supervisor Dr. M. Kaykobad, Professor, Department of CSE, BUET for teaching me how to carry on a research work in the highest level. I really appreciate his extraordinary patience in each and every of the numerous research sessions we had and his constant encouragements in the face of extreme difficulty to find results. I again express my heart-felt and most sincere gratitude to him for his constant supervision, valuable advice and continual encouragement, without which this thesis would have not been possible.

I would like to thank Dr. Md. Abul Kashem Mia, Associate Professor, Department of CSE, BUET for introducing me to Graph Theory, a fascinating and ever-growing field of Computer Science, when he supervised my B.Sc. Engg. thesis.

I would like to express my utmost gratitude to Dr. Md. Mostofa Akbar, Assistant Professor, Department of CSE, BUET for his constant support and valuable discussions frequently made with him.

I would like to thank the other members of my examination board, Prof. Dr. Shamsul Alam, Dr. Monirul Islam and Prof. Dr. M. H. A. Khan for their valuable suggestions and of course for their unconditional encouragements.

I would also like to thank my brother Md. Saifur Rahman, a final year CSE student in BUET for his fruitful discussions. He is a co-author of a submitted paper corresponding to the work described in Section 3.2. I would also have to thank anonymous referees of our published papers for their invaluable comments and suggestions.

I must have to acknowledge with sincere thanks the constant support and patience of my family members- my parents and my wife. The arduous task of having completed the thesis could not have been accomplished without there patience and understanding.

Finally I would like to acknowledge the all-out cooperation and services rendered by the members of the Department of CSE, BUET.

Contents

List of Figures

List of Tables

Abstract

This thesis deals with the classical graph theoretic structures- spanning paths, cycles and trees. We here present new sufficient conditions for a graph to possess Hamiltonian (spanning) paths. New sufficient conditions for a graph to be Hamiltonian are also presented. We here basically deal with degree related conditions. Also, a new parameter, namely shortest path distance, is introduced in a sufficient condition we present. The infamous Ore's theorem is shown to follow from our results, which prove the significance of our conditions. Furthermore the relation between independence number of a graph and hamiltonicity is also explored in this thesis.

Spanning trees have numerous applications in both practical and theoretical problems. We here present some new spanning tree problems and consider the issues of their complexity. The relation between independence number and a special spanning tree, namely degree bounded spanning tree is also explored. Finally, we introduce a new notion of "set version" of some decision problems having integer $K < |V|$ as a parameter in the input instance, where we replace K by a set $X \subseteq |V|$. For example, the set version of maximum leaf spanning tree problem asks whether there exists a spanning tree in G that contains X as a subset of the leaf set. We raise the issue of whether the set versions of NP-Complete problems are as hard as the original problems and prove that although in some cases the set versions are easier to solve, this is not necessarily true in general.

Chapter 1

INTRODUCTION

1.0 Introduction

A graph is an abstract structure that is used to model information. Precisely, graphs can represent any information that can be modeled as objects and connections between those objects. For example, graphs are used to describe hierarchies and interconnections of components in computer networks; each component is represented as a vertex in a graph and the connection between components x and y is represented by an edge from x to y.

There are various interesting structures of graphs that has been introduced and exploited for various theoretical and practical problems. Paths, cycles and trees are very basic graph structures and have always been extensive focus of research. Basically many new research areas of computer science have been evolved based on significant works done on these basic structures, and in almost every area these findings based on paths, cycles and trees have found to provide significant contributions.

A very interesting area based on paths cycles and trees are the 'spanning' versions of these structures i.e. 'spanning paths', 'spanning cycles' and 'spanning trees'. Historically spanning paths and cycles are popularly known as Hamiltonian paths and Hamiltonian cycles, respectively. Research on this particular area is motivated by numerous practical applications based on these structures [4, 9, 38, 39, 42, 43]. We discuss a few of the interesting practical applications below.

Electronic Circuitry Design-The classical and popular example of a spanning tree application contributes to the design of electronic circuitry [8]. It is often necessary to make the pins of several components electrically equivalent by wiring them together. To interconnect a set of n pins, we can use an arrangement of n-1 wires, each connecting two pins. Of all such arrangements, the one that uses the least amount of wire is usually the most desirable. This is in fact the famous minimum-weight spanning tree problem.

Minimum Connector Problem- Suppose we wish to build a railway network connecting n given cities so that a passenger can travel from any city to any other with the constraint that the total amount of track must be a minimum (in fact this should be the aim of the government building the network since minimum track would ensure minimum cost of construction). So in effect the problem is to find an efficient algorithm for deciding which of the n^{n-2} possible spanning trees uses the least amount of track.

Enumeration of Chemical Molecule- If a molecule has only carbon atoms and hydrogen atoms, then it can be represented as a graph where each C is a vertex of degree 4 and each H is a vertex of degree 1. Now consider a general class of molecules known as the *alkanes* with chemical formula C_nH_{2n+2}. Now a question is how many different molecules are there with this formula. We can use the tree structure to solve this problem since the graph of any molecule with formula C_nH_{2n+2} is a tree.

There are various other practical applications of spanning paths, cycles and trees. Problems involving spanning trees and Hamiltonian cycles (and paths) arise naturally into many network situations. Research on fault tolerance, for example, extensively exploits the theory of Hamiltonicity. Many of these problems arise in Internet context too.

Finally, in many other situations graphs and these structures are not directly suggested by the problem itself, but nevertheless they can be invaluable tool. Such indirect uses of minimum spanning tree, for instance, include optimal broadcast in an unreliable medium, optimal data storage in two-dimensional arrays, min-max path problems, and cluster analysis.

1.1 Literature Review

In this section we present a brief literature review of the topics of our thesis. As the title indicates, our thesis may be classified into two major sections; one dealing with the spanning paths and cycles i.e. Hamiltonian paths and Hamiltonian cycles and the other dealing with the spanning trees. First, in the following section, we consider Hamiltonian paths and cycles followed by a section devoted to spanning trees.

1.1.1 Hamiltonian (Spanning) Paths and Cycles

In this section we define Hamiltonian cycle and path and present a brief literature review of topics related to our thesis. A Hamiltonian cycle is a spanning cycle in a graph i.e. a cycle through every vertex and a Hamiltonian path is a spanning path. A graph containing a Hamiltonian cycle is said to be Hamiltonian. It is clear that every graph with a Hamiltonian cycle has a Hamiltonian path but the converse is not necessarily true. The study of Hamiltonian cycles and Hamiltonian paths in general and special graphs has been fueled by practical applications and by the issues of complexity. The problem of finding whether a graph G is Hamiltonian is proved to be NP-Complete for general graphs [13]. The problem remains NP-Complete [13] (i) if G is planar, cubic, 3-connected, and has no face with fewer than 5 edges, (ii) if G is bipartite, (iii) if G is the square of a graph, (iv) if a Hamiltonian path for G is given as part of the instance. On the other hand the problem of finding whether a graph G contains a Hamiltonian path is also proved to be NP-Complete for general graphs [13]. Again it remains NP-Complete (i) if G is planar, cubic, 3-connected, and has no face with fewer than 5 edges, (ii) if G is bipartite. Even the variant in which either the starting point or the end point or both are specified in the input instance is also NP-Complete. No easily testable characterization is known for Hamiltonian graphs. Nor there exists any such condition to test whether a graph contains a Hamiltonian path or not. This is why tremendous amount of research has been done in finding sufficient conditions for the existence of Hamiltonian cycles or Hamiltonian paths in graphs [1, 10, 32].

Before presenting some of the conditions in the literature we need to introduce and define some of the notations we use. Given a graph $G = (V, E)$ and a vertex $u \in V$, we mean by $d(u)$ the degree of u in G. In other words $d(u) = |N_G(u)|$, where $N_G(u)$ denotes the neighbor set of u in a graph G. If $H \subseteq G$ then $d_H(u) = |N_H(u)|$ and $d_{\bar{H}}(u) = |N_{G \setminus H}(u)|$. By $\delta(u, v)$ we denote the length of a shortest path between u and v

in G. On the other hand, by $\delta(G)$ we indicate the degree of a minimum degree vertex in G. $V[G]$ and $E[G]$ are used to denote, respectively, the vertex set and edge set of G.

Now we are ready to list some of the conditions in the literature for the existence of Hamiltonian cycles or paths in graphs.

Theorem 1.1.1 (Dirac [10]). *If G is a simple graph with n vertices where $n \geq 3$ and $\delta(G) \geq n/2$, then G is Hamiltonian.* □

Theorem 1.1.2 (Ore [32]). *Let G be a simple graph with n vertices and u, v be distinct nonadjacent vertices of G with $d(u) + d(v) \geq n$. Then G is Hamiltonian if and only if $G + (u, v)$ is Hamiltonian.* □

Theorem 1.1.3 (Bondy-Chvâtal [1]). *If G is a simple graph with n vertices, then G is Hamiltonian if and only if its closure is Hamiltonian..* □

Remark: *The (Hamiltonian) closure of a graph G, denoted C(G), is the supergraph of G on V(G) obtained by iteratively adding edges between pairs of nonadjacent vertices whose degree sum is at least n, until no such pair remains. Fortunately, the closure does not depend on the order in which we choose to add edges when more than one is available i.e. the closure of G is well defined (For a proof of this statement see [44]).*

Theorem 1.1.4 (Ore [32]). *If $d(u) + d(v) \geq n$ for every pair of distinct nonadjacent vertices u and v of G, then G is Hamiltonian.* □

In this thesis, we particularly focus on degree related conditions of Hamiltonian paths and cycles. Also we consider the independence number of a graph, and establish a connection between the existence of a Hamiltonian path or cycle and the independence number of a graph.

1.1.2 Spanning Trees

In this section we discuss spanning trees and related problems and results relevant to our thesis. A spanning subgraph of a graph $G = (V, E)$ is a subgraph with vertex set V. A spanning tree is a spanning subgraph that is a tree. Spanning trees have been found to be structures of paramount importance also in practical applications. As a result spanning trees of a connected graph have been the focus for extensive attention in graph theoretic research. In the following subsection we first present some background information and literature review of various research on spanning trees.

Spanning trees with various constraints and restricted conditions seem to pose various interesting problems [2, 19, 23, 25, 33, 37, 46]. This is why extensive amount of research focus has been on this particular structure and variants thereof. However, the most popular and may be, the first important problem on spanning tree was the famous Minimum Spanning Tree Problem as defined below.

Problem 1.1.2.1 Minimum Spanning Tree

Given a weighted connected graph G = (V, E) we wish to find out a spanning tree T of G such that $w(T) = \sum_{(u,v)\in T} w(u,v)$ *is minimized i.e. the summation of the weights of the edges of the spanning tree is minimized among all possible spanning trees of G.*

Note carefully that the phrase "Minimum Spanning Tree" is a shortened form of the phrase "Minimum-weight Spanning Tree". We are not, for example, minimizing the number of edges in *T*, since it is easy to see that all spanning trees have exactly $|V| - 1$ edges. Tremendous amount of research has been done on this particular problem. The most important, popular and cited two polynomial time algorithms for finding a Minimum Spanning Tree have to be the Prim's and Kruskal's Algorithm [8]. The most important, popular and cited two polynomial time algorithms for finding a Minimum Spanning Tree have to be the Prim's and Kruskal's Algorithm. Tarjan [41] surveys the minimum spanning tree problem and provides excellent advanced material. A history of the minimum spanning tree problem has been written by Graham and Hell [14]. Tarjan attributes the first minimum spanning tree algorithm to a 1926 paper by O. Borüvka. Kruskal's algorithm was reported by Kruskal [27] in 1956. The algorithm commonly known as the Prim's algorithm was indeed invented by Prim [36], but it was also invented earlier by V. Jarnik in 1930.

Research has also been done on devising parallel [7] and randomized [24] algorithms for Minimum Spanning Tree. The complexity of Minimum Spanning Tree problem, although proved to be polynomial long before, still is considered by many researchers from various point of view [5, 7]. Although it might seem that all the issues regarding Minimum Spanning Tree problem has been settled (Complexity issue, number of polynomial time algorithms, number of parallel algorithms etc.) every now and then a new issue seems to evolve and consequently researchers start digging resulting in more newer unexplored research issues. A very recent advance on this particular problem and also on various related spanning tree problems in general is to apply the idea of generalization. Generalized Minimum Spanning Tree Problem, first introduced by Myung, Lee and Tcha [30] was considered, studied and explored comprehensively by Pop [35].For better understanding with a brief discussion, we first start with defining a combinatorial problem as follows.

Given a graph $G = (V, E)$ and a cost function $c : E \rightarrow \Re$, following the definition of Nemhauser and Wolsey [31], a *combinatorial optimization* problem consists of determining among a finite set of feasible solutions those that minimize the cost function. If we let F be a family of subsets of the edge set E and denote by $c(F) = \sum_{e \in F} c_e$ $F \subseteq E$, a combinatorial optimization problem in its minimization form is: min { $c(F)$: $F \subseteq E$ }. Classical combinatorial optimization problems can often be *generalized* in a natural way by considering a related problem relative to a given *partition* $V = V_1 \cup V_1 \cup ... \cup V_m$ of the nodes into *clusters* $Vk \subseteq V$, $k \in \{1, 2, ..., m\}$ such that the classical problem corresponds to the trivial partition $Vk = \{k\}$ into singletons. Consider the formal definition of the Generalized Minimum Spanning Tree Problem below as an example.

Problem 1.1.2.2. Generalized Minimum Spanning Tree Problem

Given a graph $G = (V, E)$ and a cost function $c : E \rightarrow \Re$, we are asked for a cost-minimal tree T in G which spans exactly one node $i_k \in V_k$ in each cluster.

In this thesis, however, we look into a different set of combinatorial problems and basically deal with the decision problems instead of the optimization problems. In particular we are interested in spanning trees with various restrictions and constraints applied on various graph parameters. For example, consider the following problems and the known results.

Problem 1.1.2.3. Degree Constrained Spanning Tree Problem.

Given a connected graph $G = (V, E)$ and a positive integer $K < |V|$, we are asked the question whether there is a spanning tree T of G such that no vertex in T has degree larger than K.

Theorem 1.1.2.4. (See [13]) *Degree Constrained Spanning Tree Problem is NP-Complete.*□

Remark (See [13]). Problem 1.1.2.3. remains NP-Complete for any fixed $K \geq 2$.

Problem 1.1.2.5. Maximum Leaf Spanning Tree Problem.

Given a connected graph $G = (V, E)$ and a positive integer $K < |V|$, we are asked the question whether there is a spanning tree T of G such that K or more vertices in T have degree 1.

Theorem 1.1.2.6. (See [13]) *Maximum Leaf Spanning Tree Problem is NP-Complete.* □

Remark (See [13]). Problem 1.1.2.5. remains NP-Complete if G is regular of degree 4 or if G is planar with no degree exceeding 4.

In this thesis we first introduce some new problems (with relevant new results) where we impose various constraints and restrictions on parameters of spanning trees. We investigate the relationship of the independence number and a special spanning tree namely degree bounded spanning tree. Finally, we introduce a new notion "set version". The complexities of the set versions of various problems are discussed and we show that surprisingly and remarkably "set versions" of some NP-Complete problems are solvable in polynomial time although this may not necessarily be the case all the time.

1.3 Objective of This Thesis

In this thesis we present and investigate various problems, conditions and results of spanning paths, cycles and trees. Our main results can be divided into 2 parts.

The first part of our results is on Hamiltonian (spanning) paths and Hamiltonian (spanning) cycles. We present new sufficient conditions for a graph to possess

Hamiltonian paths and Hamiltonian cycles and discuss the significance of our new conditions. We start with our main focus i.e. degree related conditions and present new degree related sufficient conditions for Hamiltonicity. We also introduce new parameters namely shortest path distance between nonadjacent vertices in a presented sufficient condition. Also, we investigate the relationship between independence number, another important graph parameter, and Hamiltonicity of graphs.

The second part of our results is on spanning trees. We first pose some new spanning tree problems, and investigate the corresponding issues of complexity. We also investigate the relation between the independence number and degree bounded spanning tree (to be defined shortly). Finally we present a new notion of "set version" and apply this new idea on various spanning tree problems. Remarkably and somewhat surprisingly it is shown that set version of a spanning tree problem (maximum leaf spanning tree problem, to be specific) is polynomially solvable in spite of the NP-Completeness of the original problem. Intuitively, this fact indicated that set versions of some "hard" problems may be found to be easily solvable. However, it is further shown that, this particular trend is not necessarily true in general. In particular it is shown that both the original and the set version of minimum leaf spanning tree problem are NP-Complete.

1.4 Thesis Organization

Our thesis, as is mentioned before, can be organized into two main parts; one dedicated to spanning paths and cycles and the other to spanning trees. We start with some preliminary definitions in Chapter 2. Here we define some graph theoretic terms and present a brief idea about the notion of complexity.

Chapter 3 and 4 constitutes the heart of our thesis; Chapter 3 dealing with Hamiltonian paths and cycles and Chapter 4 with spanning trees. In Chapter 3 we present all our results on Hamiltonian paths and cycles. We start with a brief literature review, present our sufficient conditions and discuss the significance of our results.

In Chapter 4, after a brief literature review about spanning trees, we begin with investigating the relationship between independence number and a special spanning tree, namely degree bounded spanning tree. The corresponding algorithm with complexity analysis is also presented. Then we pose new problems to settle their complexity issues. Finally, we introduce a new notion of "set version" of some decision problems. We raise the issue of whether the set versions of NP-Complete problems are as hard as the original problems and prove that although in some cases the set versions are easier to solve, this is not necessarily true in general. We end our thesis with Chapter 5 with a brief summary of our results and by shading some light to the future research areas.

Chapter 2

PRELIMINARIES

2.0 Introduction

In this chapter we define some basic terms of graph theory and algorithm theory. Definitions, which are not included in this chapter, will be introduced as they are needed. We start, in Section 2.1, by giving some definitions of standard graph-theoretical terms used throughout this thesis. In Section 2.2, we introduce the notion of time complexity.

2.1 Basic Terminology

In this section we give some definitions of standard graph-theoretical terms used throughout this thesis. For a better understanding of various graph theoretic terms and for interested readers on graph theory we refer to [44].

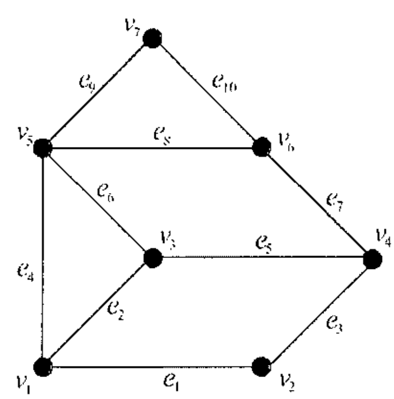


Figure 2.1: A Graph with seven vertices and ten edges

2.1.1 Graphs and Multigraphs

A graph G is a structure (V, E), which consists of a finite set of vertices V and a finite set of edges E; each edge is an unordered pair of distinct vertices. We denote the set of vertices of G by V(G) and the set of edges by E(G). Figure 2.1 depicts a graph G, where each vertex in $V(G) = \{v_1, v_2, \ldots, v_7\}$ is drawn by a small black circle and each edge in $E(G) = \{e_1, e_2, \ldots, e_{10}\}$ is drawn by a line segment. Throughout this thesis 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, $m = |E|$. Thus for the graph in Figure 2.1 $n = 7$ and $m = 10$.In a graph, multiple edges join the same pair of vertices, while a loop joins a vertex to itself. The graph in which loops and multiple edges are allowed is called a multigraph. If a graph G has no "multiple edges" or "loops", then G is said to be a simple graph. Sometimes a simple graph is simply called a graph if there is no danger of confusion. In the remainder of the thesis it is assumed that G has no multiple edges or loops.

We denote an edge between two vertices u and v of G by (u, v) or simply by uv. If $(u, v) \in E$, then two vertices u and v of graph G are said to be adjacent; edge (u, v) is then said to be incident to vertices u and v; u is a neighbor of v. The degree of a vertex v in G is the number of edges incident to v and is denoted by $d_G(v)$ or simply d(v). In the graph in Figure 2.1 vertices v_1 and v_2 are adjacent, and $d(v_1) = 3$, since three edges e_1, e_2 and e_4 are incident to v_1.

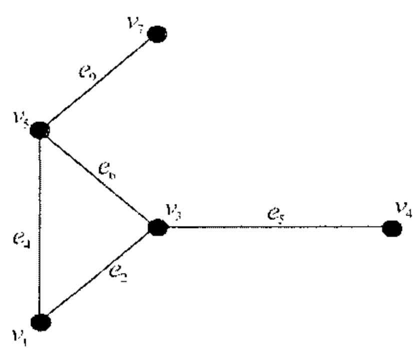


Figure 2.2: A vertex-induced subgraph

2.1.2 Subgraphs

A subgraph of a graph $G = (V, E)$ is a graph $G' = (V', E')$ such that $V' \subseteq V$ and $E' \subseteq E$; we then write $G' \subseteq G$. If G′ contains all the edges of G that join two vertices in V′, then G′ is said to be the subgraph induced by V′. Figure 2.2 depicts a subgraph of G in Figure 2.1 induced by $\{v_1, v_3, v_4, v_5, v_7\}$. A spanning subgraph of a graph $G = (V, E)$ is a subgraph of G such that $V' = V$ and $E' \subseteq E$.

We often construct new graphs from old ones by deleting some vertices or edges. If v is a vertex of a given graph G = (V, E), then G – v is the subgraph of G obtained by deleting the vertex v and all the edges incident to v. More generally, if V′ is a subset of V, then G – V′ is the subgraph of G obtained by deleting the vertices in V′ and all the edges incident to them. Then G – V′ is a subgraph of G induced by V – V′. Similarly, if e is an edge of G, then G – e is the subgraph of G obtained by deleting the edge e. More generally, if E′ ⊆ E, then G – E′ is the subgraph of G obtained by deleting the edges in E′.

2.1.3 Connectivity

A graph G is connected if for every pair {u, v} of distinct vertices, there is a path between u and v. 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 2.3 (a) is a connected graph since there is a path for every pair of distinct vertices of the graph. On the contrary, since there is no path between v_1 and v_5, the graph in Figure 2.3 (b) is a disconnected graph with two connected components indicated by dotted lines.

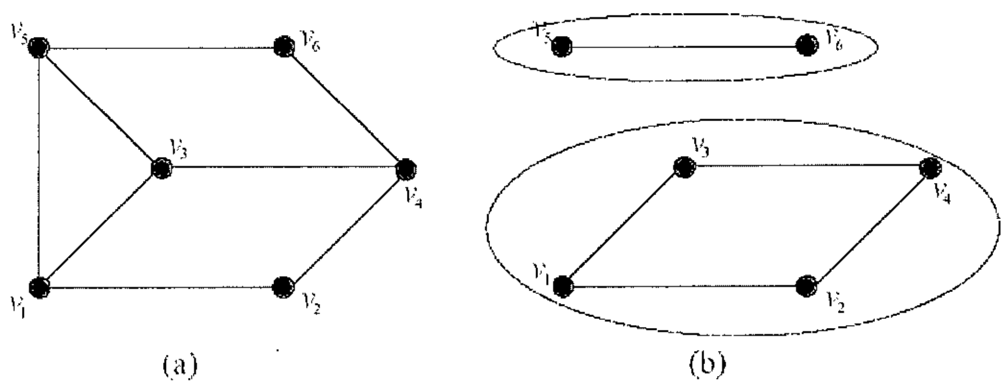


Figure 2.3: (a) A connected graph, and (b) a disconnected graph with two connected components

The connectivity κ(G) of a graph G is the minimum number of vertices whose removal results in a disconnected graph or a single-vertex graph K_1. We say that G is k-connected if $\kappa(G) \geq k$. We call a set of vertices in a connected graph G a separator or a vertex-cut if the removal of the vertices in the set results in a disconnected or single-vertex graph. If a vertex-cut contains exactly one vertex then we call the vertex a cut vertex.

2.1.4 Paths and Cycles

A $v_0 - v_l$ walk, $v_0, e_1, v_1, \ldots, v_{l-1}, e_l, v_l$, in G is an alternating sequence of vertices and edges of G, beginning and ending with a vertex, in which each edge is incident to two vertices immediately preceding and succeeding it. If the vertices $v_0, v_1, \ldots, v_l$ are distinct (except possibly v_0, v_l), then the walk is called a path and usually denoted either by the sequence of vertices $v_0, v_1, \ldots, v_l$ or by the sequence of edges $e_0, e_1, \ldots, e_l$. The length of the path is l, one less than the number of vertices on the path. A path or walk is closed if $v_0 = v_l$. A closed path containing at least one edge is called a cycle. For example consider the figure below. *(a, x, a, x, u, y, c, d, y, v, x, b, a)* is a closed walk of length 12. Now omitting first two steps yields a closed trail (no edge repetition). The edge set of this trail is the union of the edge sets of 3 pairwise edge-disjoint cycle: (*a, x, b, a*), (*x, u, y, v, x*) and (*y, c, d, y*). Finally, the *u, v*-trail (*u, y, c, d, y, v*) contains the edges of the *u, v*-path: (*u, y, v*).

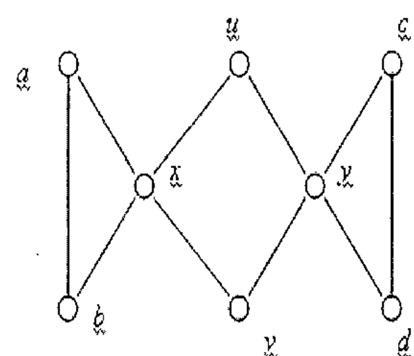


Figure 2.4: Walk, Trail, Cycle and Path

2.1.5 Trees

A tree is a connected graph without any cycle. Figure 2.5 is an example of a tree. The vertices in a tree are usually called nodes. A rooted tree is a tree in which one of the nodes is distinguished from the others. The distinguished node is called the root of the tree. The root of a tree is generally drawn at the top. In Figure 2.5, the root is v_1. Every node u other than the root is connected by an edge to some other node v called the parent of u. We also call u a child of v. For example, in Figure 2.5, v_1 is the parent of v_2, v_3 and v_4, while v_2 is the parent of v_5 and v_6; v_2, v_3 and v_4 are children of v_1, while v_5 and v_6 are children of v_2. A leaf is a node of a tree that has no children. An internal node is a node that has one or more children. Thus every node of a tree is either a leaf or an internal node. In Figure 2.5, the leaves are v_4, v_5, v_7 and v_8, and the nodes v_1, v_2, v_3 and v_6 are internal nodes. A spanning tree of G = (V, E) is a spanning subgraph of G which is a tree. Figure 2.6 depicts a spanning tree of G in Figure 2.1.

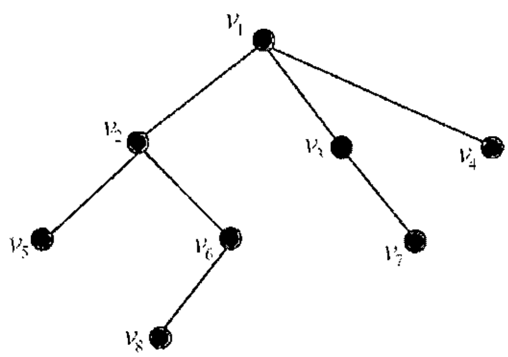


Figure 2.5: A tree

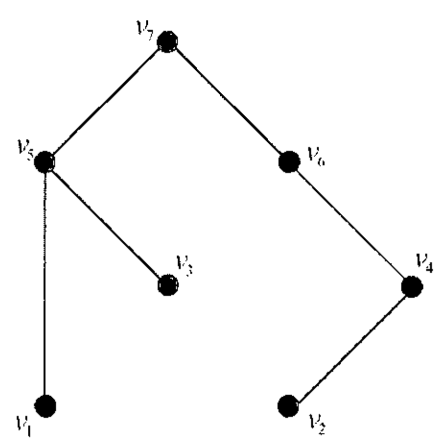


Figure 2.6: A spanning tree

2.2 Algorithms and Complexity

In this section we briefly introduce some terminologies related to complexity of algorithms. For interested readers, we refer the books of Garey and Johnson [13]. The most widely accepted complexity measure for an algorithm is the running time, which is expressed by the number of operations it performs before producing the final answer. The number of operations required by an algorithm is not the same for all problem instances. Thus, we consider all inputs of a given size together, and we define the complexity of the algorithm for that input size to be the worst case behavior of the algorithm on any of these inputs. Then the running time is a function of size n of the input.

2.2.1 The notation O(n)

In analyzing the complexity of an algorithm, we are often interested only in the asymptotic behavior, that is, the behavior of the algorithm when applied to very large inputs. To deal with such a property of functions, we shall use the following notations for asymptotic running time. Let f(n) and g(n) be the functions from the positive integers to the positive reals, then we write $f(n) = O(g(n))$ if there exists positive constants c_1 and c_2 such that $f(n) \leq c_1 g(n)+c_2$ for all n. Thus the running time of an algorithm may be bounded from above by phrasing like "takes time $O(n^2)$."

2.2.2 Polynomial algorithms

An algorithm is said to be polynomially bounded (or simply polynomial) if its complexity is bounded by a polynomial of the size of a problem instance. Examples of such complexities are $O(n)$, $O(n \log n)$ $O(n^{100})$, etc. The remaining algorithms are usually referred to as exponential or nonpolynomial. Examples of such complexity are $O(2^n)$, $O(n!)$, etc. When the running time of an algorithm is bounded by $O(n)$, we call it a linear-time algorithm or simply a linear algorithm.

2.2.3 NP-Completeness

The class of problems solvable by nondeterministic polynomial time algorithms is called "NP" problems. In other words, nondeterministically polynomial time problems are those whose solutions can be verified in polynomial time. A problem is NP-Hard if a polynomial-time algorithm for it could be used to construct a polynomial time algorithm for each problem in NP. It is NP-Complete if it belongs to NP and is NP-Hard. If some NP-Complete problems belong to P (set of polynomially solvable problems), then P = NP. However, researchers have found no polynomial time algorithm for any of the many NP-Complete problems. The implication of these facts is that if we know that a problem is NP-Complete, we should probably not waste our time looking for a polynomial time exact algorithm. This is why, for Hamiltonian cycle or path (both of which are NP-Complete problems [13]), we should be looking for sufficient conditions instead of necessary and sufficient conditions.

Chapter 3

HAMILTONIAN PATHS AND HAMILTONIAN CYCLES

3.0 Introduction

In this section we present our results on Hamiltonian (spanning) paths and cycles. Given a graph G = (V, E) a Hamiltonian cycle C is a spanning cycle of G i.e. a simple cycle of G such that V(C) = V(G), where V(H) denotes the set of vertices of a graph H. Similarly, a Hamiltonian (spanning) path of G is a simple path spanning all the vertices of G. A graph containing a Hamiltonian cycle is said to be Hamiltonian. It is clear that every graph with a Hamiltonian cycle has a Hamiltonian path but the converse is not necessarily true.

The study of Hamiltonian cycles and Hamiltonian paths in general and special graphs has been fueled by practical applications and by the issues of complexity. The problem of finding whether a graph *G* is Hamiltonian is proved to be NP-Complete for general graphs [13]. The problem remains NP-Complete [13] (i) if *G* is planar, cubic, 3-connected, and has no face with fewer than 5 edges, (ii) if *G* is bipartite, (iii) if *G* is the square of a graph, (iv) if a Hamiltonian path for *G* is given as part of the instance. On the other hand the problem of finding whether a graph *G* contains a Hamiltonian path is also proved to be NP-Complete for general graphs [13]. Again it remains NP-Complete (i) if *G* is planar, cubic, 3-connected, and has no face with fewer than 5 edges, (ii) if *G* is bipartite. Even the variant in which either the starting point or the end point or both are specified in the input instance is also NP-Complete. No easily testable characterization is known for Hamiltonian graphs. Nor there exists any such condition to test whether a graph contains a Hamiltonian path or not. This is why tremendous amount of research has been done in finding the sufficient conditions for the existence of Hamiltonian cycles or Hamiltonian paths in graphs [1, 10, 32].

3.1 Degree Sum and Distance of Vertices

In this section we present some sufficient conditions of Hamiltonian cycles and paths. Sufficient conditions in the literature are based on various graph parameters. In this section we are particularly interested in degree related conditions. In addition to the "degree" parameter of graphs, we introduce a new parameter namely the "distance" between two vertices, which is essentially the shortest path distance between any two given vertices.

Lemma 3.1.1. *Let $G = (V, E)$ be a connected graph with n vertices and P be a longest path in G. If P is contained in a cycle then P is a Hamiltonian path.*

Proof. Suppose $P \equiv \langle u = u_0, u_1, u_2, \ldots, u_k = v\rangle$ of length k and P is contained in a cycle $C \equiv \langle u = u_0, u_1, u_2, \ldots, u_k = v, u_0 = u\rangle$. Note that $V(C) = V(P)$, since otherwise P would be a part of a longer path, a contradiction. Assume for the sake of contradiction that $k < n - 1$, i.e. P is not Hamiltonian path. Since G is connected, there must be an edge of the form (x, y) such that $x \in V(P) = V(C)$ and $y \in V(G) - V(C)$. Let $x = u_i$. Then there is a path $P' \equiv \langle y, x = u_i, u_{i+1}, \ldots, u_k, u_0, u_1, u_2, \ldots, u_{i-1}\rangle$ with length $k+1$, which is a contradiction, since P is a longest path in G. □

Corollary 3.1.2. *Let $G = (V, E)$ be a connected graph with n vertices and P be a longest path in G. If P is contained in a cycle then G is Hamiltonian.* □

Now we present and prove our first sufficient condition in the form of Theorem 3.1.3.

Theorem 3.1.3. *Let $G = (V, E)$ be a connected graph with n vertices and P be a longest path in G having length k and with end vertices u and v. Then the followings must hold true:*

a. *Either $\delta(u, v) > 1$ or P is a Hamiltonian path contained in a Hamiltonian cycle.*

b. *If $\delta(u, v) \geq 3$ then $d_p(u) + d_p(v) \leq k - \delta(u, v) + 2$*

c. *If $\delta(u,\ v)\ =\ 2$, then either $d_p(u)\ +\ d_p(v)\ \leq\ k$ or P is a Hamiltonian path contained in a Hamiltonian cycle.*

Proof.

Part a:

Assume $\delta(u, v) \leq 1$. Since the graph is connected $\delta(u, v) = 1$. Let $P \equiv \langle u = u_0, u_1, u_2, \ldots, u_k = v\rangle$. Since $\delta(u, v) = 1$, we in effect have a cycle $C \equiv \langle u = u_0, u_1, u_2, \ldots, u_k = v, u_0 = u\rangle$ and the result readily follows from Lemma 3.1.1. □

Part b:

Assume $\delta(u, v) \geq 3$. In this case surely $d_p(u) + d_p(v) \leq k - \delta(u, v) + 2 \leq k - 3 + 2 = k - 1$, since otherwise we would get a path from u to v with length less than $\delta(u, v)$, a contradiction. □

Part c:

Assume that $\delta(u, v) = 2$. Now note that we can not claim that $d_p(u) + d_p(v) \leq k - \delta(u, v) + 2 = k$ by arguing contradiction on $\delta(u, v)$ as we did in *part b* because now there is a common vertex adjacent to both u and v. However, we argue in a different way as follows. Assume that $d_p(u) + d_p(v) \geq k + 1 = |V(P)|$. We rewrite the path P as follows. $P \equiv \langle v = w_1, w_2, \ldots, w_{|V(P)|-1}, w_{|V(P)|} = u\rangle$. Now we will try to find out two cross over edges (v, w_{i+1}) and (w_i, u) such that we get the cycle $C = \langle w_1, w_{i+1}, w_{i+2}, \ldots, w_{|V(P)|-1}, w_{|V(P)|}, w_i, w_{i-1}, \ldots, w_2, w_1\rangle$. To see that this is possible, consider $S = \{j \mid (v, w_{j+1}) \in E\}$ and $T = \{i \mid (w_i, u) \in E\}$. Since $S \cup T \subseteq \{1, 2, \ldots, |V(P)| - 1\}$, we have $|S \cup T| \leq |V(P)| - 1$. Again because, $|S| = d_p(v)$, $|T| = d_p(u)$, and $d_p(u) + d_p(v) \geq |V(P)|$, we must have,

$|S \cap T| = |S| + |T| - |S \cup T|$

$= d_p(u) + d_p(v) - |S \cup T|$

$\geq d_p(u) + d_p(v) - (|V(P)| - 1)$

$\geq |V(P)| - |V(P)| + 1$

$= 1.$

Hence S and T must have a common subscript so that the two crossover edges (v, w_{i+1}) and (w_i, u) exist and we get the cycle C. Hence by Lemma 3.1.1 the result follows. □

Corollary 3.1.4. *Let $G = (V, E)$ be a connected graph with n vertices and P be a longest path in G having length $k < n - 1$ and with end vertices u and v. Then we must have $d_p(u) + d_p(v) \leq k - \delta(u, v) + 2$.*

Proof. Since k < n – 1, P is not a Hamiltonian path. Hence by Theorem 3.1.3(a) we have $\delta(u, v) > 1$. Noting that if $\delta(u, v) = 2$, then $k = k - \delta(u, v) + 2$, by Theorem 3.1.3(b) and (c) we thus have $d_p(u) + d_p(v) \leq k - \delta(u, v) + 2$. □

Now we will present another sufficient condition that will ensure the existence of a Hamiltonian path in a graph. The introduction of the new parameter: "(shortest path) distance" should be noted here.

Theorem 3.1.5. *Let $G = (V, E)$ be a connected graph with n vertices such that for all pair of distinct nonadjacent pair of vertices $u, v \in V$ we have $d(u) + d(v) + \delta(u, v) \geq n + 1$. Then G has a Hamiltonian path.*

Proof. We prove it by contradiction as follows. Assume that the condition holds but there is no Hamiltonian path in G. Then let $P \equiv \langle u = u_0, u_2, \ldots, u_k = v\rangle$ be the longest path in G having length k. Surely, $k \leq n - 2$ and $\delta(u, v) \leq k$. Then by Corollary 3.1.4 we must have $d_p(u) + d_p(v) \leq k - \delta(u, v) + 2$. Now we have,

$$
\begin{aligned}
& d(u) + d(v) + \delta(u,v) \\
&= d_P(u) + d_{\bar{P}}(u) + d_P(v) + d_{\bar{P}}(v) + \delta(u,v) \\
&= \{d_P(u) + d_P(v)\} + d_{\bar{P}}(u) + d_{\bar{P}}(v) + \delta(u,v) \\
&\leq \{k - \delta(u,v) + 2\} + d_{\bar{P}}(u) + d_{\bar{P}}(v) + \delta(u,v) \\
&= k + 2 + d_{\bar{P}}(u) + d_{\bar{P}}(v) \\
&\leq n - 2 + 2 + d_{\bar{P}}(u) + d_{\bar{P}}(v) \\
&= n + d_{\bar{P}}(u) + d_{\bar{P}}(v)
\end{aligned}
$$

Since P is not a Hamiltonian path by Theorem 3.1.3(a) $\delta(u, v) > 1$ i.e. u, v are nonadjacent and hence we have $d(u) + d(v) + \delta(u, v) \geq n + 1$ according to our assumption. We thus have,

$n + d_{\bar{P}}(u) + d_{\bar{P}}(v) \geq n + 1$

$\Rightarrow d_{\bar{P}}(u) + d_{\bar{P}}(v) \geq n - n + 1$

$\Rightarrow d_{\bar{P}}(u) + d_{\bar{P}}(v) \geq 1$

The above implies that there is at least one edge of the form (x, y) such that $x \in \{u, v\}$ and $y \in V(G) - V(P)$ which means that we get a longer path in G by adding the edge (x, y) to P which is a contradiction and the result follows. □

We end this section with an illustrative example for the Theorem 3.1.5.

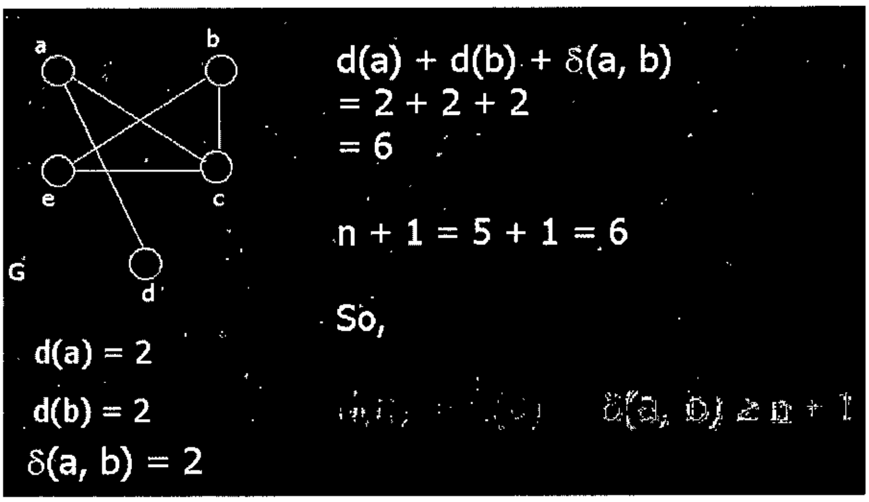


Figure 3.1: A graph satisfying the condition of Theorem 3.1.5

Example 1. Consider the graph G in Figure 3.1. There are 5 nonadjacent vertex-pairs namely (a, b), (a, e), (b, d), (c, d), and (e, d). It is easy to see that for each pair of nonadjacent vertices the hypothesis of Theorem 3.1.5 holds true (calculation for the first pair is shown in Figure 3.1). So according to our theorem, G must have a Hamiltonian path and indeed a Hamiltonian path of G is as follows: ⟨d, a, c, b, e⟩.

3.1.1 Significance of Theorem 3.1.3 & Theorem 3.1.5

Our result is interesting and a number of existing well known and very powerful theorems directly follow from our results as discussed below. Consider the Theorem 1.1.1 i.e. Dirac's condition. The proof of Dirac's Theorem very cleverly exploits the idea of extremality. The idea was if there is a non-Hamiltonian graph satisfying the hypotheses, then adding edges cannot reduce the minimum degree, so we may restrict our attention to maximal non-Hamiltonian graphs with minimum degree at least $n/2$.

By "maximal", we mean that no proper supergraph is also non-Hamiltonian, so $G + (u, v)$ is Hamiltonian whenever u, v are nonadjacent. Note that the maximality of G implies that G has a spanning path from $u = v_1$ to $v = v_n$, i.e. a Hamiltonian path. The rest of the proof tries to establish a spanning cycle through the existence of a spanning cycle [see 44 for a complete proof].

The result provided by Ore (Theorem 1.1.2) is in fact inspired from Dirac's condition. Ore observed that this argument uses $\delta(G) \geq n/2$ only to show that $d(u) + d(v) \geq n$. Therefore, we can weaken the requirement of minimum degree $n/2$ to require only that $d(u) + d(v) \geq n$ whenever u, v are nonadjacent. We also did not need that G was a maximal non-Hamiltonian graph, only that $G + (u, v)$ was Hamiltonian and thereby provided a spanning u, v-path. We here show that Ore's conditions in fact follow from our results. First we present the following lemma.

Lemma 3.1.1.1. *Let G be a simple graph with n vertices and u, v are distinct nonadjacent vertices of G with $d(u) + d(v) \geq n$. Then $\delta(u, v) = 2$.*

Proof. Let us arrange the vertices of G in a sequence such that $V(G) = \{v = w_1, w_2, \ldots, w_{|V(P)|-1}, w_{|V(P)|} = u\}$.Let $S = \{j \mid (v, w_j) \in E\}$ and $T = \{i \mid (w_i, u) \in E\}$. Since $S \cup T \subseteq \{2, \ldots, |V(P)| - 1\}$, we have $|S \cup T| \leq |V(P)| - 2$. Again because, $|S| = d_p(v)$, $|T| = d_p(u)$, and $d_p(u) + d_p(v) \geq |V(P)|$, we must have,

$$|S \cap T| = |S| + |T| - |S \cup T|$$

$$= d_p(u) + d_p(v) - |S \cup T|$$

$$\geq d_p(u) + d_p(v) - (|V(P)| - 2)$$

$$\geq |V(P)| - |V(P)| + 2$$

$$= 2.$$

Hence S and T must have common subscripts so that we have edges of the form (u, x), (x, v) which implies a u, v-path of length 2. Since u, v are nonadjacent the result follows. □

Now we are ready to prove that Ore's Theorem (Theorem 1.1.2) follows from our result.

Proof of Ore's Theorem (Theorem 1.1.2):

One direction is trivial. So we prove the other direction as follows. By Lemma 3.1.1.1 since G satisfies the sufficiency conditions, we must have $\delta(u, v) = 2$. Now since $G + (u, v)$ is Hamiltonian we must have a Hamiltonian path say P in G. So P is a longest path in G. Since P is a Hamiltonian path, we have $d_p(u) + d_p(v) = d(u) + d(v) \geq n$. Hence by Theorem 3.1.3(c) P is contained in a Hamiltonian cycle and hence G is Hamiltonian. □

Now we will consider Theorem 1.1.4, which was also due to Ore. Here again we first need to present the following lemma.

Lemma 3.1.1.2. *Let G be a simple graph and $d(u) + d(v) \geq n$ for every pair of distinct nonadjacent vertices u and v of G. Then $\delta(u, v) \leq 2$ for every pair of distinct vertices u and v of G.*

Proof. The proof is simple. First note that for every pair of distinct adjacent vertices $\delta(u, v) = 1 < 2$. Now we just need to consider every pair of distinct nonadjacent vertices. Then the result follows readily from Lemma 3.1.1.1. □

Now we are ready to show that Theorem 1.1.4 also follows from our result.

Proof of Ore's Theorem (Theorem 1.1.4):

Since we have $d(u) + d(v) \geq n$ for every pair of distinct nonadjacent vertices u and v, by Lemma 3.1.1.2 $\delta(u, v) \leq 2$ for every pair of distinct vertices u and v. And it is clear that for every pair of distinct nonadjacent vertices u and v we must have $\delta(u, v) = 2$. Now for every pair of distinct nonadjacent vertices u and v we have,

$$d(u) + d(v) \geq n > n + 1 - 2 = n + 1 - \delta(u, v).$$
$$\Rightarrow d(u) + d(v) > n + 1 - \delta(u, v)$$
$$\Rightarrow d(u) + d(v) + \delta(u, v) > n + 1$$

Thus by Theorem 3.1.5 there is a Hamiltonian path P (let) in G. Now P is a longest path in G. Let the end vertices of P be x and y. If we have $\delta(x, y) = 1$ then by Theorem 3.1.3(a), P is contained in a Hamiltonian cycle and hence G is Hamiltonian. Otherwise we must have $\delta(x, y) = 2$. And since we have $d(u) + d(v) \geq n$

$> n - 1$, by Theorem 3.1.3(b) again P is contained in a Hamiltonian cycle and hence G is Hamiltonian. □

3.2 Vertex Triples

Most of the degree related sufficient conditions in the literature consider the degree sum of vertex couples. In this section we extend this idea and present a sufficient condition based on the degree sum of vertex triplets. We also show that the theorem is the best possible.

In this section we first state and prove the following useful Lemma.

Theorem 3.2.1. Let $G = (V, E)$ be a connected graph with n vertices. If for all pairwise non-adjacent vertex-triples u, v, and w: $d(u)+d(v)+d(w) \geq \frac{1}{2}(3n-5)$ then G has a Hamiltonian path.

Proof: Let $P = \langle u_0, u_1, \ldots, u_{p-1}\rangle$ be a longest path in G. And assume for the sake of contradiction that P is not a Hamiltonian path. Now since P is a longest path but not a Hamiltonian path, by the contrapositive of Lemma 3.1.1, P cannot be contained in a cycle. And since P cannot be contained in a cycle, there cannot be any crossover edge involving u_0 and u_{p-1}. This essentially means that $d(u_0) + d(u_{p-1}) \leq p - 1$. So we must have:

$$
\begin{aligned}
& d(u_0)+d(u_{p-1})+d(w) \geq \frac{1}{2}(3n-5) \\
\Rightarrow \quad & d(w) \geq \frac{1}{2}(3n-5)-(d(u_0)+d(u_{p-1})) \\
\Rightarrow \quad & d(w) \geq \frac{1}{2}(3n-5)-(p-1) \\
\Rightarrow \quad & d(w) \geq \frac{3}{2}n-p-\frac{3}{2}
\end{aligned}
$$

Now we consider $d_P(w)$. We calculate the upper limit of $d_P(w)$ as follows. It is clear that $(w, u_0), (w, u_{p-1}) \notin E$ since otherwise P would not be a longest path in G. Again, Note that w cannot be connected to u_i and u_{i+1}, since in that case we can easily get a path $P' = \langle u_0, u_1, \ldots, u_i, w, u_{i+1}, \ldots, u_{p-1} \rangle$ which is longer than P, leading to a contradiction. So we can write that $d_p(w) \le \frac{p-2}{2} + 1 = \frac{p}{2}$.

Now we have,

$$\begin{aligned} d(w) &\ge \frac{3}{2}n - p - \frac{3}{2} \\ \Rightarrow d_p(w) + d_{\bar{p}}(w) &\ge \frac{3}{2}n - p - \frac{3}{2} \\ \Rightarrow \quad d_{\bar{p}}(w) &\ge \frac{3}{2}n - p - \frac{3}{2} - d_p(w) \\ &= \frac{3}{2}n - p - \frac{3}{2} - \frac{p}{2} \\ &= \frac{3}{2}(n - p - 1). \end{aligned}$$

This leads to a contradiction since $|V(G)\backslash(V(P) \cup \{w\})| = n - p - 1 < \frac{3}{2}(n-p-1)$, which completes the proof. □

Now that we have proved our theorem, let us consider an example, which illustrates how the condition of our theorem works.

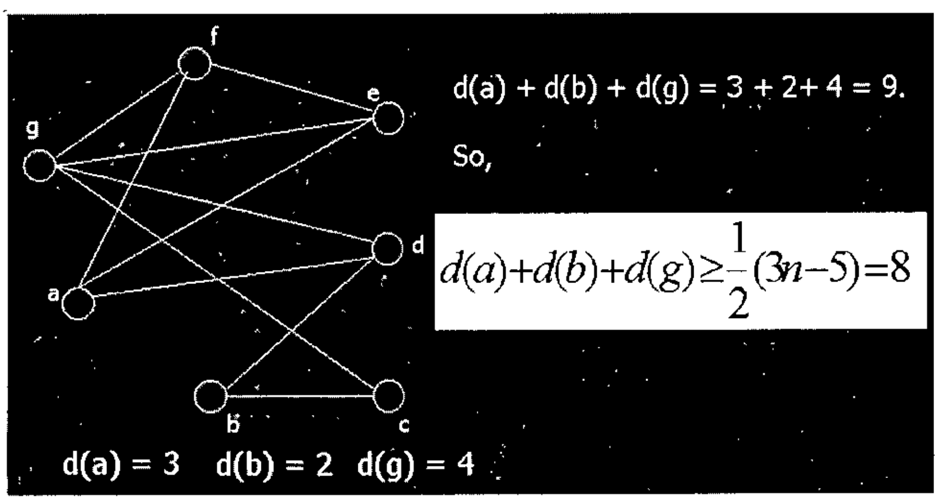


Figure 3.2: A graph satisfying the condition of Theorem 3.2.1

Example 2. Consider the graph in Figure 3.2. There are in total 3 nonadjacent vertex triples namely (a, g, b), (c, d, f), and (c, d, e). For each of these vertex triples it can be easily verified that the condition of Theorem 3.2.1 holds (calculation for the first triple is shown in the figure). So according to our theorem there exists a Hamiltonian path in G which is indeed the case since ⟨a, e, f, g, d, b, c⟩ is Hamiltonian path in G.

In the rest of this section we establish that the condition given in Theorem 3.2.1 is tight. To establish that we first disprove the following statement.

Statement 3.2.2 (To Be Disproved). Let $G = (V, E)$ be a connected graph with n vertices. If for all pairwise non-adjacent vertex-triples u, v, and w it holds that $d(u) + d(v) + d(w) \geq \frac{3}{2}(n-2)$ then G has a Hamiltonian path. □

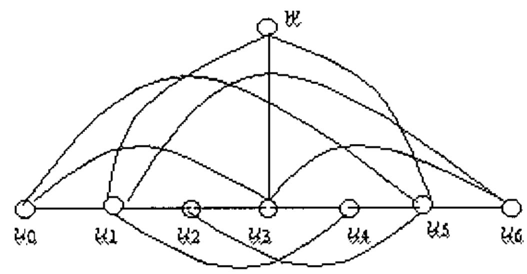


Figure 3.3: A Graph G with 8 vertices

We disprove Statement 3.2.2 by presenting a counter example as follows. Consider the graph G in Figure 3.3. It can be easily verified that for any nonadjacent vertex-triple u, v, w the condition stated in Statement 3.2.2 holds i.e. for all vertex triples u, v, and w it holds that $d(u) + d(v) + d(w) \geq \frac{3}{2}(n-2) = 9$ in graph G in Figure 3.3. However it can also be verified easily that there exists no Hamiltonian path in G, which disproves the Statement 3.2.2. Now we state the following claim.

Claim 3.2.3. The condition in Theorem 3.2.1 is tight.

Proof. The invariant in the condition in Theorem 3.2.1 is as follows: $d(u) + d(v) + d(w) \geq \frac{1}{2}(3n-5)$. Since the degree sums cannot be fractional numbers, so the next best invariant for the condition would necessarily have to be as follows:

$$d(u) + d(v) + d(w) \quad \geq \quad \frac{1}{2}(3n-5) - 1.$$

$$= \frac{1}{2}(3n-5-2)$$

$$= \frac{1}{2}(3n-7)$$

We here prove that this condition can never be achieved. Recall that the invariant in the condition in Statement 3.2.2 is as follows. $d(u) + d(v) + d(w) \geq \frac{3}{2}(n-2) = \frac{1}{2}(3n-6)$ $(> \frac{1}{2}(3n-7))$. Now since we have disproved Statement 3.2.2 our claim follows directly. □

3.3 Independence Number and Hamiltonicity

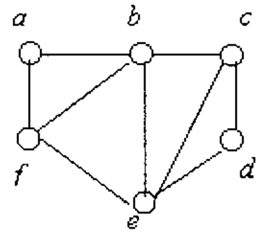
In this section we set aside the idea of degree sum and distances between vertices of a graph and consider a completely different graph parameter, namely independence number (to be defined shortly), and investigate the relation between hamiltonicity and independence number of a graph.

3.3.1 Preliminaries

In this section we introduce some notations and relevant definitions. We denote by $N_G(x)$, the set of vertices, which is adjacent to x in graph G, and its cardinality by $\deg_G(x)$. Given a graph $G = (V, E)$ and a set $S \subseteq V$, by $G[S]$ we mean the induced subgraph of the set S. $V(G)$ and $E(G)$ are used respectively to denote the vertex set and edge set of G and $\delta(G)$ denotes the minimum degree of the graph G.

Definition 3.3.1.1. *Depth-First-Maximal Path.* We use maximal path in its usual meaning i.e. the path that is not a part of a longer path. Consider the construction of a maximal path given a vertex *v, which* is said to be the *pivot vertex.* To get depth-first-maximal-path for a given pivot vertex *v*, we start expanding *v* in a direction and continue to explore in that direction in depth first manner as far as possible. Then if possible we expand *v* in another direction to get the maximal path. Consider the graph *G* in Figure 3.4 (a). Let the pivot vertex be given to be *a.* Then both P_1 and P_2 in

Figure 3.4 (b) are depth-first-maximal-paths but P_3 is not. However, if b is considered to be pivot vertex then P_3 is a depth-first-maximal-path as is P_4.



(a)

$P_1 = \langle f, a, b, e, d, c\rangle$

$P_2 = \langle a, b, e, f\rangle$

$P_3 = \langle b, a, f, e, d, c\rangle$

$P_4 = \langle b, e, f, a\rangle$

(b)

Figure 3.4: (a) A graph G with 6 vertices and 9 edges (b) Enumerations of some maximal paths of G

Definition 3.3.1.2. *Expandable nodes.* Again consider a maximal path P_m in a connected graph $G = (V, E)$. Any node $v \in V(P_m)$ is said to be an expandable node if there exists a $w \in V(G) - V(P_m)$ such that $(v, w) \in E(G)$. Clearly if $V(P_m)$ is a proper subset of $V(G)$ *i.e.* $V(P_m) \subset V(G)$, then there must be at least one expandable node in $V(P_m)$ since G is connected. Also note that since P_m is a maximal path so the two end vertices of P_m are not expandable because if one of them (or both) were expandable then P_m would be part of a longer path, a contradiction. If V' is a vertex set such that $V' \subset V(G)$ of a connected graph $G = (V, E)$ then we can generalize the notion of an expandable node by defining them as a vertex $v \in V'$ in an arbitrary subgraph $G' = (V', E')$ of $G = (V, E)$ such that $E' \subset E$ and there is at least one edge $(v, w) \in E(G)$ such that $v \in V'$ and $w \in V - V'$. To specify more clearly we sometimes refer to v as G'-expandable node.

Definition 3.3.1.3. *Lone-Degree-Expandable.* Consider a graph G and let $G' \subseteq G$. Now a G'-expandable node v is said to be lone-degree-expandable if for every possible depth-first-maximal path P_m constructed from the graph G – G′, considering v to be the pivot vertex we have $\deg_{P_m}(v) = 1$ i.e. $\deg_{G'}(v) = \deg_G(v) + 1$ where $G'' =$

$G' + P_m$. To specify more clearly we sometimes refer to v as *Lone-Degree-G'-* expandable node.

Definition 3.3.1.4. *Independence Set and Independence Number.* A set $S \subseteq V(G)$ of a graph G is said to an independent set if for every pair of vertices $u, v \in S$, $(u, v) \notin E(G)$. Independence number of G, denoted by $\alpha(G)$, is the cardinality of a maximum Independent set of G.

Definition 3.3.1.5. *Cut Vertex.* A vertex $v \in V$ of a graph $G = (V, E)$ is a cut vertex of G if the deletion of v increases the number of connected components.

Now that we have got the definitions in our hand, we are going to present the following two theorems in the next couple of sections.

Theorem 3.3.1.6. Let G = (V, E) be a connected graph such that α(G) = 2. Then G has a Hamiltonian path.

Theorem 3.3.1.7. Let G = (V, E) be a 2-connected graph such that α(G) = 2. Then G is Hamiltonian.

3.3.2. Independence Number and Hamiltonian Path

In this section we prove Theorem 3.3.1.6. However, we first present and prove following very interesting Lemma.

Lemma 3.3.2.1. *Let G be a connected graph such that* $\alpha(G) = 2$ *and* $P = \langle u_1, u_2, \ldots, u_k \rangle$ *be a maximal path of G. Then the followings must hold true:*

a. If P is not a Hamiltonian path then P is contained in a cycle.

b. If P is contained in a cycle then either P is Hamiltonian or every expandable vertex v in V(P) is Lone-Degree-P-Expandable.

Proof.

Part a.

It is clear that, since P is a maximal path, $N_G(u_1) \subset V(P)$ and $N_G(u_k) \subset V(P)$. This essentially means that both u_1 and u_k are not P-expandable. So P can be contained in a cycle C if and only if $(u_1, u_k) \in E(G)$. Now assume for the sake of contradiction that P

is not contained in a cycle. Then, since P is not Hamiltonian and G is connected there is a P-Expandable node v (let). Now we construct a depth-first-maximal-path P' in the graph $G[(V(G) - V(P)) \cup \{v\}]$, taking v as the pivot vertex. Let in P' the end vertices are x and y. It is clear that $P + P'$ is connected and acyclic. Now in $P + P'$ we have at least 3 end vertices. Note that if $v \notin \{x, y\}$ then we have 4 end vertices. Let $\Pi = \{w \mid w$ is an end vertex in $P + P'$ and $w \neq v\}$. Now we will show that Π is an independent set in G such that $|\Pi| > 2$. So without loss of generality assume that $v = x$ and hence $\Pi = \{u_1, u_k, y\}$. now according to assumption $(u_1, u_k) \notin E(G)$. Also both (u_1, y) and $(u_k, y) \notin E(G)$, since otherwise P would be part of a longer path resulting in a contradiction. Then Π must be an independent set of G implying that $\alpha(G) \geq 3$ which is a contradiction. □

Part b.

Since P is contained in a cycle C (let) we must have $(u_1, u_k) \in E(G)$ as is explained in the beginning of the proof of part a. So we must have $V(P) = V(C)$. Now if P is Hamiltonian then there can be no expandable vertex. So it suffices to show that if P is not Hamiltonian, then every expandable vertex in $V(P)$ is Lone-Degree-P-Expandable. We show it as follows. Define $\zeta = \{v \mid v$ is P-expandable$\}$. Let $v \in \zeta$ such that v is not Lone-Degree-P-Expandable. Then we can construct a depth-first-maximal-path P' in the graph $G[(V(G) - V(P)) \cup \{v\}]$, taking v as the pivot vertex such that $v \notin \{x, y\}$ where in P' the end vertices are x and y. Let $\Pi = \{w \mid w$ is an end vertex in $P + P'\}$. Then $\Pi = \{u_1, u_k, x, y\}$. Let $\Pi' = \{u_k, x, y\}$. From the construction of P' it is clear $(x, y) \notin E(G)$. Also since P is a maximal path, we must have $(u_k, x) \notin E(G)$ and $(u_k, y) \notin E(G)$. Then Π' form an independent set in G and we have $\alpha(G) \geq 3$, which is a contradiction. □

Now we are ready to prove Theorem 3.3.1.6.

Proof of Theorem 3.3.1.6:

With Lemma 3.3.2.1 in our hand we can give an easy constructive proof of Theorem 1. We first construct a maximal path $P = \langle u_1, u_2, \ldots, u_k \rangle$. If P is a Hamiltonian path then we are done. So assume that P is not Hamiltonian. Then P is contained in a cycle

C by Lemma 3.3.2.1 (a) and it is clear that $C = \langle u_1, u_2, \ldots, u_k, u_1\rangle$. Now let $\zeta = \{v \mid v$ is P-expandable$\}$. Since G is connected and according to assumption P is not Hamiltonian we must have $\zeta \neq \phi$. By Lemma 3.3.2.1 (b) every $v \in \zeta$ is Lone-Degree-P-Expandable. Now we construct a depth-first-maximal-path P' in the graph $G[(V(G) - V(P)) \cup \{v\}]$, taking v as the pivot vertex. Let in P' the end vertices are x and y. Since v is Lone-degree-P-expandable, we must have $\deg_P(v) = 1$ i.e. $v \in \{x, y\}$. Let $v = y$. It is clear that $P + P'$ is connected and acyclic and in $P + P'$ we have 3 end vertices namely u_0, u_k and x, since $\deg_{(P+P')}(v) = 3$. Now we add the edge (u_0, u_k) and delete an edge (u_i, u_{i+1}) (or equivalently (u_i, u_{i-1})) such that $u_i = v$. It is clear that the resultant graph G_n is a path with end vertices u_{i+1} (or u_{i-1}) and x. Now we claim that $V(P + P') = V(G_n) = V(\text{G})$. The claim is proved as follows.

Assume for the sake of contradiction that $V(P + P') = V(G_n) \subset V(G)$. Then $\zeta_n \neq \phi$ where $\zeta_n = \{v \mid v$ is G_n-expandable$\}$. Suppose $w \in \zeta_n$. Then we again can construct a depth-first-maximal-path P_n in the graph $G[(V(G) - V(G_n)) \cup \{w\}]$, taking w as the pivot vertex. Now considering one end vertex of P and at least one end vertex each from P' and P_n we again can show the existence of an independent set of G with cardinality at least 3 contradicting the assumption $\alpha(G) = 2$ and this completes the proof of our claim. Hence G_n is in effect a Hamiltonian path and the result follows.□

Example 3. Consider the graph G in Figure 3.5. As indicated in the figure, the independence number of the graph is 2. According to our theorem G must have a Hamiltonian path which is evident from the following Hamiltonian path: ⟨b, a, c, d, e, f⟩.

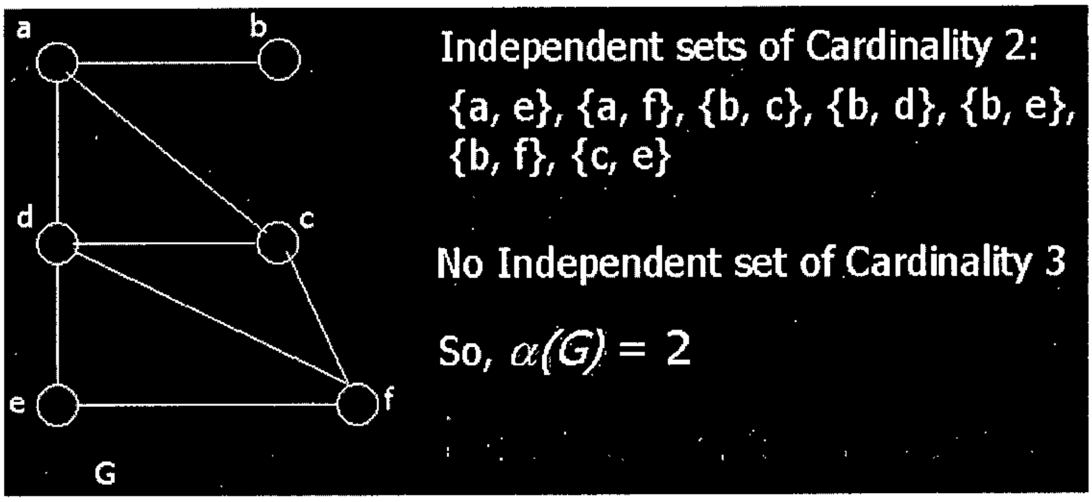


Figure 3.5: A graph satisfying the condition of Theorem 3.3.1.6

3.3.3. Hamiltonian Cycle

In Theorem 3.3.1.6 we presented a sufficient condition for the existence of a Hamiltonian path in a connected graph. A natural extension to our research should be to try to show that the graphs satisfying our condition are Hamiltonian. Unfortunately, that is not necessarily the case. We below present a simple graph G with 6 vertices (Figure 3.6) satisfying our condition but having no Hamiltonian cycle.

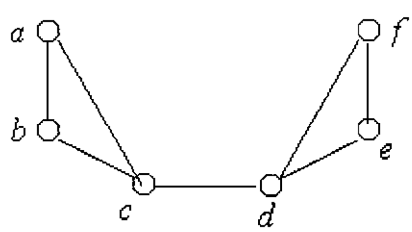


Figure 3.6: A graph satisfying the condition in Theorem 3.3.1.6, but having no Hamiltonian cycle

It is clear that $\alpha(G) = 2$, an example of maximum independent set being $\{a, f\}$. The existence of Hamiltonian path is also clear (e.g. $\langle a, b, c, d, e, f \rangle$). However, it can be easily verified that there is no Hamiltonian cycle in G. And in fact there can't be any, since G is not 2-connected and since 2-connectivity is necessary for the existence of a Hamiltonian cycle. To see that G is not 2 connected just realize that both c and d are cut vertices in G. Now the natural question would be whether a graph G satisfying the conditions in Theorem 3.3.1.6 is Hamiltonian if we add the condition that G is also 2-connected. Indeed that is the case as we have already stated in Section 1 in the form Theorem 3.3.1.7. However, before proving Theorem 3.3.1.7 we first state the following two theorems characterizing 2-connected graphs (see [44]).

Theorem 3.3.3.1. *A graph G having at least 3 vertices is 2-connected if and only if each pair $u, v \in V(G)$ is connected by a pair of internally-disjoint u, v-paths in G.*

Theorem 3.3.3.2. *If $|V(G)| \geq 3$, then the following conditions are equivalent and characterize 2-connected graphs.*

a. *G is connected and has no cut-vertex.*

b. *For all $x, y \in V(G)$, there are internally-disjoint x, y-paths.*

c. *For all $x, y \in V(G)$, there is a cycle through x and y.*

d. *$\delta(G) \geq 2$, and every pair of edges in G lies on a common cycle.*

Now we prove Theorem 3.3.1.7 as follows.

Proof of Theorem 3.3.1.7:

By Theorem 3.3.1.6, G has a Hamiltonian path let $P = \langle u_1, u_2, \ldots, u_n \rangle$. Now if $(u_1, u_n) \in E(G)$ we are done. So assume otherwise i.e. $(u_1, u_n) \notin E(G)$. Since G is 2-connected, for all $v \in V(G)$, $\deg_G(v) \geq 2$. Consider the set $\{u_1, u_3, u_n\}$. Since $\alpha(G) = 2$, we must have either $(u_1, u_3) \in E(G)$ or $(u_3, u_n) \in E(G)$ (or both). Again consider the set $\{u_1, u_4, u_n\}$ and apply the same argument there. We can continue this argument upto the set $\{u_1, u_{n-2}, u_n\}$. Now note carefully that, in above arguments if at any point we get cross over edges of the form (u_1, u_i) and (u_{i-1}, u_n) then we are done since we then get a Hamiltonian cycle $\langle u_1, u_i, u_{i+1}, \ldots, u_n, u_{i-1}, u_{i-2}, \ldots, u_1 \rangle$. So assume otherwise, i.e. we do not get any such cross over edges. This essentially means that if $u_i \in N_G(u_1)$ and $u_j \in N_G(u_n)$ then $i \leq j$. Note that since we have assumed that there is no cross over edges, $N_G(u_1)$ is of the form- $\{u_2, u_3, u_4, \ldots, u_k\}$ and $N_G(u_n)$ is of the form- $\{u_l, u_{l+1}, \ldots, u_{n-1}\}$ where $l = k$ or $l = k + 1$. Now there is an edge (x, y) such that $x \in N_G(u_1) - \{u_k\}$ and $y \in N_G(u_n) - \{u_l\}$ because otherwise u_k and u_l would be cut vertices in G (Note that it may be the case that $u_k = u_l$). Assume that $x = u_{k-i}$ and $y = u_{l+j}$, $i, j > 0$. It is clear that $(u_1, u_{k-i+1}) \in E(G)$ and $(u_n, u_{l+j-1}) \in E(G)$. Now since $(u_1, u_{k-i+1}) \in E(G)$, we have a Hamiltonian path $P_1 = \langle u_{k-i}, u_{k-i-1}, \ldots, u_1, u_{k-i+1}, u_{k-i+2}, \ldots, u_{l+j-1}, u_{l+j}, \ldots, u_n \rangle$. Note carefully that $(x = u_{k-i}, y = u_{l+j}) \in E(G)$ and $(u_n, u_{l+j-1}) \in E(G)$, which are, cross over edges for P_1. Therefore we have a Hamiltonian cycle $C_1 = \langle u_{k-i}, u_{k-i-1}, \ldots, u_1, u_{k-i+1}, u_{k-i+2}, \ldots, u_{l+j-1}, u_n, u_{n-1}, \ldots, u_{l+j}, u_{k-i} \rangle$. Hence the result follows.□

Example 4. Just making the graph G in Figure 3.6 2-connected ensures Hamiltonicity as is evident from the following figure. A Hamiltonian cycle is ⟨b, a, c, d, e, f, b⟩.

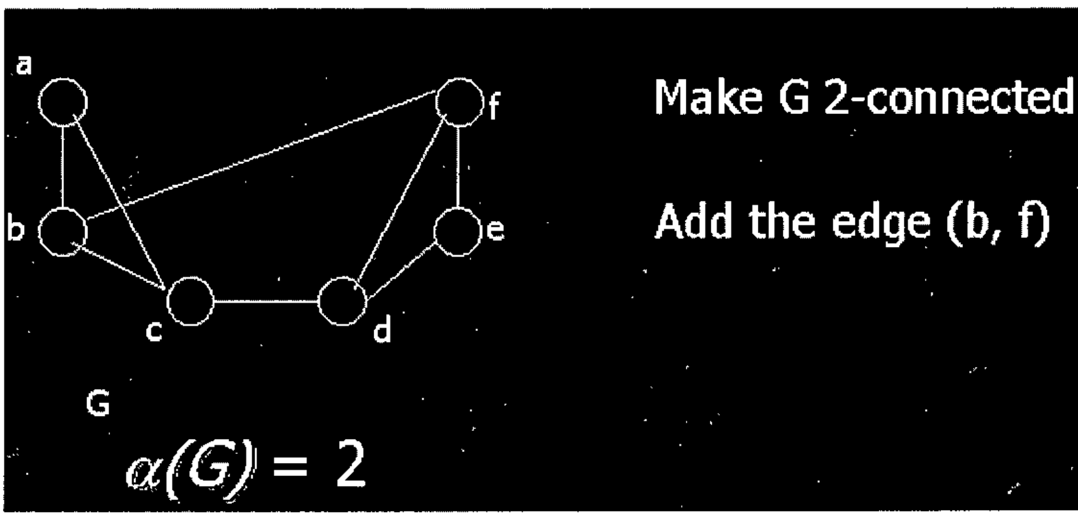


Figure 3.7: A Graph satisfying the condition of Theorem 3.3.1.7

3.4 Studies and Applications

Theory of Hamiltonicity has always been important focus of research and hence tremendous amount of research has been done on and related to this topic. In this subsection we just point out some areas where our results and theory of Hamiltonicity in general, may be applicable.

The architecture of an interconnection network is usually represented by a graph. There are a lot of mutually conflicting requirements in designing the topology of interconnection networks. It is almost impossible to design a network, which is optimum for all conditions. One has to design a suitable network depending on the requirements of their properties. The Hamiltonian property is one of the major requirements in designing the topology of networks. Fault tolerance is also desirable in massive parallel systems that have relatively high probability of failure. Hence a major research topic is the Hamiltonicity and fault-tolerance property of graphs [6, 20, 21, 22].

Another interesting idea is the idea of *Hamiltonian laceability* for Hamiltonian bipartite graphs, first introduced by Simmons [40]. Hsieh et al. [18] extended this concept into *strongly Hamiltonian laceability*. Hamiltonian laceability deals with embedding a Hamiltonian path in a given graph and is an important topic in interconnected networks. Recent studies have proposed several operations performing on Hamiltonian laceable graphs to yield several attractive properties [16, 28, 29].

Chapter 4

SPANNING TREES

4.0 Introduction

In this section we discuss spanning trees and related problems and results. A spanning subgraph of a graph $G = (V, E)$ is a subgraph with vertex set V. A spanning tree is a spanning subgraph that is a tree. Spanning trees have been found to be structures of paramount importance also in practical problems. As a result spanning trees of a connected graph have been the focus for extensive attention in graph theoretic research. Spanning trees with various constraints and restricted conditions seem to pose various interesting problems.

In this chapter we introduce some new problems (with relevant new results) where we impose various constraints and restrictions on parameters of spanning trees. We also investigate the relationship of the independence number and a special spanning tree namely degree bounded spanning tree. Further, we introduce a new notion "set version". The complexities of the set versions of various problems are discussed and we show that surprisingly and remarkably "set versions" of some NP-Complete problems are solvable in polynomial times although this may not necessarily be the case all the time.

4.1 Independence Number and Degree-Bounded-Spanning Tree

In this section we investigate the relation between independence number and degree bounded spanning tree. Degree Bounded Spanning Tree problem is formally defined as follows.

Problem 4.1.1. Degree Bounded Spanning Tree Problem

Instance: Graph $G = (V, E)$, positive integer $K \leq |V|$.

Question: Is there a spanning tree for G in which no vertex has degree larger than K?

Example 5. Consider the graph G in Figure 4.1. Suppose the value of K in the input instance is given to be 3. Then the spanning tree in the figure is a Degree Bounded Spanning Tree. However if K is given to be 2 then the spanning tree is not a desired tree for the given input instance.

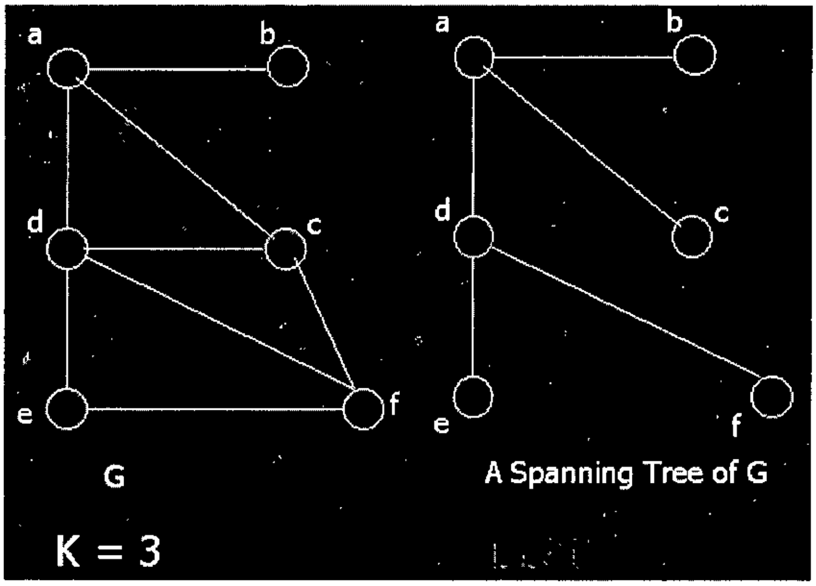
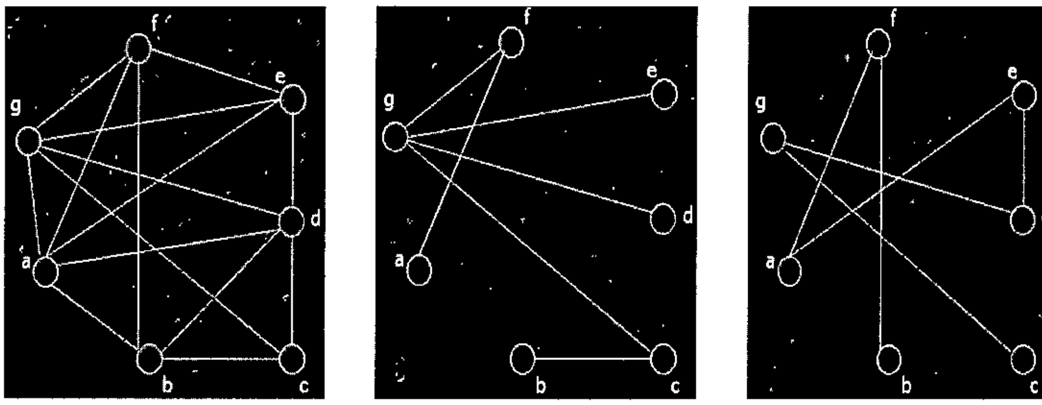


Figure 4.1: Degree Bounded Spanning Tree

Problem 4.1.1 is proved to be NP-Complete [13] and it remains NP-Complete for any fixed $K \geq 2$. NP-Completeness of Problem 4.1.1 suggests that there exists no good characterization for a graph having a degree-bounded-spanning-tree as defined in the problem. A characterization is said to be good if it can be tested easily; polynomially, to be specific. We define the term degree-d-bounded-spanning-tree to be a spanning tree in which no vertex has degree larger than d. In this paper we relate the existence of a degree-bounded-spanning-tree in a graph G to the independence number of G. In this section we present a theorem stating that a graph with independence number d must have a degree-d-bounded spanning tree. In the sequel, we outline a polynomial time algorithm to construct a degree-d-bounded spanning tree from a graph having independence number d.

Example 6. Consider the graph in Figure 4.2(a). The spanning tree T_1 in Figure 4.2(b) is a degree-4-bounded spanning tree since there are no vertices in T_1 having degree larger than 4. It should be clear that T_1 is not a degree-3-bounded spanning tree.

However, T_2 (Figure 4.2(c)), which is a degree-3-bounded spanning tree, can also be called a degree-2-bounded spanning tree. Similarly, T_1 can also be called a degree-5-bounded spanning tree, degree-6-bounded spanning tree and so on.



(a) An arbitrary Graph G (b) A spanning tree T_1 of G (c) A Spanning Tree T_2 of G

Figure 4.2: Degree-d-bounded Spanning Tree

Given a graph $G = (V, E)$ and a set $S \subseteq V$, by $G[S]$ we mean the induced subgraph of the set S. $V(G)$ and $E(G)$ are used respectively to denote the vertex set and edge set of G. By $\Pi(G)$ we denote the set of leaf nodes of the graph G. A set $S \subseteq V(G)$ of a graph G is said to an independent set if for every pair of vertices $u, v \in S$, $(u, v) \notin E(G)$. Independence number of G, denoted by $\alpha(G)$, is the cardinality of a maximum Independent set of G. An x, y-path is a path having x and y as the two end vertices of the path. A set of leaves L in a tree T is said to be *initiated* by a nonleaf node v if and only if for every pair $x, y \in L$, v lies in the x, y-path in T.

The theorem presented below establishes a relationship between independence number and degree bounded spanning tree.

Theorem 4.1.2. *Let $G = (V, E)$ be a connected graph such that $\alpha(G) = d$ where $d \geq 2$. Then G has a degree-d-bounded spanning tree.*

Before proving Theorem 4.1.2, we first present following important but simple observations.

Observation 4.1.3. *Let T be a tree. If there is a* $v \in V(T)$ *such that* $deg_T(v) = k$, *where* $k \geq 2$, *then there is a set L of leaves in T, such that* $|L| = k$ *and L is initiated by v.* □

From Observation 4.1.3 we easily get the following observation.

Observation 4.1.4. *Let T be a tree. If there is a* $v \in V(T)$ *such that* $deg_T(v) = k$, *then* $|\Pi(T)| \geq k$.□

Now we are ready to prove Theorem 4.1.2. The proof employs construction and induction simultaneously.

Proof of Theorem 4.1.2:

We first construct an arbitrary spanning tree T_0. Now if for all $v \in V(T_0) = V(G)$, $\deg_{T_0}(v) \leq d$ then we are done. So assume otherwise. Let D be the set of vertices in a spanning tree having degree greater than d. Then, $D(T_0) = \{ x \mid \deg_{T_0}(x) > d \}$. Now we show that there exists a spanning tree T such that $D(T) = \phi$. Our strategy is to show it by induction on the number of vertices belonging to D in a spanning tree . We start with a vertex $w \in D(T_0)$. Let in T_0, $\deg_{T_0}(w) = k$, where $k > d$. By Observation 4.1.3, there is a set L of leaves of T_0 such that L is initiated by w and $|L| = k$. Since $\alpha(G) = d < k$, there is a pair $z, r \in L$ such that $(z, r) \in E(G)$, because, otherwise L would be an independent set of G with cardinality $k > d$ leading to a contradiction. Since $z, r \in L$ and L is initiated by w, in $T_0 + (z, r)$ there is a cycle C such that $w \in V(C)$. Now we delete an edge (w, p) to break the cycle so as to get a spanning tree T_1. Note that $\deg_{T_1}(w) = \deg_{T_0}(w) - 1$. It is clear that we can apply above procedure until we get a spanning tree T_{k-d} such that $\deg_{T_{k-d}}(w) = d$ after a finite iteration, since $\alpha(G) = d$. So we get a spanning tree T_{k-d} such that $|D(T_{k-d})|$ is one less then $|D(T_0)|$ and this completes our proof.□

4.1.1 An Efficient Algorithm

In this section we outline a simple algorithm for constructing a degree-d-bounded spanning tree from an input graph G such that $\alpha(G) = d$. We also present a simple worst case analysis of the algorithm.

Algorithm 4.1.1.1.

//Input: A connected graph $G = (V, E)$ with $\alpha(G) = d$.

//Output: A degree-d-bounded-spanning tree T.

Begin

1. Construct and arbitrary spanning tree T of G.
2. *if* for all $v \in V$, $\deg_T(v) \leq d$, *then*
3.   *return* T
4. *else*
5.   Form the set D for T.
6.   *for* each $x \in D$ *do*
7.   Form the set L
8.   *while* $\deg_T(x) > d$ *do*
9.   Find a pair $z, r \in L$ such that $(z, r) \in E(G)$
10.   (Let C is the cycle created in $T + (z, r)$ and $(x, p) \in E(C)$)
11. $T = T + (z, r) - (x, p)$
12.   *end while*
13.   *end for*
14. *end if*
15. *return* T

End

Analysis: Let $|V| = n$ and $|E| = m$. Constructing an arbitrary spanning tree takes $O(m)$ computational effort. Forming the set D at most takes $O(n)$. To find out how many operations are actually done in the worst case in the for loop of line #6 to line #13, we just need to realize that there can be $O(n^2)$ pairs of z, r to check in line #9 in the worst case. So the overall running time is $O(n^2)$. Thus we get the following theorem.

Theorem 4.1.1.2. *Given a connected graph $G = (V, E)$ with $\alpha(G) = d$, we can find a degree-d-bounded spanning tree in $O(n^2)$ computational effort, where $n = |V(G)|$.* □

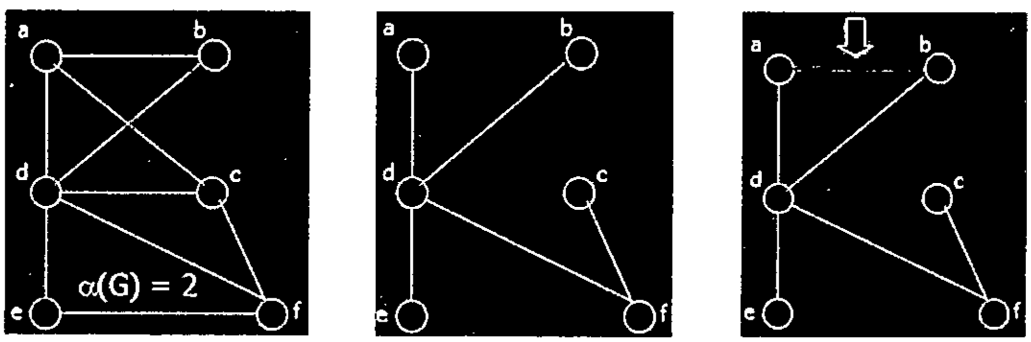
Example 7. Figure 4.3 presents an illustrative example of how Algorithm 4.1.1.1 works on an arbitrary input graph with independence number 2.

4.2 New Problems and Complexities

In this section we introduce some new interesting spanning tree problems and relevant results.

Problem 4.2.1. Minimum Leaf Spanning Tree.

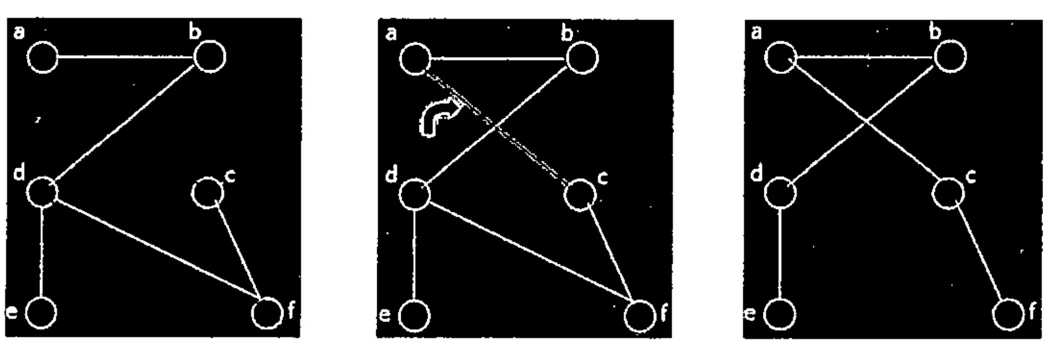
Given a connected graph $G \equiv (V, E)$ and a positive integer $K < |V|$, we are asked the question whether there is a spanning tree T of G such that K or less vertices have degree 1.



A Graph with Independence number 2

An arbitrary spanning tree. D = {d} and L = {a, b, e, c}

Add the edge (a, b) to create a cycle involving d



Break the cycle by deleting (a, d). Now degree of d is one lesser. Still D = {d}. L = {a, e, c}

Add the edge (a, c)

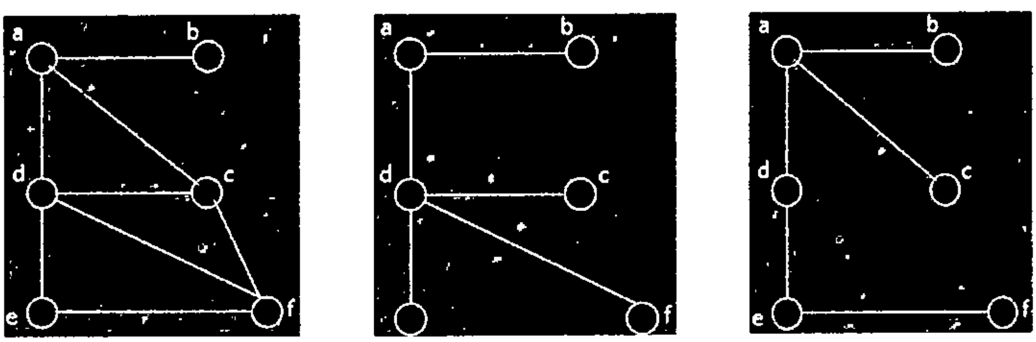
Delete (d, f). The resulting spanning tree is our desired one

Figure 4.3: Illustration of how Algorithm 4.1.1.1 works

Example 8. Consider the graph G in Figure 4.4(a) and the two spanning trees T_1 and T_2 in Figure 4.4(b) and Figure 4.4(c), respectively. Also suppose that, in the input instance, the given value for K is 3. Since in T_1, number of leaves is 4, it is not a desired Minimum Leaf Spanning Tree. The spanning tree T_2, on the other hand is a desired spanning tree for the given input instance since the number of leaves in T_2 is 3 $\leq$ K. Note carefully that T_1 would be a desire spanning tree for the cases when K $\geq$ 4.

Theorem 4.2.2. *Minimum Leaf Spanning Tree Problem is NP-Complete.*

Proof. This can be easily proved by restricting the value of K to 2, since then we in effect, have to find out a Hamiltonian path in G, which is an NP-Complete problem for general graphs [13]. □

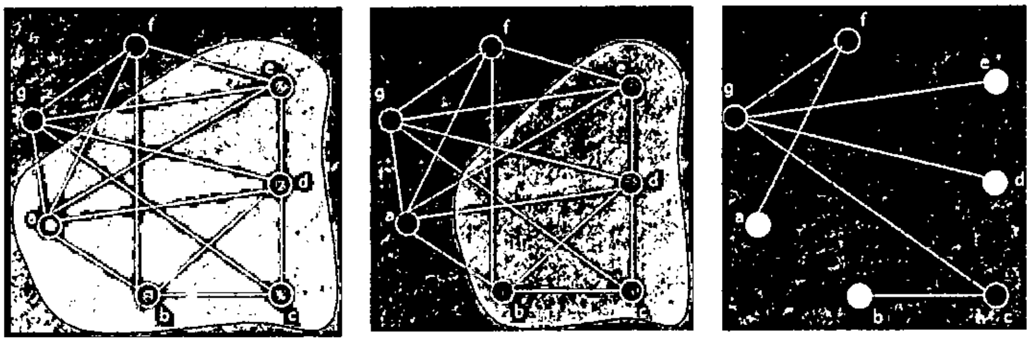


(a) An arbitrary Graph G (b) A spanning tree T_1 of G (c) A Spanning Tree T_2 of G

Figure 4.4: Minimum Leaf Spanning Tree

Problem 4.2.3. Restricted-Leaf-in-Subgraph Spanning Tree Problem.

Given $G \equiv (V, E)$ be a connected graph, X a vertex subset of G and a positive integer $K < |X|$, we are asked the question whether there is a spanning tree T_G such that number of leaves in T_G belonging to X is less than or equal to K.



(a) A graph G. K = 3. X = {a, e, b, d, c} (b) A graph G. K = 3. X = {e, b, d, c} (c) A spanning tree of G

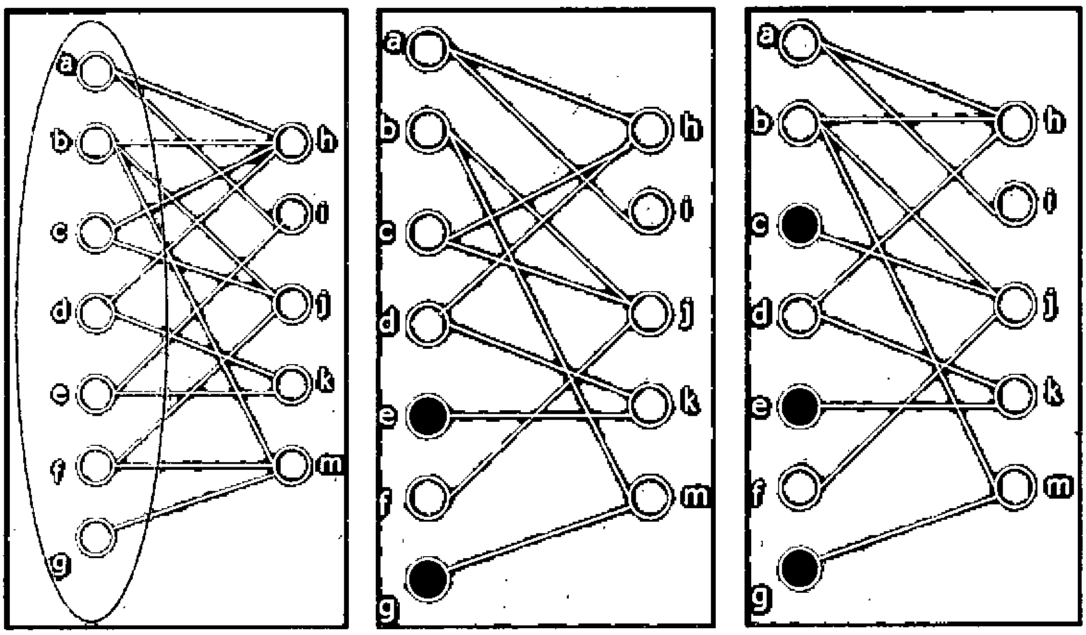
Figure 4.5: Restricted Leaf in Sub-graph Spanning Tree

Example 9. Consider the graph G in the figure 4.5(a) and 4.5(b). Both the graphs are same but with different input parameters. Although in both cases we are assuming K to be 3 the subset X is different for the two cases. In the former case, X = {a, e, b, d, c} and in the latter X = {e, b, d, c}. The tree T in Figure 8(c) is a spanning tree of G. Now, for the former input instance, T is not a desired RLSST since number of leaves belonging to X in T is not restricted according to the input i.e. it is not less than or equal to K (= 3). However for the latter input instance T is a desired spanning tree.

Theorem 4.2.4. *Restricted-Leaf-in-Subgraph Spanning Tree (RLSST) Problem is NP-Complete.*

Proof. We prove this theorem by restriction. It is easy to see that if we assume $X = V$ then RLSST problem reduces to Minimum Leaf Spanning Tree Problem (Problem 4.2.1). Hence the result follows directly from Theorem 4.2.2. □

We now consider a variant of maximum leaf spanning tree problem (Problem 1.1.2.5.) for bipartite graphs.



(a) A Bipartite graph G with partite sets X and Y

(b) A spanning tree T_1 of G

(c) A Spanning Tree T_2 of G

Figure 4.6: Variant of Maximum Leaf Spanning Tree for Bipartite Graphs

Problem 4.2.5. Variant of Maximum Leaf Spanning Tree for Bipartite Graphs

Let G be a connected bipartite graph with partite sets X and Y. Given a positive number $K \leq |X|$, we are asked the question whether there is a spanning tree T_G in G such that number of leaves in T_G belonging to X is greater than or equal to K.

Example 10. Consider the bipartite graph G in Figure 4.6(a) with partite sets X (= {a, b, c, d, e, f, g}) and Y. Also assume that the value of K in the input instance is given to be 3. Now in Figure 4.6(b) and 4.6(c) there are two spanning trees T_1 and T_2 of G. However, since in T_1, the number of leaves belonging to X is only 2 (namely e and g), it is not a desired spanning tree. On the other hand T_2 is a desired spanning tree of G for the given value of K. In the remaining of the section we investigate the necessary and sufficient conditions for the existence of such a spanning tree in a bipartite graph as defined in Problem 4.2.5. First we present the following theorem.

Theorem 4.2.6. Let G be a connected bipartite graph with partite sets X and Y and suppose K is a positive number such that $K \leq |X|$. Then there is a spanning tree T in G such that number of leaves in T belonging to X is greater than or equal to K if and only if there is a set $S \subseteq X$ such that $|X \setminus S| \geq K$ and $\langle S \cup Y \rangle$ is connected.

Proof. The proof of the theorem is simple. We first consider the if part. Suppose in G there is a set S such that $|X \setminus S| \geq K$ and $\langle S \cup Y \rangle$ is connected. We can easily find a spanning tree T' for the graph $\langle S \cup Y \rangle$. Now suppose in T' number of leaves belonging to S is K'. It is clear that $0 \leq K' \leq |S|$. Now since G is a connected bipartite graph, for each $x \in X \setminus S$ we can add an edge (x, y) such that $y \in Y$ to get a spanning tree T'' of G. This would mean that for all $x \in X \setminus S$, $\deg_{T''}(x) = 1$. Now since $|X \setminus S| \geq K$ so, T'' would necessarily be our desired spanning tree T.

Conversely, suppose G has a spanning tree T such that number of leaves in T belonging to X is greater than equal to K. Now let L be the set of leaves of T belonging to X. Since G is bipartite so in T, $N_G(L) \subseteq Y$. Hence if we delete the set L from T_G we still get a tree. This means that in G, $\langle (X \setminus L) \cup Y \rangle$ is connected. Now since according to the assumption $|L| \geq K$ so, in effect we have established the existence of the set $S(\equiv X \setminus L)$ in G as defined in the conditions. Hence the result follows. □

Now we present a more stringent condition for the existence of desired spanning tree in a bipartite graph G (as defined in Problem 4.2.5). The conditions are presented in the form of following theorem.

Theorem 4.2.7. Let G be a connected bipartite graph with partite sets X and Y and suppose K is a positive number such that $K \leq |X|$. Then there is a spanning tree T in G such that number of leaves in T belonging to X is greater than or equal to K if and only if there is a set $S \subseteq X$ such that all of the followings hold true:

a. $|X \setminus S| \geq K$

b. $\langle S \cup Y \rangle$ is connected and

c. For any subset $S' \subseteq S$, $|N_G(S')| \geq |S'| + 1$.

Proof. We first prove the only if part. Suppose there is a spanning tree T such that the number of leaves in T belonging to X is greater than or equal to K. Let $L = \{x \mid x \in X$ and $\deg_T(x) = 1\}$. So $|L| \geq K$. Now define $S = X \setminus L$. Now it is clear that $|X \setminus S| \geq K$. To show that $\langle S \cup Y \rangle$ is connected in G, we just need to realize that since G is bipartite so in T, $N_G(L) \subseteq Y$. Finally, since $S = X \setminus L$, we must have for all $x \in S$, $\deg_T(x) \geq 2$, because otherwise either x would be a leaf or x would have no edge implying that G is not connected, both of which are contradictions. We thus have for any set $S' \subseteq S$,

$$
\begin{aligned}
& |S'| + 1 \\
\leq \quad & \Sigma_{x \in S'}(\deg_T(x)\text{-}1) + 1 \\
\leq \quad & |N_T(S)| \\
\leq \quad & |N_G(S)|
\end{aligned}
$$

Now we need to show the if part. Assume that the sufficient conditions are satisfied. Let $G' \equiv \langle S \cup Y \rangle$. G' is bipartite with partite sets S and Y. Since G is bipartite, the assumptions about S hold in G' as well. So we must have for any subset $S' \subseteq S$ in G',

$$
\begin{aligned}
|N_G(S')| \quad & \geq \quad |S'| + 1 \\
& > \quad |S'|.
\end{aligned}
$$

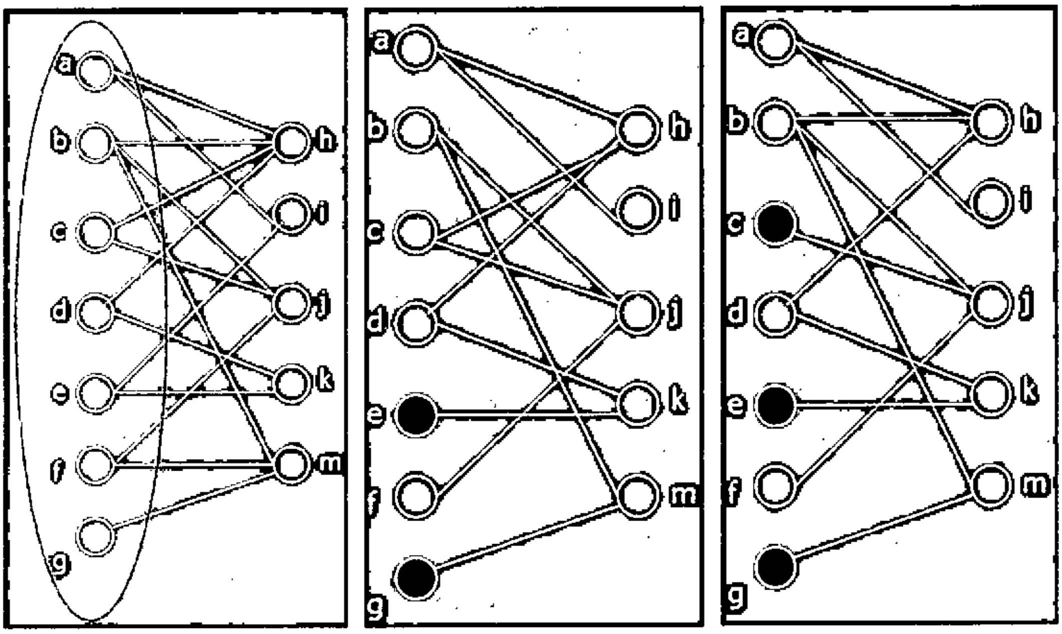
Hence there is a matching M from S to Y by Hall's theorem [15]. Now $F = (S \cup Y, M)$ is a forest where for all $x \in S$, $\deg_F(x) = 1$.Now we show that there exists a forest $\bar{F}$ such that $\deg_{\bar{F}}(x) = 1$ for all $x \in S$ as follows. We show it by an induction on the number of vertices with degree 1 in a forest. Let F' be the forest containing F such that $1 \leq \deg_{F'}(x) \leq 2$ for all $x \in S$ and let S_1 be the set of all vertices such that $\deg_{F'}(x) = 1$ for all $x \in S_1$. According to assumption we must have $N_G(S_1) > |S_1|$. Hence there must be an edge e joining S_1 and $Y \setminus N_{F'}(S_1)$. So in the forest $F' \cup e$, the number of vertices with degree 1 is fewer than $|S_1|$. Now we can obtain a spanning tree T' of G' when we add some edges to $\bar{F}$ between the connected components of $\bar{F}$. Finally since G is connected and bipartite, for each $x \in X \setminus S$, we can easily add an edge (x, y) of G to T' where $y \in Y$, to get the desired spanning tree T. □

Remark: It is clear that with respect to the Problem 4.2.5, both the conditions stated above (Theorem 4.2.6 and Theorem 4.2.7) are equivalent to each other, Theorem 4.2.6 being much simpler. The justification of presenting both the theorems lies partially in the fact that unfortunately we are still unable to settle the issue of complexity for Problem 4.2.5. However, the two theorems suggest two ways of attacking the problem. As is evident from the proof of Theorem 4.2.7, it seems that the bipartite maximum matching algorithm might be employed to solve the problem. Also note carefully that Theorem 4.2.7 is more stringent and in fact the constructive proof of the sufficient part of Theorem 4.2.7 indicates that the resulting spanning tree would have exactly *K* leaves in *X*, where as in Problem 4.2.5, the requirement is for the leaves belonging to X to be greater than or equal to *K*. So in effect Theorem 4.2.7 also gives the necessary and sufficient condition of the following problem.

Problem 4.2.8. Variant of Maximum (Exact Number of) Leaf Spanning Tree for Bipartite Graphs

Let G be a connected bipartite graph with partite sets X and Y. Given a positive number $K \leq |X|$, *we are asked the question whether there is a spanning tree* T_G *in G such that number of leaves in* T_G *belonging to X is exactly equal to K.*

Example 11. Consider the bipartite graph G and the two spanning trees T_1 and T_2 in Figure 4.7(a), (b) and (c), respectively. Assume that in the input instance the value of K is given to be 2. Then T_1 is a desired spanning tree as defined in above problem but T_2 is not. Note that, for the given value of K, both T_1 and T_2 are desired spanning trees for Problem 4.2.5 (may see Example 10) since in both the spanning trees the number of leaves belonging to X is greater than or equal to 2. However, in this exact version of the problem, only T_1 is the desired tree.



(a) A Bipartite graph G with partite sets X and Y

(b) A spanning tree T_1 of G

(c) A Spanning Tree T_2 of G

Figure 4.7: Exact Leaf Spanning Tree for Bipartite Graphs

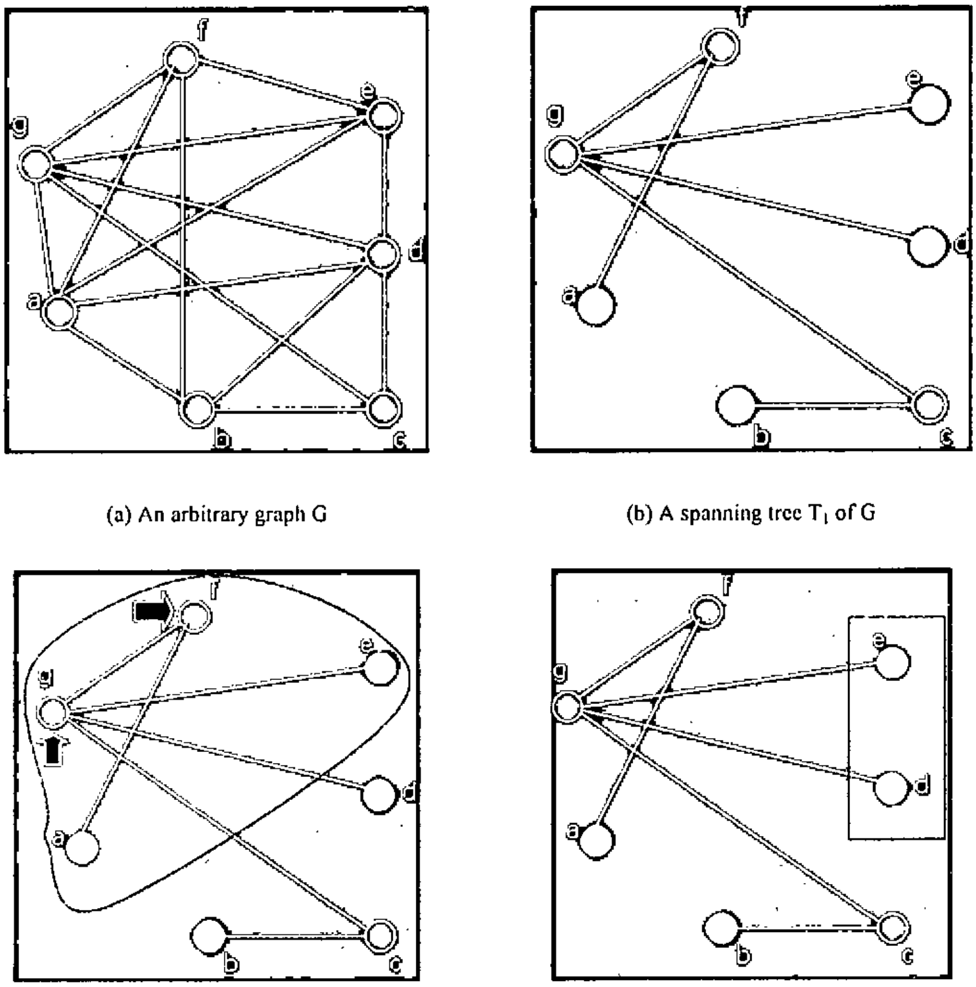
4.3. The Set Version

In this section we first introduce a new notion of "set version" of some well known decision problems. Consider the input instances of Maximum Leaf Spanning Tree Problem (Problem 1.1.2.5.) where we have an integer $K \leq |V|$ as a part of the input and we are asked the question whether there exists a spanning tree T in the input graph such that T has K or more leaves. Our notion of set version would pose a similar but different problem where the integer K in the input instance is replaced by a set $X \subseteq V$. To be specific we define the corresponding set version of Problem 1.1.2.5. as follows.

Problem 4.3.1. Set Version of Maximum Leaf Spanning Tree Problem.

Given a connected graph $G \equiv (V, E)$ and $X \subseteq V$, we are asked the question whether there is a spanning tree T such that $X \subseteq \Pi_T$, where $\Pi_T = \{v \mid v \text{ is a leaf in } T\}$.

For better understanding we first present suitable examples to illustrate both the normal and set version of Maximum Leaf Spanning Tree.



(a) An arbitrary graph G

(b) A spanning tree T_1 of G

(c) A spanning tree T_2 of G

(d) A spanning tree T_3 of G

Figure 4.8: Maximum Leaf Spanning Tree and its Set Version

Example 12. Consider the graph G in Figure 4.8(a). Assume first that the value of K is 3. Therefore the spanning tree T_1 in Figure 4.8(b) is a desired Maximum Leaf Spanning Tree. However, if K is 5 then T_1 fails to satisfy the requirements and hence is not a desired spanning tree. Note carefully that even if K is 4, T_1 remains the desired spanning tree. Now in order to consider the set version of the Maximum Leaf Spanning Tree, a subset X should be defined in the input instance instead of K.

Suppose X is defined to be the set {a, f, g, e}. Now in T_2 (Figure 4.8(c) there are two non-leaf nodes belonging to X. So T_2 can't be our desired spanning tree. However, if X = {e, d} then T_3 (Figure 4.8(d))is a desired spanning tree because all the nodes belonging to X in T_3 are leaves. Note carefully that in this case we are not at all concerned about the nodes outside the given X. We are only to ensure that all the vertices belonging to X should be leaves. There may be more leaves outside X.Now we present a necessary and sufficient condition for the existence of such a spanning tree as described in Problem 4.3.1, in the form of Theorem 4.3.1. As we point out later, the proof of Theorem 4.3.1 indicates the existence of a polynomial time algorithm to find one such spanning tree as opposed to the NP-Completeness of the original problem (Problem 1.1.2.5.).

Theorem 4.3.1. *Let $G = (V, E)$ be a connected graph and X a vertex subset of G. Also let f be a mapping from $Y = V \setminus X$ to natural number such for all $y \in Y$, $f(y) = |N_G(y) \cap X|$. Then there exists a spanning tree T such that $X \subseteq \Pi_T$, if and only if both of the following conditions hold true*

1. *$\langle Y \rangle$ is connected*
2. $$\sum_{y \in N_G(S) \cap Y} f(y) \geq |S|$$

for any $S \subseteq X$.

Proof.

First we prove the only if part:

Suppose there is a spanning tree T in G such that $X \subseteq \Pi_T$. In that case it is easy to realize that $\langle Y \rangle$ is connected since for all $x \in X$, $\deg_T(x) = 1$ and deleting X does not affect the connectivity of the rest of the graph. On the other hand let f_T be a mapping from Y to natural number such for all $y \in Y, f_T(y) = |N_T(y) \cap X|$. Note that $f_T(y) \geq 0$ for all y $\in Y$. Now since each of $x \in X$, is adjacent to only one $y \in Y$ in T so we must have for any $S \subseteq X$, $\sum_{y \in N_T(S) \cap Y} f_T(y) = |S|$. Again, since for all $y \in Y, f(y) = |N_G(y) \cap X| \geq |N_T(y) \cap X| = f_T(x)$ and $(N_T(S) \cap Y) \subseteq (N_G(S) \cap Y)$, we must have for any $S \subseteq X$,

$$\sum_{y \in N_G(S) \cap Y} f(y) \geq |S|.$$

Now we need to prove the if part:

Now let $W \subseteq Y$ be the set of vertices such that $W = \{w \mid w \in Y \text{ and } f(w) > 0\}$. We now construct a graph as follows. For all $w \in W$, we remove w and add new $f(w)$ vertices, $Z(w) = \{ z_1, z_2, \ldots, z_{f(w)}\}$ and edges $\{(w, v) \mid v \in N_G(w) \cap X \text{ and } w \in Z(w)\}$. It is easy to verify that the resultant graph H is bipartite with partite sets Z and X, where $Z = \bigcup_{w \in W} Z(w)$. Let φ be a binary relation from W to Z such that an element w corresponds to all the elements of the set $Z(w)$. Now consider any set $S \subseteq X$. Then the following must be true:

$$\begin{aligned}
|S| &\leq \sum_{y \in N_G(S) \cap Y} f(y) \\
&= \sum_{y \in N_G(S) \cap W} f(y) \\
&= \sum_{y \in \varphi(N_G(S) \cap W)} 1 \\
&= |\varphi(N_G(S) \cap W)| \\
&= |N_H(S)|
\end{aligned}$$

Hence in the bipartite graph H, by Hall's Theorem [15], there exists a matching M from X to Z. Let F' be the forest obtained from the matching M by identifying each vertex set $Z(w)$ with corresponding vertex w for all $w \in W$. Again, since $\langle Y \rangle = G_1$ (let) is connected, there is a spanning tree let T_1 in G_1. Now we get the graph $F = T_1 \cup F'$ which is connected and acyclic as follows. Each connected component in F' is either a star with a $y \in Y$ in the center or an isolated vertex $y' \in Y$. Now since $W \subseteq Y$, and $V(T_1) = Y$, it is clear that F is connected. Now it remains to show that F is acyclic. This follows from the fact that every star is acyclic and T_1 is acyclic. Thus, $F = T$ is a spanning tree of G since $V(T) = V(F) = V(G)$. Now it is clear that for all $x \in X$, $\deg_T(x) = 1$. Hence T is our desired spanning tree. □

The constructive proof of the sufficient part indicates the existence of a polynomial time algorithm to find one such spanning tree. We here first outline the algorithm as indicated in the proof and then deduce its complexity.

Algorithm 4.1

(Returns a desired spanning tree if one exists and false otherwise)

Begin

Step 1: If $\langle Y \rangle$ is connected then Find out a spanning tree T_1 from $\langle Y \rangle$ else return false.

Step 2: Construct the graph H as described in the proof of Theorem 3.1.

Step 3: Find out the matching M from X to Z in H. If X is not saturated then return false.

Step 4: Obtain the forest F'

Step 5: Return T where $T = T_1 \cup F'$ as described in the proof of Theorem 3.1.

End

The analysis of the algorithm is simple. Let $n = |V|$ and $m = |E|$. Step 1 can be done in $O(m)$ since we need an arbitrary spanning tree. Step 2 roughly takes $O(m)$. The merging in Step 5 needs $O(n)$. However the analysis of Step 3 needs some attention. Using the algorithm devised by Hopcraft and Karp [17] we can perform step 3 in $O(n_h^{0.5} \cdot m_h)$ running time, where $H = (V_h, E_h)$ and $n_h = |V_h|$ and $m_h = |E_h|$. Now from the construction of the graph H $n_h = |X| + |Z| = |X| + \sum_{y \in W} f(y) \leq |X| + O(m) = O(m)$. We now need to consider m_h. For each $z \in Z(\text{w})$ we have at most $\deg_G(w) \leq \Delta(G)$ edges in H where $\Delta(G)$ is the maximum degree in G. Since for all $w \in W$, $|Z(w)| = f(w) = |N_G(w) \cap X| \leq |N_G(w)| = \deg_G(w) \leq \Delta(G)$, we get the worst case total number of edges in H as follows. $m_h \leq \Delta(G) \cdot \Delta(G) = \Delta(G)^2 = O(n^2)$, since $\Delta(G) \leq n - 1 = O(n)$. Thus the running time of step 3 is $O(n_h^{0.5} \cdot m_h) = O(m^{0.5} \cdot n^2)$. Finally since Step 4 can be done in $O(m_h) = O(n^2)$ computational effort, the overall running time of the algorithm is dominated by Step 3 and hence is $O(\text{m}^{0.5} \cdot n^2)$. Finally the necessary and sufficient condition stated and proved in Theorem 4.3.1 ensures that Algorithm 3.1 returns false if the input graph does not have our desired spanning tree. Thus we get the following theorem.

Example 13. Figure 4.9 below gives an illustrative example of how the Algorithm 4.1 works.

Theorem 4.3.3. *Let $G = (V, E)$ be a connected graph with n vertices and m edges and X a vertex subset of G. If there exists a spanning tree T in G such that $X \subseteq \Pi_T$, then*

Algorithm 3.1 finds out one such spanning tree in $O(m^{0.5} \cdot n^2)$ computational effort and it returns false otherwise.□

Remark: Note carefully that although the original Maximum Leaf Spanning Tree problem is NP-Complete the corresponding set version of the problem can be solved in polynomial time.

Now we consider the set version of the Problem 4.2.5 defined in section 2 and consider its complexity issue.

Problem 4.3.4. Set Version of Problem 4.2.5

Let G be a connected bipartite graph with partite sets X and Y and $X_I \subseteq X$. We are asked the question whether there is a spanning tree T_G in G such that $X_I \subseteq \Pi_T$, where $\Pi_T = \{v \mid v \text{ is a leaf in } T\}$.

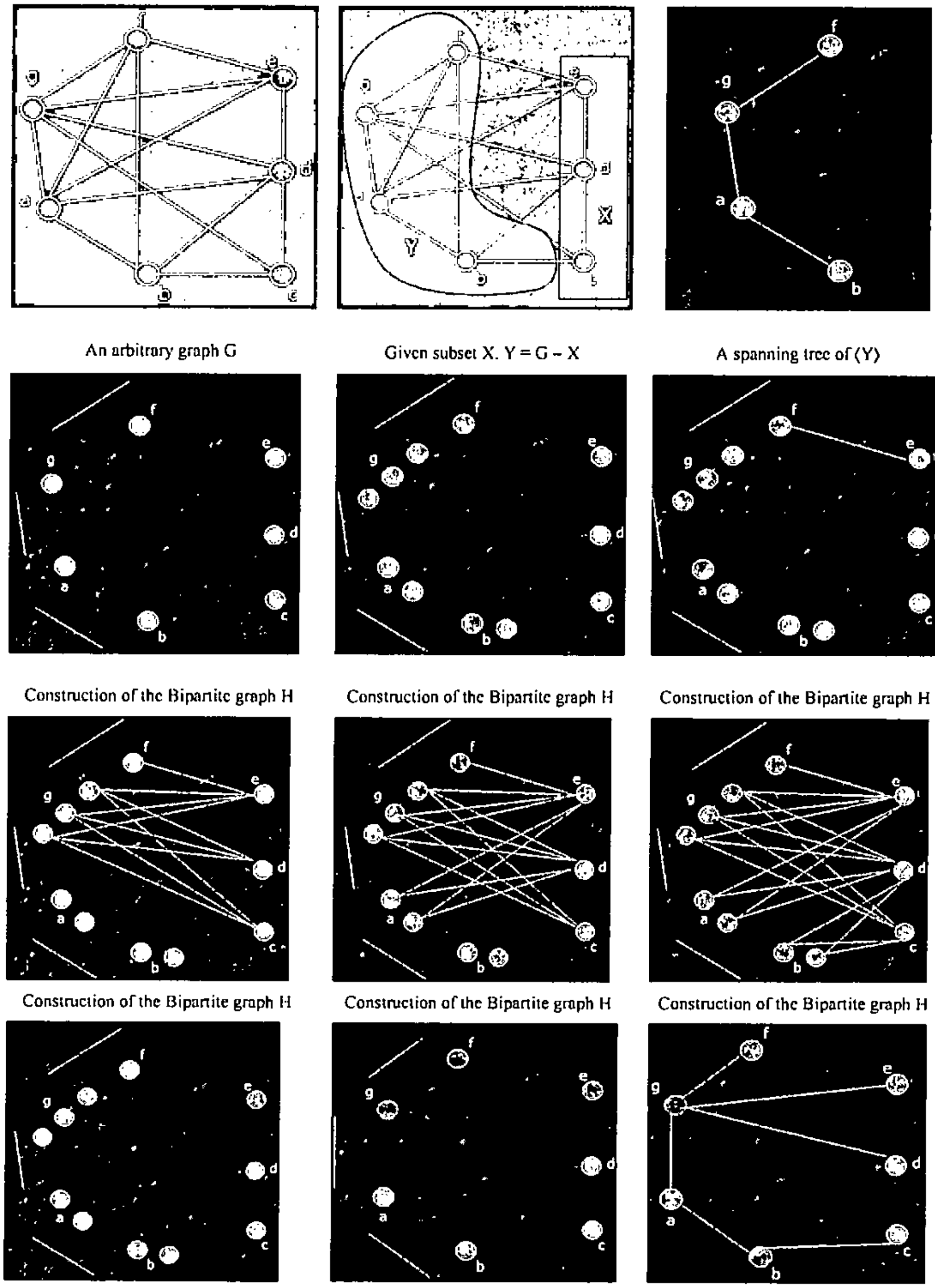
Theorem 4.3.5. Problem 4.3.4 *is polynomially solvable.*

Proof. The proof is simple and makes use of the Theorem 4.2.6. From Theorem 4.2.6 it is clear that G possess a spanning tree as defined in Problem 4.3.4 if and only if $\langle (X \setminus X_I) \cup Y \rangle$ is connected. Since this can be decided polynomially, the result follows. □

Above discussions may lead us to believe that the corresponding set versions of NP-Complete problems are not as hard as the original ones. However, this may not necessarily be the case as follows. We here consider the set version of Problem 4.2.1 i.e. Minimum Leaf Spanning Tree Problem.

Problem 4.3.6. Set Version of Minimum Leaf Spanning Tree Problem.

Given a connected graph $G \equiv (V, E)$ and $X \subseteq V$, we are asked the question whether there is a spanning tree T such that $\Pi_T \subseteq X$, where $\Pi_T = \{v \mid v \text{ is a leaf in } T\}$.



An arbitrary graph G | Given subset X. Y = G – X | A spanning tree of ⟨Y⟩

Construction of the Bipartite graph H | Construction of the Bipartite graph H | Construction of the Bipartite graph H

Construction of the Bipartite graph H | Construction of the Bipartite graph H | Construction of the Bipartite graph H

Finding a Matching from H | Identifying to the original graph | Resulting desired spanning tree

Figure 4.9: An example of how Algorithm 4.1 works

By employing the same strategy used to prove Theorem 4.2.2 (NP-Completeness of Minimum Leaf Spanning Tree Problem), Problem 4.3.6 can also be proved to be NP-Complete by assuming any $X \subseteq V$ such that $|X| = 2$. So we in effect get the following theorem.

Theorem 4.3.7. *Set Version of Minimum Leaf Spanning Tree Problem is NP-Complete.*□

4.4 Applications

Research on spanning trees has always been fueled by numerous theoretical and practical applications. In any implementation of communication algorithms, a common approach is to embed spanning trees with special properties on those networks. The root of the tree is the origin of the messages. The links of the embedded tree are used for message transmission. For example, in [11], 3 major communication problems for the start interconnection network, namely the *multi-node broadcasting, the single node scattering* and *the total exchange* problems, are considered and algorithms are devised using the aforementioned idea.

Spanning tree structure is of paramount importance in network and communication problems. New spanning tree structures are continuously evolving due to various network and communication problems arising in practical situations. For example, a special kind of spanning tree, namely *edge-disjoint* spanning tree that reduce the communication time of the single node broadcasting problem on the star network and offer many applications in the area of fault tolerant communication algorithms have been constructed in [12]. Another kind of spanning tree is the *optimal communication* spanning tree, first introduced by Hu in [19], which connects all given nodes and satisfies their communication requirements for a minimum total cost. This kind of spanning tree structures and variants thereof are extensively used in the design of optimal communication and transportation networks satisfying a given set of requirements [3, 26]. Relevant constrained spanning tree problems are for example the optimal communication spanning tree problem, the degree-constrained minimum spanning tree problem, the minimum Steiner tree problem or the capacitated minimum spanning tree problem.

The new problems introduced in this thesis are believed to be of importance in relevant areas. The various spanning tree structures and problem thereof, arising from applying various constraints and restrictions, in fact, correspond to various real world network and transportation problems. Also, the new notion of set version introduced here, seem to be applicable to numerous practical problems since there are many real world situations when a particular set of nodes or links are to be subjected to particular restrictions or constraints.

Chapter 5

CONCLUSION AND FUTURE RESEARCH

This thesis deals with spanning paths, cycles and trees. We have first presented sufficient conditions for the existence of Hamiltonian (spanning) paths and cycles in graphs. In particular, we investigated degree related conditions. We have also introduced new parameter in our sufficient conditions. The relation between independence number and Hamilton city also is investigated. Table 5.1 lists the conditions related to Hamiltonicity of graphs that we present in this thesis.

Then we consider spanning trees continue our investigation to find a relation between spanning tree problem, degree bounded spanning tree problem in particular, and independence number of graphs. Some new spanning tree problems are presented and there issues of complexity are considered. In particular we define Minimum Leaf Spanning Tree problem and RLSST Problem and prove both of them to be NP-Complete. We also consider variants of maximum leaf spanning tree problem for bipartite graphs (Problem 4.2.5 & Problem 4.2.8) and present two necessary and sufficient conditions. However, we neither prove the problem to be NP-Complete nor have presented any polynomial time algorithm to solve it. So the complexity of the problem still remains an open question. Finally a completely new notion, the "Set Version" is introduced and we raise the issue of whether the set versions of NP-Complete problems are as hard as the original problems. We here prove the set version of maximum leaf spanning tree problem to be polynomially solvable. It seems that the set versions of spanning tree problems should be generally easier to solve because in the case of set version,

-the question about a spanning tree should be answered *only for one set,* and we might have a polynomial time algorithm for answering the question;

-while in the original problem we might need to answer the question for *exponential number of sets,* and thus the corresponding algorithm might need exponential time;

However, it is shown here that the idea that set version is easier than the original problem may not necessarily be true in general. Our findings are shown in Table 5.2 and Table 5.3.

No.	Sufficient Conditions	Reference
1.	*Let G = (V, E) be a connected graph with n vertices and P be a longest path in G having length k and with end vertices u and v. Then the followings must hold true:* a. *Either $\delta(u, v) > 1$ or P is a Hamiltonian path contained in a Hamiltonian cycle.* b. *If $\delta(u, v) \geq 3$ then $d_p(u) + d_p(v) \leq k - \delta(u, v) + 2$* c. *If $\delta(u, v) = 2$, then either $d_p(u) + d_p(v) \leq k$ or P is a Hamiltonian path contained in a Hamiltonian cycle.*	***Theorem 3.1.3***
2.	*Let G = (V, E) be a connected graph with n vertices such that for all pair of distinct nonadjacent pair of vertices $u, v \in V$ we have $d(u) + d(v) + \delta(u, v) \geq n + 1$. Then G has a Hamiltonian path.*	***Theorem 3.1.5***
3.	Let $G = (V, E)$ be a connected graph with n vertices. If for all pairwise non-adjacent vertex-triples u, v, and w it holds that $d(u)+d(v)+d(w) \geq \frac{1}{2}(3n-5)$ then G has a Hamiltonian path.	***Theorem 3.2.1***
4.	Let G = (V, E) be a connected graph such that $\alpha(G) = 2$. Then G has a Hamiltonian path.	***Theorem 3.3.1.6***
5.	Let G = (V, E) be a 2-connected graph such that $\alpha(G) = 2$. Then G is Hamiltonian.	***Theorem 3.3.1.7***

Table 5.1: Sufficient Conditions for Hamiltonian Paths and Cycles

No.	Problem	Complexity Issue	Reference
1	Minimum Leaf Spanning Tree. *Given a connected graph $G \equiv (V, E)$ and a positive integer $K < \|V\|$, we are asked the question whether there is a spanning tree T of G such that K or less vertices have degree 1.*	NP-Complete	*Problem 4.2.1.*

No.	Problem	Complexity Issue	Reference
2	Restricted-Leaf-in-Subgraph Spanning Tree Problem. *Given $G \equiv (V, E)$ be a connected graph, X a vertex subset of G and a positive integer $K < \|X\|$, we are asked the question whether there is a spanning tree T_G such that number of leaves in T_G belonging to X is less than or equal to K.*	NP-Complete	*Problem 4.2.3.*
3	Variant of Maximum Leaf Spanning Tree for Bipartite Graphs *Let G be a connected bipartite graph with partite sets X and Y. Given a positive number $K \leq \|X\|$, we are asked the question whether there is a spanning tree T_G in G such that number of leaves in T_G belonging to X is greater than or equal to K.*	Open	*Problem 4.2.5.*
4	Variant of Maximum (Exact Number of) Leaf Spanning Tree for Bipartite Graphs *Let G be a connected bipartite graph with partite sets X and Y. Given a positive number $K \leq \|X\|$, we are asked the question whether there is a spanning tree T_G in G such that number of leaves in T_G belonging to X is exactly equal to K.*	Open	*Problem 4.2.8.*
5	Set Version of Maximum Leaf Spanning Tree Problem. *Given a connected graph $G \equiv (V, E)$ and $X \subseteq V$, we are asked the question whether there is a spanning tree T such that $X \subseteq \Pi_T$, where $\Pi_T = \{v \mid v \text{ is a leaf in } T\}$.*	Polynomial	*Problem 4.3.1.*
6	Set Version of Problem 4.2.5 *Let G be a connected bipartite graph with partite sets X and Y and $X_1 \subseteq X$. We are asked the question whether there is a spanning tree T_G in G such that $X_1 \subseteq \Pi_T$, where $\Pi_T = \{v \mid v \text{ is a leaf in } T\}$.*	Polynomial	*Problem 4.3.4.*
7	Set Version of Minimum Leaf Spanning Tree Problem. *Given a connected graph $G \equiv (V, E)$ and $X \subseteq V$, we are asked the question whether there is a spanning tree T such that $\Pi_T \subseteq X$, where $\Pi_T = \{v \mid v \text{ is a leaf in } T\}$.*	NP-Complete	*Problem 4.3.6.*

Table 5.2: Spanning Tree Problems

No.	Conditions	Reference
	Let $G = (V, E)$ be a connected graph such that $\alpha(G) = d$ where $d \geq 2$. Then G has a degree-d-bounded spanning tree.	*Theorem 4.1.2.*
	Given a connected graph $G = (V, E)$ with $\alpha(G) = d$, we can find a degree-d-	*Theorem 4.1.1.2.*

No.	Conditions	Reference
	bounded spanning tree in $O(n^2)$ computational effort, where $n = \|V(G)\|$.	
	Let G be a connected bipartite graph with partite sets X and Y and suppose K is a positive number such that $K \leq \|X\|$. Then there is a spanning tree T in G such that number of leaves in T belonging to X is greater than or equal to K if and only if there is a set $S \subseteq X$ such that $\|X \setminus S\| \geq K$ and $\langle S \cup Y \rangle$ is connected.	*Theorem 4.2.6.*
	Let G be a connected bipartite graph with partite sets X and Y and suppose K is a positive number such that $K \leq \|X\|$. Then there is a spanning tree T in G such that number of leaves in T belonging to X is greater than or equal to K if and only if there is a set $S \subseteq X$ such that all of the followings hold true: d. $\|X \setminus S\| \geq K$ e. $\langle S \cup Y \rangle$ is connected and f. For any subset $S' \subseteq S$, $\|N_G(S')\| \geq \|S'\| + 1$.	*Theorem 4.2.7.*
	Let $G = (V, E)$ be a connected graph and X a vertex subset of G. Also let f be a mapping from $Y = V \setminus X$ to natural number such for all $y \in Y$, $f(y) = \|N_G(y) \cap X\|$. Then there exists a spanning tree T such that $X \subseteq \Pi_T$, if and only if both of the following conditions hold true *1. $\langle Y \rangle$ is connected* *2.* $\sum_{y \in N_G(S) \cap Y} f(y) \geq \|S\|$ *for any $S \subseteq X$.*	*Theorem 4.3.1.*

Table 5.3: Spanning Tree Conditions

5.1 Future Research

Theorem 3.1.5 provides a sufficient condition for the existence of Hamiltonian path in graphs. A natural extension to this theorem should be to look for similar condition or to extend the theorem for Hamiltonian cycle. So this should be an area where future research effort can be given.

Theorem 3.2.1 (vertex triple) considers vertex triples. It would be interesting to see whether the idea can be extended for vertex quadruples and if possible to generalize the theorem for vertex k-tuples.

The rational behind each and every degree related sufficient conditions, in fact, is to ensure sufficient amount of edges in the graph considered, to force Hamiltonicity. Another interesting direction for research would be to find a lower bound of the number of edges to ensure the Hamiltonicity.

Various new problems on spanning trees are considered in this thesis. These problems are raised, in fact, by posing various restrictions and constraints on various graph parameters. Research in this direction can also be continued. For each new problem the corresponding complexity issue is definitely an area to be investigated. In our thesis, we could not settle the issue of complexity for the Problem 4.2.5 (Variant of Maximum Leaf Spanning Tree for Bipartite Graphs): surely this is another area for future research. Finally the new notion of set version seems to pose a completely new area for thorough research. In particular, the set version of various NP-Complete and NP-Hard problems should be investigated to see whether they are easier to solve and thus to see whether this new theory can be as fruitful in practical applications as it seems to be.

References

[1] J.A. Bondy and V. Chvátal, A Method in Graph Theory, Discr. Math. 15 (1976), pp 111-136.

[2] T. Böhme, H.J. Broersma, F. Göbel, A.V. Kostochka, M. Stiebitz, Spanning trees with pairwise nonadjacent endvertices, Discrete Math.,170 (1997), pp 219-222.

[3] R.S. Chan, Wide area network design, concepts and tools for optimization. San Francisco: Morgan Kaufmann Publishers, 1998.

[4] G. Chartand, O. R. Oellermann, Applied and algorithmic graph theory, McGraw–Hill, New York, 1993

[5] B. Chazelle, A minimum spanning tree algorithm with inverse-Ackermann type complexity, Journal of the ACM, 47:6 (2000), pp. 1028-1047.

[6] Y-Chuang Chen, Chang-Hsiung Tsai, Lih-Hsing Hsu, and Jimmy J.M. Tan, On some super fault-tolerant Hamiltonian graphs, Applied Mathematics and Computation, 148 (2004), pp. 729-741.

[7] K.W. Chong, Y. Han, and T.W. Lam, Concurrent threads and optimal parallel minimum spanning trees algorithm, Journal of the ACM, 48:2 (2001), pp. 297-323.

[8] T. H. Cormen, C. E. Leiserson, and R. L. Rivest, Introduction to Algorithms, MIT Press, Cambridge, MA, USA, Reprint of the 1990 original, 1998.

[9] N. Deo, Graph Theory with Applications to Engineering and Computer Science, Prentice-Hall, Englewood Cliffs, 1974.

[10] G.A. Dirac, Some Theorems on Abstract Graphs, Proc. Lond. Math. Soc. 2 (1952), pp 69-81.

[11] P. Fragopoulou, and S.G. Akl, Optimal communication algorithms on star graphs using spanning tree constructions, Technical Report No. 93-346, Department of Computing & Information Science, Queen's University, Kingston, Ontario, Canada, February 1993.

[12] P. Fragopoulou, and S.G. Akl, Edge-disjoint spanning trees on the star network with applications to fault tolerance, Technical Report No. 93-354, Department of Computing & Information Science, Queen's University, Kingston, Ontario, Canada, October 1993.

[13] M.R. Garey and D.S. Jhonson, Computers and Intractability: A Guide to the Theory of NP-Completeness, W.H. Freeman and Company, New York, Reprint of the 1979 original, 2000.

[14] R.L. Graham, and P. Hell, On the history of the minimum spanning tree problem, Annals of the History of Computing, 7:1 (1985), pp. 43-57.

[15] P. Hall, On representation of subsets, J. London Math Soc., 10 (1935), pp 26-30.

[16] F. Harary, and M. Lewinter, Hypercubes and other recursively defined Hamiltonian laceable graphs, Congressus Numerantium, 60 (1987), pp. 81-84.

[17] J. Hopcraft and R.M. Karp, An $O(n^{2.5})$ algorithm for maximum matching in bipartite graphs, SIAM J. Computing, 2 (1973), pp 225-231.

[18] S.Y. Hsieh, G.H. Chen, and C.W. Ho, Hamiltonian-laceability of star graphs, Networks, 36 (2000), 225-232.

[19] T. C. Hu, Optimum Communication Spanning Trees, SIAM J. Computing, 3 (1974), pp 188-195.

[20] W.T. Huang, M.Y. Lin, J.M. Tan, and L.H. Hsu, Fault-tolerant ring embedding in faulty crossed cubes, Proceedings of World Multiconf. Syst., Cybernetics and Inform. SCI'2000 IV, 2000, pp. 97-102.

[21] W.T. Huang, J.M. Tan, C.N. Hung, and L.H. Hsu, Fault-tolerant Hamiltonicity of twisted cubes, J. Parallel Distrib. Comput., 62 (2002), pp. 591-604.

[22] W.T. Huang, Y.C. Chuang, J.M. Tan, and L.H. Hsu, Fault-free Hamiltonian cycle in faulty Möbius cubes, J. Comput. Syst., 4 (2000), pp. 106-114.

[23] D. S. Johnson and J. K. Lenstra and A. H. G. Rinnooy Kan, The Complexity of the Network Design Problem, Networks, 8 (1978), pp 279-285.

[24] D.R. Karger, P.N. Klein, and R.E. Tarjan, A randomized liner-time algorithm to find minimum spanning trees, Journal of the ACM, 42:2 (1995), pp. 321-328.

[25] A. Kaneko, K.Yoshimoto, On spanning trees with restricted degrees, Information Processing Letters, 73 (2000), pp 163-165.

[26] A. Kershenbaum, Telecommunications network design algorithms. New York: McGraw Hill, 1993

[27] J.B. Kruskal, On the shortest spanning subtree of a graph and the traveling salesman problem, Proceedings of the American Mathematical Society, 7 (1956), pp. 48-50.

[28] M. Lewinter, and W. Widulski, Hyper-hamilton laceable and caterpillar-spannable product graphs, Computers Math. Appl., 34 (1997), 99-104.

[29] Jiping Liu, Construct Hamilton-type graphs, Congressus Numerantium, 122 (1996), pp. 90-98.

[30] Y.S. Myung, C.H. Lee, and D.W. Tcha, On the Generalized Minimum Spanning Tree Problem, Networks, 26 (1995), pp. 231-241.

[31] G.L. Nemhauser and L.A. Wolsey, Integer and Combinatorial Optimization, John Wiley & Sons, New York, Reprint of the 1988 original, 1999.

[32] O. Ore, Note on Hamiltonian Circuits, Am. Mat. Monthly 67 (1960), pp 55.

[33] C. H. Papadimitriou, M. Yannakakis, The Complexity of Restricted Spanning Tree Problems, Journal of the ACM, 29:2 (1982), pp 285-309

[34] S. Pettie, and V. Ramachandran, An optimal minimum spanning tree algorithm, Journal of the ACM, 49:1 (2002), pp. 16-34.

[35] P. Pop, The generalized minimum spanning tree problem, Ph.D Thesis, University of Twente, the Netherlands, 2002.

[36] R.C. Prim, Shortest connection networks and some generalizations, Bell System Technical Journal, 36 (1957), pp. 1389-1401.

[37] Mohammad Sohel Rahman and Md. Abul Kashem, Degree Restricted Spanning Trees of Graphs, accepted for presentation and publication in the AÇM SIGAPP Symposium of Applied Computing (SAC), Special Track on Computational Sciences, 2004.

[38] F. Roberts, Graph Theory and its Applications to Problems of Society, Society for Industrial and Applied Mathematics, Philadelphia, 1978.

[39] F. Roberts, (ed.), Applications of Combinatorics and Graph Theory to the Biological and Social Sciences, Springer-Verlag, New York, 1989.

[40] G. Simmons, Almost all n-dimensional rectangular lattices are Hamiltonian laceable, Congressus Numerantium, 21 (1978), pp. 649-661.

[41] R.E. Tarjan, Data structures and network algorithms, Society for Industrial and Applied Mathematics, 1983.

[42] H. Temperley, Graph Theory and Its Applications, Halsead Press, New York, 1981.

[43] K. Thulasiraman, and M. Swamy, Graphs: Theory and Applications, Wiley, New York, 1992.

[44] D.B. West, Introduction to Graph Theory, Prentice-Hall, Inc., New Jersey, Reprint of the 1996 original, 2000.

[45] H. Whitney, Congruent graphs and the connectivity of graphs. Amer. J. Math. 54 (1932), pp. 150-168

[46] M. Yannakakis, Node-and edge-deletion NP-Complete problems, Proceedings of the tenth annual ACM symposium on Theory of computing, May 01-03, 1978, San Diego, California, United States, p.253-264.