A STUDY ON GRAPH LABELINGS AND GRAPH SPECTRA

Thesis

Submitted in partial fulfillment of the requirements for the degree of

DOCTOR OF PHILOSOPHY

by

SAUMYA Y M

DEPARTMENT OF MATHEMATICAL AND COMPUTATIONAL SCIENCES
NATIONAL INSTITUTE OF TECHNOLOGY KARNATAKA,
SURATHKAL, MANGALORE - 575 025
SEPTEMBER 2022

Dedicated to

My family

DECLARATION

By the Ph.D. Research Scholar

I hereby declare that the Research Thesis entitled **A STUDY ON GRAPH LABELINGS AND GRAPH SPECTRA** which is being submitted to the **National Institute of Technology Karnataka, Surathkal** in partial fulfillment of the requirements for the award of the Degree of **Doctor of Philosophy** in **Mathematical and Computational Sciences** is a *bonafide report of the research work carried out by me*. The material contained in this Research Thesis has not been submitted to any University or Institution for the award of any degree.

(SAUMYA Y. M.)
Reg. No: 177030MA003
Department of Mathematical and Computational Sciences

Place: NITK, Surathkal.
Date: 12/09/2022

CERTIFICATE

This is to *certify* that the Research Thesis entitled **A STUDY ON GRAPH LABELINGS AND GRAPH SPECTRA** submitted by **Ms. SAUMYA Y. M.**, (Register Number: 177030MA003) as the record of the research work carried out by her is *accepted as the Research Thesis submission* in partial fulfillment of the requirements for the award of degree of **Doctor of Philosophy**.

12/9/2022

Dr. S. M. HEGDE

Research Supervisor

12/9/22

Chairman - DRPC

(Signature with Date and Seal)

Chairman
DUGC / DPGC / DRPC
Dept. of Mathematical and Computational Sciences
National Institute of Technology Karnataka, Surathkal
MANGALORE - 575 025

ACKNOWLEDGEMENTS

First and foremost, I would like to express my deep and sincere gratitude to my thesis advisor, Prof. S. M. Hegde, for his patience, motivation, and continuous support. I am grateful to him for being my guide and giving me the freedom to explore whatever I liked. His understanding, encouragement, and personal guidance have helped me immensely. I could not have imagined having a better advisor and mentor for my Ph.D study.

I would like to thank Prof. Annappa, Department of Computer Science and Engineering, and Dr. Srinivasa Rao Kola, Department of MACS, National Institute of Technology Karnataka, Surathkal, for their comments and suggestions as RPAC members during my research work.

I would like to thank former heads of MACS department Prof. Santhosh George, Prof. B. R. Shankar, Prof. S. S. Kamath, and the present head Dr. R. Madhusudhan, for providing an independent working environment with all the computing facilities required to carry out the research work. Further, I would like to express my heart-filled thanks to all the professors, faculty members, and office staff of the MACS department for helping me at various stages of my Ph.D.

I would like to thank all my teachers who have taught during various stages of my life. My journey began with the encouraging words of my colleagues, Ms. Usha Kiran and Ms.Thanmayee. I cannot thank them enough for making me realize what I can accomplish.

I would like to record special thanks to Dr. Vasudeva, Professor, NMAM Institute of Technology, Nitte, for the valuable discussions during this research work.

I am happy to thank my friends Rashmi K, Priya Kamath, Febin I. P, Shashanka Kulamarva, Chaitanya G. K, Mahesh Krishna, Kanagaraj K, Palanivel R, Sumukha S, Sachin Naik, for their help and support in the journey of my doctoral studies.

I am grateful to my father Shri. P. Mahalinga Patali and my mother Smt. Yashoda M, whose constant love and support kept me motivated and confident. My accomplishments and success are because they believed in me. Deepest thanks to my brother Mr. Sheethal Kumar Y. M, who kept me grounded, reminded me of what is important in life, and was always supportive of my adventures.

I express my sincere gratitude to Shri. Ramakrishna Kamath and Smt. Rekha Kamath, for their support and continuous encouragement. I am deeply grateful to my parents-in-law for their unfailing emotional support.

Finally, I would like to express my deep gratitude to my husband Mr. Vinay P and my dear daughter Samanvi P and acknowledge with pleasure that without their affection, moral support, and care, it would never have been possible for me to pursue my research. Finally, I thank and remember each and every person who helped me directly or indirectly in carrying out this work.

Place: NITK, Surathkal SAUMYA Y. M.
Date: 12/09/2022

ABSTRACT

The thesis mainly involves the study of graph labelings and graph spectra with a focus on their applications.

Labeled graphs provide a compact representation in which each element of an n-element set S (the signature of the graph) is assigned to a vertex of a graph with n vertices. Edges between vertices only exist when the sum or difference of their respective vertex numbers is one of the elements of the signature. A graph defined by such a signature is a sum graph or difference graph accordingly, and the labeling is called sum or difference labeling.

A graph $G(V,E)$ is referred to as a sum graph if there is an injective labeling known as sum labeling f from $V(G)$ to a set of distinct positive integers S such that $ab \in E(G)$ if and only if there is a vertex w in $V(G)$ such that $f(w) = f(a) + f(b) \in S$. Here w is called a working vertex. A sum labeling f is called an exclusive sum labeling with respect to a subgraph H of G, if f is a sum labeling of G, where H contains no working vertex. In this thesis, we obtain the exclusive sum number of several graphs. A possible application of the exclusive sum labeled complete k-partite graph in a relational database is given in this thesis. An autograph(difference graphs) with a signature whose elements are positive integers and contains no repeated elements is called a proper monograph. This thesis investigates the proper monograph labelings of several graphs and obtains their maximum independence number from their signatures.

Graphs serve as models for multiprocessor interconnection networks. A link exists between the graph spectra and the design of multiprocessor interconnection networks from the literature. The graphs are termed as well-suited if the value of $\mathsf{m}\Delta$ is small. Here m is the number of distinct eigenvalues and Δ is the maximum vertex degree. This thesis defines two new graph tightness values, $t_3(G)$ and $t_4(G)$, based on the literature's four types of graph tightness values. Further, we present several well-suited graphs for the design of the multiprocessor interconnection networks. Load balancing attempts to improve the performance of a distributed system by transferring some of the workloads of a congested node to other nodes for processing. This thesis studies the dynamic load balancing approach and presents a modified algorithm for load balancing in integer arithmetic. The proposed algorithm generates a balancing flow with a minimum l_2 norm.

Keywords: Exclusive sum labeling, proper monographs, idle vertices, maximum independent set, graph tightness, multiprocessor interconnection networks, load balancing, l_2-norm

Table of Contents

List of Figures

List of Tables

CHAPTER 1

INTRODUCTION

Graphs have evolved as a powerful mathematical tool in many applications, from Euler's solution to Konigsberg's Seven-Bridge problem (Euler (1741)) to the modern-day worldwide web. The definition of a graph is simple, but the theory that has grown up around it is enormous. A few of the essential topics of interest in graph theory include labeling, algebraic graph theory, coloring, domination, etc. Graph theory has immense potential for applications in computer science, physics, chemistry, operation research, and social networks, among others.

1.1 GRAPHS AND THEIR REPRESENTATIONS

A graph G consists of a set V of vertices (points, nodes) and a set E of edges (lines, connections). Every edge $e \in E$ is associated with ordered or unordered pair of elements of V, i.e., there is a mapping from the set of edges E to a set of ordered or unordered pairs of elements of V. The graph G with vertex set V and edge set E is written as $G = (V, E)$ or $G(V, E)$.

If an edge $e \in E$ is associated with an ordered pair (u, v) or an unordered pair (u, v), where $u, v \in V$, then e is said to connect u and v and u, v are called endpoints of e. If $e = uv$ is an edge of G, we say that u and v are adjacent and that each vertex is incident with e. The number of vertices in G is denoted as $|V(G)|$ and is called the order of G. Similarly, $|E(G)|$ denotes the number of edges in G and is called the size of G. If G is a (p, q) graph then G has p vertices and q edges.

Two or more edges joining the same pair of vertices are known as multiple edges, and an edge with identical ends is called a loop. A graph with no loops and multiple edges is called a *simple graph*. In a graph, if multiple edges are allowed but no loops, then the graph is known as a *multi graph*. If there is a path between two vertices v_i and

v_j, they are said to be connected in G. If all pairs of vertices in a graph G are connected, then G is called as a connected graph.

A subgraph $G^{'}$ of a graph G is a graph in which $V(G^{'}) \subseteq V(G)$ and $E(G^{'}) \subseteq E(G)$ and the assignment of vertices is same as in G. A spanning subgraph is a subgraph containing all the vertices of G. A tree T is a connected acyclic graph. A set of trees of G forms a forest. A spanning tree of G is a connected, acyclic, spanning subgraph of G. The degree of a vertex v is the number of edges incident with v; it is denoted by $deg(v)$. The minimum degree among the vertices of G is denoted by $\delta(G)$ and the maximum degree by $\Delta(G)$. If $\delta(G) = \Delta(G) = r$, then G is called a regular graph of degree r. If $r = n-1$ then the graph is a complete graph. A vertex with degree 1 is called a pendant vertex.

A bipartite graph is one whose vertex set can be partitioned into two subsets X and Y, so that each edge has one end in X and the other end in Y. If every vertex of X is joined with every vertex of Y then G is said to be complete bipartite graph and is denoted by $K_{m,n}$ with $|X| = m$ and $|Y| = n$. In particular a complete bipartite graph $K_{1,n}$ is called a star. Every non-trivial tree is a bipartite graph.

For all standard notations and terminologies we refer (Harary (1969)) and (West (1996)).

1.2 GRAPH LABELINGS

A graph labelling is a conditional assignment of real values or subsets of a set to the vertices, or the edges, or both. There are an infinite number of ways to label a graph. When conditions are imposed on specific parameters, such as vertices, edges, or both, such problems become interesting. The very first types of graph labelings or valuations (α,β, γ, and ρ valuations) were introduced by Rosa (1966) as a tool to solve the famous Ringel's conjecture. The β-valuation is known by the popular term graceful labeling. The graceful labeling of graphs has been extensively studied over the last few decades. Several graceful labeling problems have proven to be challenging to solve, and investigators have come up with an alternative, related but weaker, labelings to gain insight into those difficult problems.

Alternative graph labeling schemes have been investigated sporadically and often as an adjunct to an investigation of a related graceful labeling scheme. In recent years, different labeling schemes have been developed and studied in their own right as legitimate research topics unrelated to graceful labelings. Examples include sum and difference labelings.

Over the last few decades, enormous papers have produced an astounding array of graph labeling methods, which are still being enhanced due to many application-driven concepts. An extensive literature on labeling problems is given by Gallian (2021). Labeled graphs are becoming a more common type of mathematical model for a variety of applications. These labelled graphs are used to solve a variety of coding theory problems, such as the creation of good radar-type codes. They are also used to solve ambiguities in X-ray crystallography, create a communication network addressing system, and find the best circuit layouts. Bloom and Golomb (1977) have described the applications of labeled graphs.

1.3 SUM LABELINGS AND EXCLUSIVE SUM LABELINGS

The concept of a sum graph was introduced by Harary (1990). A graph $G(V,E)$ is referred to as a sum graph if there is an injective labeling known as sum labeling f from $V(G)$ to a set of distinct positive integers S such that for $a,b \in V(G)$, $ab \in E(G)$ if and only if there is a vertex w in $V(G)$ such that $f(w) = f(a) + f(b) \in S$. Here w is called a working vertex. An example of sum labeling is shown in Figure 1.1.

There can be no connected sum graph since an edge from the vertex with the largest label would require an even larger vertex. A graph can be converted into a sum graph by adding the isolated vertices whenever necessary. The sum number $\sigma(H)$ of a connected graph H is the least number of isolated vertices required for $G = H \cup \overline{K_{\sigma(H)}}$ to be a sum graph, where $\overline{K_{\sigma(H)}}$ is the complement of the complete graph, $K_{\sigma(H)}$. Such a graph G is said to be optimally labeled.

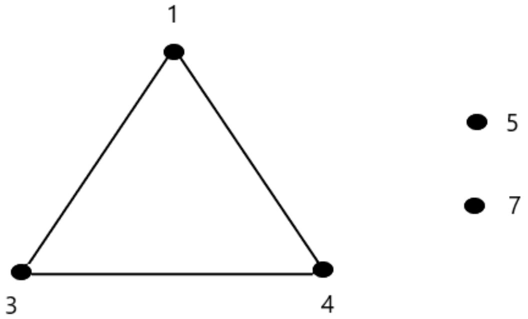


Figure 1.1 Sum graph labeling of $K_3 \cup \overline{K_2}$.

In an analogous manner, Harary et al. (1991) defined a real sum graph by allowing S to be any finite set of positive real numbers. Further, they also proved that every real sum is a sum graph. Harary (1994) generalised sum graphs by allowing S to be any set of integers, and termed them as integral sum graphs. Mod sum graphs were defined by Boland et al. (1990) using addition modulo m, a positive integer. Here $\rho(H)$ is called the mod sum number of a connected graph H and is the least number of isolated vertices required for $G = H \cup \overline{K_{\rho(H)}}$ to be a mod sum graph, where $\overline{K_{\rho(H)}}$ is the complement of the complete graph, $K_{\rho(H)}$.

Miller et al. (2005) introduced the concept of exclusive sum labeling. A sum labeling f is called an exclusive sum labeling with respect to a subgraph H of G, if f is a sum labeling of G, where H contains no working vertex. The smallest number r such that there exists an exclusive sum labeling f, which realizes $H \cup \overline{K_r}$ as a sum graph is called as the exclusive sum number $\varepsilon(H)$. A labeling f is an optimal exclusive sum labeling of a graph H if f is a sum labeling of $H \cup \overline{K_{\varepsilon(H)}}$ and H contains no working vertex. Figure 1.2 gives the exclusive sum labeling of $K_4 \cup \overline{K_5}$. Every exclusive sum graph is also a sum graph, but not the other way around. As a result, the exclusive sum number always exceeds or equals the sum number.

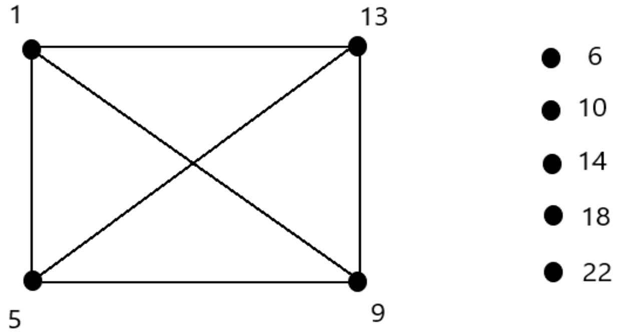


Figure 1.2 Exclusive Sum graph labeling of $K_4 \cup \overline{K_5}$.

Observation 1.3.1. *(Miller et al. (2005)) For any graph G, $\varepsilon(G) \geq \sigma(G)$.*

For any sum graph or exclusive sum graph labeling, the following holds.

Observation 1.3.2. *(Miller et al. (2005)) If F is a sum graph labeling of a graph G then so is kF, where k is a positive integer.*

The observation below gives the lower bound for exclusive sum number.

Observation 1.3.3. *(Miller et al. (2005)) Let Δ be the maximum degree of vertices in a graph G. Then $\varepsilon(G) \geq \Delta(G)$.*

The exclusive sum number is known for the following graphs:

i). Complete graphs. $\varepsilon(K_n) = 2n - 3$, for $n \geq 3$ (Bergstrand et al. (1989)).

ii). Paths. $\varepsilon(P_n) = 2$, for $n \geq 3$ (Miller et al. (2005)).

iii). Cycles. $\varepsilon(C_n) = 3$, for $n \geq 3$ (Miller et al. (2005)).

iv). Complete Bipartite graph. $\varepsilon(K_{m,n}) = m + n - 1$, for $m > 2$, $n > 2$ (Miller et al. (2005)).

v). Fan of order $n + 1$. $\varepsilon(F_n) = n$, for $n \geq 4$ (Tuga and Miller (2005)).

vi). Friendship graph. $\varepsilon(F_m) = 2m$, for $m \geq 2$ (Tuga and Miller (2005)).

vii). Wheels. $\varepsilon(W_n) = n$, for $n \geq 5$ (Tuga and Miller (2005)).

viii). Caterpillar. ε(caterpillar G) = $\Delta(G)$ (Tuga et al. (2005)).

1.4 APPLICATIONS OF SUM LABELINGS

Sum graph labelings allow storing a full graph as a collection of non-negative integers without having to record the edges explicitly. When sum graph labeling is used, all we need to store is the set of vertices together with a few isolated nodes, if required. The reason for this is that edges are implicitly specified in sum graphs.

When graph labeling is used to store a graph, the original component being stored is referred to as the primary graph, and any vertices added during the graph labeling process are referred to as isolates. There is no way to distinguish between the vertices of the primary graph and the isolates when all vertices are represented as a single set; they are all just vertices of the graph. The storage and manipulation of edges of a primary graph is the focus of applications. Sutton and Miller (2000) have used sum labeling of multipartite graphs to store links between individual rows from different tables. According to Sutton (2000), a sum-labeled multipartite graph can be used to represent a relational model in RDBMS. A relational database is wholly composed of tables. A table is a list of information about items like students, clients, employees, etc. Each table row contains information on a single occurrence of an item. The relationship between two rows belonging to two separate tables is called links. The vertices of the primary graph represent table rows, and the edges indicate links between these rows in the graph model of a relational database. Isolates are an artifact of the linkage storage mechanism and do not represent rows. It quickly became clear that within this domain, the vertices can be thought of as being partitioned into two sets because the labels assigned to the isolates can be stored separately from the labels assigned to the vertices of the primary graph, which are stored in the table rows. Assume an edge exists between a primary graph vertex and an isolate or between two isolates. Since one or both of the vertices do not reflect a table row, this edge cannot possibly represent a valid link.

A single integer is necessary to store all the links for a particular row in a sum labeling. The storage overhead of this system depends on the number of isolates that will be needed to create a sum labeling. Some of the advantages of using sum labeling in DBMS (These advantages are elaborately discussed in Ph.D. Thesis of Sutton (2000)) are Linkages as metadata, Referential Integrity, Recreating links, Flexibility, and Access paths.

Slamet et al. (2006) demonstrated how to distribute secret information to a group of people using sum graph labelings so that only approved subsets can recover the secret. For storing geographic information, Arlinghaus et al. (1993)) used sum graphs as data structures. The algebraic rule for assigning edges requires the sum graph to have at least one isolated vertex. Hence, the sum graph finds application in situations that require isolating one geographic location from others, such as facility location for toxic waste sites, detention centers, or other similar societally obnoxious facilities.

1.5 AUTOGRAPHS AND MONOGRAPHS

The concept of autograph was introduced by Bloom et al. (1979). An autograph labeling of G is a map f from set of vertices V to a set S of real numbers, with the property that for the vertices $a,b \in V$, $ab \in E$ if and only if there is a vertex $c \in V$ such that the difference of the vertex labels satisfy $|\mathsf{f}(a) - \mathsf{f}(b)| = \mathsf{f}(c)$. The set S is the signature of G and is defined as $S = \{\mathsf{s} \in R | \mathsf{s} = \mathsf{f}(v), \forall\, v \in V\}$.

If the autograph G has f as a mapping from vertex set V to a set of positive integers, then G is called a proper autograph. If the signature elements assigned as labels to the vertices of the autograph G contains only distinct elements, then G is a proper monograph (Harary called it as difference graph). Figure 1.3 and Figure 1.4 gives the autograph and proper monograph of C_4.

Bloom et al. (1979) gave the following necessary conditions for proper autographs.

Proposition 1.5.1. *(*Bloom et al. (1979)*) (1) The signature values s and $2s$ belong to the nodes adjacent to each other.*
(2) If a node is labeled as $r+t$, the signature values r and t belong to its adjacent nodes.
(3) There are no other types of adjacencies in proper autographs.

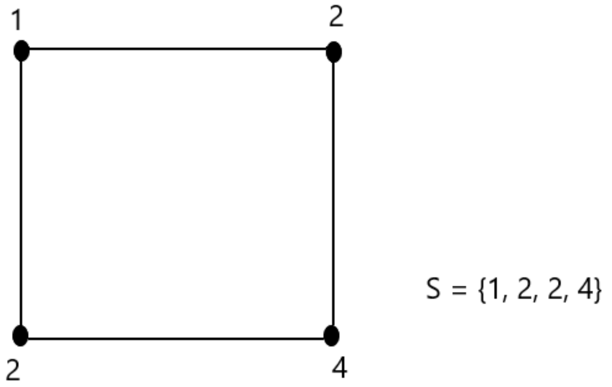


Figure 1.3 The autograph of C_4 from signature $S = \{1,2,2,4\}$.

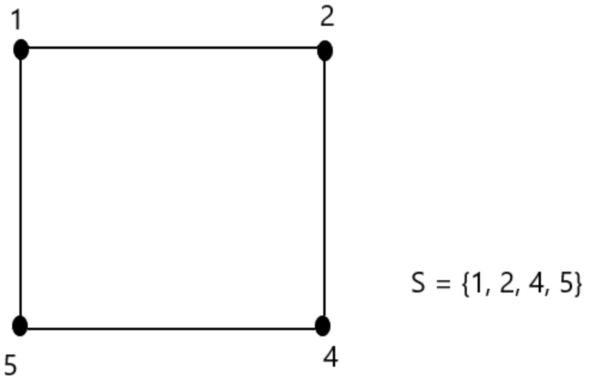


Figure 1.4 The proper monograph of C_4 from signature $S = \{1,2,4,5\}$.

According to Bloom et al. (1979), the autograph G consists of two types of edges. The edge of the first kind exists between nodes given values s and $2s$. The edge of the second kind exists between nodes with values r and $r+t$. Further, it was observed that a proper monograph does not have more than two edges of the first kind, i.e., at least $deg(u)-2$ edges that are incident with the vertex u must be of the second kind.

Some observations and theorem from Sugeng and Ryan (2007), are given as follows:

Observation 1.5.2. *(Sugeng and Ryan (2007)) Let α be a monograph labeling for a graph G and k be a positive integer. Then $k\alpha$ is also a monograph labeling for G.*

Observation 1.5.3. *(Sugeng and Ryan (2007)) Let G be a monograph. Then mG, for some positive integer m, is also a monograph.*

Observation 1.5.4. *(Sugeng and Ryan (2007)) A graph G is a monograph if and only if each of its components is a monograph.*

Bloom et al. (1979) constructed signatures of autographs such as complete graphs, complete bipartite graphs, paths, and cycles. Gervacio and Panopio (1982) studied generalized Petersen graphs that are proper monographs. Sonntag (2003) investigated the Difference labeling of Cacti. Sonntag (2004) later investigated the Difference labeling of digraphs. Sugeng and Ryan (2007) studied the properties of monographs and discovered signatures for cycles, fan graphs, kite graphs, and necklaces. Hegde and Vasudeva (2009) explored the construction of mod difference digraphs. Fontanil and Panopio (2014) observed that the independent set and vertex covering of a graph could be derived from the signatures of a proper monograph.

1.6 GRAPH SPECTRA

The theory of graph spectra (Spectral Graph Theory) can be considered an attempt to utilize linear algebra, particularly the well-developed theory of matrices, as a tool to study graph theory and its applications.

A graph can be linked to a matrix, or, to put it another way, a graph can be represented using matrices. Let G be a simple graph with the edge set $E(G) = \{e_1, e_2, ..., e_m\}$ and the vertex set $V(G) = \{v_1, v_2, ..., v_n\}$. $A(G)$ is the $n \times n$ matrix in which element a_{ij} is the number of edges in G with end vertices v_i, v_j. It's worth noting that every adjacency matrix is symmetric. A simple graph G's adjacency matrix has entries 0 or 1, with 0s on the diagonal. The sum of the items in the rows for v in $A(G)$ is the degree of v.

The eigenvalues of A are the n roots of the characteristic polynomial $P_{G(x)} = det(xI - A)$. The eigenvalues are independent of the labeling of the vertices of G because similar matrices have the same characteristic polynomial. Since A is a symmetric matrix with real entries, these eigenvalues are real and are denoted as $\lambda_1, \lambda_2, ..., \lambda_n$. Here $\mathsf{m} = \mathsf{m}(G)$ denotes the number of distinct eigenvalues of G. Unless and otherwise indicated, it is assumed that the eigenvalues can be arranged as $\lambda_1 \geq \lambda_2 \geq ... \geq \lambda_n$. The largest eigenvalue $\lambda_1(G)$ is called the index of G. The eigenvalues of A are the real numbers λ satisfying $Ax = \lambda x$ for some non-zero vector $x \in \mathbb{R}^n$. Each such vector x is called an eigenvector of the matrix A (or of the labelled graph G) corresponding to the eigenvalue λ. The unique positive unit eigenvector corresponding to the index of a connected graph G is called the principal eigenvector of G.

For a graph G with vertex set $\{1, ..., n\}$, let D be the diagonal matrix $diag(d_1, ..., d_n)$, where d_i denotes the degree of vertex $i(i = 1, ..., n)$. The Laplacian matrix of a graph G is the matrix $L = D - A$ and the signless Laplacian is the matrix $Q = D + A$. $L_G(x)$ denotes the characteristic polynomial of L of G. Matrix L is symmetric, positive semi-definite and has n non-negative, real-valued eigenvalues: $v_1(G) \geq v_2(G) \geq ... \geq v_n(G) = 0$. Note that $v_n = 0$ since $Lj = 0$, where j is the all-1 vector in $\mathbb{R}^n$. Here $Q_G(x)$ denotes the characteristic polynomial of Q and is called the Q-polynomial of G. Q-spectrum and Q-eigenvalues denotes the spectrum and the eigenvalues of Q respectively. We denote the i^{th} largest eigenvalue of Q by $q_i = q_i(G)$. Since Q is a positive semi-definite matrix the eigenvalues satisfy: $q_1(G) \geq q_2(G) \geq ... \geq q_n(G) \geq 0$.

1.7 APPLICATIONS OF GRAPH SPECTRA

Graph spectra has several applications in Computer Science. Some of them are Data mining, Complex networks and Internet, Pattern Recognition, Load balancing and Multiprocessor Interconnection Networks, Statistical Databases, Social Networks, and many other areas. Cvetković and Simić (2011) gave a brief survey of Graph Spectra in Computer Science.

A graph G can serve as a model for the Multiprocessor Interconnection Networks (MINs) in which the vertices represent the processors, while the edges represent connections between processors. The time to exchange data between different processing units is one of the main communication overheads in multiprocessor systems. Interconnection networks with shorter paths between processors and the average number of connections per processor are preferred. In order to minimize communication time within multiprocessor networks, they must comply with two contradictory characteristics: reduce the number of wires (diameter D) and maximize the rate of exchange of data (maximum vertex degree Δ). The diameter D along with maximum vertex degree Δ of the graph play an essential role in designing multiprocessor topologies.

Let δ and Δ be the minimum and maximum degree, respectively, $\overline{d}$ be the average vertex degree, λ_1 be the largest eigenvalue, D be the diameter, and m be the number of distinct eigenvalues. Elsässer et al. (2003) established a link between the graph spectra and the design of multiprocessor topologies. The key conclusion was as follows: for a given graph G, if $\mathsf{m}\Delta$ is small, it was anticipated that the corresponding multiprocessor topology would have excellent communication properties and was called well-suited. Also, it was noted that there exists an optimal load balancing algorithm that completed load balancing in $\mathsf{m} - 1$ computational steps. Further graphs having large $\mathsf{m}\Delta$ have been named ill-suited and hence they are found to be unsuitable for multiprocessor network design.

From Cvetković et al. (1995), considering the inequalities, $\delta \leq \overline{d} \leq \lambda_1 \leq \Delta$ and $D \leq$

$\mathsf{m}-1$, Cvetković and Davidović (2008), defined four types of graph tightness values, namely $t_1(G), stt(G), spt(G)$, and $t_2(G)$. Here the use of largest eigenvalue $\lambda_1(G)$ and diameter D, instead of Δ and m, was considered more suitable. Cvetković (1971) proved that the index of the graph $\lambda_1(G)$ is equal to the dynamic mean value of the vertex degrees. Since the dynamical mean value of the vertices takes into account not only the immediate vertex neighbors but also the neighbors of the neighbors, Cvetković and Davidović (2008) suggested that it was appropriate to use the largest eigenvalue (index). Furthermore, they showed that the four tightness values are partially ordered by the relation '$\leq$' as follows:

$$t_2(G) \leq stt(G) \leq t_1(G)$$
$$t_2(G) \leq spt(G) \leq t_1(G)$$

Later, they concluded that the graphs having small tightness values of $t_2(G)$ are more suitable for designing multiprocessor interconnection networks.

According to Krueger and Finkel (1984), load balancing is concerned with distributing the workload among the processors of a distributed system to prevent some processors from being idle when others have a substantial amount of workload. The load balancing algorithm reduces the workload difference by performing local load exchange across processors. It is important to note that many communication exchanges should not add considerably to the load balancing algorithm's overhead.

There are two load balancing approaches: static and dynamic load balancing. As in Eager et al. (1986), static algorithms assign tasks to the processors based on predetermined rules, and once assigned, the load does not alter. A task is either assigned to the processor that received it or transferred to another processor; however, the decision to transfer the task is made independently of the system state. In the dynamic load-balancing approach as in Cybenko (1989); Barmon et al. (1991), the load distribution decisions are based on the current workload at each node in the distributed system. Loads can move dynamically from an overloaded node to an underloaded node to improve performance. We consider dynamic load balancing rather than the static approach because the former produces a better performance since it makes load balancing decisions based on the system's current state.

There are numerous dynamic load balancing algorithms, such as diffusion type algorithms [Cybenko (1989),Boillat (1990), Song (1994)] and the dimension exchange algorithm [Cybenko (1989), Xu and Lau (1992), Xu et al. (1995)]. The diffusion approach has gained a lot of attention from researchers over the last few decades in order to address the load-balancing problem. Dimension exchange employs only pairwise communication, iteratively balancing with one neighbor after the other. In contrast, diffusion techniques assume that a node of the graph can send and receive messages to/from all of its neighbors at the same time.

1.8 ORGANIZATION OF THE THESIS

Chapter 2 presents a modified construction of exclusive sum labeling for the odd cycle C_n when $n > 5$. We also give the exclusive sum labeling of several graphs. In Chapter 3, we provide the exclusive sum number of the complete k-partite graph $K_{r_1,r_2,\ldots,r_k}$ and show their application to store and manipulate links in the relational database system.

In Chapter 4, we present the proper monograph labelings of several graphs and then show that the signatures of these proper monographs give the maximum independent sets of these graphs.

In Chapter 5, we present several graphs as models for multiprocessor interconnection networks for which the tightness values range from $O(\sqrt[4]{N})$ to $O(\sqrt{N})$, where N is the order of the graph under consideration. Also, we define two new graph tightness values; namely, the Third type mixed tightness $t_3(G)$ and the Second type of Structural tightness $t_4(G)$ and show that these tightness types are more straightforward to calculate than the others for the considered graphs.

Chapter 6 proposes an algorithm that results in a balancing flow with a lesser l_2-norm than the l_2-norm of the balancing flow generated by the existing algorithm. Further, we show that the load balancing by the proposed algorithm is done in $O(n^3)$ time.

In Chapter 7, we give a conclusion and scope for future research.

CHAPTER 2

EXCLUSIVE SUM LABELINGS OF GRAPHS

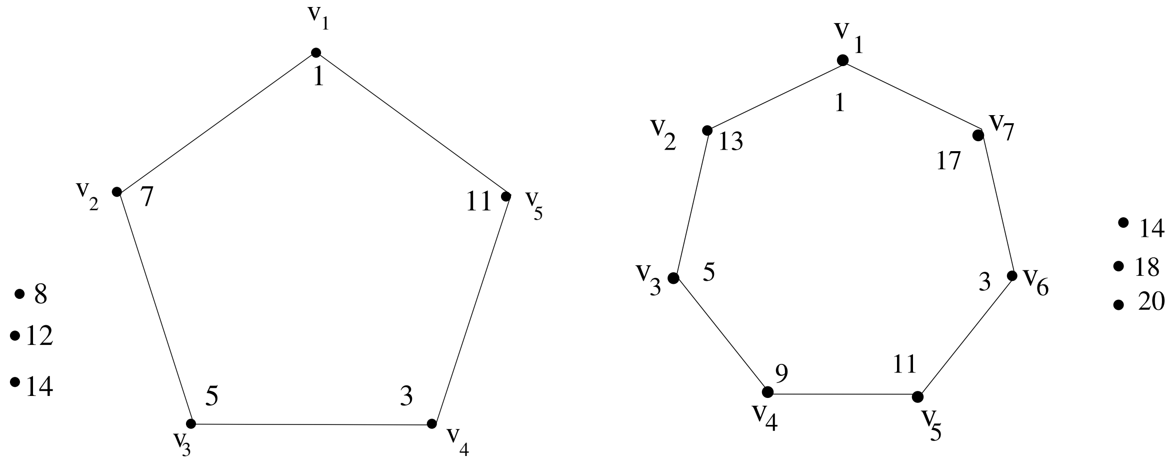
Miller et al. (2005) gave the construction of exclusive sum labeling for odd cycles. This chapter shows that the above construction failed to produce an exclusive sum labeling for odd cycle C_n when $n = 5$ and $n = 7$. We present a modified exclusive sum labeling for cycles C_n of odd length $n > 5$. Also, we give exclusive sum labeling of the Cartesian product of cycle C_n and K_2, i.e., $C_n \square K_2$, Cartesian product of complete graph K_n and K_2, i.e., $K_n \square K_2$, the disjoint union of paths, the disjoint union of cycles.

2.1 EXCLUSIVE SUM NUMBER OF AN ODD CYCLE C_n

In the proof for the exclusive sum labeling of a cycle in (Miller et al. (2005), Theorem 4), the exclusive sum labeling of odd cycles C_5 and C_7 is obtained as shown in Figure 2.1. On observation, we found that the labeling defined here does not produce an exclusive sum labeling. For the cycle, C_5, the sum of the isolated vertex label(or working vertex label) 8 and the non-working vertex label 3 produces another non-working vertex label 11. Similarly, for the cycle, C_7, the sum of the working vertex label 14 and the non-working vertex label 3 produces another non-working vertex label 17. By observing, it is clear that the given labeling violates the definition of exclusive sum labeling in both cases. In the following theorem, we provide the modified exclusive sum labeling of an odd cycle.

Theorem 2.1.1. *The exclusive sum number of an odd cycle C_n is $\varepsilon(C_n)$=3, for $n > 5$.*

Proof. Let $f(G)$ denote the set of the labels assigned to the vertices of the graph G. Consider the cycle C_5 with $v_1, v_2, \ldots, v_5$ as its vertices. One can assign the vertices



(a) Cycle C_5.

(b) Cycle C_7.

Figure 2.1 Exclusive Sum labeling of Cycle C_5 and Cycle C_7 as given in (Miller et al. (2005), Theorem 4.)

of the C_5 with labels as $f(v_1) = 1, f(v_2) = 17, f(v_3) = 7, f(v_4) = 11$ and $f(v_5) = 23$ (resulting in three isolated vertices receiving the labels 18, 24 and 34).

Consider an odd cycle with $n > 5$. Let $v_1, v_2, \dots, v_n$ be the vertices of the cycle and denote the vertices v_{n-2} and v_{n-1} as x and y respectively as shown in the Figure 2.2. We

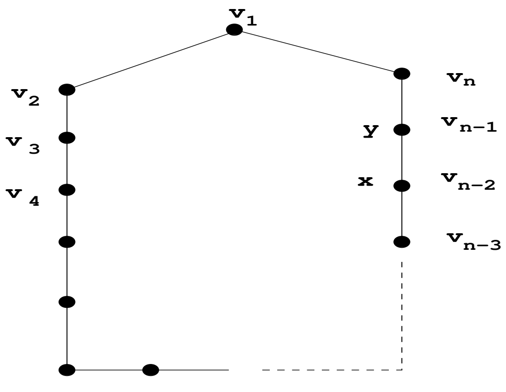


Figure 2.2 Cycle C_n.

first consider the path of length three whose vertices are v_n, y, x, v_{n_3}. Apart from these vertices, we label the odd vertices beginning from v_1 up to the last odd vertex available with a difference of six between two consecutive labels in the sequence. We switch to label the even vertices by adding a sum of four to the last odd vertex label. Similarly, we label the even vertices with a difference of six. A sum of six is added to the label of v_2 to obtain the label of v_n. Finally, we label the remaining vertices such that three edge sums $v_n + y$, $x + y$, and $v_{n-3} + x$ result in only three isolated vertices.

Then define the labeling $f : V(G) \to N$ by

$$f(v_i) = \begin{cases} 3i-2, \text{ for odd i, } 1 \leq i \leq n-4 \\ f(v_n) - 3i, \text{ for even i, } 2 \leq i \leq n-3 \end{cases} \tag{2.1.1}$$

where $f(v_n) = 6n - 19$. The labels $f(x)$ and $f(y)$ are defined as

$$f(x) = \begin{cases} \dfrac{9n-33}{2}, \text{ if n} \equiv 3(mod\ 4) \\ \dfrac{9n-27}{2}, \text{ if n} \equiv 1(mod\ 4) \end{cases}, f(y) = \begin{cases} \dfrac{3n-15}{2}, \text{ if n} \equiv 3(mod\ 4) \\ \dfrac{3n-9}{2}, \text{ if n} \equiv 1(mod\ 4) \end{cases} \tag{2.1.2}$$

The three working vertex labels are:

$$\begin{cases} f(w_1) = (f(v_n) - 5), \\ f(w_2) = (f(v_n) + 1) \text{ and} \\ f(w_3) = (f(v_n) + f(y)). \end{cases} \tag{2.1.3}$$

To prove that there are no additional edges between the vertices, we consider the following cases:

1. From (2.1.1) and (2.1.2), it follows that the labeling f assigns odd labels to the non-working vertices (vertices of the cycle) and even labels to the working vertices (isolated vertices). Hence, it is clear that two non-working vertex labels cannot produce any other non-working vertex labels. Also, two working vertex labels cannot produce any other non-working vertex labels.

2. From (2.1.3), one can observe that two working vertex labels will not produce another working vertex label since we cannot have

$$(f(v_n) - 5) = (f(v_n) + 1) + (f(v_n) + f(y))$$
$$(f(v_n) + 1) = (f(v_n) - 5) + (f(v_n) + f(y))$$
$$(f(v_n) + f(y)) = (f(v_n) - 5) + (f(v_n) + 1)$$

3. The remaining possibility is the sum of labels of a non-working vertex and a working vertex resulting in another non-working vertex label. Let $f(v_i), f(v_j), f(v_k) \in f(C_n)$ where $1 \leq i, j, k \leq n$, be the labels assigned to the non-working vertices. From (2.1.3), we have $f(w_1), f(w_2), f(w_3)$, as the working vertex labels. Let

$1 \leq i, j \leq n$, where $i \neq j$. As we observe, $f(v_n) = 6n - 19$ is the largest label of the cycle. Then for any non-working vertex label say $f(v_i)$, we have

$$f(v_i) \neq \begin{cases} f(w_2) + f(v_j) \text{ for } 1 \leq i, j \leq n \text{ where} f(w_2) = f(v_n) + 1 \\ f(w_3) + f(v_j), \text{ for } 1 \leq i, j \leq n \text{ where} f(w_3) = f(v_n) + f(y) \end{cases}$$

Suppose for some $i, j \in \{1, 2, \ldots, n\}$, where $i \neq j$ and a working vertex label $f(w_1) = f(v_n) - 5$, we have

$$(f(v_n) - 5) + f(v_j) = f(v_i) \leq f(v_n) \tag{2.1.4}$$

Then from (2.1.4) and the labeling defined in (2.1.1) and (2.1.2), one can observe that $f(v_j) < 5$ and can only be 1 or 3.

(a) If $f(v_j) = 1$, then

$$f(v_i) = (f(v_n) - 5) + 1 = f(v_n) - 4 = 6n - 23 \tag{2.1.5}$$

(b) If $f(v_j) = 3$, then

$$f(v_i) = (f(v_n) - 5) + 3 = f(v_n) - 2 = 6n - 21 \tag{2.1.6}$$

Now we verify that the labels $6n - 23$ and $6n - 21$ cannot be the labels of the vertices of the cycle.

- From (2.1.1) and (2.1.5), it follows that

$$3i - 2 = 6n - 23 \implies i = 2n - 7.$$

  (a) if $i = 1$, then $1 = 2n - 7$ which results in $n = 4$ contradicting the fact that n is odd.

  (b) if $i = n - 4$, then $n - 4 = 2n - 7$ which results in $n = 3$, contradicting the fact that $n > 5$.

- Consider (2.1.1) and (2.1.5). It follows that $6n - 23 = f(v_n) - 3i \implies 3i = 4$, which contradicts the fact that i is an integer.

Now from (2.1.2) and (2.1.5), we consider the following cases:

- if $6n - 23 = \dfrac{9n - 33}{2}$, then $n = \dfrac{13}{3}$

- if $6n-23 = \dfrac{9n-27}{2}$, then $n = \dfrac{19}{3}$
- if $6n-23 = \dfrac{3n-15}{2}$, then $n = \dfrac{31}{9}$
- if $6n-23 = \dfrac{3n-9}{2}$, then $n = \dfrac{55}{9}$

On observation, the above four cases also contradict the fact that n is an integer. Hence, it is clear that none of the vertices of the cycle are assigned with the label $6n-23$.

Similarly, we now verify that none of the vertices of the cycle is assigned with the label $6n-21$.

- From (2.1.1) and (2.1.6), we have $3i-2 = 6n-21 \implies 3i = 6n-19$ which contradicts the fact that i is an integer.
- From (2.1.1) and (2.1.6), we have $f(v_n)-3i = 6n-21 \implies 3i = 2$, which contradicts the fact that i is an integer.

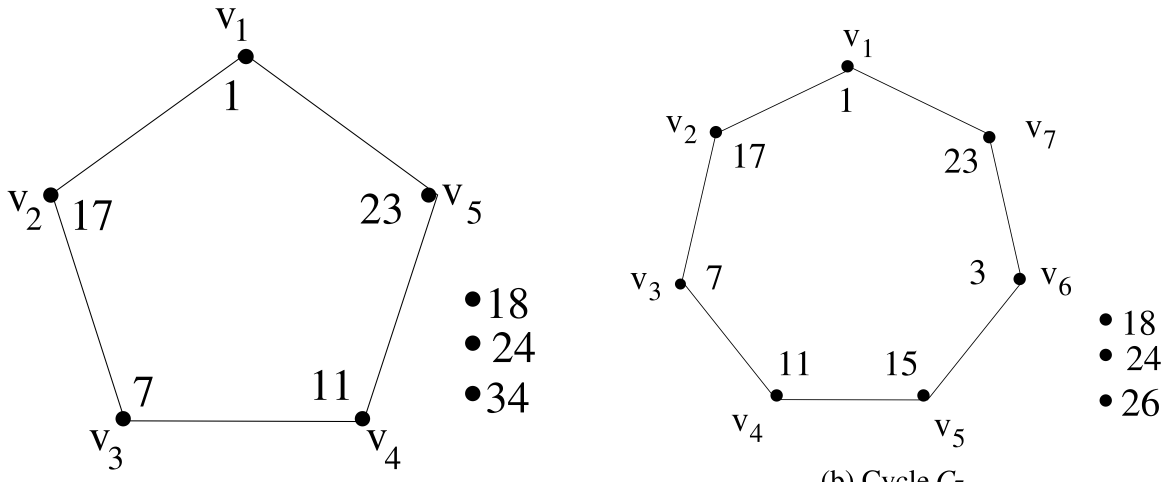
Now from (2.1.2) and (2.1.6), we consider the following cases:

- if $6n-21 = \dfrac{9n-33}{2}$, then $n = 3$
- if $6n-21 = \dfrac{9n-27}{2}$, then $n = 5$
- if $6n-21 = \dfrac{3n-15}{2}$, then $n = 3$
- if $6n-21 = \dfrac{3n-9}{2}$, then $n = \dfrac{33}{9}$

The above four cases contradict the fact that $n > 5$ and n is an integer, and it is clear that the label $6n-21$ is also not assigned to any of the vertices of the cycle. Hence, we conclude that no extra edges are induced between the working vertices or between the non-working and working vertices. Therefore the exclusive sum number for an odd cycle is $\varepsilon(C_n) = 3$, for $n > 5$.

□

Figure 2.3 illustrates the exclusive sum labeling of C_5 and C_7 obtained from Theorem 2.1.1.



(a) Cycle C_5.

(b) Cycle C_7.

Figure 2.3 Exclusive Sum labeling of Cycle C_5 and Cycle C_7.

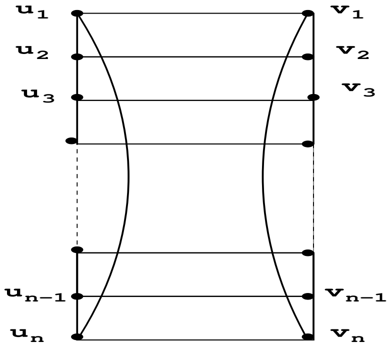


Figure 2.4 Cartesian product of $C_n \square K_2$.

2.2 EXCLUSIVE SUM NUMBER OF $C_n \square K_2$

Consider the Cartesian product of cycle C_n with the complete graph K_2. The vertices of the two cycles are $u_1, u_2, ..., u_n$ and $v_1, v_2, ..., v_n$ as shown in Figure 2.4.

There are two possible ways of labeling $C_n \square K_2$:

- Case 1: Label the vertices of $C_n \square K_2$ such that $u_i + u_{i+1} = v_i + v_{i+1}$ for $1 \leq i \leq n-1$. Then, the sum $u_i + v_i$ will result in n different isolated (or working) vertices.

- Case 2: Label the vertices in the path from $v_1, u_1, u_2, v_2, v_3, u_3, u_4, ..., v_{n-2}, u_{n-2}$, u_{n-1}, v_{n-1} when n is odd, or the path from $v_1, u_1, u_2, v_2, v_3, u_3, u_4, ..., u_{n-2}, v_{n-2}$, v_{n-1}, u_{n-1} when n is even. It is known that $\varepsilon(P_n) = 2$. Hence the above labeling will produce two distinct sums. Also, the vertex sums $v_1 + v_2 = u_2 + u_3 = v_3 + v_4$

$= ... = v_{n-2} + v_{n-1}$, when n is odd or $v_1 + v_2 = u_2 + u_3 = v_3 + v_4 = ... = u_{n-2} + u_{n-1}$, when n is even, results in a single distinct sum. Hence, a total of three isolated vertices will be required. But the graph $C_n \Box K_2$ still consists of the following as adjacent vertices: (v_1, v_n), (u_1, u_n), (u_n, v_n), (u_{n-1}, u_n) and (v_{n-1}, v_n), which implies that more than three isolated vertices are required. The following theorem gives the exclusive sum number of the graph $C_n \Box K_2$, for $n \geq 4$.

Theorem 2.2.1. *The exclusive sum number $\varepsilon(C_n \Box K_2) = 5$, for $n \geq 4$.*

Proof. Let $k \geq 1$ and odd, $d \geq 4$ and even, where $k < d$ and are co-prime. The labeling $f : V(G) \to N$ is defined as follows:

$$f(u_i) = \begin{cases} k + (2i-2)d, & \text{for odd i, i} < n \\ k + (4n-2i-4)d, & \text{for even i, i} < n \end{cases} \tag{2.2.1}$$

$$f(v_i) = \begin{cases} k + (2i-2)d, & \text{for even i, i} < n \\ k + (4n-2i-4)d, & \text{for odd i, i} < n \end{cases} \tag{2.2.2}$$

$$f(u_n) = \begin{cases} k + (3n-5)d, & \text{if n is even} \\ k + (3n-6)d, & \text{if n is odd} \end{cases} \tag{2.2.3}$$

$$f(v_n) = \begin{cases} k + (n-3)d, & \text{if n is even} \\ k + (n-2)d, & \text{if n is odd} \end{cases} \tag{2.2.4}$$

The working vertices receive the labels as follows: $2k + (4n-4)d$, $2k + (4n-6)d$, $2k + (4n-8)d$, $2k + (3n-6)d$, and $2k + (5n-8)d$ when n is odd. When n is even, their labels are $2k + (4n-4)d$, $2k + (4n-6)d$, $2k + (4n-8)d$, $2k + (3n-5)d$, and $2k + (5n-9)d$. Now consider the following:

1. Let A be the set of labels assigned to the vertices of the first cycle.

$$A = \{\ a_i = f(u_i),\ \text{for } 1 \leq i \leq n\}$$

2. Let B be the set of labels assigned to the vertices of the second cycle.

$$B = \{\ b_i = f(v_i),\ \text{for } 1 \leq i \leq n\}$$

3. Let $C = V(mK_1)$, where $m = \varepsilon(C_n \Box K_2)$, be the set of labels assigned to the work-

ing vertices $w_1, w_2, \ldots, w_m$

$$C = \{\ c_i = f(w_i),\ \text{for } 1 \leq i \leq m\}$$

We know that $(A \cup B) \cap C = \emptyset$. Since the set A and B contains odd labels, and the set C contains even labels, one can easily verify that there are no additional edges between the vertices by considering the following:

(i) $a_i + a_j \neq a_k$, for any $a_i, a_j, a_k \in A$

(ii) $b_i + b_j \neq b_k$, for any $b_i, b_j, b_k \in B$

(iii) $a_i + c_i \neq c_j$, for any $a_i \in A$, $c_i, c_j \in C$

(iv) $b_i + c_i \neq c_j$, for any $b_i \in B$, $c_i, c_j \in C$

From (2.2.1), (2.2.2),(2.2.3) and (2.2.4), we have

$$\begin{aligned} &a_i \equiv k(\ mod\ d),\ \forall\ i \in \{1, 2, \ldots, n\} \text{ and } a_i \in A, \\ &b_i \equiv k(\ mod\ d),\ \forall\ i \in \{1, 2, \ldots, n\} \text{ and } b_i \in B, \\ &c_i \equiv 2k(\ mod\ d),\ \forall\ i \in \{1, 2, \ldots, m\} \text{ and } c_i \in C, \end{aligned}$$

and since $k < d$ and $k,\ d$ are relatively prime, it is easy to verify the following:

(v) $\{a_i + a_j\} \subseteq C$, $\{b_i + b_j\} \subseteq C$ and $\{a_i + b_i\} \subseteq C$

(vi) $a_i + c_j \neq a_k$, for any $a_i, a_k \in A$, $c_j \in C$

(vii) $b_i + c_j \neq b_k$, for any $b_i, b_k \in B$, $c_j \in C$

(viii) $a_i + c_j \neq b_k$, for any $a_i \in A$, $c_j \in C$, and $b_k \in B$

(ix) $b_i + c_j \neq a_k$, for any $b_i \in B$, $c_j \in C$, and $a_k \in A$

(x) $c_i + c_j \neq c_k$, for any $c_i, c_j, c_k \in C$

We now arrange the labels as an arithmetic sequence with the common difference $d^{'} = |(a_i - b_{i+1})|$ for $1 \leq i \leq n-2$ as follows: when n is odd

$$a_1 < b_2 < a_3 < b_4 < \ldots < a_{n-2} < b_{n-1} < a_{n-1} < b_{n-2} < \ldots < a_4 < b_3 < a_2 < b_1 \quad (2.2.5)$$

when n is even

$$a_1 < b_2 < a_3 < b_4 < \ldots < b_{n-2} < a_{n-1} < b_{n-1} < a_{n-2} < \ldots < a_4 < b_3 < a_2 < b_1 \quad (2.2.6)$$

We now prove that $\varepsilon(C_n \square K_2) = 5$. Since $\Delta(C_n \square K_2) = 3$, let us assume that $\varepsilon(C_n \square K_2) \geq 3$. Let c_1, c_2 and c_3 be the three isolated vertices. Now from (2.2.5) and (2.2.6), we have

(i) $a_i + b_i = c_1$, for $1 \leq i \leq n-1$

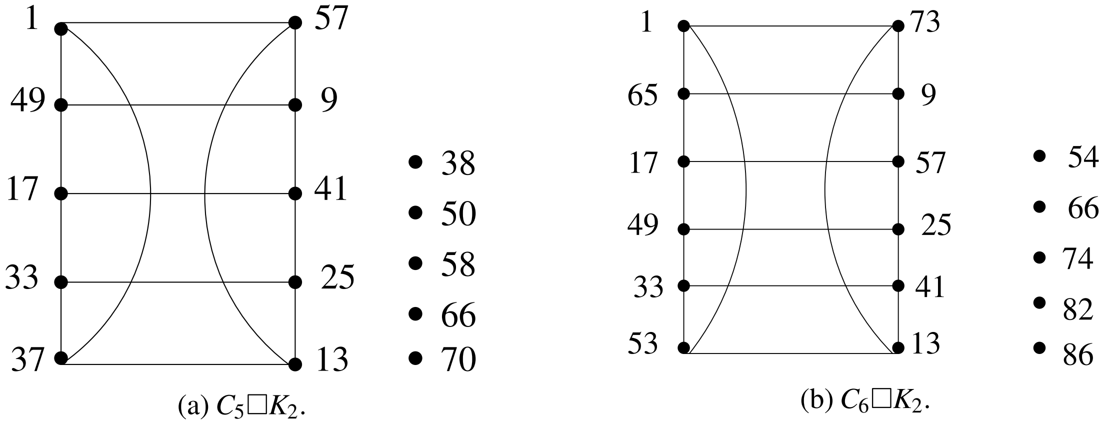
(ii) $a_i + a_{i+1} = b_{i+1} + b_{i+2} = c_2$, for $1 \leq i \leq n-3$

(iii) $b_i + b_{i+1} = a_{i+1} + a_{i+2} = c_3$, for $1 \leq i \leq n-3$

(iv) $a_i + a_{i+1} = a_n + b_n = c_2$ or $b_i + b_{i+1} = a_n + b_n = c_3$ for $1 \leq i \leq n-3$

The remaining adjacent vertices are $(a_1, a_n), (a_{n-1}, a_n), (b_{n-1}, b_n), (b_1, b_n)$. From the labeling we have $a_1 + a_n \neq a_{n-1} + a_n$ and $b_1 + b_n \neq b_{n-1} + b_n$. Further $a_1 + a_n = b_{n-1} + b_n$ and $b_1 + b_n = a_{n-1} + a_n$, resulting in two more distinct sums. This implies that three isolated vertices are not sufficient, thus contradicting our assumption. Hence, we conclude that, a total of five isolated vertices are required. Therefore, the exclusive sum number of $\varepsilon(C_n \square K_2) = 5$. □

Figure 2.5 illustrates the exclusive sum labeling of $C_5 \square K_2$ and $C_6 \square K_2$ when $k = 1, d = 4$.



(a) $C_5 \square K_2$.

(b) $C_6 \square K_2$.

Figure 2.5 Exclusive Sum labeling of $C_5 \square K_2$ and $C_6 \square K_2$, when $k = 1, d = 4$.

2.3 EXCLUSIVE SUM NUMBER OF $K_n \square K_2$

The following theorem gives the exclusive sum number of the graph $K_n \square K_2$, for $n \geq 3$.

Theorem 2.3.1. *The exclusive sum number for $\varepsilon(K_n \square K_2) = (2n-1)$, for $n \geq 3$.*

Proof. Consider the Cartesian product of complete graph K_n with K_2. The vertices of the two complete graphs are $u_1, u_2, ..., u_n$ and $v_1, v_2, ..., v_n$ respectively. Let $k \geq 1$ and odd, $d \geq 4$ and even, where $k < d$ and are co-prime. The labeling $f : V(G) \rightarrow N$ is defined as follows:

$$f(u_i) = k + 2(i-1)d, \text{ for } 1 \leq i \leq n \tag{2.3.1}$$

and

$$f(v_i) = k + (2(n-i)+1)d, \text{ for } 1 \leq i \leq n \tag{2.3.2}$$

The working vertices receive the labels as: $2k+2d$, $2k+4d$, $2k+6d$,..., $2k+2(2n-2)d$ and $2k+(2n-1)d$. Now consider the following:

1. Let A be the set of labels assigned to the vertices of the first complete graph.

$$A = \{\ a_i = f(u_i), \text{ for } 1 \leq i \leq n\}$$

2. Let B be the set of labels assigned to the vertices of the second complete graph.

$$B = \{\ b_i = f(v_i), \text{ for } 1 \leq i \leq n\}$$

3. Let $C = V(mK_1)$, where $m = \varepsilon(K_n \Box K_2)$, be the set of labels assigned to the working vertices $w_1, w_2, \ldots, w_m$

$$C = \{\ c_i = f(w_i), \text{ for } 1 \leq i \leq m\}$$

We know that $(A \cup B) \cap C = \emptyset$. Since the set A and B contains odd labels, and the set C contains even labels, one can verify that there are no additional edges between the vertices by considering the following:

(i) $a_i + a_j \neq a_k$, for any $a_i, a_j, a_k \in A$

(ii) $b_i + b_j \neq b_k$, for any $b_i, b_j, b_k \in B$

(iii) $a_i + c_i \neq c_j$, for any $a_i \in A$, $c_i, c_j \in C$

(iv) $b_i + c_i \neq c_j$, for any $b_i \in B$, $c_i, c_j \in C$

From (2.3.1), (2.3.2), we have

$$a_i \equiv k(\ mod\ d),\ \forall\ i \in \{1, 2, \ldots, n\} \text{ and } a_i \in A,$$
$$b_i \equiv k(\ mod\ d),\ \forall\ i \in \{1, 2, \ldots, n\} \text{ and } b_i \in B,$$
$$c_i \equiv 2k(\ mod\ d),\ \forall\ i \in \{1, 2, \ldots, m\} \text{ and } c_i \in C,$$

and since $k < d$ and $k,\ d$ are relatively prime, one can verify the following:

(v) $\{a_i + a_j\} \subseteq C$, $\{b_i + b_j\} \subseteq C$ and $\{a_i + b_i\} \subseteq C$

(vi) $a_i + c_j \neq a_k$, for any $a_i, a_k \in A$, $c_j \in C$

(vii) $b_i + c_j \neq b_k$, for any $b_i, b_k \in B$, $c_j \in C$

(viii) $a_i + c_j \neq b_k$, for any $a_i \in A$, $c_j \in C$, and $b_k \in B$

(ix) $b_i + c_j \neq a_k$, for any $b_i \in B$, $c_j \in C$, and $a_k \in A$

(x) $c_i + c_j \neq c_k$, for any $c_i, c_j, c_k \in C$

We now arrange the labels as an arithmetic sequence with the common difference $d^{'} = |(a_i - b_{n-i+1})|$ as follows:

$$a_1 < b_n < a_2 < b_{n-1} < \ldots < a_{n-1} < b_2 < a_n < b_1 \tag{2.3.3}$$

From (2.3.3), the following can be verified:

(i) In the graph $K_n \Box K_2$, smallest label is adjacent to $(n-1)$ other labels and the largest label is also adjacent to $(n-1)$ other labels. Therefore, there will be at least $(n-1)+(n-1) = 2n-2$ distinct sums, resulting in a total of $2n-2$ isolated vertices.

(ii) It can be observed from (2.3.3), that the following sums are equal,

$$a_1 + b_1 = a_2 + b_2 = \cdots = a_{n-1} + b_{n-1} = a_n + b_n$$

which results in another working vertex. Hence the Cartesian product $K_n \Box K_2$ has the total number of working vertices given by $(2n-2)+1 = (2n-1)$.

Therefore, we conclude that, the exclusive sum number of $\varepsilon(K_n \Box K_2) \leq (2n-1)$ □

Figure 2.6 illustrates the exclusive sum labeling of $K_5 \Box K_2$.

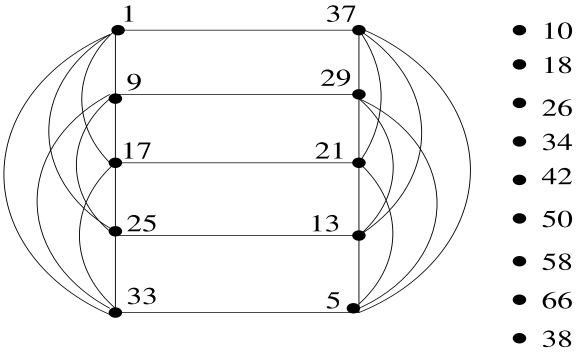


Figure 2.6 Exclusive Sum labeling of $K_5 \Box K_2$ when $k = 1, d = 4$.

From the above labeling f, it follows that the exclusive sum number for the disjoint union of two copies of the complete graph K_n is $\varepsilon(K_n \cup K_n) \leq (2n-2)$, for $n \geq 3$.

2.4 EXCLUSIVE SUM NUMBER OF kP_n AND kC_n

In this section we consider the exclusive sum labeling of kP_n and kC_n. Consider the disjoint union of k copies of path P_n, denoted by kP_n.

Theorem 2.4.1. *The exclusive sum number for disjoint union of k copies of paths is* $\varepsilon(kP_n) = 2$, *for* $n \geq 4$.

Proof. Let $v_1^j, v_2^j, \dots, v_n^j$ be the vertices of j^{th} copy of the graph P_n in kP_n. The labeling $f : V(G) \to N$ is defined as follows:

Let $m = 2nk$ and $j \in \{1, 2, \dots, k\}$.

When n is odd,

$$f(v_i^j) = \begin{cases} 4j-3, \text{ for } i = 1 \\ m + f(v_{i-2}^j), \text{ for odd } i, \ 3 \leq i \leq n \\ m + f(v_{i+2}^j), \text{ for even } \ i, \ 2 \leq i \leq n-3 \\ m - 8(j-1) + f(v_n^j), \text{ for } i = n-1 \end{cases} \tag{2.4.1}$$

When n is even,

$$f(v_i^j) = \begin{cases} 4j-3, \text{ for } i = 1 \\ m + f(v_{i-2}^j), \text{ for odd } i, \ 3 \leq i \leq n-1 \\ m + f(v_{i+2}^j), \text{ for even } \ i, \ 2 \leq i \leq n-2 \\ m - 8(j-1) + f(v_{n-1}^j), \text{ for } i = n \end{cases} \tag{2.4.2}$$

The labeling function f assigns distinct labels to the vertices of every copy of P_n in kP_n in such a way that, the two working vertex labels $f(w_1)$ and $f(w_2)$ for $j \in \{1,2,\ldots,k\}$ are given by the following

$$\begin{aligned} &f(w_1) = f(v_1^j) + f(v_2^j), \text{ and} \\ &f(w_2) = f(v_2^j) + f(v_3^j) \end{aligned}$$

We consider the following:

1. Let A be the set of labels assigned to the non-working vertices of kP_n.

$$A = \{ \ a_i^j = f(v_i^j), \text{ for } 1 \le i \le n, \ 1 \le j \le k\}$$

2. Let $C = V(2K_1)$ be the set of labels assigned to the working vertices w_1, w_2

$$C = \{ \ c_i = f(w_i), \text{ for } 1 \le i \le 2\}$$

We know that $A \cap C = \emptyset$. The set A consists of odd labels which are congruent to the following

$$a_i^j \equiv \begin{cases} 1 \ (mod \ m), \text{ for } j = 1 \text{ and } \ 1 \le i \le n \\ 4j-3 \ (mod \ m), \text{ for } j \in \{2,\ldots,k\}, \text{ and odd } i, \ 1 \le i \le n \\ m-4j+5 \ (mod \ m), \text{ for } j \in \{2,\ldots,k\}, \text{ and even } i, \ 1 \le i \le n. \end{cases} \tag{2.4.3}$$

and the set C consists of even labels which are congruent to the following

$$c_i \equiv 2 \ (mod \ m), \text{ for } 1 \le i \le 2 \tag{2.4.4}$$

Since the set A contains odd labels, and the set C contains even labels, one can easily verify that there are no additional edges between the vertices by considering the following:

(i) $a_x^j + a_y^j \neq a_z^j$, for any $a_x^j, a_y^j, a_z^j \in A$

(ii) $a_x^j + c_i \neq c_j$, for any $a_x^j \in A$, $c_i, c_j \in C$

(iii) $c_i + c_j \neq a_x^j$, for any $c_i, c_j \in C$, $a_x^j \in A$

(iv) From (2.4.4), we have $c_i + c_j \neq c_k$, because for any $c_i, c_j, c_k \in C$,

$$c_i + c_j \equiv 4 \ (mod \ m) \tag{2.4.5}$$

results in the label that is not an element of set C.

(v) From (2.4.3) and (2.4.4), it is clear that $a_x^j + c_i \neq a_y^j$, because for any for any $a_x^j, a_y^j \in A$, $j \in \{1,2,\ldots,k\}$ and $c_i \in C$,

$$a_x^j + c_i \equiv \begin{cases} 3\ (mod\ m), \\ 4j-1(mod\ m), \\ m-4j+7(mod\ m) \end{cases} \tag{2.4.6}$$

and (2.4.6) generates labels that are not members of the set A, i.e., any of the non-working vertices in the graph do not receive these labels. Therefore, we conclude that there are no edges between the working vertex labels and non-working vertex labels and hence, the exclusive sum labeling for $\varepsilon(kP_n)$ = 2. □

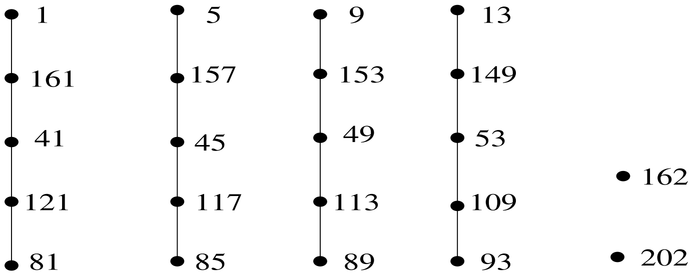


Figure 2.7 Exclusive Sum labeling of kP_n when $k = 4$, $n = 5$.

Figure 2.7 illustrates the exclusive sum labeling of kP_n when $k = 4$ and $n = 5$. Consider the disjoint union of k copies of even cycles, denoted by kC_n.

Theorem 2.4.2. *The exclusive sum number for disjoint union of k copies of cycles C_n, where $n \geq 4$ and even, is $\varepsilon(kC_n) = 3$.*

Proof. Let $v_1^j, v_2^j, \ldots, v_n^j$ be the vertices of j^{th} copy of the graph C_n in kC_n. Let $m = 2nk$ and $j \in \{1,2,\ldots,k\}$ and the labeling $f : V(G) \to N$ is defined as follows:

$$f(v_i^j) = \begin{cases} 4j-1, \text{ for } i = 1 \\ m + f(v_{i-2}^j), \text{ for odd } i,\ 3 \leq i \leq n-1 \\ m + f(v_{i+2}^j), \text{ for even } \ i,\ 2 \leq i \leq n-2 \\ m - 8j + 4 + f(v_{n-1}^j), \text{ for } i = n \end{cases} \tag{2.4.7}$$

The labeling function f assigns distinct labels to the vertices of every copy of C_n in kC_n in such a way that, the three working vertex labels $f(w_1)$, $f(w_2)$ and $f(w_3)$ for $j \in \{1,2,\ldots,k\}$ are given by the following:

$$\begin{aligned}
&f(w_1) = f(v_1^j) + f(v_n^j),\\
&f(w_2) = f(v_1^j) + f(v_2^j) \text{ and}\\
&f(w_3) = f(v_2^j) + f(v_3^j)
\end{aligned}$$

We consider the following:

1. Let A represent the set of labels assigned to the vertices of kC_n.

$$A = \{\ a_i^j = f(v_i^j), \text{ for } 1 \le i \le n,\ 1 \le j \le k\}$$

2. Let $C = V(3K_1)$ be the set of labels assigned to the working vertices w_1, w_2, w_3

$$C = \{\ c_i = f(w_i), \text{ for } 1 \le i \le 3\}$$

We know that $A \cap C = \emptyset$. The set A consists of odd labels which are congruent to the following

$$a_i^j \equiv \begin{cases} 4j-1\ (mod\ m), \text{ for } j \in \{1,\ldots,k\}, \text{ and odd } i,\ 1 \le i \le n \\ m-4j+3\ (mod\ m), \text{ for } j \in \{1,\ldots,k\}, \text{ and even } i,\ 1 \le i \le n \end{cases} \tag{2.4.8}$$

and the set C consists of even labels which are congruent to the following

$$c_i \equiv 2\ (mod\ m), \text{ for } 1 \le i \le 3 \tag{2.4.9}$$

Since the set A contains odd labels, and the set C includes even labels, one can easily verify that there are no additional edges between the vertices by considering the following:

(i) $a_x^j + a_y^j \neq a_z^j$, for any $a_x^j, a_y^j, a_z^j \in A$

(ii) $a_x^j + c_i \neq c_j$, for any $a_x^j \in A$, $c_i, c_j \in C$

(iii) $c_i + c_j \neq a_x^j$, for any $c_i, c_j \in C$, $a_x^j \in A$

(iv) From (2.4.9), we have $c_i + c_j \neq c_k$, because for any $c_i, c_j, c_k \in C$,

$$c_i + c_j \equiv 4\ (mod\ m) \tag{2.4.10}$$

results in a label that is not an element of set C.

(v) From (2.4.8) and (2.4.9), it is clear that $a_x^j + c_i \neq a_y^j$, because for any for any $a_x^j, a_y^j \in A$, $j \in \{1,2,\ldots,k\}$ and $c_i \in C$,

$$a_x^j + c_i \equiv \begin{cases} 4j+1(mod\ m), \\ m-4j+5(mod\ m) \end{cases} \tag{2.4.11}$$

and (2.4.11) generates labels that are not members of the set A, i.e., any of the non-working vertices in the graph do not receive these labels. Therefore, we conclude that there are no edges between the working vertex labels and non-working vertex labels and hence, the exclusive sum labeling for $\varepsilon(kC_n)$ = 3 □

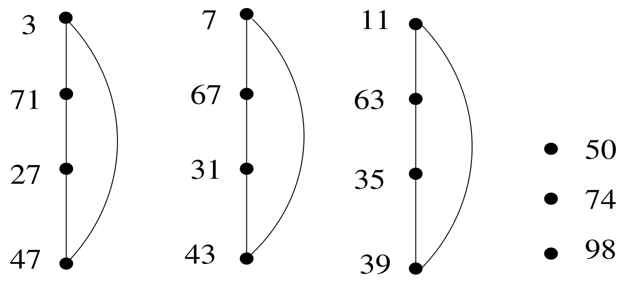


Figure 2.8 Exclusive Sum labeling of kC_n when $k = 3$, $n = 4$.

Figure 2.8 illustrates the exclusive sum labeling of kC_n when $k = 3$ and $n = 4$.

CHAPTER 3

APPLICATIONS OF EXCLUSIVE SUM LABELING IN RELATIONAL DATABASES

In this chapter we present the exclusive sum labeling complete k-partite graph $K_{r_1,r_2,}$ $\dots,r_k$. Sutton (2000) proposed that the sum labeled multipartite graphs could serve as a model for storing links in a relational database. We show that the exclusive sum labeling of complete k-partite graph $K_{r_1,r_2,\dots,r_k}$, can be used to model links in the relational database.

The theorem below provides an exclusive sum labeling for the complete k-partite graph $K_{r_1,r_2,\dots,r_k}$, where $r_1,r_2,\dots,r_k$ is the sequence of the sizes of each set in the k partition and $r_1 < r_2 < \cdots < r_k$ with $n = r_1+r_2+\dots+r_k$. The vertex set V is partitioned into k independent sets $V_1,V_2,\dots,V_k$, where $V_i = \{v_{i1},v_{i2},\dots,v_{ir_i}\}$, $1 \le i \le k$, such that $v_{ix}v_{jy} \in E$ for all $i,j \in \{1,2,...,k\}$, $i \neq j$ and $x \in \{1,2,...,r_i\}$, $y \in \{1,2,...,r_j\}$ and $|V_i| \neq |V_j|$. The idea of exclusive sum labeling of the complete k-partite graph can be used to store and manipulate the links inside the relational database.

3.1 EXCLUSIVE SUM NUMBER OF COMPLETE k-PARTITE GRAPH $K_{r_1,r_2,\dots,r_k}$

Theorem 3.1.1. The exclusive sum number for the complete k-partite graph $K_{r_1,r_2,\dots,r_k}$, where $r_1 < r_2 < \cdots < r_k$ and $n = r_1+r_2+\dots+r_k$ is $\varepsilon(K_{r_1,r_2,\dots,r_k}) = \frac{(2n-k)(k-1)}{2}$.

Proof. Let f be any exclusive sum labeling of a complete k-partite graph $K_{r_1,r_2,\dots,r_k}$, where $r_1 < r_2 < \cdots < r_k$ and $n = r_1+r_2+\dots+r_k$. Let $V_1,V_2,\dots,V_k$, where $|V_i| = r_i$ where $1 \le i \le k$ be the vertex sets. Suppose that the labels of $V_i = \{v_{i1},v_{i2},\dots,v_{ir_i}\}$, for $1 \le i \le k$ are arranged into an ascending sequence, so that $v_{i1} < v_{i2}$, for $1 \le i \le k$. Then one can observe for any two V_i and V_j, $i,j \in \{1,2,\dots,k\}$ each of the following sums is distinct.

$$v_{i1}+v_{j1} < v_{i2}+v_{j1} < v_{i3}+v_{j1} < ... < v_{ir_i}+v_{j1} < v_{ir_i}+v_{j2} < ... < v_{ir_i}+v_{jr_j}$$

Since there are exactly $i+j-1$ distinct sums, it follows that for any two vertex sets, a total of $i+j-1$ distinct sums are produced and we have $\binom{k}{2}$ such pairs of vertex sets resulting in $\frac{(2n-k)(k-1)}{2}$ distinct sums. Hence it follows that atleast $\frac{(2n-k)(k-1)}{2}$ isolated vertices are required to label the graph exclusively, i.e. $\varepsilon(K_{r_1,r_2,...,r_k}) \geq \frac{(2n-k)(k-1)}{2}$.

Let $M = 3^k + 1$, $|V_i| = r_i$, for all $i \in \{1,2,...,k\}$ The labeling $f : V(G) \to N$ is defined as follows:

$$f(v_{ix}) = (|V_{i-1}|^2 + x)M + 2(3^{i-1}) + 1, \text{ where } 1 \leq i \leq k,\ 1 \leq x \leq r_i,\ |V_0| = 0 \quad (3.1.1)$$

Let $A = A_1 \cup A_2 \cup ... \cup A_k$ be the set of labels of all the non-working vertices. Here A_i is the set of labels assigned to the non-working vertices of the i^{th} partite set V_i as

$$A_i = \{a_{ix} = f(v_{ix}), \text{ where } 1 \leq i \leq k, 1 \leq x \leq r_i\} \text{ and } |V_i| = |A_i|$$

The labels in any set A_i, where $A_i \in A$ are congruent to the following:

$$a_{ix} \equiv 2(3^{i-1}) + 1\ (mod\ M) \quad (3.1.2)$$

and it is clear that for any $i,\ j \in \{1,2,...,k\}, i \neq j, A_i \cap A_j = \emptyset$. The working vertex labels represented by the set $W = \{w_1, w_2, ..., w_m\}$, $m = \varepsilon(K_{r_1,r_2,...,r_k})$ are given as follows: For $l \in \{1,2,...,m\}$,

$$W = \{w_l = a_{ix} + a_{jy}, \text{ for } i,\ j \in \{1,2,...,k\},\ i \neq j,\ x \in \{1,2,...,r_i\},\ y \in \{1,2,...,r_j\}\}$$

and labels in W are congruent to the following:

$$w_l \equiv 2(3^{i-1}) + 2(3^{j-1}) + 2\ (mod\ M) \quad (3.1.3)$$

The labels from any set A_i of the bipartite set V_i where $i \in \{1,2,...,k\}$ is of the form:

$$a_{ix} = a_{i1} + M(x-1), \text{ for } 1 \leq i \leq k,\ 2 \leq x \leq r_i.$$

The labels from any two sets say A_i, A_j where $i < j$, $i, j \in \{1,2,...,k\}$ can be arranged as an arithmetic sequence with the common difference M as follows:

$$\begin{aligned} &a_{i1} < a_{i2} < a_{i3} ... < a_{ir_i} < a_{j1} < a_{j2} < a_{j3} < ... < a_{jr_j} = a_{i1} < (a_{i1} + M) < ... \\ &< (a_{i1} + (r_i - 1)M) < a_{j1} < (a_{j1} + M) < ... < (a_{j1} + (r_j - 1)M) \end{aligned} \tag{3.1.4}$$

From (3.1.4), the smallest vertex label say $a_{i1} \in A_i$ is adjacent to all the vertex labels in A_j, resulting in $|V_j|$ distinct sums. Also the remaining $(r_i - 1)$ vertex labels in A_i on being adjacent to the largest vertex label in A_j, produces $|V_i| - 1$ distinct sums. Therefore any two vertex sets V_i ,V_j where $i < j$, $i, j \in \{1, 2, ..., k\}$, results in $|V_j| + |V_i| - 1$ distinct sums. On solving for $\binom{k}{2}$ such pairs of vertex sets, the labeling results in the number of working vertices $|W|$ given as follows:

$$|W| = n(k-1) - \binom{k}{2} = \frac{(2n-k)(k-1)}{2}, \text{ where } n = r_1 + r_2 + ... + r_k$$

From (3.1.2) and (3.1.3) it can be observed that the set $A = A_1 \cup A_2 \cup ... \cup A_k$ consists of odd labels and the set W consists of even labels only. Let $S = A \cup W$ and $A \cap W = \emptyset$. Now one can verify that there are no new edges between two non-working vertices belonging to the same partite set, or between the non-working vertices and the working vertices or between two working vertices by considering the following:

(i) $a_{ix} + a_{iy} \neq S$, for any $a_{ix}, a_{iy} \in A_i$

(ii) $a_{ix} + w_l \neq S$, for any $a_{ix} \in A_i$, $w_l \in W$

(iii) $w_l + w_p \neq S$, for any $w_l, w_p \in W$

Thus, we know that the above labeling is an exclusive sum labeling of $K_{r_1,r_2,...,r_k}$. Hence $\varepsilon(K_{r_1,r_2,...,r_k}) \leq \frac{(2n-k)(k-1)}{2}$. Therefore, $\varepsilon(K_{r_1,r_2,...,r_k}) = \frac{(2n-k)(k-1)}{2}$, where $n = r_1 + r_2 + ... + r_k$. □

Figure 3.1 illustrates the exclusive sum labeling of $K_{2,3,4}$.

3.2 APPLICATION OF EXCLUSIVE SUM LABELINGS IN RELATIONAL DATABASES

A relational database is entirely comprised of tables. A table is a collection (or, more formally, a set) of information about similar 'things,' such as students, faculties, and courses. A table can only hold information about one sort of thing at a time, therefore all information about faculties is stored in a faculty table, all information about students

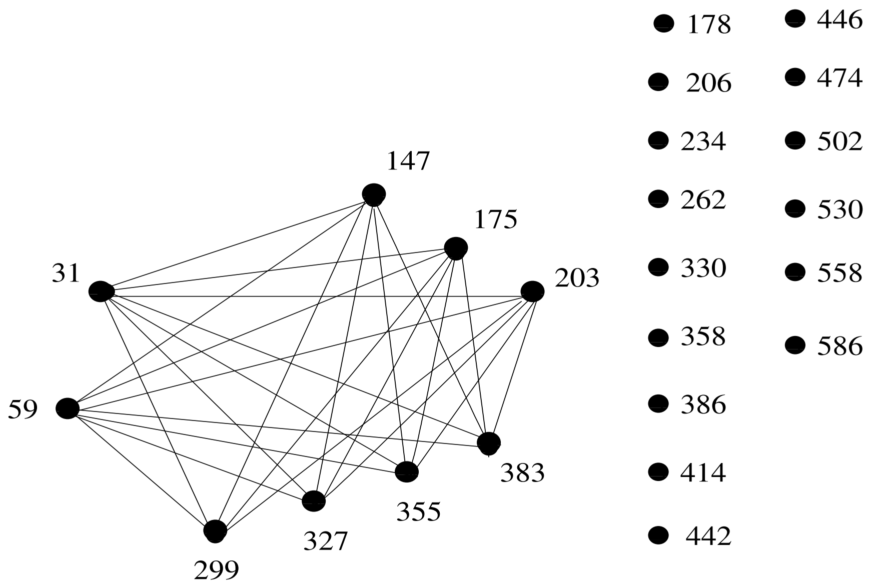
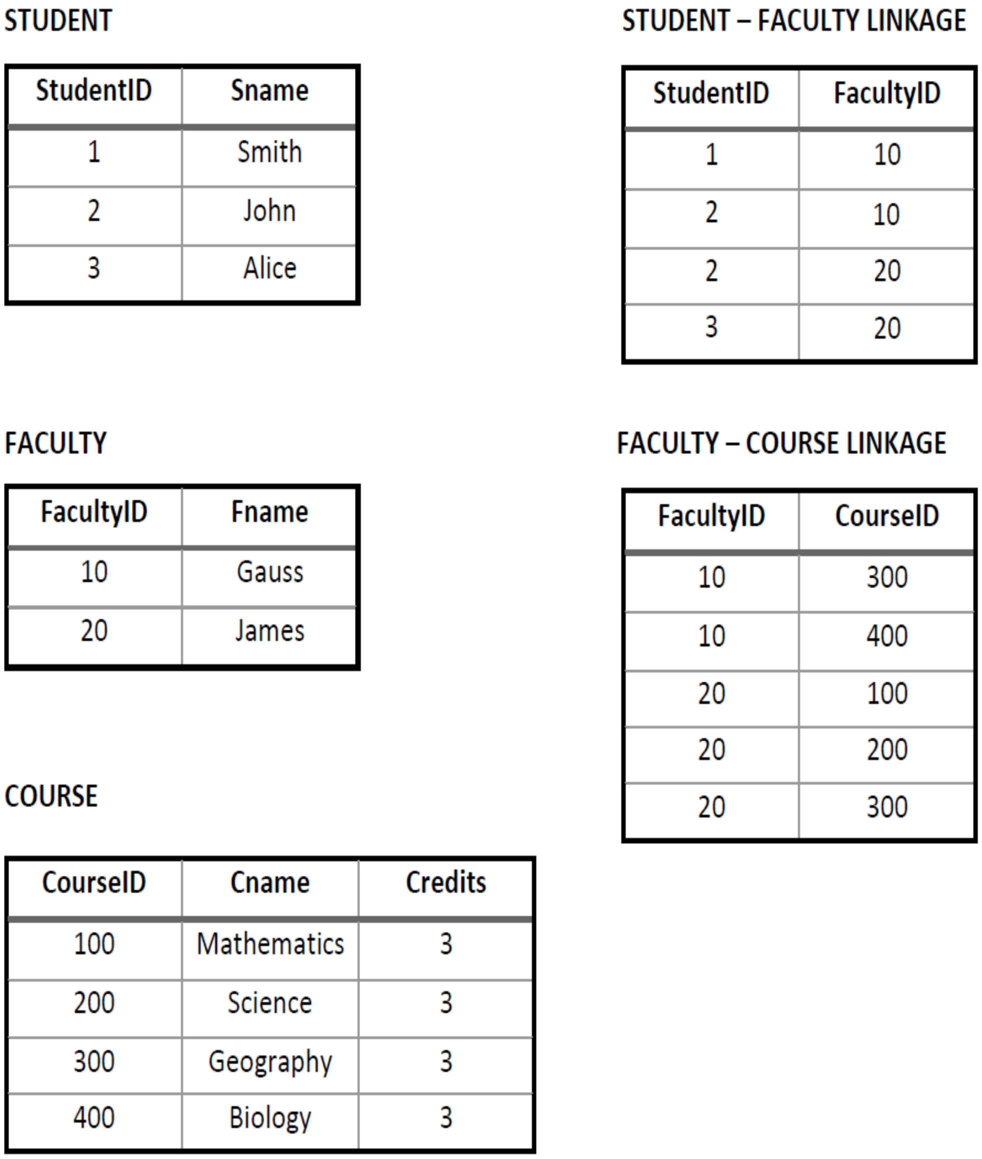


Figure 3.1 Exclusive Sum labeling of $K_{2,3,4}$.

is stored in an student table, and so on. Each row in a table contains data on a single instance of a thing. In a Student table, each separate row contains information about a specific student. More than one student's information is never kept in a single row. There is no duplication and the rows are not ordered since the rows of a table are data instances in a set. The relationship between two rows belonging to two separate tables is called links. The term linkage refers to the link stored in the database. Each database would need some mechanism to save every single link as data inside the database. During query processing, these saved links(linkage) are used to recreate the links in the database. Within the database, the operations such as insertion, deletion, and updating of each link must remain valid.

Consider the example of the Student-Faculty database that stores the information about the Student, Faculty, and the Courses given in Figure 3.2. The STUDENT-FACULTY Linkage table and the FACULTY-COURSE Linkage table highlight the relationship between the STUDENT, FACULTY, and COURSE tables. Sutton (2000) proposed that sum labeled multipartite graphs could serve as a model for storing the links in a relational database. A partite set represents each table, and a vertex within the partite set represents each row in that table. The edges in the graph reflect the connections(links) between the rows of individual tables. Isolates are present as an artifact of the process for storing linkages. The idea is to have one more positive integer in every table row. This additional integer refers to the label given to the vertex that belongs to the partite set representing the table. The storage overhead of this system is the number of isolated vertices(working vertices) created during the sum labeling of the multipartite



STUDENT

StudentID	Sname
1	Smith
2	John
3	Alice

STUDENT – FACULTY LINKAGE

StudentID	FacultyID
1	10
2	10
2	20
3	20

FACULTY

FacultyID	Fname
10	Gauss
20	James

FACULTY – COURSE LINKAGE

FacultyID	CourseID
10	300
10	400
20	100
20	200
20	300

COURSE

CourseID	Cname	Credits
100	Mathematics	3
200	Science	3
300	Geography	3
400	Biology	3

Figure 3.2 The Student-Faculty Database, with the tables on the right representing the relationships between the three tables STUDENT, FACULTY and COURSE.

graph.

When a multipartite graph $H_{m,n}$ models the relational database as stated by Sutton (2000), the vertices are partitioned into n sets each of fixed-size m, imposing a fixed number of rows in each table in the database. Practically, this may not be the case, as the number of rows in a table can vary. Therefore, it is more appropriate to consider the complete k-partite graph where each partition size can differ. In this case, each partition size varies, allowing the number of rows in the tables to vary. If the multipartite graph $H_{m,n}$ modeled the database, then the partition size will have to be fixed to the largest number of rows among the tables. As this would result in some unused vertex labels, it is more appropriate to use the complete k-partite graph to model the relational database.

The complete 3-partite graph $K_{2,3,4}$ in Figure 3.3 shows the relationship between the three tables in the database. The FACULTY table maps to the partite set of size two, the STUDENT maps to the partite set of size three, and the COURSE table maps to the partite set of size four. As given in the linkage tables, only those edges that represent the relationships are highlighted here. The links generate the isolates or the working vertex labels, as provided in Figure 3.3.

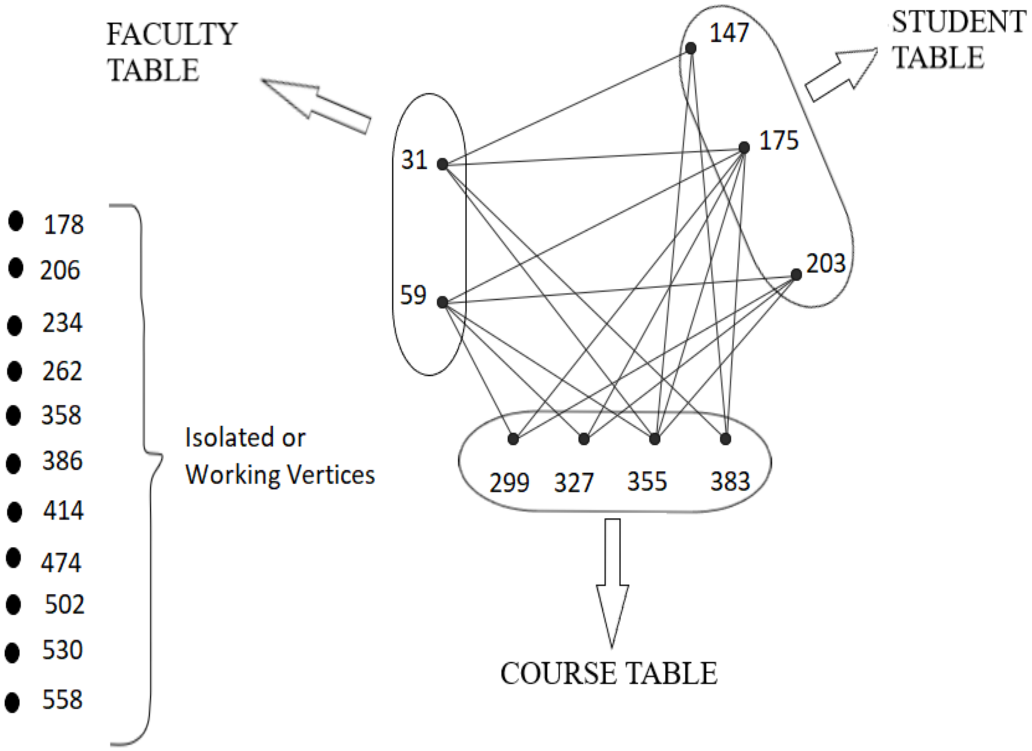


Figure 3.3 Exclusive Sum labeling of $K_{2,3,4}$ used to model Student-Faculty Database.

From Theorem 3.1.1, the number of isolated vertices required to obtain an exclusively sum labeled complete k-partite graph, $K_{r_1,r_2,\ldots,r_k} = \frac{(2n-k)(k-1)}{2}$. Therefore, all the links that occur in the database can be stored by an exclusive sum labeled complete k-partite graph, $K_{r_1,r_2,\ldots,r_k}$, with $\frac{(2n-k)(k-1)}{2}$ number of isolated vertices.

CHAPTER 4

MAXIMUM INDEPENDENT SETS FROM THE SIGNATURES OF PROPER MONOGRAPHS

Since the introduction of autographs, several authors have investigated which graphs are autographs. Fontanil and Panopio (2014) observed that the independent set and vertex covering of a graph could be derived from the signatures of a proper monograph. Further, it was observed by Fontanil and Panopio (2014) that not all proper monographs could have their maximum independent sets derived from the sets of idle vertices. Hence, in this chapter, we give the proper monograph labelings of several graphs for which the set of idle vertices gives the corresponding maximum independent set of the graphs. We present the proper monograph labelings of classes of graphs such as cycles, $C_n \bigodot K_1$, cycles with paths attached to one or more vertices, and cycles with an irreducible tree attached to one or more vertices. We show that the signatures of these proper monographs determine the maximum independent sets of these graphs.

4.1 PRELIMINARIES

Several definitions and theorems that will be used in the remainder of this chapter are given in this section. Consider G to be a graph with $V = \{v_1, v_2, \ldots, v_n\}$ vertices and $E = \{[v_i, v_j] | \text{ for some } v_i, v_j \in V\}$ edges. Let $S = \{s_1, s_2, \ldots, s_n\}$ be the set of signature values assigned to the vertices of autograph G. Here s_i is the label given to vertex v_i, for $i = 1, 2, ..., n$. Note that the vertices are represented by their corresponding signature values. For G to be an autograph with labels assigned from S, it must satisfy the following: for any $s_i, s_j \in S$, if s_i is adjacent to s_j, then: $|s_i - s_j| \in S$ and the corresponding vertices v_i and v_j are neighbors in G.

Definition 4.1.1. *The corona product of two graphs G_1 and G_2, denoted by $G_1 \bigodot G_2$, is obtained by taking one copy of G_1 along with $|V(G_1)|$ copies of G_2 and adding edges to make the i^{th} vertex of G_1 adjacent to every vertex of the i^{th} copy of G_2, where $1 \leq i \leq |V(G_1)|$. The corona product $C_n \bigodot K_1$ is obtained by attaching one pendant vertex to every vertex of C_n.*

Some of the terminologies from Bloom et al. (1979) are given here. A node of degree one in a tree T is called an *endnode*. The nodes with a degree at least 3 in a tree T are called *branchnodes*. The nodes with degree 2 are termed *limb* nodes. Consider a k-length sequence of consecutive adjacent nodes $u_0, u_1, u_2, \ldots, u_k$. The sequence is called a *limb* if the adjacent nodes in the sequence satisfy the following: $deg(u_0) \geq 3$, $deg(u_1) = deg(u_2) = ... = deg(u_{k-1}) = 2$, and $deg(u_k) = 1$.

According to (Theorem 1, Bloom et al. (1979)), every tree T is a proper monograph. The theorem was proved using the following lemmas describing the class of irreducible trees. If no node is adjacent to more than one node of degree 1, then the tree is termed as an irreducible tree.

Lemma 4.1.2. *(*Bloom et al. (1979)*) A limb of a tree T that is irreducible has length greater than one.*

Lemma 4.1.3. *(*Bloom et al. (1979)*) If a tree T is an irreducible tree, then it is a monograph.*

If the elements of the set I are pairwise non-adjacent vertices of graph G, then set I is called an independent set of G. The set I is maximal if it is not contained as a proper subset by another independent set. The notion of idle and working vertices was adapted to determine the vertex coverings and the independent sets in a proper monograph by Fontanil and Panopio (2014).

Definition 4.1.4. *(*Fontanil and Panopio (2014)*) Consider $G(S)$ as an autograph G with signature S. A vertex $s \in G(S)$ is termed as a working vertex if the following is satisfied: there exists $s_i, s_j \in S$, such that $|s_i - s_j| = s$. If $|s_i - s_j| \neq s$ for $s_i, s_j \in S$, then s is an idle vertex.*

Fontanil and Panopio (2014) investigated the relations between the idle vertices and the independent sets of a proper monograph in the following theorem.

Theorem 4.1.5. *(*Fontanil and Panopio (2014)*) In an autograph G whose signature is given by the set of positive elements S, the independent set is given by the set of positive idle vertices of $G(S)$.*

If the set of vertices $I \subset V$ gives the independent set of G, then the set of vertices $V' = V \setminus I$ gives the vertex covering of G.

Corollary 4.1.6. *(*Fontanil and Panopio (2014)*) In an autograph G whose signature is given by the set of positive elements S, the set of vertex cover is given by the set of working vertices of $G(S)$.*

Furthermore, it was observed that the property stating that the maximally independent sets are bijectively mapped onto the set of idle vertices does not hold for all proper monographs. In the next section, we present the proper monograph labelings of graphs for which the maximum independent sets can be determined from their signatures.

4.2 PROPER MONOGRAPH LABELINGS OF C_n AND $C_n \bigodot K_1$

This section presents the proper monograph labelings of cycles C_n, when $n > 5$, and $C_n \bigodot K_1$ when $n > 5$ and n is odd. Also, we show that the signatures of these proper monographs determine the maximum independent sets of these graphs.

Theorem 4.2.1. *For the cycles C_3, C_4 and C_5, the proper monograph labeling results in only one idle vertex.*

Proof. Consider the cycle C_3 with v_1, v_2, and v_3 as its vertices. Label the vertices as $s, 2s$, and $3s$, respectively. The vertex with label $3s$ is the idle vertex from the above labeling. Suppose the vertices are given the labels s, t, and $s+t$ for $s,t > 1$, then the labeling is not proper monograph labeling because there is no vertex with label $|s-t|$. Hence, the labeling must be $s, 2s$, and $3s$. Now consider the cycle C_4. Let v_1, v_2, v_3, and v_4 be its vertices, as shown in Figure 4.1. The vertex v_4 is left out without

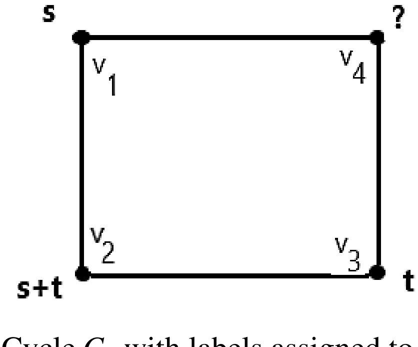


Figure 4.1 Cycle C_4 with labels assigned to v_1, v_2, and v_3.

proper labeling. Now label the vertices v_1, v_2, v_3, and v_4 with labels $s, 2s, 4s$, and $5s$, respectively. The vertex v_4 with label $5s$ becomes the idle vertex. Hence, in cycle C_4, there is only one idle vertex. Similarly, one can verify that the proper monograph labeling of C_5 with labels $s, 2s, 4s, 5s$, and $9s$ assigned to its vertices results in one idle vertex. □

The following theorem gives the proper monograph labelings of C_n, $n > 5$.

Theorem 4.2.2. *The set of idle vertices resulting from the proper monograph labeling of cycle C_n, $n > 5$ gives the maximum independent set of C_n, whose cardinality is $\lfloor \frac{n}{2} \rfloor$.*

Proof. Let $v_1, v_2, \ldots, v_n$ be the vertices of C_n. The proper monograph labeling f of C_n, $n > 5$ is defined as follows: When $n > 5$ and is odd,

$$f(v_i) = \begin{cases} 5^{\frac{(i+1)}{2}} + 1, \text{ for } i = 1,3,5,...,n-2 \\ \\ 5^{\frac{(i)}{2}} + 5^{\frac{(i+2)}{2}} + 2, \text{ for } i = 2,4,6,...,n-3 \\ \\ 2[5^{\frac{(n-1)}{2}} + 1], \text{ for } i = n-1 \\ \\ 2[5^{\frac{(n-1)}{2}} + 1] + 6, \text{ for } i = n \end{cases} \tag{4.2.1}$$

When $n > 5$ and is even,

$$f(v_i) = \begin{cases} 5^{\frac{(i+1)}{2}} + 1, \text{ for } i = 1,3,5,...,n-1 \\ \\ 5^{\frac{(i)}{2}} + 5^{\frac{(i+2)}{2}} + 2, \text{ for } i = 2,4,6,...,n-2 \\ \\ 7 + 5^{\frac{(i)}{2}}, \text{ if } i = n \end{cases} \tag{4.2.2}$$

There are no edges between the non-adjacent vertices of graph C_n. Let u, $v \in V(C_n)$ be the non-adjacent vertices. The difference $|f(u) - f(v)|$ of the labels results in labels of the form $5^{\frac{u}{2}} - 5^{\frac{v+1}{2}}$, $5^{\frac{u}{2}} + 5^{\frac{u+2}{2}} - 5^{\frac{v}{2}} - 5^{\frac{v+1}{2}}$, which are not the labels of any vertices in set C_n when n is even. Similarly, one can verify the above when n is odd.

When n is even, the labeling results in all the even indexed vertices v_i of C_n for $i = \{2,4,6,...,n\}$, as idle vertices. When n is odd, the labeling results in the even indexed vertices v_i of C_n for $i = \{2,4,6,...,n-3\}$, and $i = n$, as idle vertices. The vertices in the idle vertex set are the vertices that belong to the maximum independent set. The proper monograph labelings of cycles C_8 and C_9 are given in Figure 4.2 and Figure 4.3. For the graphs shown in Figure 4.2 and Figure 4.3, the cardinality of the maximum independent set is $\lfloor\frac{n}{2}\rfloor$ = 4. There are four vertices in the set of idle vertices, giving the maximum independent set. In Figure 4.2, the set of vertices with labels $I = \{32, 152, 632, 752\}$ forms the maximum independent set of C_8. In Figure 4.3, the set of vertices with labels $I = \{32, 152, 752, 1258\}$ forms the maximum independent set of C_9. □

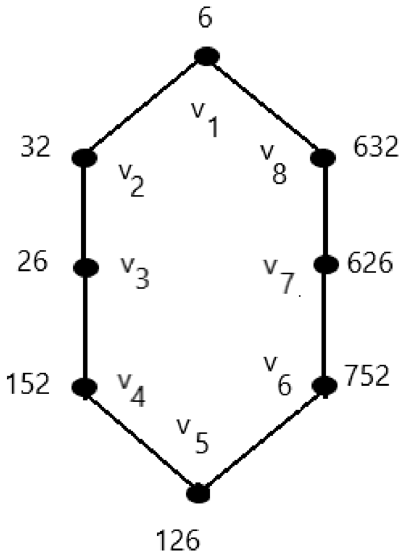


Figure 4.2 Cycle C_8.

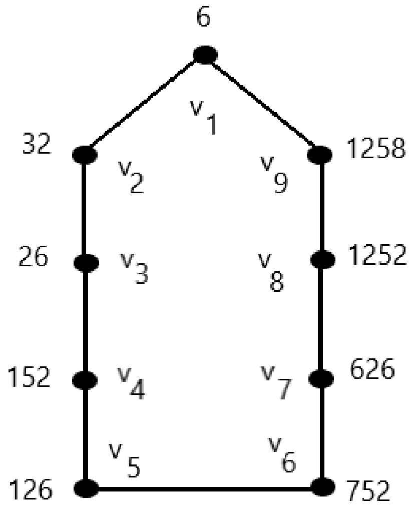


Figure 4.3 Cycle C_9.

The following theorem gives the proper monograph labelings of $C_n \bigodot K_1$, $n > 5$ and n is even.

Theorem 4.2.3. *The set of idle vertices resulting from the proper monograph labeling of $C_n \bigodot K_1$, $n > 5$ and n is even gives the maximum independent set of $C_n \bigodot K_1$, whose cardinality is n.*

Proof. Let $v_1, v_2, \ldots, v_n$ be the vertices of C_n. Let $v_1^{out}, v_2^{out}, \ldots, v_n^{out}$ be the K_1 vertices attached to the vertices of C_n. Here v_i^{out} is the vertex K_i attached to vertex v_i of C_n. The proper monograph labeling f of $C_n \bigodot K_1$, $n > 5$ and n is even is defined as follows:

When $n > 5$ and is even,

$$f(v_i) = \begin{cases} 5^{\frac{(i+1)}{2}} + 1, \text{ for } i = 1,3,5,...,n-1 \\ \\ 5^{\frac{(i)}{2}} + 5^{\frac{(i+2)}{2}} + 2, \text{ for } i = 2,4,6,...,n-2 \\ \\ 7 + 5^{\frac{(i)}{2}}, \text{ if } \; i = n \end{cases} \tag{4.2.3}$$

The vertices $v_1^{out}, v_2^{out}, \ldots, v_n^{out}$ are labeled as follows:

$$f(v_i^{out}) = \begin{cases} 2[5^{\frac{(i+1)}{2}} + 1], \text{ for } i = 1,3,5,...,n-1 \\ \\ \frac{1}{2}[5^{\frac{(i)}{2}} + 5^{\frac{(i+2)}{2}} + 2], \text{ for } i = 2,4,6,...,n-2 \\ \\ \frac{1}{2}[7 + 5^{\frac{(i)}{2}}], \text{ if } \; i = n \end{cases} \tag{4.2.4}$$

From Theorem 4.2.2, it is observed that there are no edges between the non-adjacent vertices of graph C_n. There are no edges between the vertices of graph C_n and K_1, when $i \neq j$ for $v_i \in C_n$ and $v_j^{out} \in$ {set of K_1's} that are attached to C_n. Suppose a vertex $v_i \in C_n$ is adjacent to a vertex v_j^{out} where $i \neq j$. Then we have the following:

$$|f(v_i) - f(v_j^{out})| = \begin{cases} |5^{\frac{(i+1)}{2}} + 1 - \frac{1}{2}[5^{\frac{(j)}{2}} + 5^{\frac{(j+2)}{2}} + 2]|, \text{ for } i = 1,3,5,...,n-1 \\ \qquad\qquad \text{and } j = 2,4,6,...,n-2 \\ \\ |5^{\frac{(i)}{2}} + 5^{\frac{(i+2)}{2}} + 2 - 2[5^{\frac{(j+1)}{2}} + 1]|, \text{ for } i = 2,4,6,...,n-2 \\ \qquad\qquad \text{and } \; j = 1,3,5,...,n-1 \\ \\ |7 + 5^{\frac{(i)}{2}} - 2[5^{\frac{(j+1)}{2}} + 1]|, \text{ if } i = n \text{ and } \; j = 1,3,5,...,n-1 \\ \\ |7 + 5^{\frac{(i)}{2}} - \frac{1}{2}[5^{\frac{(j)}{2}} + 5^{\frac{(j+2)}{2}} + 2]|, \text{ for } i = n \text{ and} \\ \qquad\qquad \text{for } j = 1,3,5,...,n-2 \end{cases} \tag{4.2.5}$$

The labels in (4.2.5) are not assigned to any of the vertices of $C_n \bigodot K_1$. Hence there are no edges between the vertices of graph C_n and K_1, when $i \neq j$ for $v_i \in C_n$ and $v_j^{out} \in \{\text{set of } K_1\text{'s}\}$ that are attached to C_n. When n is even, the labeling results in all the even indexed vertices v_i of C_n for $i = \{2,4,6,...,n\}$ and the odd indexed vertices $v_j^{out} \in \{\text{set of } K_1\text{'s}\}$ that are attached to C_n for $j = \{1,3,5,...,n-1\}$ as the set of idle vertices whose cardinality is n. The vertices in the idle vertex set are the vertices that belong to the maximum independent set. The proper monograph labeling of cycle $C_8 \bigodot K_1$ is given in Figure 4.4. For the graph shown in Figure 4.4, the cardinality of the maximum independent set is n = 8. There are eight vertices in the set of idle vertices, giving the maximum independent set. In Figure 4.4, the set of vertices with labels $I = \{12, 32, 52, 152, 252, 632, 752, 1252\}$ forms the maximum independent set of $C_8 \bigodot K_1$. □

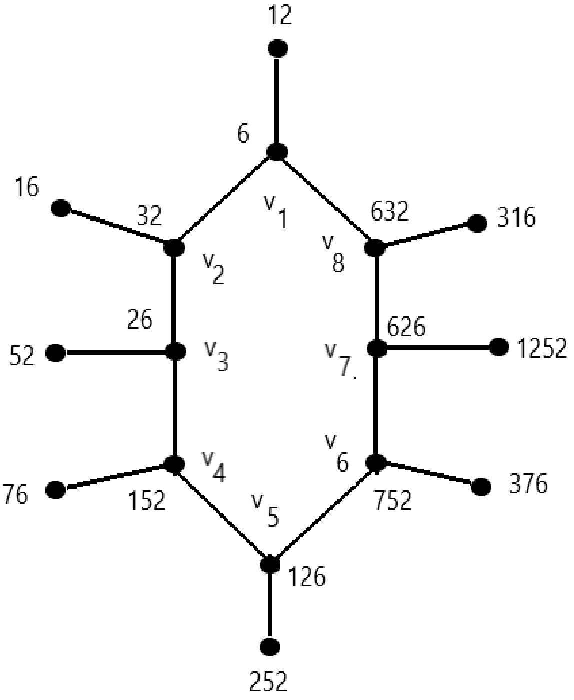


Figure 4.4 Cycle $C_8 \bigodot K_1$.

Remark 4.2.4. *The proper monograph labeling of $C_n \bigodot K_1$ fails to produce the maximum independent sets when $n > 5$ and n is odd. Consider an example of $C_7 \bigodot K_1$. The proper monograph labeling is assigned as in Figure 4.5. The idle vertices are $\{2a, a+b, 2b, b+c, 4c, a+2c\}$. The vertices labeled as c at v_{n-2} (vertex v_5 in C_7) and $\frac{c}{2}$ at v_{n-2}^{out} (K_1 attached to v_5) are working vertices. The vertex v_{n-2}^{out} should be idle, but* *it cannot be assigned a label that makes it an idle vertex. Hence the labeling fails to produce the maximum independent sets for $C_7 \bigodot K_1$. Similarly, the proper monograph labeling of $C_n \bigodot K_1$ fails to produce the maximum independent sets when $n > 5$ and n is odd.*

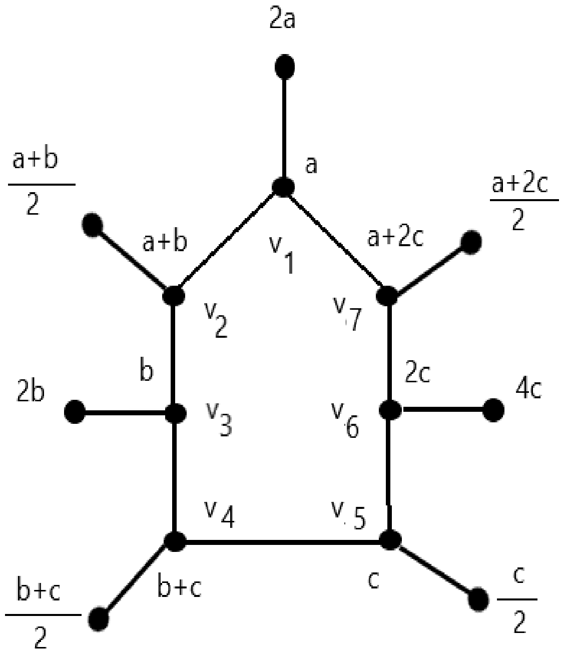


Figure 4.5 Cycle $C_7 \bigodot K_1$.

4.3 PROPER MONOGRAPH LABELINGS OF C_n WITH PATHS ATTACHED TO ITS VERTICES

In this section, we present the proper monograph labelings of cycles with paths attached to one or more vertices and show that the signatures of these proper monographs determine the maximum independent sets of these graphs.

The following theorem gives the proper monograph labelings of cycle C_n, $n > 5$ with paths attached to one or more vertices of C_n.

Theorem 4.3.1. *The set of idle vertices resulting from the proper monograph labeling of cycle C_n, $n > 5$ with paths attached to one or more vertices of C_n gives the maximum independent set of the resulting graph.*

Proof. Let $v_1, v_2, \ldots, v_n$ be the vertices of C_n. Let $u_j^{m,i,k}$ denote the j^{th} vertex of the k^{th} path P_m attached to the i^{th} vertex of C_n. Theorem 4.2.2 gives the proper monograph labeling f of C_n, $n > 5$. The vertices of the path attached to the cycle vertices are labeled as follows: Consider C_n, $n > 5$ and one or more paths are attached to even indexed vertices $i = 2,4,6,...,n$, when n is even, or $i = 2,4,6,...,n-3,n$, when n is odd. The proper monograph labeling f is given as follows: For the sake of simplicity, represent $X = [(2ik-1)5]$ and $Y = \lfloor \frac{n}{2} \rfloor + mk$.

$$f(u_j^{m,i,k}) = \begin{cases} \frac{f(v_i)}{2}, & \text{if } j = 1 \\ X^{Y+j+1}+1, & \text{for } j = 3,5,...,m-1;\ m \text{ is even} \\ & \text{and } \ j = 3,5,...,m;\ m \text{ is odd} \\ \frac{f(v_i)}{2} + X^{Y+j+2}+2, & \text{for } j = 2;\ m \text{ is odd} \\ & \text{and } \ j = 2;\ m \text{ is even} \\ X^{Y+j}+X^{Y+j+2}+2, & \text{for } j = 4,6,...,m-1;\ m \text{ is odd} \\ & \text{and } \ j = 4,6,...,m-2;\ m \text{ is even} \\ 2[X^{Y+j}+1], & \text{if } j = m,\ m \text{ is even} \end{cases} \tag{4.3.1}$$

Now consider C_n, $n > 5$ and one or more paths are attached to odd indexed vertices $i = 1,3,5,...,n-1$, when n is even, or $i = 1,3,5,...,n-2,n-1$, when n is odd. The proper monograph labeling f is given as follows:

$$f(u_j^{m,i,k}) = \begin{cases} f(v_i) + X^{Y+j+2} + 1, \quad \text{if } j = 1 \\ \\ X^{Y+j+1} + 1, \quad \text{for } j = 2,4,6,...,m; \ m \text{ is even} \\ \qquad\qquad\quad \text{and } \ j = 2,4,6,...,m-1; \ m \text{ is odd} \\ \\ X^{Y+j} + X^{Y+j+2} + 2, \quad \text{for } j = 3,5,7,...,m-1; \ m \text{ is odd} \\ \qquad\qquad\qquad\qquad \text{and } \ j = 3,5,7,...,m-1; \ m \text{ is even} \\ \\ 2[X^{Y+j} + 1], \quad \text{if } j = m, \ m \text{ is odd} \end{cases} \tag{4.3.2}$$

From Theorem 4.2.2, there are no edges between the non-adjacent vertices of graph C_n. Also, there are no edges between non-adjacent vertices of graph C_n and P_m. Suppose there is an additional edge between a vertex $i \in C_n$, n is even and vertex $j \in P_m$. Then, $|f(i) - f(j)|$ is given as follows when $i \in V(C_n)$, for $j \in P_m$ and $j = 3,5,...,m-1$; m is even and $j = 3,5,...,m$; m is odd:

$$|f(i) - f(j)| = \begin{cases} |5^{\frac{i+1}{2}} + 1 - X^{Y+j+1} - 1| \\ \\ |5^{\frac{(i)}{2}} + 5^{\frac{(i+2)}{2}} + 2 - X^{Y+j+1} - 1| \\ \\ |7 + 5^{\frac{(i)}{2}} - X^{Y+j+1} - 1| \end{cases} \tag{4.3.3}$$

The labels in (4.3.3) are not assigned to any of the vertices of the resulting graph. Similarly, one can verify the above for the remaining vertex labels of C_n as in (4.2.1) and (4.2.2) with the labels assigned to vertices of P_m as in (4.3.1) and (4.3.2). Hence, there are no additional edges between the vertices of C_n and P_m. The vertices in the idle vertex set are the vertices that belong to the maximum independent set. The proper monograph labeling of cycle C_8 with a path attached to two vertices of C_8 is given in Figure 4.6. For the graph shown in Figure 4.6, the cardinality of the maximum independent set is 8. There are eight vertices in the set of idle vertices, giving the maximum independent set. In Figure 4.6, the set of vertices with labels $I = \{32, 152, 632, 752, [6+5^{10}+1], [5^{10}+1], [76+35^{12}+1], 2[35^{12}+1]\}$ forms the maximum independent set of cycle C_8 with a path attached to two vertices of C_8. $\square$

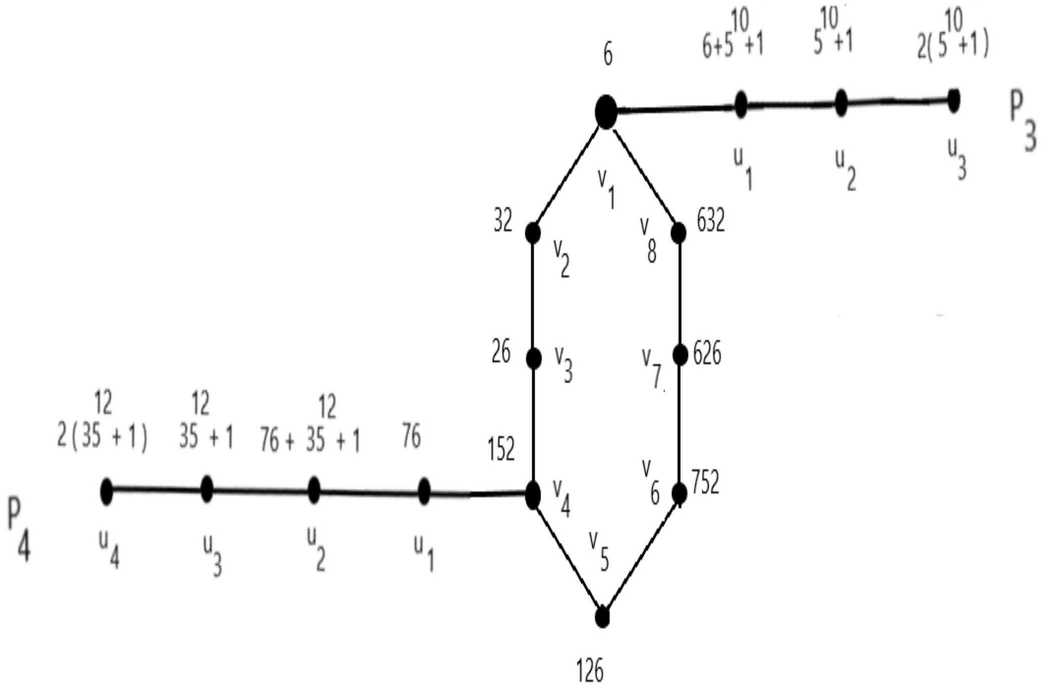


Figure 4.6 Cycle C_8 with a path attached to two vertices of C_8.

Remark 4.3.2. *In a cycle C_n, $n > 5$, only a single path can be attached to its idle vertex that belongs to the maximum independent set of vertices I. Let $u_i \in P_m$, $v_i \in C_n$, and $v_i \in I$. Since $v_i \in I$, there can only be one vertex label that is derived from v_i. The label $\frac{v_i}{2}$ is assigned to u_i. Suppose $u_i \in P_{m_1}$ and $u_j \in P_{m_2}$, where P_{m_1} and P_{m_2} are the paths attached to the idle vertex $v_i \in C_n$. Then u_i has to be labeled as $\frac{v_i}{2}$ and u_j labeled as $2(v_i)$. This labeling results in $v_i \notin I$. Hence, only one path can be attached to an idle vertex $v_i \in C_n$ such that $v_i \in I$. For the cycle C_n, $n > 5$, the vertices that form the maximum independent set can have only one path attached, whereas the working vertices in C_n can have any number of paths attached to them.*

4.4 PROPER MONOGRAPH LABELINGS OF C_n WITH ATTACHED IRREDUCIBLE TREES

This section presents the proper monograph labelings of cycles with an irreducible tree attached to one or more vertices. We show that the signatures of these proper monographs determine the maximum independent sets of these graphs.

The irreducible trees are also assumed to satisfy the following two conditions:

- The length of the limb should be greater than one.

- The branchnodes must occur at an even distance from the vertex of the cycle to which the tree is connected.

The following theorem gives the proper monograph labeling of cycle C_n, $n > 5$ with an irreducible tree attached to one or more vertices

Theorem 4.4.1. *The set of idle vertices resulting from the proper monograph labeling of cycle C_n, $n > 5$ with an irreducible tree attached to one or more vertices of C_n gives the maximum independent set of the resulting graph.*

Proof. Let $v_1, v_2, \ldots, v_n$ be the vertices of C_n. Theorem 4.2.2 gives the proper monograph labeling f of C_n, $n > 5$. Let $u_j^{l,i}$ denote the j^{th} vertex of the irreducible tree T whose root is attached to the i^{th} vertex of C_n where $v_i \notin I$. Here l is the level of vertex j. The total number of vertices in level l is k_l as shown in Figure 4.7.

The proper monograph labeling f is given as follows: For the sake of simplicity, represent $X = [(2(i+1)-1)5]$ and $Y = \lfloor \frac{n}{2} \rfloor$. For the non-pendant vertices at the levels $l = 3, 5, ..., L$, we consider the parent vertex p and the child vertex c to which vertex j is attached, i.e., $u_j^{l,i}$ is adjacent to $u_p^{l-1,i}$ and $u_c^{l+1,i}$. If $u_j^{l,i}$ is not a pendant vertex(end vertex), then

$$f(u_j^{l,i}) = \begin{cases} X^{Y+(k_1+k_2+...+k_{l-1})+j} + 1, & \text{for } j = 1,2,...,k_l;\ l = 2,4,...,L; \\ & \text{and } i = 1,3,5,...,n-1;\ n \text{ is even;} \\ & \text{and } i = 1,3,5,...,n-2;\ \text{and } i = n-1;\ n \text{ is odd;} \\ f(v_i) + X^{Y+k_1+j} + 1, & \text{if } l = 1; \\ X^{Y+(k_1+k_2+...+k_{l-1})+p} + X^{Y+(k_1+k_2+...+k_{l-1})+c} + 2, & \\ & \text{for } j = 1,2,...,k_l;\ l = 3,5,...,L; \\ & p \text{ is the parent vertex index and} \\ & c \text{ is the child vertex index} \end{cases} \tag{4.4.1}$$

If $u_j^{l,i}$ is a pendant vertex(end vertex) and is adjacent to $u_p^{l-1,i}$, then

$$f(u_j^{l,i}) = \Big\{ 2[X^{Y+(k_1+k_2+...+k_{l-2})+p} + 1], \ \text{if } j \text{ is end/pendant vertex} \tag{4.4.2}$$

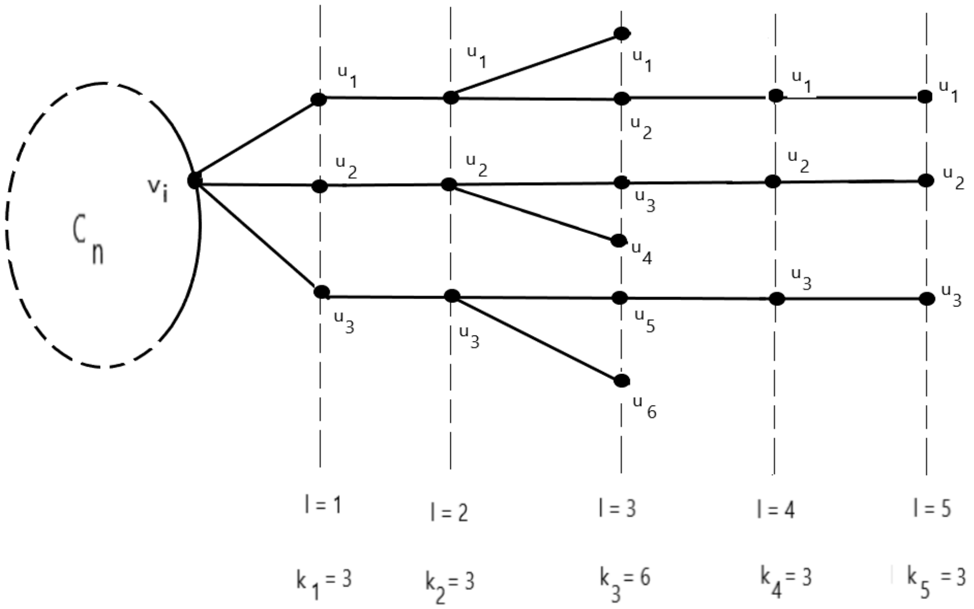


Figure 4.7 Cycle C_n with an irreducible tree attached to a vertex of C_n.

Thus, f gives the proper monograph labeling that results in a set of idle vertices. □

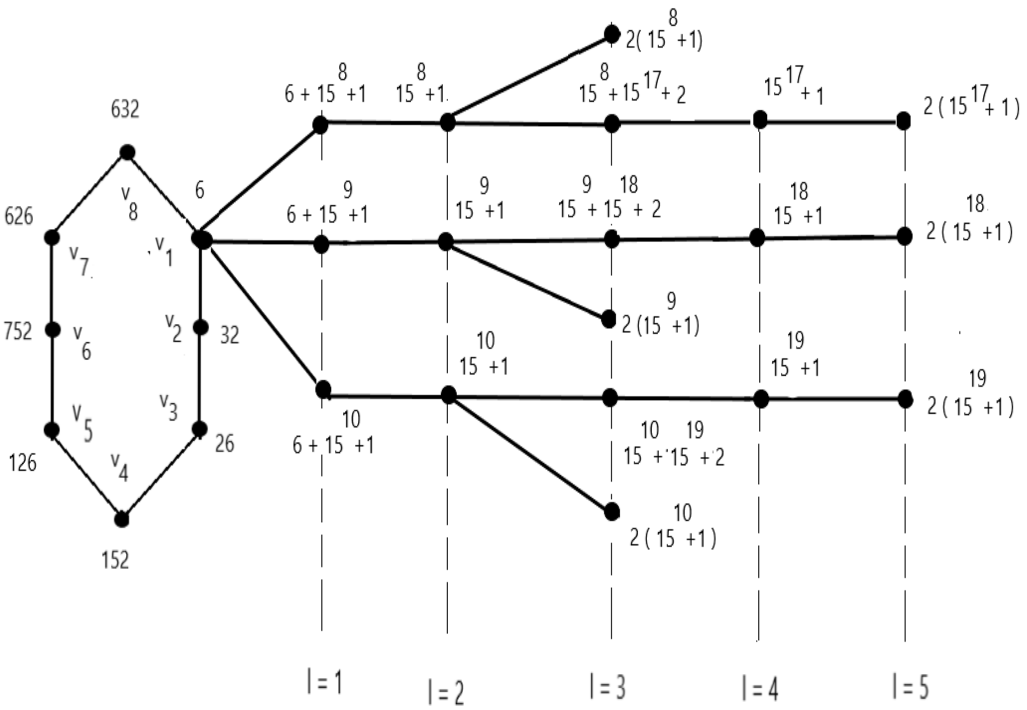


Figure 4.8 Cycle C_8 with an irreducible tree attached to a working vertex v_1 of C_8.

There are no additional edges between the vertices of C_n and T, and it can be verified as in Theorem 4.3.1. The idle vertices in the graph give the maximum independent set I of G. The proper monograph labeling of C_8 with an irreducible tree attached to a working vertex v_1 of C_8 is shown in Figure 4.8. The proper monograph labeling of C_8 results in 4 idle vertices, and the tree attached to v_1 results in 12 idle vertices. Hence, the resulting graph has 16 vertices in the maximum independent set I.

CHAPTER 5

GRAPH MODELS FOR MULTIPROCESSOR INTERCONNECTION NETWORKS

In this chapter, a few interesting graphs are considered, demonstrating that they could be suitable models for MINs. It is noted that determining the chromatic number for these and similar graphs is easy. This allows the introduction of two additional tightness values, $t_3(G)$ and $t_4(G)$, which could be calculated efficiently. Keeping in mind the emphasis on λ_1, from Wilf (1967) we consider the inequality $\chi(G) \leq 1 + \lambda_1(G)$ and define $t_3(G)$ and $t_4(G)$ based on the chromatic number of the graph. The new tightness values can also be partially ordered using the '$\leq$' relation. Also, we show that graphs with small values of $t_3(G)$ and $t_4(G)$ are well suited for the design of multiprocessor interconnection networks.

5.1 PRELIMINARIES

Several definitions and theorems that will be used in the remainder of this chapter are given in this section. Harary and Norman (1960) used the term line graph for the very first time. However, these concepts were studied by Whitney (1932) and Krausz (1943).

Definition 5.1.1. *(Harary and Norman (1960)) The line graph $L(G)$ of a graph G has $E(G)$ as its vertex set, and two vertices are adjacent in $L(G)$ if and only if they are adjacent as edges in G.*

A proper vertex (edge) coloring of a graph G is an assignment of colors to the vertices (edges) of G, so that adjacent vertices (edges) are uniquely colored. A proper vertex (edge) coloring that uses colors from a set of k colors is a k-vertex (edge) coloring.

Definition 5.1.2. *(Chartrand and Zhang (2009)) The minimum positive integer k for which G is k-vertex colorable is called the* ***chromatic number*** *of G and is denoted by* $\chi(G)$*. The* ***chromatic index*** *(or edge chromatic number)* $\chi^{'}(G)$ *of a graph G is the minimum positive integer k for which G is k-edge colorable. Furthermore,* $\chi^{'}(G) = \chi(L(G))$ *for every non empty graph G.*

According to Vizing (1965), the definition of Class one and Class two graphs are given below:

Definition 5.1.3. *(Vizing (1965)) Let* $\Delta(G)$ *be the maximum vertex degree of the graph G. Graphs that have* $\chi^{'}(G) = \Delta(G)$ *are called Class one graphs. Graphs with* $\chi^{'}(G) = \Delta(G)+1$ *are called Class two graphs.*

The concept of total graphs was introduced by Behzad (1970).

Definition 5.1.4. *(Behzad (1970)) The total graph* $T(G)$ *of a graph G is that graph whose vertex set is* $V(G) \cup E(G)$*, and in which two vertices are adjacent if and only if they are adjacent or incident in G.*

Let $G = (V(G), E(G))$ and $H = (V(H), E(H))$ be two graphs. For the construction of new well-suited graphs, the following graph operations are considered.

Definition 5.1.5. *(Hammack et al. (2011)) The Cartesian product* $G\Box H$ *of graphs G and H has vertex set* $V(G\Box H) = V(G) \times V(H)$*, and edge set* $E(G\Box H) = \{(u_1, v_1)(u_2, v_2) \,|u_1u_2 \in E(G) \text{ and } v_1 = v_2 \text{ , or } u_1 = u_2 \text{ and } v_1v_2 \in E(H)\}$ *.*

Definition 5.1.6. *(Hammack et al. (2011)) The Tensor product (direct product)* $G \times H$ *of graphs G and H has vertex set* $V(G \times H) = V(G) \times V(H)$*, and edge set* $E(G \times H) = \{(u_1, v_1)(u_2, v_2)|u_1u_2 \in E(G) \text{ and } v_1v_2 \in E(H)\}$*.*

Definition 5.1.7. *(Brouwer et al. (1989)) Let X be a finite set. The Johnson graph of the e-sets in X has vertex set* $\binom{X}{e}$*, the collection of e-subsets of X. Two vertices* γ*,* δ *are adjacent whenever* $\gamma \cap \delta$ *has cardinality* $e-1$*. When X is some unspecified n-set, the graph is denoted as* $\binom{n}{e}$ *or* $J(n, e)$*.*

Definition 5.1.8. *(Brouwer et al. (1989)) The Rook's graph is defined as the Cartesian product of two complete graphs* K_n *and* K_m*, expressed as* $K_n\Box K_m$*. It is also called as* $m \times n$ *grid.*

Definition 5.1.9. *(Brouwer et al. (1989)) The n-crown graph is defined as the complement of the* $2 \times n$ *grid, i.e., it is isomorphic to the complement of Rook's graph* $\overline{K_2\Box K_n}$*.*

The following are the definitions of four types of graph tightness introduced by Cvetković and Davidović (2008).

Definition 5.1.10. *(Cvetković and Davidović (2008))* ***First type mixed tightness*** $t_1(G)$ *of a graph G is defined as the product of the number of distinct eigenvalues* m *and the maximum vertex degree* Δ *of G, i.e.,* $t_1(G) = \mathsf{m}\Delta$.

Definition 5.1.11. *(Cvetković and Davidović (2008))* ***Structural tightness*** $stt(G)$ *is the product* $(D+1)\Delta$*, where D is diameter and* Δ *is the maximum vertex degree of a graph G, i.e.,* $stt(G) = (D+1)\Delta$.

Definition 5.1.12. *Cvetković and Davidović (2008)* ***Spectral tightness*** $spt(G)$ *is the product of the number of distinct eigenvalues* m *and the largest eigenvalue* λ_1 *of a graph G, i.e.,* $spt(G) = \mathsf{m}\lambda_1$.

Definition 5.1.13. *Cvetković and Davidović (2008)* ***Second type mixed tightness*** $t_2(G)$ *is defined as a product of the diameter D of G and the largest eigenvalue* λ_1*, i.e.,* $t_2(G) = (D+1)\lambda_1$.

In the analysis of a graph's tightness, the following theorem from Cvetković and Davidović (2008) seems to be of fundamental importance .

Theorem 5.1.14. *(Cvetković and Davidović (2008)) For any kind of tightness, the number of connected graphs with a bounded tightness is finite.*

The following theorem gives the eigenvalues of $G \times H$:

Theorem 5.1.15. *(Cvetković (1971)) The eigenvalues of* $G \times H$ *are just the pairwise products of the eigenvalues of G and H.*

The following theorem gives the eigenvalues of $G \square H$:

Theorem 5.1.16. *(Cvetković (1971)) The eigenvalues of* $G \square H$ *are just the pairwise sums of the eigenvalues of G and H.*

The following result gives an explicit formula for the eigenvalues of $L(G)$ in terms of the eigenvalues of a regular graph G.

Corollary 5.1.17. *(Brouwer and Haemers (2011)) If G is a regular graph of degree r, with n vertices and* $m(=\frac{nr}{2})$ *edges, and eigenvalues* θ_i *for* $i = 1, 2, ..., n$*, then line graph* $L(G)$ *is* $(2r-2)$*-regular with eigenvalues* $(\theta_i + r - 2)$ *for* $i = 1, ..., n$*, and* -2 *with the multiplicity* $(m-n)$.

The following result gives an explicit formula for the eigenvalues of $L(G)$ in terms of the signless Laplace eigenvalues of a non-regular graph G.

Proposition 5.1.18. *(Brouwer and Haemers (2011)) Let G be a graph on n vertices, having m edges, and let $q_1 \geq q_2 \geq ... \geq q_n$ be the signless Laplace eigenvalues of G, then the eigenvalues of Line graph of G are $\theta_i = q_i - 2$ for $i = 1, 2, ..., n$, and $\theta_i = -2$ if $n < i \leq m$.*

Theorem 5.1.19. *(Chartrand and Zhang (2009)) (Konig's Theorem) If G is a non empty bipartite graph, then $\chi^{'}(G) = \Delta(G)$.*

A factor of a graph refers to its spanning subgraph. A sequence of pairwise edge-disjoint subgraphs $G_1, G_2, \ldots, G_n$ whose union is G is called a decomposition of G, and is represented as $G = \bigcup_1^n G_i$. If G_i is r-regular spanning subgraph of G, then every G_i is called an r-factor, and G is called r-factorable graph. A graph M is a matching if each vertex has a degree of 0 or 1. Thus, the edge set of a 1-factor in a graph G is a perfect matching in G. So, a graph G has a 1-factor if and only if G has a perfect matching.

Theorem 5.1.20. *(Chartrand and Zhang (2009)) A regular graph G is of Class one if and only if G is 1-factorable.*

Corollary 5.1.21. *(Chartrand and Zhang (2009)) Every regular graph of odd order is of Class two.*

Theorem 5.1.22. *(Mahamoodian (1981)) If $\chi^{'}(G) = \Delta(G)$, then $\chi^{'}(G \square H) = \Delta(G + H) = \Delta(G) + \Delta(H)$*

Theorem 5.1.23. *(Jaradat (2005)) Let G and H be two graphs such that $\chi^{'}(H) = \Delta(H)$. Then $\chi^{'}(G \times H) = \Delta(G \times H) = \Delta(G)\Delta(H)$*

5.2 NEW TIGHTNESS VALUES BASED ON THE CHROMATIC NUMBER

Cvetković and Davidović (2008) showed that the tightness values $t_1(G), stt(G), spt(G)$, and,$t_2(G)$ are partially ordered by the relation '$\leq$' as follows:

$$t_2(G) \leq stt(G) \leq t_1(G)$$
$$t_2(G) \leq spt(G) \leq t_1(G)$$

Later, they concluded that the graphs with small tightness values of $t_2(G)$ are more suitable for the design of multiprocessor interconnection networks.

All the graphs presented in this chapter are line graphs of regular graphs, bipartite graphs, or products of these graphs. It is a well known fact that the index of r-regular graph is equal to the vertex degree r, while the complete bipartite graph $K_{p,q}$ has spectrum $\pm\sqrt{pq}$, and 0 whose multiplicity is $p+q-2$. As a result, we can determine the eigenvalues of the graphs based on whether the graph is regular or bipartite, as mentioned below.

To get the eigenvalues of a Line graph, one must first compute the spectrum of the original graph, knowing whether it is a regular or bipartite graph. The spectrum of the resulting Line graph is then computed using the Corollary 5.1.17 and the Proposition 5.1.18. The eigenvalues of the graphs generated from graph operations, such as Tensor product and Cartesian product are computed using the Theorem 5.1.15 and the Theorem 5.1.16. However, by determining whether the graph is a Class one or Class two graph, using the results from the preliminaries section, the chromatic index of these graphs is easily obtained without any complicated calculations. Since, it is known that $\chi^{'}(G) = \chi(\text{Line Graph}(G))$ from Chartrand and Zhang (2009) for any non-empty graph G, determining the edge chromatic number of the graphs presented here is all that is required.

Considering the Line graph of Tensor product $K_n \times K_p$, we show that computing the chromatic number is more straightforward than computing the largest eigenvalue for this graph. The largest eigenvalue λ_1 is computed as follows:
The complete graph K_n is an $(n-1)$ regular graph and the characteristic polynomial is $P(K_n,x) = (x-n+1)(x+1)^{n-1}$. From the polynomial, it is clear that the eigenvalues of K_n are $(n-1)$ and -1 with the multiplicities 1 and $n-1$. From Theorem 5.1.15, it is known that the eigenvalues of $K_n \times K_p$ are $(n-1)(p-1)$,$(n-1)(-1)^{p-1}$, $(-1)^{n-1}(p-1)$, and $(-1)^{n-1}(-1)^{p-1}$. From Corollary 5.1.17, the eigenvalues of the line graph of Tensor product of $K_n \times K_p$ are calculated as follows:

$$\begin{aligned}
&\lambda_1 = np-n-p+1+np-n-p+1-2 = 2np-2(n+p)\\
&\lambda_2 = (n-1)(-1)+np-n-p+1-2 = np-2n-p\\
&\lambda_3 = (-1)(p-1)+np-n-p+1-2 = np-2p-n\\
&\lambda_4 = (-1)(-1)+np-n-p+1-2 = np-n-p\\
&\text{and } \lambda_5 = -2
\end{aligned}$$

Therefore $\lambda_1 = 2np-2(n+p)$.

The chromatic number of the Line graph of $K_n \times K_p$ can be quickly computed as follows: the first step is to figure out whether the graph is a Class one or Class two graph; Theorem 5.1.20, Corollary 5.1.21, and Theorem 5.1.23 are then used to compute the chromatic index of the graph. Here, the chromatic index is $np-n-p+2$ when the number of vertices in the original graph is odd, and $np-n-p+1$ when the number of vertices in the original graph is even. The observation that one could quickly determine the chromatic number for the graphs presented as examples in this chapter leads to the introduction of two additional tightness values, $t_3(G)$ and $t_4(G)$, which can be partially ordered by the relation '$\leq$'. The basis for the present investigation is the following result from Wilf (1967).

Theorem 5.2.1. *(Wilf (1967)) If χ is the chromatic number and λ_1 is the largest eigenvalue, then*

$$\chi \leq 1+\lambda_1 \tag{5.2.1}$$

with equality if and only if G is a complete graph or an odd circuit.

The maximum and minimum vertex degree of graph G is denoted by $\Delta = \Delta(G)$ and $\delta = \delta(G))$, respectively. The average vertex degree of G is represented as $\overline{d} = \overline{d}(G)$. From Cvetković et al. (1995), we have

$$\delta \leq \overline{d} \leq \lambda_1 \leq \Delta \text{ and} \tag{5.2.2}$$

$$D \leq \mathsf{m}-1, \text{ where D is the diameter.} \tag{5.2.3}$$

Rewrite (5.2.1) as

$$\chi - 1 \leq \lambda_1 \tag{5.2.4}$$

Recalling Definition 5.1.12, which states $spt(G) = \mathsf{m}\lambda_1$ and from (5.2.4), the new tightness value called the Third type mixed tightness $t_3(G)$ can be defined as follows:

Definition 5.2.2. ***Third type mixed tightness*** *$t_3(G)$ is the product of the number of distinct eigenvalues* m *and $(\chi-1)$, where χ is the chromatic number of a graph G, i.e., $t_3(G) = \mathsf{m}(\chi-1)$.*

Considering Definition 5.1.13, which states $t_2(G) = (D+1)\lambda_1$ and equation(5.2.4), the new tightness value called the Second type of Structural tightness $t_4(G)$ can be defined as follows:

Definition 5.2.3. ***Second type of Structural tightness*** *$t_4(G)$ is the product $(D+1)(\chi-1)$, where D is diameter and χ is the chromatic number of a graph G.*

From Definition 5.2.2, Definition 5.2.3, equations (5.2.2), (5.2.3), and (5.2.4), the new tightness values can be partially ordered as follows:

$$
\begin{aligned}
&t_3(G) \leq spt(G) \leq t_1(G)\\
&t_4(G) \leq t_2(G) \leq stt(G) \leq t_1(G)\\
&t_4(G) \leq t_2(G) \leq spt(G) \leq t_1(G), \text{ and}\\
&t_4(G) \leq t_3(G)
\end{aligned}
$$

Hence, from the above inequalities, it is clear that the graphs with small values of $t_3(G)$ and $t_4(G)$ are well suited for the design of the multiprocessor interconnections topologies. It has been proved that the number of connected graphs with bounded tightness is finite for the four types of tightness values defined in (Theorem 5.1.14, Cvetković and Davidović (2008)). The following theorem proves that this criterion also applies to the two new tightness values defined in this chapter.

Theorem 5.2.4. *The number of connected graphs with the bounded Third type mixed tightness $t_3(G)$ and Second type of Structural tightness $t_4(G)$ is finite.*

Proof. The following inequality holds for the number of vertices n in a graph G:

$$n \leq 1 + \Delta + \Delta(\Delta-1) + \Delta(\Delta-1)^2 \cdots + \Delta(\Delta-1)^{D-1} \tag{5.2.5}$$

As in the proof of (Theorem 5.1.14, Cvetković and Davidović (2008)) we assume that $t(G) \leq a$, for a given positive integer a, where $t(G)$ represents the two new tightness values $t_3(G)$ and $t_4(G)$. We now prove that for the new tightness values, both the diameter D and maximum vertex degree Δ are bounded by a number denoted as b. According to Brooks (1941), it is known that $\chi(G) \leq 1 + \Delta(G)$, and from Cvetković et al. (1995), we have $D \leq \mathsf{m} - 1$ for the diameter D. Here, m is the number of distinct eigenvalues, and $\chi(G)$ is the chromatic number of G. Note that $\Delta \leq a$ and $D \leq a-1$, as shown in the proof of (Theorem 5.1.14, Cvetković and Davidović (2008)). Now for $t_3(G) = \mathsf{m}(\chi - 1)$, $t_3(G) \leq a$ implies

$$
\begin{aligned}
&\mathsf{m}(\chi-1) \leq a \Rightarrow \ \mathsf{m} \leq a \text{ and } (\chi-1) \leq a, \text{ which implies}\\
&D \leq a-1,\ \Delta \leq a, \text{ and we assign } b = a;
\end{aligned}
$$

and for $t_4(G) = (D+1)(\chi-1)$, when $t_4(G) \leq a$, the following holds :

$$
\begin{aligned}
&(D+1)(\chi-1) \leq a \Rightarrow \ D+1 \leq a \ \text{ and } (\chi-1) \leq a,\\
&\text{which implies } D \leq a-1, \Delta \leq a, \text{ and we assign } b = a;
\end{aligned}
$$

Based on the relationship in (5.2.5), and assuming that both D and Δ are bound by the number b, we have the following:

$$
\begin{aligned}
n &\leq 1+\Delta+\Delta^2+\Delta^3\cdots+\Delta^D \leq 1+\Delta+\Delta^2+\Delta^3\cdots+\Delta^b \\
&\leq 1+b+b^2+b^3\cdots+b^b
\end{aligned}
$$

Hence, we prove that a connected graph with the given number of vertices n and a bounded tightness is also bounded. Therefore, we conclude that the number of connected graphs with the bounded tightness $t_3(G)$ and $t_4(G)$ is finite. □

5.3 GRAPHS SUITABLE FOR MINs

One can find examples of well-suited MINs resulting from some graph operations with tightness values as $O(\sqrt{N})$ or $O(N)$ in Cvetković et al. (2016). In this section, we present examples of graphs resulting from several graph operations. Graph operations include line graphs of graph products, such as the Cartesian product and the Tensor product of graphs. Also, we consider the line graphs of Johnson graphs, Rook graphs, and Crown graphs. The resulting graphs are considered as well-suited interconnection network models since their tightness values range from $O(\sqrt[4]{N})$ to $O(\sqrt{N})$, where N is the number of vertices of the graph that is considered.

Obtaining the chromatic number of an arbitrary graph is NP-Hard, but one can get the chromatic number for the well-known graphs using SageMath a free and open-source software by Stein (2007). The graphs presented here are line graphs of some regular or bipartite graphs or products of these graphs. For every non-empty graph G, $\chi'(G) = \chi(L(G))$, according to Definition 5.1.2. As a result, obtaining the chromatic index of the graphs provided here is sufficient. We can determine the edge chromatic index of these graphs using the results in the Preliminaries section. The computations in this section are performed using SageMath.

The set of connected graphs having at least two vertices is represented as G_c, and $t(G) \in \{t_1(G),\, stt(G), spt(G), t_2(G), t_3(G), t_4(G)\}$. Now consider the following notations:

$$
\begin{aligned}
S^{O(\sqrt{N})} &= \{G : G \in G_c,\ t(G) = O(\sqrt{N})\} \\
S^{O(\sqrt[3]{N})} &= \{G : G \in G_c,\ t(G) = O(\sqrt[3]{N})\} \\
S^{O(\sqrt[4]{N})} &= \{G : G \in G_c,\ t(G) = O(\sqrt[4]{N})\}
\end{aligned}
$$

Some of the notations used in the examples are given below in Table 5.1. Throughout the examples, we consider the order of the original graph and its regularity. The graph parameters such as D, Δ, m, λ_1, and N^{OG} are computed. Since $\chi^{'}(G) = \chi(L(G))$, the graph's chromatic number is derived from the edge chromatic index obtained for such graphs using the theorems stated in Section 5.1

Table 5.1 Notations

N	Number of vertices in the newly constructed graph
D	Diameter
m	Number of distinct eigenvalues of G.
Δ	Maximum degree
λ_1	Largest eigenvalue of G
N^{OG}	Number of vertices in the original graph

Example 5.3.1. The set $S^{O(\sqrt{N})}$ contains the following graphs:

5.3.1.1 Line graph of Tensor product $K_n \times K_p$.

5.3.1.2 Line graph of Tensor product $K_n \times K_{p,p}$.

5.3.1.3 Line graph of Cartesian product $K_{1,n-1} \square K_{1,p-1}$.

5.3.1.4 Line graph of Complete graph K_n.

5.3.1.5 Line graph of Complete Bipartite graph $K_{n,n}$.

5.3.1.6 Line graph of Crown graph $K_{n,n} - I$.

5.3.1.7 Line graph of Complete Tripartite graph $K_{n,n,n}$.

5.3.1.1 Line graph of Tensor product $K_n \times K_p$: Consider $G_1 = L(K_n \times K_p)$ = Line graph of Tensor product $K_n \times K_p$, for $n > 2$ and $p > 2$. All relevant parameters of G_1 are summarized in Table 5.2.

Table 5.2 Line graph of Tensor product $K_n \times K_p$, for $n > 2$ and $p > 2$.

N	D	m	Δ	λ_1
$\frac{n^2p^2-n^2p-np^2+np}{2}$	≤ 3	≤ 5	$2np-2(n+p)$	$2np-2(n+p)$

Table 5.3 presents some properties of the Tensor product $K_n \times K_p$, for $n > 2$:

Table 5.3 Tensor product $K_n \times K_p$, for $n > 2$ and $p > 2$.

N^{OG}	Δ	Is Regular?
$n * p$	$np - n - p + 1$	Yes

The chromatic number of the Line graph of Tensor product $K_n \times K_p$ is calculated as follows: if the number of vertices N^{OG} is odd, then from Corollary 5.1.21 it is clear that the edge chromatic number (chromatic index) of $K_n \times K_p$ is $np - n - p + 2$; if the number of vertices N^{OG} is even, then from Theorem 5.1.20 the edge chromatic number (chromatic index) of $K_n \times K_p$ is $np - n - p + 1$. Also, from Definition 5.1.2, $\chi^{'}(K_n \times K_p)$ = $\chi(L(K_n \times K_p))$. If $n = p$, then the tightness values are given as follows:

$$t_1(G_1) \leq 5(2n^2 - 2(n+n)) \leq 10n^2 - 10(2n) = O(\sqrt{N});$$
$$stt(G_1) \leq 4(2n^2 - 2(n+n)) \leq 8n^2 - 8(2n) = O(\sqrt{N});$$
$$spt(G_1) \leq 5(2n^2 - 2(n+n)) \leq 10n^2 - 10(2n) = O(\sqrt{N});$$
$$t_2(G_1) \leq 4(2n^2 - 2(n+n)) \leq 8n^2 - 8(2n) = O(\sqrt{N}).$$

If $n = p$, the new tightness values $t_3(G_1)$ and $t_4(G_1)$ are also given as follows:

$t_3(G_1) = \mathrm{m}(\chi - 1) \leq 5(n^2 - 2n) = O(\sqrt{N})$; if no. of vertices is even
$t_3(G_1) = \mathrm{m}(\chi - 1) \leq 5(n^2 - 2n + 1) = O(\sqrt{N})$; if no. of vertices is odd
$t_4(G_1) = (D+1)(\chi - 1) \leq 4(n^2 - 2n) = O(\sqrt{N})$; if no. of vertices is even
$t_4(G_1) = (D+1)(\chi - 1) \leq 4(n^2 - 2n + 1) = O(\sqrt{N})$; if no. of vertices is odd

The tightness values t_1, stt, spt, t_2, t_3, and t_4 are bounded by $O(\sqrt{N})$, and hence G_1 can be used as a model for MINs. The Line graph of Tensor product $K_3 \times K_3$ is given in Figure 5.1.

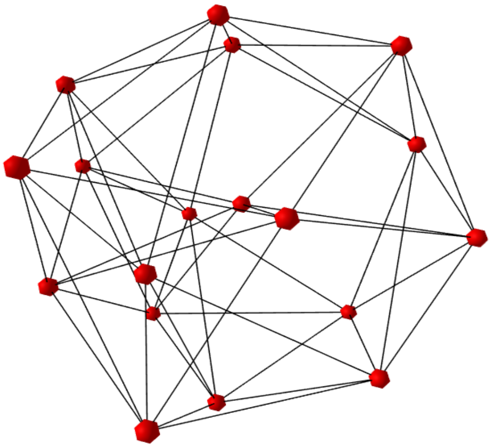


Figure 5.1 Line graph of Tensor product $K_3 \times K_3$.

5.3.1.2 Line graph of Tensor product $K_n \times K_{p,p}$: Consider $G_2 = L(K_n \times K_{p,p}) =$ Line graph of Tensor product $K_n \times K_{p,p}$, for $n > 2$. All relevant parameters of G_2 are summarized in Table 5.4.

Table 5.4 Line graph of Tensor product $K_n \times K_{p,p}$, for $n > 2$.

N	D	m	Δ	λ_1
$n^2p^2 - np^2$	≤ 3	≤ 5	$2np - 2p - 2$	$2np - 2p - 2$

Table 5.5 presents some properties of the Tensor product $K_n \times K_{p,p}$, for $n > 2$:

Table 5.5 Tensor product $K_n \times K_{p,p}$, for $n > 2$.

N^{OG}	Δ	Is Regular?
$2 * n * p$	$np - p$	Yes

The chromatic number of the Line graph of Tensor product $K_n \times K_{p,p}$ is calculated as follows: from Theorem 5.1.23, it can be observed that the edge chromatic number (chromatic index) of $K_n \times K_{p,p}$ is $np - p$. Also, from Definition 5.1.2, the edge chromatic number $\chi^{'}(K_n \times K_{p,p})$ = $\chi(L(K_n \times K_{p,p}))$. If $n = p$, then the tightness values are given as follows:

$$\begin{aligned} t_1(G_2) &\leq 5(2n^2 - 2n - 2) = O(\sqrt{N}); \\ stt(G_2) &\leq 4(2n^2 - 2n - 2) = O(\sqrt{N}); \\ spt(G_2) &\leq 5(2n^2 - 2n - 2) = O(\sqrt{N}); \\ t_2(G_2) &\leq 4(2n^2 - 2n - 2) = O(\sqrt{N}). \end{aligned}$$

If $n = p$, the new tightness values $t_3(G_2)$ and $t_4(G_2)$ are also given as follows:

$$\begin{aligned} t_3(G_2) &= \mathrm{m}(\chi - 1) \leq 5(n^2 - n - 1) \leq 5n^2 - 5n - 5 = O(\sqrt{N}); \\ t_4(G_2) &= (D+1)(\chi - 1) \leq 4(n^2 - n - 1) \leq 4n^2 - 4n - 4 = O(\sqrt{N}). \end{aligned}$$

The tightness values t_1, stt, spt, t_2, t_3, and t_4 are bounded by $O(\sqrt{N})$, and hence G_2 can be used as a model for MINs. The Line graph of Tensor product $K_3 \times K_{2,2}$ is given in Figure 5.2.

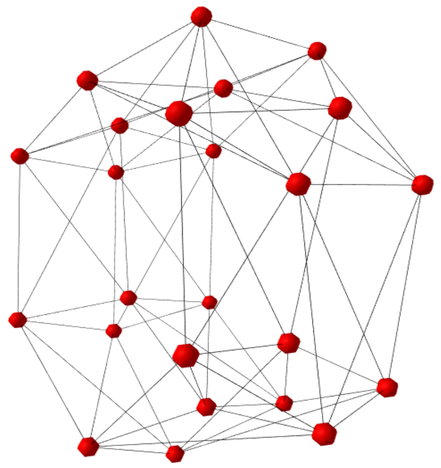


Figure 5.2 Line graph of Tensor product $K_3 \times K_{2,2}$.

5.3.1.3 Line graph of Cartesian product $K_{1,n-1} \Box K_{1,p-1}$**:** Consider $G_3 = L(K_{1,n-1} \Box K_{1,p-1})$ = Line graph of Cartesian product $K_{1,n-1} \Box K_{1,p-1}$. Table 5.6 summarizes all the relevant parameters of G_3 .

Table 5.6 Line graph of Cartesian product $K_{1,n-1} \Box K_{1,p-1}$.

N	D	m	Δ	λ_1
$2np - n - p$	≤ 3	≤ 8	$2p + n - 4$	$n + p - 2$

The properties of Cartesian product $K_{1,n-1}\square K_{1,p-1}$ are given in Table 5.7.

Table 5.7 Cartesian product $K_{1,n-1}\square K_{1,p-1}$.

N^{OG}	Δ	Is Regular?
$n*p$	$n+p-2$	No

The chromatic number of the Line graph of Cartesian product $K_{1,n-1}\square K_{1,p-1}$ is calculated as follows: from Theorem 5.1.22 the edge chromatic number (chromatic index) of $K_{1,n-1}\square K_{1,p-1}$ is $n+p-2$. Also, from Definition 5.1.2, the edge chromatic number $\chi^{'}(K_{1,n-1}\square K_{1,p-1}) = \chi(L(K_{1,n-1}\square K_{1,p-1}))$. If $n=p$, then the tightness values are given as follows:

$$\begin{aligned}
t_1(G_3) &\leq 8(2n+n-4) \leq 8(3n-4) = O(\sqrt{N});\\
stt(G_3) &\leq 4(2n+n-4) \leq 8(3n-4) = O(\sqrt{N});\\
spt(G_3) &\leq 8(n+n-2) \leq 8(2n-2) = O(\sqrt{N});\\
t_2(G_3) &\leq 4(n+n-2) \leq 4(2n-2) = O(\sqrt{N}).
\end{aligned}$$

If $n=p$, the new tightness values $t_3(G_3)$ and $t_4(G_3)$ are also given as follows:

$$\begin{aligned}
t_3(G_3) &= \mathrm{m}(\chi-1) \leq 8(2n-3) = O(\sqrt{N});\\
t_4(G_3) &= (D+1)(\chi-1) \leq 4(2n-3) = O(\sqrt{N}).
\end{aligned}$$

The tightness values t_1, stt, spt, t_2, t_3, and t_4 are bounded by $O(\sqrt{N})$, and hence G_3 can be used as a model for MINs. Figure 5.3 gives the Line graph of Cartesian product $K_{1,2}\square K_{1,3}$.

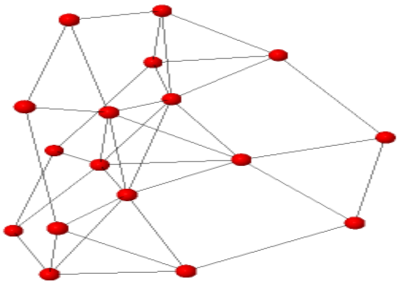


Figure 5.3 Line graph of Cartesian product $K_{1,2}\square K_{1,3}$.

5.3.1.4 Line graph of Complete graph K_n: Consider G_4 = $L(K_n)$ = Line graph of Complete graph of K_n, for $n > 2$. Table 5.8 summarizes all the relevant properties of G_4 .

Table 5.8 Line graph of Complete graph K_n.

N	D	m	Δ	λ_1
$\frac{n^2-n}{2}$	≤ 2	≤ 3	$2(n-2)$	$2(n-2)$

The properties of the Complete graph K_n are given in Table 5.9.

Table 5.9 Complete graph K_n.

N^{OG}	Δ	Is Regular?
n	$n-1$	Yes

The chromatic number of the Line graph of Complete graph K_n is calculated as follows: from Theorem 5.1.20 and Corollary 5.1.21, if N^{OG} is odd, the edge chromatic number (chromatic index) of K_n is n, and $n-1$ if N^{OG} is even. Also, from Definition 5.1.2, the edge chromatic number $\chi^{'}(K_n)$ = $\chi(L(K_n))$. The tightness values are given as follows: :

$$\begin{aligned} t_1(G_4) &\leq 3 \times 2(n-2) \leq 6(n-2) = O(\sqrt{N}); \\ stt(G_4) &\leq 3 \times 2(n-2) \leq 6(n-2) = O(\sqrt{N}); \\ spt(G_4) &\leq 3 \times 2(n-2) \leq 6(n-2) = O(\sqrt{N}); \\ t_2(G_4) &\leq 3 \times 2(n-2) \leq 6(n-2) = O(\sqrt{N}). \end{aligned}$$

The new tightness values $t_3(G_4)$ and $t_4(G_4)$ are also given as follows:

$$\begin{aligned} t_3(G_4) &= \mathsf{m}(\chi-1) \leq 3(n-2) \leq 3n-6 = O(\sqrt{N}); \text{ if n is even} \\ t_3(G_4) &= \mathsf{m}(\chi-1) \leq 3(n-1) \leq 3n-3 = O(\sqrt{N}); \text{ if n is odd} \\ t_4(G_4) &= (D+1)(\chi-1) \leq 3(n-2) \leq 3n-6 = O(\sqrt{N}); \text{ if n is even} \\ t_4(G_4) &= (D+1)(\chi-1) \leq 3(n-1) \leq 3n-3 = O(\sqrt{N}); \text{ if n is odd} \end{aligned}$$

The tightness values t_1, stt, spt, t_2, t_3, and t_4 are bounded by $O(\sqrt{N})$, and hence G_4 can be used as a model for MINs. The Line graph of Complete graph K_5 is given in Figure 5.4.

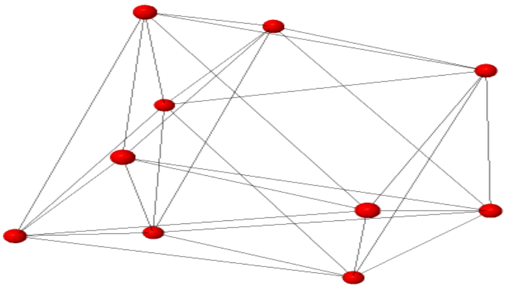


Figure 5.4 Line graph of Complete graph K_5.

5.3.1.5 Line graph of Complete Bipartite graph $K_{n,n}$: Consider G_5 = $L(K_{n,n})$ = Line graph of Complete Bipartite graph $K_{n,n}$. Table 5.10 summarizes all the relevant properties of G_5 .

Table 5.10 Line graph of Complete Bipartite graph $K_{n,n}$.

N	D	m	Δ	λ_1
n^2	2	3	$2(n-1)$	$2(n-1)$

The properties of the Complete Bipartite graph $K_{n,n}$ are given in Table 5.11.

Table 5.11 Complete Bipartite graph $K_{n,n}$.

N^{OG}	Δ	Is Regular?
$2*n$	n	Yes

The chromatic number of the Line graph of Complete Bipartite graph $K_{n,n}$ is calculated as follows: from Theorem 5.1.19, the edge chromatic number (chromatic index) of $K_{n,n}$ is n. Also, from Definition 5.1.2, the edge chromatic number $\chi^{'}(K_{n,n})$ = $\chi(L(K_{n,n}))$. The tightness values are given as follows:

$$t_1(G_5) = 3 \times 2(n-1) = 6(n-1) = O(\sqrt{N});$$
$$stt(G_5) = 3 \times 2(n-1) = 6(n-1) = O(\sqrt{N});$$
$$spt(G_5) = 3 \times 2(n-1) = 6(n-1) = O(\sqrt{N});$$
$$t_2(G_5) = 3 \times 2(n-1) = 6(n-1) = O(\sqrt{N}).$$

The new tightness values $t_3(G_5)$ and $t_4(G_5)$ are also given as follows:

$$t_3(G_5) = \mathrm{m}(\chi - 1) = 3(n-1) = 3n-3 = O(\sqrt{N});$$
$$t_4(G_5) = (D+1)(\chi - 1) = 3(n-1) = 3n-3 = O(\sqrt{N}).$$

The tightness values t_1, stt, spt, t_2, t_3, and t_4 are bounded by $O(\sqrt{N})$, and hence G_5 can be used as a model for MINs. The Line graph of Complete Bipartite graph $K_{3,3}$ is shown in Figure 5.5.

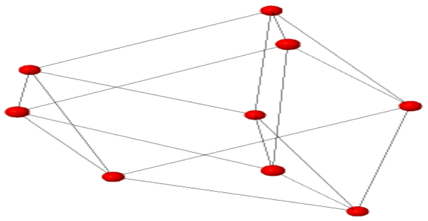


Figure 5.5 Line graph of Complete Bipartite graph $K_{3,3}$.

5.3.1.6 Line graph of Crown graph $K_{n,n} - I$: Consider G_6 = $K_{n,n} - I$ = Line graph of Crown graph $K_{n,n} - I$. Table 5.12 summarizes all the relevant properties of G_6 .

Table 5.12 Line graph of Crown graph $K_{n,n} - I$.

N	D	m	Δ	λ_1
$n^2 - n$	3	4	$2(n-2)$	$2(n-2)$

The properties of the Crown graph $K_{n,n} - I$ are given in Table 5.13.

Table 5.13 Crown graph $K_{n,n} - I$.

N^{OG}	Δ	Is Regular?
$2 * n$	$n-1$	Yes

The chromatic number of the Line graph of Crown graph $K_{n,n} - I$ is calculated as shown below. The Crown graph $K_{n,n} - I$ is a $(n-1)$-regular bipartite graph with the number of vertices is given as $2n$. It can be observed that the edge chromatic number (chromatic index) of $K_{n,n} - I$ is $n-1$ (from Theorem 5.1.19). Also, from Definition 5.1.2 the edge chromatic number $\chi^{'}(K_{n,n} - I) = \chi(L(K_{n,n} - I))$. The tightness values are given as follows:

$$t_1(G_6) = 4 \times 2(n-2) = 8n - 8 = O(\sqrt{N});$$
$$stt(G_6) = 4 \times 2(n-2) = 8n - 8 = O(\sqrt{N});$$
$$spt(G_6) = 4 \times 2(n-2) = 8n - 8 = O(\sqrt{N});$$
$$t_2(G_6) = 4 \times 2(n-2) = 8n - 8 = O(\sqrt{N}).$$

The new tightness values $t_3(G_6)$ and $t_4(G_6)$ are also given as follows:

$$t_3(G_6) = \mathsf{m}(\chi - 1) = 4(n-2) = 4n - 8 = O(\sqrt{N});$$
$$t_4(G_6) = (D+1)(\chi - 1) = 4(n-2) = 4n - 8 = O(\sqrt{N}).$$

The tightness values t_1, stt, spt, t_2, t_3, and t_4 are bounded by $O(\sqrt{N})$, and hence G_6 can be used as a model for MINs. In Figure 5.6 the Line graph of Crown graph $K_{4,4} - I$ is shown.

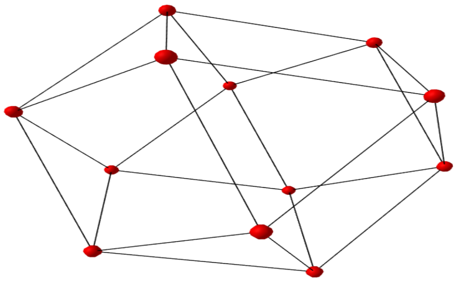


Figure 5.6 Line graph of Crown graph $K_{4,4} - I$.

5.3.1.7 Line graph of Complete Tripartite graph $K_{n,n,n}$: Consider $G_7 = L(K_{n,n,n})$ = Line graph of Complete Tripartite graph $K_{n,n,n}$. Table 5.14 summarizes all the relevant properties of G_7.

Table 5.14 Line graph of Complete Tripartite graph $K_{n,n,n}$.

N	D	m	Δ	λ_1
$3n^2$	2	4	$4n-2$	$4n-2$

The properties of the Complete Tripartite graph $K_{n,n,n}$ are given in Table 5.15.

Table 5.15 Complete Tripartite graph $K_{n,n,n}$.

N^{OG}	Δ	Is Regular?
$3*n$	$2*n$	Yes

The chromatic number of the Line graph of Complete Tripartite graph $K_{n,n,n}$ is calculated as shown below. It can be observed that the edge chromatic number (chromatic index) of Complete Tripartite graph $K_{n,n,n}$ is $2n+1$ if the number of vertices is odd (from Corollary 5.1.21), and $2n$ if number of vertices is even (from Theorem 5.1.20). Also, from Definition 5.1.2 the edge chromatic number $\chi'(K_{n,n,n})$ = $\chi(L(K_{n,n,n}))$. The tightness values are given as follows:

$$\begin{aligned}
t_1(G_7) &= 4\times(4n-2) = 16n-8 = O(\sqrt{N});\\
stt(G_7) &= 3\times(4n-2) = 12n-6 = O(\sqrt{N});\\
spt(G_7) &= 4\times(4n-2) = 16n-8 = O(\sqrt{N});\\
t_2(G_7) &= 3\times(4n-2) = 12n-6 = O(\sqrt{N}).
\end{aligned}$$

The new tightness values $t_3(G_7)$ and $t_4(G_7)$ are also given as follows:

$$\begin{aligned}
&t_3(G_7) = \mathsf{m}(\chi-1) = 4(2n-1) = 8n-4 = O(\sqrt{N}); \text{ if n is even}\\
&t_3(G_7) = \mathsf{m}(\chi-1) = 4(2n) = 8n = O(\sqrt{N}); \text{ if n is odd}\\
&t_4(G_7) = (D+1)(\chi-1) = 3(2n-1) = 6n-3 = O(\sqrt{N}); \text{ if n is even}\\
&t_4(G_7) = (D+1)(\chi-1) = 3(2n) = 6n = O(\sqrt{N}); \text{ if n is odd}
\end{aligned}$$

The tightness values t_1, stt, spt, t_2, t_3, and t_4 are bounded by $O(\sqrt{N})$, and hence G_7 can be used as a model for MINs. The Line graph of Complete Tripartite graph $K_{2,2,2}$ is given in Figure 5.7.

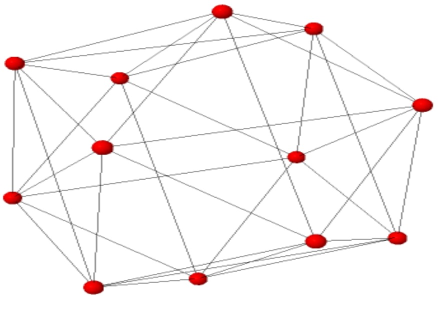


Figure 5.7 Line graph of Complete Tripartite graph $K_{2,2,2}$.

Example 5.3.2. The set $S^{O(\sqrt[3]{N})}$ contains the following graphs:

5.3.2.1 Line graph of Johnson graph $J(n,2)$.

5.3.2.2 Line graph of Cartesian product $K_{1,n-1}\Box K_p$.

5.3.2.3 Line graph of Rook graph $K_n\Box K_n$.

5.3.2.4 Line graph of Total graph of complete bipartite graph $K_{n,n}$.

5.3.2.5 Line graph of Total graph of complete graph K_n.

5.3.2.1 Line graph of Johnson graph $J(n,2)$: Consider $G_1 = L(J(n,2))$ = Line graph of Johnson graph $J(n,2)$, for $n > 3$. The graph parameters of G_1 are given in Table 5.16.

Table 5.16 Line graph of Johnson graph $J(n,2)$.

N	D	m	Δ	λ_1
$\frac{n^3-3n^2+2n}{2}$	≤ 3	≤ 4	$4n-10$	$4n-10$

The properties of Johnson graph $J(n,2)$ are given in Table 5.17.

Table 5.17 Johnson graph $J(n,2)$.

N^{OG}	Δ	Is Regular?
$\binom{n}{2}$	$2(n-2)$	Yes

It can be observed that the edge chromatic number (chromatic index) of $J(n,2)$ is $2(n-2)+1$ if the number of vertices is odd (from Corollary 5.1.21), and $2(n-2)$ if number of vertices is even (from Theorem 5.1.20). Also, from Definition 5.1.2 the edge chromatic number $\chi^{'}(J(n,2)) = \chi(L(J(n,2)))$. The tightness values are given as follows:

$$\begin{gathered} t_1(G_1) \leq 4(4n-10) \leq 16n-40 = O(\sqrt[3]{N}); \\ stt(G_1) \leq 4(4n-10) \leq 16n-40 = O(\sqrt[3]{N}); \\ spt(G_1) \leq 4(4n-10) \leq 16n-40 = O(\sqrt[3]{N}); \\ t_2(G_1) \leq 4(4n-10) \leq 16n-40 = O(\sqrt[3]{N}). \end{gathered}$$

The new tightness values $t_3(G_1)$ and $t_4(G_1)$ are also given as follows:

$t_3(G_1) = \mathsf{m}(\chi - 1) \leq 4(2n-5) \leq 8n-20 = O(\sqrt[3]{N})$; if no. of vertices is even
$t_3(G_1) = \mathsf{m}(\chi - 1) \leq 4(2n-4) \leq 8n-16 = O(\sqrt[3]{N})$; if no. of vertices is odd
$t_4(G_1) = (D+1)(\chi - 1) \leq 4(2n-5) \leq 8n-20 = O(\sqrt[3]{N})$; if no. of vertices is even
$t_4(G_1) = (D+1)(\chi - 1) \leq 4(2n-4) \leq 8n-16 = O(\sqrt[3]{N})$; if no. of vertices is odd

The tightness values t_1, stt, spt, t_2, t_3, and t_4 are bounded by $O(\sqrt[3]{N})$, and hence G_1 can be used as a model for MINs. The Line graph of Johnson graph $J(4,2)$ is provided in Figure 5.8.

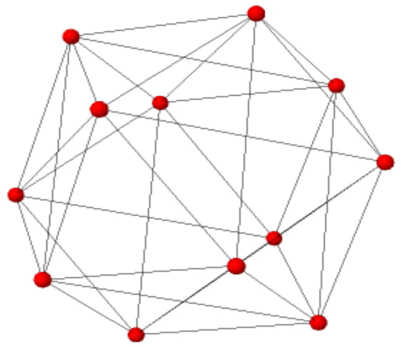


Figure 5.8 Line graph of Johnson graph $J(4,2)$.

5.3.2.2 Line graph of Cartesian product $K_{1,n-1}\Box K_p$: Consider $G_2 = L(K_{1,n-1}\Box K_p)$ = Line graph of Cartesian product $K_{1,n-1}\Box K_p$. The graph parameters of G_2 are given in Table 5.18.

Table 5.18 Line graph of Cartesian product $K_{1,n-1}\Box K_p$.

N	D	m	Δ	λ_1
$\frac{np^2+np-2p}{2}$	≤ 4	≤ 7	$2n+2p-6$	$n+2p-4$

The properties of Cartesian product $K_{1,n-1}\Box K_p$ are given in Table 5.19.

Table 5.19 Cartesian product $K_{1,n-1}\Box K_p$.

N^{OG}	Δ	Is Regular?
$n*p$	$n+p-2$	No

The chromatic number of the Line graph of Cartesian product $K_{1,n-1}\Box K_p$ is calculated as follows: from Theorem 5.1.22, the edge chromatic number (chromatic index) of $K_{1,n-1}\Box K_p$ is $n+p-2$. Also, from Definition 5.1.2, the edge chromatic number $\chi'(K_{1,n-1}\Box K_p) = \chi(L(K_{1,n-1}\Box K_p))$. If $n = p$, then the tightness values are given as follows:

$$\begin{gathered}
t_1(G_2) \leq 7(2n+2n-6) \leq 7(4n-6) = O(\sqrt[3]{N});\\
stt(G_2) \leq 5(2n+2n-6) \leq 5(4n-6) = O(\sqrt[3]{N});\\
spt(G_2) \leq 7(n+2n-4) \leq 7(3n-4) = O(\sqrt[3]{N});\\
t_2(G_2) \leq 5(n+2n-4) \leq 5(3n-4) = O(\sqrt[3]{N}).
\end{gathered}$$

If $n = p$, the new tightness values $t_3(G_2)$ and $t_4(G_2)$ are also given as follows:

$$\begin{gathered}
t_3(G_2) = \mathrm{m}(\chi-1) \leq 7(n+n-3) \leq 7(2n-3) = O(\sqrt[3]{N});\\
t_4(G_2) = (D+1)(\chi-1) \leq 5(n+n-3) \leq 5(2n-3) = O(\sqrt[3]{N})
\end{gathered}$$

The tightness values t_1, stt, spt, t_2, t_3, and t_4 are bounded by $O(\sqrt[3]{N})$, and hence G_2 can be used as a model for MINs. In Figure 5.9, the Line graph of Cartesian product $K_{1,2}\Box K_3$ is shown.

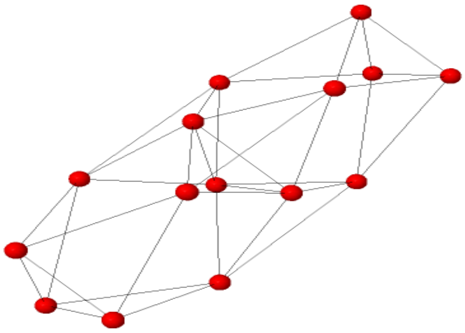


Figure 5.9 Line graph of Cartesian product $K_{1,2}\Box K_3$.

5.3.2.3 Line graph of Rook graph $K_n\Box K_n$: Consider $G_3 = L(K_n\Box K_n)$ = Line graph of Rook graph $K_n\Box K_n$, for $(n > 2)$. The graph parameters of G_3 are given in Table 5.20.

Table 5.20 Line graph of Rook graph $K_n\Box K_n$.

N	D	m	Δ	λ_1
n^3-n^2	3	4	$4n-6$	$4n-6$

The properties of Rook graph $K_n \Box K_n$ are given in Table 5.21.

Table 5.21 Rook graph $K_n \Box K_n$.

N^{OG}	Δ	Is Regular?
n^2	$2*n-2$	Yes

The chromatic number of the Line graph of Rook graph $K_n \Box K_n$ is calculated as follows: it can be observed that the edge chromatic number (chromatic index) of $K_n \Box K_n$ is $2n-1$ if the number of vertices is odd (from Corollary 5.1.21), and $2n-2$ if number of vertices is even (from Theorem 5.1.20). Also, from Definition 5.1.2 the edge chromatic number $\chi^{'}(K_n \Box K_n) = \chi(L(K_n \Box K_n))$. The tightness values are given as follows:

$$t_1(G_3) = 4(4n-6) = 16n-24 = O(\sqrt[3]{N});$$
$$stt(G_3) = 4(4n-6) = 16n-24 = O(\sqrt[3]{N});$$
$$spt(G_3) = 4(4n-6) = 16n-24 = O(\sqrt[3]{N});$$
$$t_2(G_3) = 4(4n-6) = 16n-24 = O(\sqrt[3]{N}).$$

The new tightness values $t_3(G_3)$ and $t_4(G_3)$ are also given as follows:

$t_3(G_3) = \mathrm{m}(\chi-1) = 4(2n-3) = 8n-12 = O(\sqrt[3]{N})$; if n is even
$t_3(G_3) = \mathrm{m}(\chi-1) = 4(2n-2) = 8n-8 = O(\sqrt[3]{N})$; if n is odd
$t_4(G_3) = (D+1)(\chi-1) = 4(2n-3) = 8n-12 = O(\sqrt[3]{N})$; if n is even
$t_4(G_3) = (D+1)(\chi-1) = 4(2n-2) = 8n-8 = O(\sqrt[3]{N})$; if n is odd

The tightness values t_1, stt, spt, t_2, t_3, and t_4 are bounded by $O(\sqrt[3]{N})$, and hence G_3 can be used as a model for MINs. The Line graph of Rook graph $K_3 \Box K_3$ is shown in Figure 5.10.

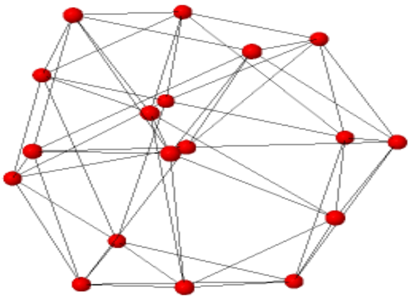


Figure 5.10 Line graph of Rook graph $K_3 \Box K_3$.

5.3.2.4 Line graph of Total graph of complete bipartite graph $K_{n,n}$: Consider G_4 = $L(T(K_{n,n}))$ = Line graph of Total graph of complete bipartite graph $K_{n,n}$. The graph parameters of G_4 are given in Table 5.22.

Table 5.22 Line graph of Total graph of complete bipartite graph $K_{n,n}$.

N	D	m	Δ	λ_1
n^3+2n^2	≤ 3	≤ 7	$4n-2$	$4n-2$

The properties of Total graph of complete bipartite graph $K_{n,n}$ are given in Table 5.23.

Table 5.23 Total graph of complete bipartite graph $K_{n,n}$.

N^{OG}	Δ	Is Regular?
n^2+2n	$2*n$	Yes

The chromatic number of the Line graph of Total graph of complete bipartite graph $K_{n,n}$ is calculated as follows: it can be observed that the edge chromatic number (chromatic index) of Total graph of complete bipartite graph $K_{n,n}$ is $2n+1$ if the number of vertices is odd (from Corollary 5.1.21), and $2n$ if number of vertices is even (from Theorem 5.1.20). Also, from Definition 5.1.2 the edge chromatic number $\chi^{'}(T(K_{n,n}))$ = $\chi(L(T(K_{n,n})))$. The tightness values are given as follows:

$$t_1(G_4) \leq 7(4n-2) \leq 28n-14 = O(\sqrt[3]{N});$$
$$stt(G_4) \leq 4(4n-2) \leq 16n-8 = O(\sqrt[3]{N});$$
$$spt(G_4) \leq 7(4n-2) \leq 28n-14 = O(\sqrt[3]{N});$$
$$t_2(G_4) \leq 4(4n-2) \leq 16n-8 = O(\sqrt[3]{N}).$$

The new tightness values $t_3(G_4)$ and $t_4(G_4)$ are also given as follows:

$$t_3(G_4) = \mathrm{m}(\chi-1) \leq 7(2n-1) \leq 14n-7 = O(\sqrt[3]{N}); \text{ if n is even}$$
$$t_3(G_4) = \mathrm{m}(\chi-1) \leq 7(2n) \leq 14n = O(\sqrt[3]{N}); \text{ if n is odd}$$
$$t_4(G_4) = (D+1)(\chi-1) \leq 4(2n-1) \leq 8n-4 = O(\sqrt[3]{N}); \text{ if n is even}$$
$$t_4(G_4) = (D+1)(\chi-1) \leq 4(2n) \leq 8n = O(\sqrt[3]{N}); \text{ if n is odd}$$

The tightness values t_1, stt, spt, t_2, t_3, and t_4 are bounded by $O(\sqrt[3]{N})$, and hence G_4 can be used as a model for MINs. The Line graph of Total graph of complete bipartite graph $K_{2,2}$ is given in Figure 5.11.

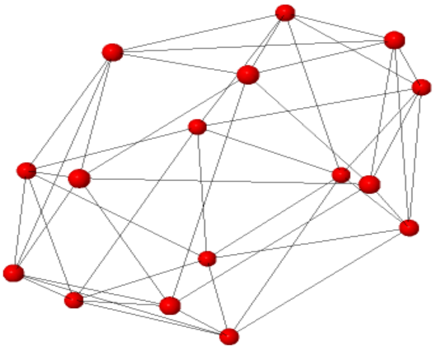


Figure 5.11 Line graph of Total graph of complete bipartite graph $K_{2,2}$.

5.3.2.5 Line graph of Total graph of complete graph K_n: Consider $G_5 = L(T(K_n))$ = Line graph of Total graph of complete graph K_n. The graph parameters of G_5 are given in Table 5.24.

Table 5.24 Line graph of Total graph of complete graph K_n.

N	D	m	Δ	λ_1
$\frac{n^3-n}{2}$	≤ 2	≤ 4	$4n-6$	$4n-6$

The properties of Total graph of complete graph K_n are given in Table 5.25.

Table 5.25 Total graph of complete graph K_n.

N^{OG}	Δ	Is Regular?
$\frac{n^2+n}{2}$	$2n-2$	Yes

The chromatic number of the Line graph of Total graph of complete graph K_n is calculated as follows: it can be observed that the edge chromatic number (chromatic index) of Total graph of complete graph K_n is $2n-1$ if the number of vertices is odd (from Corollary 5.1.21), and $2n-2$ if number of vertices is even (from Theorem 5.1.20). Also, from Definition 5.1.2 the edge chromatic number $\chi^{'}(T(K_n)) = \chi(L(T(K_n)))$. The tightness values are given as follows:

$$\begin{aligned} t_1(G_5) &\leq 4(4n-6) \leq 16n-24 = O(\sqrt[3]{N}); \\ stt(G_5) &\leq 3(4n-6) \leq 12n-18 = O(\sqrt[3]{N}); \\ spt(G_5) &\leq 4(4n-6) \leq 16n-24 = O(\sqrt[3]{N}); \\ t_2(G_5) &\leq 3(4n-6) \leq 12n-18 = O(\sqrt[3]{N}). \end{aligned}$$

The new tightness values $t_3(G_5)$ and $t_4(G_5)$ are also given as follows:

$$\begin{aligned} t_3(G_5) &= \mathsf{m}(\chi-1) \leq 4(2n-3) \leq 8n-12 = O(\sqrt[3]{N}); \text{ if n is even} \\ t_3(G_5) &= \mathsf{m}(\chi-1) \leq 4(2n-2) \leq 8n-8 = O(\sqrt[3]{N}); \text{ if n is odd} \\ t_4(G_5) &= (D+1)(\chi-1) \leq 3(2n-3) \leq 6n-9 = O(\sqrt[3]{N}); \text{ if n is even} \\ t_4(G_5) &= (D+1)(\chi-1) \leq 3(2n-2) \leq 6n-6 = O(\sqrt[3]{N}); \text{ if n is odd} \end{aligned}$$

The tightness values t_1, stt, spt, t_2, t_3, and t_4 are bounded by $O(\sqrt[3]{N})$, and hence G_5 can be used as a model for MINs. Figure 5.12 shows the Line graph of Total graph of complete graph K_3.

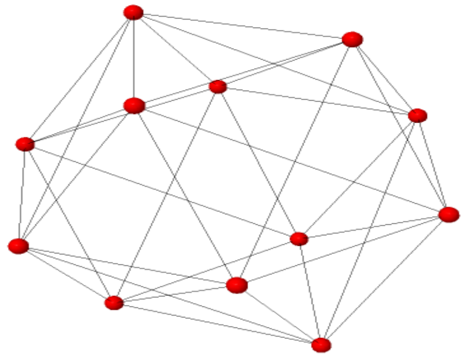


Figure 5.12 Line graph of Total graph of complete graph K_3.

Example 5.3.3. The Line graph of Johnson graph $J(n,3)$ belongs to the set $S^{O(\sqrt[4]{N})}$

Consider G_1 = $L(J(n,3))$ = Line graph of Johnson graph $J(n,3)$, for $n > 5$. The graph parameters of G_1 are given in Table 5.26.

Table 5.26 The Line graph of Johnson graph $J(n,3)$.

N	D	m	Δ	λ_1
$\frac{n^4-6n^3+11n^2-6n}{4}$	≤ 4	≤ 5	$6n-20$	$6n-20$

The properties of Johnson graph $J(n,3)$ are given in Table 5.27.

Table 5.27 Johnson graph $J(n,3)$.

N^{OG}	Δ	Is Regular?
$\binom{n}{3}$	$3(n-3)$	Yes

The chromatic number of the Line graph of Johnson graph $J(n,3)$ is calculated as follows: it can be observed that the edge chromatic number (chromatic index) of $J(n,3)$ is $3(n-3)+1$ if the number of vertices is odd (from Corollary 5.1.21), and $3(n-3)$ if number of vertices is even(from Theorem 5.1.20). Also, from Definition 5.1.2 the edge chromatic number $\chi^{'}(J(n,3))$ = $\chi(L(J(n,3)))$. The tightness values are given as follows:

$$t_1(G_1) \leq 5(6n-20) \leq 30n-100 = O(\sqrt[4]{N});$$
$$stt(G_1) \leq 5(6n-20) \leq 30n-100 = O(\sqrt[4]{N});$$
$$spt(G_1) \leq 5(6n-20) \leq 30n-100 = O(\sqrt[4]{N});$$
$$t_2(G_1) \leq 5(6n-20) \leq 30n-100 = O(\sqrt[4]{N}).$$

The new tightness values $t_3(G_1)$ and $t_4(G_1)$ are also given as follows:

$t_3(G_1) = \mathrm{m}(\chi-1) \leq 5(3n-10) = O(\sqrt[4]{N})$; if no. of vertices is even
$t_3(G_1) = \mathrm{m}(\chi-1) \leq 5(3n-9) = O(\sqrt[4]{N})$; if no. of vertices is odd
$t_4(G_1) = (D+1)(\chi-1) \leq 5(3n-10) = O(\sqrt[4]{N})$; if no. of vertices is even
$t_4(G_1) = (D+1)(\chi-1) \leq 5(3n-9) = O(\sqrt[4]{N})$; if no. of vertices is odd

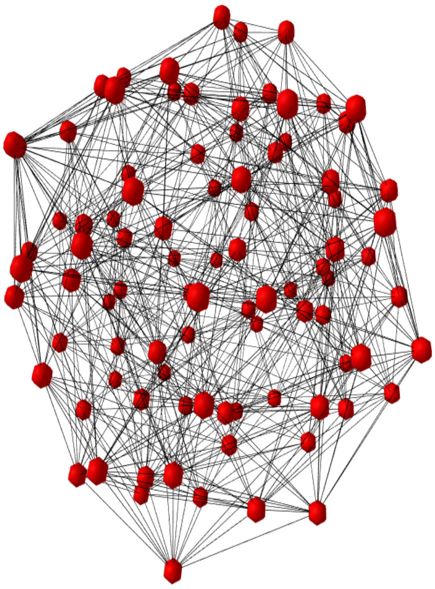


Figure 5.13 Line graph of Johnson graph $J(6,3)$.

The tightness values t_1, stt, spt, t_2, t_3, and t_4 are bounded by $O(\sqrt[4]{N})$, and hence G_1 can be used as a model for MINs. The Line graph of Johnson graph $J(6,3)$ is shown in Figure 5.13.

CHAPTER 6

DYNAMIC LOAD BALANCING OF INTEGER LOADS

This chapter presents a load balancing algorithm as a modified approach to the existing load balancing algorithm in Stevanović (2014). The proposed algorithm results in a balancing flow with a lesser l_2-norm than the l_2-norm of the balancing flow generated by the existing algorithm. Further, we show that the load balancing by the proposed algorithm is done in $O(n^3)$ time.

6.1 PRELIMINARIES

Given a connected graph $G = (V,E)$, the processor network is represented as follows: The processors are represented by the graph's nodes, while the edges represent the communication channels. Each node $v_i \in V$ is given a unit size token w_i, an independent task. The standard approach for dynamic load balancing is to split the balancing process into two steps as in Hu et al. (1998). The first step generates a balancing flow that specifies the exact amounts of load to be exchanged between a processor and its neighboring processors; the second step performs the actual load movement. The following assumptions from Diekmann et al. (1999) are considered in this chapter: The loads(tokens) are non-negative integers and each processor(node) has the same processing capacity. There is no load generated or absorbed throughout the balancing process. Non-adjacent nodes cannot communicate with each other. Any amount of tokens can be moved from a node to its neighboring nodes in each step. The goal is to devise an algorithm for moving loads across edges so that the load on each node is approximately equal.

An integral graph is defined by Harary and Schwenk (1974) as a graph whose graph spectrum consists entirely of integers. Cvetković et al. (2010) presented a dynamic load balancing algorithm and claimed that their algorithm would work in integer arithmetic only if the network of processors was represented as an integral graph. Stevanović (2014) observed that the algorithm in Cvetković et al. (2010) produced balancing flows with non-integer values, hindering the emphasis on integer arithmetic. A set of balancing flows transferring a unit load from a selected vertex to every other vertices of the graph was necessary to manage all integer load distributions. As a result, *Algorithm 2* that executes in integer arithmetic was proposed by Stevanović (2014) for load balancing in any given network. But when compared to Hu et al. (1998)'s algorithm, *Algorithm 2* from Stevanović (2014) gave sub optimal balancing flows in terms of l_2-norm. This chapter presents an algorithm as a modified approach to *Algorithm 2* from Stevanović (2014). The resulting balancing flow has the l_2-norm less than the l_2-norm of the balancing flow generated by *Algorithm 2* in Stevanović (2014). The proposed algorithm assigns the loads to the nodes so that the balancing flow computed has a minimum l_2-norm. The algorithm is presented in two parts: The first algorithm does the parameter computation and the second algorithm then uses these parameters to find the balancing flow.

6.1.1 NOTATIONS

In this chapter, we follow the notations that are used in Stevanović (2014). Let $G = (V,E)$ be a directed graph, which is weakly connected, with the vertices in V representing processors and E edges representing the links. Here, $|V| = n$ and $|E| = m$. The vector $w \in \mathbb{Z}^n$ represents the vector of load values, where the load at processor $i \in V$ is denoted as w_i. The vector $\overline{w} := \frac{1}{n}(\sum_{i=1}^{n})(1,...,1)$ represents the average load. Let N be an $n \times m$ vertex-edge incidence matrix of G. For the two nodes incident to the corresponding edge, each column has exactly two non-zero entries, '1' and '-1' with the remaining entries as 0.

Consider the flow $x \in \mathbb{Z}^E$ on the graph edges. The flow direction is determined by the signs of the entries in x and the directions in N, i.e., $x_e > 0$ indicates a flow along the edge e's direction, where $e \in E$, $x_e < 0$ indicates flow along the opposite direction. The balancing flow is defined as the flow x on the edges of E, represented as $f \in \mathbb{Z}^E$ and satisfying equation given below:

$$Nf = \overline{w} - w \tag{6.1.1}$$

The difference between a node's initial load and the mean load value is the flow balance at that node, i.e., the load is globally balanced after sending exactly f_e tokens through every edge $e \in E$, as expressed in (6.1.1). The balancing flow with the smallest l_2-norm

$\|f\|_2 = (\sum_{i=1}^{m} f_i^2)^{1/2}$ is computed among all the balancing flows satisfying (6.1.1).

6.2 EXISTING ALGORITHMS

Cvetković et al. (2010) proposed an algorithm for dynamic load balancing of the network of processors. The proposed algorithm used the eigenvectors of the graph that modeled the network. They claimed that the algorithm works in integer arithmetic only if the network of processors was represented as an integral graph. The algorithm in Cvetković et al. (2010) was analyzed by Stevanović (2014) as *Algorithm1'* and is described as follows: Consider the adjacency matrix for a regular integral graph G. The integral eigenvectors (excluding the all-one vector j) of the adjacency matrix of G are stored in an $n \times (n-1)$ matrix B. The entries of eigenvectors are assigned as weights to the graph's vertices. The balancing flow is then computed. The edges that transmit the loads from every source (a vertex with positive weight) to a sink (a vertex with negative weight) are considered. The flow is considered as '1' if it is traversing along the direction of the underlying edge. The flow is '-1' if it is traversing against the direction of the underlying edge. The flow across the other edges is considered as zero. For every eigenvector, the flow is computed, and the entries are stored in the $m \times (n-1)$ matrix F.

Now delete the last row from the matrices B and N, and denote them by B_* and N_*, respectively. The uniform load distribution is known to be given by the zero vector for every column of B. In addition, the flow vectors in F help to balance the loads in B's columns. Hence, $-B = NF$, and $-B_* = N_*F$ as seen in (Stevanović (2014), eq.2). The vector w represents the loads assigned to the processors. The vector of uniform load distribution is represented by $\overline{w}$. The sum of the components of the vector $(\overline{w} - w)$ is zero because the sums of components of w and $\overline{w}$ are equal. Now delete the last element from $(\overline{w} - w)$ and denote it as $(\overline{w} - w)_*$. From *Algorithm 1'*, the balancing flow is obtained as $f = -FB_*^{-1}(\overline{w} - w)_*$.

Dragan Stevanović observed that since matrix B_*^{-1} is involved in calculating the balancing flow, *Algorithm1'* failed to execute in integer arithmetic even for the integral graphs. It was concluded that for the integer loads, *Algorithm1'*, produced the balancing flow in terms of integers if and only if it satisfied $FB_*^{-1} \in \mathbb{Z}^{m\times(n-1)}$ as seen in (Stevanović (2014), eq.3). From (Stevanović (2014), eq.4), it is observed that

$$N_*C = -N_*FB_*^{-1} = B_*B_*^{-1} = I \qquad (6.2.1)$$

Considering the necessary and sufficient condition from (Stevanović (2014), eq.3), Dragan Stevanović restated *Algorithm 1'* as *Algorithm 2*. It states that for w and $\overline{w}$, the flow vector that balances the load is obtained as $f = C(\overline{w} - w)_*$, if matrix $C \in \mathbb{Z}^{m\times(n-1)}$ satisfies (6.2.1). It is clear that the resulting balancing flow depends only on $C = FB_*^{-1}$ that must satisfy (6.2.1). Dragan Stevanović stated that for every connected graph G, *Algorithm 2* executes in integer arithmetic due to (Stevanović (2014), Claim 1). It states that there exists a matrix $C \in \mathbb{Z}^{m\times(n-1)}$ satisfying (6.2.1) for every directed graph that is weakly connected.

The matrix C is obtained as shown here: Consider G to be a weakly connected directed graph with the vertex v corresponding to the matrix N's last row. Consider the underlying undirected graph of G. A path p_i from v to i exists for every $i \in V \setminus \{v\}$ because G is weakly connected. The matrix C is computed by finding a path from v to the other vertices in the underlying undirected graph. The directed graph indicates if the path traverses the edge in the same direction or the opposite direction. Let c_i be an m-dimensional vector with $(-1,0,1)$ entries for $i \in V \setminus \{v\}$ defined as:

$$(c_i)_k = \begin{cases} 0, & \text{if path } p_i \text{ does not traverse along edge } k \\ 1, & \text{if path } p_i \text{ traverses in the same direction as edge } k \\ -1, & \text{if path } p_i \text{ traverses in the opposite direction of edge } k \end{cases}$$

To construct matrix C, for each $i \in V \setminus \{v\}$, we set the i^{th} column of C to vector c_i.

Let $||f_A||_2$ and $||f_B||_2$ represent the balancing flow generated by *Algorithm 2* from Stevanović (2014) and Hu et al. (1998). For the Petersen graph with the loads considered in Stevanović (2014), the l_2-norm is obtained as follows: $||f_A||_2 = 41.95 \approx 42$ and $||f_B||_2 = 28$. The l_2-norm of the balancing flow resulting from *Algorithm 2* is much more than the l_2-norm of the balancing flow resulting from Hu et al. (1998)'s algorithm. In this chapter we modify the existing algorithm such that the balancing flow generated has its l_2-norm less than the l_2-norm of the balancing flow produced by the existing algorithm.

6.3 THE PROPOSED ALGORITHMS FOR LOAD BALANCING

In this section, we present a brief description of the proposed algorithms to find the balancing flow along with the pseudo-codes of the algorithm in SageMath. We modify the existing algorithm such that the resulting balancing flow has a lesser l_2-norm. We achieve this by finding the spanning tree that acts as a backbone of the network. Next, we compute the order in which the loads are to be assigned such that the resulting balancing flow uses fewer network edges and has a lesser l_2-norm. The algorithm is implemented in SageMath. The algorithm is presented in two parts: Algorithm 3.1 computes the parameters needed to find the balancing flow. Algorithm 3.2 calculates the balancing flow using the parameters computed by Algorithm 3.1. The l_2-norm of the resulting balancing flow is much lesser than the l_2-norm of the balancing flow from *Algorithm 2* in Stevanović (2014). The rows of matrix C represent the edges of G, and columns represent the nodes of G. Algorithm 3.1 involves the construction of the spanning tree $G^{'}$ of G from the $m \times (n-1)$ matrix C. The following theorem proves that the tree resulting from matrix C is a spanning tree.

Theorem 6.3.1. *The graph constructed by considering the non-zero rows of matrix C is a spanning tree $G^{'}$ of G.*

Proof. Consider an $m \times (n-1)$ matrix C, whose rows and columns represent the edges and vertices, respectively. The entries in C are obtained by deleting v_n and finding its path to every other vertex by considering the underlying undirected graph of G. Since G is a connected graph, there exists a path from v_n to every remaining vertex in G. The non-zero rows in C represent the edges through which the path traverses from v_n to the remaining vertices. Let $K = V - \{v_n\}$ be the set of remaining vertices of G. We have to find the path from v_n to every vertex in K. Since $|K| = n-1$; we obtain $n-1$ unique paths from v_n to the remaining vertices of G. The incidence matrix N is an $n \times m$ matrix, whose columns represent the edges of G. The matrix N' is obtained by retaining those columns in N for which the rows in C contain non-zero entries and deleting the remaining columns. The matrix N' gives the vertex-edge incidence connection of G' and has $n-1$ non-zero columns in it. Thus the resulting graph G' has $n-1$ edges, which is the spanning tree of G. □

Algorithm 3.1 computes the distance of every vertex from v_n in graph $G^{'}$, for which the diameter of graph G is required. The following theorem gives the diameter of graph G.

Theorem 6.3.2. *The sum of the columns of matrix C with maximum non-zero entries is the diameter of graph G.*

Proof. The entries of matrix C represents the path from vertex v_n to every other vertex by considering the underlying undirected graph of G. The i^{th} column in C gives the path from vertex v_n to vertex i, where $i \in V$. Since G is a weakly connected graph, there exists at least one vertex, say v in $V \setminus \{v_n\}$, which is at the farthest distance from v_n. The path from v_n to v will have the most number of edges in it because v is the farthest vertex from v_n. Hence column v of matrix C will have the most number of non-zero entries, which gives the diameter of G when summed up. □

We present the algorithms as follows: Algorithm 3.1, named Parameter Computing Algorithm , constructs the spanning tree $G^{'}$ and the required parameters from G'. The parameters that are calculated and stored are: distance of every vertex from v_n stored in list D_L, index of the vertices in increasing order of their distances from v_n stored in list I_L, all the vertices adjacent to v_n stored in the list N_L, and degree of all vertices of the spanning tree $G^{'}$ stored in list Deg_L.

Since the rows of matrix C represent the edges of G, the spanning tree $G^{'}$ is constructed by considering only the non-zero rows of the matrix C. The incidence matrix N is updated by deleting those columns(edges) from N for which all the corresponding row entries in C are zero. The incidence matrix now has entries only for those rows(edges) in C that have non-zero entries in it. The SageMath built-in function $from\_incidence\_matrix(G',N)$, where $G^{'}$ is a spanning tree and N is an incidence matrix, is used to construct the spanning tree. The spanning tree $G^{'}$ gives a picture of how v_n is connected to the remaining vertices.

The spanning tree $G^{'}$ is constructed from matrix C in steps 1-6. The degree of all the vertices of G' are stored in the list Deg_L. The distance of all vertices to v_n in $G^{'}$ is computed from C and stored in the list D_L as shown in step 8. Through steps 9-18, the index of vertices based on the increasing order of the distances from v_n are found and stored in I_L. The neighbors of v_n are obtained using a SageMath built-in function $G'.neighbors(v_n)$ and stored in the list N_L. The Algorithm returns the four parameters D_L,I_L, N_L, and Deg_L.

Algorithm 3.2 first assigns the largest load to the vertex v_n and stores the load in $w_{pendant}$. The adjacent nodes of v_n that are pendant vertices are assigned with the loads and stored in the list $w_{pendant}$ as shown in steps 7-14. The remaining loads are assigned to the remaining non-pendant adjacent vertices and stored in the list w_{new}, as shown in steps 16-24. If vertex v_n does not have any pendant neighbors, then the adjacent non-pendant vertices of v_n are assigned loads in the increasing order of the indices. The loads are assigned to v_n's neighbors in the following order: the first adjacent vertex (with the lowest index) is assigned a load that is less than the load assigned to v_n but greater than the remaining loads. This is shown in steps 25-29. The assignment of loads to the remaining neighbors of v_n is done in steps 30-37. The remaining non-adjacent vertices are then assigned with the loads based on their indices in steps 40-42. Finally, the loads in w_{new} gives the order in which they are to be assigned such that the resulting balancing flow f_{new} has a minimum l_2-norm. The difference between the average and the initial loads are computed and stored in w_{diff} as in steps 43-44. The balancing flow f_{new} is computed as the product of $C \times w_{diff}$ in step 45. The l_2-norm of f_{new} is then computed using the built-in SageMath function $norm()$.

Algorithm 3.1: Parameter Computing Algorithm

```
Input  : An $m \times (n-1)$ matrix $C$ obtained by deleting vertex $v_n$, Incidence
         matrix $N$ of $G$.
Output: $D_L$, $I_L$, $N_L$, and $Deg_L$.
$C_{temp} \leftarrow C.apply\_map(lambda\ x:\ x * x)$        /* applying the function to each element of $C$ */
$S \leftarrow sum(C_{temp}.columns())$        /* $S \leftarrow$ temporary list */
for $i \leftarrow 0$ to $len(S)$ do        /* Construct $G^{'}$ by considering only non-zero row entries in $C$ */
    if $S[i] == 0$ then
        delete the $i^{th}$ column from $N$
$from\_incidence\_matrix(G^{'}, N)$        /* SageMath built-in function to obtain $G'$ */
$Deg_L \leftarrow G'.degree()$        /* degree of all vertices of $G'$ */
$D_L \leftarrow sum(C_{temp})$        /* $D_L \leftarrow$ distance from vertex $v_n$ to all other vertices */
for $i \leftarrow 0$ to $n-1$ do        /* Store the index of the vertices that are adjacent to $v_n$ */
    if $D_L[v_i] == 1$ then        /* vertex $v_i$ is at distance 1 from $v_n$ */
        $I_L \leftarrow$ store the index $i$
$D = max(D_L)$        /* $D$ is the diameter */
while $(D >= 2)$ do        /* Store the index of the vertices not adjacent to $v_n$ in list $I_L$ */
    for $i \leftarrow 0$ to $n-1$ do
        if $D_L[v_i] == D$ then        /* vertex $v_i$ is at distance $D$ from $v_n$ */
            $I_L \leftarrow$ store the index $i$
    $D \leftarrow D - 1$
$N_L \leftarrow G'.neighbors(v_n)$        /* Find neighbors of vertex $v_n$ using SageMath built-in function */
return $D_L, I_L, N_L$, and $Deg_L$
```

Algorithm 3.2 is named as Balancing Flow Algorithm. It uses the following parameters: matrix C, parameters D_L, I_L, N_L, and Deg_L computed by Parameter Computing Algorithm, the loads w sorted in non-decreasing order, the average load $\overline{w}$. The loads are assigned to the nodes such that the balancing flow has a minimum l_2 -norm.

Algorithm 3.2: Balancing Flow algorithm

```
Input  : An $m \times (n-1)$ matrix $C$ obtained by deleting vertex $v_n$, loads $w$ in
         non-decreasing order, average load $\overline{w}$, $D_L$,$I_L$,$N_L$, and $Deg_L$.
Output: The balancing flow $f_{new}$.
$w_{new} \leftarrow [\,]$    /* The list of loads assigned to the nodes of $G$ */
$w_{pendant} \leftarrow [\,]$    /* The list of loads assigned to the pendant nodes of $G'$ and also to $v_n$ */
$Length_{N_L} \leftarrow len(N_L)$
$w_{N_L} \leftarrow w[0 : Length_{N_L}]$    /* $w_{N_L}[\,]$ ←The list of loads to be assigned to the neighbors of $v_n$ */
$flag \leftarrow 0$
$j \leftarrow Length_{N_L} - 1$
$w_{pendant}[v_n] \leftarrow max(w)$    /* Assign the the largest load in $w$ to $v_n$ */
foreach $v_i$ in $N_L$ do
    if $Deg_L[v_i] == 1$ and $D_L[v_i] == 1$ then    /* $v_i$ is pendant vertex and at distance $D == 1$ */
        $w_{pendant}[v_i] \leftarrow w_{N_L}[j]$
        $N_L.remove(v_i)$    /* delete the neighbor assigned with load */
        $w_{N_L}.remove(w_{N_L}[j])$    /* delete the load assigned */
        $flag \leftarrow 1$
        $j \leftarrow j - 1$
$Length_{N_L} \leftarrow len(N_L)$
if $flag == 1$ then
    $j \leftarrow 0$
    while $v_i$ is in $N_L$ do    /* assign the loads to the remaining vertices adjacent to $v_n$ */
        if $Deg_L[v_i]\ != 1$ and $D_L[v_i] == 1$ then    /* $v_i$ is not pendant and is at distance $D == 1$ */
            $w_{new}[v_i] \leftarrow w_{N_L}[j]$
            $v_i \leftarrow$ next neighbor in $N_L$
            $j \leftarrow j + 1$
        else
            $v_i \leftarrow$ next neighbor in $N_L$
```

```
if $(flag == 0)$ then        /* If no pendant vertices, assign loads to vertices adjacent to $v_n$ */
    $v_i \leftarrow$ least indexed neighbor in $N_L$
    if $Deg_L[v_i]$ ! $= 1$ and $D_L[v_i] == 1$ then        /* $v_i$ is not pendant and is at distance $D == 1$ */
        $w_{new}[v_i] \leftarrow w[Length_{N_L} - 1]$        /* load assigned to least indexed adjacent vertex of $v_n$ */
        $v_i \leftarrow$ next neighbor in $N_L$
    $j \leftarrow 0$
    while $v_k$ in $N_L \setminus \{v_i\}$ do        /* assign loads to remaining vertices adjacent to $v_n$ */
        if $Deg_L[v_k]$ ! $= 1$ and $D_L[v_k] == 1$ then
            $w_{new}[v_k] \leftarrow w[j]$
            $v_k \leftarrow$ next neighbor in $N_L \setminus \{v_i\}$
            $j \leftarrow j + 1$
        else
            $v_k \leftarrow$ next neighbor in $N_L \setminus \{v_i\}$
$w_{new} = w_{new} + w_{pendant}$
$i \leftarrow 0$
foreach vertex $v_k$ in $V \setminus \{N_L\}$ do        /* assign the loads to the remaining non-adjacent vertices */
    $w_{new}[I_L[v_k]] \leftarrow w[i]$
    $i \leftarrow i + 1$
foreach $v_i$ in $V \setminus \{v_n\}$ do        /* store the values of $(\overline{w} - w_{new})_*$ */
    $w_{diff}[v_i] = \overline{w} - w_{new}[v_i]$
Compute $f_{new} = C * w_{diff}$        /* Compute the balancing flow */
Compute $l_2$-norm        /* Compute the $l_2$-norm of $f_{new}$ using the SageMath built-in function norm() */
return $f_{new}$
```

Theorem 6.3.3. *The balancing flow generated by the Balancing Flow Algorithm produces an optimal l_2-norm.*

Proof. The balancing flow is computed as $f_{new} = C * w_{diff}$, with C, w - loads in non-decreasing order and other parameters as inputs. In order to produce a balancing flow with an optimal l_2-norm, the algorithm must yield f_{new} as a column vector with very few non-zero entries. This is possible only if the linear combination of the entries of C and w_{diff} produces zero. In this algorithm, we choose the order in which loads are assigned to nodes so that when the row entries of C are multiplied by the entries of w_{diff}, the resulting column vector contains the majority of zero entries.

Consider the graph G with $V = \{v_1, v_2, ..., v_n\}$ and E edges. The matrix C is obtained by deleting v_n and finding a path from v_n to every other vertex in G. This results in a spanning tree $G^{'}$ whose root is vertex v_n. $G^{'}$ acts as a backbone of the network and the loads are balanced among the nodes through the edges of $G^{'}$. In $G^{'}$, arrange the vertices at distance $D = 1, D = 2$, etc., as shown in the Figure 6.1. Reassign the vertex labels so that the vertices at distance $D = 1$ have labels from $v_1, v_2, ...$ etc., then relabel the vertices at distance $D = 2$ and so on.

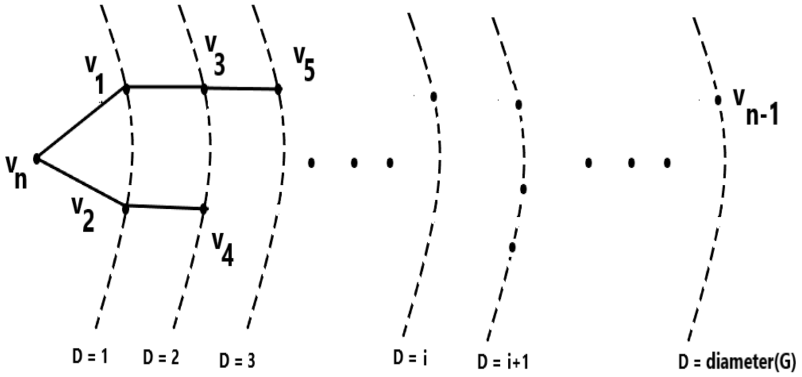


Figure 6.1 Vertices arranged at their corresponding distances from vertex v_n.

Initially, the largest load is assigned to vertex v_n. The next step is to assign the loads in non-decreasing order, starting with the smallest, to v_n's neighbors. The remaining loads are assigned to the vertices in decreasing order of distance from vertex v_n. Loads are assigned to vertices at distance $D = diameter(G)$, followed by the vertices at $D - 1, D - 2$, and so on, up to vertices at distance $D = 2$.

We now have the loads assigned to the nodes, and subtracting the average load from the assigned loads yields a set of at least $\frac{n}{2}$ positive and $\frac{n}{2}$ negative load values as column w_{diff}. The matrix C has $n - 1$ non-zero rows, with every such row consisting of $(-1, 0, 1)$ entries. The linear combination of the row entries in C and the entries in w_{diff}(set of positive and negative balance values) yields a column vector f_{new} with the majority of the entries as zero, resulting in fewer network edges used for the load balancing. As a result, the balancing flow produced has an optimal l_2-norm. □

Example: Consider the Petersen graph as shown in Figure 6.2 with the edge labels. The edge labels represent the edges. The initial loads are $w = [20, 22, 10, 33, 57, 49, 13, 30, 35, 31]^T$ and the average load is $\overline{w} = 30$, as in Stevanović (2014). The loads in w are sorted in non-decreasing order. The matrix C obtained by deleting vertex 9 is given in Figure 6.3.

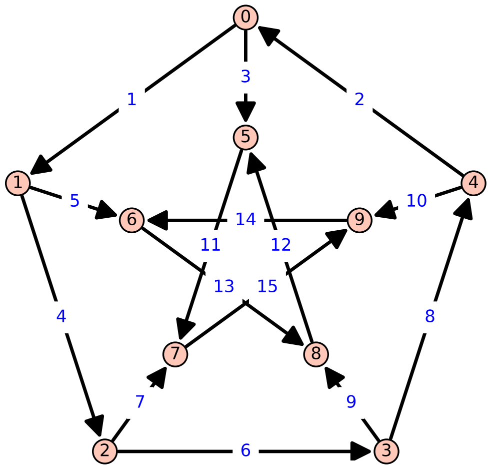


Figure 6.2 Petersen Graph G.

```
      [ 0  0  0  0  0  0  0  0  0]
      [ 1  0  0  0  0  0  0  0  0]
      [ 0  0  0  0  0  0  0  0  0]
      [ 0  0  0  0  0  0  0  0  0]
      [ 0 -1  0  0  0  0  0  0  0]
      [ 0  0  0  0  0  0  0  0  0]
C  =  [ 0  0 -1  0  0  0  0  0  0]
      [ 0  0  0 -1  0  0  0  0  0]
      [ 0  0  0  0  0  0  0  0  0]
      [-1  0  0 -1 -1  0  0  0  0]
      [ 0  0  0  0  0 -1  0  0  0]
      [ 0  0  0  0  0  0  0  0  0]
      [ 0  0  0  0  0  0  0  0  1]
      [ 0  1  0  0  0  0  1  0  1]
      [ 0  0 -1  0  0 -1  0 -1  0]
```

Figure 6.3 Matrix C obtained by deleting vertex 9.

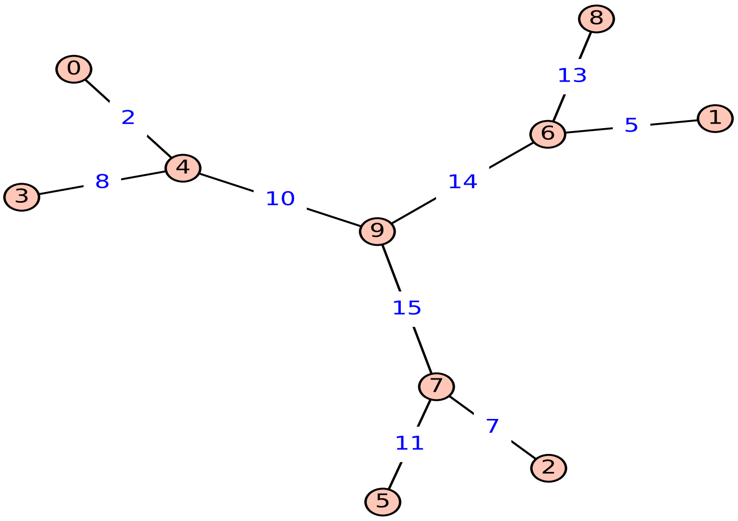


Figure 6.4 Spanning tree $G^{'}$ from C.

The Parameter Computing Algorithm computes the following: The spanning tree $G^{'}$ obtained from C is shown in Figure 6.4. The list of distances D_L from vertex $v_n = 9$ is computed as $[2,2,2,2,1,2,1,1,2]$. The index of the vertices in the increasing order of its distance from v_n is $I_L = [4,6,7,0,1,2,3,5,8]$. The Neighbors of vertex 9 are $[4,6,7]$ and loads to be assigned are $w_{N_L} = [10,13,20]$. The Balancing Flow Algorithm computes the balancing flow. The final assigned loads are $w_{new} = [22,30,31,33,20,35,10,13,49,$ $57]$. The difference between the average and initial loads is $w_{diff} = [8,0,-1,-3,10,-5,$

$20, 17, -19]$. The balancing flow f_{new} is $(0,8,0,0,0,0,1,3,0,-15,5,0,-19,1,-11)$. The l_2-norm of $||f_{new}||_2$ = 29 whereas the l_2-norm of the balancing flow in Stevanović (2014) is $||f_2||_2 = 41.95 \approx 42$. Since the loads are assigned as in w_{new}, the balancing flow results in the l_2-norm that is lesser compared to l_2-norm in Stevanović (2014).

Table 6.1 gives the comparison of l_2-norm of the balancing flow $||f_2||_2$ generated by *Algorithm 2* and $||f_{new}||_2$ generated by the proposed algorithm. For the Generalized Petersen graph, the loads assigned are $w = [10,40,20,30,60,50,80,90,70,110,100,120]$. The loads assigned to the nodes of Clebsch graph are $w = [10,20,40,30,60,50,70,80,$ $100,90,150,120,130,110,140,160]$. Once the loads arrive, the proposed algorithm assigns them to the nodes of the spanning tree $G^{'}$ such that the loads are balanced using fewer network edges. Hence, the proposed algorithm generates a balancing flow with a lesser l_2-norm irrespective of the loads' order.

Table 6.1 Graphs with l_2-norm of $||f_2||_2$ and $||f_{new}||_2$.

Graph	$\|V\| = n$	$\|E\| = m$	$\|\|f_2\|\|_2$	$\|\|f_{new}\|\|_2$
Petersen Graph	10	15	42	29
Generalized Petersen Graph GP(6,1)	12	18	107	68
Clebsch Graph	16	40	245	149

For the graphs in Table 6.1, it can be observed that the l_2-norm of the balancing flow $||f_{new}||_2$ generated by the proposed algorithm is much less than that of the l_2-norm generated by *Algorithm 2* in Stevanović (2014).

6.4 TIME COMPLEXITY

Theorem 6.4.1 and 6.4.2 give an evaluation of the worst-case time complexity of the proposed algorithms for balancing the loads.

Theorem 6.4.1. *Parameter Computation Algorithm takes $O(n^3)$ running time.*

Proof. In Parameter Computation Algorithm, the *apply_map*() function takes $O(n^3)$ running time since the entries of C are to be accessed and mapped. The for loop in step 3 repeats at most n times, and for every iteration a column of matrix N is deleted in $O(n)$ running time. So, the for loop takes an $O(n^2)$ running time. The spanning tree is constructed from the vertex-edge incidence matrix N' in $O(nm) \approxeq O(n^3)$ running time. The for loop in step 9, in the worst case, takes an $O(n)$ running time. In step 13, the while loop repeats at most $D = n-1$ times, which takes an $O(n)$ running time, and the for loop inside iterates at most n times. So, the while loop that generates list I_L has a running time of $O(n^2)$. Construction of the list N_L takes an $O(n^2)$ running time. As a result, the algorithm has an $O(n^3)$ running time. □

Theorem 6.4.2. *The Balancing Flow Algorithm runs in $O(n^3)$ time.*

Proof. In the Balancing Flow Algorithm, the for loop in step 8 repeats at most n times which takes an $O(n)$ running time and for every iteration the list N_L and w_{N_L} is updated and it takes an $O(n)$ running time. So, the for loop that assigns the loads to the adjacent (pendant) neighbors of v_n has an $O(n^2)$ running time. If v_n has no pendant neighbors, the load distribution to the adjacent (non-pendant) neighbors takes an $O(n)$ running time. In addition, the assignment of the remaining loads to the non-adjacent vertices of a takes an $O(n)$ running time. The computation of balancing flow f_{new} requires the multiplication of the $m \times (n-1)$ matrix C with an $(n-1) \times 1$ column vector w_{diff}, which takes an $O(n^3)$ running time. So, the time complexity of this algorithm is $O(n^3)$. □

Theorem 6.4.3. *For every graph G with integer loads to be assigned to its vertices, the load balancing is done in $O(n^3)$ time.*

Proof. To compute the balancing flow that balances the loads at the nodes, we have presented two algorithms, namely: the Parameter Computation Algorithm and the Balancing Flow Algorithm. Parameter Computation Algorithm takes an $O(n^3)$ running time (Theorem 6.4.1) to compute several parameters. The Balancing Flow Algorithm takes an $O(n^3)$ running time (Theorem 6.4.2) for assigning the loads and computing the balancing flow with the help of the parameters determined by the previous algorithm. As a result, load balancing can be done in $O(n^3)$ time for any graph G with integer loads assigned to its vertices. □

CHAPTER 7

CONCLUSIONS AND FUTURE SCOPE

In this thesis, the focus is on studying some Graph Labeling approaches and Graph Spectra, along with their applications in relevant areas. In this chapter, conclusions and future scope are presented.

Chapter 2 showed that the construction of exclusive sum labeling for odd cycles given by Miller et al. (2005) failed to produce an exclusive sum labeling of odd cycles C_5 and C_7. Further, we presented a modified exclusive sum labeling for cycles C_n of odd length $n > 5$. Also, we give exclusive sum labeling of the Cartesian product of cycle C_n and K_2, i.e., $C_n \Box K_2$, Cartesian product of complete graph K_n and K_2, i.e., $K_n \Box K_2$, the disjoint union of paths, the disjoint union of cycles. One can obtain the exclusive sum numbers for graphs resulting from various graph operations in future work.

In Chapter 3, we presented the exclusive sum number of complete k-partite graph $K_{r_1,r_2,\ldots,r_k}$, which can model links in the relational database. Since the exclusive sum labeling is a method of graph representation in terms of integers, and one can further, look for its applications in situations where brevity is desirable.

Chapter 4 deals with proper monographs labelings. We have presented the proper monograph labelings of classes of graphs such as cycles, $C_n \odot K_1$, cycles with paths attached to one or more vertices, and cycles with an irreducible tree attached to one or more vertices. From the signatures of these proper monographs, we show that one can determine their maximum independent sets. Further, one can investigate the proper monograph labelings for other classes of graphs and also determine their relationship with the properties of the graph.

In Chapter 5, we present several graphs that qualify as models for efficient MINs based on the small values of the graph tightness introduced by Cvetković and Davidović (2008). These graphs are constructed using some extensively used graph operations. The tightness values of these graphs range from $O(\sqrt[4]{N})$ to $O(\sqrt{N})$, where N is the order of the graph under consideration. Also, two new graph tightness values, namely *Third* *type mixed tightness* $t_3(G)$ and *Second type of Structural tightness* $t_4(G)$, are defined in this chapter. It has been shown that these tightness types are easier to calculate than the others for the considered graphs. Moreover, their values are significantly smaller. In the future, we can extend examples by comparing the tightness values of various graphs obtained from different graph operations.

Chapter 6 presents an algorithm as a modified approach to the existing algorithm in Stevanović (2014). The proposed algorithm results in a balancing flow with a lesser l_2-norm than the l_2-norm of the balancing flow generated by the existing algorithm. Further, we show that the load balancing by the proposed algorithm is done in $O(n^3)$ time. One could investigate and examine some other properties of matrix C as a future direction.

BIBLIOGRAPHY

Arlinghaus, S., Arlinghaus, W., and Frank, H. (1993). Sum graphs and geographic information. *Solstice : An Electronic Journal of Geography and Mathematics*.

Barmon, C., Faruqui, M., and Battacharjee, G. (1991). Dynamic load balancing algorithm in a distributed system. *Microprocessing and Microprogramming*, 29(5):273–285.

Behzad, M. (1970). A characterization of total graphs. *Proceedings of the American Mathematical Society.*, 26:383–389.

Bergstrand, D., Hodges, K., Jennings, G., Kuklinski, L., Wiener, J., and Harary, F. (1989). The sum number of a complete graph. *Bull. Malaysian Math. Soc. (2)*, 12(1):25–28.

Bloom, G. and Golomb, S. (1977). Applications of numbered undirected graphs. *Proceedings of the IEEE*, 65(4):562–570.

Bloom, G. S., Hell, P., and Taylor, H. (1979). Collecting autographs: n-node graphs that have n-integer signatures. *Annals of the New York Academy of Sciences*, 319(1):93–102.

Boillat, J. E. (1990). Load balancing and poisson equation in a graph. *Concurrency: practice and experience*, 2(4):289–313.

Boland, J., Laskar, R., Turner, C., and Domke, G. (1990). On mod sum graphs. In *Proceedings of the Twentieth Southeastern Conference on Combinatorics, Graph Theory, and Computing (Boca Raton, FL, 1989)*, volume 70, pages 131–135.

Brooks, R. L. (1941). On colouring the nodes of a network. *Mathematical Proceedings of the Cambridge Philosophical Society*, 37(2):194–197.

Brouwer, A., Cohen, A., and Neumaier, A. (1989). *Distance-Regular Graphs.* Springer-Verlag, Berlin.

Brouwer, A. E. and Haemers, W. H. (2011). *Spectra of graphs*. Springer.

Chartrand, G. and Zhang, P. (2009). *Chromatic graph theory*. Discrete Mathematics and its Applications. CRC Press, Boca Raton, FL.

Cvetković, D. and Davidović, T. (2008). Application of some graph invariants to the analysis of multiprocessor interconnection networks. *Yugoslav Journal of Operations Research.*, 18(2):173–186.

Cvetković, D., Davidović, T., Ilić, A., and Simić, S. K. (2010). Graphs for small multiprocessor interconnection networks. *Applied mathematics and computation*, 217(6):2468–2480.

Cvetković, D., Davidović, T., and Jovanović, I. M. (2016). Some new models for multiprocessor interconnection networks. *Yugoslav Journal of Operations Research.*, 26(4):423–439.

Cvetković, D. M., Doob, M., and Sachs, H. (1995). *Spectra of graphs : Theory and applications*. Johann Ambrosius Barth, Heidelberg - Leipzig,, 3rd edition.

Cvetković, D. and Simić, S. (2011). Graph spectra in computer science. *Linear Algebra and its Applications*, 434(6):1545–1562.

Cvetković, D. M. (1971). Graphs and their spectra. *Publikacije Elektrotehničkog fakulteta. Serija Matematika i fizika*, (354/356):1–50.

Cybenko, G. (1989). Dynamic load balancing for distributed memory multiprocessors. *Journal of Parallel and Distributed Computing*, 7(2):279–301.

Diekmann, R., Frommer, A., and Monien, B. (1999). Efficient schemes for nearest neighbor load balancing. *Parallel Computing*, 25(7):789–812.

Eager, D. L., Lazowska, E. D., and Zahorjan, J. (1986). Adaptive load sharing in homogeneous distributed systems. *IEEE Transactions on Software Engineering*, SE-12(5):662–675.

Elsässer, R., Kralovic, R., and Monien, B. (2003). Sparse topologies with small spectrum size. *Theoretical Computer Science.*, 307:549–565.

Euler, L. (1741). Solutio problematis ad geometriam situs pertinentis. *Commentarii academiae scientiarum Petropolitanae*, pages 128–140.

Fontanil, L. and Panopio, R. (2014). Independent set and vertex covering in a proper monograph determined through a signature. *Australas. J Comb.*, 59:64–71.

Gallian, J. A. (2021). A dynamic survey of graph labeling. *Electronic Journal of combinatorics*, 1(DynamicSurveys):DS6.

Gervacio, S. and Panopio, R. (1982). Are generalized petersen graphs proper monographs. *Matimyas Matemetika*, 6:1–6.

Hammack, R., Imrich, W., and Klavžar, S. (2011). *Handbook of product graphs*. CRC press.

Harary, F. (1969). *Graph theory*. Addison-Wesley Publishing Co., Reading, Mass.-Menlo Park, Calif.-London.

Harary, F. (1990). Sum graphs and difference graphs. *Congressus Numerantium*, 72:101–108.

Harary, F. (1994). Sum graphs over all the integers. *Discrete Mathematics*, 124(1):99–105.

Harary, F., Hentzel, I., and Jacobs, D. (1991). Digitizing sum graphs over the reals. *Caribb. J. Math. Comput. Sci.*, 1:1–4.

Harary, F. and Norman, R. Z. (1960). Some properties of line digraphs. *Rendiconti del Circolo Matematico di Palermo*, 9(2):161–168.

Harary, F. and Schwenk, A. J. (1974). Which graphs have integral spectra? In *Lecture Notes in Mathematics*, pages 45–51. Springer Berlin Heidelberg.

Hegde, S. M. and Vasudeva (2009). On mod difference labeling of digraphs. *AKCE International Journal of Graphs and Combinatorics*, 6(1):79–84.

Hu, Y. F., Blake, R. J., and Emerson, D. R. (1998). An optimal migration algorithm for dynamic load balancing. *Concurr. Pract. Exp.*, 10:467–483.

Jaradat, M. M. M. (2005). On the Anderson-Lipman conjecture and some related problems. *Discrete Mathematics.*, 297(1-3):167–173.

Krausz, J. (1943). Démonstration nouvelle d'une théoreme de whitney sur les réseaux. *Mat. Fiz. Lapok*, 50(1):75–85.

Krueger, P. and Finkel, R. A. (1984). An adaptive load balancing algorithm for a multicomputer. Technical report, University of Wisconsin-Madison Department of Computer Sciences.

Mahamoodian, E. S. (1981). On edge-colorability of Cartesian products of graphs. *Canadian Mathematical Bulletin.*, 24(1):107–108.

Miller, M., Patel, D., Ryan, J., Sugeng, K. A., Slamin, and Tuga, M. (2005). Exclusive sum labeling of graphs. *J. Combin. Math. Combin. Comput.*, 55:137–148.

Rosa, A. (1966). On certain valuations of the vertices of a graph. In *Theory of Graphs (Internat. Symposium, Rome*, pages 349–355.

Slamet, S., Sugeng, K., and Miller, M. (2006). Sum graph based access structure in a secret sharing scheme. *Journal of Prime Research in Mathematics*, 2:113–119.

Song, J. (1994). A partially asynchronous and iterative algorithm for distributed load balancing. *Parallel Computing*, 20(6):853–868.

Sonntag, M. (2003). Difference labelling of cacti. *Discussiones Mathematicae Graph Theory*, 23(1):55–65.

Sonntag, M. (2004). Difference labelling of digraphs. *Discussiones Mathematicae Graph Theory*, 24(3):509–527.

Stein, W. (2007). Sage mathematics software. *http://www. sagemath. org/.*

Stevanović, D. (2014). Remarks on dynamic load balancing of integer loads and integral graphs. *Applied Mathematics and Computation*, 226:38–43.

Sugeng, K. A. and Ryan, J. (2007). On several classes of monographs. *Australasian Journal of Combinatorics*, 37:277.

Sutton, M. (2000). *Summable Graph Labelings and Their Applications*. PhD dissertation, The University of Newcastle.

Tuga, M. and Miller, M. (2005). Δ-optimum exclusive sum labeling of certain graphs with radius one. In *Combinatorial geometry and graph theory*, volume 3330 of *Lecture Notes in Comput. Sci.*, pages 216–225. Springer, Berlin.

Tuga, M., Miller, M., Ryan, J., and Ryjáček, Z. (2005). Exclusive sum labelings of trees. *J. Combin. Math. Combin. Comput.*, 55:109–121.

Vizing, V. G. (1965). Critical graphs with given chromatic class (in russian). *Metody Discret. Analiz.*, 5:9–17.

West, D. B. (1996). *Introduction to graph theory*. Prentice hall Upper Saddle River, NJ.

Whitney, H. (1932). Congruent graphs and the connectivity of graphs. *American Journal of Mathematics*, 54(1):150–168.

Wilf, H. S. (1967). The eigenvalues of a graph and its chromatic number. *Journal of the London Mathematical Society*, s1-42(1):330–332.

Xu, C., Lau, F. C., Lau, F. C., Monien, B., and Lüling, R. (1995). Nearest-neighbor algorithms for load-balancing in parallel computers. *Concurrency: Practice and Experience*, 7(7):707–736.

Xu, C.-Z. and Lau, F. C. M. (1992). Analysis of the generalized dimension exchange method for dynamic load balancing. *Journal of Parallel and Distributed Computing*, 16(4):385–393.

PUBLICATIONS

Journal publications:

1. S. M. Hegde and Y. M, Saumya. Construction and Analysis of Graph Models for Multiprocessor Interconnection Networks. *Yugoslav Journal Of Operations Research*, 32(1), 87–109, 2022. ISSN 2334-6043. *doi*:10.2298/*YJOR*200915017*H*.

2. S. M. Hegde, and Y. M. Saumya. Maximum Independent Sets in a Proper Monograph Determined through a Signature, *Discrete Mathematics, Algorithms and Applications*, *doi*: 10.1142/*S*1793830922500926.

3. S. M. Hegde and Y. M, Saumya. Further results on Exclusive Sum Labeling of graphs and its application in Relational Database Systems, Communicated to *International Journal of Applied and Computational Mathematics* (Under review)

4. S. M. Hegde, and Y. M. Saumya. A Modified Approach for Dynamic Load Balancing of Integer Loads, Communicated to *Journal of Parallel and Distributed Computing* (Under Review)

Paper presented at Conferences:

1. Suresh M Hegde, and Saumya Y. M. Further results on Exclusive Sum Labeling of graphs and its application in Relational Database Systems, *International Conference on Graph Labeling and Applications ICGLA-2020.* at Department of Mathematics, Goa University, Goa-India, September 17-19, 2020.

BIODATA

Name	:	Saumya Y. M.
Email	:	saumya2087@gmail.com
Date of Birth	:	20th May 1987.
Permanent address	:	Saumya Y. M., D/o P. M. Patali, D.No 1-70/5, 'Manasa', Mugrodi Road, Padavinangady Mangaluru-575008.

Educational Qualifications :

Degree	**Year**	**Institution / University**
B.E.(Information Science & Engineering)	2009	P. A. College of Engineering, Mangaluru. VTU,Belgaum
M.Tech.(Systems Analysis & Computer Applications)	2012	NITK, Surathkal