Broadcasting in graphs containing intersecting cliques

Akash Ambashankar

A Thesis

in

The Department

of

Computer Science and Software Engineering

Presented in Partial Fulfillment of the Requirements

for the Degree of

Master of Computer Science (Computer Science) at

Concordia University

Montréal, Québec, Canada

August 2024

CONCORDIA UNIVERSITY

School of Graduate Studies

This is to certify that the thesis prepared

By: **Akash Ambashankar**

Entitled: **Broadcasting in graphs containing intersecting cliques**

and submitted in partial fulfillment of the requirements for the degree of

Master of Computer Science (Computer Science)

complies with the regulations of this University and meets the accepted standards with respect to originality and quality.

Signed by the Final Examining Committee:

______________________________ Chair
Dr. Lata Narayanan

______________________________ Examiner
Dr. Lata Narayanan

______________________________ Examiner
Dr. Dhrubajyoti Goswami

______________________________ Supervisor
Dr. Hovhannes A. Harutyunyan

Approved by ______________________________
Joey Paquet, Chair
Department of Computer Science and Software Engineering

____________ 2024 ______________________________
Mourad Debbabi, Dean
Faculty of Engineering and Computer Science

Abstract

Broadcasting in graphs containing intersecting cliques

Akash Ambashankar

This thesis is a study on the broadcasting problem in various graph topologies containing intersecting cliques. Broadcasting, is a fundamental message dissemination problem in interconnection networks, requiring an informed originator node to distribute information to all network nodes efficiently. Determination of the broadcast time of any node in an arbitrary network is known to be NP-hard. Polynomial time solutions are known only for a few network topologies. There also exist various heuristic and approximation algorithms for different network topologies. The work in this thesis addresses this challenge through the development and enhancement of algorithms tailored to specific graph structures such as windmill graphs, star of cliques, path-connected cliques, fully connected cliques, and block graphs.

Chapter 3 studies the broadcasting problem in windmill graphs and stars of cliques. A windmill graph is defined on n vertices containing k cliques of size l, and a universal vertex. We present a constant time method introduced for determining the broadcast time of an arbitrary node in any windmill graph. We also introduce *Star of Cliques*, a generalization of the Windmill graph composed of cliques with arbitrary sizes. We study the importance of unaddressed vertices in an optimal scheme for star of cliques and show the use of binary representations to track unaddressed vertices. We present an $O(n \cdot \log n)$ algorithm to find the broadcast time of any vertex in an arbitrary star of cliques and discuss the optimality of our algorithm.

Chapter 4 delves into broadcasting within families of block graphs, such as path-connected cliques and fully connected cliques. Every biconnected component in a block graph is a clique. We study broadcasting in block graphs with two cliques, present a constant time method to calculate the broadcast time for an arbitrary path-connected cliques graph, and discuss a $O(n \log \log s_{center})$

method to determine the broadcast time for arbitrary fully connected cliques, where s_{center} is the number of vertices in the central clique of the graph.

Acknowledgments

Firstly, I extend my heartfelt gratitude to my supervisor, Professor Hovhannes Harutyunyan. His unwavering support, invaluable guidance, and encouragement have profoundly enriched my academic journey and made this thesis an inspiring and fulfilling experience for me.

I owe my deepest thanks to all my teachers and professors, mentors, and motivators in life, and to every experience and insight that has guided me to where I am today.

Additionally, I want to express my appreciation to my friends and colleagues for their support, camaraderie, and insightful discussions throughout this thesis journey. Their friendship and collaboration have contributed greatly to my personal and academic growth.

I am most grateful for my family for their encouragement, understanding, and patience throughout this endeavor. Their love and encouragement have been a constant source of strength and motivation.

Above all, I would like to express my deepest gratitude to God for granting me the strength, wisdom, and perseverance throughout this journey, and for blessing me with such wonderful people in my life.

I'd like to conclude with a thought that has inspired me throughout this journey: "You can do anything you set your mind to." This message has been a guiding principle as I worked on this thesis.

Contents

List of Figures

Chapter 1

Introduction

Communication is a fundamental aspect of the world that plays a pivotal role in the evolution, survival, and progress of humanity. It serves as the cornerstone of societal development, enabling the exchange of ideas, fostering cooperation, and allowing us to advance collectively. Through communication, humans have been able to pass down wisdom and transmit information across generations, contributing to accumulating knowledge and improving society. The advent of technology, particularly the internet, has revolutionized communication at an unprecedented scale, reshaping the way individuals and societies interact globally. The internet has transcended geographical barriers, allowing instantaneous communication across vast distances through emails, social media platforms, video conferencing, and instant messaging services. This is possible due to multiprocessing and distributed systems, which disperse data and computational tasks across various servers, thereby improving the speed and fault tolerance of the networks.

However, to facilitate future endeavors of humanity, including Artificial General Intelligence and Space Exploration, substantial progress in distributed systems and communication protocols is essential. A lot of attention is being directed towards dedicated AI acceleration hardware, Photonic Computing, Edge Computing, and Algorithmic Improvements.

The focus of our work in this thesis is on making algorithmic improvements in the problem of broadcasting, which is a message dissemination problem in a connected network. One informed node, called the originator, must distribute a message to all other nodes by placing a series of calls along the communication lines of the network. This problem finds applications in multiprocessor systems, parallel and distributed systems, and epidemic algorithms among other fields.

The rest of the thesis is structured as follows: In Chapter 2, we elaborate on the broadcasting problem and present a review of some important results in the field. In Chapter 3 we study the broadcasting problem in the Windmill graph, containing n vertices across k cliques with l vertices each, and a universal vertex. We also introduce *Star of Cliques*, a generalization of the Windmill graph composed of cliques with arbitrary sizes. We present a $O(n \cdot \log n)$ algorithm to find the broadcast time of any vertex in an arbitrary star of cliques and discuss the optimality of our algorithm. In Chapter 4, we discuss the broadcast problem in various families of block graphs including path-connected cliques and fully connected cliques. In Chapter 5, we present our conclusions and directions for future work.

Chapter 2

Literature Review and Preliminaries

In recent years, much work has been dedicated to studying the properties of interconnection networks to find the best communication structures for parallel and distributed computing. Information dissemination in interconnection networks can be represented by an undirected graph $G = (V, E)$, with the set of vertices V representing the nodes or processors in the network and the set of edges E representing the communication lines of the network. We use the terms network and graph interchangeably. Some communication protocols in interconnection networks include:

(1) **Unicast**: one-to-one – A vertex u sends information to another vertex

(2) **Multicast**: one-to-many – A vertex u sends information to many other vertices

(3) **Broadcast**: one-to-all – A vertex u sends information to all other vertices

(4) **Gossiping**: all-to-all – Every vertex sends unique information to every other vertex.

In this thesis, we explore the broadcasting protocol, with particular emphasis on the classical broadcasting model. We will delve into its specifics in the upcoming sections. Additionally, we will touch upon alternative broadcasting models in Section 2.3.

2.1 Classical Broadcasting

Broadcasting is one of the main areas of research in information dissemination. It is a message dissemination problem in a connected network where one informed vertex, called the originator, must distribute a message to all other vertices by placing a series of calls along the edges of the graph. An informed vertex sends the message to one of its uninformed neighbors by making a call. At each time unit, the newly informed nodes assist the originator by informing their neighboring vertices. The process ends when all vertices are informed. Broadcasting is to be completed using the least possible time units and is subject to the following constraints:

(1) Time units are discrete.

(2) Each call requires one time unit and involves two neighboring vertices.

(3) Each vertex can participate in only one call per time unit.

(4) Multiple calls can occur in parallel between distinct pairs of neighboring vertices.

According to Hedetniemi, Hedetniemi, and Liestman (1988), the broadcasting problem was introduced by Slater, Cockayne, and Hedetniemi in 1977 as a variant of the gossiping problem. They studied the problem of determining the minimum time required for one person to transmit one piece of information to everyone else in a communication graph. By comparison, gossiping is an all-to-all information dissemination process.

Given a connected graph G and an originator vertex u, the number of time units required to complete broadcasting in G from u is denoted by $b(u, G)$ or simply $b(u)$. This may also be referred to as the *broadcast time of vertex u*. During each time unit, the number of informed vertices can double at most, if each informed vertex calls a different uninformed vertex. Thus we obtain the following lower bound: $b(u) \geq \lceil \log n \rceil$ where $n = |V|$ is the number of nodes in the network (all logarithms presented in this thesis are in base 2). By definition of the broadcasting problem, there must be at least one new informed vertex in each time unit. This leads to the following upper bound: $b(u) \leq n - 1$. The broadcast time of a graph is the maximum among the broadcast time of each vertex and is denoted by $b(G) = \max\limits_{\forall u \in V}\{b(u, G)\}$. On the other hand, the set of vertices with minimum broadcast time is called the *broadcast center* of the graph.

The set of calls used to distribute the message from originator u to all other vertices is a *broadcast scheme* for vertex u. The broadcast scheme forms a spanning tree rooted at the originator, known as a *broadcast tree*. A graph G with $b(G) = \lceil \log n \rceil$ is called a *broadcast graph*.

For example, the complete graph K_n with n vertices and $\binom{n}{2}$ edges is a broadcast graph since $b(K_n) = \lceil \log n \rceil$ for all $n \geq 2$.

The literature on the broadcasting problem can be divided primarily into two major areas: the minimum broadcast graph problem and the minimum broadcast time problem. These problems are discussed in the following sections.

2.1.1 Minimum broadcast graph problem

$B(n)$ denotes the minimum number of edges in any broadcast graph on n vertices and is called the *broadcast function*. A graph with $b(G) = \lceil \log n \rceil$ and $B(n)$ edges is called a *minimum broadcast graph (mbg)* on n vertices. The problem was first introduced by Farley, Hedetniemi, Mitchell, and Proskurowski (1979) along with mbgs for $n \leq 15$ and the hypercube as an infinite family of mbgs on $n = 2^k$. Later, the recursive circulant graph was proven to be a non-isomorphic alternative for mbg on $n = 2^k$ (Park & Chwa, 1994), and the Knödel Graph, introduced in Knödel (1975), was independently proven to be an mbg for $n = 2^k$ and $n = 2^k - 2$ by Khachatrian and Harutounian (1990) and Dinneen, Fellows, and Faber (1991).

The following is a list of all values of n for which $B(n)$ is known:

- $1 \leq n \leq 15$ and $n = 2^k$ (Farley et al., 1979)
- $n \leq 17$ (Mitchell & Hedetniemi, 1980)
- $n = 18, 19$ (Bermond, Hell, Liestman, & Peters, 1992; Xiao & Wang, 1988)
- $n = 20, 21, 22$ (Maheo & Saclé, 1994)
- $n = 26$ (Saclé, 1996; Zhou & Zhang, 2001)
- $n = 27, 28, 29$ (Saclé, 1996)
- $n = 30, 31$ (Bermond et al., 1992)
- $n = 58, 59, 60, 61$ (Saclé, 1996)
- $n = 62$ (Farley, 1979)

- $n = 63$ (Labahn, 1994)
- $n = 127$ (H. A. Harutyunyan, 2008)
- $n = 1023, 4095$ (Shao, 2006)

However, an infinite family of mbgs for the general case is not known yet and remains an open problem. Another direction of research has been to connect some of these smaller broadcast graphs to construct larger broadcast graphs, which inherently have an even number of vertices. (Bermond, Fraigniaud, & Peters, 1995; Chau & Liestman, 1985; Chen, 1990; Dinneen et al., 1991; Dinneen, Ventura, Wilson, & Zakeri, 1999; Gargano & Vaccaro, 1989; H. A. Harutyunyan & Liestman, 1999; Khachatrian & Harutounian, 1990). An upper bound on $B(n)$ for an odd number of vertices was presented by H. A. Harutyunyan and Liestman (2012), following which improved upper bounds on $B(n)$ for odd and even vertices were presented by Averbuch, Shabtai, and Roditty (2014) and H. A. Harutyunyan and Li (2017)

2.1.2 Minimum broadcast time problem

Finding $b(u, G)$ and $b(G)$ for arbitrary graphs with arbitrary originators has been proven to be NP-complete (Slater, Cockayne, & Hedetniemi, 1981). The problem also remains NP-complete in more restricted families such as bounded degree graphs (Dinneen, 1994) and 3-regular planar graphs (Jakoby, Reischuk, & Schindelhauer, 1998; Middendorf, 1993). Hence, a lot of work has been done to identify (1) Exact algorithms (2) Approximation algorithms, and (3) Heuristics.

Also, we mention a few survey papers here that deal with this problem, from which the reader can trace back all the previous works (Fraigniaud & Lazard, 1994; H. A. Harutyunyan, Liestman, Peters, & Richards, 2013; Hedetniemi et al., 1988; Hromkovič, Klasing, Monien, & Peine, 1996). As a recent study of the complexity of the problem, we also refer the reader to Fomin, Fraigniaud, and Golovach (2023) and Tale (2024).

Exact Algorithms

Some exact algorithms have been proposed to determine $b(u, G)$ and $b(G)$ for an arbitrary originator in an arbitrary graph. These include the Dynamic Programming algorithm suggested in Scheuermann (1984) and the ILP model proposed in de Sousa et al. (2018), with the latter being considered the best exact method known. These approaches can effectively process networks containing no more than 50 nodes within a reasonable time frame, which may render them impractical for some real-world applications.

Another direction of research has been to identify exact algorithms for specific families of graphs. This was initiated by Slater et al. (1981) with the proposal of a linear algorithm for Trees, followed by algorithms for Grids and Tori (Farley & Hedetniemi, 1978), Cube Connected Cycles (Liestman & Peters, 1988) and Shuffle Exchange (Hromkovič, Jeschke, & Monien, 1990).

Eventually, exact algorithms were developed for more non-trivial topologies such as Unicyclic graphs (H. Harutyunyan & Maraachlian, 2007; H. A. Harutyunyan & Maraachlian, 2008), Tree of Cycles (H. A. Harutyunyan & Maraachlian, 2009), Fully Connected Trees (Gholami, Harutyunyan, & Maraachlian, 2023), and Necklace Graphs (H. Harutyunyan, Laza, & Maraachlian, 2009).

Approximation Algorithms

Research by Schindelhauer (2000) has shown that it is NP-Hard to approximate the solution of the broadcast time problem within a factor $\frac{57}{56} - \epsilon$. However, this result has been improved to within a factor of $3-\epsilon$ in Elkin and Kortsarz (2002), who also presented an approximation algorithm which produces a broadcast scheme with $O\left(\frac{\log(|V|)}{\log\log(|V|)}b(G)\right)$ rounds. This is the best approximation known for the problem.

Approximation algorithms have also been designed for specific families of graphs such as k-path graphs (Bhabak & Harutyunyan, 2019), k-cycle graphs (Bhabak & Harutyunyan, 2015), flower graphs (Ehresmann, 2021), graphs with known broadcast time of the base graph (Bhabak & Harutyunyan, 2022), and Harary graphs and graphs similar to Harary graphs (Bhabak, Harutyunyan, & Kropf, 2017; Bhabak, Harutyunyan, & Tanna, 2014), to name a few.

Heuristics

Many heuristics have been proposed for the broadcasting problem, including the Matching-based approach, Coloring-based approach, The Round-Heuristic (Beier & Sibeyn, 2000), the Random and semi-random heuristics (H. A. Harutyunyan & Wang, 2010), Deep Heuristic for arbitrary graphs (H. A. Harutyunyan, Hovhannisyan, & Magithiya, 2022), Heuristics for directed graphs (Elkin & Kortsarz, 2002) the Tree-based heuristic, introducing the concept of "bright border" (H. A. Harutyunyan & Shao, 2006).

We refer the reader to Fraigniaud and Vial (1997, 1999); H. A. Harutyunyan and Jimborean (2014); Ravi (1994); Scheuermann (1984) for more results in this category.

2.2 Broadcasting in some known topologies

In this section, we review broadcasting in some network topologies under the classical broadcasting model. We study properties of graphs related to communication in interconnection networks such as the diameter, the number of edges, and the maximum degree.

2.2.1 Path Graph P_n

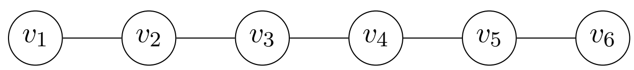

Figure 2.1: Path Graph with $n = 6$

The path graph P_n is a sequence of n vertices such that every vertex is connected to the next vertex in the sequence. With $V = \{1, \dots, n\}$, we have $E = \{(u, u + 1) | 1 \leq u \leq n - 1\}$. P_n has $n - 1$ edges and a diameter of $n - 1$. The first and the last vertices of the sequence have a degree of 1, while all other vertices have a degree of 2. The vertex with the minimum broadcast time in a path graph, which is the midpoint (either of the midpoints if n is even), has a broadcast time of $\lceil \frac{n}{2} \rceil$. The vertices with the maximum broadcast time are the two endpoints, making $b(P_n) = n - 1$. Figure 2.1 illustrates P_6.

2.2.2 Ring Graph R_n (Cycle)

The ring R_n is a path such that the end vertex is connected to the start vertex. With $V = \{1, \dots, n\}$, we have $E = \{(u, u + 1) | 1 \leq u \leq n - 1\} \cup \{(n, 1)\}$. R_n has n edges and a diameter of $\lfloor \frac{n}{2} \rfloor$. All vertices have a degree of 2, and exhibit an identical broadcast time. Therefore, $b(R_n) = \lceil \frac{n}{2} \rceil$. Figure 2.2 illustrates R_5.

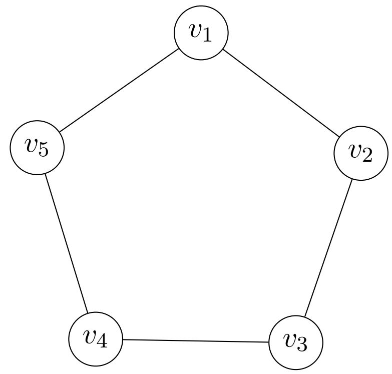

Figure 2.2: Ring Graph with $n = 5$

2.2.3 Star Graph S_n

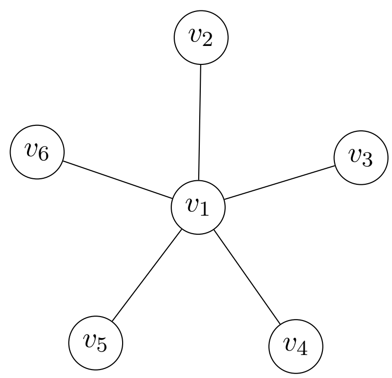

Figure 2.3: Star Graph with $n = 6$

The star is a graph on n vertices consisting of a central vertex which is connected to several other nodes, called leaves. With $V = \{1, \ldots, n\}$, we have the edges $E = \{(1, i) | 2 \leq i \leq n\}$. S_n has $n - 1$ edges and a diameter of 2. All vertices have a degree of 1, but the central vertex has a degree of $n - 1$. The vertices exhibit an identical broadcast time, making $b(S_n) = n - 1$. Figure 2.3 illustrates S_6.

2.2.4 Complete Graph K_n

The complete graph consists of n vertices, with every vertex being connected by an edge to every other vertex. For $V = \{1, \ldots, n\}$ we have $\binom{n}{2}$ edges, which are all the possible edges between the vertices. As a result, the degree of every vertex is $n - 1$ and the diameter of the graph is 1. All vertices exhibit an identical broadcast time of $\lceil \log n \rceil$. Figure 2.4 illustrates K_5.

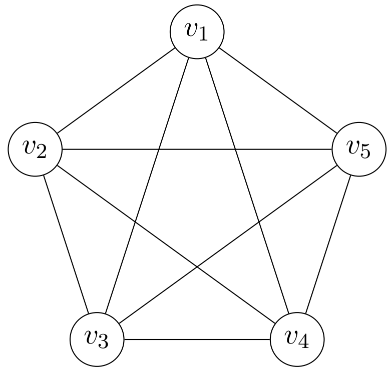

Figure 2.4: Complete Graph with $n = 5$

2.2.5 Fork Graph $F_{n,k}$

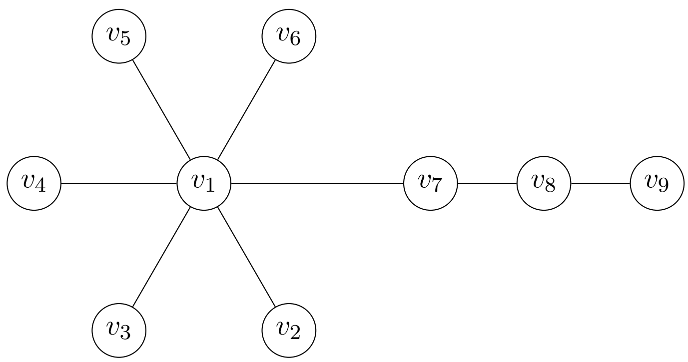

Figure 2.5: Fork Graph with $n = 9$ and $k = 5$

The fork $F_{n,k}$, is a graph on n vertices, containing a path with $n - k$ vertices, where one of the leaves of the path is the center of a star graph with k leaves. $F_{n,k}$ has $n-1$ edges with a diameter of $n-k$. The broadcast time of the fork graph is given by $b(F_{n,k}) = n-1$. Figure 2.5 illustrates $F_{9,5}$.

2.2.6 Wheel Graph W_n

The wheel W_n is a star such that neighboring leaves of the star are connected by an edge. For $V = \{1, \ldots, n\}$, the graph has the following edge set: $E = \{(1, i) | 2 \leq i \leq n\} \cup \{(i, i + 1) | 2 \leq i \leq n\} \cup \{(n, 2)\}$. W_n has $2n - 1$ edges and a diameter of 2 (when $n > 4$). The central vertex has the highest degree of $n - 1$. The vertices exhibit an identical broadcast time, making $b(W_n) = \lceil \frac{\sqrt{4n-3}+1}{2} \rceil$. Figure 2.6 illustrates W_6.

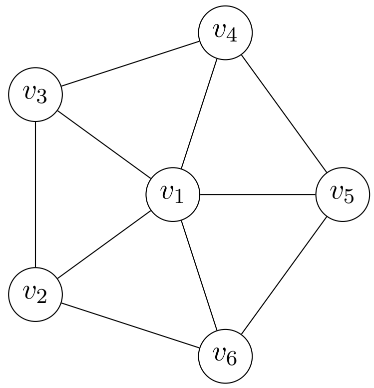

Figure 2.6: Wheel Graph with $n = 6$

2.2.7 Binomial Tree B_d

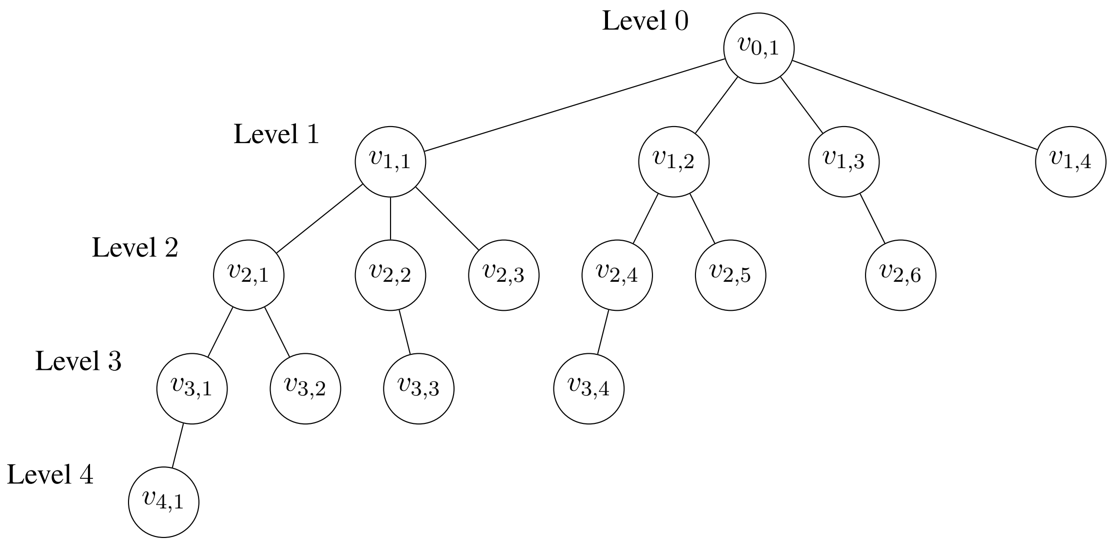

Figure 2.7: Binomial Tree B_4

A binomial tree of dimension d is a tree on $n = 2^d$ vertices and has a recursive construction. B_d can be constructed by connecting two copies of B_{d-1} at their roots and selecting one of the roots as the root of the new tree. The smallest binomial tree B_0 is a single vertex. The number of vertices at level i of a d-dimensional binomial tree is $\binom{d}{i}$ for all $0 \leq i \leq d$. Like other trees, the binomial tree has $n - 1$ edges. The height of a binomial tree is d and the diameter is $2d - 1$. The maximum degree of B_d is d, which is given by the root. The root has a minimum broadcast time $b(root, B_d) = d = \lceil \log n \rceil$, however $b(B_d) = 2d - 1 = 2\lceil \log n \rceil - 1$. Figure 2.7 illustrates B_4.

2.2.8 Hypercube H_d

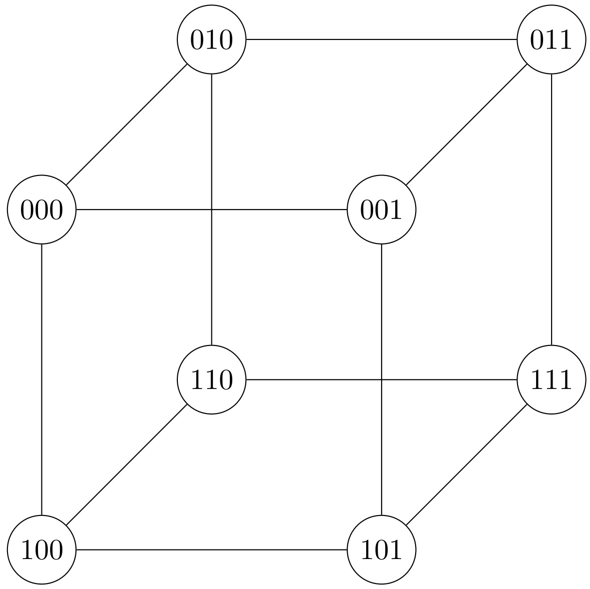

Figure 2.8: Hypercube with dimension $d = 3$

A hypercube of dimension d is a graph on $n = 2^d$ vertices. Considering each vertex to be a d bit binary string, H_d can be constructed by connecting vertices that have a bit-wise difference of one. Thus every vertex has degree d since flipping each of its d bits points to a new neighbor. For example, if vertices u and v represent the binary strings 0110 and 1110, then there would be an edge connecting them. H_d has $d \cdot 2^{d-1}$ edges and a diameter d. As mentioned in Section 2.1.1, it is an infinite family of minimum broadcast graphs and has a broadcast time of $d = \lceil \log n \rceil$. Figure 2.8 illustrates H_3.

2.2.9 Cube-Connected Cycles CCC_d

The cube-connected cycles graph (CCC_d) is a modified hypercube H_d where every vertex is replaced by a cycle on d vertices. CCC_d has $d \cdot 2^d$ vertices, indexed by pairs of numbers (i, j) such that i indexes the cycle in the hypercube and j indexes the position of the vertex within each cycle. Thus, $1 \leq i \leq 2^d$ since there are 2^d cycles and $1 \leq j \leq d$ since there are d vertices within each cycle. Each vertex has three neighbors: two within its cycle, and one in a neighboring cycle. More formally, we can identify the neighbors of a vertex by its indices as follows: $(i, j + 1 \mod d)$,

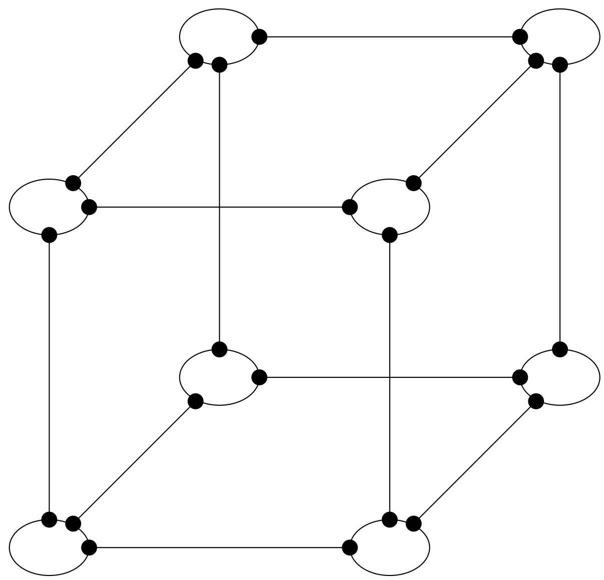

Figure 2.9: Cube-Connected Cycles with $d = 3$

$(i, j - 1 \mod d)$, and $(i \oplus 2^y, j)$, where "$\oplus$" denotes the bitwise-exclusive-or (XOR) operation on binary numbers. CCC_d has $3 \cdot d \cdot 2^{d-1}$ edges and a diameter of $\lfloor \frac{5d}{2} \rfloor - 2$. Its broadcast time is given by $b(CCC_d) = \lceil \frac{5d}{2} \rceil - 1$ (Liestman & Peters, 1988). Figure 2.9 illustrates CCC_3.

2.2.10 Butterfly Graph BF_d

The Butterfly Graph BF_d of dimension d is a graph with vertices arranged in d rows and 2^d columns. The graph has $d \cdot 2^d$ vertices, and its vertex set is given by $V = \{1, 2, \ldots, d\} \times \{0, 1\}^d$ where $\{0, 1\}^d$ is the set of all binary strings of length m. Each vertex $v \in V$ is denoted by the tuple $v = (i, S)$, for $i \in \{1, 2, \ldots, d\}$ and $S \in \{0, 1\}^d$.

The butterfly graph has two types of edges: *straight edges* which connect vertices (i, S) and $((i + 1) \mod d, S)$; and *cross edges* which connect vertices (i, S) and $((i + 1) \mod d, S_{i'})$, where the string $S_{i'} = s_0 \ldots s_{i-1} s'_i s_{i+1} \ldots s_d$ and s'_i is the binary complement of the bit s_i. The butterfly graph is 4-regular, with $d \cdot 2^{d+1}$ edges, and diameter $\lfloor \frac{3d}{2} \rfloor$. The broadcast time of the butterfly graph is bounded by $1.7417d \leq b(BF_d) \leq 2d - 1$ (Klasing, Monien, Peine, & Stöhr, 1994). Figure 2.10 illustrates BF_3.

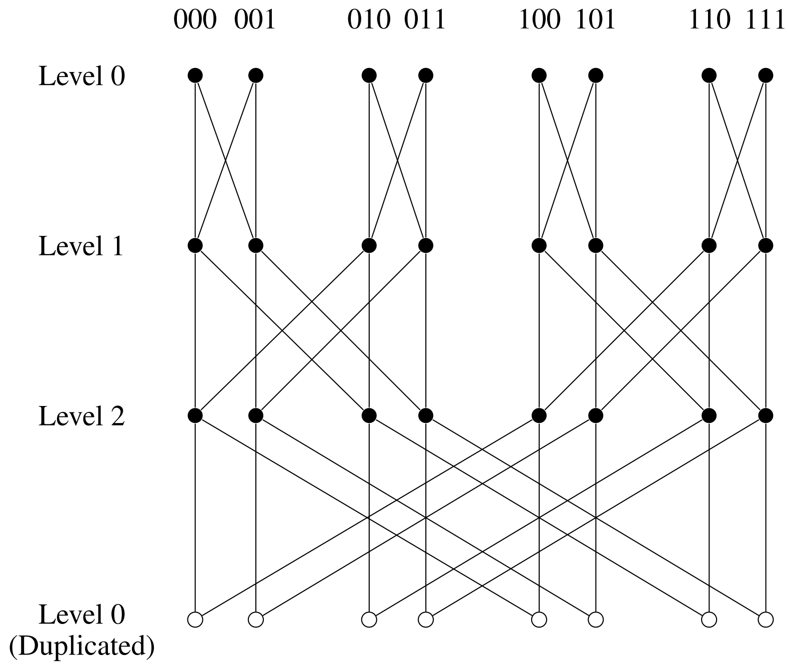

Figure 2.10: Butterfly Graph with $d = 3$

2.2.11 Shuffle-Exchange Graph SE_d

The Shuffle-Exchange Graph SE_d is defined on 2^d vertices. Similar to the hypercube, each vertex in SE_d is denoted by a binary string of length d. SE_d has two types of edges: *shuffle edges* which connect vertices $(S+b)$ and $(b+S)$; and *exchange edges* which connect vertices $(S+b)$ and $(S+b')$, where S is a binary string of length $d-1$, $b \in \{0,1\}$, and b' is the binary complement of b. The graph has a diameter of $2d-1$ and a maximum degree of 3. The broadcast time of the shuffle exchange graph is given by $b(SE_d) = 2d-1$ (Hromkovič, Jeschke, & Monien, 1993). Figure 2.11 illustrates SE_3.

2.2.12 DeBruijn Graph DB_d

The DeBruijn Graph DB_d is a directed graph defined on 2^d vertices. Each vertex is denoted by a binary string of length d. DB_d has two types of edges: *shuffle edges* which connect vertices $(S+b)$ and $(b+S)$; and *shuffle-exchange edges* which connect vertices $(b+S)$ and $(S+b')$, where S is a binary string of length $d-1$, $b \in \{0,1\}$, and b' is the binary complement of b. The graph has a diameter of d and a maximum degree of 4. The broadcast time of the butterfly graph is bounded

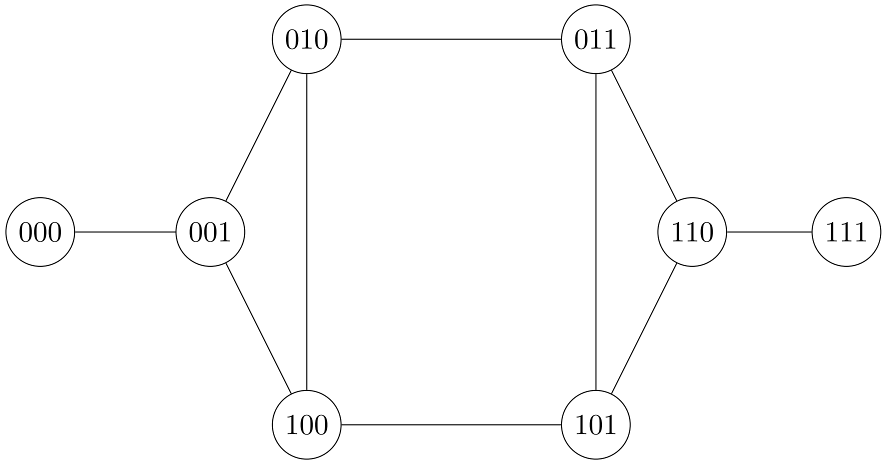

Figure 2.11: Shuffle-Exchange Graph with $d = 3$

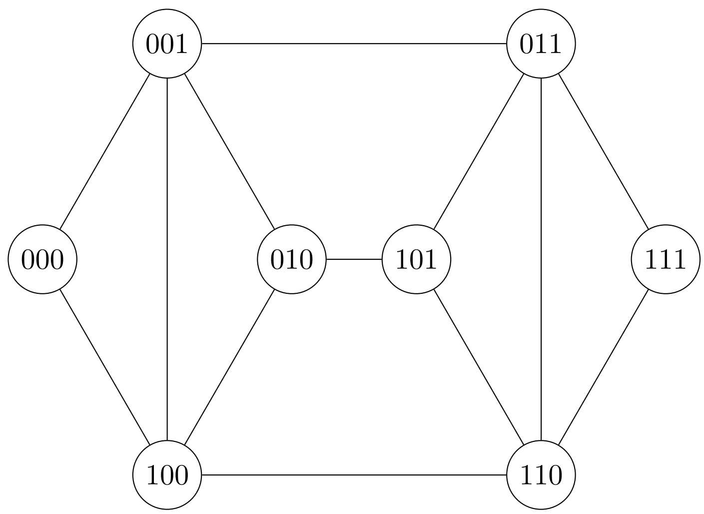

Figure 2.12: DeBruijn Graph with $d = 3$

by $1.3171d \leq b(DB_d) \leq \frac{3}{2}(d+1)$ (Bermond & Peyrat, 1988; Klasing et al., 1994). Figure 2.12 illustrates DB_3.

2.2.13 Grid $G_{m \times n}$

The Grid $G_{m \times n}$ is a graph on $m \times n$ vertices in a 2-dimensional lattice structure. Each vertex represents a unique tuple of positive integers (i, j) for all $1 \leq i \leq m$ and $1 \leq j \leq n$. Two vertices are connected if and only if their corresponding tuples differ in at most one value, and the absolute difference between those values is no more than 1. $G_{m \times n}$ has $(m-1)n+(n-1)m = 2mn-(m+n)$ edges, a diameter of $m + n - 2$, and a maximum degree of 4. The broadcast time of the grid graph

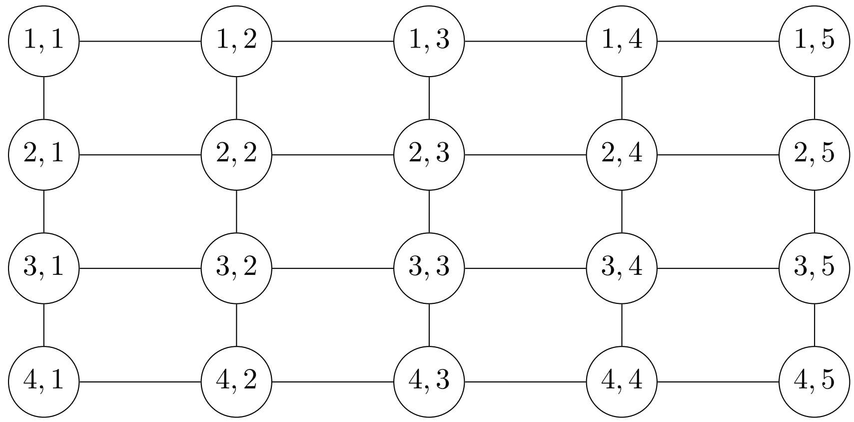

Figure 2.13: Grid with $m = 4$ and $n = 5$

is given by $b(G_{m\times n}) = m + n - 2$ (Farley & Hedetniemi, 1978). Figure 2.13 illustrates $G_{4\times5}$.

2.2.14 Torus $T_{m\times n}$

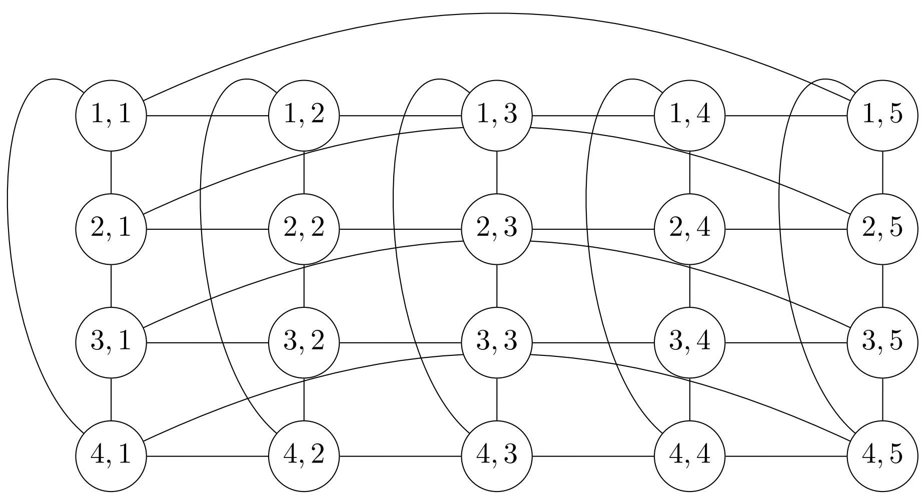

Figure 2.14: Torus with $m = 4$ and $n = 5$

The Torus $T_{m\times n}$ is a graph on $m \times n$ vertices which is constructed by connecting the ends of the rows and columns of the grid $G_{m\times n}$. Consequently, the edge set of the torus is similar to the grid, with the addition of edges $[(i, n), (i, 1)]$ and $[(m, j), (m, 1)]$, for all $1 \le i \le m$ and $1 \le j \le n$, which brings the total number of edges up to $2mn$. The torus is a 4-regular graph, with diameter $\lfloor \frac{m}{2} \rfloor + \lfloor \frac{n}{2} \rfloor$. When m and n are even, the broadcast time of the torus is given by $b(T_{m\times n}) = \frac{m+n}{2}$, and in all other cases $b(T_{m\times n}) = \lfloor \frac{m}{2} \rfloor + \lfloor \frac{n}{2} \rfloor + 1$ (Farley & Hedetniemi, 1978). However, $b(T_{3\times3})$

is an exception. Figure 2.14 illustrates $T_{4\times 5}$.

2.2.15 k-ary Tree T_k^h

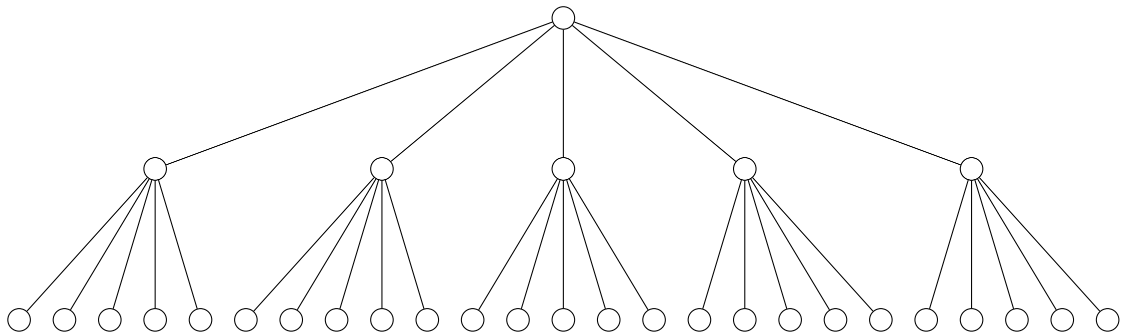

Figure 2.15: Complete k-ary Tree with $k = 5$ and $h = 2$

The k-ary Tree is a rooted tree in which each internal node has k children. As a result, the root has degree k, the internal vertices have degree $k+1$ and the leaves have degree 1. A complete k-ary tree T_k^h is a rooted tree on $\frac{k^{h+1}-1}{k-1}$ vertices with height h, such that all leaves are on the same level h. T_k^h has a diameter of $2h$ and a maximum degree of $k+1$. The broadcast time of a k-ary Tree is given by $b(T_k^h) = kh + h - 1$. Figure 2.15 illustrates T_5^2.

2.2.16 Knödel Graph $KG_{\Delta,n}$

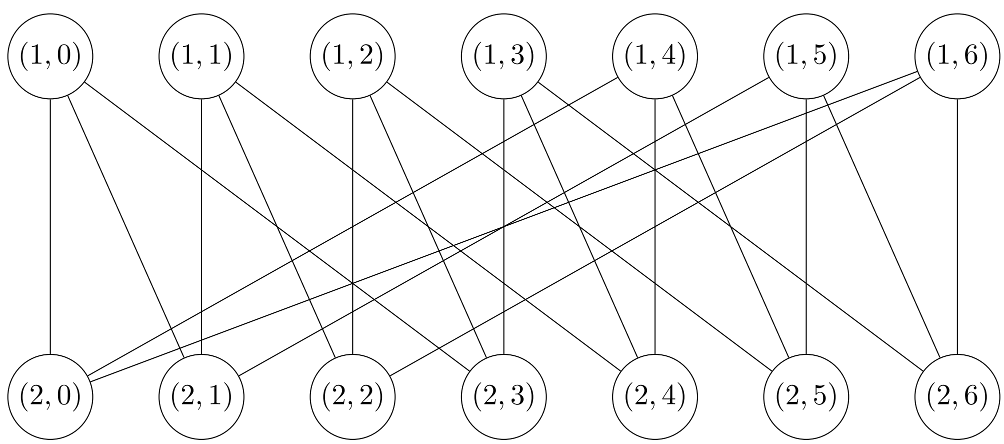

Figure 2.16: Knödel Graph with $\Delta = 3$ and $n = 14$

The Knödel Graph $KG_{\Delta,n}$ is defined as a graph on n vertices where $n = 2k$ for any integer $k \geq 3$. The edge set is given by $E = \{(a,b) \mid a + b \equiv 2^{\Delta} - 1 \mod n\}$, where $1 \leq \Delta \leq$

$\lfloor \log n \rfloor$. When $n = 2^{\Delta}$, the Knödel graph is a Δ-regular graph with diameter $\lceil \frac{\Delta+2}{2} \rceil$. As seen in 2.1.1, $KG_{\Delta,n}$ is a minimum broadcast graph for $n = 2^{\Delta} - 2$ ($\Delta \geq 2$) and $n = 2^{\Delta}$. Hence, $b(KG_{\Delta,2^{\Delta}-2}) = b(KG_{\Delta,2^{\Delta}}) = \Delta = \lceil \log n \rceil$ (Knödel, 1975). Figure 2.16 illustrates $KG_{3,14}$.

2.2.17 Recursive Circulant Graph $RCG_{n,d}$

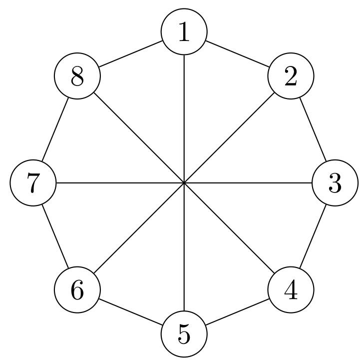

Figure 2.17: Recursive Circulant Graph with $n = 8$ and $d = 4$

The Recursive Circulant graph $RCG_{n,d}$ is a graph on n vertices. The edge set of $RCG_{n,d}$ is given by $E = \{(a, b)|$ there exists $i, 0 \leq i \leq \lceil \log_d n \rceil - 1$, such that $v + d^i \equiv w \mod n\}$, where d is called the *jump*. When $n = 2^k$ and $d = 4$, the recursive circulant graph has $k \cdot 2^{k-1}$ edges and broadcast time $b(RCG_{2^k,4}) = \lceil \log 2^k \rceil = k$. Interestingly, the number of vertices, edges, and broadcast time of $RCG_{2^k,4}$ all match the hypercube H_k (Park & Chwa, 1994). Figure 2.17 illustrates $RCG_{8,4}$.

2.2.18 Complete Bipartite Graph $K_{k,n-k}$

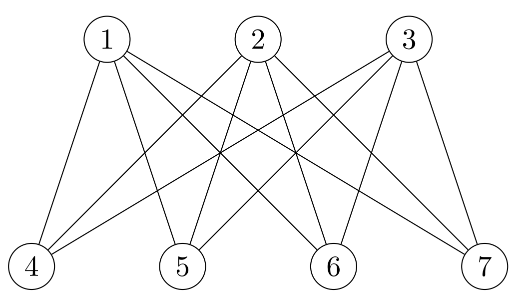

Figure 2.18: Complete Bipartite Graph $K_{3,4}$

The Complete Bipartite Graph $K_{k,n-k}$ is a graph composed of two partitions P_1 and P_2 with

k and $n-k$ vertices each, such that there are no edges between vertices that belong to the same partition. The edge set of $K_{k,n-k}$ is given by $E = \{(a,b) \mid \forall a \in P_1, \forall b \in P_2\}$. Consequently, the degree of every vertex in P_1 is $n-k$ and the degree of every vertex in P_2 is k (Gholami & Harutyunyan, 2024). Figure 2.18 illustrates $K_{3,4}$.

2.3 Additional Models of Broadcasting

Until now, our discussion has been centered on the Classical Model of Broadcasting as described in Section 2.1. In this section, we present various broadcast models previously investigated in the literature, which differ in their communication setup between vertices. However, the main focus of this study will be on the classical model.

2.3.1 Multiple Originator Broadcasting

The Multiple Originator model introduces the concept of the message being transmitted within the network from multiple originators. Instead of a single originator, there can be several nodes with the message that need to broadcast it to all other nodes. One interesting problem in this model is calculating the number of originators required to inform every vertex within some target broadcast time. Farley and Proskurowski (1981) proposed a linear algorithm for decomposing trees into a minimum number of subtrees while ensuring that broadcasting can be completed in t time units within every subtree. This model is similar to the k *shortest path spanning tree* problem ($k-SPST$) which is: Given a graph G with the length function l, k sources $s_1, \ldots, s_k \in V$, and a positive integer K is there a spanning tree T of G whose cost (the sum of edge lengths on the paths from all sources to all vertices) does not exceed K? This is defined by Farley, Fragopoulou, Krumme, Proskurowski, and Richards (2000) and shown to be NP-Complete. In addition, the k-originator broadcast time of the complete k-partite graph and the Hypercubes have been shown by Chia, Kuo, and Tung (2007) along with the 2-originator broadcast time of Grids.

2.3.2 Multiple Message Broadcasting

In this model, the originator has k messages to be transmitted to all vertices. $B_m(n)$ denotes the minimum number of edges in any m-message broadcast graph on n vertices. H. A. Harutyunyan (2006) presents bounds on $B_m(n)$. Suderman (1999) investigated multiple message broadcasting across common topologies such as trees, cycles, and paths, while Gregor, Škrekovski, and Vukašinović (2018) discuss the model in the three infinite families of minimum broadcast graphs: Hypercube, Knodel and Circulant graphs as well as Tori and other topologies.

2.3.3 Fault-tolerant Broadcasting

The fault-tolerant model facilitates broadcasting in the presence of k faulty edges or vertices. This model can simulate failures occurring in real-life scenarios and hence is a critical area of research. A faulty edge stops the transmission of information after some time unit, either permanently or temporarily (transiently). Nodes are also susceptible to failures, where they might encounter issues with sending, receiving, or both. However, the connectivity of the graph is maintained at all times. The model was introduced by Liestman (1985). Later, Ahlswede, Gargano, Haroutunian, and Khachatrian (1996) studied the minimum number of communication lines of any minimal k-fault-tolerant broadcast network on n vertices. Pelc (1996) explored different types of failures including permanent edge failure, permanent vertex failure, permanent edge and vertex failure, transient faults, as well as distribution of failures and probabilistic fault model.

2.3.4 k-broadcasting

The k-broadcasting model is a generalization of the classical model. In this model, an informed vertex informs up to k of its neighboring vertices in each time unit. Thus, the classical model is a specific case of the k-broadcasting model where $k = 1$. k-broadcasting in general graphs has been studied by Grigni and Peleg (1991); H. A. Harutyunyan and Liestman (2001a), and k-broadcasting in trees has been studied by H. A. Harutyunyan and Liestman (2001b); Labahn (1986).

2.3.5 Universal-list Broadcasting

In this model, every vertex is given a universal list of vertices. When a vertex v receives the message, it informs the neighbors by the order in the list, regardless of the originator. This model was introduced by Diks and Pelc (1996); Rosenthal and Scheuermann (1987) with two submodels. The **adaptive model** allows the vertices to keep track of the neighbors from which it received a message and skip them while going through the list to minimize redundant calls. On the other hand, the **non-adaptive model** does not allow vertices to distinguish where the message comes from, resulting in redundant calls and possibly worse broadcast time overall. Gholami and Harutyunyan (2024); H. A. Harutyunyan, Liestman, Makino, and Shermer (2011) discuss broadcasting in trees under the non-adaptive model. In addition, Gholami and Harutyunyan (2022) introduced the **fully adaptive model** where vertices skip all of their informed neighbors while following the prescribed list.

2.3.6 Messy Broadcasting

In this model, each vertex sends the message randomly to its neighbors without any knowledge about the originator or the time at which the message was sent. Vertices only know to transmit to their neighbors, regardless of whether they are informed. They are also unaware of the current time unit and the originator since there is no centralized coordination. The model was introduced by Ahlswede, Haroutunian, and Khachatrian (1994) and is useful to obtain upper bounds on broadcast time due to its unpredictable behavior. There are three types of messy models, depending on the information available to the vertices:

- *Model M1*: Each vertex knows the state of its neighbors (informed or uninformed) at any time unit.
- *Model M2*: Each vertex considers the vertices it has received the message from to be informed.
- *Model M3*: Each node keeps a list of all neighbors to which it sends a message. Once informed, in each time unit the vertex sends the message to a neighbor not present in the list.

H. A. Harutyunyan and Liestman (1998) studied broadcasting under all three messy models for complete graphs, paths, cycles, and complete k-ary trees. Li, Hart, Henry, and Neufeld (2008) studied the average-case messy broadcasting time of stars, paths, cycles, complete k-ary trees, and hypercubes, and Comellas, Harutyunyan, and Liestman (2003) have studied complete bipartite graphs and multidimensional directed tori.

2.3.7 Radio Broadcasting

In this model, similar to the k-broadcasting model, the message transmitted by a vertex in a given time unit is delivered to all of its neighbors. However, a vertex acting as a receiver successfully receives a message only if precisely one of its neighbors transmits during that time unit. If two or more neighbors transmit simultaneously, a collision occurs, resulting in none of the messages being received by the vertex in that time unit. This simulates real-world collision scenarios encountered when multiple incoming messages share the same frequency.

Chapter 3

Broadcasting in Stars of Cliques

In this chapter, we study the broadcasting problem in the Windmill graph $Wd_{k,l}$, a graph on n vertices containing k cliques of size l, and a universal vertex. $Wd_{k,2}$ appears as a solution to the Friendship Theorem studies by Erdős, Rényi, and Sós (1966) and is known as the Friendship graph. It is defined on k pairs of vertices where every distinct pair has exactly one common adjacent vertex. Other popular sub-families of windmill graphs include the Dutch windmill (k cliques of 2 vertices each) and the French windmill (k cliques of 3 vertices each), which have been studied in the topic of graceful and harmonious graph labeling along with the radio-astronomy problem of movable antennae (Bermond, 1979). We also introduce *Star of Cliques*, a generalization of the Windmill graph composed of cliques with arbitrary sizes. We present an $O(n \cdot \log n)$ algorithm to find the broadcast time of any vertex in an arbitrary star of cliques and discuss the optimality of our algorithm.

3.1 Broadcasting in Windmill Graphs

Let $Wd_{k,l}$ be a windmill graph on n vertices containing k cliques $C_1, C_2, \ldots, C_k$ each of size l, and a vertex u connected to all other vertices. We refer to vertex u as the universal vertex.

In any windmill graph, all vertices have equal broadcast time; at time unit 1, either vertex u informs a vertex in a clique, or vice versa. In either case, by time unit 1, the universal vertex and a vertex from one of the cliques will be the only informed vertices. Since all cliques are the same size, the broadcast time is the same no matter which clique was informed first. Consequently, we opt to regard only the universal vertex as the originator for simplicity and symmetry.

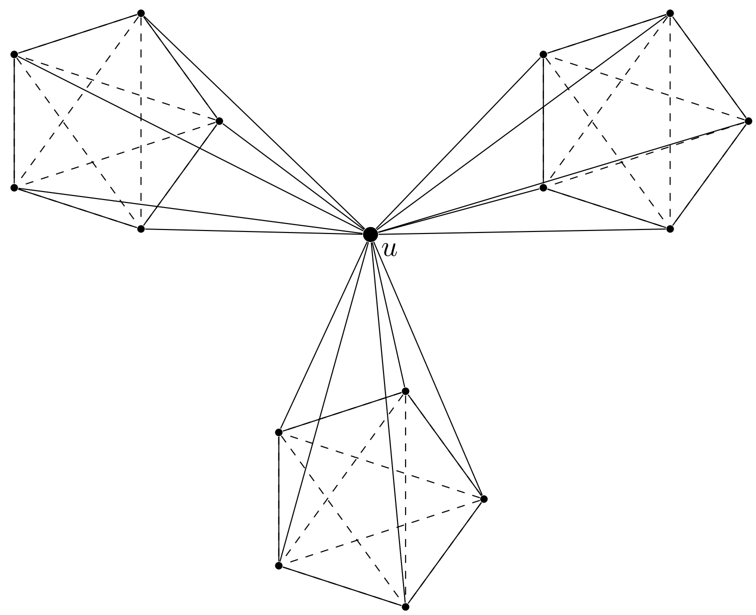

Figure 3.1: Windmill Graph $Wd_{k,l}$ with $k = 3$ and $l = 5$

Lemma 1. *Let $Wd_{k,l}$ be a windmill graph on n vertices containing k cliques $C_1, C_2, \ldots, C_k$ each of size l, and a universal vertex u. There exists an optimal broadcast scheme, wherein after informing one vertex in each of the first $k-1$ cliques, the universal vertex will broadcast only to the last clique starting at time unit k.*

Proof. Let S_{alg} be a broadcast scheme where during the first $k-1$ time units, vertex u informs a vertex in each clique C_i for all $1 \leq i \leq k-1$, ensuring that broadcasting in C_i finishes by time $i + \lceil \log l \rceil$ without additional broadcasting from vertex u. Assume that starting at time unit k, the originator exclusively informs vertices in C_k until all l vertices within it are informed, effectively creating a clique $C_k \cup \{u\}$ with $l+1$ vertices. Broadcasting in $C_k \cup \{u\}$ finishes at time $k - 1 + \lceil \log(l+1) \rceil$, while broadcasting in C_{k-1} finishes at $k - 1 + \lceil \log l \rceil$, indicating that broadcasting in C_k finishes no sooner than any other clique.

By contradiction, assume there exists a optimal broadcast scheme S_{opt} where the originator informs more than one clique multiple times. Since the graph has k cliques and C_k is the last clique that vertex u informs for the first time, C_k first receives the message at time unit p such that $p \geq k$. Let $b(C_k)$ be the time the last vertex of C_k gets informed under scheme S_{opt}. Then, $b_{S_{alg}}(u, Wd_{k,l}) = b(C_k)$ since C_k is the last to finish broadcasting in scheme S_{alg}, despite being the only clique to be informed multiple times. Similarly, $b_{S_{opt}}(u, Wd_{k,l}) = b(C_k) \geq p-1+\lceil \log(l+1) \rceil$ since by assumption, C_k is not the only clique to be informed multiple times by the originator. We can see that $b_{S_{alg}} \leq b_{S_{opt}}$ since $b_{S_{alg}} = k - 1 + \lceil \log(l+1) \rceil$ and $k \leq p$. Hence there exists a minimum broadcast scheme where after time $k - 1$, vertex u only informs vertices in clique C_k. Thus, $b_{S_{alg}}(u, Wd_{k,l}) = k - 1 + \lceil \log(l+1) \rceil$.

Algorithm 1 Broadcast algorithm for Windmill graph from universal vertex u

Input Windmill graph $Wd_{k,l}$, originator vertex u.
Output A broadcast scheme for $Wd_{k,l}$.

for $i = 1 \to k$ **do**
    u informs a vertex v_i in clique C_i
    v_i broadcasts in C_i
end for
while C_k has an uninformed vertex **do**
    u broadcasts in C_k
end while

□

According to Lemma 1, broadcasting in a Windmill graph can be described as $k - 1$ rounds of the universal vertex informing one vertex in each clique $C_1 \ldots C_{k-1}$. These informed vertices broadcast within their respective cliques while vertex u acts as a part of C_k, informing only vertices in C_k after time $k - 1$. Broadcasting will finish in another $\lceil \log(l+1) \rceil$ time units. Thus, the broadcast time of the Windmill graph is given by,

$$b(G) = k - 1 + \lceil \log(l+1) \rceil$$

3.2 Bounds on Broadcast Time of Star of Cliques

We define a *Star of Cliques* G_k to be a connected graph consisting of k cliques $C_1, C_2, \ldots, C_k$, each of arbitrary sizes, and a vertex u connected to all other vertices. We refer to vertex u as the universal vertex. Assume $l_1 \geq l_2 \geq \cdots \geq l_k \geq 1$ where l_i is the number of vertices in clique C_i for all $1 \leq i \leq k$. Figure 3.2 depicts a star of cliques composed of 3 cliques with 10, 9, and 5 vertices respectively.

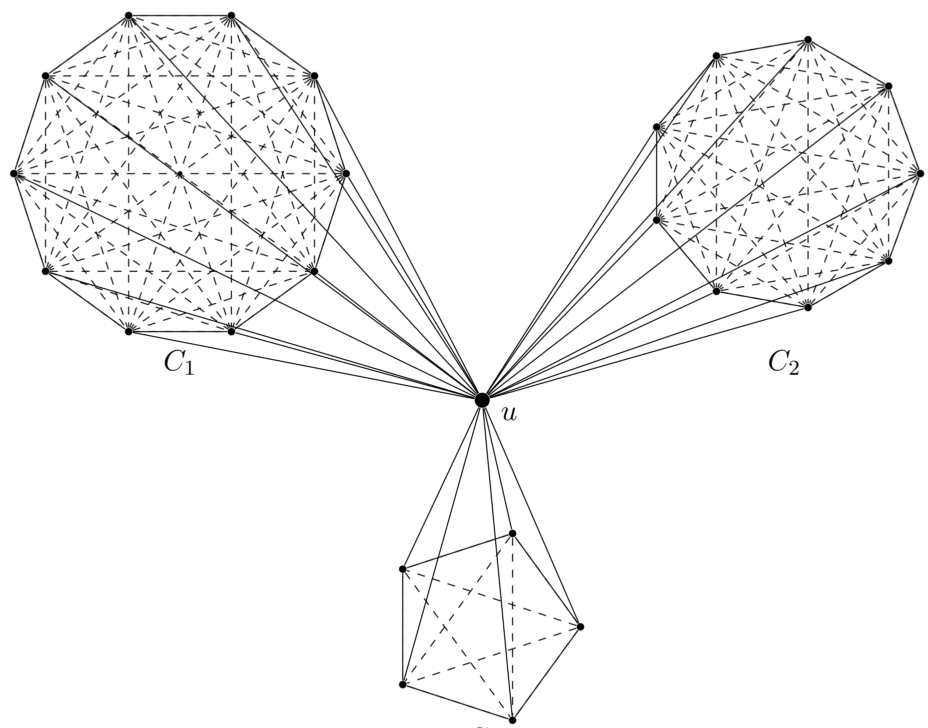

Figure 3.2: Star of Cliques with 3 cliques of sizes 10, 9, 5

Lemma 2. *Let G_k be a star of cliques and the universal vertex u be the originator. Then, $b(u, G_k) \geq \max_{1 \leq i \leq k} \{i - 1 + \lceil \log{(l_i + 1)} \rceil\}$.*

Proof. Consider a clique C_p, where $1 \leq p \leq k$. For any such C_p, by definition, there are at least p cliques that have no fewer than l_p vertices. Thus, by the pigeonhole principle, there exists a clique C_q which is first informed in time unit $t_q \geq p$ and $q \leq p$. Let $b(C_q)$ be the time the last vertex of C_q gets informed under any optimal broadcast scheme. Assume vertex u informs other cliques up to time $t_q - 1$ and thereafter only informs vertices in C_q. Thus, $b(C_q) \geq t_q - 1 + \lceil \log{(l_q + 1)} \rceil \geq p - 1 + \lceil \log{(l_p + 1)} \rceil$ as $t_q \geq p$ and $l_q \geq l_p$.

We know that $b(u, G_k) = \max_{1 \leq i \leq k} \{b(C_i)\} \geq \max_{1 \leq i \leq k} \{i - 1 + \lceil \log{(l_i + 1)} \rceil\}$. Hence, $b(u, G_k) \geq \max_{1 \leq i \leq k} \{i - 1 + \lceil \log{(l_i + 1)} \rceil\}$. □

Lemma 3. *Let G_k be a star of cliques and the universal vertex u be the originator. Then, $b(u, G_k) \leq \max_{1 \leq i \leq k} \{i + \lceil \log{l_i} \rceil\}$.*

Proof. Consider the broadcast scheme in Algorithm 2 where the originator u informs exactly one vertex in each clique in descending order of clique sizes; u informs a vertex in clique C_i at time unit i for all $1 \leq i \leq k$.

Algorithm 2 Broadcast algorithm for Star of Cliques from the universal vertex u

Input Star of Cliques G_k $(l_1 \geq l_2 \geq \cdots \geq l_k \geq 1)$.
Output A broadcast scheme for G_k with time $\max\limits_{1\leq i\leq k}\{i + \lceil \log l_i \rceil\}$.

for $i = 1 \rightarrow k$ **do**
  u informs a vertex v_i in clique C_i
  v_i broadcasts in C_i
end for

When a clique has one informed vertex, it requires another $\lceil \log l_i \rceil$ time units to inform all other vertices within the clique. Then all vertices of C_i will be informed by time $i + \lceil \log l_i \rceil$. Thus, $b(u, G_k) \leq \max\limits_{1\leq i\leq k}\{i + \lceil \log l_i \rceil\}$

□

Lemma 4. *Let G_k be a star of cliques where the originator is the universal vertex u. Then, the difference between the upper and lower bounds of $b(u, G_k)$ is at most 1.*

Proof. Assume C_p is a clique that maximizes the upper bound from Lemma 3 such that $UB = \max\limits_{1\leq i\leq k}\{i + \lceil \log l_i \rceil\} = p + \lceil \log l_p \rceil$. Then, from the lower bound in Lemma 2 we have $LB = \max\limits_{1\leq i\leq k}\{i - 1 + \lceil \log (l_i + 1) \rceil\} \geq p - 1 + \lceil \log (l_p + 1) \rceil \geq p + \lceil \log l_p \rceil - 1 = UB - 1$. Hence, $LB + 1 \geq UB$.

Also, assume C_q is a clique that maximizes the lower bound such that $LB = \max\limits_{1\leq i\leq k}\{i - 1 + \lceil \log (l_i + 1) \rceil\} = q - 1 + \lceil \log (l_q + 1) \rceil$. Then, $UB = \max\limits_{1\leq i\leq k}\{i + \lceil \log l_i \rceil\} \geq q + \lceil \log l_q \rceil = q - 1 + \lceil \log (2l_q) \rceil \geq q - 1 + \lceil \log (l_q + 1) \rceil = LB$. Hence, $UB \geq LB$. Putting it together, we have $LB \leq UB \leq LB + 1$. □

Observation 1. From Lemma 4, we know $b(u, G_k)$ has two possible values. Thus, Algorithm 2 is a 1-additive approximation algorithm.

3.3 Decision Algorithm for the Broadcast Time of Star of Cliques

Let G_k be a star of cliques where the originator is the universal vertex u. While Algorithm 2 provides a 1-additive approximation algorithm, we are left with the problem of determining the exact value of $b(u, G_k)$ for an arbitrary star of cliques. In this section, we discuss a decision algorithm that determines if broadcasting in a star of cliques from the universal vertex can be completed in $t = t' - 1$ time units where $t' = \max_{1 \le i \le k} \{i + \lceil \log l_i \rceil\}$ is the upper bound of a given star of cliques and t is the target broadcast time. We also show how this decision algorithm can be used to determine the broadcast time of a given star of cliques for an arbitrary originator.

Let $CS[1 \dots k]$ be a 1-indexed array of binary strings, sorted in decreasing order, where each string represents a clique in G_k. Each binary string $CS[i]$, when read left to right, represents l_i, the number of vertices in clique C_i, for all $1 \le i \le k$, such that $l_1 \ge l_2 \ge \dots \ge l_k$. Every clique string $CS[i]$ has t bits (smaller strings are left-padded with zeros). For instance, if C_1 is the clique K_9 and the target broadcast time is 5 time units, then $CS[1]$ would be the binary string "01001".

Each bit $CS[i][j]$ represents the action required by C_i from vertex u at time unit j, for $1 \le j \le t$. If $CS[i][j]$ is 1, vertex u must inform a vertex in C_i at time unit j; if $CS[i][j]$ is 0 no action is required. Thus, if $CS[1]$ is "01001" and target broadcast time $t = 5$, then vertex u must inform a vertex in C_1 at time units 2 and 5. No actions are required in the other time units.

When u informs a vertex in a clique at time unit j it can lead to 2^{t-j} vertices being informed by time unit t. This phenomenon is perfectly captured by binary numbers since any integer can be represented as a sum of the powers of 2. The number of vertices in C_i can be represented by $l_i = 2^{p_1} + 2^{p_2} + \dots + 2^{p_c}$ where $p_1 \dots p_c$ can be any integers. If $CS[i][j]$ is 1, it means that $l_i \ge 2^{t-j}$, and informing C_i at time unit j results in 2^{t-j} vertices being informed by time unit t. Let $j_1 \dots j_c$ be the indices of the bits within the string $CS[i]$ which are 1. Then $l_i = 2^{p_1} + 2^{p_2} + \dots + 2^{p_c} = 2^{t-j_1} + 2^{t-j_2} + \dots + 2^{t-j_c}$. If u informs a new vertex in C_i at every time unit where $CS[i][j]$ is 1, then by time unit t all vertices of C_i will be informed. Thus, the binary strings provide an individual broadcast scheme for each clique.

It is also interesting to note that if u informs a vertex in C_i when $CS[i][j]$ is 0, then C_i never needs to be informed again even if any remaining bits in $CS[i]$ are 1, since $2^{t-j} > 2^{t-j+1} +$

$2^{t-j+2} + \ldots + 2^1 + 2^0$ (eg: $1000 > 0111$).

3.3.1 Broadcasting from the Universal Vertex u

In Algorithm 3, we present STARОFCLIQUESBROADCAST(CS, t), which returns $True$ if broadcasting in a given star of cliques from the universal vertex u can be completed in t time units, and returns $False$ otherwise.

Definition 1. Unaddressed Vertices: *When a vertex in some clique C_p is informed at time j, this vertex becomes the root of a binomial tree B_{t-j} within C_p. The vertices in B_{t-j} are guaranteed to be informed at or before time t. All uninformed vertices in C_p that do not belong to such a binomial tree are called unaddressed vertices.*

In the algorithm STAROFCLIQUESBROADCAST(CS, t), we first check if vertex u needs to inform an unaddressed vertex in C_1 at the current time unit, i.e. if $CS[1][1] = 1$. If so, this means that $l_1 \geq 2^{t-j}$. Hence, u cannot inform any other clique in the current time unit as that would result in C_1 not being able to finish broadcasting, since $l_1 \geq 2^{t-j} > 2^{t-j+1} + 2^{t-j+2} + \ldots + 2^1 + 2^0$ and we cannot make up for the lost vertices. However, if $CS[2][1] = 1$ as well, then there is a conflict and the algorithm returns $False$. Note that it is sufficient to check only $CS[2]$ for conflicts since the array is in descending order and the remaining cliques $C_3 \ldots C_k$ are no bigger than C_2. On the other hand, if $CS[1][1] \neq 1$, this implies that no clique needs to be informed in the current time unit. But instead of keeping u idle, by broadcasting to C_1 anyway, we ensure that C_1 never needs to be informed again, potentially eliminating future conflicts. Either way, we inform an unaddressed vertex in C_1. Then, we recursively call STAROFCLIQUESBROADCAST(CS, $t-1$) to begin the next iteration where $CS[i] = CS[i][2\ldots t]$, i.e. the first bit, which represents the now elapsed time unit, is removed from every binary string. Finally, the algorithm runs until two cliques need to be informed in the same time unit, in which case it returns $False$, or we make it to the last time unit and it returns $True$.

Algorithm 3 Decision algorithm for broadcasting in Star of Cliques from the universal vertex u in t time units. (*Note:* Arrays are indexed starting from 1)

Input $CS[1 \dots k]$: Cliques in G_k as descending sorted binary strings, t: target broadcast time.
Output $True$ if $b(u, G_k) = t$; $False$ otherwise.

```
procedure StarOfCliquesBroadcast(CS, t)
    if CS[1][1] == "1" then
        if CS.length > 1 and CS[2][1] == "1" then
            return False
        end if
    else
        CS[1] ← 1 + 0...0   (t−1 zeros)
    end if
    if t == 1 then
        return True
    else
        for i ← 1 to k do
            CS[i] ← substring(CS[i], 2, t)
        end for
        CS.sort()
        return StarOfCliquesBroadcast(CS, t − 1)
    end if
end procedure
```

$$CS[1] \leftarrow 1 + \underbrace{0 \dots 0}_{t-1}$$

Complexity Analysis

Step 7 takes $O(t)$ time to update a string if the corresponding clique is informed early. Step 13 takes $O(k \cdot t)$ time to remove the first bit of all k strings. In Step 15, since only the first element of the array is out of place, the sorted array can be obtained through one iteration of insertion sort in $O(k)$ time. One iteration of the STARОFCLIQUESBROADCAST algorithm takes $O(k \cdot t)$ time. The algorithm is called t times, which in total requires $O(k \cdot t^2)$ time. In addition, $O(k \cdot \log k)$ time is required to calculate the upper bound t', and $O(k \cdot \log l_1)$ time is required to generate the binary strings for all cliques. The maximum possible value of t is $k + \lceil \log l_1 \rceil$. We know that $k, l_1 \leq n$, since $n = l_1 + l_2 + \ldots + l_k$ is the total number of vertices in the graph. Thus, $t, k = O(n)$, which brings the overall complexity of the algorithm to $O(n^3)$.

3.3.2 Algorithm Correctness

Lemma 5. (Greedy Choice) *Let G_k be a star of cliques where the originator is the universal vertex u and $l_1 \geq l_2 \geq \cdots \geq l_k \geq 1$ where l_i is the number of vertices in clique C_i (excluding vertex u) for all $1 \leq i \leq k$. Then, there is an optimal broadcast scheme where the largest clique C_1 is informed in the first time unit.*

Proof. Let S be a minimum broadcast scheme such that $b(u, G_k) = t_S$. We consider the following cases based on the size of clique C_1:

$l_1 \geq 2^{t_S - 1}$: When a vertex is informed at time x, it can inform a maximum of $2^{t_S - x}$ vertices within its clique. Thus, even if u informs a new vertex in C_1 at every time unit except the first time unit, C_1 will have a maximum of only $2^{t_S-2} + 2^{t_S-3} + \ldots + 2^1 + 2^0 = 2^{t_S-1} - 1$ informed vertices by time t_S. Thus, a vertex in C_1 must be informed in the first time unit.

$l_1 < 2^{t_S-1}$: In some graphs, C_1 has to be informed in the first time unit despite $l_1 < 2^{t_S-1}$ since other cliques need to be informed in the remaining time units and we may not be able to inform even $2^{t_S-1} - 1$ vertices as seen above.

However, for the other cases where $l_1 < 2^{t_S-1}$ and it is possible to inform C_1 later than the first time unit and still finish in t_S time units, we aim to prove that informing C_1 in the first time unit does not affect the broadcast time. By contradiction, let some clique C_x be informed in the first time unit and C_1 be first informed at some time j in the minimum broadcast scheme S. Let Q_i be the set of time units where u informs a vertex in C_i for $1 \leq i \leq k$. We must prove that (1) Q_1 will be sufficient to inform all vertices in C_x and (2) Q_x will be sufficient to inform all vertices in C_1.

The first part is trivial; since $l_x \leq l_1$, Q_1 will be enough to inform all vertices in C_x. For the second part, as seen earlier, since C_1 doesn't have to be informed in the first time unit, $l_1 < 2^{t-1}$. Therefore time unit 1 alone is enough for all vertices in C_1 to be informed. Hence there exists a minimum broadcast scheme where C_1 is informed at time unit 1. □

Lemma 6. (Optimal Substructure) *Let G_k be a star of cliques where the originator is the universal vertex u and the set $S_j = \{C_1^j, \ldots, C_k^j\}$ be the unaddressed vertices in cliques $C_1, \ldots, C_k$ respectively at time unit j. Then, an ordering of S_j which is an optimal broadcast scheme from the universal vertex u must also include an optimal broadcast scheme for S_{j+1}.*

Proof. Let $A_j = (i_0, i_1, \ldots, i_{t-j})$ be the sequence of calls at time units $j, j+1, \ldots, t$ where $1 \leq i_1, \ldots, i_n \leq k$. In particular, the originator informs a vertex in C_{i_p} at time unit $j+p$ for all $1 \leq p \leq t-j$. Note that the sequence A_j may contain multiple references to the same clique. For example, $i_0 = i_3 = i_{t-j} = 1$ means that at time units $j, j+3$ and t clique C_1 is called. Let $S_{j+1} = \{C_1^{j+1}, \ldots, C_k^{j+1}\}$. We are left with the subproblem of finding an ordering which is an optimal broadcast scheme from u in S_{j+1}. Let $A_{j+1} = A_j \setminus \{i_0\}$ be such an ordering, where A_{j+1} is a subsequence of A_j. Since A_j and A_{j+1} start at time j and $j+1$ respectively but finish at the same time, $j + b_{A_j}(u) = (j+1) + b_{A_{j+1}}(u)$ or $b_{A_j}(u) = b_{A_{j+1}}(u) + 1$.

We want to prove that an optimal solution to S_j must also include an optimal solution to S_{j+1}. If a sequence A'_{j+1} existed such that $b_{A'_{j+1}}(u) < b_{A_{j+1}}(u)$, then we would use A'_{j+1} rather than A_{j+1} in a solution to the subproblem of S_j, which contradicts the assumption that A_j is an optimal solution, since $b_{A'_{j+1}}(u) + 1 < b_{A_{j+1}}(u) + 1 = b_{A_j}(u)$. □

Consider the star of cliques depicted in Figure 3.2, consisting of three cliques, K_{10}, K_9, and K_5, along with a universal vertex u. From Lemmata 2 and 3, the lower bound on the broadcast time $b(u, G_k)$ when the universal vertex serves as the originator is 5 time units, while the upper bound is 6 time units. By applying the procedure outlined in Algorithm 3, we can attempt to identify a broadcasting scheme that completes broadcasting within 5 time units. If no such scheme is found, we conclude that the broadcast time for the given graph, with the universal vertex as the originator, is 6 time units.

Consider the binary strings 01010, 01001, and 00101, which represent the cliques K_{10}, K_9, and K_5, respectively. Analyzing these strings from left to right reveals that no cliques need to be informed at time units 1 and 4. Additionally, there are conflicts at time unit 2 between K_{10} and K_9, and at time unit 5 between K_9 and K_5.

Instead of remaining idle at time unit 1, the universal vertex can inform a vertex in K_{10}. By doing so earlier than necessary, the universal vertex no longer needs to inform K_{10} again, since $10000 > 01010$. In other words, the universal vertex can address $2^{5-1} = 16$ vertices with this action, leaving no uninformed vertices in K_{10}, which helps resolve the conflict between K_{10} and K_9. At time unit 2, the universal vertex informs a vertex in K_9, addressing $2^{4-1} = 8$ vertices. At time unit 3, the universal vertex informs a vertex in K_5, addressing $2^{3-1} = 4$ vertices. At time unit 4, rather than remaining idle, the universal vertex informs another vertex in K_9, addressing $2^{2-1} = 2$ vertices. This action resolves the conflict between K_9 and K_5. Finally, at time unit 5, the universal vertex informs the last uninformed vertex in K_5, completing the broadcast process in $6 - 1 = 5$ time units, which is faster than the upper bound.

3.3.3 Bounds on Broadcast Time from an Arbitrary Originator

Let G_k be a star of cliques with universal vertex u. Let an arbitrary vertex v be the originator where $v \neq u$ and $v \in C_i$ for $1 \leq i \leq k$. Consider the broadcast scheme for an arbitrary originator presented in Algorithm 4. In an optimal broadcast scheme, v will inform u at time unit 1 since u is connected to every vertex in G_k. If $v \in C_1$, then $b(v, G_k) = b(u, G_k)$ since in either case after time unit 1 the two informed vertices in G_k will be v and u. When $C_i \neq C_1$ but $l_i = l_1$, still $b(v, G_k) = b(u, G_k)$ since we can swap the calls of C_i and C_1. For all other cases, we have the following bounds.

Lemma 7. *Let G_k be a star of cliques where the originator $v \in C_i$, for $1 < i \leq k$ and $|C_i| < |C_1|$. Then, $b(v, G_k) \geq b(u, G_k \setminus C_i) + 1$.*

Proof. As seen earlier, v informs u at time 1. Since v cannot contribute to broadcasting in $G_k \setminus C_i$ and $|G_k \setminus C_i| \geq |C_1| > |C_i|$, u exclusively handles broadcasting in $G_k \setminus C_i$ while v handles broadcasting in C_1. The broadcast time of u in $G_k \setminus C_i$ can be determined using Algorithm 3. Moreover, since $|G_k \setminus C_i| > |C_i|$, $b(u, G_k \setminus C_i) \geq b(v, C_i)$. Hence, including time unit 1, we have $b(v, G_k) \geq b(u, G_k \setminus C_i) + 1$. □

Lemma 8. *Let G_k be a star of cliques where the originator $v \in C_i$, for $1 < i \leq k$ and $|C_i| < |C_1|$. Then, $b(v, G_k) \leq b(u, G_k \setminus C_i) + 1$.*

Proof. Consider the following broadcast algorithm. First, v informs u. Then u informs vertices in $G_k \setminus C_i$ according to Algorithm 3 and v informs vertices in C_i. As seen earlier, $b(u, G_k \setminus C_i) \geq b(v, C_i)$. Hence, including time unit 1, we have $b(v, G_k) \leq b(u, G_k \setminus C_i) + 1$.

Algorithm 4 Broadcast algorithm for Star of Cliques from an arbitrary originator v

Input Star of Cliques G_k, originator vertex v
Output A broadcast scheme for G_k from originator v with time $b(v, G_k) \leq b(u, G_k \setminus C_i) + 1$ where u is the universal vertex.

1: v informs u at time 1
2: u broadcasts to $G_k \setminus C_i$ according to Algorithm 3
3: v broadcasts to C_i

□

Observation 2. From Lemmata 7 and 8, we can see that when $v \neq u$ and $v \in C_i$, $b(v, G_k) = b(u, G_k \setminus C_i) + 1$ for all $1 < i \leq k$ if $l_1 > l_i$. For all other cases, $b(v, G_k) = b(u, G_k)$.

3.3.4 Improved Decision Algorithm for the Broadcast Time of Star of Cliques

Algorithm 3 can determine the $b(u, G_k)$ of an arbitrary star of cliques in $O(n^3)$ time. However, in Algorithm 5, we present an $O(n \cdot \log n)$ algorithm to compute $b(u, G_k)$ for an arbitrary star of cliques.

We use base 10 digits and store them in a max-heap called $cliqueHeap$. In each iteration, we require the two largest cliques from the top of the heap. At each time unit j, where $1 \leq j \leq t$, we can address 2^{t-j} vertices. If any clique, besides the first, needs at least 2^{t-j} vertices in the current time unit, then the algorithm returns $False$. Otherwise, we inform 2^{t-j} vertices in the first clique. We replace the first element in the heap by $l_1 - 2^{t-j}$. We repeat this process t times, calling StarOfCliquesHeapBroadcast($cliqueHeap$, t, $j+1$) at the end of each iteration.

Algorithm 5 Heap-based decision algorithm for broadcasting in Star of Cliques from the universal vertex u in t time units

Input $cliqueHeap$: G_k as a max-heap of clique sizes, t: target broadcast time, j: current time unit $(1 \leq j \leq t)$.
Output $True$ if $b(u, G_k) = t$; $False$ otherwise.

1: **procedure** StarOfCliquesHeapBroadcast($cliqueHeap$, t, j)
2:     $l_1 \leftarrow cliqueHeap$.extractMax()
3:     $l_2 \leftarrow cliqueHeap$.peek()
4:     **if** $l_1 \geq 2^{t-j}$ **then**
5:         **if** $l_2 \geq 2^{t-j}$ **then**
6:             **return** $False$
7:         **else**
8:             $cliqueHeap$.insert($l_1 - 2^{t-j}$)
9:         **end if**
10:     **else**
11:         $cliqueHeap$.insert(0)
12:     **end if**
13:     **if** $j + 1 \leq t$ **then**:
14:         **return** StarOfCliquesHeapBroadcast($cliqueHeap$, t, $j+1$)
15:     **else**
16:         **return** $True$
17:     **end if**
18: **end procedure**

Complexity Analysis

Since the heap contains k elements, one iteration of the algorithm requires $O(\log k)$ time due to the heap operations in Steps 2, 8, and 11. We call the algorithm t times, which requires $O(t \cdot \log k)$ time for all iterations. In addition, $O(k \cdot \log k)$ time is required to sort the input and calculate the upper bound t', and $O(k)$ time is required to create the heap. The maximum possible value of t is $k + \lceil \log l_1 \rceil$ and $k, l_1 \leq n$, since $n = l_1 + l_2 + \ldots + l_k$ is the total number of vertices in the graph. Thus, $t, k = O(n)$, which brings the overall complexity of the algorithm to $O(n \cdot \log n)$.

Chapter 4

Block Graphs

In this chapter, we study the broadcasting problem in some families of Block Graphs. A block graph, also known as a clique tree, is an undirected graph in which every biconnected component is a clique. A biconnected component is a maximal biconnected subgraph. This structure ensures that block graphs are always chordal, meaning that every cycle of four or more vertices has a chord, a property that simplifies many computational problems. Block graphs play a significant role in various fields such as network design and biology due to their unique properties that combine simplicity and robustness.

The star of cliques graph discussed in the previous chapter can be categorized as a block graph. In this chapter, we discuss broadcasting in a few other families of block graphs such as Path-connected Cliques and Fully Connected Cliques. We present an $O(1)$ method to calculate the broadcast time for an arbitrary path-connected cliques graph, show a simplified version of Algorithm 5 that can be used for block graphs with two cliques, and discuss a $O(|V| \log \log n)$ method to determine the broadcast time for arbitrary fully connected cliques.

4.1 Broadcasting in Path-connected Cliques

In this section, we study the broadcasting problem in Path-connected cliques, a graph defined on n vertices and containing k cliques of arbitrary sizes $l_1, l_2, \ldots, l_k$, with neighboring cliques being connected by an edge. Path-connected cliques can be classified as block graphs since the edge connecting two neighboring cliques is also a clique K_2.

Consider a graph G_k consisting of a path P with vertices $u_1, u_2, \ldots, u_k$ (in the path order), and k pairwise vertex disjoint arbitrarily-sized cliques $C_1, C_2, \ldots, C_k$ which are attached to P with the following properties:

- $v_i \in V(C_i)$ for all $1 \leq i \leq k$ and
- For all $1 \leq i \leq k$, $1 \leq j \leq k$, $i = j$, an arbitrary vertex $v \in C_i$, and an arbitrary vertex $w \in C_j$, $(v, w) \notin E(G_k)$.

We may denote such a graph as $G_k = (P, C_1, C_2, \ldots, C_k)$. For all $1 \leq i \leq k - 1$ We define $t_i = \lceil \log l_i \rceil$ as the time units required for clique C_i to complete broadcasting without needing to inform C_{i+1}. We also define a_i to be the last possible time unit at which $u_i \in V(C_i)$ informs $u_{i+1} \in V(C_{i+1})$ such that the overall broadcasting is completed in minimum time. Figure 4.1 illustrates the general structure of path-connected cliques.

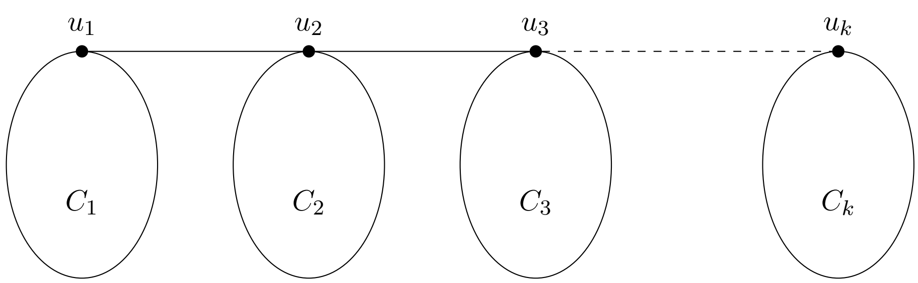

Figure 4.1: General Structure of the Path-Connected Cliques graph

Given a graph $G = (P, G_1, G_2, \ldots, G_k)$, where $G_1, G_2, \ldots, G_k$ can be any graphs and $P = \{u_1, u_2, \ldots, u_k\}$, Hovhannisyan (2024) presents a fixed-deadline message dissemination algorithm for general Path-connected graphs. However, in this work, we present an $O(1)$ method to calculate the broadcast time for an arbitrary Path-connected cliques graph.

4.1.1 Broadcasting in Path-connected Cliques when $k = 2$

Lemma 9. *Let $G_k = (P, C_1, C_2, \ldots, C_k)$ be a graph of path-connected cliques. If $k = 2$, then $b(u_1, G_2) = \max\{\lceil \log(l_1 + 2^{t_1 - a_1}) \rceil, a_1 + \lceil \log l_2 \rceil\}$.*

Proof. In any broadcast scheme for originator u_1 in G_2, u_1 must inform u_2 at some time unit a_1 to ensure broadcasting is completed in both cliques. In an optimal scheme, u_1 waits until the very last moment to inform u_2 in such a way that the broadcasting process in C_2 completes either simultaneously with or before the broadcasting process in C_1. Additionally, u_1 tries to minimize the number of vertices in C_1 that it doesn't inform because it's busy informing u_2. Once u_2 is informed, C_2 can finish broadcasting in $\lceil \log l_2 \rceil$ time units. Thus, all vertices in C_2 will be informed by time unit $a_1 + \lceil \log l_2 \rceil$.

Let $t_1 = \lceil \log l_1 \rceil$ be the time unit at which broadcasting in C_1 would finish if u_1 does not inform u_2 at time unit a_1. If $2^{t_1} - l_1 < 2^{t_1 - a_1}$, then C_1 may require an extra time unit to complete broadcasting; since vertex u_1 informs u_2 instead of a vertex in C_1 at time unit a_1, $2^{t_1 - a_1}$ vertices could remain uninformed at time unit t_1. In any case, all vertices in C_1 will be informed by time unit $\lceil \log(l_1 + 2^{t_1 - a_1}) \rceil$. Hence the broadcast time of G_2 from originator u_1 is given by $b(u_1, G_2) = \max\{\lceil \log(l_1 + 2^{t_1 - a_1}) \rceil, a_1 + \lceil \log l_2 \rceil\}$. A similar result can be shown for broadcasting from vertex u_2 in G_2. □

4.1.2 Broadcasting in Path-connected Cliques from Path Vertices

Vertices in the Path-connected cliques graph can be categorized as *path vertices* and *clique vertices*. Within path vertices, vertices u_1 and u_k are collectively referred to as *path-end vertices*, while all other path vertices are called *internal path vertices*.

Lemma 10. *Let $G_k = (P, C_1, C_2, \ldots, C_k)$ be a graph of path-connected cliques. Let $H_i = (P_i, C_i, C_{i+1}, \ldots, C_k)$ be a subgraph of G_k where P_i consists of vertices $u_i, u_{i+1}, \ldots, u_k$ for all $1 \leq i \leq k$. If vertex u_1 is the originator, then $b(u_i, H_i) = \max\{\lceil \log(l_i + 2^{t_i - a_i}) \rceil, a_i + b(u_{i+1}, H_{i+1})\}$ and $b(u_k, H_k) = \lceil \log l_k \rceil$.*

Proof. In any broadcast scheme for originator u_i in subgraph H_i, u_i must inform u_{i+1} at some time unit a_i to ensure broadcasting is completed in clique C_i as well as subgraph H_{i+1}. In an optimal scheme, u_i waits until the very last moment to inform u_{i+1} in such a way that the broadcasting process in H_i completes either simultaneously with or before the broadcasting process in C_i. Additionally, u_i tries to minimize the number of vertices in C_i that it doesn't inform because it's busy informing u_{i+1}.

Consider the subgraph H_{k-2}. We know that vertex u_{k-2} informs u_{k-1} at time unit a_{k-2}. From Lemma 9, we also know that $b(u_{k-1}, H_{k-1}) = \max\{\lceil \log(l_{k-1} + 2^{t_{k-1}-a_{k-1}})\rceil, a_{k-1} + \lceil \log l_k \rceil\}$. Once u_{k-1} is informed, broadcasting in H_{k-1} finishes in $b(u_{k-1}, H_{k-1})$ time units.

As seen in Lemma 9, all vertices in C_{k-2} will be informed by time unit $\lceil \log(l_{k-2}+2^{t_{k-2}-a_{k-2}})\rceil$. Hence the broadcast time of H_{k-2} from originator u_{k-2} is given by $b(u_{k-2}, H_{k-2}) = \max\{\lceil \log(l_i + 2^{t_i-a_i})\rceil, a_i + b(u_{i+1}, H_{i+1})\}$ and $b(u_k, H_k) = \lceil \log l_k \rceil$. In general, for all $1 \leq i \leq k$ the broadcast time of any subgraph H_i from originator u_i is given by $b(u_i, H_i) = \max\{\lceil \log(l_i + 2^{t_i-a_i})\rceil, a_i + b(u_{i+1}, H_{i+1})\}$ and $b(u_k, H_k) = \lceil \log l_k \rceil$. A similar result can be obtained when the originator is vertex u_k. $\square$

Observation 3. From Lemma 10 we know that $b(u_1, H_1) = \max\{\lceil \log(l_1+2^{t_1-a_1})\rceil, a_1+b(u_2, H_2)\}$. Also, it is evident that $b(u_1, G_k) = b(u_1, H_1)$. Hence $b(u_1, G_k) = \max\{\lceil \log(l_1 + 2^{t_1-a_1})\rceil, a_1 + b(u_2, H_2)\}$ where $b(u_k, H_k) = \lceil \log l_k \rceil$. Alternatively, $b(u_1, G_k)$ is also given by,

$$b(u_1, G_k) = \begin{cases} 1 + b(u_2, H_2) & \text{if } b(u_2, H_2) \geq \lceil \log l_1 \rceil \\ \lceil \log l_1 \rceil & \text{if } b(u_2, H_2) < \lceil \log l_1 \rceil \text{ and } 2^{t_1} - l_1 \geq 2^{b(u_2,H_2)} \\ \lceil \log l_1 \rceil + 1 & \text{if } b(u_2, H_2) < \lceil \log l_1 \rceil \text{ and } 2^{t_1} - l_1 < 2^{b(u_2,H_2)} \end{cases}$$

Lemma 11. *Let $G_k = (P, C_1, C_2, \ldots, C_k)$ be a graph of path-connected cliques. Then for all $2 \leq i \leq k-1$, $\max\{b(u_1, G_k), b(u_k, G_k)\} \geq b(u_i, G_k)$*

Proof. For an originator vertex u_i, let H_l be the subgraph of G_k with all cliques to the left of C_i. Then, $H_l = (P_l, C_{i-1}, C_{i-2}, \ldots, C_1)$, where P_l contains vertices $u_{i-1}, u_{i-2}, \ldots, u_1$. Similarly, let H_r be the subgraph with all cliques to the right of C_i. Then, $H_r = (P_r, C_{i+1}, C_{i+2}, \ldots, C_k)$, where P_r contains vertices $u_{i+1}, u_{i+2}, \ldots, u_k$. Without loss of generality, assume $b(u_{i-1}, H_l) \geq b(u_{i+1}, H_r)$. Consider the following broadcast scheme:

(1) Originator u_i informs vertex u_{i+1} at time 1.

(2) Vertices u_i and u_{i+1} broadcast separately within $G_k \setminus H_r$ and H_r respectively.

Since $b(u_{i-1}, H_l) \geq b(u_{i+1}, H_r)$, we have $b(u_i, G_k \setminus H_r) \geq b(u_{i+1}, H_r)$. Thus, $b(u_i, G_k) \leq 1 + b(u_i, G_k \setminus H_r)$

Consider the following broadcast scheme from originator u_k, which begins by informing path vertices $u_{k-1}, u_{k-2}, \ldots, u_i$ before informing any clique vertices. Once vertex u_i is informed it begins informing the subgraph $G_k \setminus H_r$. Then, $b(u_k, G_k) \geq k-i+b(u_i, G_k \setminus H_r)$ since $b(u_k, G_k) \geq b(u_k, G_k \setminus H_r) \geq$ minimum time required to inform u_i from $u_k + b(u_i, G_k \setminus H_r)$. By assumption, $k - i \geq 1$. Hence,

$$\begin{aligned} b(u_k, G_k) &\geq k - i + b(u_i, G_k \setminus H_r) \\ &\geq 1 + b(u_i, G_k \setminus H_r) \\ &\geq b(u_i, G_k) \end{aligned}$$

The same proof applies when $b(u_{i-1}, H_l) < b(u_{i+1}, H_r)$. $\square$

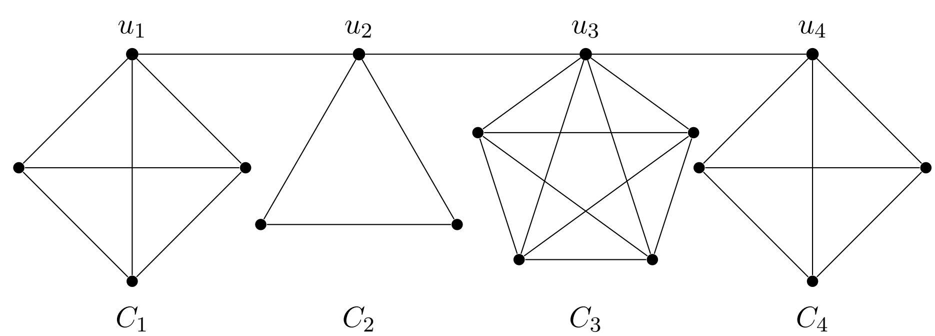

Figure 4.2: Path-connected cliques with $k = 4$ and 4 cliques of sizes 4, 3, 5, 4

According to Lemma 11, Path-end vertices have a larger broadcast time than any other path vertices for any path-connected cliques graph.

4.1.3 Broadcasting in Path-connected Cliques from Clique Vertices

All non-path vertices within some clique C_i will have the same broadcast time; In any optimal scheme, a non-path originator v_i will inform the nearest path vertex u_i at time unit 1 since u_i can contribute to broadcasting within clique C_i as well as the path. Since all non-path vertices within some clique C_i are all connected to the same path vertex, they will all have the same broadcast time. Thus, to determine the broadcast time from an arbitrary originator, we must first identify the worst originator among all clique vertices.

Lemma 12. *Let $G_k = (P, C_1, C_2, \ldots, C_k)$ be a graph of path-connected cliques. Then, for some vertex $v_1 \in C_1$, $b(v_1, G_k) = \max\{\lceil \log(l_1 + 2^{t_1 - a_1}) \rceil, 1 + a_1 + b(u_2, H_2)\}$ where $b(u_i, H_i) = \max\{\lceil \log(l_i + 2^{t_i - a_i}) \rceil, a_i + b(u_{i+1}, H_{i+1})\}$ and $b(u_k, H_k) = \lceil \log l_k \rceil$.*

Proof. Without loss of generality, assume when u_1 is the originator, v_1 is the first vertex to be informed in C_1. Then, if $a_1 \neq 1$, originator u_1 informs v_1 at time unit 1. In an optimal broadcast scheme, originator v_1 informs u_1 at time unit 1 since u_1 is a vertex in C_1 and a path vertex. For either originator, the only informed vertices in G_k after 1 time unit are v_1 and u_1. Hence, the remaining steps of broadcasting and the broadcast time are the same for both originators when $a_1 \neq 1$.

When $a_1 = 1$, by definition originator u_1 informs u_2 at time unit 1. However, this is not possible when v_1 is the originator as v_1 informs u_1 at time unit 1, delaying broadcasting in H_2 by 1 time unit. Overall, broadcasting in H_2 completes at time unit $2 + b(u_2, H_2)$. Since $a_1 = 1$, we know that $b(u_2, H_2) \geq b(u_1, C_1)$. Broadcasting in C_1 finishes before H_2 due to the added delay in H_2. Thus, $b(v_1, G_k) = 2 + b(u_2, H_2)$ when $a_1 = 1$.

In general, the broadcast time of G_k from originator v_1 is given by $b(v_1, G_k) = \max\{\lceil \log(l_1 + 2^{t_1 - a_1}) \rceil, 1 + a_1 + b(u_2, H_2)\}$ where $b(u_i, H_i) = \max\{\lceil \log(l_i + 2^{t_i - a_i}) \rceil, a_i + b(u_{i+1}, H_{i+1})\}$ and $b(u_k, H_k) = \lceil \log l_k \rceil$. Alternatively, $b(v_1, G_k)$ is also given by,

$$b(v_1, G_k) = \begin{cases} 2 + b(u_2, H_2) & \text{if } b(u_2, H_2) \geq \lceil \log l_1 \rceil \\ \lceil \log l_1 \rceil & \text{if } b(u_2, H_2) < \lceil \log l_1 \rceil \text{ and } 2^{t_1} - l_1 \geq 2^{b(u_2, H_2)} \\ \lceil \log l_1 \rceil + 1 & \text{if } b(u_2, H_2) < \lceil \log l_1 \rceil \text{ and } 2^{t_1} - l_1 < 2^{b(u_2, H_2)} \end{cases}$$

A similar result can be shown for broadcasting in G_k from vertex $v_k \in C_k$. □

Observation 4. From Lemma 12, we can see that $b(v_1, G_k) \geq b(u_1, G_k)$ by at most 1. Similarly, $b(v_k, G_k) \geq b(u_k, G_k)$. Thus, $b(G_k) = \max\{b(v_1, G_k), b(v_k, G_k)\}$.

4.2 Broadcasting in Block Graphs with two cliques

Consider block graphs G_k with two cliques C_1 and C_2 each containing l_1 and l_2 vertices respectively. We define a junction vertex to be the shared vertex between the two cliques. Every block graph G_2 with two cliques can be viewed as an equivalent star of cliques G'_2. C_1 and C_2 in the block graph can be mapped to C'_1 and C'_2 in the star of cliques such that $l'_1 = l_1 - 1$ and $l'_2 = l_2 - 1$. The junction vertex shared by C_1 and C_2 in the block graph is equivalent to the universal vertex u in the star of cliques.

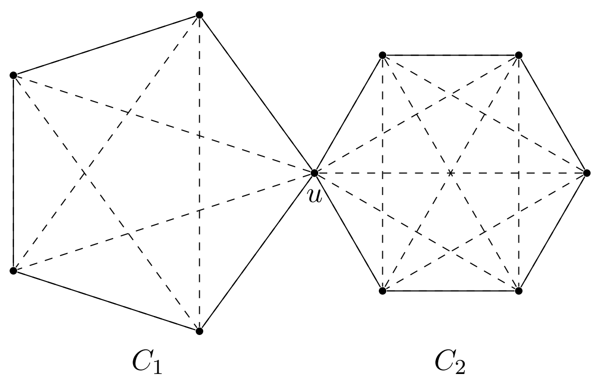

Figure 4.3: Block graph with 2 cliques of sizes 5 and 6

For a given block graph with $k = 2$, using Lemmata 2 and 3 we can determine the bounds on the broadcast time of the equivalent star of cliques from the shared junction vertex. Using the decision algorithm seen in Algorithm 5 we can identify the exact broadcast time. However, since we know $k = 2$, the algorithm can be simplified, as shown in Algorithm 6.

As seen in Algorithm 5 (Section 3.3.4), each iteration requires only the two largest cliques to determine which clique needs to be informed at time unit j for all $1 \leq j \leq t$. However, since G'_2 has exactly two cliques, Algorithm 6 does not require a heap to determine the scheme from the universal vertex u; as long as at least one of the two cliques has an unaddressed vertex (see Definition 1 in Section 3.3.1), according to Algorithm 6, vertex u will inform a new vertex in the clique with the most unaddressed vertices at time unit j. This newly informed vertex can inform 2^{t-j} (including itself) vertices by time unit t.

From Observation 2 (Section 3.3.3), we know that in any star of cliques with arbitrary originator v and universal vertex u, when $v \neq u$ and $v \in C'_i$, $b(v, G_k) = b(u, G_k \setminus C'_i) + 1$ if $l_1 > l_i$ (for all

Algorithm 6 Decision algorithm for broadcasting in Block Graphs from the universal vertex u in t time units when $k = 2$ (based on Algorithm 5)

Input l_1 and l_2: sizes of the two cliques C_1 and C_2 in G_k respectively, t: target broadcast time, j: current time unit $(1 \leq j \leq t)$.

Output $True$ if $b(u, G_k) = t$; $False$ otherwise.

```
l'_1 ← l_1 − 1
l'_2 ← l_2 − 1
while (l'_1 > 0 or l'_2 > 0) and (j ≤ t) do
    if l'_1 − 2^{t−j} ≥ l'_2 then
        l'_1 ← l'_1 − 2^{t−j}
    else
        l'_2 ← l'_2 − 2^{t−j}
    end if
    j += 1
end while
```

$1 < i \leq k$), and in all other cases, $b(v, G_k) = b(u, G_k)$. However, given that $k = 2$, we have,

$$b(v, G_k) = \begin{cases} b(u, G_k) & \text{if } v \in C'_1 \text{ or } (v \in C'_2 \text{ and } l'_1 = l'_2) \\ b(u, G_k \setminus C'_2) = b(u, C'_1) & \text{otherwise} \end{cases}$$

4.3 Broadcasting in Fully Connected Cliques

We define Fully Connected Cliques (FCC) as a graph with a central clique C_{center} containing s_{center} vertices and k external cliques $C_1, \ldots, C_k$, where each C_i contains s_i vertices for $1 \leq i \leq k$, and where $k \leq s_{center}$. An external clique C_i is connected to the central clique through vertex $u_i \in C_{center}$. The graph has a total of n vertices such that $n = \sum_{i=1}^{k} s_i + s_{center}$. The structural properties of Fully Connected Cliques make them suitable for modeling networks where high connectivity within subgroups is crucial, and sparse connectivity between groups is sufficient.

The name of this graph is derived from Fully Connected Trees (FCT), a graph previously studied by H. Harutyunyan and Maraachlian (2009) and Gholami et al. (2023) in which each vertex of the central clique is the root of a tree. Each Fully Connected Tree (FCT) can be uniquely transformed into a Fully Connected Clique (FCC) by including all possible edges between the vertices, thereby converting each tree into a clique. Conversely, an FCC can be converted into multiple distinct FCTs by deleting various combinations of edges.

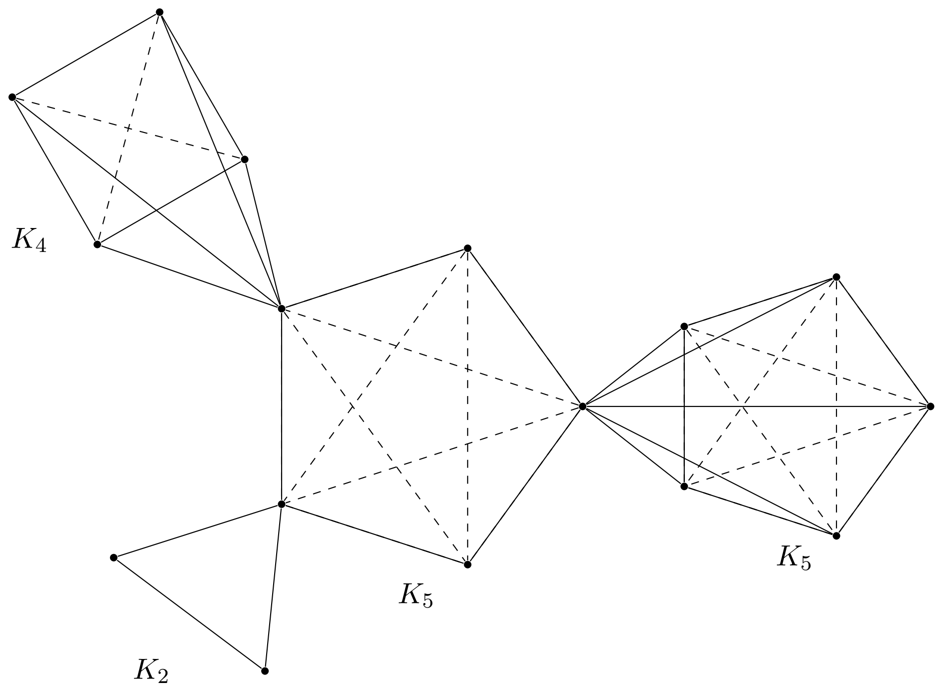

Figure 4.4: Fully Connected Cliques with central clique K_5 and external cliques K_2, K_4, K_5

In the FCT graph, vertices in the central clique are called *root vertices*, and those in the external cliques or trees are referred to as *tree vertices*. To ensure consistency, we adopt the same terminology for the FCC graph, with root vertices denoting vertices in the central clique and tree vertices for those in the external cliques.

Gholami et al. (2023) present a $O(n \log\log s_{center})$ time algorithm to calculate the broadcast time of any vertex in an arbitrary FCT. Given the structural similarity between the graphs, we intend to utilize this algorithm to determine the broadcast time of any FCC. We first present and discuss the Fully Connected Trees (FCT) algorithm in the following section, and then explore its application to determine the broadcast time of Fully Connected Cliques (FCC).

4.3.1 Broadcast Algorithm for Fully Connected Trees FCT

Vertices in FCT can either be *root vertices* or *tree vertices*. Let $V_{C_{center}}$ and $E_{C_{center}}$ be the vertex and edge sets of the central clique. For each tree T_i, the root vertex is denoted by i such that $1 \leq i \leq k$, while the vertex and edge sets are denoted by V_i and E_i. Let V_T and E_T be the sets of all tree vertices and tree edges in the graph such that $V_T = \bigcup_{i=1}^{k} V_i$ and $E_T = \bigcup_{i=1}^{k} E_i$. Then, the members of the set $V \cap V_{C_{center}}$ are known as root vertices while vertices in $V \setminus V_{C_{center}}$ are known as tree vertices. Each root vertex i has $d(i)$ children within the tree. These children labeled $i_1 \ldots i_{d(i)}$ are roots of subtrees $T_{i_1} \ldots T_{i_{d(i)}}$ such that $b(i_i, T_{i_1}) \geq \ldots \geq b(i_{d(i)}, T_{i_{d(i)}})$.

Broadcasting when the originator is a root vertex

In this section, we note that when discussing broadcasting in Fully Connected Cliques and Fully Connected Trees, τ denotes the target broadcast time, while t represents intermediate time units, maintaining consistency with the notation used in the FCT algorithm presented by Gholami et al. (2023).

When the originator is a root vertex, BR_τ is used in conjunction with B_{search} to determine the broadcast time. Given an FCT, a root vertex originator u, and a candidate broadcast time τ, BR_τ returns TRUE only if it is possible to complete broadcasting in FCT from originator u within τ time units. Otherwise, it returns FALSE. To confirm that the broadcast time is τ, BR_τ must return TRUE for τ and FALSE for $\tau - 1$. We can narrow the range of possible values for the broadcast time of the given FCT by establishing a lower bound lb and upper bound ub on the broadcast time.

$$\begin{aligned} lb &= \max\{\lceil \log s_{center} \rceil, \max_{1 \leq i \leq k}(b(i, T_i))\} \\ ub &= \lceil \log s_{center} \rceil + \max_{1 \leq i \leq k}(b(i, T_i)) \end{aligned} \tag{1}$$

Instead of searching this range of values in ascending order to find the first value of τ for which BR_τ returns TRUE, the algorithm B_{search}, shown in Algorithm 7, applies a modified version of binary search to determine the broadcast time of the given FCT.

Algorithm 8 shows BR_τ which is the main broadcast algorithm that determines if broadcasting in the given FCT can be completed within τ time units, where τ is the candidate broadcast time. Let t be the current time unit such that $0 \leq t \leq \tau$. BR_τ begins by assigning weights $w(i, t)$ to every root vertex, and calculating m_{i_j}s. These weights were developed by Slater et al. (1981) for broadcasting in trees. $w(i, t)$ is equal to the time needed to complete broadcasting in the subtrees of root vertex i. If vertex i does not have any uninformed children in T_i, then its weight is 0. For a tree T_i, m_{i_j} denotes the time needed to finish the broadcasting in subtree T_{i_j} originating at tree vertex

Algorithm 7 The modified Binary Search algorithm $B_{Search}(FCT, u, lb, ub)$

Input: $FCT = (V, E)$, originator u, lower bound lb, and upper bound ub.
Output: Broadcast time τ such that $\tau = b(u, FCT)$

$t = lb + \lfloor \frac{ub-lb}{2} \rfloor$
if $lb == ub$ **then**
    if $BR_\tau(FCT, u, lb)$ **then**
        return lb
    end if
end if
if $lb + 1 == ub$ **then**
    if $BR_\tau(FCT, u, lb)$ **then**
        return lb
    end if
    if $!BR_\tau(FCT, u, lb) \quad \& \quad BR_\tau(FCT, u, ub)$ **then**
        return ub
    end if
end if
if $BR_\tau(FCT, u, t)$ **then**
    return $B_{Search}(FCT, u, lb, t)$
else
    return $B_{Search}(FCT, u, t, ub)$
end if

i_j, such that $m_{i_j} = b(i_j, T_{i_j})$ for $1 \leq j \leq d(i)$. Since BR_τ makes use of the algorithm presented by Slater et al. (1981), the children of root vertex i labeled $i_1 \ldots i_{d(i)}$ will be arranged such that $m_{i_1} \geq \ldots \geq m_{i_{d(i)}}$. When $t \geq 1$, $w(i,t)$ can be calculated using the m_{i_j} weights of $i_1 \ldots i_{d(i)}$. Thus, $w(i,t) = \max\limits_{1 \leq j \leq d(i)} \{j + m_{i_j}\}$.

For each root vertex i, $l_i = \tau - t - w(i,t) - 1$ is the number of time units remaining by which vertex i must be informed if broadcasting is to be completed in T_i within τ time units. Since vertex u is the originator, l_u is set to NULL. Let V_I be the set of informed vertices and V_U be the set of uninformed vertices. Then $\forall i \in V_I : l_i =$ NULL. For broadcasting to uninformed root vertices, those with smaller l_i values are prioritized and must be informed before other uninformed root vertices. When several root vertices have the same value of l_i, the algorithm chooses a vertex randomly to proceed. If a root vertex has l_i below 0, then i cannot complete broadcasting in T_i within τ time units.

At each time unit, BR_τ considers every vertex in the graph. Particularly, for an uninformed root vertex i, l_i is updated every time unit. In addition, the algorithm considers the best action to be performed by informed vertices. The optimal decision for an informed tree vertex is to follow the broadcast algorithm given by Slater et al. (1981). However, for an informed root vertex i, the best action may involve either contributing to broadcasting in the central clique or its tree T_i. If the algorithm detects some root vertex that cannot inform all vertices in its tree by in the remaining time ($w(y,t) > \tau - t$), then it immediately returns FALSE. Of course, if $w(i,t) = 0$, then root vertex i has no uninformed children and will inform a vertex in the central clique. If $w(i,t) > 0$, then the algorithm makes a decision based on the remaining time units. If $w(i,t) < \tau - t$ then there is more than enough time to inform vertices in T_i and therefore root vertex i can inform another root vertex. However, if $w(i,t) = \tau - t$, then vertex i informs the vertex in its tree which has the highest value of m_{i_j}. Essentially, tree vertices are informed at the latest time unit possible.

Broadcasting when the originator is a tree vertex

When the originator is a tree vertex v, the broadcast scheme from Algorithm 9 is used. For any originator v which is a tree vertex, there is a unique path P connecting this tree vertex to its root vertex i. Let i_j be the direct child of i that is on path P. Then, v is in subtree T_{i_j}. Let T_i' be a tree rooted at i such that T_i' includes all subtrees of T_i except those rooted at i_j. In other words, $T_i' = T_i \setminus T_{i_j}$. Now, construct FCT' by replacing T_i in FCT with T_i'. Then, broadcasting in FCT' from root vertex i can be completed using Algorithms 7 and 8. Let T' be the broadcast tree generated by broadcasting in FCT'. We can now construct a tree $T = T' \cup T_{i_j}$ using the edge (i, i_j). Finally, broadcasting in tree T can be performed from originator v using the broadcast algorithm for trees provided by Slater et al. (1981), and the resulting broadcast time is the broadcast time for tree vertex v in FCT.

For further details and the proof of correctness of the FCT algorithms discussed so far, we refer readers to Gholami et al. (2023).

4.3.2 Broadcasting from root vertices in Fully Connected Cliques FCC

Utilizing the FCT algorithm seen in the previous section, our objective is to develop a method for determining the broadcast time of Fully Connected Cliques. We first examine the application of the FCT algorithm to determine the broadcast time of a given FCC where the originator is a root

Algorithm 8 The broadcast algorithm $BR_\tau(FCT, u, \tau)$

Input: $FCT = (V, E)$, originator u, candidate broadcast time τ

Output: FALSE if τ cannot be the broadcast time, TRUE if broadcasting can be accomplished in at most τ time units.

Initialize: the labels $w(i, t)$ and m_{i_j} for all root vertices
Initialize: $V_I = \{u\}$, $V_U = V \setminus V_I$, $l_u = NULL$
for each t such that $0 \le t \le \tau - 1$ **do**
    for each $v \in V_U$ **do**
        if v is a root vertex then **then**
            update l_v as follows: $l_v = \tau - t - w(v, t) - 1$
        end if
    end for
    for each $v \in V_I$ **do**
        if v is a root vertex then **then**
            if $w(v, t) < \tau - t$ then **then**
                if there exists at least one uninformed root vertex then **then**
                    v informs vertex j at time t such that j has the smallest value of l_a in V_U
                    $l_j = NULL$, $V_I = V_I \cup \{j\}$, $V_U = V_U \setminus \{j\}$
                else
                    v stays idle
                end if
            else
                if $w(v, t) = \tau - t$ then **then**
                    v informs one of its children which has the highest value of m_v in the tree rooted at T_v, $1 \le j \le d(v)$
                    $m_{v_j} = NULL$, $V_I = V_I \cup \{v_j\}$, $V_U = V_U \setminus \{j\}$
                    update $w(v, t) = \max_{1 \le k \le d(v)} \{k + m_{v_k}\}$
                else
                    return FALSE
                end if
            end if
        else
            v informs a tree vertex v_T in the uninformed sub-tree rooted at v based on the well-known broadcasting algorithm in trees
            $V_I = V_I \cup \{v_T\}$, $V_U = V_U \setminus \{v_T\}$
        end if
    end for
end for
return TRUE

Algorithm 9 The broadcast algorithm $BR_\tau(FCT, v)$

Input: $FCT = (V, E)$, originator v
Output: $b(v, FCT)$

$P =$ The path connecting v to a root vertex in FCT
$i =$ The root vertex, i_j = The neighbor of i on P
Construct $FCT' = (V', E')$ as follows $V' = V \setminus V(T_{i_j})$ and $E' = E \setminus E(T_{i_j}) \setminus \{(i, i_j)\}$
Calculate lb and ub based on Equation 1 for FCT'
Solve $B_{Search}(FCT', i, lb, ub)$
$T' =$ Broadcast tree obtained by the previous step
Construct $T = (V^T, E^T)$ as follows: $V^T = V(T') \cup V(T_{i_j})$ and $E^T = E(T') \cup E(T_{i_j}) \cup \{(i, i_j)\}$
Solve the broadcast problem for T based on the well-known broadcast algorithm for trees
return $b(v, T)$

vertex and later analyze the scenario with arbitrary originators.

Definition 2. Complete Binomial FCT: *A Binomial Fully Connected Tree (Binomial FCT) is a specific type of Fully Connected Tree (FCT) characterized by the following properties: (1) For each root vertex i which is the root of the tree T_i, every subtree $T_{i_1} \ldots T_{i_{d(i)}}$ must be a binomial tree, assuming the subtrees are labeled in decreasing order of their broadcast times from their roots. (2) The binomial subtrees $T_{i_1} \ldots T_{i_{d(i)}}$ must have dimensions corresponding to the binary representation of $|V(T_i)|$. Specifically, for $|V(T_i)| = 2^{p_1} + 2^{p_2} + \ldots + 2^{p_c}$, where $p_1, \ldots, p_c$ are non-negative integers, T_i must have exactly $d(i) = c$ binomial subtrees $T_{i_1} \ldots T_{i_{d(i)}}$ such that $T_{i_1} = B_{p_1}, T_{i_2} = B_{p_2}, \ldots, T_{i_{d(i)}} = B_{p_c}$.*

Any FCC can be transformed into its corresponding distinct complete binomial FCT. Figure 4.6 shows an example of a complete binomial FCT derived from an FCC containing external cliques of sizes 41, 10, 3, 2, 1.

In the following lemmata, we show that the broadcast time of a given FCC from a root vertex is equivalent to the broadcast time of its corresponding binomial FCT. Thus, the problem of determining the broadcast time of an FCC reduces to determining the broadcast time of its corresponding binomial FCT.

Lemma 13. *Let H be a spanning subgraph of G. For a common originator vertex v in both graphs, $b(v, G) \leq b(v, H)$.*

Proof. Let $G = (V, E)$ be a graph and $H = (V, E_H)$ be a spanning subgraph of G, where $E_H \subseteq E$. By definition, H contains all vertices of G but potentially fewer edges. Consider a common originator vertex $v \in V$. Any optimal broadcast scheme from v in H will also be a broadcast scheme from v in G, but not necessarily optimal; there may be a scheme from v in G with lower broadcast time that relies on edges not present in H. Thus, it follows that $b(v, G) \leq b(v, H)$ for any common originator v. □

Lemma 14. *Let G_k be a FCC containing k external cliques and H_k be the corresponding complete binomial FCT. Then $b(u_i, G_k) \leq b(u_i, H_k)$, where u_i is a root vertex for all $1 \leq i \leq k$.*

Proof. Given that H_k is a spanning subgraph of G_k (since a complete binomial FCT is a spanning subgraph of the FCC from which it was derived), by the same reasoning as Lemma 13, any optimal broadcast scheme from u_i in H_k will also be a broadcast scheme from u_i in G_k, but not necessarily optimal. Hence, $b(u_i, G_k) \leq b(u_i, H_k)$. □

Lemma 15. *Let G_k be an FCC containing k external cliques and H_k be the corresponding complete binomial FCT. Then $b(u_i, G_k) \geq b(u_i, H_k)$, where u_i is a root vertex for all $1 \leq i \leq k$.*

Proof. Let S_{opt} be an optimal scheme for broadcasting in FCC from an arbitrary root vertex u_i. To show $b(u_i, G_k) \geq b(u_i, H_k)$, the calls in S_{opt} can be modified to incorporate those from the FCT algorithm for broadcasting to an external clique from the corresponding root vertex while ensuring the broadcast time of this modified scheme remains no greater than that of S_{opt}.

The FCT algorithm informs tree vertices at the latest possible time unit. Hence, it must be the case that S_{opt} informs tree vertices in FCC no later than the corresponding tree vertex in the FCT algorithm. If they are informed at the same time, then calls from the FCT algorithm can be used instead (which in turn uses the optimal algorithm for broadcasting in trees shown by Slater et al. (1981)).

However, if a tree vertex is informed earlier in S_{opt} than in the FCT algorithm, the root vertex may never broadcast to this clique again, utilizing its available time units to broadcast to other root vertices instead. To incorporate the FCT algorithm's calls to broadcast to external cliques, tree vertices must be informed at the same time units as in the FCT algorithm. The early call in S_{opt}

can instead be used to inform another root vertex, which then has additional time to inform the other root vertices that were originally meant to be informed by u_i, ensuring that this happens no later than in S_{opt}. Hence $b(u_i, G_k) \geq b(u_i, H_k)$.

Figure 4.5 illustrates how calls in S_{opt} can be modified to incorporate tree vertex calls from the FCT algorithm through an example. Note that the time units $t_1 < t_2 < \ldots < t_7$ are labeled in increasing order. The arrows on the edges indicate the direction of calls made, even though the graph is undirected. Figure 4.5 (a) illustrates the calls made by vertex u_i to inform root vertices u_x, u_y, u_z and tree vertices v_1, v_2, v_3, v_4 according to the FCT algorithm. The tree vertices are called at time units t_1, t_2, t_4, t_6 and the root vertices are called at time units t_3, t_5, t_7. The tree vertices will broadcast to the remaining vertices within the clique. In Figure 4.5 (b), calls are made according to an optimal scheme S_{opt} for FCC. Tree vertices v_1 and v_2 are called at time units t_1 and $t_2 - 1$, becoming roots of trees, which may be subtrees of the binomial trees B_{t-t_1} and $B_{t-(t_2-1)}$, respectively. Root vertices u_x, u_y, u_z can now be informed in advance, at time units t_2, t_3, t_4. In Figure 4.5 (c), we show how S_{opt} can be modified to incorporate the calls from the FCT algorithm to inform tree vertices while making certain that the other root vertices are informed no later than in the original S_{opt} scheme. The tree vertices v_1, v_2, v_3, v_4 are called at time units t_1, t_2, t_4, t_6 according to the FCT algorithm. Vertex u_i makes the early call to root vertex u_x at time unit $t_2 - 1$. Vertex u_x uses this extra time to broadcast to the other root vertices u_y and u_z at time units t_2 and $t_2 + 1$ respectively. $\square$

Based on Lemmata 14 and 15, it is clear that the broadcast time of a given FCC is equivalent to the broadcast time of its corresponding binomial FCT when the originator is a root vertex. Algorithm 10 outlines the procedure for determining the broadcast time of a given FCC and root vertex u. This procedure involves first converting the FCC into its corresponding binomial FCT, and then using B_{search} and BR_τ to calculate the broadcast time of the FCT.

Consider the fully connected cliques and root vertex originator u shown in Figure 4.6, with a central clique K_6 and 5 external cliques K_{41}, K_{10}, K_3, K_2, and K_1. According to Algorithm 10, the first step to determine the broadcast time is to convert the FCC into its corresponding binomial FCT. For instance, K_{41} in the FCC will be replaced by 3 binomial trees B_5, B_3 and B_0 in the FCT, as shown in Figure 4.6. Then, make a call to B_{search} and BR_τ to obtain τ such that $b(u, FCT) = b(u, FCC) = \tau$.

Let us now analyze the operations of $BR_\tau(FCT, u, \tau)$ for candidate broadcast time $\tau = 6$. Figure 4.6 shows the series of calls made from time units 1 to 6. At time unit 1, since $w(1, 0) = 6 = \tau - t$, vertex u must inform a neighboring tree vertex. According to the broadcasting algorithm for trees, this will be the root of subtree T_{1_1}. Thus, m_{1_1} will be set to NULL. We also update $w(1, 1)$. Finally, l_i is updated for all root vertices.

At time unit 2, the informed tree vertex will begin broadcasting to uninformed vertices in T_{1_1}. Meanwhile, vertex u has to make a choice between informing a root vertex and a tree vertex. Since $w(1, 1) < \tau - t$, it does not need to continue broadcasting within its tree for now. Thus, it informs v_2, the root vertex with the smallest value of l_i. Then, l_i is updated for all cliques, l_2 is set to NULL.

At time unit 3, all informed tree vertices will continue broadcasting within their subtrees. However, the informed root vertices u and v_2 need to decide between making a call to a root vertex and a tree vertex. Since both root vertices have neighboring tree vertices with $m_{i_j} = \tau - t$, they broadcast within their respective trees. This continues for remaining time units as shown in Figure 4.6 until all vertices of the graph are informed. Thus, we can conclude that $b(u, FCC) \leq 6$. However, to confirm that $b(u, FCC) = 6$, one must call $BR_\tau(FCT, u, \tau)$ with candidate broadcast time $\tau = 5$ and receive FALSE as the output.

We note here that according to the algorithm for broadcasting in trees presented by Slater et al. (1981), once a root vertex i is informed, the optimal action for that vertex is to complete broadcasting in its tree T_i. However, in the case of fully connected cliques, this can sometimes be sub-optimal. For instance, in the example above, if vertex u informed a tree vertex instead of root vertex v_2 at time unit 2, it would be impossible to complete broadcasting within the next 4 time units. The optimal scheme for vertex u would be to inform a tree vertex in the first time unit and then inform root vertices, only informing tree vertices when necessary.

Algorithm 10 Broadcast Algorithm for Fully Connected Cliques and root vertex originator

Input: $FCC = (V, E)$, root vertex originator u

Output: $b(u, FCC)$

1: **procedure** FCCBROADCAST(FCC, u)
2: Convert FCC to the corresponding binomial FCT
3: Determine the broadcast time of the FCT from root vertex originator u using B_{search} and BR_{τ}
4: **end procedure**

Complexity Analysis

Determining the broadcast time of a given FCC when the originator is a root vertex has two phases as outlined in Algorithm 10: (1) Building FCT from FCC (2) Running the FCT Algorithm.

- **Building FCT from FCC**: When constructing an FCT from a given FCC, the objective is to retain the central clique C_{center} while replacing each external clique C_i with a tree T_i that includes all s_i vertices from C_i for $1 \leq i \leq k$.

  Let $p \leq \lceil \log s_i \rceil$ be the number of 1s in $bin(s_i)$, the binary representation of s_i. The external tree T_i will then consist of p binomial trees as subtrees, where the roots of these binomial subtrees, $v_1, \ldots, v_p$, are connected to the root vertex $u_i \in C_{center}$. The dimensions of these binomial trees correspond to the positions of the p 1s in $bin(s_i)$. If $r_1, \ldots, r_p$ are the positions of the 1s in the binary string, then the corresponding binomial trees are $B_{r_1}, \ldots, B_{r_p}$.

  Constructing a binomial tree B_{r_j} takes $O(2^{r_j})$ time, where $1 \leq j \leq p$. Connecting the roots of the binomial trees $v_1 \ldots v_p$ to the root vertex u_i requires $O(p)$ time. Altogether, constructing each binomial tree within one external tree T_i requires $\sum_{j=1}^{p} O(2^{r_j}) + O(p) = O(\sum_{j=1}^{p} 2^{r_j}) + O(p) = O(s_i) + O(\lceil \log s_i \rceil) = O(s_i)$ time.

  A total of k trees are required to replace the external cliques. The overall complexity of building these external trees is $\sum_{i=1}^{k} O(s_i) = O(\sum_{i=1}^{k} s_i) = O(n)$

- B_{search} **and** BR_{τ}: The call to $BR_{\tau}(FCT, u)$, where u is a root vertex originator, takes $O(n \log \log s_{center})$ time. The range of candidate broadcast time values searched is $ub - lb \in O(\log s_{center})$. Since we use B_{search}, we have $O(\log(ub - lb)) = O(\log \log s_{center})$.

Each iteration of B_{search} calls BR_{τ}, bringing the final complexity to $O(n \log\log s_{center}) = O(n \log\log n)$ in the worst case.

4.3.3 Broadcasting from tree vertices in Fully Connected Cliques FCC

In an FCC, the originator can be either a root vertex or a tree vertex. As seen previously, the FCT algorithm can be utilized to determine the broadcast time of an FCC when the originator is a root vertex. In this section, we will focus on analyzing the broadcasting process in an FCC when the originator is a tree vertex.

Lemma 16. *Let G_k be an FCC containing k external cliques and an arbitrary tree vertex v_i be the originator such that $v_i \in C_i$ and $v_i \neq u_i$ for $1 \leq i \leq k$ where $u_1 \dots u_k$ are root vertices. Then, there exists an optimal scheme in which originator v_i informs root vertex u_i at time unit 1.*

Proof. Let S_{opt} represent an optimal broadcasting scheme for an FCC when the originator is a tree vertex $v_i \in C_i$. At time unit 1, vertex v_i has the option of either broadcasting to another vertex $v_x \in C_i$ or to the nearest root vertex u_i. Suppose in S_{opt}, v_i chooses to inform v_x at time unit 1. Alternatively, v_i could instead make a call to u_i, which is connected not only to all vertices within C_i but also to other root vertices, whereas v_x is only connected to vertices within C_i. Therefore, if v_i informs u_i at time unit 1, it does not negatively impact the overall broadcast time ☐

From Lemma 16, it is established that there exists an optimal broadcasting scheme for FCC in which a tree vertex originator v_i informs the nearest root vertex u_i at time unit 1. Let t denote the optimal broadcast time of an FCC. Within the remaining $t-1$ time units, v_i can inform at most 2^{t-1} vertices within C_i. Let T_i represent the broadcast tree formed during these $t-1$ time units, rooted at vertex v_i. We then define C'_i as a sub-clique of C_i which is induced by all vertices of C_i that are not in T_i and G'_k as a sub-graph of G_k where C_i is replaced by C'_i (see Figure 4.7).

Formally, the vertex and edge sets of C'_i are as follows: $V(C'_i) = V(C_i) \setminus V(T_i)$ and $E(C'_i) = E(C_i) \setminus E(T_i)$. The vertex and edge sets of G'_k are given by: $V(G'_k) = (V(G_k) \setminus V(C_i)) \cup V(C'_i)$ and $E(G'_k) = (E(G_k) \setminus \{(u, v) \in E(G_k) | u \notin V(C_i) \text{ and } v \notin V(C_i)\}) \cup E(C'_i)$.

Lemma 17. *Let G_k be an FCC containing k external cliques and an arbitrary tree vertex v_i be the originator such that $v_i \in C_i$ and $v_i \neq u_i$ for $1 \leq i \leq k$ where $u_1 \ldots u_k$ are root vertices. Then, $b(v_i, G_k) = b(u_i, G'_k) + 1$.*

Proof. At time unit 1, tree vertex originator v_i informs root vertex u_i. Starting from time unit 2, u_i can broadcast within G'_k using the FCT algorithm, which, as previously established, is an optimal scheme for broadcasting in an FCC when the originator is a root vertex. Therefore, the broadcast time $b(v_i, G_k)$ equals $b(u_i, G'_k) + 1$ and $b(u_i, G'_k)$ can be determined using the FCT algorithm.

$\square$

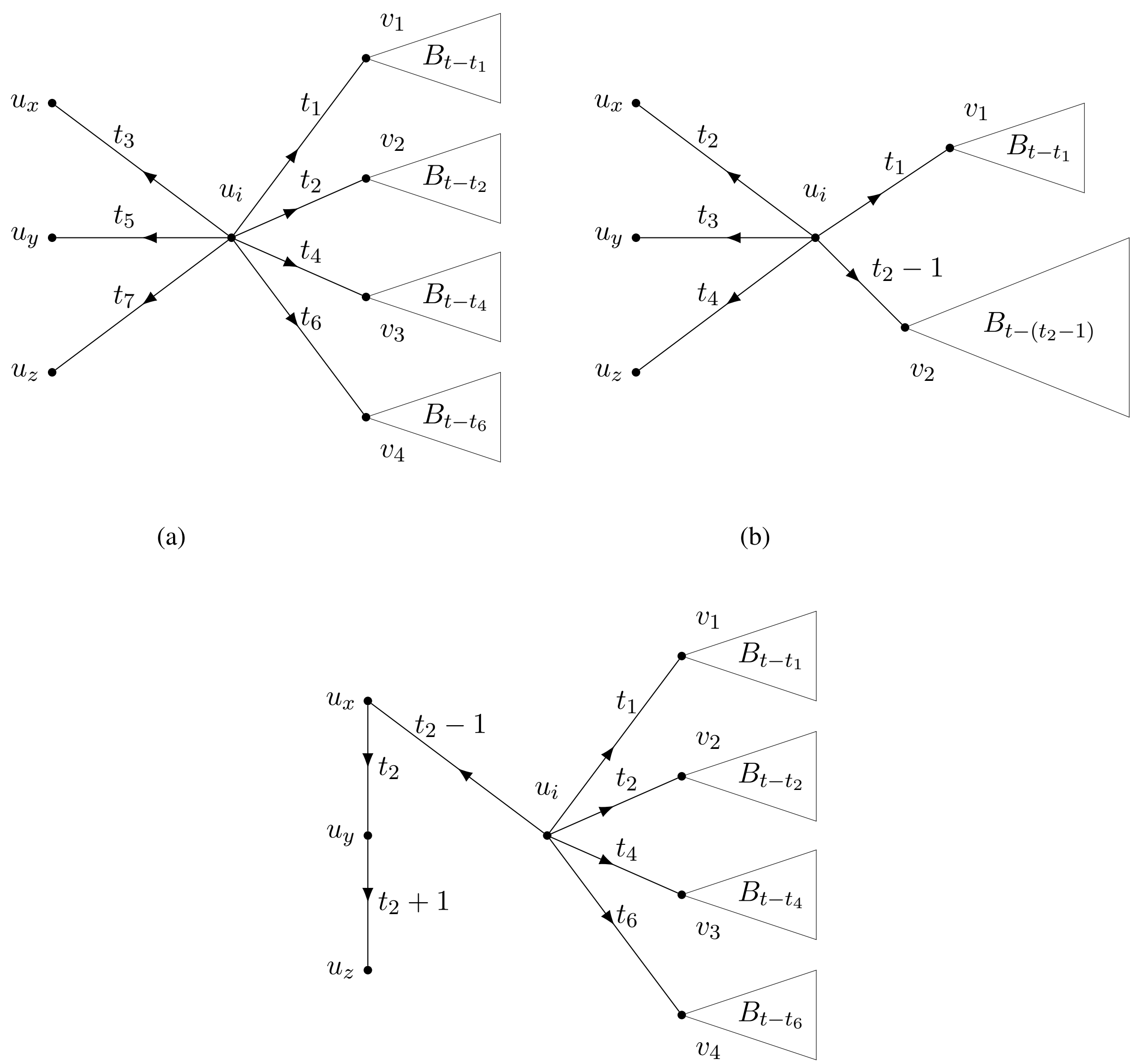

Figure 4.5: Broadcasting in an FCC with broadcast time t, under the following schemes: (a) the FCT algorithm (b) S_{opt}: an optimal scheme for broadcasting in FCC (c) the modified S_{opt} which uses calls from the FCT algorithm to broadcast to external cliques

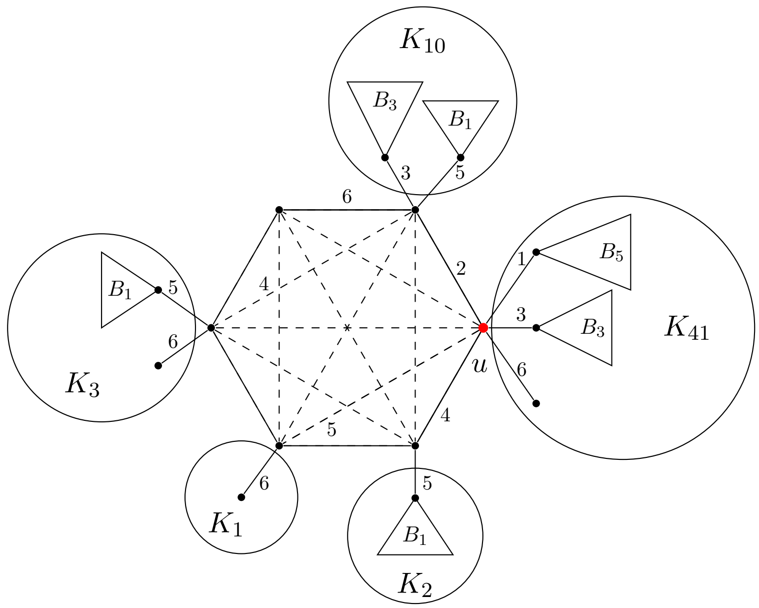

Figure 4.6: Complete Binomial FCT derived from an FCC with external cliques K_{41}, K_{10}, K_3, K_2, K_1

and originator vertex u

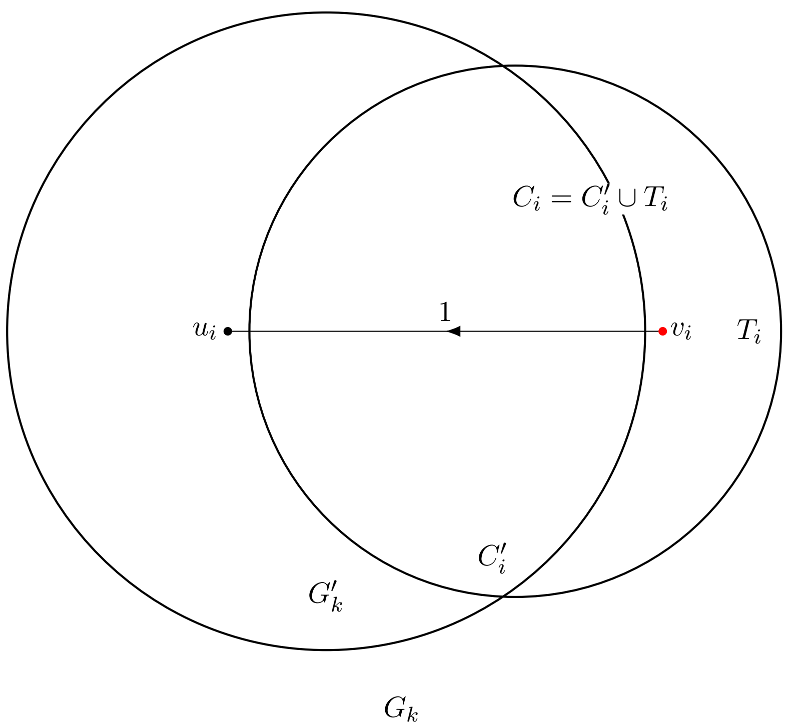

Figure 4.7: Broadcasting in a Fully Connected Cliques graph G_k from a tree vertex originator v_i

Chapter 5

Conclusion

In this thesis, we studied the problem of broadcasting in interconnection networks. Broadcasting is a message dissemination problem in a connected network where one informed node, called the originator, must distribute a message to all other nodes by placing a series of calls along the communication lines of the network. The topic of broadcast time has been a subject of research for over four decades. However, many unresolved problems and questions persist, indicating the ongoing need for further investigation in this area. Efficient broadcasting is critical in network design, impacting both theoretical studies and practical applications. The problems addressed in this thesis revolve around various graph topologies containing intersecting cliques, exploring the broadcast problem on specific graph structures such as windmill graphs, star of cliques, and fully connected cliques.

The focus of Chapter 3 was on windmill graphs and stars of cliques. Windmill graphs are defined on n vertices and contain k cliques of size l along with a universal vertex which is connected to all other vertices in the graph. A constant time algorithm for determining the broadcast time of windmill graphs was introduced, providing a foundation for efficient communication in this topology. Additionally, a new topology, the Star of Cliques, was proposed, which is a generalization of the windmill graph containing arbitrarily sized cliques. This structure is ideal for modeling networks characterized by clustered subgroups that share a central node. We first identified the bounds on the broadcast time of the star of cliques and then reduced the problem of finding the exact broadcast time to a decision problem. The decision problem required determining if broadcasting could be completed in $t = t' - 1$ time units, where t' is the upper bound on the broadcast time and t is the target broadcast time.

To solve this, an algorithm with a complexity of $O(n^3)$ was initially presented, leveraging binary representations to identify an individual broadcast scheme for each clique and integrating them to obtain an optimal scheme for the graph as a whole. Further optimization using heaps improved the complexity to $O(n \log n)$. We also prove the optimality of our algorithm and show how it can be used to determine the broadcast time from an arbitrary originator.

In Chapter 4, we study broadcasting in various families of block graphs. A block graph is an undirected graph where each block (or biconnected component) is a clique, and any two blocks share at most one common vertex. Due to their unique properties that blend simplicity and robustness, block graphs are significant in fields such as network design and biology.

Hovhannisyan (2024) presented a fixed-deadline broadcast algorithm for general path-connected graphs. Let G_k be a path-connected cliques graph consisting of a path P with vertices $u_1, u_2, \ldots, u_k$ (in the path order), and k pairwise vertex disjoint arbitrarily-sized cliques $C_1, C_2, \ldots, C_k$ which are attached to P. In this work, we present a constant time algorithm to determine the broadcast time specifically for path-connected cliques from an arbitrary originator. Further, we discuss broadcasting in general block graphs of two cliques by reducing it to broadcasting in a simplified star of cliques.

We also explore broadcasting in fully connected cliques, which consists of a central clique with s_{center} vertices connected to up to k external cliques of varying sizes, each linked to the central clique through a distinct vertex. These structural characteristics make the Fully Connected Cliques graph suitable for modeling networks that require high intra-group connectivity while sparse inter-group connectivity suffices. Gholami et al. (2023) presented an optimal algorithm for broadcasting fully connected trees. In this work, we use this algorithm to develop optimal broadcast schemes for fully connected cliques.

Further research studying the broadcast time of all structures studied in this thesis with different subgraphs in place of cliques is an open research direction. These structures can also be studied under other models of broadcasting, some of which were discussed in Section 2.3. In addition, many other families of block graphs remain open problems under broadcasting, and we believe that the work in this thesis will contribute towards further studies in this area.

During this master's program, portions of this thesis were presented at the 35th International Workshop on Combinatorial Algorithms (IWOCA 2024) (Ambashankar & Harutyunyan, 2024).

References

Ahlswede, R., Gargano, L., Haroutunian, H., & Khachatrian, L. H. (1996). Fault-tolerant minimum broadcast networks. *Networks: An International Journal*, *27*(4), 293–307.

Ahlswede, R., Haroutunian, H., & Khachatrian, L. H. (1994). *Messy broadcasting in networks*. Springer.

Ambashankar, A., & Harutyunyan, H. A. (2024). Broadcasting in stars of cliques. In *International workshop on combinatorial algorithms* (pp. 485–496).

Averbuch, A., Shabtai, R. H., & Roditty, Y. (2014). Efficient construction of broadcast graphs. *Discrete Applied Mathematics*, *171*, 9–14.

Beier, R., & Sibeyn, J. F. (2000). A powerful heuristic for telephone gossiping. In *Proceedings of the 7th International Colloquium on Structural Information and Communication Complexity (SIROCCO), Laquila, Italy, June 20-22, 2000* (pp. 17–35). Carleton Scientific.

Bermond, J.-C. (1979). Graceful graphs, radio antennae and french windmills. In *Proceedings One day Combinatorics Conference, Research Notes in Mathematics, vol34, Pitman* (pp. 18–37).

Bermond, J.-C., Fraigniaud, P., & Peters, J. G. (1995). Antepenultimate broadcasting. *Networks*, *26*(3), 125–137.

Bermond, J.-C., Hell, P., Liestman, A. L., & Peters, J. G. (1992). Sparse broadcast graphs. *Discrete Applied Mathematics*, *36*(2), 97–130.

Bermond, J.-C., & Peyrat, C. (1988). Broascasting in de bruijn networks. *Congressus numerantium*, *66*, 283–292.

Bhabak, P., Harutyunyan, H., & Kropf, P. (2017). Efficient broadcasting algorithm in harary-like networks. In *2017 46th International Conference on Parallel Processing Workshops (ICPPW)* (pp. 162–170).

Bhabak, P., & Harutyunyan, H. A. (2015). Constant approximation for broadcasting in k-cycle graph. In *Conference on Algorithms and Discrete Applied Mathematics* (pp. 21–32).

Bhabak, P., & Harutyunyan, H. A. (2019). Approximation algorithm for the broadcast time in k-path graph. *Journal of Interconnection Networks*, *19*(04), 1950006.

Bhabak, P., & Harutyunyan, H. A. (2022). Approximation algorithms in graphs with known broadcast time of the base graph. In *Conference on Algorithms and Discrete Applied Mathematics* (pp. 305–316).

Bhabak, P., Harutyunyan, H. A., & Tanna, S. (2014). Broadcasting in harary-like graphs. In *2014 IEEE 17th International Conference on Computational Science and Engineering* (pp. 1269–1276).

Chau, S.-C., & Liestman, A. L. (1985). Constructing minimal broadcast networks. *J. Combin. Inform. Syst. Sci*, *10*, 110–122.

Chen, X. (1990). An upper bound for the broadcast function b (n), chinese j. *Computers*, *13*, 605–611.

Chia, M.-L., Kuo, D., & Tung, M.-F. (2007). The multiple originator broadcasting problem in graphs. *Discrete Applied Mathematics*, *155*(10), 1188–1199.

Comellas, F., Harutyunyan, H. A., & Liestman, A. L. (2003). Messy broadcasting in multidimensional directed tori. *Journal of Interconnection Networks*, *4*(01), 37–51.

de Sousa, A., Gallo, G., Gutierrez, S., Robledo, F., Rodríguez-Bocca, P., & Romero, P. (2018). Heuristics for the minimum broadcast time. *Electronic Notes in Discrete Mathematics*, *69*, 165–172.

Diks, K., & Pelc, A. (1996). Broadcasting with universal lists. *Networks: An International Journal*, *27*(3), 183–196.

Dinneen, M. J. (1994). The complexity of broadcasting in bounded-degree networks. *arXiv preprint math/9411222*.

Dinneen, M. J., Fellows, M. R., & Faber, V. (1991). Algebraic constructions of efficient broadcast networks. In *Applied Algebra, Algebraic Algorithms and Error-Correcting Codes: 9th International Symposium, AAECC-9 New Orleans, LA, USA, October 7–11, 1991 Proceedings 9* (pp. 152–158).

Dinneen, M. J., Ventura, J. A., Wilson, M. C., & Zakeri, G. (1999). Compound constructions of broadcast networks. *Discrete Applied Mathematics*, *93*(2-3), 205–232.

Ehresmann, A.-L. (2021). *Approximation algorithms for broadcasting in flower graphs* (Unpublished doctoral dissertation). Concordia University.

Elkin, M., & Kortsarz, G. (2002). Combinatorial logarithmic approximation algorithm for directed telephone broadcast problem. In *Proceedings of the thiry-fourth annual ACM symposium on Theory of computing* (pp. 438–447).

Erdős, P., Rényi, A., & Sós, V. (1966). On a problem of graph theory, studia sci. *Math. Hungar.*, *1*, 215–235.

Farley, A. M. (1979). Minimal broadcast networks. *Networks*, *9*(4), 313–332.

Farley, A. M., Fragopoulou, P., Krumme, D., Proskurowski, A., & Richards, D. (2000). Multi-source spanning tree problems. *Journal of Interconnection Networks*, *1*(01), 61–71.

Farley, A. M., & Hedetniemi, S. T. (1978). Broadcasting in grid graphs. In *Proc. 9th SE Conf. Combinatorics, Graph Theory, and Computing, Utilitas Mathematica* (pp. 275–288).

Farley, A. M., Hedetniemi, S. T., Mitchell, S. M., & Proskurowski, A. (1979). Minimum broadcast graphs. *Discret. Math.*, *25*(2), 189–193.

Farley, A. M., & Proskurowski, A. (1981). Broadcasting in trees with multiple originators. *SIAM Journal on Algebraic Discrete Methods*, *2*(4), 381–386.

Fomin, F. V., Fraigniaud, P., & Golovach, P. A. (2023). Parameterized complexity of broadcasting in graphs. *arXiv preprint arXiv:2306.01536*.

Fraigniaud, P., & Lazard, E. (1994). Methods and problems of communication in usual networks. *Discrete Applied Mathematics*, *53*(1), 79–133.

Fraigniaud, P., & Vial, S. (1997). Approximation algorithms for broadcasting and gossiping. *Journal of Parallel and Distributed Computing*, *43*(1), 47–55.

Fraigniaud, P., & Vial, S. (1999). Comparison of heuristics for one-to-all and all-to-all communications in partial meshes. *Parallel Processing Letters*, *9*(01), 9–20.

Gargano, L., & Vaccaro, U. (1989). On the construction of minimal broadcast networks. *Networks*, *19*(6), 673–689.

Gholami, S., & Harutyunyan, H. A. (2022). Fully-adaptive model for broadcasting with universal lists. In *2022 24th International Symposium on Symbolic and Numeric Algorithms for Scientific Computing (SYNASC)* (pp. 92–99).

Gholami, S., & Harutyunyan, H. A. (2024). A note to non-adaptive broadcasting. *Parallel Processing Letters*, *34*(01), 2340017.

Gholami, S., Harutyunyan, H. A., & Maraachlian, E. (2023). Optimal broadcasting in fully connected trees. *Journal of Interconnection Networks*, *23*(01), 2150037.

Gregor, P., Škrekovski, R., & Vukašinović, V. (2018). Broadcasting multiple messages in the 1-in port model in optimal time. *Journal of Combinatorial Optimization*, *36*, 1333–1355.

Grigni, M., & Peleg, D. (1991). Tight bounds on mimimum broadcast networks. *SIAM Journal on Discrete Mathematics*, *4*(2), 207–222.

Harutyunyan, H., Laza, G., & Maraachlian, E. (2009). Broadcasting in necklace graphs. In *Proceedings of the 2nd Canadian Conference on Computer Science and Software Engineering* (pp. 253–256).

Harutyunyan, H., & Maraachlian, E. (2007). Linear algorithm for broadcasting in unicyclic graphs. In *Computing and Combinatorics: 13th Annual International Conference, COCOON 2007, Banff, Canada, July 16-19, 2007. Proceedings 13* (pp. 372–382).

Harutyunyan, H., & Maraachlian, E. (2009). Broadcasting in fully connected trees. In *2009 15th international conference on parallel and distributed systems* (p. 740-745). doi: 10.1109/ICPADS.2009.48

Harutyunyan, H. A. (2006). Minimum multiple message broadcast graphs. *Networks: An International Journal*, *47*(4), 218–224.

Harutyunyan, H. A. (2008). An efficient vertex addition method for broadcast networks. *Internet Mathematics*, *5*(3), 211–225.

Harutyunyan, H. A., Hovhannisyan, N., & Magithiya, R. (2022). Deep heuristic for broadcasting in arbitrary networks. In *2022 21st International Symposium on Parallel and Distributed Computing (ISPDC)* (pp. 1–8).

Harutyunyan, H. A., & Jimborean, C. (2014). New heuristic for message broadcasting in networks. In *2014 IEEE 28th International Conference on Advanced Information Networking* *and Applications* (pp. 517–524).

Harutyunyan, H. A., & Li, Z. (2017). Broadcast graphs using new dimensional broadcast schemes for knödel graphs. In *Algorithms and Discrete Applied Mathematics: Third International Conference, CALDAM 2017, Sancoale, Goa, India, February 16-18, 2017, Proceedings 3* (pp. 193–204).

Harutyunyan, H. A., & Liestman, A. L. (1998). Messy broadcasting. *Parallel Processing Letters*, *8*(02), 149–159.

Harutyunyan, H. A., & Liestman, A. L. (1999). More broadcast graphs. *Discrete Applied Mathematics*, *98*(1-2), 81–102.

Harutyunyan, H. A., & Liestman, A. L. (2001a). Improved upper and lower bounds for k-broadcasting. *Networks: An International Journal*, *37*(2), 94–101.

Harutyunyan, H. A., & Liestman, A. L. (2001b). k-broadcasting in trees. *Networks: An International Journal*, *38*(3), 163–168.

Harutyunyan, H. A., & Liestman, A. L. (2012). Upper bounds on the broadcast function using minimum dominating sets. *Discrete Mathematics*, *312*(20), 2992–2996.

Harutyunyan, H. A., Liestman, A. L., Makino, K., & Shermer, T. C. (2011). Nonadaptive broadcasting in trees. *Networks*, *57*(2), 157–168.

Harutyunyan, H. A., Liestman, A. L., Peters, J. G., & Richards, D. (2013). Broadcasting and gossiping. *Handbook of graph theory*, 1477–1494.

Harutyunyan, H. A., & Maraachlian, E. (2008). On broadcasting in unicyclic graphs. *Journal of combinatorial optimization*, *16*, 307–322.

Harutyunyan, H. A., & Maraachlian, E. (2009). Linear algorithm for broadcasting in networks with no intersecting cycles. In *PDPTA* (pp. 296–301).

Harutyunyan, H. A., & Shao, B. (2006). An efficient heuristic for broadcasting in networks. *Journal of Parallel and Distributed Computing*, *66*(1), 68–76.

Harutyunyan, H. A., & Wang, W. (2010). Broadcasting algorithm via shortest paths. In *2010 IEEE 16th International Conference on Parallel and Distributed Systems* (pp. 299–305).

Hedetniemi, S. M., Hedetniemi, S. T., & Liestman, A. L. (1988). A survey of gossiping and broadcasting in communication networks. *Networks*, *18*(4), 319–349.

Hovhannisyan, N. (2024). *Exact and factor two algorithms for broadcast time* (Unpublished doctoral dissertation). Concordia University. (Unpublished)

Hromkovič, J., Jeschke, C.-D., & Monien, B. (1990). Optimal algorithms for dissemination of information in some interconnection networks. In *Mathematical Foundations of Computer Science 1990: Banská Bystrica, Czechoslovakia August 27–31, 1990 Proceedings 15* (pp. 337–346).

Hromkovič, J., Jeschke, C. D., & Monien, B. (1993). Optimal algorithms for dissemination of information in some interconnection networks. *Algorithmica*, *10*, 24–40.

Hromkovič, J., Klasing, R., Monien, B., & Peine, R. (1996). Dissemination of information in interconnection networks (broadcasting & gossiping). *Combinatorial network theory*, 125–212.

Jakoby, A., Reischuk, R., & Schindelhauer, C. (1998). The complexity of broadcasting in planar and decomposable graphs. *Discrete Applied Mathematics*, *83*(1-3), 179–206.

Khachatrian, L., & Harutounian, O. (1990). Construction of new classes of minimal broadcast networks. In *Conference on Coding Theory, Armenia* (pp. 69–77).

Klasing, R., Monien, B., Peine, R., & Stöhr, E. A. (1994). Broadcasting in butterfly and debruijn networks. *Discrete Applied Mathematics*, *53*(1-3), 183–197.

Knödel, W. (1975). New gossips and telephones. *Discrete Mathematics*, *13*, 95.

Labahn, R. (1986). The telephone problem for trees. *Journal of Information Processing and Cybernetics*, *22*(9), 475–485.

Labahn, R. (1994). A minimum broadcast graph on 63 vertices. *Discrete Applied Mathematics*, *53*(1-3), 247–250.

Li, C., Hart, T. E., Henry, K. J., & Neufeld, I. A. (2008). Average-case ”messy” broadcasting. *Journal of Interconnection Networks*, *9*(04), 487–505.

Liestman, A. L. (1985). Fault-tolerant broadcast graphs. *Networks*, *15*(2), 159–171.

Liestman, A. L., & Peters, J. G. (1988). Broadcast networks of bounded degree. *SIAM journal on Discrete Mathematics*, *1*(4), 531–540.

Maheo, M., & Saclé, J.-F. (1994). Some minimum broadcast graphs. *Discrete Applied Mathematics*, *53*(1-3), 275–285.

Middendorf, M. (1993). Minimum broadcast time is np-complete for 3-regular planar graphs and deadline 2. *Information Processing Letters*, *46*(6), 281–287.

Mitchell, S., & Hedetniemi, S. (1980). A census of minimum broadcast graphs. *J. Combin. Inform. System Sci*, *5*, 141–151.

Park, J.-H., & Chwa, K.-Y. (1994). Recursive circulant: A new topology for multicomputer networks. In *Proceedings of the International Symposium on Parallel Architectures, Algorithms and Networks (ISPAN)* (pp. 73–80).

Pelc, A. (1996). Fault-tolerant broadcasting and gossiping in communication networks. *Networks: An International Journal*, *28*(3), 143–156.

Ravi, R. (1994). Rapid rumor ramification: Approximating the minimum broadcast time. In *Proceedings 35th Annual Symposium on Foundations of Computer Science* (pp. 202–213).

Rosenthal, A., & Scheuermann, P. (1987). Universal rankings for broadcasting in tree networks. In *Proceedings of the 25th Allerton Conference on Communication, Control and Computing* (pp. 641–649).

Saclé, J.-F. (1996). Lower bounds for the size in four families of minimum broadcast graphs. *Discrete Mathematics*, *150*(1-3), 359–369.

Scheuermann. (1984). Heuristic algorithms for broadcasting in point-to-point computer networks. *IEEE Transactions on Computers*, *100*(9), 804–811.

Schindelhauer, C. (2000). On the inapproximability of broadcasting time. In *Proc. of the 3rd International Workshop on Approximation Algorithms for Combinatorial Optimization Problems* (pp. 226–237).

Shao, B. (2006). *On k-broadcasting in graphs* (Unpublished doctoral dissertation). PhD thesis, Concordia University.

Slater, P. J., Cockayne, E. J., & Hedetniemi, S. T. (1981). Information dissemination in trees. *SIAM Journal on Computing*, *10*(4), 692–701.

Suderman, M. (1999). *Multiple message broadcasting / by matthew suderman.* Simon Fraser University.

Tale, P. (2024). Double exponential lower bound for telephone broadcast. *arXiv preprint arXiv:2403.03501*.

Xiao, J., & Wang, X. (1988). A research on minimum broadcast graphs. *Chinese J. Computers*, *11*, 99–105.

Zhou, J.-g., & Zhang, K.-m. (2001). A minimum broadcast graph on 26 vertices. *Applied mathematics letters*, *14*(8), 1023–1026.