A STUDY OF MULTILOOP NETWORKS

EDWARD MARAACHLIAN

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
CONCORDIA UNIVERSITY
MONTRÉAL, QUÉBEC, CANADA

APRIL 2006

Your file Votre référence
ISBN: 0-494-14329-0
Our file Notre référence
ISBN: 0-494-14329-0

Abstract

A Study of Multiloop Networks

Edward Maraachlian

Multiloop networks is a family of network topologies which is an extension of the ring topology. In this thesis we study the structural properties of bipartite double loop networks using the plane tessellation technique. We also study the problem of broadcasting in the bipartite double loop networks and in triple loop networks. For the first kind of graphs we find that the broadcast time is $d+2$ where d is the diameter of the graph. For the triple loop graphs, we give a $d+5$ upper bound on the broadcast time by providing an algorithm that completes broadcasting in at most $d+5$ time units. We also find a $d+2$ lower bound for the optimal triple loop graphs, these are the graphs with maximum number of nodes given a diameter d. Finally we give an upper bound for the broadcast time of undirected Circulant (also called multiloop) graphs of degree $2k$ which is $d+2k-1$.

Acknowledgments

I would like to express my sincere gratitude to my supervisor Hovhanness Harutyunyan who helped, guided, and motivated me since the first day that I arrived to Concordia.

I would also like to thank my friend and colleague Bin Shao for providing his source code to run the simulations of this thesis.

I would also like to thank my childhood friends Arthur, Kevork, Paul, and Vahe who constantly tried to make me feel not so away from home. Special thanks to my new friends Dikran, Nick, Pedro, and Samer who gave me very beautiful memories in Montreal hence made my two years so far very enjoyable. The Armenian Community Center of Montreal also provided me with a very warm environment and I am grateful to all of my friends there specially Zorig. Finally, my friends at the department made my long nights shorter with their support. Thank you Alaa, Emad, George, Issam, and Khaled.

I owe everything to my mom, my dad, and my sister Hilda. Without their endless love and support I wouldn't have been here.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

The need for high performance computing is constantly increasing. Examples of applications that are computation intensive or require large memory are weather prediction systems, astronomical calculations, protein dynamics simulations (on the IBM Blue Gene), and human genome sequencing, to name a few. The traditional von-Neumann model serial computers either cannot solve these problems in an acceptable amount of time, or cannot deal with the enormous size of these problems. Parallelism, where more than one processors works on solving the problem, is the answer to the restrictions of the traditional computers. There are different models for parallel computing. One of the most common models is the MIMD (Multiple Instruction and Multiple Data) which is sometimes referred to multicomputers or multiprocessors. The different processors, working in parallel, will most probably need to exchange data among each other. This is done either through a shared memory or an interconnection network. Shared memory multicomputers have a limitation on the number of processors that can be connected together. Hence, it is not practical if a very large number of processors is to be connected. A more realistic way of designing multicomputers is to make each processor have its own main memory. Communication between the processors will be accomplished by passing messages using an interconnection network. It turns out that the performance of these multicomputers not only depends on the processing power of the processors but also on the performance of the interconnection network in disseminating data among the processor. Research has shown that the structural properties of a network determine many of its properties such as the minimum broadcast time, ease of routing, and fault tolerance.

Multiloop networks, (also known as chordal graphs), are an extension of the simple ring topology. Ring topologies have been the network of choice for implementing local area networks (LANs) and SONETs ([26],[27],[4]) because of their simplicity, regularity, low degree, symmetry, and expendability. They have some major disadvantages too which are their low fault tolerance and long diameter. Improvements on the ring topologies can be done by adding extra links, but this should be done carefully to preserve, as much as possible, the symmetry and regularity of the network which are two desirable properties which would make routing and switching protocols easy to implement.

Broadcasting is the problem of disseminating a piece of information, owned by certain node called the *originator*, to all other nodes [25]. This is one of the communication primitives of parallel processing, hence inefficient broadcasting can be a bottleneck in the performance of multicomputers. Broadcasting is performed by placing a series of calls along the communication lines of the network. At any time, the informed nodes contribute to the information dissemination process by informing one of their uninformed neighbors. In this thesis we will adopt the often used classical broadcast model which is the following:

- Each call involves only one informed node and one of its uninformed neighbors.
- Each call requires one unit of time.
- A node can participate in only one call per unit of time.
- In one unit of time, many calls can be performed in parallel.

The *k-port* broadcast model is similar to the classical model described above with only one difference which allows an informed neighbor to inform k of its uninformed neighbors in parallel and in one time unit.

A *broadcast scheme* of an originator u is a set of calls that completes the broadcasting in the network originated at vertex u. An optimal broadcast scheme informs all the vertices in the least amount of time as possible. Two very good surveys on broadcasting can be found at [24], [25].

As it is usually done, we will represent a network by a graph $G = (V, E)$ where the vertices V represent the nodes or the processing elements of the network, and the edges E represent the communication links between the processors. In the rest of this paragraph we will present some graph theoretic definitions that will be used in the rest of this thesis. A vertex v is said to be a neighbor of u if there is edge connecting the two. A path between two vertices is a sequence of edges that connect the two vertices together. The distance between two vertices u and v, $d(u,v)$ is the length of the shortest path between the two vertices. The diameter of a graph is defined to be the largest distance between any two vertices, $d = max_{u,v \in V}\{d(u,v)\}$. The degree of a vertex v, $\delta(v)$, in an undirected graph $G = (V, E)$, is the number of neighbors that v has. The degree of a graph $G = (V, E)$, $\Delta(G)$, is the maximum degree of its vertices, $\Delta(G) = max_{v \in V}\{\delta(v)\}$. A graph $G = (V, E)$ is bipartite if there exists a partition $V = V_1 \cup V_2$ where $V_1 \cap V_2 = \phi$ and any edge in E is incident on one vertex in V_1 and another vertex in V_2. Two graphs with the same number of vertices connected in the same way are considered to be *isomorphic*. More formally, two graphs $G = (V, E)$ and $G' = (V', E')$ are isomorphic, $G \cong G'$, if there exists a bijection $\varphi : V \rightarrow V'$ such $(x, y) \in E \Leftrightarrow (\varphi(x), \varphi(y)) \in E'$. The map φ is called *isomorphism*. If $G = G'$ then φ is called *automorphism*. A graph $G = (V, E)$ is called *vertex transitive* if for any two vertices $u, v \in V$ there is an automorphism of G mapping u to v. For all other definitions of graph theoretic terms refer to [14].

Given a connected graph $G = (V, E)$ and a message originator vertex, u, the *broadcast time* of u, $b(u, G)$ or $b(u)$, is the minimum number of time units required to complete broadcasting from the vertex u. Note that, for any vertex u in a connected graph G on N vertices, $b(u) \geq \lceil \log N \rceil$ since during each time unit the number of informed vertices can at most be doubled. The broadcast time of the graph G denoted as $b(G)$ is defined as $b(G) = max\{b(u) | u \in V\}$ [25].

One direction of research is to find the minimum broadcast time of various networks. But calculating the broadcast time of a general graph is proven to be NP-complete [41]. On the other hand, the broadcast time of common network architectures have already been calculated [3], [7], [16], [18]. For the case of trees, [41] gave a polynomial time broadcast algorithm which calculates the optimum broadcast scheme. In [20] a polynomial time algorithm was given that performs broadcasting in any unicyclic graph in a minimum possible time. If the broadcast time cannot be calculated easily one approach is to find some lower and upper bounds on the time. For the case of general graphs, given that finding the broadcast time is NP-complete, one possible approach is to design polynomial time heuristics that calculate near optimum broadcast schemes.

Another direction of research in broadcasting is to design graphs G on N vertices such that the broadcast time $b(u) = \lceil \log N \rceil$ and G has as few edges as possible. These graphs are called minimum broadcast graphs (mbg). There is extensive research in determining the minimum number of edges, $B(N)$, required to construct a mbg on N vertices. This problem proved to be very difficult. Results concerning the value of $B(N)$ for different values of N (1 to 22, 26 to 32, 58 - 63, 127, 2^p, $2^p - 2$) can be found in [13], [17], [6], [44], [45], [40],[28], [29]. For a graph $G = (V, E)$ and diameter d, $b(G) \geq d$. As a result, the problem of designing graphs with maximum number of vertices N for a given diameter d has been of interest. A graph G of a given diameter d is called an *optimal graph* if it has the maximum number of vertices for that given value of diameter. One variant of this problem is finding the maximum number of vertices (maximum order) of graphs with a fixed diameter and fixed degree. This problem is known as the *diameter-degree* problem or (d, δ) problem, and has been studied quite extensively [38],[31]. An upper bound on the number of vertices N was given by Moore. We will be interested in the problem of maximizing N in multiloop networks only. This is a restricted form of the (d, δ) problem where in addition to the degree and the diameter the graphs are required to have a certain structure.

In this thesis we design a degree 4 regular bipartite double loop graphs that have the maximum number of vertices for a given diameter. We also present the minimum broadcast time, which is $d+2$, of these graphs and present an algorithm that informs all vertices in a $d+2$ time units. Then we study the broadcast problem in triple loop graphs. First we consider the optimal triple loop graph and find upper and lower bounds on the broadcast time. In particular we find that $d+2 \leq b(G) \leq d+5$. We present an algorithm and prove that it informs all vertices in at most $d+5$ time units. Finally, we deal with the multiloop graphs of degree $2k$, we present an upper bound of the broadcast time which is: $b(G) \leq d+k-1$. We present the algorithm that informs all the vertices in a time less than or equal to this upper bound. In order to have a more comprehensive study of multiloop graphs we apply the broadcast algorithm developed of [21] to the double and triple loop graphs. This algorithm has not been tested in multiloop graphs and we get excellent match between the simulation and theoretical results. Moreover, the results of simulations in the case of triple loop graphs come to suggest that an improvement on the theoretical results can be obtained.

The thesis is structured as follows: The first chapter is an introduction that defines most of the basic graph theoretic terminology used in this thesis and also gives the motivation for this work. Chapter 2 gives a literature review of the multiloop graphs. Chapter 3 presents the theoretical contributions of this thesis. Chapter 4 contains the simulation results of the broadcast algorithm developed by Harutyunyan and Shao [21]. The final chapter is a conclusion and contains possible directions for future work related to our current work.

Chapter 2

A Review of Multi-Loop Networks

Multiloop Networks are an extension of the ring networks. In this section we will present a survey of the important results concerning multiloop networks, very good surveys can be found in [26],[27], and [4]. One of the central problems in computer networks research is to design network topologies that have good properties. These properties can be grouped into two major categories: better performance and lower cost. Cost usually stands for the number of the links, and performance usually refers to fault tolerance, broadcast time, or ease of routing schemes. Fault tolerance is the ability of the network to respond gracefully to an unexpected failure of a node or a link. The best performance can be achieved when all vertices of a graph are connected to all the rest, i.e in a complete graph. But this is practically impossible and very costly. A compromise between cost and performance should be done and that is why there is research to design networks with certain structural characteristics, for example having as few edges as possible, and optimum performance. Knowing that the broadcast time of a graph $G = (V, E)$ with diameter d is $b(G) \geq d$, one can see the importance of designing networks on N vertices with as small diameters as possible. In this respect, researchers studying multiloop networks have addressed two problems: *Problem 1*: Given N vertices and a degree δ, design the δ-regular multiloop network having the smallest diameter d possible. This problem has proved to be difficult and people have instead tackled a different and somehow an easier version of it. *Problem 2*: Given the diameter d and a degree δ construct the δ-regular circulant network with diameter d with the maximum possible number of vertices N. In what follows we will present results of how this second problem was solved for various multiloop networks [31].

2.1 Design of Multiloop Networks

In this section we will survey the different mulitloop networks: their structural properties, minimum broadcast times, and broadcast algorithms. One of the techniques to deal with this problem was the plane tessellation technique which will be described in subsection 2.1.1. First we will give the definitions of the different graphs that will be encountered in this thesis.

Definition 1 (Chordal rings). *A chordal ring , $CR_N(c)$, on a set V of N vertices, where N is even, is the bipartite graph over $V = \{0, \cdots, N-1\}$ where $V = V_0 \cup V_1$, $V_0 = \{0, 2, \cdots, N-2\}$, and $V_1 = \{1, 3, \cdots, N-1\}$. A vertex $i \in V_0$ is connected to vertices $j = i+1, i-1, i+c(modN) \in V_1$ while a vertex $j \in V_1$ is connected to* *vertices* $i = j+1, j-1, j-c(modN) \in V_0$*, where* $c < N-1$ *is an odd integer.*

Chordal rings ([2],[37]) are nothing but the basic ring structure augmented by one extra edge per vertex. This edge, similar to a chord, connects two vertices on the ring. To preserve the symmetry of the graph, the chord length is kept constant for all vertices. This definition can be generalized and the generalized chordal rings can be defined as follows [46]:

Definition 2 (Generalized chordal rings). *Generalized chordal rings,* $GCR_N(a, b, c)$*, are bipartite graphs on the set of vertices* $V = \{V_0 \cup V_1\}$ *where* $|V| = N$ *and is even,* $V_0 = \{0, 2, \cdots, N-2\}$*, and* $V_1 = \{1, 3, \cdots, N-1\}$*. A vertex* $i \in V_0$ *is connected to vertices* $j = i+a, i+b, i+c\ (modN) \in V_1$ *while a vertex* $j \in V_1$ *is connected to vertices* $i = j-a, j-b, j-c\ (modN) \in V_0$*, where* $a, b,$ *and* c *are odd integers.*

The multiloop graph family is a general family which includes the double and triple loop graphs. The multiloop graphs are sometimes referred to as circulant graphs.

Definition 3 (Undirected Multiloop Graphs). *Multiloop (also called circulant) graphs* $C_N(\pm s_1, \pm s_2, \cdots, \pm s_k)$ *are defined on the set of vertices* $V = \{0, 1, \cdots, N-1\}$*, where* $|V| = N$*, and the set of edges* E *such that a vertex* i *is connected to vertices* $i \pm s_1, i \pm s_2, \cdots, i \pm s_k(modN)$*.*

The multiloop graphs are also called circulant graphs because the matrices representing these graphs (adjacency matrix representations) are circulant, i.e. rows are cyclic shifts of each other. In the literature, sometimes the double loop graphs are referred to as chordal graphs [8], [10].

2.1.1 Plane Tessellation Technique

The plane tessellation is a technique that maps the vertices of a graph to cells on a lattice. This visualization method was originated by Wong and Coppersmith [43] and Fiol, Yebra, Morillo, Alegre [46]. The mapping starts by assigning vertex 0 to an arbitrary cell. The rest of the mappings are performed according to some rules as it is explained below. After mapping a graph on a lattice one can define a tile or a minimum distance diagram (MDD) which is a set of the lattice cells. Every cell in the MDD corresponds to one and only one vertex of the graph and vice versa. Moreover, among all the cells corresponding to a vertex i the cell that is chosen to be part of the MDD is the one that is at the shortest distance from the cell corresponding to vertex 0. To simplify the notation we will refer to a cell corresponding to vertex i as cell i. For vertex transitive circulant graphs (all the graphs considered in this thesis are vertex transitive) there is no loss of generality when we take vertex zero as the source from which the distances are measured. Wong and Coppersmith [43] gave an algorithm to construct the MDD. First the $\mathbb{R}^d$ space is filled by a d dimensional lattice. In order to do the mapping, the cells in the lattice should have as many neighbors as the degree of the graph. Once a vertex i is mapped to a cell i, then every cell neighboring cell i will be assigned to one of the vertices $i \pm s_j$, $1 \leq j \leq k$, neighboring vertex i in the graph $C_N(\pm s_1, \pm s_2, \cdots, \pm s_k)$. All additions and subtractions are modulo N. This mapping assigns to the each of the infinitely many lattice cells a number between 1 and N. It was proven that every MDD tessellates the $\mathbb{R}^d$ space [27]. In other words, this means that the MDD gets repeated periodically next to each other and completely fills the

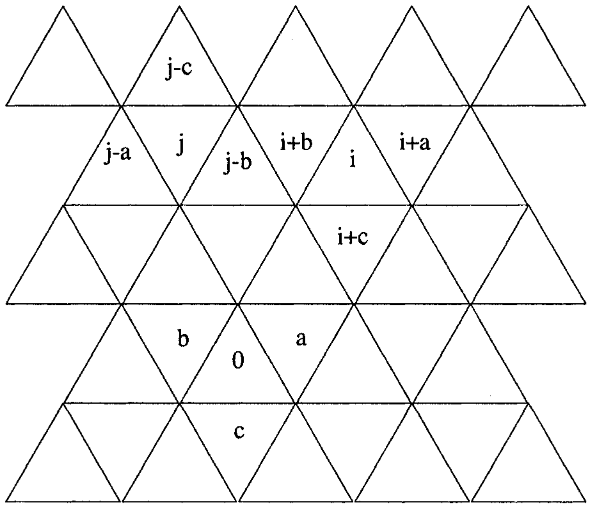

Figure 1: The mapping rules from the GCR vertices to the triangular lattice cells.

space. This property can be used in designing graphs. But it should be noted that not every tessellated MDD (tile) corresponds to a graph.

In particular, this geometric method was used in designing multiloop networks that have the maximum possible number of vertices for a fixed diameter. Some examples are the generalized chordal graphs [37], double loop graphs [46], double loop digraphs [11], triple loop graphs [46], and triple loop digraphs [36],[1]. The geometric visualization technique was also used in finding minimum broadcast times and broadcast schemes [12],[33].

To illustrate the plane tessellation technique we give the following example. Consider the generalized chordal rings, $GCR_N(a, b, c)$ were a vertex has 3 neighbors. A cell corresponding to a vertex should neighbor 3 cells, hence the triangular lattice is needed to apply the plane tessellation technique. Every vertex of the GCR will be

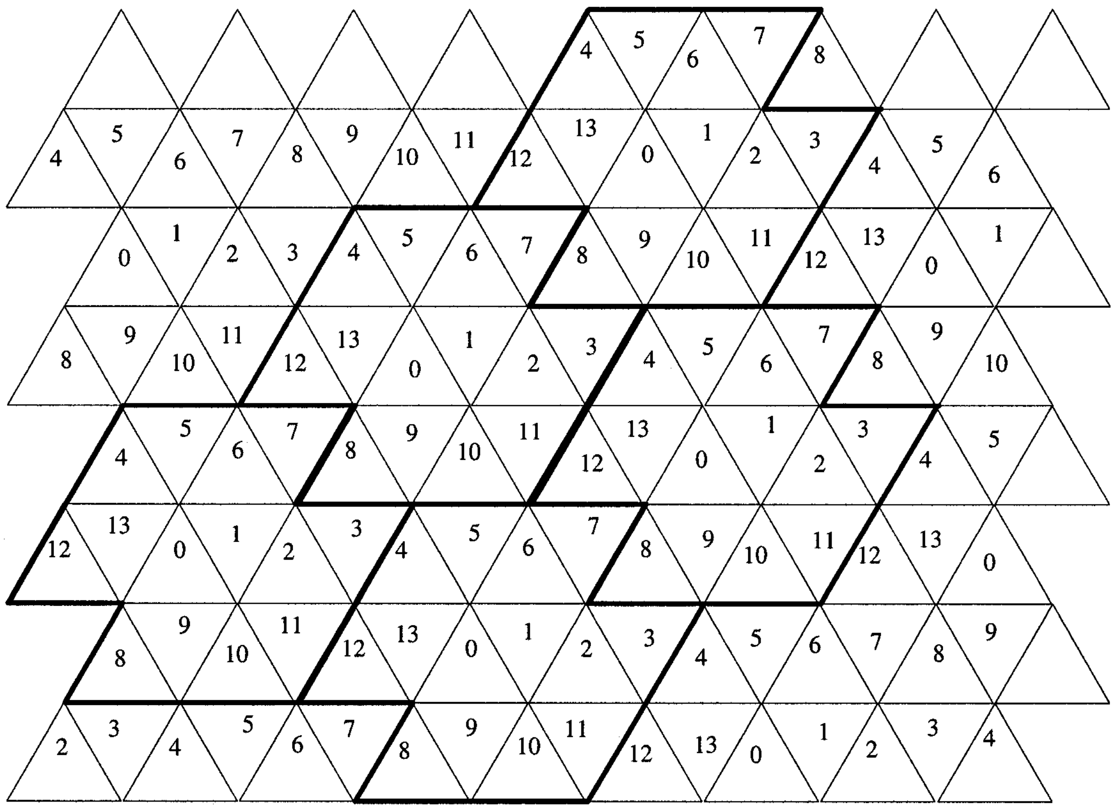

Figure 2: The mapping of $GCR_{14}(-1, 1, 9)$ onto a triangular lattice. The MDD is bordered by a bold line.

mapped to a set of triangular cells according to certain rules. The mapping (see figure 1) can be done by first choosing an arbitrary "up" pointing triangle to correspond to vertex 0. Then for every triangle corresponding to an even vertex i, the triangle on its right is mapped to vertex $i + a$, the triangle on its left is mapped to vertex $i + b$, and the triangle below it is mapped to vertex $i + c$. There are similar mapping rules for odd vertices: for every triangle corresponding to an odd vertex j, the triangle on its left is mapped to $j - a$, the one on its right is mapped to $j - b$, and the one on its top is mapped to $j - c$. It can be observed that, as seen in Fig. 2, every node of the graph will be mapped to infinitely many triangles and taking one triangle of each label will form a tile that tessellates the plane [37] . If the problem is to design a GCR with a certain N one has to find a MDD of N cells that tessllates the plane. Hence the design problem reduces to a geometric problem. Usually, it is easy to construct a tile that tessellates the plane, but one has to show that there is an $GCR_N(a, b, c)$ that corresponds to the drawn tile. For GCRs in [37] it has been shown how an optimal GCR with diameter d is calculated. The details of the illustration are of the same spirit as the details of Chapter 3.

2.1.2 A Survey of Important Results

In this subsection we present some of the results concerning the structural properties of the different graphs defined above. For GCRs it was proven, in [37], that $GCR_N(1, -1, 3d)$ where $N = \frac{3d^2+1}{2}$ has the maximum number of vertices for an an odd diameter d. This GCR is called the optimal GCR for odd values of the diameter

d. For an even diameter d it was proven that the upper bound on the number of vertices is $N \leq \frac{3d^2}{2}$. Using the plane tessellation technique it was shown that this upper bound is not attainable, instead a GCR with even diameter d and $N = \frac{3d^2-d}{2}$ was found to exist. This graph is $GCR_N(1, -1, 3d+1)$ and is called *quasioptimal* GCR. Therefore, the maximum value of vertices that a GCR with even diameter can have is $\frac{3d^2-d}{2} \leq N < \frac{3d^2}{2}$.

In [46] the double loop graphs were considered and it was shown that $C_N(\pm 1, \pm 2d+1)$ are optimal double loop graphs where $N = 2d^2+2d+1$. The graph $C_N(d, d+1)$ can also achieve the same upper bound on N. Furthermore, $C_N(d, d+1)$ is a graph with minimum possible diameter d for $2(d-1)^2+2(d-1)+1 = 2d^2-1 < N \leq 2d^2+2d+1$. In [39] a node removal procedure was suggested that obtains the tile corresponding to a double loop graph with diameter d and N vertices such that $2d^2-1 < N \leq 2d^2+2d+1$. The node removal procedure removes nodes off the tile corresponding to the optimal double loop graph and creates a tile corresponding to a non-optimal double loop graph. In [10] the average distance of the graph $C_N(1, 2d+1)$ with diameter d was calculated to be $\frac{2d(d+1)(2d+1)}{3(2d^2+2d+1)}$ where $N = 2d^2+2d+1$. In [9] it was proven that the $C_N(d, d+1)$ with $N = 2d^2+2d+1$ has the minimum average distance among all circulant graphs of degree 4 and diameter d. The average distance was calculated to be $d[1 - \frac{2(d^2-1)}{3(N-1)}]$. In [34]it was proven that the optimal $C_N(1, 2d+1)$ is isomorphic to the optimal $C_N(d, d+1)$ graph.

The triple loop graph $C_N(\pm a, \pm b, \pm c)$ was considered in [46] for the case where $c = -(a+b)$. Using the plane tessellation technique it was shown that the maximum number of vertices that a graph of a certain diameter can have is $N = 3d^2 + 3d + 1$. The graph that has this many vertices is $C_N(3d+1, 1, -3d-2)$ and $C_N(d, d+1, 2d+1)$. The optimal graph where $a = 1$, and arbitrary values of a and b was solved by E. Monakhova [35], this results are presented in Chapter 3.

2.2 Broadcasting in Multiloop Networks

As it was seen in the previous section, the structural properties of the different multiloop graphs have been studied to a great extent in the last couple of decades. The routing problem in these networks, which is not discussed in this thesis, has also been discussed to some extent. On the other hand the broadcast problem is studied only for three of the multiloop graphs: the generalized chordal graphs, the double loop graphs, and the triple loop networks. In this section we will present these results. Comellas and Hell in [12] studied the broadcasting problem in generalized chordal rings and provided a broadcast scheme that informs all vertices in at most $d+2$ time where d is the diameter of the graph. They proved that the minimum broadcast time of a $GCR_N(a, b, c)$ of diameter d is $d+2$ when d is odd and is equal to $d+1$ when the d is even. The results were obtained based on the work of Ko on broadcasting in triangular grids [12]. These results were obtained by making use of the plane tessellation technique.

Broadcasting in double loop graphs was studied in [33],[39]. There exists a broadcast scheme [33] that guarantees that broadcasting will be completed in at most $d+2$

time where d is the diameter of the graph. They also proved that $d + 2$ is the exact broadcast time of optimal double loop graphs.

They presented the following broadcast algorithm that performs broadcasting in a minimum time. On the square grid where the edge $(i, i + d + 1)$ corresponds to vertically adjacent squares and the edge $(i, i+d)$ corresponds to horizontally adjacent squares, the broadcasting scheme is defined as follows [33] :

1. Vertex 0 sends the message to the vertices to the vertices to the left, right, above, and below it in that order.

2. A vertex that receives a message from the right sends the message to the vertices to the left, above, and below it in that order.

3. A vertex that receives a message from the left sends the message to the vertices to the right, above, and below it in that order.

4. A vertex that receives a message from below sends the message to the vertex above it.

5. A vertex that receives a message from above sends the message to the vertex below it.

This scheme completes broadcasting in $d + 2$ time steps, the verification that the broadcast time is minimal is done by using geometric arguments on the tile that corresponds to the optimal graph.

In [39], Obradovic et al studied the problem of broadcasting in double loop graphs for different broadcast models. In particular, they studied the k-port broadcast model where a vertex can inform k of its neighbors in one time unit. They also considered cases where every vertex v has a certain lifetime, $A(v)$, during which it can be active and inform its neighbors. This model was suggested as a new approach to study the fault tolerance of the double loop graphs.

Broadcasting in triple loop graphs was studied only for the restricted case of $s_3 = -(s_1 + s_2)$ in [33]. A broadcast scheme was given that completes broadcasting in at most $d + 3$ time. It was also proven that $d + 3$ is the minimum broadcast time of optimal triple loop graphs of the restricted type.

Chapter 3

New Results on MultiLoop Graphs

In the first part of this chapter we will present a new kind of circulant graphs called the bipartite double loop graphs of degree 4. We will calculate the size of the optimal graph i.e. the maximum number of vertices N that a graph of diameter d can have. We will also find the minimum broadcast time of these graphs. In the second part we will study the problem of broadcasting in optimal and non-optimal triple loop graphs. Finally we will present an upper bound on the broadcast time of circulant graphs $C_N(\pm s_1, \pm s_2, \cdots, \pm s_k)$ of degree $2k$.

3.1 Bipartite Double Loop Graphs

First we define the bipartite double loop graphs as follows:

Definition 4. *A bipartite double loop graph, BDLG, is a bipartite circulant graph* $BC_N(\pm s_1, \pm s_2)$ *on the set of vertices* $V = V_0 \bigcup V_1$ *where* $V_0 = \{0, 2, \cdots, N-2\}$

and $V_1 = \{1, 3, \cdots, N-1\}$ *and such that a vertex* $i \in V_0$ *is a neighbor of vertices* $j = i \pm s_1, i \pm s_2 \in V_1$ *and a vertex* $j \in V_1$ *is a neighbor of vertices* $j = i \mp s_1, i \mp s_2 \in V_0$*, where* s_1 *and* s_2 *are two different odd integers.*

3.1.1 Structural Properties

Theorem 1. *The BDLGs are vertex transitive.*

Proof. A graph is vertex transitive if for any two vertices i and j there exists some automorphism mapping i to j. Define the family of functions $f(i) = i + \alpha$ where α can be any integer different than 0. Let a and b be any two vertices. The automorphism satisfying $f(a) = b$ is $f(i) = i + (b - a)$. Applying this function on any two vertices x and y we get $f(x) = x + b - a$ and $f(y) = y + b - a$. $(f(x), f(y))$ will be an edge only if $f(y) = f(x) \pm k$ where $k = s_1$ or $k = s_2$. Substituting the values of the functions we get: $y + b - a = x + b - a \pm k$ which is equivalent to $y = x \pm k$. This means that the transformed vertices will be adjacent only if the original ones were too. Therefore, the suggested family of functions preserves the structure of the graph hence is an automorphism. □

In order to study the connectedness of the graph $BC_N(\pm s_1, \pm s_2)$ we have to show that starting from any vertex i one can reach all of the vertices $j \in V$. Since the graph is vertex transitive it is enough to check if starting from vertex 0 one can reach all the vertices j of the graph by taking a certain number of the $\pm s_1$ and $\pm s_2$ steps. Starting at vertex 0, in order to reach an odd vertex j, it is enough to reach one of the even vertices $j \pm s_1$ and $j \pm s_2$. Therefore, it is enough to show that one can reach any of the even vertices from vertex 0 taking a certain number of steps. In particular, if vertex 0 can reach vertex 2 after a number of steps then it is guaranteed that any other even vertex $2k$ can be reached by repeating the steps that took 0 to 2 k times. Therefore, any odd vertex can be reached too from one of its even vertices, which implies that the graph is connected. Since going from one even vertex to another involves taking an even number of steps, we will define the composite steps $A = s_1 + s_2$ and $B = s_1 - s_2$. The connectedness condition can be written mathematically as follows:

$$(\alpha A + \beta B) \text{mod } N = \alpha A + \beta B + \gamma N = 2 \tag{1}$$

where α and β can be any positive or negative integers. Using Bezout's lemma one can write condition 1 as follows ($GCD \equiv$ greatest common devisor):

$$GCD(A, B, N) = GCD(s_1 + s_2, s_1 - s_2, N) = 2 \tag{2}$$

First we will calculate the maximum number of vertices that a $BDLG$ can have. We will use the plane tessellation technique to study the structural properties. To start with, a square lattice is assumed to fill the $\mathbb{R}^2$ space. The vertex 0 will be mapped to an arbitrary square on the lattice. As it was done earlier, we will adopt the convention that square i refers to a square corresponding the vertex i. The mapping rules are as follow: The square on the right of square i will correspond to vertex $i + s_1$. The square on the left will correspond to $i - s_1$. Similarly, the squares on the top and bottom of square i correspond to vertices $i + s_2$ and $i - s_2$ respectively.

Mapping all the squares on the lattice to a vertex of the graph we obtain a pattern that gets repeated. We want to determine the shape of the minimum distance diagram (MDD), that tessellates the plane. The diameter of an MDD is is a quantity which is equal to the value of the diameter of the graph corresponding to the MDD.

Since we do not know the structure of the optimal graph we will try to guess the shape of an MDD with maximum number of squares that can tessellate the infinite square lattice. For a given diameter d we first need to know how many squares can an MDD have. According to the definition of BDLGs the number of squares should be even, and the number of even squares (squares corresponding to even vertices) should be equal to the number of the odd squares. There are at most $4i$ squares at distance i from square 0 [34]. Moreover, if i is odd (even) then all the squares at distance i correspond to odd (even) vertices. Hence, we can calculate the maximum number of vertices that are at a distance of d or less from vertex 0 by the following summation: $N \leq \sum_{i=1}^{d} = 1 + 4i = 2d^2 + 2d + 1$. We can improve on this upper bound since we know that there should be equal number of even and odd squares. If d is even, then we can calculate the upper bound of the even squares as: $N_{even} \leq \sum_{i=2,even}^{d} = 1 + 4i = 1 + (\frac{8+4d}{2})\frac{d}{2} = d^2 + 2d + 1$. The upper bound on the odd squares is $N_{odd} \leq \sum_{i=1,odd}^{d} = 4i = (\frac{4+4(d-1)}{2})\frac{d}{2} = d^2$. The number of even squares is more than the number of odd squares so a tighter upper bound on the size of the MDD is $N \leq 2N_{odd} = 2d^2$. In case d is odd we will have $N_{even} \leq \sum_{i=2,even}^{d} = 1 + 4i = 1 + (\frac{8+4(d-1)}{2})\frac{d-1}{2} = d^2$ while $N_{odd} \leq \sum_{i=1,odd}^{d} = 4i = (\frac{4+4(d)}{2})\frac{d+1}{2} = d^2 + 2d + 1$. In this case the number of even squares is less than the number of odd squares and

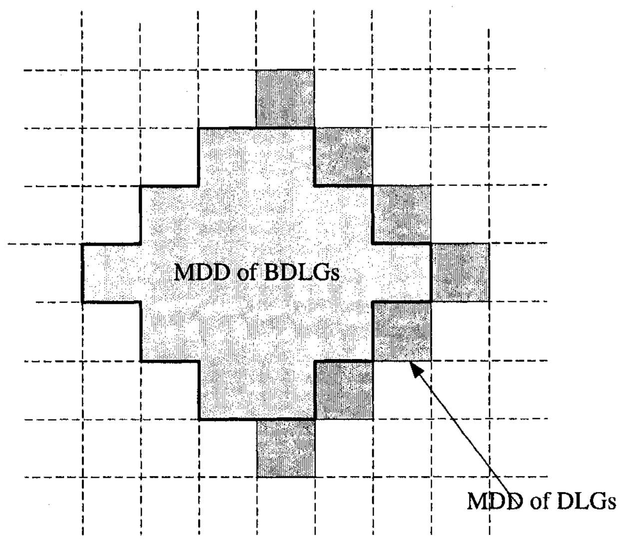

Figure 3: The MDD of the BDLG is a subset of the MDD of DLGs

hence, $N \leq 2N_{even} = 2d^2$. We notice that the upper bound on the size of the MDD is independent of the parity of the diameter d. Next we will prove that this upper bound can be attained by showing the existence of a BDLG of diameter d that has $2d^2$ vertices.

Optimal *BDLG*s

Above, we calculated the maximum size of the MDD of a certain diameter. Now we will check if this size can be attained. The first step is to find an MDD that tessellates the $2d$ plane. Then, we should find if there are any *BDLG*s that correspond to the specific MDD.

In order to guess the shape of the MDD one should notice there should be equal number of even and odd squares on the boundary of the MDD. This is because the graph is bipartite and every odd vertex has only even vertices as neighbors and vice versa. We will construct the MDD by removing cells from the MDD corresponding to an optimal double loop graph with the same diameter (Fig. 3). The squares at distance d are of the same parity, but in the MDD of the $BDLG$ there should be equal number of even and odd squares on the boundary. The MDD of the optimal double loop graph has $2d^2 + 2d + 1$ squares, while the MDD of the $BDLG$ has only $2d^2$. It is obvious that the MDD of the BDLGs (Fig. 3) is a subset of the MDD of the DLGs. One approach to construct the MDD of BDLGs is to remove $2d + 1$ squares from the boundary squares of the MDD of the DLG. We will remove the squares off the upper right and lower right boundaries of the MDD. As a result, on the boundary of the MDD there will be $2d - 1$ squares at distance d from vertex 0 having the same parity as d and $2d - 1$ squares at distance $d - 1$ from square 0 and hence having the opposite parity of d. This resulting MDD will prove to be sufficient to design the optimal BDLG.

The second step is to check if there exists a $BDLG$ that maps to the MDD suggested above. This will be done by tessellating the plane with the MDD and calculating the values of s_1 and s_2. It can be observed that, (see Fig. 4), there is more than one way where the MDDs can tessellate the plane. In more details, once the first tile is positioned, a second tile can be placed by either positioning its leftmost square next to the rightmost square of the first MDD, or positioning it on the top of the rightmost square of the first MDD, or on the top of any of rightmost boundary squares of the first MDD. We will not adopt the usual method of calculating the step

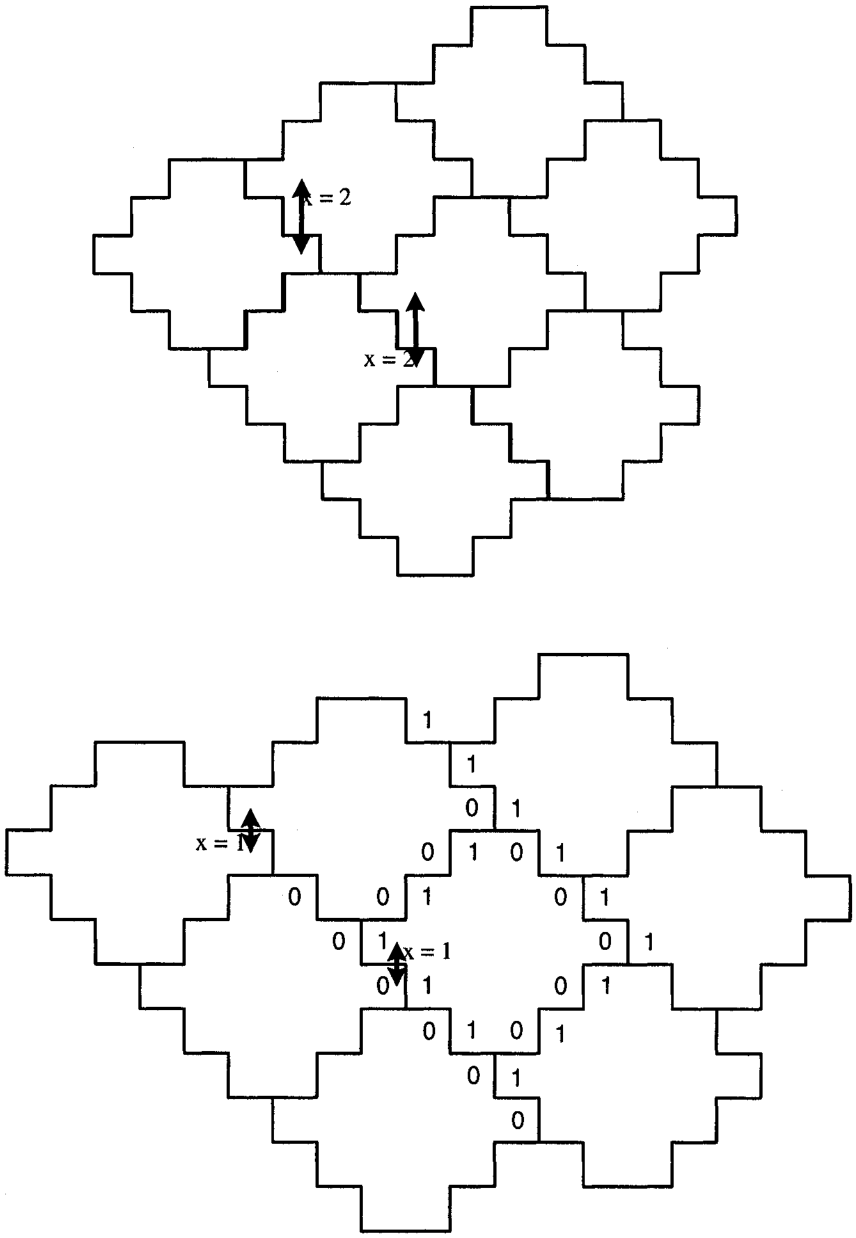

Figure 4: Shows two possible tessellation of the same MDD. The squares mark with 0 are even squares while those marked with 1 are odd squares. The first figure shows that the boundary has equal number of even odd squares, and that even squares on the boundary are neighboring odd squares on the boundary of the neighboring MDD

sizes s_1 and s_2 from the tessellation because there are plenty of tessellations but most of them do not correspond to a BDLG. We will use a simple algebraic technique where we will define the tessellation in terms of a variable x and we will calculate the value of x after imposing the condition that the tessellation corresponds to a legal $BDLG$. The variable x is taken to be the vertical distance between the rightmost square of the first tile and the leftmost square of the MDD that is on the right of the first MDD (Fig 4). In terms of x we can find the position of the zero square on two neighboring MDDs.

$$((2d-x)s_1 + xs_2) mod N = 0$$

$$((d-x)s_1 + (d+x)s_2) mod N = 0$$

These equations can be written without using the mod, as follows:

$$(2d-x)s_1 + xs_2 = \alpha N$$

$$(d-x)s_1 + (d+x)s_2 = \beta N$$

Writing the equations in the matrix form and using the fact that $N = 2d^2$ allows us to solve s_1 and s_2.

$$\begin{pmatrix} 2d-x & x \\ d-x & d+x \end{pmatrix} \begin{pmatrix} s_1 \\ s_2 \end{pmatrix} = 2d^2 \begin{pmatrix} \alpha \\ \beta \end{pmatrix}$$

By inverting the matrix we get:

$$\begin{pmatrix} s_1 \\ s_2 \end{pmatrix} = \begin{pmatrix} d+x & -x \\ x-d & 2d-x \end{pmatrix} \begin{pmatrix} \alpha \\ \beta \end{pmatrix}$$

We get the following values of s_1 and s_2 in terms of x, α, and β.

$$s_1 = (d + x)\alpha - x\beta$$

$$s_2 = (x - d)\alpha + (2d - x)\beta$$

Equation (2) is satisfied if $s_1 + s_2 = s_1 - s_2 + 2$, and if $s_1 + s_2$ and $s_1 - s_2$ are both even. First we enforce the condition $s_1 + s_2 = s_1 - s_2 + 2$ and we get:

$$2\alpha x - 2\beta x + 2d\beta = 2\alpha d - 2d\beta + 2$$

Solving for x we get:

$$x = \frac{2\alpha d - 2d\beta - 2d\beta + 2}{2(\alpha - \beta)}$$

$$x = \frac{\alpha d - 2d\beta + 1}{(\alpha - \beta)} \tag{3}$$

Since α and β are arbitrary numbers we can choose the values of $\alpha = 2$ and $\beta = 1$. This gives that $x = 1$. Hence, the value of $s_1 = 1$ and $s_2 = 2d + 1$. Note that $s_1 + s_2$ and $s_1 - s_2$ turn out to be even and hence condition 2 is satisfied.

We want to underline the differences between the BDLGs that were discussed above and the DLGs studied in [46] and [33]. BDLGs are, by definition, bipartite where the set of vertices $V = V_0 \cup V_1$ with the restriction that $|V_0| = |V_1| N/2$. On the other hand, the DLGs are not bipartite in general, but the optimal DLGs happen to be bipartite since the step lengths s_1 and s_2 are odd but the number of even vertices is not equal to the number of odd vertices as in the case of $BDLG$s. The idea of designing a topology with this restriction came from the GCRs (generalized chordal rings) which are of the same form but are of degree 3. The method, plane tessellation technique, used to find the optimal graphs was the same one used for GCRs with differences in the details of application.

3.1.2 Broadcasting

In this section we will give a broadcast scheme that completes broadcasting in $d+2$ time units. We will also prove that $d+2$ is a lower bound on the broadcast time of BDLGs of diameter d. Hence, we prove that the broadcast time of bipartite double loop graphs of diameter d is $d+2$. The broadcast scheme is very similar to that of double loop graphs presented in [33]. Since the graph is vertex transitive, we will describe the broadcast scheme for the originator 0. Note that $s_2 > s_1$.

Notation 1. *We say that a vertex v receives the information through its edge s_1 if the information was sent to it by vertex $u = v + s_1$.*

Broadcast Scheme for Originator 0

1. Vertex 0 sends the message to the vertices to s_2, $-s_2$, $-s_1$, and s_1 in that order.

2. A vertex v, that receives the message through its s_2 edge, sends the message to the vertices $v - s_2$, $v - s_1$, and $v + s_1$ in that order.

3. A vertex v, that receives the message through its s_2 edge, sends the message to the vertices $v + s_2$, $v - s_1$, and $v + s_1$ in that order.

4. A vertex v, that receives the message through its $-s_1$ edge, sends the message to the vertex $v + s_1$.

5. A vertex v, that receives the message through its s_1 edge, sends the message to the vertex $v - s_1$.

The above message forwarding rules do not include a mechanism to check if the vertex to whom the message is being forwarded is already informed or not. As a result, there might be cases where two informed vertices might try to inform the same vertex, or an informed vertex tries to inform an informed vertex. According to the broadcast model, a call involves only one informed and one uninformed vertices. In order to decide on how to forward the message in these cases we augment the above algorithm with two other rules as follows:

1. If a vertex v is informed, and because of the above mentioned rules there is a new vertex v' at time t that wants to inform v again, two cases may arise: *Case 1*: v is still forwarding the message according to the rules 1 to 5. In this case we compare the edge d, through which v is going to forward the message at time $t+1$, to the edge s through which v would have received the message from v' at time t. . If $d = -s$, then v continues to forward the message without any changes. If $d = s$ and $|d| = s_2$ then v forwards the message to $v - s_1$ and $v + s_1$ in that order in the following two time units. If $d = -s$ and $|d| = s_1$, then v stops forwarding the message. If $|d| > |s|$, then v proceeds according to the rules 1 to 5 and assuming that it was informed at time t by the edge s. If $|d| < |s|$, then v does not change its forwarding schedule. *Case 2*: v has finished forwarding the message. In this case, v does not have to do anything,

and v' does not put a call on edge s and instead skips to the next neighbor on its message forwarding schedule.

2. If at a certain time unit t there are more than one vertices who want to inform a vertex v, the vertex that puts a call to inform v is chosen as follows. Let v_i where $1 \leq i \leq 4$ be the vertices who want to inform v via the edges e_i where $1 \leq i \leq 4$ (edges e_i are from v's perspective). We compare the absolute values of e_i and choose the edge corresponding to the minimum. If there are two edges with equal absolute values, then if the value is $|s_2|$, then v forwards the message via $-s_1$ and s_1 in that order. If the value is $|s_1|$ then v does not try to inform any other vertex.

The additional 2 rules follow from this observation. When at a certain time t a vertex $v + s_1$ or $v - s_1$ considers informing vertex v it is guaranteed that the vertices $v \pm s_2$ will be informed at time $t + 1$ without the intervention of vertex v. To see why, assume that $v - s_1$ considers informing vertex v at time t. Then the vertices $v - is_1$, $0 \leq i \leq \alpha$, are all informed at time t where α is chosen to be the smallest positive integer such that $v - \alpha s_1$ was informed by an s_2 or $-s_2$ edge. Without loss of generality assume that s_2 informed $v - \alpha s_1$, then $v - \alpha s_1$ informs the vertices $v - \alpha s_1 - s_2$, $v - (\alpha + 1)s_1$, and $v - (\alpha - 1)s_1$ in the following 3 time units. On its turn vertex $v - (\alpha - 1)s_1$ will need $\alpha - 2$ time units to inform $v - s_1$ at time $t - 1$. Therefore, we conclude that $v - \alpha s_1$ was informed at time $t - \alpha - 2$. $v - \alpha s_1$ informs $v - \alpha s_1 - s_2$ at time $t - \alpha - 1$, which informs $v - \alpha s_1 - 2s_2$, $v - (\alpha + 1)s_1 - s_2$,

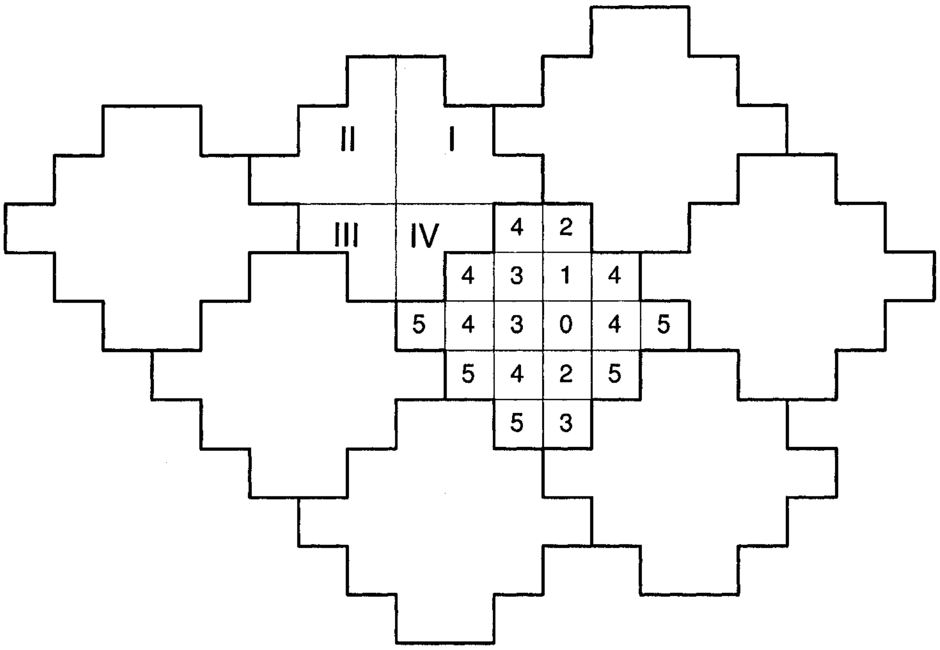

Figure 5: Shows the 4 different regions of the BDLG MDD. It also shows the time at which every square will receive the message if only steps 1 to 5 of the algorithm are considered.

and $v - (\alpha - 1)s_1 - s_2$ at times $t - \alpha$, $t - \alpha + 1$ and $t - \alpha + 2$ respectively. Finally, $v - (\alpha - 1)s_1 - s_2$ will need $\alpha - 1$ time units to inform $v - s_2$. As a conclusion we see that $v - s_2$ will be informed at time $t + 1$ without having v to forward the message to it.

Theorem 2. *The above scheme completes broadcasting in $d + 2$ time units.*

Proof. Let v be a vertex in the graph. Every vertex v is mapped to a unique square in an MDD, so we will use the MDD to visualize the path taken by the message to inform v. We divide the MDD into 4 quadrants as shown in Fig. 5. We can write v as $v = as_1 + bs_2$ and prove the theorem by taking 4 cases:

Case 1: $a \geq 0$ and $b \geq 0$, i.e. the vertex is in quadrant 1. Then from the shape of the MDD we know that $a + b \leq d - 1$. According to the above scheme, the first s_2 step will be taken at time 0. After that, at every time unit a new s_2 step can be taken. Therefore, completing the all the s_2 steps will take b time units. After taking the b^{th} s_2 step, two time units will pass and afterwards, the first s_1 step can be taken. Therefore, after taking all the s_2 steps, the a-many s_1 steps will be completed in $a+2$ time units. In total, taking the a and b steps will take at most $a+b+2$ time units. If $a = 0$ then informing v will take b time units, and if $b = 0$ then informing v will take $a + 3$ time units. Since $a + b \leq d - 1$, then all vertices in quadrant 1 can be informed in at most $d + 2$ time units.

Case 2: $a < 0$ and $b \geq 0$, i.e the vertex is in quadrant 2. Then from the shape of the MDD we get that $|a| + b <= d$. The first b steps of length s_2 can be completed in b time units. Then after waiting for one time unit, the $|a|$-many $-s_1$ steps can be completed in $|a|$ time units. Therefore, the total time to inform any vertex is $|a|+b+1$ which implies that informing the vertices in the second quadrant of the MDD takes at most $d + 1$ time units.

Case 3: $a < 0$ and $b < 0$, i.e. the vertex is in quadrant 3. From the shape of the MDD we know that $|a| + |b| <= d$. The first $-s_2$ step will be taken at time 1. Therefore, the $|b|$-many $-s_2$ steps will be completed in $|b| + 1$ time units. One time unit after the $|b|^{th}$ $-s_2$ step was taken a $-s_1$ step will be taken. So the $|a|^{th}$ $-s_1$ step will be taken at time $|b| + 1 + |a| + 1$. Therefore, the vertices in quadrant 3 will be informed in $|a|+|b|+2$ time units. Since $|a|+|b| <= d$ we get that the time to inform all the vertices takes at most $d + 2$ time units.

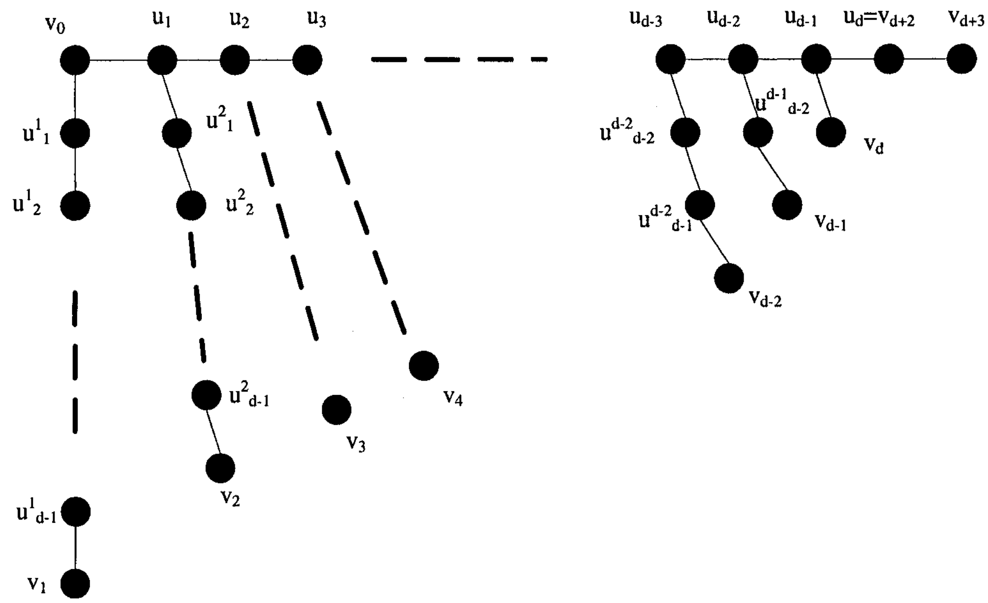

Figure 6: Shows the maximum number of vertices that can be at a distance d from the originator if the broadcast time is equal to $d+1$.

Case 4: $a \geq 0$ and $b < 0$, i.e. the vertex is in quadrant 4. From the shape of the MDD we know that $a + |b| < d - 1$. The $|b|$-many $-s_2$ steps will be completed in $|b| + 1$ time units. After that 2 time unit passes before the first s_1 step is taken. Hence, the vertices in quadrant 4 will be informed in at most $a + |b| + 3$ time units. Since $a + |b| < d - 1$, we get that the vertices in quadrant 4 will be informed in at most $d + 2$ time units. □

Now we want to prove that there is no other scheme that completes the broadcasting in a more efficient way. In order to achieve this we will prove an auxiliary result stated in the following theorem.

Theorem 3. *If a graph $G = (V, E)$ has more than $d + 2$ vertices at a distance d from another vertex v_0, then the broadcast time, $b(G)$, of G satisfies the following* *inequality:* $b(G) \geq d + 2$.

Proof. We will prove this by contradiction. Assume that in a graph G there are $d+3$ vertices $v_1, v_2, \cdots, v_{d+3}$ such that the distance between v_0 and v_i is equal to d, the diameter of G, i.e. $d(v_0, v_i) = d$,$1 \leq i \leq d+3$. Since there are more than 2 vertices at distance d from v_0, from [19] we conclude that $b(G) \geq d+1$. Now assume that $b(G) = d+1$, we will prove that this will lead to a contradiction with the fact that there are more than $d+3$ vertices at distance d for v_0.

Assuming that $b(G) = d+1$, implies that there is an optimal broadcast scheme which informs all vertices in at most $d+1$ time units. Let P_i be the path that is taken by the message to inform vertex v_i in this optimal broadcast scheme where the originator is v_0. Since $d(v_0, v_i) = d$, then the length of the paths P_i are greater or equal to d. But since $b(G) = d+1$ then they are also less than or equal to $d+1$. Assume that there are two paths P_i and P_j of length $d+1$. These two paths have the originator vertex in common. Since there is a common vertex then it is impossible that both v_i and v_j were informed at $d+1$, one of them will be informed at $d+2$. Therefore, we conclude that there is no more than one path of length $d+1$. Hence, we can have two cases:

There is exactly *one path* of length $d+1$ and the rest are each of length d (Fig. 6). Without loss of generality let $P_{d+3} = [v_0, u_1, \cdots, u_d, v_{d+3}]$ be this path. Let $P_i[v_0, u_1^i, \cdots, u_{d-1}^i, v_i]$, $1 \leq i \leq d+2$ be the rest of the paths. All paths have the originator in common. Assume that all of the vertices u_1^i were different from each other and different from u_1 then the broadcast time will be more than $d+1$. The only way that the broadcast time will be $d+1$ is when there is at most one vertex $u_1^{i'} \neq u_1$ and the rest u_1^i, $i \neq i'$, are the same as u_1. Let path P_1 be the path such that $u_1^1 \neq u_1$. None of the paths P_j , $2 \leq j \leq d+2$ can have a vertex $u_2^j = u_2^1$,otherwise the broadcast time of the graph will be more than $d+1$. So we conclude that $u_2^j \neq u_2^1$ for $2 \leq j \leq d+2$. Moreover, applying the same argument, as it was done above, and considering paths P_j, where $2 \leq j \leq d+2$, and the path P_{d+3} that have $u_1^j = u_1$ we conclude that there is at most one path P_k that does not have $u_2^k \neq u_2$. Repeating the same arguments $d-1$ times we conclude that only v_0 and u_j, $1 \leq j \leq d-1$, can be branching vertices. A branching vertex is a vertex where two paths having a common part branch into two different directions. We also argued that at every branching vertex u_j on P_{d+3} there could be only one new path. It could be possible that u_d on P_{d+3} is at a distance d from v_0. If we assume that u_d coincides with v_{d+2} we can see that v_{d+2} is informed at time d. So we are left with $d+1$ paths that should branch off P_{d+3} to inform the $d+1$ vertices $v_1, \cdots, v_{d+1}$. From the above presented argument we deduced that there are d branching vertices while there are $d+1$ paths. By the pigeon hole principle we conclude that there will be 1 branching point that will have two paths branching off P_{d+3}, which implies that the broadcast time is greater than $d+1$ which contradicts the initial assumption. Hence, our theorem is proved.

There are *no paths* of length $d+1$, which implies all of them are of length d. Applying the arguments of the previous case, it can be seen that if there is a path P_j such that all the other paths branch off P_j then there could be only $d+1$ vertices at distance d from the originator v_0 under the condition of $b(G) = d+1$. If such a path P_j cannot be found the number of vertices that can be at distance d from the originator will be even less than $d+1$.

$\square$

We know that the BDLG of diameter d has $2d-1$ vertices at distance d from vertex 0. Therefore, using Theorem 3 we get that $b(BDLG) \geq d+2$. Given the upper bound by the broadcast scheme described above we conclude that $b(BDLG) = d+2$.

3.2 Triple Loop Graphs

In this section we will study the undirected triple loop graphs. These are circulant graphs of degree 6 and are a special case of the multiloop graphs defined in Chapter 2. For convenience we repeat the definition of the triple loop graphs.

Definition 5. *A triple loop graph (undirected circulants of degree 6) is the graph $G = (V, E)$, denoted by $C_N(\pm s_1, \pm s_2, \pm s_3)$, defined on the set of vertices $V = \{0, 1, \cdots, N-1\}$, where $|V| = N$, and the set of edges E such that a vertex i is connected to vertices $i \pm s_1, i \pm s_2, i \pm s_3 (modN)$.*

The special case where $s_3 = -(s_1+s_2)$ was studied using the tessellation technique. The broadcast time of the optimal graph with diameter d was proved to be $d+3$ [33]. In this section we we will study the broadcast problem in triple loop graphs where $s_1 = 1$ while s_2 and s_3 can have arbitrary values. The optimal graph of this family was obtained by E. Monakhova [35]. Her results concerning the optimal triple loop graphs will be presented in subsection 3.2.1 .

3.2.1 Structure of Optimal Triple Loop Graphs

In [35] the following class of triple loop graphs was studied: $s_1 = 1$ and $1 < s_2 < s_3 \leq \lfloor \frac{N}{2} \rfloor$. Given a diameter d, the maximum number of vertices N that a triple loop graph can have was calculated together with the values of s_2 and s_3.

Theorem 4. *The maximum order of a triple order graph with diameter $d > 1$ is*

$$N = \begin{cases} \frac{32}{27}d^3 + \frac{16}{9}d^2 + 2d + 1 & for \quad d \equiv 0(mod\ 3) \\ 32[\frac{d}{3}]^3 + 48[\frac{d}{3}]^2 + 30[\frac{d}{3}] + 7 & for \quad d \equiv 1(mod\ 3) \\ 32[\frac{d}{3}]^3 + 80[\frac{d}{3}]^2 + 70[\frac{d}{3}] + 21 & for \quad d \equiv 2(mod\ 3) \end{cases} \tag{4}$$

This bound can be achieved by taking the following values of s_1 and s_2.

$$(s_2, s_3) = \begin{cases} (\frac{8}{9}d^2 + \frac{2}{3}d, \frac{8}{9}d^2 2d + 2) & for \quad d \equiv 0(mod\ 3) \\ (8[\frac{d}{3}]^2 + 6[\frac{d}{3}] + 2, 8[\frac{d}{3}]^2 + 8[\frac{d}{3}] + 4) & for \quad d \equiv 1(mod\ 3) \\ (8[\frac{d}{3}]^2 + 10[\frac{d}{3}] + 4, 8[\frac{d}{3}]^2 + 14[\frac{d}{3}] + 6) & for \quad d \equiv 2(mod\ 3) \end{cases} \tag{5}$$

Writing d as $3n + \alpha$ where $\alpha = 1, 2$, or 3 we get the following result.

$$N = \begin{cases} 32n^3 + 16n^2 + 6n + 1 & for \quad r = 3n, n \geq 1, \\ 32n^3 + 48n^2 + 30n + 7 & for \quad r = 3n + 1, n \geq 0, \\ 32n^3 + 80n^2 + 70n + 21 & for \quad r = 3n + 2, n \geq 0, \end{cases} \tag{6}$$

$$(s_2, s_3) = \begin{cases} (8n^2 + 2n, 8n^2 + 6n + 2) & for \quad d = 3n \\ (8n^2 + 6n + 2, 8n^2 + 10n + 4) & for \quad d = 3n + 1 \\ (8n^2 + 10n + 4, 8n^2 + 14n + 6) & for \quad d = 3n + 2 \end{cases} \tag{7}$$

3.2.2 Broadcasting in Optimal Triple Loop Graphs

We will prove that $d+2$ is a lower bound for the broadcast time of optimal triple loop graphs of diameter d. First we will study some structural properties of the graph, then using these properties we will calculate the lower bound. Triple loop graphs are vertex transitive, therefore it is enough to study the broadcast problem where the originator is vertex 0.

Structural Properties

Define the quantity Δ as follows: $\Delta = s_3 - s_2$. The values of s_2 and s_3 are different depending on the value of d mod 3. But note that the value of Δ is always $4n + 2$. We can write N and s_3 in terms of Δ and n as follows: $N = (4n+\beta)s_3+\Delta+1$, where $\beta = -1, 1$, and 3 for $d = 3n, 3n+1$, and $3n+2$ respectively. Note that $\beta = 2\alpha - 1$. The value of s_3 written in terms of Δ is $s_3 = (2n+\alpha)\Delta + 2n + 2$.

We will find the shortest distance between two vertices in the optimal triple loop graphs (TLG). Since the graph is vertex transitive we will be considering only the shortest distance between vertex 0 and all the other vertices. In particular, we are interested in those vertices whose distance from vertex 0 is equal to the diameter, d, of the optimal TLGs. Any vertex can be reached from vertex 0 by taking a series of s_i steps (edges) where $1 \leq i \leq 3$. Let a_i be the number of steps taken along the edge s_i. Note that a_i can be positive or negative, a negative a_i implies that $|a_i|$ steps were taken along $-s_i$. To get to vertex v from 0 the following condition should be satisfied $(a_1s_1 + a_2s_2 + a_3s_3) \bmod N = v$. There could be more than one triplet (a_1, a_2, a_3) that satisfies the above condition. A triplet corresponds to a set of paths, the exact steps of the paths are not specified but the number of steps along each step size s_i is specified. The length of the path represented by the triplet (a, b, c) joining 0 to v is $|a_1| + |a_2| + |a_3|$. It should be noted that if a triplet (a'_1, a'_2, a'_3) satisfies the condition $(a'_1s_1 + a'_2s_2 + a'_3s_3) \bmod N = v$, then it does not follow that $l = |a'_1| + |a'_2| + |a'_3|$ is the shortest distance between 0 and v because the paths specified by the triplet (a'_1, a'_2, a'_3) might not be the shortest. Therefore, it is not straightforward to find the shortest distance since one has to find all the possible triplets (a_1, a_2, a_3) and chose the one with the shortest value of $|a_1| + |a_2| + |a_3|$.

Now we will define the idea of a representation of a vertex. The triplet (a_1, a_2, a_3) will be considered to be a representation of v in the space where the basis vectors are s_i, $1 \leq i \leq 3$. It is also possible to calculate v as the sum $(k_3 s_3 + k_2 \Delta + k_1 s_1) mod N$. Hence, the triplet (k_1, k_2, k_3) is another representation of v but in a different space whose basis vectors are s_1, Δ, and s_3. The only difference between the two spaces is the presence of Δ or s_2, hence we will denote the first space by $s_2 - space$ and the second by the $\Delta - space$. The two ways of representing v are equivalent because for every representation (a_1, a_2, a_3) of v in the $s_2 - space$ there is a unique representation (k_1, k_2, k_3) in the $\Delta - space$ such that $(k_1 s_1 + k_2 \Delta + k_3 s_3) \bmod N = v$ and vice versa. The triplets are related as follows: $k_1 = a_1$, $a_2 = -k_2$, and $a_3 = k_3 + k_2$. In order to list all the triplets (a_1, a_2, a_3) corresponding to a vertex $v = a_1 s_1 + a_2 s_2 + a_3 s_3$ in the $s_2 - space$ we will first list all the possible triplets (k_1, k_2, k_3) in the $\Delta - space$ and then convert them to the corresponding representation in the $s_2 - space$. It turns out that for a vertex v, it is more intuitive to list the possible triplets in the $\Delta - space$. The next paragraph explains why it is so.

The optimal TLG has all its vertices distributed on a loop and every vertex has additional chords connecting it to other vertices on the loop. We will divide the vertices of the loop into different sets by using the chords s_2 and s_3. Starting from vertex 0 one can take s_3 or $-s_3$ steps to reach the vertices is_3 where $-(2n-1) \leq i \leq 2n$. A set S_i will be defined to be the vertices v such that $(i-1)s_3 \leq v < is_3$ where

$i \neq 0$ and $-(2n-1) \leq i \leq 2n$. Note that vertex 0 is in both sets S_1 and S_{-1}. In addition to the vertices in these sets there are $4n+4$ vertices on the loop between the sets S_{2n} and $S_{-(2n-1)}$, let S_0 be the set containing these $4n+4$ vertices. Any vertex v, except 0, will be only in one of the sets S_i , $-(2n-1) \leq i \leq 2n$. If v is in set S_k then we can list all the triplets (a, b, c) such that $(cs_3 + b\Delta + a) mod N = v$. Note that for the optimal triple loop graph $s_1 = 1$ so in what follows we will use 1 instead of s_1 and will call this step *unit step.* A negative unit step is a unit step that connects a vertex i to $(i-1) \bmod N$. To get to a vertex in set S_k one can start from 0 and take a series of steps s_3 and get to vertex $(k-1)s_3$ and then take a series of Δ steps followed by as many unit steps as it takes to get to v. Another option is to take a series of $-s_3$ steps followed by a $-\Delta$ step and a negative unit step to get to vertex $(k-1)s_3$, afterwards one has to take some Δ and units steps. Equivalently, one can go first to ks_3 in two different ways (similar to the above case) and then take the necessary Δ and units steps. Since we are interested in the broadcast time we will list the representations in the $\Delta - space$ of specific vertices which are at a distance of d from 0.

Consider the vertex v such that $v = (is_3 + n\Delta + (n-i+\alpha)) mod N$ where $i < n - \alpha$. One triplet representing v in the $\Delta - space$ can be immediately written as follows:

$$(n - i + \alpha, n, i) \tag{8}$$

We will list some of the representations of v in a systematic way such that it will be obvious that the length of the corresponding paths is increasing. The guidelines to list the representations are to first take $\pm s_3$ steps to get as close as possible to v, there are two such positions each on one side of v. Then from each position, reached by the $\pm s_3$ steps, take $\pm\Delta$ steps to get to a vertex as close to the v as possible. Again two such positions exist, one on each side of v. Variations on these steps can be introduced, for example, by taking $\pm s_3$ to get to the second nearest position to v from both sides. Or by taking $\pm\Delta$ steps to get to the second nearest position to v again from both sides. These two approaches can be combined or even modified so that the $\pm s_3$ are taken to get to the j^{th} nearest position to v (from either sides) and then $\pm\Delta$ are taken to get to the k^{th} nearest position to v and finally the required number of unit steps are taken to get to v. It is obvious that as k and/or j is increased the length of the path increases. Couple of examples will be enough to convince someone. In what follows when we use the term "top side" of a vertex v, where $v < \lceil N/2 \rceil$, we refer to that side which has vertices u such that $u < v$, while the bottom side refers to the side with vertices u such that $u > v$. If $v > \lceil N/2 \rceil$ top refers to the side with vertices u such that $u > v$ and bottom refers to the side with $u < v$.

A second representation of v can be obtained by noticing that it is possible to reach is_3 by taking a certain number of $-s_3$ steps with one $-\Delta$ and one negative unit step. After reaching is_3 it is possible to take n Δ steps and $(n - \alpha + 1)$ units steps. Therefore, $v = (-(2n + \alpha - 1)s_3 - (2n + \alpha - i)s_3 - \Delta - 1 + n\Delta + (n - i + \alpha)) \bmod N$ which corresponds to the triplet

$$(n - i + \alpha - 1, n - 1, -(4n + 2\alpha - i - 1))$$

A third representation can be obtained from the first one with the difference that one more Δ step can be taken to get to the closest vertex to v from the bottom. Hence, v can be written as $v = (is_3 + (n+1)\Delta - (3n + i - 2 - \alpha))\text{mod } N$. The triplet corresponding to this representation will be:

$$(-[3n + i - 2 - \alpha], n + 1, i)$$

A fourth representation can be obtained from the third one with the difference that at the beginning $-s_3$ steps are taken to get to is_3 as it was done for the second representation. Therefore, $v = (-(2n+\alpha-1)s_3 - (2n+\alpha-i)s_3 - \Delta - 1 + (n+1)\Delta - (3n + i - 2 - \alpha))\text{mod } N$ and the corresponding triplet is:

$$(-[3n + i - 2 - \alpha], n, -(4n + 2\alpha - i - 1))$$

In the following 4 representation we will first get to $(i+1)s_3$ and then take the necessary $-\Delta$ to get to the closest two vertices on both sides and then take the required unit steps. Without any explanation we will present all 4 for them.

$$\begin{aligned}
v &= ((i+1)s_3 - (n+\alpha)\Delta - (n + i - \alpha + 2))\text{mod } N && (9)\\
&= ((i+1)s_3 - (n+\alpha+1)\Delta + (3n - i + \alpha))\text{mod } N && (10)\\
&= (-(4n + 2\alpha - 2 - i)s_3 - (n+\alpha+1)\Delta - (n + i - \alpha + 3))\text{mod } N && (11)\\
&= (-(4n + 2\alpha - 2 - i)s_3 - (n+\alpha+2)\Delta + (3n - i + \alpha - 1))\text{mod } N && (12)
\end{aligned}$$

Continuing in this fashion one can get other representations of v and the list is infinite since one can go around the loop several times before getting to v. But there is no need to list them since it can be seen that all of these paths will be much longer than d. In particular if the Δ steps are taken to get to the k^{th} nearest position to v, where

$k \geq 2$ then the number of unit steps needed will be considerably numerous hence the length of the path will be larger than d. Similarly, as less $\pm s_3$ steps are taken to get to the j^{th} closest vertex to v more Δ steps will be needed. For all of the vertices that we considered the length of the path increased as additional representations were obtained by the above mentioned guidelines. The length of the path corresponding to each of the above mentioned 8 representations is calculated and the results are summarized in Table 1. It can be easily observed that the first representation (first row of Table 1) was corresponding to the shortest path.

Triplet	**Length**
$(n-i+\alpha, n, i)$	$3n+\alpha=d$
$(n-i+\alpha-1, n-1, -(4n+2\alpha-i-1))$	$5n-2i-2+2\alpha$
$(-[3n+i-2-\alpha], n+1, i)$	$5n+2i-\alpha$
$(-[3n+i-2-\alpha], n, -(4n+2\alpha-i-1))$	$7n+\alpha-3$
$(-(n+i-\alpha+2), -(n+\alpha), i+1)$	$3n+\alpha+1=d+1$
$((3n-i+\alpha), -(n+\alpha+1), i+1)$	$5n+2\alpha-2i+1$
$(-(n+i-\alpha+3), -(n+\alpha+1), -(4n+2\alpha-2-i))$	$7n+3\alpha+3$
$((3n-i+\alpha-1), -(n+\alpha+2), -(4n+2\alpha-2-i))$	$9n+5\alpha-2i+3$

Table 1: Displays 8 representations of v and the length of the corresponding path.

Now without any explanation we will list some vertices which are at distance d from the origin, they are shown in table 2. We are not giving a complete list of all the vertices at distance d. In order to prove the required lower bound on the broadcast time of TLGs we need to find at least $d+3$ vertices, and that is what we are trying to do below.

From what was presented in Table 2, it can be seen that the number of vertices at distance d from the originator vertex 0 is much more than $d+3$, by using theorem 3 we conclude that $d+2$ is a lower bound on the broadcast time of optimal triple loop

Vertex v	
$v = (i+1)s_3 - (n+\alpha)\Delta - (n+i+1-\alpha)$	$i < n-\alpha$
$v = is_3 + n\Delta - (n-i+\alpha)$	$i < n-\alpha$
$v = (i+1)s_3 - (n+\alpha+1)\Delta - (n+i-1-\alpha)$	$i < n-\alpha$
$v = is_3 + n\Delta - (i-n-\alpha+1)$	$i > n+\alpha$
$v = (i+1)s_3 - (n+\alpha+1)\Delta - (3n-i-1+\alpha)$	$i > n+\alpha$
$v = is_3 + n\Delta + (i-n-\alpha+2)$	$i > n+\alpha$
$v = (i+1)s_3 - (n+\alpha)\Delta - (3n-i-1+\alpha)$	$i > n+\alpha$
$v = is_3 + (n+1)\Delta - (i-n)$	$i < n-\alpha$
$v = is_3 + (n+1)\Delta + (i-n+2)$	$i < n-\alpha$

Table 2: A list of vertices that are at distance d from vertex 0.

graphs.

3.3 A Broadcast Algorithm in MultiLoop Graphs

In this section we will give an upper bound on the broadcast time of the undirected multiloop graphs of degree $2k$. Let the steps s_i, $1 \leq i \leq k$, be ordered such that $s_1 \leq s_2 \leq \cdots \leq s_k$. First we will consider the case where $k = 3$ i.e. the general triple loop graph. We will prove that an upper bound on the broadcast time of triple loop graphs is $d + 2k - 1 = d + 5$ by describing an algorithm that guarantees broadcasting in $d + 5$ time units. After considering the triple loop graph, we will generalize the results to the case where there are k steps, i.e. degree $2k$ multiloop graphs.

3.3.1 Upper Bound on Broadcast Time of Triple Loop Graphs

First we will describe a broadcast algorithm that informs all the vertices of any triple loop graph in less than $d + 5$ time units. Then we will prove the correctness of the algorithm.

The Broadcast Algorithm

The vertices of the triple loop graph are numbered from 0 to $N-1$. We will adopt the following to name the edges of a vertex i: an edge will be called s_i if it leads to a new vertex j such that $j = (i + s_i) \bmod N$, similarly it will be called $-s_i$ if it leads to a new vertex j such that $j = (i - s_i) \bmod N$. Since the graph is vertex transitive, it does not matter which vertex is the originator so without loss of generality we will assume that vertex 0 is the originator. Given that $s_1 \leq s_2 \leq s_3$ we suggest the following algorithm:

Broadcast Algorithm for vertex 0

1. Vertex 0 first sends the message via edge s_3,$-s_3$,s_2,$-s_2$, s_1, and $-s_1$ in that order.

2. If a vertex receives the message through edge $-s_3$, it forwards it via s_3, s_2, $-s_2$, s_1, and $-s_1$ in that order.

3. If a vertex receives the message through edge s_3, it forwards it via $-s_3$, $-s_2$, $+s_2$, $-s_1$, and s_1 in that order.

4. If a vertex receives the message through edge $-s_2$, it forwards it via s_2, s_1, and $-s_1$ in that order.

5. If a vertex receives the message through edge s_2, it forwards it via $-s_2$, $-s_1$, and s_1 in that order.

6. If a vertex receives the message through the edge s_1 it forwards it via the edge $-s_1$.

7. If a vertex receives the message through the edge $-s_1$ it forwards it via the edge s_1.

As it was done for the broadcast algorithm of bipartite double loop graphs we have to augment the above algorithm with more rules so that the the rules of the classical broadcast model are respected, namely, that every call involves an informed vertex and one of its uninformed neighbors.

1. For the case where there is a vertex v is informed and has finished informing its neighbors, and because of the above mentioned rules if there is a new vertex v' at time t that wants to inform v again, the following is done: v' does not make the call to inform v and skips to the next vertex on its schedule.

2. If a vertex v is informed but still is in the process of informing its neighbors, and there is another vertex v' at time t that according to the rules 1 to 7 is considering to inform v, then the following happens. Let d be the edge through which v was going to inform its neighbor at time $t+1$. Let s be the edge through which v would have received the message from v' had it not been informed. If $|d| > |s|$ or $d = -s$, then v does not change its message forwarding policy, and v' does not inform v. If $|d| < |s|$, then v continues its message forwarding policy at time $t + 1$ based on the rules 1 to 7 and assuming that it was informed by v' at time t, and v' does not inform v and skips to the next vertex on its schedule.

If $s = d$ then v stops forwarding the message, and v' does not inform v at time t.

3. If according to the rules 1 to 7, two or more vertices try to inform another vertex v through the edges e_i then the following happens. We choose the edge e_j such that $|e_j| < |e_i|$ for all values of $i \neq j$ and use that edge to inform v. If v is already informed, then we get back to the previous case where vertex $v' = v + e_j$. If there are two edges e_j and $-e_j$ such that $|e_j| < |e_i|$ for all values of $i \neq j$, then either one of them is chosen to inform v.

We need to justify the additional steps. If a vertex v is informed it forwards the message to some of its neighbors such that when it is done with forwarding the message, all the vertices $v \pm s_1$, $v \pm s_2$, and $v \pm s_3$ are informed. Steps 1 through 7 make sure that this result is obtained. When a vertex is informed and idle it means that it finished forwarding the message to all of its uninformed neighbors, so there is nothing else to be done. To justify the steps 2 and 3 of the additional set of steps, we present the following theorem.

Theorem 5. *When a vertex v receives a message via s_i at time t then the vertices $v \pm s_j$, where $|s_j| > |s_i|$, will be informed at the latest by the time $t + 1$.*

Proof. Assume that vertex v is informed at time t and $v + s_k$ where $|s_k| > |s_i|$ is not informed at $t + 1$. Since v received the message through the edge s_i then the vertex $u = v + s_i$ is the sender of the message. Vertex u on its turn received the message from u' through an edge s_i' such that $|s_i'| \geq |s_i|$. Along the path of the vertices which informed each other ending in u and v there should be a vertex that got informed via an edge s'_k such that $|s'_k| = |s_k|$. Let this vertex be called w. Let Σ be the series of calls that starting at w informed v in T time units by following a path P of length L, path P takes a vertex w to v. Since the graph is vertex transitive it can be concluded that path P takes a vertex i to $i + v - w$. If $s_k = -s'_k$ then let w_1 and w_2 be two vertices such that $w_1 - s'_k = w$ and $w - s'_k = w_2$. Clearly since w was informed by its s'_k edge then w_1 was the vertex informing w. After being informed, w first informs w_2 by its $-s'_k$ edge. Since w and w_2 were informed by their s'_k edge, then the same series of calls were done by both of them. After being informed it took T time units for w to inform v. Since w_2 is informed one time unit after w is informed then it will take $T + 1$ time units before $w_2 + v - w = (w - s'_k) + v - w$ is informed. Note that $w - s'_k + v - w = v - s'_k$, but since $s_k = -s'_k$ we get that the vertex $v + s_k$ will be informed by the time $T + 1$. This contradicts the initial assumption.

If $s'_k = s_k$ then let w_1 be the vertex such that $w_1 - s'_k = w$. So w_1 is the one who informed w. But we do not know through which vertex w_1 was informed, this implies that w_1 might not inform the vertex $w_1 + v - w$ T time units after being informed. The order in which w uses its edges depends on the edge through which it was informed. If w was informed by an edge s_m such that $s_k s_m > 0$ (i.e. both edges had the same sign) then after w_1 is informed and after it forwards the message through $-s'_k$ the order of the edges that w_1 will use will be the same at that of w. Hence T time units after w_1 is informed the vertex $w_1 + v - w$ will be informed. If $s_k s_m < 0$ (i.e. the edges have different signs) then after w_1 sends the message via $-s'_k$ the order of the remaining edges used by w_1 will not be the same as that of w. There can be a maximum of 2 time units difference between the time at which w_1 forwards the message over an edge s_i and the time at which w forwards the message along the same edge s_i. Given that w_1 is informed one time unit before w and adding the possible 2 time unit difference we conclude that the vertex $w_1 + v - w = w + s'_k + v - w = v + s'_k = v + s_k$ will be informed at most one time unit after v is informed. This again contradicts the initial assumption. $\square$

Theorem 5 explains why when an informed but active vertex, v, is attempted to be informed again, it uses this fact to modify its forwarding schedule. If a vertex was considering to inform v via the edge s at time t then we are sure that all the neighbors $v + e$ of v where $|e| > |s|$ are all informed by time $t + 1$ the latest. For the case where v is attempted to be informed by several edges simultaneously the same reasoning explains why the edge with greatest absolute value is chosen.

Broadcast Time

An upper bound on the broadcast time can be obtained by calculating the time needed by the above algorithm to inform all the vertices of the graph. A vertex v in a triple loop graph can be written as $v = as_1 + bs_2 + cs_3$. We are not concerned with calculating the values of a, b, and c that give the shortest distance $d(0, v) = |a|+|b|+|c|$ between vertex 0 and v. The existence of such a shortest distance is sufficient for our purposes. It is clear that every vertex v can be written as $v = as_1 + bs_2 + cs_3$ such that $|a| + |b| + |c| \leq d$ where d is the diameter of the graph. According to the above algorithm the originator, vertex 0, first sends the message via s_3 if $c > 0$ then after c time units all the necessary s_3 will have been taken. If $c < 0$ then during the first time unit vertex 0 will notify vertex s_3 but in the second time unit vertex $-s_3$ will be informed. In the following c time units all the necessary $-s_3$ will have been taken. The cs_3 vertex will spend one time unit to send the information via s_3 or $-s_3$ depending on the sign of c. Then vertex cs_3 will send the message first via s_2 then via $-s_2$ or vice versa depending on the sign of c. Therefore, the maximum delay is 2 time units before the first $|s_2|$ step is taken. An example of how this case may arise is the following: $c > 0$ and $b < 0$. Similarly the maximum delay before an $|s_1|$ step is taken is 2. Therefore, the total time needed to inform vertex v is less than or equal to $1 + |c| + 2 + |b| + 2 + |a| = 5 + |a| + |b| + |c|$. Since $|a| + |b| + |c| \leq d$ then the maximum time to inform any vertex v is $d + 5$.

Theorem 6. *The broadcast time of the above algorithm in a triple loop graph with diameter d is at most $d + 5$.*

Proof. Assume that there is a vertex $v = as_1 + bs_2 + cs_3$ with $|a| + |b| + |c| \leq d$ such that v is not informed at time $d + 5$. According to the algorithm cs_3 will be informed in c time units if $c > 0$ or $|c| + 1$ time units otherwise. After cs_3 is informed it will take $|b| + 1$ additional time units to inform $cs_3 + bs_2$ if b and c are positive or b and c are negative, otherwise it will take $|b| + 2$ additional time steps to inform $cs_3 + bs_2$. Similarly, once $cs_3 + bs_2$ is informed it will take $|a| + 1$ time units if $a, b > 0$ or $a, b < 0$. Otherwise it takes $|a| + 2$ time units. Summing the maximums we get that the maximum time needed to inform $as_1 + bs_2 + cs_3$ is less than or equal to

$|a|+1+|b|+2+|c|+2 = 5+|a|+|b|+|c|$. According to the assumption at time $d+5$ v was not informed which implies that $d+5 < 5+|a|+|b|+|c|$. This implies that $|a|+|b|+|c| > d$ where d is the diameter of the graph, which is a contradiction. $\square$

3.3.2 Upper Bound on Broadcast Time in Multiloop Graphs

For the general multiloop graph we can generalize the result of the previous section easily. For a multiloop graph G with diameter d and steps $\pm s_i$, $1 \leq i \leq k$, an upper bound on the broadcast time is $b(G) \leq d + 2k - 1$. The algorithm that provides us with this bound will be presented later, but first we like to make the following observations. As k becomes in the order of $O(N)$, where N is the number of vertices, then our upper bound becomes on the order of $O(N)$ which is somehow bad. For example in the case of $k = \lfloor \frac{N}{2} \rfloor$, the multiloop graph becomes a complete graph whose broadcast time is known to be $\lceil log(N) \rceil$ which is drastically different than our result which is $b(G) = N$. Hence, it can be seen that our algorithm does not provide an efficient scheme when the number of steps is very high. Having said this, we want to note that the reason behind the interest in multiloop graphs is their similarity to the simple loop structure without its disadvantages which are long diameter and low fault tolerance. So the object of interest is the set of graphs with a constant number of steps.

The algorithm generalizes the first 6 steps of the one for the triple loop graphs. As before it is assumed that step sizes are labeled such that $s_k \geq s_{k-1} \geq \cdots \geq s_1$.

Broadcast Algorithm for the Originator 0

1. Vertex 0 (the originator) forwards the message through the edges in this order: s_k, $-s_k, \cdots$, s_1, and $-s_1$.

2. A vertex receiving a message via edge s_i where $s_i > 0$ forwards the message in the following order $-s_i$, s_{i-1}, $-s_{i-1}, \cdots$, s_1, and $-s_1$.

3. A vertex receiving a message via edge $-s_i$ where $s_i > 0$ forwards the message in the following order s_i, $-s_{i-1}$, $s_{i-1}, \cdots$, $-s_1$, and s_1.

The additional rules are exactly the same as for the broadcast algorithm for triple loop graphs.

The proof of correctness follows from theorem 5 which was proved in the general case without assuming that there are only 3 edges.

Theorem 7. *The above algorithm informs all vertices of the multiloop graph in at most $d + 2k - 1$ time units.*

Proof. Consider that there is a vertex $v = a_k s_k + \cdots + a_1 s_1$ that was not informed by the time $d + 2k - 1$, as before $|a_k| + \cdots + |a_1|$ is assumed to be the shortest distance. Following the same logic as above we can calculate the maximum time needed to inform v according to the above mentioned algorithm. For the edge s_k one time unit might be spent waiting, then for each step s_i there can be a maximum wait of 2 time units. Hence, the maximum time will be $2k-1+|a_k|+\cdots+|a_1|$. If this time is greater than $d + 2k - 1$ we conclude that $|a_k| + \cdots + |a_1| > d$ which is a contradiction. $\square$

Chapter 4

Simulations

Finding the minimum broadcast time of a given graph is in general an NP-complete problem. That is why there are many attempts in the literature to find near optimal polynomial broadcast algorithms. In this chapter we will consider only one such algorithm developed by Harutyunyan and Bin [21]. This heuristic was tested on several regular and arbitrary graphs and produced good results. We will be interested in testing their heuristic on the different multiloop graphs presented in this thesis. Another reason for our interest in the simulation is to find get pointers if the lower bound on the broadcast time of triple loop graphs can be improved or not. In case the simulations yield a broadcast time of $d + 2$, we can immediately conclude that $d + 2$ is a tight lower bound. Otherwise, there is a good reason to look for a better lower bound. In the first section, we will describe the heuristic and in the following section we will present the obtained results.

4.1 The Heuristic

The basic idea of this algorithm is to find a matching between the set of informed and uninformed vertices at each round and then inform the matched vertices. The algorithm does a BFS (breadth first search) from the informed vertices at every round and as a result the uninformed vertices are labeled with the shortest distance from an informed vertex. These are used to define parent child relationship and consequently are used to assign a weight to each uninformed vertex. Finally a minimum number weight matching [21] is done to choose which vertex is to be informed next.

Let the set $I(t)$ represent those informed vertices that have at least one uninformed neighbor at time t. Let $U(t)$ be the set of those uninformed vertices at time t that have an informed neighbor v such that $v \in I(t)$. At every round t, every uninformed vertex v will be assigned a weight $EB(v, t)$ after doing a BFS and some calculations. After defining the necessary terms we can provide the pseudocode of the heuristic called Tree Based Algorithm (TBA) [21]:

TBA

1. Initialize $I(0)$ such that it contains the originator of the broadcasting.

2. Compute $EB(v, t)$ for every uninformed vertex.

3. Do a matching between $I(t)$ and $U(t)$ by applying the minimum number weight matching of [21].

4. Mark all the matched vertices of $U(t)$ as informed, and update $I(t)$ to obtain

$I(t+1)$.

5. Continue steps 2,3, and 4 until $I(t+1)$ is empty. In that case t will be the broadcast time.

4.2 Simulation Results

Tables 3, 4, 5, and 6 summarize the obtained results. Table 3 shows that the simulation results of the heuristic on optimal generalized chordal graphs are the same as the theoretical values. For graphs with diameter d when d is even the broadcast time is $d+1$ which agrees with our simulations. When d is odd the exact broadcast time is $d+2$ which is again the same as the simulation results. For optimal double loop graphs, the broadcast time was found to be $d+2$, here again the simulation results are the same as the theoretical one. The same scenario happens with the optimal triple loop graphs, where $s_3 = -(s_1+s_2)$. The theoretical broadcast time $d+3$ is the same as the simulation results. The third table presents the simulation results of the heuristic on optimal triple loop graphs that were studied in section 3.2. We had proved that a lower bound on the broadcast time is $d+2$ and an upper bound is $d+5$. The simulation results turn out to be $d+4$ in all of the cases that were tried. For the bipartite double loop graphs (Table 6) the results of the heuristics showed that the tested algorithm complete broadcasting in the optimal time which is $d+2$.

GCR_n (1, -1, 3D)			GCR_n(1, -1, 3D+1)		
D	T_{Opt}	$T_{Heuristic}$	D	T_{Opt}	$T_{Heuristic}$
11	13	13	10	11	11
15	17	17	12	13	13
17	19	19	16	17	17
29	31	31	18	19	19
39	41	41	20	21	21
49	51	51	40	41	41
59	61	61	50	51	51
69	71	71	60	61	61
79	81	81	80	81	81
89	91	91	90	91	91
99	101	101	100	101	101

Table 3: Simulation results of the broadcast heuristic in GCR_n. T_{Opt} refers to the theoretically calculated broadcast time. $T_{Heuristic}$ refers to the broadcast time obtained by the heuristic of [21]. D is the diameter of the graph.

Opt. Double Loop Graph			**Opt. Triple Loop Graph**		
D	T_{Opt}	$T_{Heuristic}$	D	T_{Opt}	$T_{Heuristic}$
10	12	12	10	13	13
20	22	22	20	23	23
30	32	33	30	33	33
40	42	43	40	43	43
50	52	53	50	53	53
60	62	63	60	63	63
70	72	73	70	73	73
80	82	83	80	83	83
90	92	93	90	93	93
100	102	103	100	103	103

Table 4: Simulation results in optimal double and triple loop graphs where $s_3 = -(s_1 + s_2)$. The algorithm was run 100 times for every value of the diameter.

d	$T_{optimal}$	$T_{heuristic}$
9	13	13
10	14	14
11	15	15
12	16	16
13	17	17
14	18	18
15	19	19
16	20	20
17	21	21
18	22	22
19	23	23
20	24	24
21	25	25
22	26	26
23	27	27
30	34	34
31	35	35
32	36	36
45	49	49
46	50	50
47	51	51

Table 5: Simulation results in optimal triple loop graphs in the sense defined in [35]. The algorithm was run 100 times for all values of the diameter except 45,46,and 47. It was run 25 times for these cases.

d	$T_{optimal}$	$T_{heuristic}$
10	12	12
20	22	22
30	32	32
40	42	42
50	52	52
60	62	62
70	72	72
80	82	82
90	92	92
100	102	102

Table 6: Simulation results in optimal BDLGs. Algorithm was run 100 times for each diameter.

Chapter 5

Conclusion and Future Work

In this thesis we studied the problem of broadcasting in some circulant graphs. In particular we considered the bipartite double loop graphs, the optimal triple loop graphs, the general triple loop graphs, and the multiloop graphs. Using the plane tessellation technique, we calculated the maximum number of vertices that a bipartite double loop graph of diameter d can have. We also showed how to construct such a graph. The broadcast time of these graphs was calculated to be $d+2$ where d is the diameter. Concerning triple loop graphs of diameter d, we found that $d+2$ is a lower bound on the broadcast time of optimal graphs. This was due to the large number (more than $d+2$) of vertices at distance d for vertex 0. We also gave an algorithm that is guaranteed to complete broadcasting in $d+5$ time units. Finally, we generalized the above mentioned algorithm to get a $d+2k-1$ upper bound on the broadcast time in mulitloop graphs of degree $2k$.

Finding the broadcast time in a general graph is NP complete, therefore, there has been many heuristics that do broadcasting in a polynomial time. We tested one such algorithm developed by Harutyunyan and Shao [21]. In this thesis we presented the results of simulations of the above mentioned algorithm on different multiloop and chordal graphs. The obtained broadcast times were quite close to the theoretical bounds or the exact theoretical times.

We presented guidelines that help in calculating the shortest distance between two vertices in the optimal triple loop graphs. We used them to find some vertices at distance d from vertex 0, d being the diameter of the graph. It is possible to build up on these guidelines and get analytic formulas to calculate the shortest distance and determine the shortest path between any two vertices. These results might be of interest for routing purposes in optimal triple loop graphs. Further studies on optimal triple loop graphs can be done to improve the lower bound on the broadcast time. We believe that a $d+3$ lower bound on the broadcast time can be attained, and that is what we will try to prove in our future work. Moreover, the numerical simulations never got a broadcast time less than $d+4$, this comes to lure us to believe that even a $d+4$ lower bound can be possible to find. We will try to improve the general algorithm and customize it for the optimal triple loop graph such that it completes broadcasting in $d+4$ time units instead of $d+5$ as it stands now. In short, our ultimate goal will be to find the exact broadcast of optimal triple loop graphs.

We also intend to study the fault tolerance of triple loop graphs. In particular we believe that it is possible to get some analytic results concerning the fault tolerance of optimal triple loop graphs. In particular, the change in the diameter of the graph can be studied in the case of link failures and hence the additional required time to complete broadcasting can be calculated.

One natural generalization of the subject matter of this thesis is to study $k-$ *broadcasting* in multiloop graphs. $k-$ *broadcasting* is a broadcast model where an informed vertex can inform up to k of its neighbors simultaneously. To the best of our knowledge $k-$ *broadcasting* was studied only in double loop networks by Obradović, et. al. [39]. We will address this problem in our future work.

Our upper bound on the broadcast time of multiloop graphs deteriorates in value as the number of steps s_i increases. But as the degree the graph increases to become of the same order as the number of vertices then our result predicts a very large broadcast time which is linear in the number of vertices. We believe that in case of very large number of edges there is a different upper bound on the broadcast time. This issue can be part of our future research too.

Bibliography

[1] F. Aguiló, M.A. Fiol, G. Garcia, Triple loop networks with small transmission delay, *Discrete Mathematics*, 167/168, 1997, pp. 3-16.

[2] B. Arden and H. Lee, Analysis of chordal ring networks, *IEEE Trans. Comput.* C-30, 1981, pp. 291-295.

[3] J.-C. Bermond and P. Fraigniaud, Broadcasting and gossiping in de Bruijn networks, *SIAM J. Comput.*, 23, 1994, pp. 212-225.

[4] J.-C. Bermond, F. Comellas, and D.F. Hsu, Distributed Loop Computer Networks: A Survey, *Journal of Parallel and Distributed Computing*, 24, 1995, pp.2-10

[5] J.-C. Bermond, P. Hell, A. L. Liestman and J. G. Peters, Broadcasting in bounded degree graphs, *SIAM J. Discr. Math.*, 5, 1992, pp. 10-24.

[6] J.-C. Bermond, P. Hell, A. L. Liestman and J. G. Peters, Sparse broadcast graphs, *Discrete Appl. Math.*, 36, 1992, pp. 97-130.

[7] J.-C. Bermond and C. Peyrat, Broadcasting in de bruijn networks, *Proceeding 19th Southeastern Conference on Combinatorics, Graph Theory, and Computing, FL Congressus Numerantium 66*, 1988, pp. 283-292.

[8] R. Beivide, C. Martínez, C. Izu, J. Gutierrez, J.A. Gregorio, and J. Miguel-Alonso, Chordal Topologies for Interconnection Networks, *Lecture Notes in Computer Science*, Volume 2858, Jan 2003, pp. 385 - 392

[9] R. Beivide, E. Herrada, J. Balcázar, and A. Arruabarrena, Optimal Distance Networks of Low Degree for Parallel Computers, *IEEE Trans. on Comp.*, Vol 40, No 10, 1991, pp. 1109-1124.

[10] R.F. Browne and R.M. Hodgson, Symmetric Degree-four chordal ring networks, *IEEE Proc.*, 137(E), 4, 1990, pp. 310-318.

[11] C. Chen and F.K. Hwang, The minimum distance diagram of double loop graphs, *IEEE Trans. on Comp.*, Vol 49, No. 9, 2000, pp. 977-979.

[12] F. Comellas and P. Hell, Broadcasting in Generalized Chordal Rings, *Networks*, 42(3), 2003, pp. 123-134.

[13] M. J. Dinneen, M. R. Fellows and V. Faber, Algebraic constructions of efficient broadcast networks, *Applied Algebra, Algebraic Algorithms and Error-Correcting Codes 9, Lecture Notes in Computer Science*, 539, Springer, Berlin, 1991, pp. 152-158.

[14] Diestel, R., (2005), *Graph Theory*, 3^{rd}*edition*,Heidelberg: Springer-Verlag.

[15] J. Fàbrega and M. Zaragozá, Fault tolerant routings in double fixed-step networks, *Discrete Applied Mathematics*, 78, 1997, pp. 61-74.

[16] A. Farley and S. Hedetniemi, Broadcasting in grid graphs, *Proc. Eighteenth Southeastern Conference on Combinatorics, Graph Theory and Computing.* Utilitas Mathematica, Winnipeg, 1978, pp. 275-288.

[17] A. Farley, S. Hedetniemi, S. Mitchell and A. Proskurowski, Minimum broadcast graphs, *Discr. Math.*. 25, 1979, pp. 189-193.

[18] A. Farley and A. Proskurowski, Gossiping in grid graphs, *J.Combin.Inform. Systems Sci.*, 5, 1980, pp. 161-172.

[19] P. Fraigniaud and E. Lazard, Methods and problems of communication in usual networks, *Discrete Appl. Math.*, 53, 1994, pp. 79-133.

[20] H. A. Harutyunyan and E. Maraachlian, Linear Algorithm for Broadcasting in Unicyclic Graphs, *submitted.*

[21] H. A. Harutyunyan and B. Shao, An efficient heuristic for broadcasting in networks, *Journal of Parallel and Distributed Computing*, Vol. 66, Jan 2006, pp.68-76.

[22] H. A. Harutyunyan and B. Shao, Optimal k-broadcast in Trees, *Congressus Numernatium*, 160, 2003, pp. 117-127.

[23] H. A. Harutyunyan and B. Shao, A Heuristic for k-broadcasting in arbitrary networks, *Seventh IEEE Conference on Graph Theory Application (IV 2003)*, London, England, 2003, pp. 287-293.

[24] S.M Hedetniemi, S.T. Hedetniemi and A.L. Liestman, A survey of gossiping and broadcasting in communication networks, *Networks*, 18, 1988, pp. 319-349.

[25] J. Hromkovic, R. Klasing, B. Monien and R. Peine, Dissemination of information in interconnection networks, *Combinatorial Network Theory*, D.-Z. Du, D.F. Hsu(eds.), Kluwer Academic Publishers, 1996, pp. 125-212.

[26] F.K. Hwang, A complementary survey on double-loop networks, *Theoretical Computer Science*, 263, 2001, pp. 211-229.

[27] F.K. Hwang, A survey on multi-loop networks, *Theoretical Computer Science*, 299, 2003, pp. 107-121.

[28] L. Khachatrian and H. S. Haroutunian, Minimal broadcast trees, *XIV All Union School of Computing Networks*, Minsk, 1989, pp. 36-40.

[29] L. H. Khachatrian and H. S. Haroutunian, Construction of new classes of minimal broadcast networks, *Proceedings 3rd International Colloquium on Coding Theory*, Dilijan, Armenia, 1990, pp. 69-77.

[30] R. Klasing, B. Monien, R. Peine and E. A. Stöhr, Broadcasting in butterfly and deBruijn networks, *Discrete Appl. Math.*, 53, 1994, pp. 183-197.

[31] Kotsis, G. (1992), *Interconnection Topologies for Parallel Processing Systems*, PhD Thesis, Universität Wien, Austria.

[32] T. Leighton, *Introduction to parallel algorithms and architectures: array-trees-hypercubes*, Morgan-Kaufmann Publishers, San Mateo, California, 1992.

[33] A. L. Liestman, J. Opatrny and M. Zaragozá, Network Properties of Double and Triple Fixed Step Graphs, *International Journal of Foundations of Computer Science*, 9(1), 1998, pp. 57-76.

[34] C. Martínez, R. Beivide, C. Izu and J. Miguel-Alonso, Characterization of the Class of Optimal Dense Circulant Graphs of Degree Four, *Proc. de las XIV Jornadas de Paralelismo*, Legane´s, Madrid, España, 2003, pp 133-138.

[35] E. Monakhova, Optimal Triple Loop Networks with Given Transmission Delay: Topological Design and Routing, *Proceedings of the International Network Optimization Conference (INOC'2003)*, Evry/Paris, France, 2003, pp. 410-415.

[36] P. Morillo, F. Comellas, and M. A. Fiol, Metric Problems in Triple Loop Graphs and Digraphs Associated to an Hexagonal Tessellation of the Place, Tech. Report 05-0286, UPC, 1986

[37] P. Morillo, F. Comellas, and M. A. Fiol, The optimization of chordal ring networks, *Commnication technology*, Q. Yasheng and W. Xiuying (Editors), World Scientific, Singapore, 1987, ISBN 9971-50-349-9, pp. 295-299.

[38] Muga, F., Undirected Circulant Graphs, *Proc. Int. Symp. on Parallel Architectures, Algorithms and Networks (ISPAN'94)*, Kanazawa, Japan, 1994, pp. 113 - 118.

[39] Obradović, N., Peters, J., and Ružić, G., Reliable Broadcasting in Double Loop, *Networks*, 46 (2), Sept. 2005, pp. 88 - 97.

[40] J.-F. Saclé, Lower bounds for the size in four families of minimum broadcast graphs, *Discrete Math.*, 150, 1996, pp. 359-369.

[41] P.J. Slater, E.J. Cockayne and S.T. Heditniemi, Information dissemination in trees, *SIAM J.Comput.*, vol. 10, no. 4, 1981, pp. 692-701.

[42] B. Shao, On k-broadcasting in Graphs, *PhD Thesis* Departmenet of Computer Science and Software Engineering, Concordia University, 2006.

[43] C.K. Wong, and D. Coppersmith, A Combinatorial problem related to multimodule memory organizations, *J. ACM*, 21, 1974, pp. 392-402.

[44] J. Xiao and X. Wang, A research on minimum broadcast graphs, *Chin. J. Comput.*, 11, 1988, pp. 99-105.

[45] X. Xu, Broadcast networks on $2^p - 1$ nodes and minimum broadcast network on 127 nodes, *Master Thesis* Department of Computer Science and Software Engineering, Concordia University, 2003.

[46] J. L. A. Yebra, M. A. Fiol, P. Morillo, and I. Alegre, The diameter of undirected graphs associated to plane tessellations, *Ars Combin 20B*, 20B, 2985, pp. 159-172.