DEPARTMENT OF MATHEMATICS COMPUTER SCIENCE & ENGINEERING TECHNOLOGY

Dominion on Ladders

by

Erin O'Connor Gray

A Thesis submitted to the Graduate Faculty of
Elizabeth City State University
in partial fulfillment of the
requirements for the Degree of
Master of Science in
Applied Mathematics.

April

2023

APPROVED BY

Julian D. Allagan, Ph.D.
Committee Chair

Mohammed Talukder, Ph.D.
Committee Member

Mohamed Elbakary, Ph.D.
Committee Member

Gabriela H. del Villar, Ph.D.
Committee member

ABSTRACT

A grid graph $G_{(m,n)}$ is often used to model certain computer networking configurations. To investigate the 'robustness' of such network against an adversarial attack, we study two related parameters on its subgraph, called Ladder, which we denote by L_n: Domination (γ) and Dominion (ζ). The smaller the γ value for a graph, the less resources are needed to secure the network represented by the graph. On the other hand, the greater the ζ value for a graph, the more secure is the network represented by the graph. We concluded that, for $n \geq 2$,

$$\gamma(L_n) = \left\lfloor \frac{n}{2} + 1 \right\rfloor$$

and,

$$\zeta(L_n) = \begin{cases} 3 & \text{if } n = 3 \\ 2 & \text{if } n = 2k+1,\ k > 1 \\ 6 & \text{if } n = 2 \\ 12 & \text{if } n = 4 \\ 17 & \text{if } n = 6 \\ 2n+4 & \text{if } n = 2k,\ k \geq 4. \end{cases}$$

DEDICATION

This Thesis is dedicated to Justin Gray and Denise O'Connor. Thank you for your support and love.

ACKNOWLEDGEMENT

I would like to thank all of my teachers at Elizabeth City State University. Particularly, I would like to thank Dr. Allagan for his support while writing this thesis and completing my courses. I would also like to thank Dr. Jones for his insightful teaching while in the program.

Contents

1 Introduction **1**

1.1 Background and Overview 1

1.1.1 Applications of Discrete Mathematics 2

1.2 Graph Theory Background 4

1.2.1 Basic Concepts of Graph Theory 5

1.2.2 Planar Graphs 7

1.2.3 Hamiltonian Graphs 9

1.2.4 Grid Graphs 11

1.2.5 Ladder Graphs 14

2 Computing Domination Number of Graphs **16**

2.1 Common Methods for Computing the Domination Number of a Graph 16

2.1.1 Brute Force Method 16

2.1.2 Exact Algorithms 17

2.1.3 Approximation Algorithms 17

2.1.4 Study Methods 20

2.2 Domination Number of Ladder Graphs 20

3 Computing Dominion of Ladder Graphs **25**

3.1 Dominion of Ladder Graphs L_1 through L_{10} 25

4 Extension to Other Grid Graphs **36**

5 Conclusion and Further Research **40**

5.1 Wireless Sensor Networks 41

5.2 Social Network Analysis 42

5.3 Distributed Systems 43

5.4 Network Optimization 44

5.5 Further Research 45

List of Figures

1.1 A depiction of Euler's 7 Bridges of Konigsberg Problem 3
1.2 Planar graph (left), plane drawing (center), and straight line drawing (right) all of the same $G(4,6)$ 9
1.3 Nonplanar graph of $G(5,10)$ and nonplanar graph if $G(6,9)$ 9
1.4 On the left a Hamiltonian Graph of $G(4,4)$ and a Non-Hamiltonian Graph of $G(5,6)$ on the right 10
1.5 Grid graph with notation G_{3x4} 11
1.6 Ladder Graphs L_1, L_2, L_3, L_4 (left to right) 15

2.1 Ladder Graphs of L_1 to L_4 21
2.2 Ladder Graphs of L_1 to L_4 with colored vertices showing a dominating set 22
2.3 Ladder Graphs of L_5 to L_{10} 22
2.4 Ladder Graphs of L_5 to L_{10} with colored vertices showing a dominating set 23

3.1 Dominion of ladder graph, L_1; $\zeta = 2$ 26
3.2 Dominion of ladder graph, L_2; $\zeta = 6$ 27
3.3 Dominion of ladder graph, L_3; $\zeta = 3$ 28
3.4 Dominion of ladder graph, L_5; $\zeta = 2$ 28
3.5 Dominion of ladder graph, L_7; $\zeta = 2$ 29
3.6 Dominion of ladder graph, L_9; $\zeta = 2$ 29
3.7 Dominion of ladder graph, L_4; $\zeta(L_4) = 12$ 30
3.8 angle=90 33
3.9 angle=90 34
3.10 angle=90 35

4.1 angle=90 37
4.2 angle=90 38
4.3 Dominion of path graph, $G_{(3,3)}$; $\zeta(G_{(3,3)}) = 10$ 39

List of Tables

2.1 Domination Number of first ten ladder graphs 24

3.1 Domination Number and Dominion of first ten ladder graphs 31

4.1 Dominion of Some Grid Graphs 36

Chapter 1 Introduction

1.1 Background and Overview

Discrete mathematics is a branch of mathematics that deals with discrete or finite sets, as opposed to continuous mathematical concepts such as calculus and analysis. Discrete mathematics has a wide range of applications, from computer science and cryptography to telecommunications and engineering. The study of discrete mathematics provides a foundation for many fields of computer science, including algorithms, data structures, and computer networks.

Discrete mathematics has a rich history that dates back to the ancient Greeks. Euclid's Elements, written in 300 BC, contains some of the earliest examples of discrete mathematics. Euclid's work on geometry involved dividing space into discrete points, lines, and planes. These concepts form the basis of modern geometry, which is an important part of discrete mathematics [4].

In the 17th century, Blaise Pascal and Pierre de Fermat laid the groundwork for probability theory, which is an essential part of discrete mathematics. Pascal's work on gambling problems and Fermat's work on the calculus of probabilities are considered some of the earliest examples of probability theory. Probability theory deals with the study of random events and their probabilities [21].

In the 18th century, Leonhard Euler made significant contributions to graph theory, which is now one of the most important branches of discrete mathematics. Graph theory provides a way to represent and analyze networks, such as computer networks and social networks. Euler's work on the Seven Bridges of Königsberg problem (shown in Figure 1.1), which involved finding a path that crossed each bridge exactly once, is considered the first example of graph theory. Euler's work on graph theory laid the foundation for modern graph theory and paved the way for the development of algorithms that can solve complex graph problems.

In the 19th century, George Boole developed Boolean algebra, which is a branch of discrete mathematics that deals with logical operations. Boolean algebra forms the basis of digital logic circuits, which are used in computers and other digital devices. Boolean algebra also deals with the study of logical propositions and their truth values. Boole's work on Boolean algebra provided the foundation for the development of digital circuits and computer hardware [19].

In the late 19th and early 20th centuries, Georg Cantor developed set theory, which is the foundation of modern mathematics. Set theory deals with discrete objects and provides a way to represent and manipulate mathematical objects. Cantor's work on set theory led to the discovery of new mathematical concepts, such as infinity and transfinite numbers. Set theory provided the foundation for the development of modern mathematics, including the development of axiomatic systems and the study of mathematical logic.

The 20th century saw a significant expansion of discrete mathematics, with the development of number theory, combinatorics, and computational complexity theory. Number theory deals with the properties of integers, and it has applications in cryptography and coding theory. Combinatorics deals with the study of finite sets and counting techniques. It has applications in computer science, statistics, and optimization. Computational complexity theory deals with the study of algorithms and their efficiency. It has applications in computer science and cryptography.

1.1.1 Applications of Discrete Mathematics

Discrete mathematics has many applications in computer science and related fields. Some of the most important applications include:

1. Cryptography: Cryptography is the science of secure communication. It involves the use of mathematical techniques to ensure the confidentiality and integrity of information. Discrete mathematics provides the mathematical foundations for

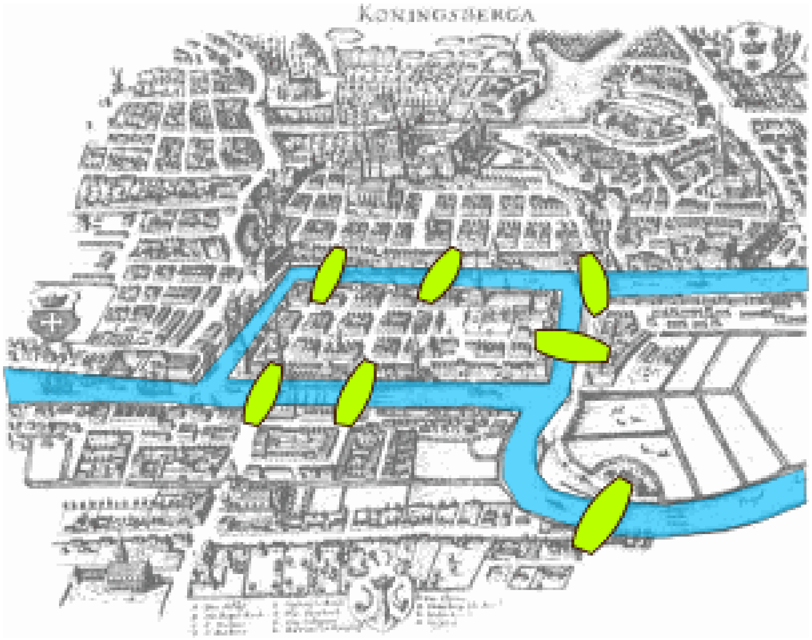


Figure 1.1: A depiction of Euler's 7 Bridges of Konigsberg Problem

many cryptographic techniques, including encryption, digital signatures, and secure key exchange.

2. Algorithms: Algorithms are sets of instructions for solving problems. Discrete mathematics provides the theoretical foundation for the design and analysis of algorithms. The study of computational complexity theory, which is a branch of discrete mathematics, provides a way to measure the efficiency of algorithms.

3. Data structures: Data structures are the building blocks of computer programs. Discrete mathematics provides a way to represent and manipulate data structures, such as arrays, linked lists, and trees.

4. Computer networks: Computer networks are systems of interconnected computers. Discrete mathematics provides a way to represent and analyze computer networks, using graph theory and other mathematical techniques.

5. Optimization: Optimization is the process of finding the best solution to a problem. Discrete mathematics provides a way to model and solve optimization problems, using techniques from combinatorics and graph theory.

Discrete mathematics is a vital branch of mathematics that has many important applications in computer science and related fields. Its history dates back to ancient times, and it has evolved to become a central part of modern mathematics. The study of discrete mathematics provides a foundation for many fields of computer science, including algorithms, data structures, and computer networks. Its applications in cryptography, algorithms, data structures, computer networks, and optimization make it an essential tool for solving problems in the digital age [22].

In the next section, we will delve more into graph theory and the different graphs that arise in this realm. As we continue into Chapter 2, we will look at calculating the dominion of graphs. Then we will look into the patterns of dominion that arise in ladder graphs and some grid graphs.

1.2 Graph Theory Background

Graph theory is a branch of discrete mathematics that deals with the study of graphs. A graph is a mathematical structure that consists of a set of vertices and a set of edges. The edges represent connections between the vertices, and they can be directed or undirected. Graph theory has applications in a wide range of fields, including computer science, operations research, physics, chemistry, and social sciences [29].

The origins of graph theory can be traced back to the work of Leonhard Euler in the 18th century. Euler's work on the Seven Bridges of Königsberg problem, which involved finding a path that crossed each bridge exactly once, is considered the first example of graph theory. Euler showed that it was not possible to find such a path, which led to the development of graph theory.

In the 19th century, graph theory was further developed by mathematicians such as Augustin-Louis Cauchy and Gustav Kirchhoff. Cauchy was the first mathematician to use the term "graph" to describe a mathematical structure, while Kirchhoff developed the theory of electrical circuits, which is based on graph theory.

In the 20th century, graph theory became a major area of research in mathematics and computer science. Mathematicians such as Paul Erdős and Rényi Alfréd made significant contributions to the development of random graph theory, while computer scientists such as Edsger Dijkstra and Robert Tarjan developed algorithms for solving graph problems.

A. Basic Notation Meanings

G .... A graph.

V .... The vertex set of G.

E .... The edge set of G.

$G = (V, E)$ ....A graph with vertex set V and edge set E.

L_n ....A ladder graph with 2 columns 2n vertices

n.............................The number of rows in a ladder graph

G_{mxn} ....A grid graph with m rows and n columns

$\gamma(G)$.............................Domination number of a graph

$\zeta(G)$.............................Dominion of a graph

1.2.1 Basic Concepts of Graph Theory

A graph is a mathematical structure that consists of a set of vertices and a set of edges. The vertices represent objects or entities, while the edges represent connections between them.

The degree of a vertex in a graph is the number of edges that are incident to that vertex. In an undirected graph, the degree of a vertex is equal to the number of edges that are adjacent to that vertex. In a directed graph, the degree of a vertex is the sum of the indegree (the number of edges that point to the vertex) and the outdegree (the number of edges that point away from the vertex).

A path in a graph is a sequence of vertices in which each vertex is adjacent to the next vertex in the sequence. A cycle is a path in which the first and last vertices are the same. A connected graph is a graph in which there is a path between every pair of vertices. A disconnected graph is a graph in which there are two or more disjoint sets of vertices that are not connected by any edges.

Types of Graphs

There are several types of graphs that are commonly used in graph theory. Some of the most important types include:

1. Simple graph: A simple graph is an undirected graph with no self-loops or multiple edges.

2. Multigraph: A multigraph is an undirected graph that allows multiple edges between pairs of vertices.

3. Directed graph: A directed graph (also known as a digraph) is a graph in which each edge has a direction.

4. Weighted graph: A weighted graph is a graph in which each edge is assigned a weight or cost.

5. Complete graph: A complete graph is a simple graph in which every pair of distinct vertices is connected by a unique edge.

6. Bipartite graph: A bipartite graph is a simple graph in which the vertices can be partitioned into two sets such that every edge connects a vertex in one set to a vertex in the other set.

7. Planar graph: A planar graph is a graph that can be drawn on a plane without any edges crossing.

8. Regular graph: A regular graph is a graph in which every vertex has the same degree.

9. Connected graph: A connected graph is a graph in which there is a path between every pair of vertices.

10. Disconnected graph: A disconnected graph is a graph in which there are two or more disjoint sets of vertices that are not connected by any edges.

11. Tree: A tree is a connected acyclic graph.

12. Forest: A forest is a disjoint union of trees.

13. Eulerian graph: An Eulerian graph is a graph that contains a cycle that passes through every edge exactly once.

14. Hamiltonian graph: A Hamiltonian graph is a graph that contains a cycle that passes through every vertex exactly once.

15. Strongly connected graph: A strongly connected graph is a directed graph in which there is a directed path between every pair of vertices.

16. Weakly connected graph: A weakly connected graph is a directed graph in which there is a path between every pair of vertices, ignoring the direction of the edges.

1.2.2 Planar Graphs

In graph theory, a planar graph is a graph that can be drawn on a plane without any edges crossing. This means that no two edges in the graph can intersect or cross over each other. As seen in Figure 1.2, the planar graphs all show connections. While the graph on the left looks like two edges cross, the other depictions show how the edges and vertices can be reconfigured to not cross. In Figure 1.3, there is no way to reconfigure these combinations of vertices and edges to create a version where all of the edges do not cross. Planar graphs have important applications in a variety of fields, including computer science, mathematics, and physics [5].

One of the key properties of planar graphs is that they can be represented by a planar embedding, which is a specific arrangement of the vertices and edges on the plane that satisfies the planarity condition. Planar embeddings can be used to visualize planar graphs and to study their properties.

Planar graphs have several important properties that set them apart from other types of graphs. One of the most notable is the Euler's formula, which relates the number of vertices, edges, and faces of a planar graph. Specifically, the formula states that for any connected planar graph with v vertices, e edges, and f faces, v - e + f = 2. This formula is important because it provides a way to determine the number of faces in a planar graph, given the number of vertices and edges.

Another important property of planar graphs is that they have a maximum number of edges, known as the planar graph theorem. This theorem states that any planar graph with n vertices has at most 3n-6 edges. Moreover, if the graph is a simple graph (no self-loops or multiple edges), then it has at most 3n-6 edges.

One of the most famous problems in graph theory is the four-color theorem, which states that any planar graph can be colored with four or fewer colors such that no two adjacent vertices have the same color. This theorem was first conjectured in the 19th century and was finally proven in 1976 by Kenneth Appel and Wolfgang Haken using computer-assisted methods [3].

Planar graphs have a variety of applications in computer science, particularly in the design and analysis of algorithms. For example, planar graphs can be used to represent geographical maps or circuit layouts, and algorithms can be developed to solve problems related to these representations. Planar graphs also have applications in network optimization, where they can be used to model communication networks and transportation networks.

In addition, planar graphs have been used in the study of other mathematical problems, such as graph coloring, graph embeddings, and topological graph theory. Planar graphs have also been used in the study of physical systems, such as the behavior of electrons in solids and the properties of soap bubbles.

One of the most well-known planar graphs is the complete graph, which is a graph in which every pair of vertices is connected by an edge. The complete graph with n vertices, denoted K_n, has $\frac{n(n-1)}{2}$ edges and is not planar for $n \geq 5$. However, it is possible to construct planar graphs that are closely related to the complete graph, such as the complete bipartite graph and the Turán graph. In Figure 1.1, the planar graph is a complete graph as each vertex is connected to one another through edges that do not cross.

Planar graphs can be classified into several different types based on their properties. For example, a planar graph is called a triangulated graph if every face is a triangle. A planar graph is called a 3-connected graph if removing any two vertices from the graph does not disconnect it. A planar graph is called a maximal planar graph if adding any new edge would cause the graph to become non-planar.

There are several types of planar graphs, which can be defined based on certain characteristics of the graph. Here are some common types:

1. Simple planar graphs: These are planar graphs with no loops or multiple edges.

2. Connected planar graphs: These are planar graphs in which there is a path between any two vertices.

3. Maximal planar graphs: These are planar graphs in which every face is a triangle. In other words, adding any edge to the graph would make it non-planar.

4. Outerplanar graphs: These are planar graphs in which all vertices lie on a single face.

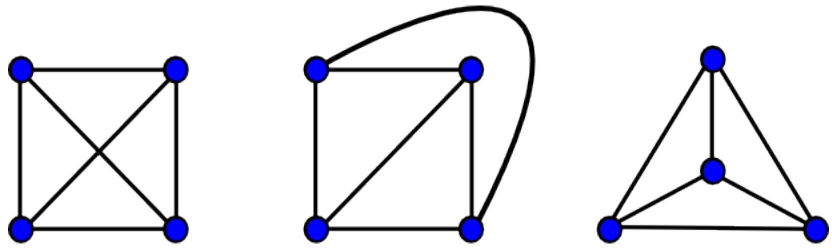
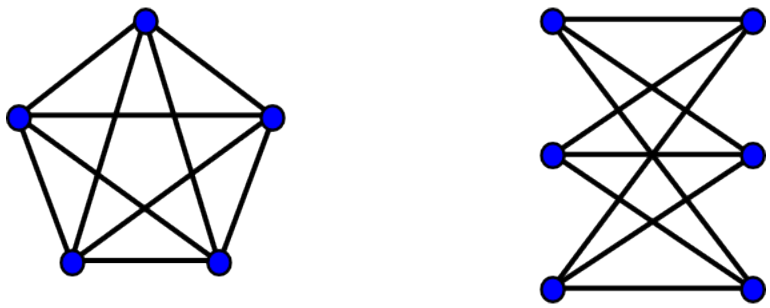


Figure 1.2: Planar graph (left), plane drawing (center), and straight line drawing (right) all of the same $G\,(4,6)$



[30]

Figure 1.3: Nonplanar graph of $G\,(5,10)$ and nonplanar graph if $G\,(6,9)$

5. Planar graphs with bounded degrees: These are planar graphs in which every vertex has a bounded degree. For example, a 4-regular planar graph is a planar graph in which every vertex has degree 4.

6. Planar graphs with special properties: There are many planar graphs with special properties, such as Eulerian planar graphs (in which there is a closed walk that visits every edge exactly once), Hamiltonian planar graphs (in which there is a cycle that visits every vertex exactly once), and bipartite planar graphs (in which the vertices can be divided into two disjoint sets such that every edge connects a vertex in one set to a vertex in the other set).

1.2.3 Hamiltonian Graphs

A Hamiltonian graph is a graph that contains a Hamiltonian cycle, which is a cycle that passes through every vertex of the graph exactly once. In other words, a Hamiltonian graph is a graph that has a closed path that visits every vertex of the graph

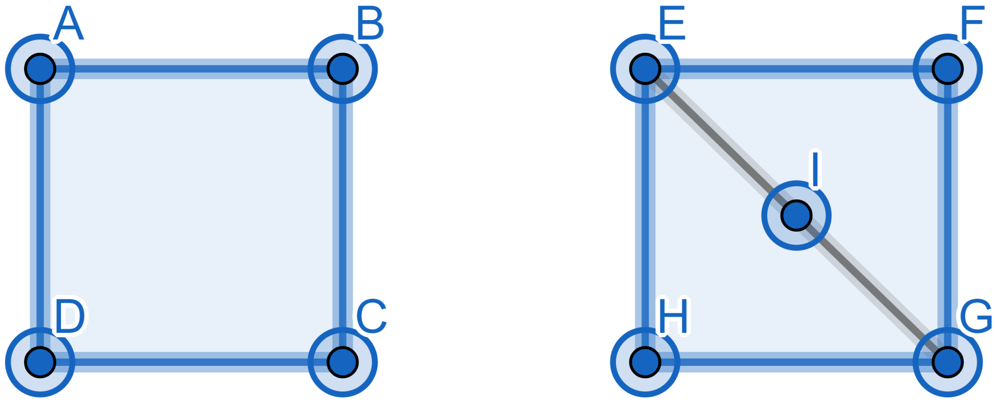


Figure 1.4: On the left a Hamiltonian Graph of $G\,(4, 4)$ and a Non-Hamiltonian Graph of $G\,(5, 6)$ on the right

exactly once [12].

Formally, a graph G = (V, E) is Hamiltonian if there exists a cycle C in G that contains every vertex in V. A Hamiltonian cycle is a cycle that passes through every vertex of G exactly once. As shown in Figure 1.4, the Hamiltonian Graph on the left has a full cycle; whereas, the Non-Hamiltonian Graph on the right does not have a full cycle that includes all vertices. For instance, there is a cycle that goes through vertices E, F, G, and H, but it does not include the vertex I. There is also a cycle that goes through E, F, G, and I, but it does not include the vertex H.

The concept of Hamiltonian graphs is named after the mathematician Sir William Rowan Hamilton, who studied these types of graphs in the 19th century.

Determining whether a graph is Hamiltonian or not is an NP-complete problem, meaning that there is no known polynomial time algorithm to solve it. However, there are some special types of graphs, such as complete graphs and cycles, which are always Hamiltonian. Hamiltonian graphs can be planar or nonplanar. Also, while all complete graphs are Hamiltonian, not all Hamiltonian graphs are complete.

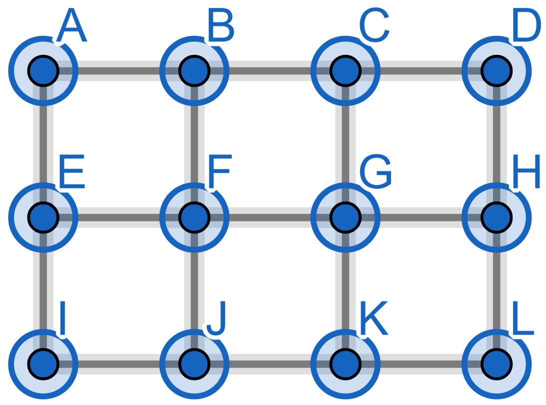


Figure 1.5: Grid graph with notation G_{3x4}

1.2.4 Grid Graphs

Grid graphs are a particular type of graphs that have a regular structure, often used to represent a two-dimensional space. Grid graphs have applications in various fields such as computer science, physics, and biology. They have a straightforward representation and are used to solve various problems such as routing, scheduling, and network optimization. In this article, we will explore the definition of grid graphs, their properties, and various algorithms used to solve problems on grid graphs. The grid graph represented by Figure 1.5, shows a 3 row by 4 column graph with 12 vertices and 17 edges.

1.2.4.1 Definition of Grid Graphs

A grid graph is a graph that represents a two-dimensional space, where each vertex corresponds to a point in the space, and each edge corresponds to a pair of adjacent points. The graph has a regular structure, where each vertex has the same degree. In most cases, the degree of each vertex is four, corresponding to the four neighbors of the point in the space. However, other degrees are possible, such as six for hexagonal grid graphs [2].

1.2.4.2 Properties of Grid Graphs

Grid graphs have some unique properties that make them interesting and useful for various applications. Here are some of the most important properties of grid graphs:

1. Regular Structure: Grid graphs have a regular structure, which means that each vertex has the same degree. This property makes grid graphs easy to represent and manipulate, as well as to analyze.

2. Planar Graphs: Grid graphs are planar graphs, which means that they can be drawn on a plane without any edges crossing. This property makes grid graphs useful in many applications, such as designing computer chips or routing algorithms.

3. Symmetric: Grid graphs are symmetric, which means that each vertex has the same number of neighbors, and the distances between vertices are the same. This property makes grid graphs useful in physics, where they can be used to represent regular lattices in materials.

4. Hamiltonian Cycles: Grid graphs have Hamiltonian cycles, which means that it is possible to find a path that visits every vertex exactly once. This property makes grid graphs useful in scheduling algorithms, where the goal is to find a sequence of tasks that visit every point in a two-dimensional space.

1.2.4.3 Algorithms on Grid Graphs

Grid graphs have numerous applications in various fields, and several algorithms are used to solve problems on grid graphs. Here are some of the most common algorithms used on grid graphs [13]:

1. Breadth-First Search: Breadth-first search is a graph traversal algorithm that starts at a particular vertex and visits all the vertices in the graph in breadth-first order. Breadth-first search is used to find the shortest path between two vertices in a grid graph, as well as to find connected components in the graph.

2. Dijkstra's Algorithm: Dijkstra's algorithm is a shortest path algorithm that finds the shortest path between two vertices in a weighted graph. Dijkstra's algorithm is used on grid graphs to find the shortest path between two points, where the edges of the graph have weights that represent the distance between adjacent points.

3. A* Algorithm: A* algorithm is a pathfinding algorithm that finds the shortest path between two vertices in a graph. A* algorithm is similar to Dijkstra's algorithm, but it uses a heuristic function to estimate the distance between the current vertex and the target vertex. A* algorithm is used on grid graphs to find the shortest path between two points, where the edges of the graph have weights that represent the distance between adjacent points.

4. Flood Fill Algorithm: Flood fill algorithm is a graph traversal algorithm that fills a region of connected vertices in a graph. Flood fill algorithm is used on grid graphs to find connected components, as well as to fill regions in an image processing application.

1.2.4.4 Applications of Grid Graphs

Grid graphs have numerous applications in various fields, such as computer science, physics, biology, and more. In this section, we will explore some of the most common applications of grid graphs.

1. Computer Networks: Grid graphs are used in computer networks to model the network topology. In a grid network, each vertex represents a node in the network, and each edge represents a communication link between two nodes. Grid graphs are used to optimize network performance, such as minimizing the average distance between nodes or maximizing the bandwidth of the network.

2. Routing Algorithms: Routing algorithms are used to find the shortest path between two points in a network. Grid graphs are used in routing algorithms to find the shortest path between two points in a two-dimensional space. The A* algorithm is commonly used in grid graphs to find the shortest path, where the edges of the graph have weights that represent the distance between adjacent points.

3. Image Processing: Grid graphs are used in image processing applications, such as image segmentation and object recognition. In image segmentation, grid graphs are used to partition an image into regions of similar color or texture. The flood fill algorithm is commonly used in grid graphs to fill regions in an image. In object recognition, grid graphs are used to model the shape of objects in an image.

4. Physics: Grid graphs are used in physics to model regular lattices in materials. The symmetric property of grid graphs makes them useful in physics, where they can be used to represent the structure of materials. Grid graphs are also used to model physical phenomena such as fluid flow, where the edges of the graph represent the flow of fluid between adjacent points.

5. Scheduling Algorithms: Scheduling algorithms are used to schedule tasks on a two-dimensional space, such as a factory floor or a warehouse. Grid graphs are used to represent the two-dimensional space, where each vertex represents a location on the floor, and each edge represents the distance between adjacent locations. Hamiltonian cycles in grid graphs make them useful in scheduling algorithms, where the goal is to find a sequence of tasks that visit every location exactly once.

6. Game Theory: Grid graphs are used in game theory to model board games such as chess and checkers. In board games, grid graphs are used to represent the board, where each vertex represents a square on the board, and each edge represents the adjacency between two squares. Game theory algorithms are used on grid graphs to find optimal strategies for the players.

1.2.5 Ladder Graphs

A ladder graph is a type of grid graph that consists of two columns of vertices, with each vertex in one row connected to exactly one vertex in the other row. In other words, a ladder graph can be thought of as a graph that looks like a ladder, with rungs connecting the two sides.

More formally, a ladder graph can be defined as follows: Let n be a positive integer. The ladder graph L_n is defined as the graph with 2n vertices, labeled v_1, v_2, ..., v_n in the left column, and v'_1, v'_2, ..., v'_n in the right column. Each vertex v_n in the left column is connected to the corresponding vertex v'_n in the right column, and adjacent vertices in each row and column are connected to each other by edges. Figure2.1 shows a family of Ladders.

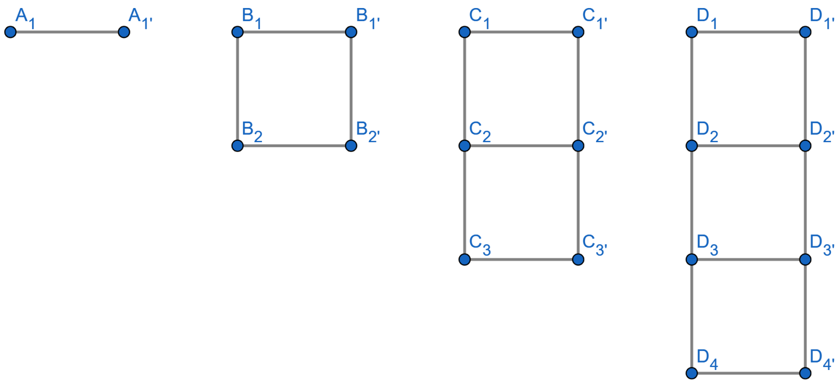


Figure 1.6: Ladder Graphs L_1, L_2, L_3, L_4 (left to right)

Ladder graphs are interesting because they are a type of planar graph. Specifically, ladder graphs are planar for $n \leq 3$, but for $n \geq 4$, they are not planar. Additionally, ladder graphs have some interesting properties when it comes to connectivity and Hamiltonicity. For example, ladder graphs are bipartite, meaning that the vertices can be divided into two disjoint sets, such that every edge connects a vertex in one set to a vertex in the other set. Moreover, ladder graphs are Hamiltonian, meaning that there exists a cycle that visits every vertex exactly once.

Chapter 2 Computing Domination Number of Graphs

Definition 2.0.1 (Definition: Domination Number)**.** A subset S of vertices in a graph S is said to dominate G if every vertex in G is either in S or adjacent to a vertex in S. The domination number of a graph G, denoted by $\gamma(G)$ is the size of the smallest dominating set among all dominating sets of $V(G)$.

More formally, a *dominating set* for a graph $G = (V, E)$ is a subset $S \subseteq V$ such that every vertex $v \in V$ is either in S or has a neighbor $u \in S$. The vertex $u \in S$ is said to *cover*, *monitor*, or *dominate* the vertex $v \in V$ if either $u = v$ or $uv \in E$. A dominating set S is a *minimal dominating set* if no proper subset $S' \subset S$ is a dominating set. The *domination (number)* $\gamma(G)$ of a graph G is the minimum cardinality among all dominating sets of G.

Determining the domination number is a significant graph theory problem having applications in social networks, computer networks, and operations research.

There are several algorithms that are often used to compute the domination number of a graph, each with its strengths and weaknesses. In the next section, we will discuss some of the common methods for computing the domination number of a graph.

2.1 Common Methods for Computing the Domination Number of a Graph

2.1.1 Brute Force Method

The brute force method is an easy way to determine a graph's domination number. With this approach, we create every feasible subset of the graph's vertices and determine whether or not each one is a dominating set. The graph's domination number is equal to the size of the smallest dominating set among all subsets. The number of potential subsets is exponential in the number of vertices, hence the brute force method is impractical for big graphs [15].

2.1.2 Exact Algorithms

Exact algorithms incorporate using computer programming to determine the domination number. For the dominating number problem, exact algorithms promise to identify the best answer. These algorithms combine various methods, including dynamic programming, linear programming, and branch and bound. Large graphs can be handled by exact algorithms, which can also issue an optimality certificate.

The Bron-Kerbosch algorithm is one of the most well-known exact algorithms to the domination number problem. Finding all maximal cliques in a network is done via a recursive technique. Every pair of vertices in a clique are adjacent. A clique is a subset of vertices in a graph. The algorithm searches the graph for all maximal cliques, and then determines whether or not each clique is a dominant set. The graph's domination number is equal to the size of the smallest dominating set among all maximal cliques.

The Bron-Kerbosch approach is faster than the brute force method but still exponential in the number of vertices, with a temporal complexity of $O\left(3\left(\frac{n}{3}\right)\right)$. Yet, the process can be improved using a variety of methods, including pivot selection and pruning [8]

2.1.3 Approximation Algorithms

Algorithms that promise their result will be relatively near to the ideal solution are known as approximation algorithms. These algorithms can handle enormous graphs and are frequently quicker than precise algorithms. Yet, the algorithm's approximation factor determines how accurate the solution is [20].

2.1.3.1 Greedy Algorithm

The greedy algorithm is an approximation algorithm that is a heuristic method for determining a graph's domination number. Starting with an empty set, the procedure iteratively adds vertices until the set is still a dominating set. The method chooses a vertex and includes it in the set at each iteration based on how many uncovered neighbors it has. The method keeps going until the dominating set has all of the vertices covered. The size of the resulting dominant set determines the graph's domination number [10].

The greedy algorithm has a time complexity of $O(n^2)$, where n is the number of vertices in the graph. However, the algorithm does not always produce the optimal solution since it relies on trial and error.

2.1.3.2 LP-relaxation Algorithm

The LP-relaxation approach is another widely used approximation method. The LP-relaxation method converts the dominance number problem's integer programming formulation into a linear programming problem. A linear programming solver, such as the simplex algorithm, can effectively solve the LP-relaxation problem. An upper constraint on the domination number is provided by the ensuing fractional solution. The LP-relaxation approach frequently provides a decent approximation of the domination number, is simple to use, and has a polynomial time complexity. The LP-relaxation method, however, might sometimes result in a fractional solution that does not match to a legitimate dominant set, necessitating the use of extra rounding methods to arrive at a legitimate solution [28].

2.1.3.3 Randomized Rounding Method

Another well-liked approximation method for the dominance number problem is the randomized rounding approach. The LP-relaxation problem is initially solved using the randomized rounding technique to arrive at a fractional answer. A randomization procedure is then used to round the fractional answer to an integral solution. In randomized rounding, each fractional variable is independently rounded to a value of 0 or

1, with probabilities that depend on the value of the fractional variable. The set that results is next verified to see if it constitutes a dominating set. The size of the answer produced by the randomized rounding method is frequently within a constant factor of the ideal solution and has polynomial time complexity. The randomized rounding method, however, might be sensitive to the choice of the rounding probabilities and may need a lot of random samples to get an accurate result [26].

2.1.3.4 Local Search Method

Another approximation method for the dominance number problem is the local search approach. Using a random dominating set as a starting point, the local search approach iteratively enhances the answer by modifying the set locally. A local change entails removing a vertex from the set and replacing it with one that is not currently there. The local search approach is a heuristic strategy, thus it cannot ensure that the outcome is the best one. Nonetheless, because it can deliver accurate approximations in a reasonable amount of time, it is frequently utilized in practice [26].

Typically, the local search technique begins with a random dominant set S. The domination number, $\gamma(S)$, is used to calculate its quality. By making minor adjustments to the set, the local search approach seeks to raise the quality of S. A local modification entails the removal of one vertex from the set and the addition of another vertex that is not currently present in the set. The new set is then compared to the old set, and if the new set has a lower domination number, it replaces the old set as the dominant set. Up until there is no more improvement possible, the process is repeated.

To increase the effectiveness of the local search strategy, it can also be paired with other approximation methods. The local search approach, for instance, can be used with the greedy algorithm, the LP-relaxation method, or the randomized rounding method. It is possible to get better results than those from utilizing any one of the methods alone by combining the local search method with other approximation methods [26].

The efficiency of the local search approach for estimating the dominance number of social networks was examined by Kundu et al. (2017). They demonstrated that for a number of social networks, the local search strategy can yield results that are within 10% of the ideal solution [?].

2.1.4 Study Methods

For this study, I used the brute force and local search method to find the domination number of the graph. After I found the domination number of the graph, I would start looking for the patterns that were generated from the dominating number and the dominating set. From there I looked at the patterns that arose vertically throughout a ladder graph. I then found the many different ways that this dominating number could be represented in the graph to find the dominion. Many of these then built upon each other to find the dominating number and dominion of the next graph. The domination number for each grid graph was already known. I used this information to check my own work.

2.2 Domination Number of Ladder Graphs

When starting this research, I strictly looked at ladder graphs. Since all ladder graphs are a subset of grid graphs, I then extended the work to look at grid graphs. The domination number of grid graphs is known with various mathematicians having worked within this field. Goncalves, Pinlou, Rao, and Thomasse worked out the domination number of grid graphs for various configurations [16]. T.Y. Chang also theorized in his doctoral thesis about the domination number for grid graphs [11].

Starting with ladder graphs, I found the domination number for ladder graphs L_1 to L_{10}. At the same time, I was trying to find the dominion of the graphs to generate a pattern. We will look at the dominion of the graphs in Chapter 3.

Figure 2.1 shows the ladder graphs of L_1 to L_4. These graphs show the basic graph without a recognized domination number. Looking at the ladder graph of L_1

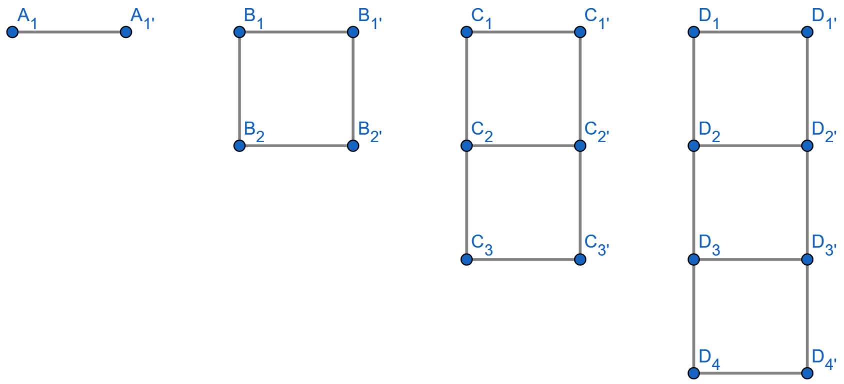


Figure 2.1: Ladder Graphs of L_1 to L_4

we can see the naming of each vertex starting with A_1 on the left side. The vertex to the right is the connected vertex within the rung of the ladder and thus names $A_{1'}$. To help differentiate, L_2 is labeled with a different letter which continues as we add more rungs to the ladder.

Figure 2.2 showcases the a dominating set of the ladder graphs L_1 to L_4. In L_1, we can see the domination number is 1, as only one vertex is marked. However, in L_2, both B_1 and $B_{2'}$ are marked. If just B_1 was marked, then only B_1, $B_{1'}$, and B_2 would be marked or next to a marked vertex. This would not include all of the vertices in the graph.

Ladder graphs L_5 through L_{10} are shown in figure 2.3, and a domination set for each ladder graph L_5 to L_{10} is shown in Figure 2.4. At this point, I started to look for a pattern with the domination number. The domination numbers for the first ten ladder graphs, (L_1 to L_{10}), are shown in Table 2.1. In this table we see that a pattern

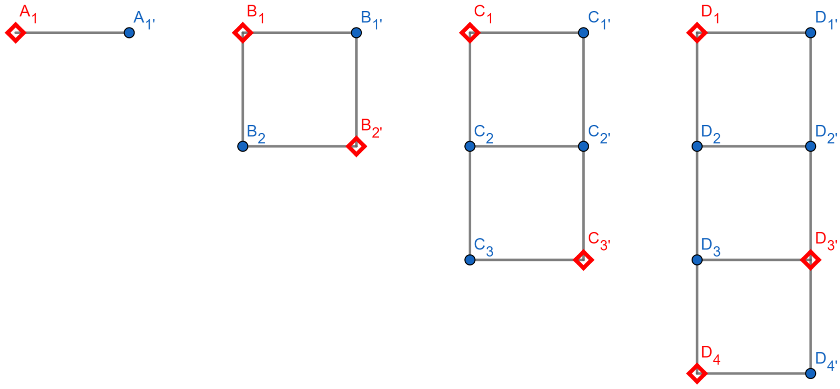


Figure 2.2: Ladder Graphs of L_1 to L_4 with colored vertices showing a dominating set

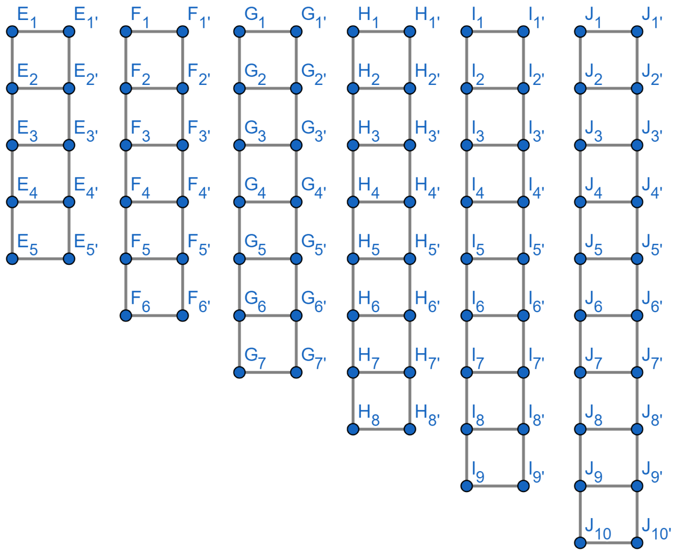


Figure 2.3: Ladder Graphs of L_5 to L_{10}

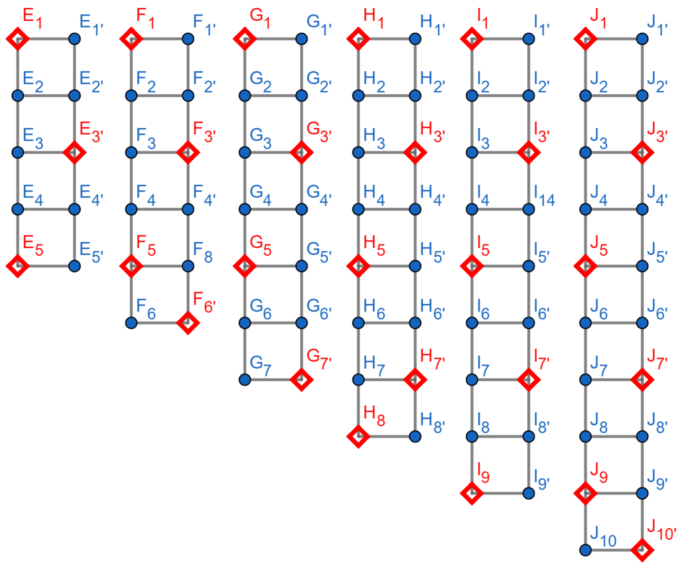


Figure 2.4: Ladder Graphs of L_5 to L_{10} with colored vertices showing a dominating set

emerges to find the vertices, edges, and domination number, γ, of a ladder graph. These result in the following equations where n is the number of rows in a (vertical) ladder graph.

$$V = 2n \tag{2.1}$$

$$E = 3n - 2 \tag{2.2}$$

$$\gamma(L_n) = \lfloor \frac{n}{2} + 1 \rfloor \tag{2.3}$$

The domination number ends up being a floor function based on the number of rows. For instance, a ladder graph with four rows, has a γ of 3 because $\lfloor \frac{4}{2} + 1 \rfloor = 3$. Likewise, a ladder graph of five rows also has a γ of 3 because $\lfloor \frac{5}{2} + 1 \rfloor = 3$.

Ladder Graph	**Vertices**	**Edges**	γ
L_1	**2**	**1**	**1**
L_2	**4**	**4**	**2**
L_3	**6**	**7**	**2**
L_4	**8**	**10**	**3**
L_5	**10**	**13**	**3**
L_6	**12**	**16**	**4**
L_7	**14**	**19**	**4**
L_8	**16**	**22**	**5**
L_9	**18**	**25**	**5**
L_{10}	**20**	**28**	**6**

Table 2.1: Domination Number of first ten ladder graphs

Chapter 3 Computing Dominion of Ladder Graphs

When computing the dominion of a ladder graph, I first started with the basic sets from figures 2.2 and 2.4. From there, I would try out different configurations. This allowed me to see the patterns that arose, and extrapolate from there.

3.1 Dominion of Ladder Graphs L_1 through L_{10}

Definition 3.1.1 (Definition: Dominion)**.** The dominion of a graph denoted by ζ counts all minimum dominating sets.

Because a given graph may have multiple γ-sets, this chapter focuses on a natural but fundamental question: *How many γ-sets does a given graph have?* To answer this question, the notion of **dominion** was first introduced by Allagan in 2021 [1].

The *dominion (number)* of a graph $G = (V, E)$, denoted by $\zeta(G)$, is the number of its γ-sets. In other words,

$\zeta(G) := |\{S : S \text{ is a dominating set in } G \text{ and } |S| = \gamma\}|$. For instance, given a complete graph $G = K_n$, it is obvious that each $v \in V(G)$ covers the remaining $n-1$ vertices of G. Therefore $\zeta(G) = n$ while $\gamma(G) = 1$. Also, for a star graph $G = K_{1,n}$ of order $n + 1 \geq 3$, it is easy to see that $\gamma(G) = 1 = \zeta(G)$ since only the central vertex can cover all of the remaining $n \geq 2$ vertices.

A ladder graph with one row, L_1 as shown in Figure 3.1, only has two vertices; because these two vertices are linked, one vertex can cover or 'monitor' the other in case there is a local intrusion, given two possible monitoring options. In practice, an adversary will not be able to distinguish between a target or a monitor. All sites in the network are considered 'target'. Thus, any vertex can represent a 'target' site or 'monitor' site. Throughout this chapter, because dominion counts all possible

Figure 3.1: Dominion of ladder graph, L_1; $\zeta = 2$

minimum dominating sets, we illustrate such different sets, and count the number of possibilities: this notion is often referred to as an *ennumeration*. To distinguish sequentially between a target vs a monitor, we color a 'target' site red while a monitor site is colored blue. For instance, in Figure 3.1, A_1 is the target while A_2 is the monitor. One site is sufficient to monitor the other, giving $\gamma = 1$. Also, there are two possibilities of selecting the target (A_1 or B_1) vs the monitor (A_2 or B_2), giving $\zeta = 2$.

A ladder graph with two rows, L_2, has four vertices, so there are more configurations that could generate a domination number. The domination number for L_2 is 2 and the dominion is 6. We can see the dominion represented in Figure 3.2. Since there are two vertices that need to be colored in L_2 and any two will work, this is a simple combination problem of 4 options choose 2.

A ladder graph with three rows and six vertices, L_3, is also special in that all odd row ladder graphs but L_3 have a $\zeta = 2$. These can be represented by alternating each column as you go down and coloring every other row. For instance, in Figure 3.3 we see the dominion of L_3. In the A figure, we see A_1 and A_6 colored. However, in the B figure, we see B_2 and B_5 colored. Because of how few rows the L_3 graph has, we also are able to have one more graph representing the domination number as show in figure C. All of these graphs have a domination number of 2. We will see other odd

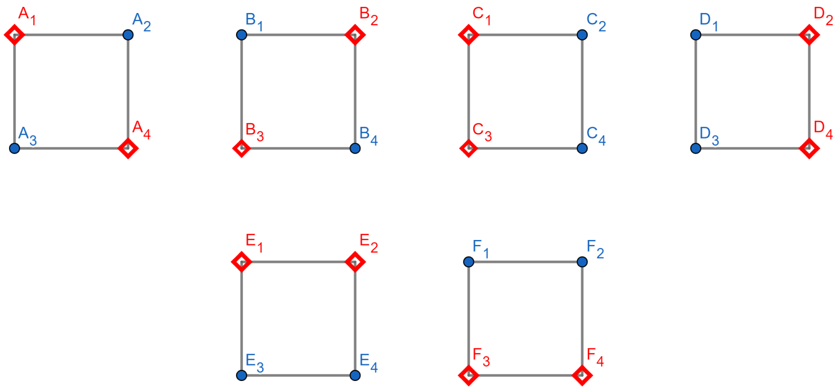


Figure 3.2: Dominion of ladder graph, L_2; $\zeta = 6$

row ladder graphs showcasing this alternating pattern in Figures 3.4, 3.5, and 3.6 as well each representing L_5, L_7, and L_9 respectively.

Once I found the pattern of the dominion of odd row graphs, I looked to find a pattern for the even row ladder graphs. I started by creating all of the different versions representing the domination number of L_4. These are represented in Figure 3.7. The top row of figures, A through F, are the originals and the bottom row, G through L, are the mirrors of these. The ladder graph L_4 has a dominion of 12. At this point, I had not found a pattern yet, so I went on to L_6 and L_8.

I started to notice a pattern emerging in the dominion sets of L_6 and L_8. Figure 3.8 shows L_6, and in this figure, you can see many similarities between it and Figure

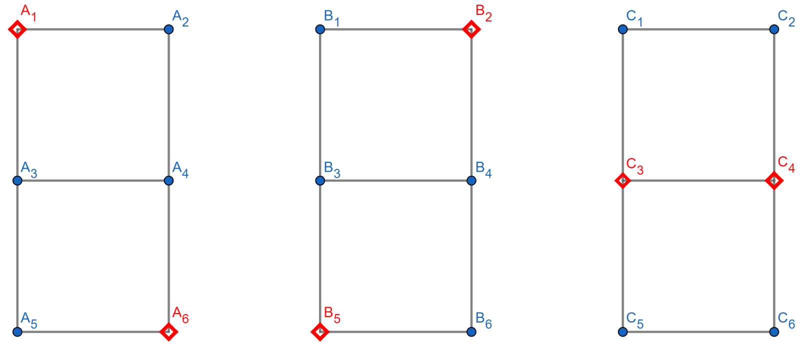


Figure 3.3: Dominion of ladder graph, L_3; $\zeta = 3$

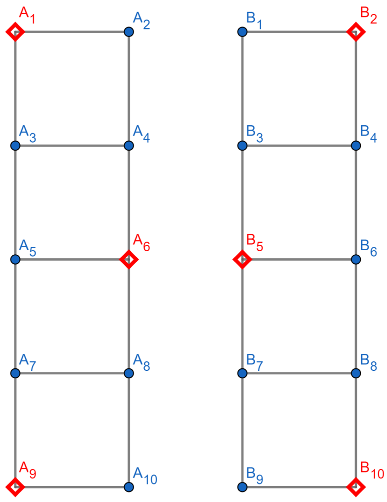


Figure 3.4: Dominion of ladder graph, L_5; $\zeta = 2$

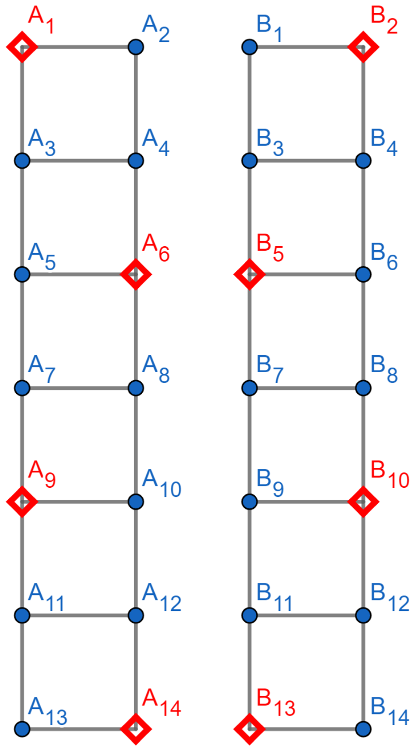


Figure 3.5: Dominion of ladder graph, L_7; $\zeta = 2$

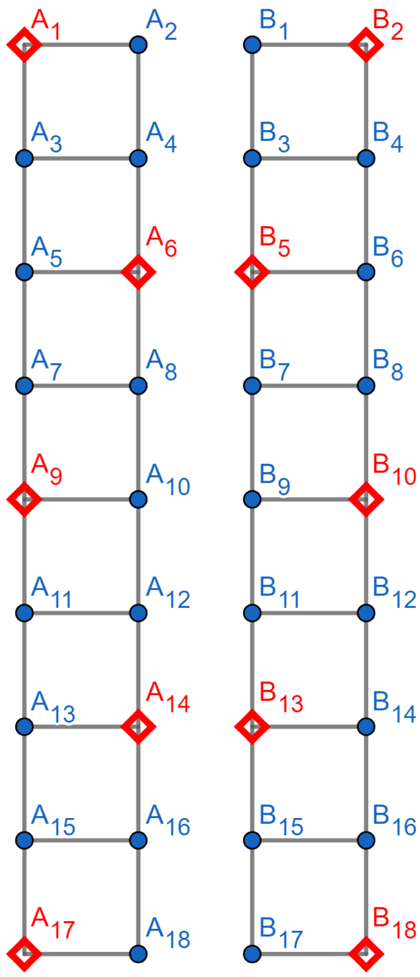


Figure 3.6: Dominion of ladder graph, L_9; $\zeta = 2$

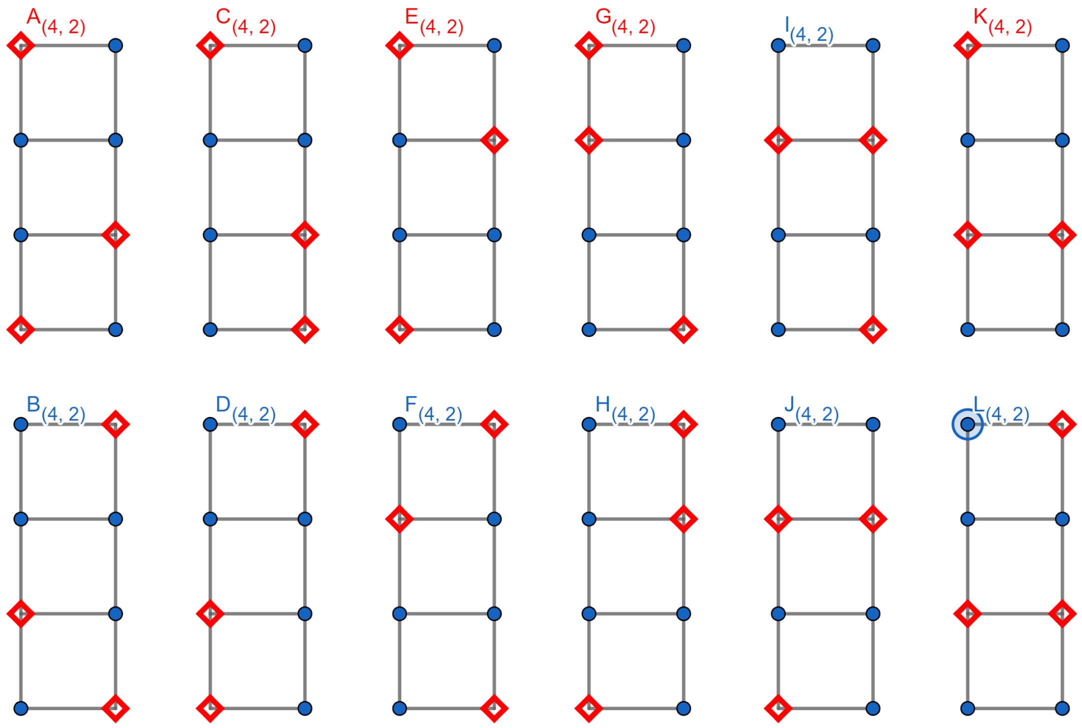


Figure 3.7: Dominion of ladder graph, L_4; $\zeta(L_4) = 12$

3.7 showcasing L_4. For example, the first graph, $A_{(6,2)}$ looks similar in that the first four rows are exactly the same as $A_{(4,2)}$ in Figure 3.7; however, for $A_{(6,2)}$ in L_6, we have added two more rows and colored the vertex on the alternating column of the colored vertex in the last row of $A_{(4,2)}$ in L_4. This works for A, C, E, G, I and their mirrors. It does not work for $K_{(4,2)}$ or its mirror $L_{(4,2)}$ because these do not have a dominating vertex in the bottom row. Therefore, these can not be used to generate graphs in the dominion set of L_6. Instead, we look at any more that can be added.

I noticed that we could add graphs K, M, O and their mirrors to the dominating set by adding two more rows and a colored vertex to the top of A, C, K and there mirrors. This does not work for graphs E and G because they would repeat patterns. This does not work for graph I because there is not a dominating vertex in the top row (similar to K not working in the last version).

Ladder graph L_6 has an additional entry in the dominating set due to the graph being a double of 3. We can see in graph Q that graph C in Figure 3.3 is doubled to make this graph. This does not continue to happen. Ladder graph L_9 would need to have a domination number of 6 to repeat the pattern and $\gamma(L_9) = 5$ for this graph. I also looked if this would work for L_{12} as it is double L_6, but that would need a domination number of 8 to work and $\gamma(L_{12}) = 7$ here.

In ladder graphs L_8 and L_{10}, the pattern continues. These graphs can be seen in Figures 3.9 and 3.10 respectively. We will add two more rows to each of these graphs and alternate the dominating vertex to the opposite column in the bottom row. The two graphs without a dominating vertex in the bottom row can not be used in new sets, but we can add six new graphs. By continuing this pattern of subtracting two graphs and adding six more, we have a net change of positive four graphs. From what I can infer, this pattern will continue indefinitely. Therefore, we can generate the piecewise function below to find the dominion of any ladder graph or any grid graph with either 2 columns or 2 rows labeled as either $G_{(2,n)}$ or $G_{(m,2)}$. In table 3.1, we can see the addition of the dominion, ζ.

Ladder Graph	**Vertices**	**Edges**	γ	ζ
L_1	**2**	**1**	**1**	**2**
L_2	**4**	**4**	**2**	**6**
L_3	**6**	**7**	**2**	**3**
L_4	**8**	**10**	**3**	**12**
L_5	**10**	**13**	**3**	**2**
L_6	**12**	**16**	**4**	**17**
L_7	**14**	**19**	**4**	**2**
L_8	**16**	**22**	**5**	**20**
L_9	**18**	**25**	**5**	**2**
L_{10}	**20**	**28**	**6**	**24**

Table 3.1: Domination Number and Dominion of first ten ladder graphs

Following the heuristic algorithm as elaborated through the previous cases, we conclude the following:

Theorem 3.1.1. *If ζ denotes the dominion of a Ladder on $n \geq 2$ vertices, L_n, then*

$$\zeta(L_n) = \begin{cases} 3, & \text{if } n = 3 \\ 2, & \text{if } n = 2k+1, \; k > 1 \\ 6 & \text{if } n = 2 \\ 12 & \text{if } n = 4 \\ 17 & \text{if } n = 6 \\ 2n+4 & \text{if } n = 2k, \; k \geq 4 \end{cases} \tag{3.1}$$

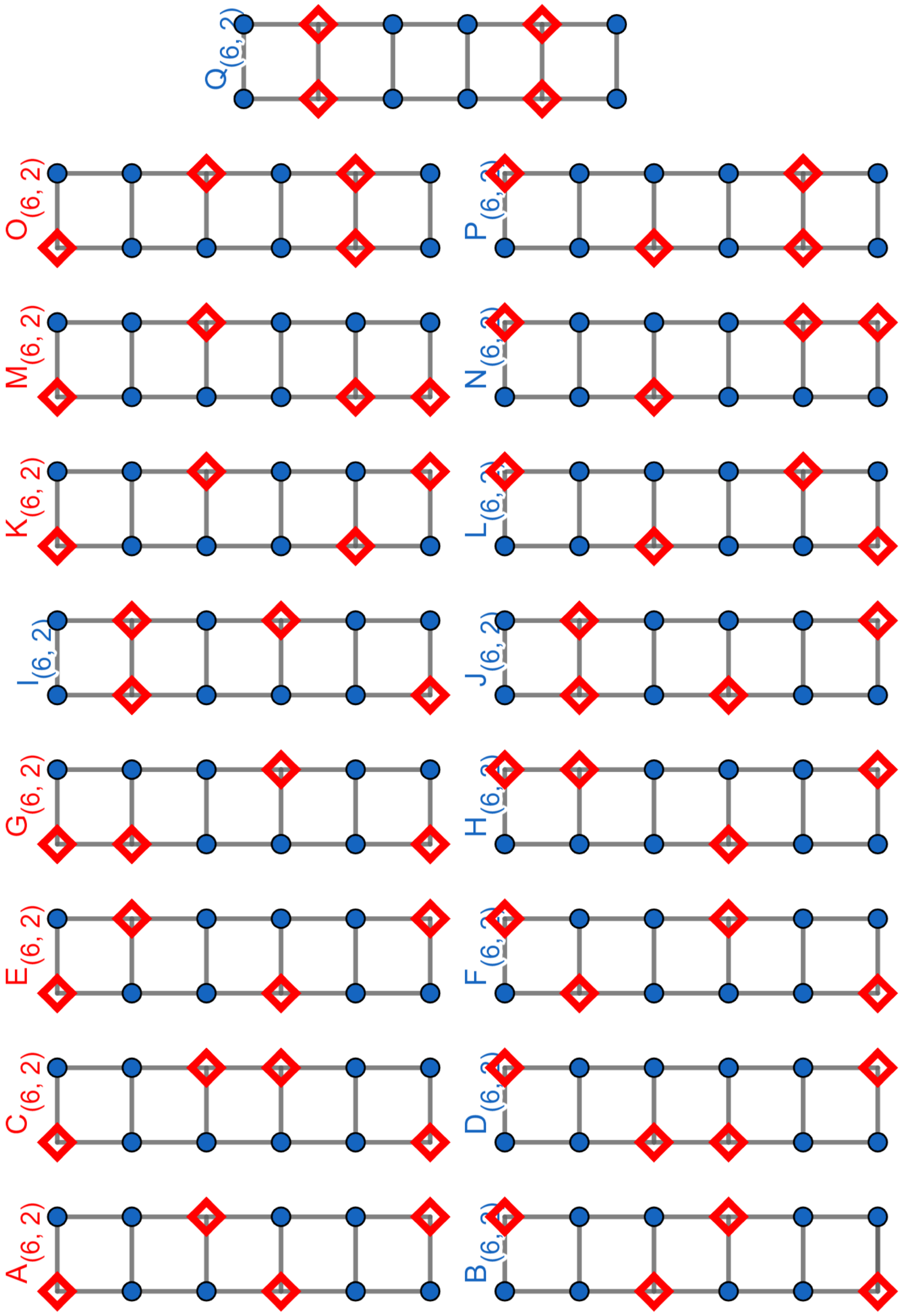


Figure 3.8: Dominion of ladder graph, L_6; $\zeta(L_6) = 17$

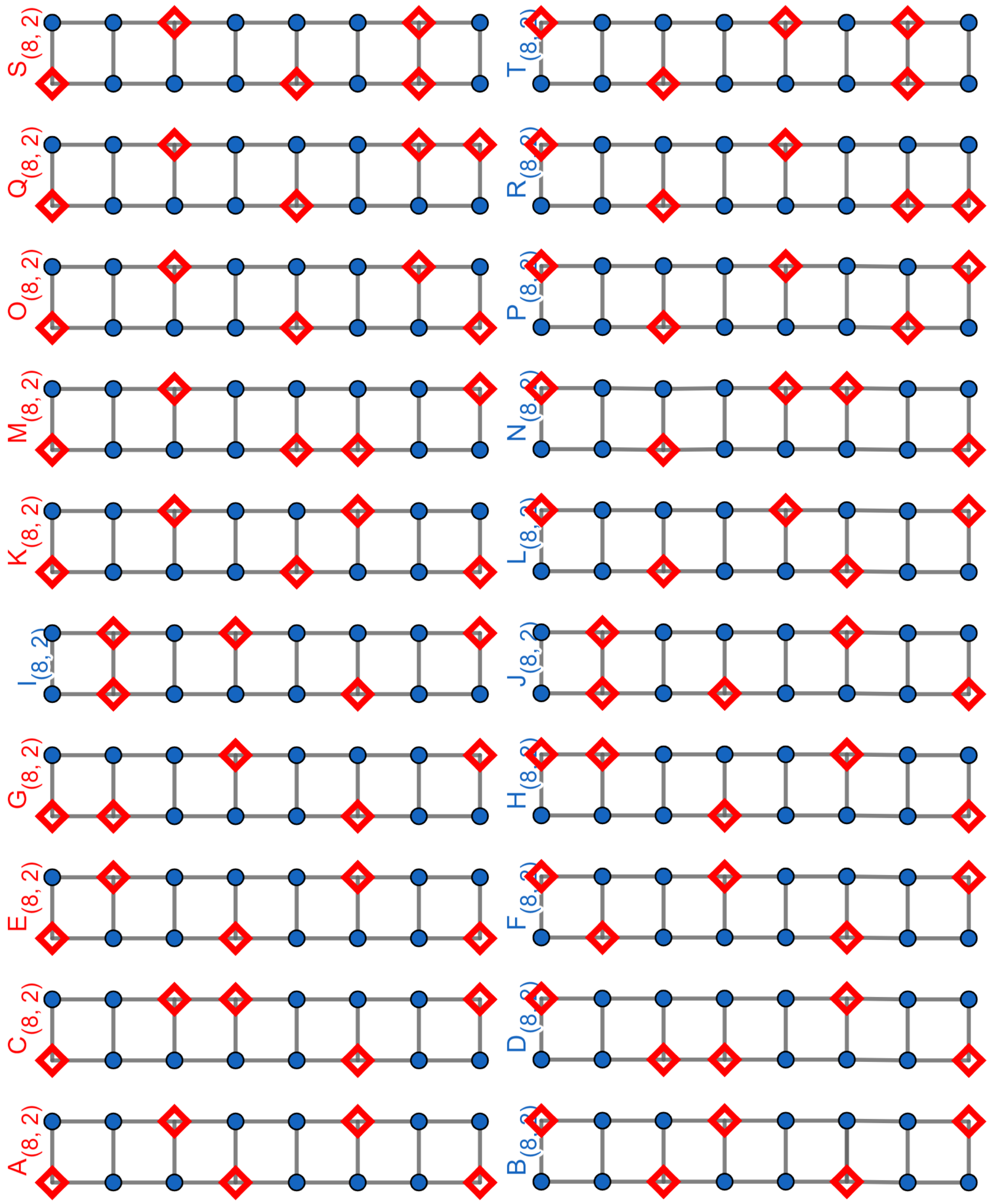


Figure 3.9: Dominion of ladder graph, L_8; $\zeta(L_8) = 20$

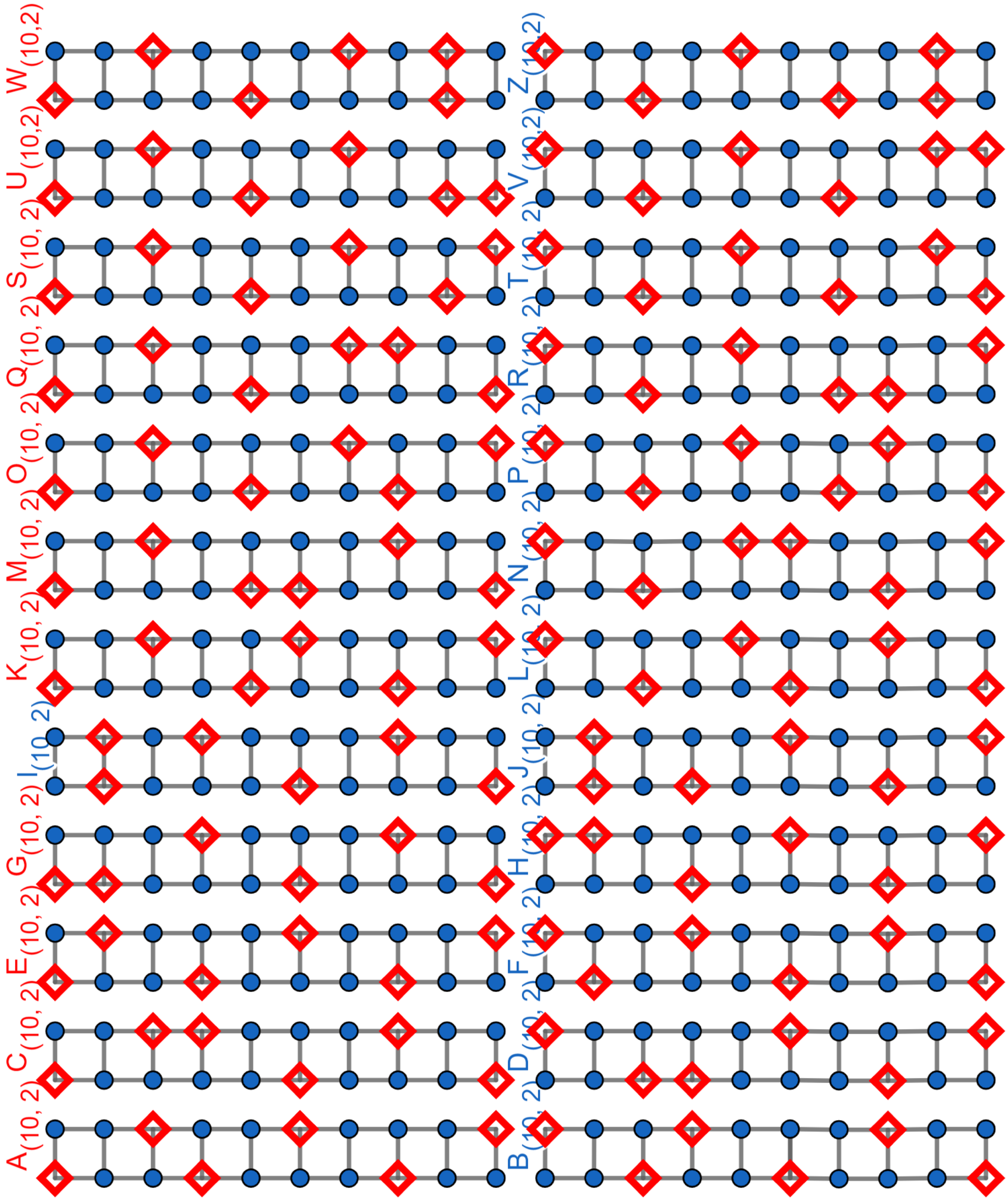


Figure 3.10: Dominion of ladder graph, L_{10}; $\zeta(L_{10}) = 24$

Chapter 4 Extension to Other Grid Graphs

Now we can look at other grid graphs. Since we worked with 2 column grid graphs of n rows, these graphs would also work for 2 row grid graphs of n columns. Therefore, if we were to create a table where we show the dominion based on the number of rows or columns, we could fill it out as shown in Table 4.1 below.

In Figures 4.1 and 4.3, we show dominion set of each path graph which is either one row and n columns or the rotation of this to one column and n rows. In a path graph every 3 additional rows or columns creates a dominion of 1. Otherwise, one could argue based on the first 10 iterations that to get the ζ for the $3n+1$ row, you would add the previous three ζ numbers.

We can also see a grid graph of three columns and three rows in Figure 4.3. There is not apparent pattern starting and further research into $G_{(3,n)}$ or bigger would need to be completed to see what dominions could be elicited.

Rows x Columns	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
1	**1**	**2**	**1**	**4**	**3**	**1**	**8**	**4**	**1**	**13**
2	**2**	**6**	**3**	**12**	**2**	**17**	**2**	**20**	**2**	**24**
3	**1**	**3**	**10**							
4	**4**	**12**								
5	**3**	**2**								
6	**1**	**17**								
7	**8**	**2**								
8	**4**	**20**								
9	**1**	**2**								
10	**13**	**24**								

Table 4.1: Dominion of Some Grid Graphs

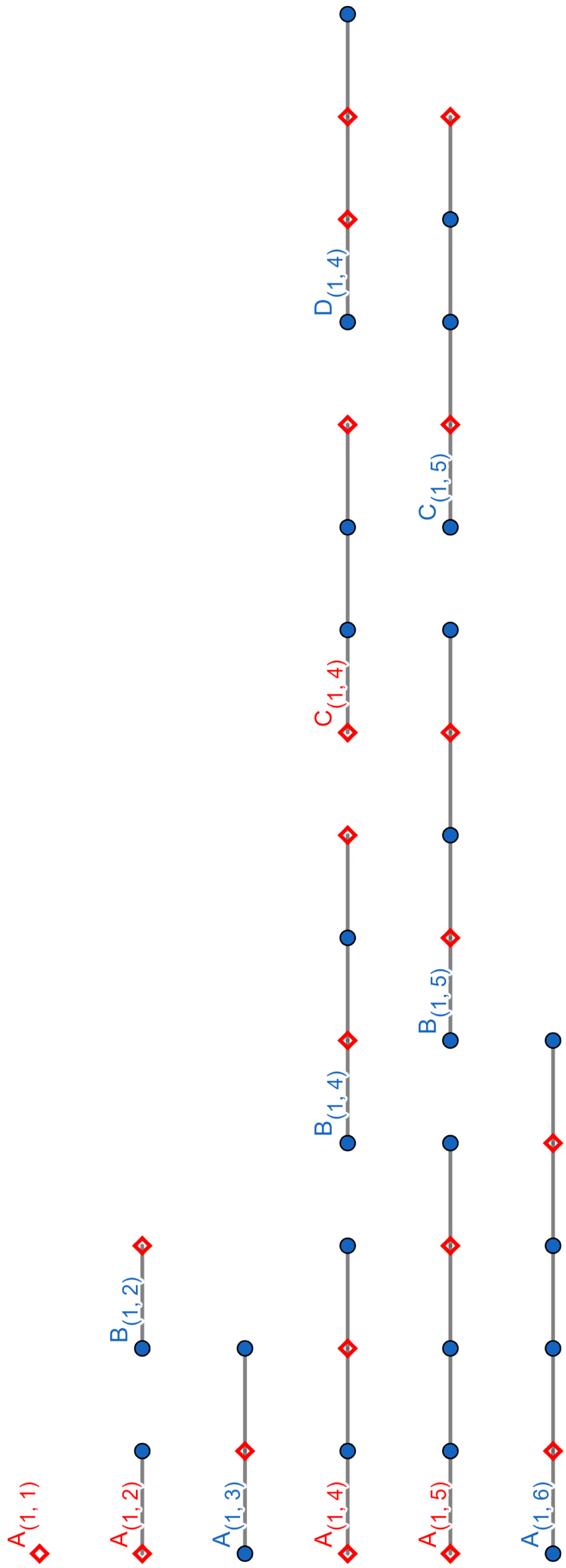


Figure 4.1: Dominion of path graphs, $G_{(1,1)} - G_{(1,6)}$

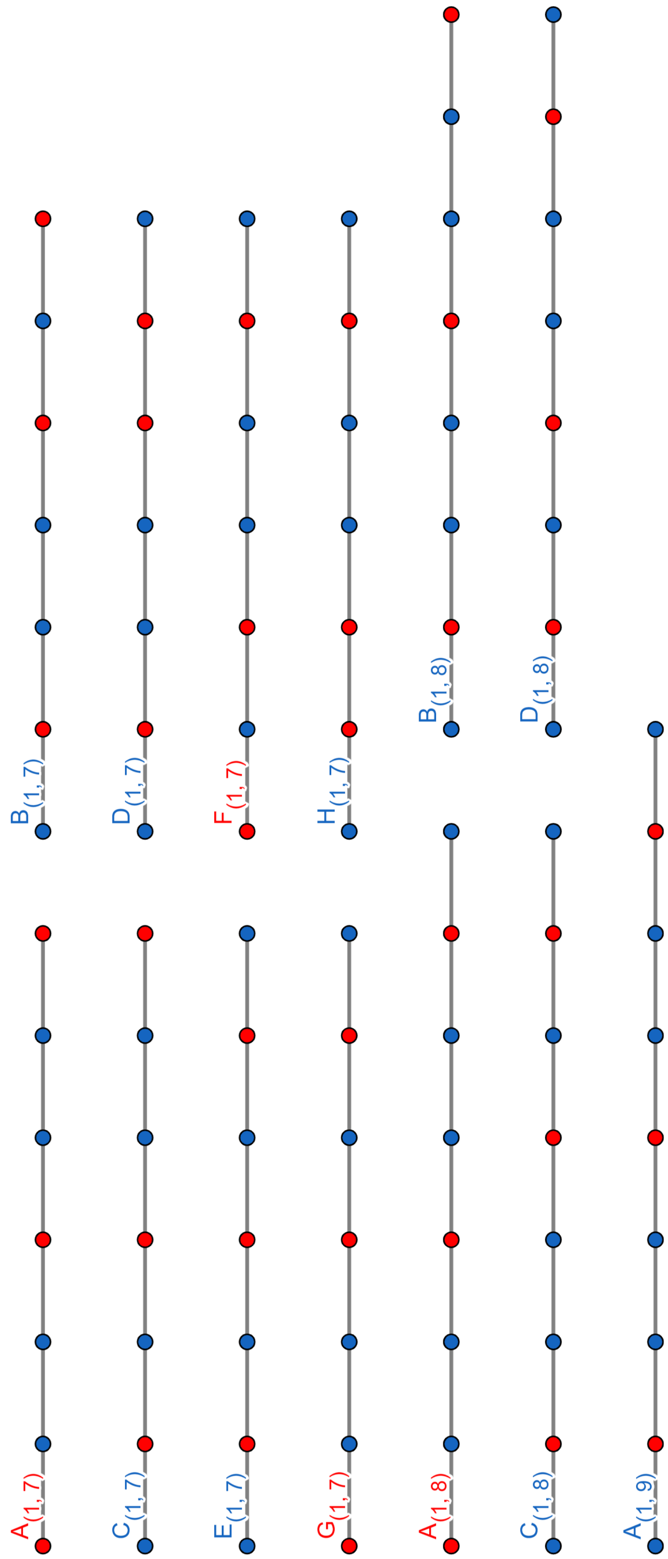


Figure 4.2: Dominion of path graphs, $G_{(1,7)} - G_{(1,9)}$

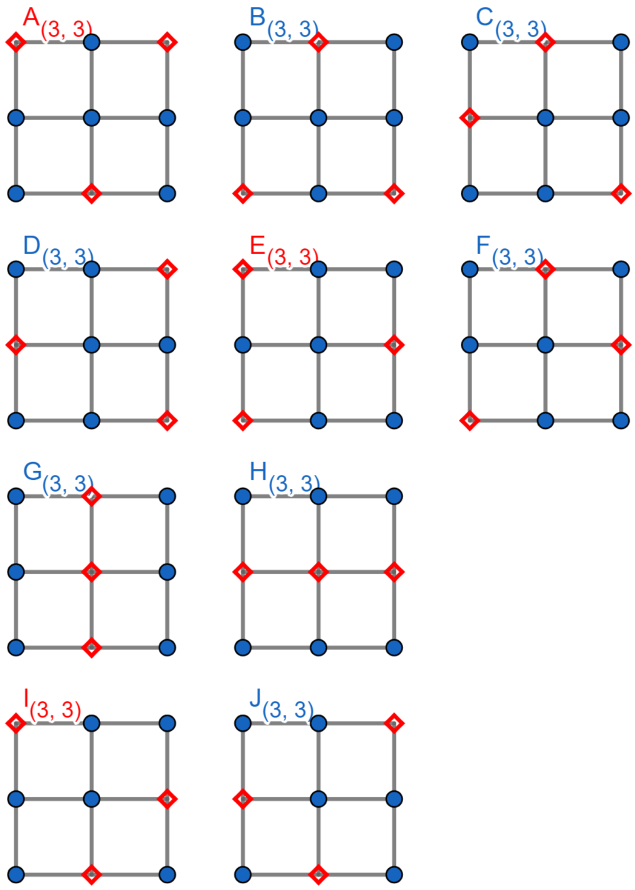


Figure 4.3: Dominion of path graph, $G_{(3,3)}$; $\zeta(G_{(3,3)}) = 10$

Chapter 5 Conclusion and Further Research

There are many applications to studying the domination number and dominion of graphs. One particular relevance is to the field of network security. Network security has become a major problem in recent years as a result of the expansion of the internet and the growing reliance on digital communication. Malware, phishing, and denial-of-service assaults are just a few of the many types of attacks that networks are susceptible to. Understanding a network's structure and its weaknesses is crucial for ensuring its security. Important network security concepts like dominion and domination can clarify how resilient a network is against attacks.

The domination number of a network is the minimum number of nodes that need to be monitored or controlled in order to ensure that all nodes in the network are protected or monitored. As a result, even if an attacker takes over a node outside the domination, the security of the entire network will remain intact. For instance, the dominating set in a computer network might be made up of intrusion detection or firewall nodes. Network administrators are able to stop assaults from propagating over the entire network by keeping an eye on or managing these crucial nodes.

On the other hand, the maximum number of nodes that may be simultaneously watched over or under the control of an attacker is known as a network's dominion number. If the dominion number is low, a hacker might potentially jeopardize the security of the entire system by taking over a sizable piece of the network with ease. For instance, in a social network, the dominion set might include a collection of powerful individuals with the capacity to affect the thoughts and behaviors of other members. The network may be infected with misleading information or propaganda if an attacker seizes control of this organization.

It is essential to comprehend a network's domination and dominion numbers in order to assess its susceptibility to assaults and create efficient security measures to defend against them. Network administrators can devise plans to stop an attacker from taking over the dominion set by evaluating this data to pinpoint the crucial nodes that need to be safeguarded or kept an eye on. For instance, they might build access controls to prevent unauthorized access to these nodes or employ encryption to secure data transit between crucial nodes.

Moreover, the concept of domination and dominion numbers can also help in designing a resilient network architecture. Network administrators can design redundancy and backup systems to guarantee that the network continues to function even if one or more nodes are compromised by determining the minimal number of essential nodes that need to be monitored or controlled. Administrators can also create networks that limit the harm an attacker can do by compartmentalizing certain areas of the network by determining the maximum number of nodes that can be controlled by an attacker.

The lower the domination number, the more cost effective the system becomes. However, as stated, the lower the dominion, the more susceptible a system becomes. The configuration of the physical network, or rather the graph type that is used, can also be factored into the cost as well. Therefore, there becomes this perfect number or ratio in which the domination number is low, the dominion is high, and the configuration of the network is ideal for the system.

There are other applications in network security including, but not limited to, wireless sensor networks, social network analysis, distributed systems, and network optimization. We will discuss some of the benefits to studying domination and dominion below.

5.1 Wireless Sensor Networks

A form of network known as a wireless sensor network (WSN) uses several small sensors spread out over a vast area to gather and send data. The coverage problem, or the difficulty of ensuring that the entire area is monitored by the sensors, is a significant issue with WSNs. In order to solve this issue, domination gives the minimum number of sensors necessary for comprehensive coverage of a given area. On the other hand, the dominion gives the number of different location options one has to place the required sensors.

Sensors in WSNs are typically placed at random or according to a preset arrangement. With this approach, it is challenging to guarantee thorough coverage of the area, as some parts might be overlooked or undersampled. Researchers have suggested using the ideas of dominion and domination number to determine the bare minimum of sensors necessary for full coverage in order to solve this issue. In WSNs, the sensors can be modeled as nodes in a graph, and the coverage problem can be formulated as a domination problem. Also in WSNs, the domination number corresponds to the minimum number of sensors required for complete coverage, where each sensor covers its own location and the locations of its neighboring nodes.

By using the concepts of dominion and domination numbers, researchers can design algorithms to deploy the minimum number of sensors required for complete coverage of an area. This can lead to significant cost savings and improved efficiency in WSNs, as fewer sensors are required to achieve complete coverage [17].

5.2 Social Network Analysis

A social network's most influential users should be identified for a variety of reasons. For example, finding influencers in social media marketing is essential for promoting goods and services. Finding powerful people can be crucial in political campaigns for gaining support for a candidate or a cause. Additionally, in the context of social network analysis, pinpointing prominent people can offer information on the composition, behavior, and development of the network.

Domination number is a graph-theoretic concept that can be used to identify influential individuals in social networks. These concepts are based on the idea of node centrality, which measures the importance of a node in a network. If a node covers many other nodes, it means that it is influential and can control a significant portion of the network. The domination of a node in a social network, on the other hand, is defined as the minimum number of nodes required to cover all the nodes not directly connected to the node. In other words, if a node has a high dominion number, it means that it is influential and can control a significant portion of the network, even if it is not directly connected to many nodes [7].

By using the concepts of dominion and domination number, researchers can identify the most influential individuals in a social network. In a social media network, for example, those with the greatest domination are probably the most well-liked and connected, while those with the highest dominion number are probably the most persuasive and influential when it comes to influencing ideas and behavior.

Also, finding powerful people in a social network can be helpful in a variety of contexts, including political campaigns, public relations, and marketing. Organizations can broaden their reach, engage more people, and have a greater impact by focusing on these individuals. The development of tactics for influencing people can also be influenced by an understanding of the structure and behavior of social networks, which can offer insights into how ideas, views, and behaviors propagate throughout society.

5.3 Distributed Systems

In a distributed system, multiple nodes work together to achieve a common goal, and the efficient coordination of these nodes is essential for achieving optimal performance and scalability. Dominion and domination number can be used to design algorithms that efficiently coordinate the actions of nodes in a distributed system. Specifically, these concepts can be used to identify nodes that are most critical for achieving the system's goals, and to allocate resources and responsibilities among nodes in an optimal way.

The domination, for instance, can be used to determine the bare minimum of active and reachable nodes required to guarantee the availability and dependability of a distributed computing system. Similar to this, the dominion can be used to determine the bare minimum of nodes that must coordinate their actions in order to complete a certain activity, like data aggregation or decision-making [24].

Moreover, fault-tolerant algorithms for distributed systems can be created using the notions of domination and dominion. The system can ensure that crucial nodes are replicated or backed up in case of failures by recognizing nodes with high dominion or domination numbers, hence enhancing the system's resilience and availability.

In addition to these uses, dominion and domination can be used to create algorithms for distributed systems' load balancing, routing, and resource allocation. These methods can optimize the distribution of resources and responsibilities, lessen congestion and bottlenecks, and increase the overall performance and scalability of the system by taking the centrality of nodes into account [7].

5.4 Network Optimization

A crucial technique for comprehending the behavior and structure of complicated networks is network analysis. In several disciplines, including social science, computer science, biology, and physics, complex networks are pervasive. These networks frequently feature a lot of nodes and edges, and they also frequently have complex topological characteristics including small-worldness, community structure, and scale-free degree distribution [27].

Identification of network communities and hierarchies can be aided by domination and dominion number. Nodes are grouped together into communities or clusters that have similar characteristics or roles in many complex networks. These groups frequently have their own internal hierarchy and structure. We can ascertain a node's function within the network's community and hierarchy by identifying its most important nodes. For instance, in biological networks, recognizing nodes with high dominion or dominance numbers might assist in comprehending the modular structure of the network and the various functional functions played by various genes or proteins.

5.5 Further Research

Further research should delve into what is that perfect ratio of domination to dominion to configuration in order to create the most secure network. Also, further research would explore different types of graphs and their effectiveness in a system. In this research, I only looked at one small subset of graphs, but this work could be extended to any graphs within discrete mathematics. More work can also be done in the verification of these and other graphs by creating a code to check for dominion of graphs.

In conclusion, the concepts of domination and dominion numbers are critical in ensuring the security of a network. They help network administrators understand the vulnerabilities of a network, identify critical nodes that need to be protected or monitored, and develop effective security measures to prevent attacks. By understanding these concepts, network administrators can design more resilient networks that are less vulnerable to attacks and can respond more effectively to security breaches.

Bibliography

[1] ALLAGAN, J. A., BOBGA, BENKAM, *Dominion of Some Graphs*, International Journal of Mathematics and Computer Science, 16 no. 4 (2021), 1709–1720

[2] ACHARYA, B. D. AND GILL, M. K. *On the Index of Gracefulness of a Graph and the Gracefulness of Two-Dimensional Square Lattice Graphs.* Indian J. Math. 23, 81-94. 1981.

[3] APPEL, K. AND HAKEN, W. *Every Planar Map is Four-Colorable, II: Reducibility.* Illinois J. Math. 21, 491-567, 1977.

[4] ARTMAN, BENNO, *Euclid – The Creation of Mathematics.* New York, Berlin, Heidelberg: Springer, 1999.

[5] BARTHELEMY, MARC, *Morphogensis of Spatial Networks.* Springer, 2018. (6-11)

[6] BONATO, ANTHONY; LOZIER, MARC; MITSCHE, DIETER; PÉREZ-GIMÉNEZ, XAVIER; PRALAT, PAWEL. (2015). *The Domination Number of On-line Social Networks and Random Geometric Graphs.* 10.1007/978-3-319-17142-5_14.

[7] BONDY, J. A. AND MURTY, U. S. R. *Graph Theory with Applications,* North-Holland, New York, 1976.

[8] BRON, C.; AND KERBOSCH, J. (1973). *Algorithm 457: finding all cliques of an undirected graph.* Communications of the ACM, 16(9), 575-577.

[9] CARLSON, STEPHAN C., *"Königsberg bridge problem".* Encyclopedia Britannica, 30 Jul. 2010, https://www.britannica.com/science/Konigsberg-bridge-problem. Accessed 20 March 2023.

[10] Casado, A.; Bermudo, S.; Lopez-Sanchez, A.D.; and Sanchez-Oro, J. (2023) *An iterated greedy algorithm for finding the minimum dominating set in graphs.* Mathematics and Computers in Simulation, Vol 207, 41-58. doi.org/10.1016/j.matcom.2022.12.018.

[11] Chang, T.Y. (1992) *Domination Numbers of Grid Graphs, Ph.D. Thesis.* Department of Mathematics, University of South Florida.

[12] Chartrand, G.; Kapoor, S. F.; and Kronk, H. V. *The Many Facets of Hamiltonian Graphs.* Math. Student 41, 327-336, 1973.

[13] Cormen, Thomas H.; Leiserson, Charles E.; Rivest, Ronald L.; Stein, Clifford. (2001). *Introduction to Algorithms (Second ed.).* MIT Press and McGraw–Hill.

[14] Corominas-Murtra, B; Fuchs, B.; Thurner, S. (2014) *Detection of the elite structure in a virtual multiplex social system by means of a generalized k-core*

[15] Galby, Esther; Lima, Paloma; and Ries, Bernard. (2019) *Reducing the Domination Number of Graphs via Edge Contractions.* Leibniz International Proceedings in Informatics (LIPIcs). 41:1–41:13 https://doi.org/10.4230/LIPIcs.MFCS.2019.41

[16] Gonçalves, D.; Pinlou, A.; Rao, M.; and Thomassé, S. (2011). *The domination number of grids.* SIAM Journal on Discrete Mathematics, 25(3), 1443–1453. https://doi.org/10.1137/11082574

[17] Hassani Karbasi, Amir; Ebrahimi Atani, Reza. (2013). *Application of dominating sets in wireless sensor networks.* International Journal of Security and its Applications. 7. 185-202.

[18] Johnson, D. S. (1974). *Approximation algorithms for combinatorial problems.* Journal of computer and system sciences, 9(3), 256-278.

[19] Johnsonbaugh, Richard, *Discrete Mathematics.* Prentice Hall, 2008.

[20] Klasing, Ralf; Laforest, Christian. (2004), *Hardness results and approximation algorithms of k-tuple domination in graphs,* Information Processing Letters, 89(2) 75–83.

[21] Knoebel, A.; Laubenbacher, R.; Lodder, J.; and Pengelley, D., *Mathematical Masterpieces: Further Chronicles by the Explorers.* Springer Verlag, New York, 2007.

[22] Michaels, J. G.; Rosen, K., *Applications of discrete mathematics.* McGraw-Hill, 1992.

[23] Rajarajan, M.; Lestas, M.; and Rakocevic, V. (2018). *Critical Node Identification for Accessing Network Vulnerability, A Necessary Consideration.*

[24] Sun, Xuemei; Yongxin Yang; and Maode Ma. (2019). *Minimum Connected Dominating Set Algorithms for Ad Hoc Sensor Networks.* Sensors 19, no. 8: 1919. https://doi.org/10.3390/s19081919

[25] Truta, T.M.; Campan, Alina; Beckerich, Matthew. (2018). *Efficient Approximation Algorithms for Minimum Dominating Sets in Social Networks.* International Journal of Service Science, Management, Engineering, and Technology. 9. 1-32. 10.4018/IJSSMET.2018040101.

[26] Vazirani, V. V. (2001). *Approximation algorithms.* Springer Science and Business Media.

[27] Watts DJ, Strogatz SH. (June 1998). *Collective dynamics of 'small-world' networks.* Nature. 393, 440–442. https://doi.org/10.1038/30918

[28] Williamson, D. P.; and Shmoys, D. B. (2011). *The design of approximation algorithms.* Cambridge University Press.

[29] Wilson, Robin, *History of Graph Theory* from Handbook of Graph Theory. CRC Press. 17 Dec 2013.

[30] Windsor, A. *Planar Graphs.* Boost C++ libraries. Retrieved March 16, 2023, from

https://www.boost.org/doc/libs/1_66_0/libs/graph/doc/planar_graphs.html. 2007.

[31] Wu, J.; Li, H. (2001) *A Dominating-Set-Based Routing Scheme in Ad Hoc Wireless Networks.* Springer Science+Business Media B.V., Formerly Kluwer Academic Publishers B.V.