THE PENNSYLVANIA STATE UNIVERSITY
SCHREYER HONORS COLLEGE

DEPARTMENT OF MATHEMATICS

OPTIMIZATION APPLIED IN GRAPH THEORY: ZERO-FORCING DIAMETER

JAKOB LOEDDING
SPRING 2022

A thesis
submitted in partial fulfillment
of the requirements
for a baccalaureate degree
in Mathematics
with honors in Mathematics

Reviewed and approved* by the following:

Thomas Cameron
Professor of Mathematics
Thesis Supervisor

Daniel Galiffa
Professor of Mathematics
Honors Adviser

*Electronic approvals are on file in the Schreyer Honors College.

Abstract

Zero-forcing is a coloring game on a graph in which an initial set of vertices is colored gray while the remaining vertices are colored white. An iterative color change rule, where some vertices have the ability to force others to change their color to gray, is then applied until no more vertices may be forced. A zero-forcing set of a graph G is defined as an initial set of gray vertices in which the remaining white vertices are forced gray after some number of iterations of the color change rule. The minimum cardinality of a zero-forcing set is defined as the zero-forcing number of G.

In this thesis, we define a new graph parameter called the zero-forcing diameter, which quantifies the minimum intersection of two minimum zero-forcing sets of a graph with respect to its zero-forcing number. Furthermore, we present the numerical bounds of the zero-forcing diameter and find its value, with theoretical proof, for specific graph families. We then introduce an integer programming model for calculating the zero-forcing number of a graph. Finally, we build upon this model to develop our own integer program for calculating the zero-forcing diameter of a graph.

Table of Contents

List of Figures

Acknowledgements

First, I would like to thank my thesis supervisor, Dr. Thomas Cameron, for introducing me to zero-forcing and optimization, teaching me how to approach research with rigor, and challenging me while being patient with my progress. I would also like to thank my honors advisor, Dr. Daniel Galiffa, for answering all of my questions and keeping me on track during my time at Schreyer. Next, I would like to thank Dr. Michael Brown, Dr. Amy Carney, and Mrs. Stephanie Gummerson for orchestrating an exciting and inclusive Schreyer Honors experience at Penn State Behrend. Additionally, I would like to thank Dr. Michael Rutter and Professor Andrew George for not only contributing to my academic succes, but also providing guidance and mentorship during my pursuit of a career in data science.

Last, and certainly not least, I would like to thank my family. Thanks to my mother for urging me to pursue applied mathematics and constantly challenging me to step outside my comfort zone. If not for her, this thesis may have never existed. Thanks to my father for always supporting me in all my academic and career endeavors. Finally, thanks to my brothers—Mitch, Nate, and Luke—for always being there to help me decompress on weekends during the stress and rigor of college.

Chapter 1

Introduction

1.1 Overview

This thesis begins by exploring the concepts of graph theory, zero-forcing, and optimization. In Chapter 1, we introduce notation and relevant definitions regarding simple graphs and zero-forcing, a coloring game on a graph. We continue by outlining the standard and skew zero-forcing rules and providing the zero-forcing numbers, defined in Section 1.3, for common graph families described in Section 1.2. Furthermore, we provide a foundation of knowledge concerning integer programming in Section 1.4.

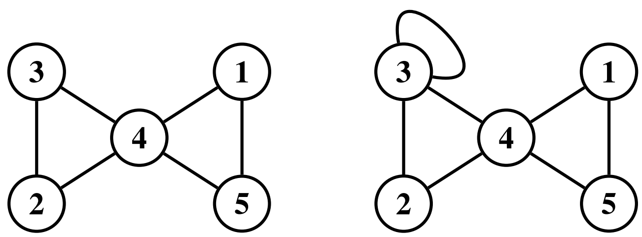
In Chapter 2, we formulate a new graph parameter called the zero-forcing diameter. Moreover, we derive its numerical bounds and calculate its value, with respect to the standard and skew zero-forcing rules, for different graph families. We utilize the concepts of integer programming in Chapter 3 by introducing a model, developed by Brimkov et al. in [1], for computing the zero-forcing number of a simple graph with respect to the standard zero-forcing rule. Furthermore, we alter Brimkov's model to calculate the zero-forcing number of a simple graph with respect to the skew zero-forcing rule.

Finally, in Chapter 3, we build upon Brimkov's model to develop our own integer program for computing the zero-forcing diameter of a simple graph with respect to the standard zero-forcing rule. Similarly, we develop another integer program to calculate the zero-forcing diameter with respect to the skew zero-forcing rule. We then conclude this thesis in Chapter 4 by summarizing our accomplishments, explaining the significance of our work, and discussing possible avenues for future research.

1.2 Introduction to Graph Theory

In this section, we introduce notation and vocabulary regarding graph theory that will be used throughout this thesis. We cover fundamental concepts necessary to understand zero forcing; however, see [2] and [3] to gain a further understanding of graph theory. We define a *graph* $G = (V, E)$ to be a pair containing a non-empty finite set of vertices, V, and a multi-set of two-element multi-sets of V, denoted by E, which we call edges. Generally, we refer to the *vertex* and *edge sets* of G as $V(G)$ and $E(G)$ respectively. Two vertices $u, v \in V(G)$ are said to be *adjacent*, or *neighbors*, provided that $\{u, v\} \in E(G)$.

The work in this thesis is concerned only with simple graphs. We define a simple graph to be a graph such that there are no loops nor multi-edges. Moreover, we denote the collection of all simple graphs as $\mathbb{G}$. Given a graph $G \in \mathbb{G}$ and $u, v \in V(G)$, we have that $\{u, u\} \notin E(G)$ and the multiplicity of $\{u, v\} \in E(G)$ is less than or equal to one. The *order* of G is equal to the total number of vertices in G, i.e., $|V(G)|$. The *size* of G is the total number of edges in G, i.e., $|E(G)|$. Unless otherwise stated, we typically use the letters n and m to refer to the order and size of G respectively. We now demonstrate the definitions discussed thus far through an example of two graphs $K \in \mathbb{G}$ and $K' \notin \mathbb{G}$ shown in Figure 1.1. Note that $V(K) = \{1, 2, 3, 4, 5\}$ and $E(K) = \{\{1, 4\}, \{1, 5\}, \{2, 4\}, \{2, 3\}, \{3, 4\}, \{4, 5\}\}$, $n = 5$ and $m = 6$. Additionally, $V(K') = V(K)$ and $E(K') = E(K) \cup \{\{3, 3\}\}$. We can see that $K' \notin \mathbb{G}$ since K' has a loop, meaning that K' contains and edge from one vertex to itself. That is, $\{3, 3\} \in E(K')$.



Graph $K \in \mathbb{G}$ Graph $K' \notin \mathbb{G}$

Figure 1.1: Simple and non-simple graphs of order 5

For a graph $G \in \mathbb{G}$, we define the *neighborhood* of a vertex u, denoted by $N(u)$, to be the set of all vertices $v \in V(G)$ that are adjacent to u. The *degree* of a vertex is equal to the cardinality of its neighborhood. We say that $deg(u) = |N(u)|$ for some vertex $u \in V(G)$. A vertex $v \in V(G)$ is said to be *isolated* if $deg(v) = 0$. Moreover, a vertex $v \in V(G)$ is a *leaf* provided that $deg(v) = 1$.

We now define various graph operations. Given graphs $G_1, G_2 \in \mathbb{G}$, the *union* of G_1 and G_2, denoted as $G_1 \cup G_2$, is defined to be a graph H where $V(H) = V(G_1) \cup V(G_2)$ and $E(H) = E(G_1) \cup E(G_2)$. Similarly, the *intersection* of G_1 and G_2, denoted as $G_1 \cap G_2$, is defined to be a graph H where $V(H) = V(G_1) \cap V(G_2)$ and $E(H) = E(G_1) \cap E(G_2)$. Let F be the set of all edges that join each vertex in $V(G_1)$ to every vertex in $V(G_2)$. It follows that the *join* of G_1 and G_2, denoted as $G_1 \vee G_2$, is a graph H for which $V(H) = V(G_1) \cup V(G_2)$ and $E(H) = E(G_1) \cup E(G_2) \cup F$. Given a graph $G \in \mathbb{G}$, the *complement* of G, denoted by $\overline{G}$, is a graph with vertex set $V(\overline{G}) = V(G)$ and edge set $E(\overline{G})$ where $\{u, v\} \in E(\overline{G})$ if and only if $\{u, v\} \notin E(G)$ for all $u, v \in G$.

A *subgraph* of a graph G is defined to be a graph H such that $V(H) \subseteq V(G)$ and $E(H) \subseteq E(G)$. A *path*, denoted by P_n, is a non-empty graph of order n such that $V(P_n) = \{p_1, p_2, ..., p_n\}$ and $E(P_n) = \{\{p_1, p_2\}, \{p_2, p_3\}, ..., \{p_{n-1}, p_n\}\}$ where each p_i is distinct. Furthermore, G is *connected* if there exists a path, which is a subgraph of G, between any two distinct vertices in G. A graph that is not connected is said to be *disconnected*.

We conclude this section by defining and formulating proper notation for various families of simple graphs that are frequently used throughout this thesis. A graph K_n is *complete* if all the vertices in K_n are pairwise adjacent. The *empty graph* is the complement of the complete graph. In other words, the empty graph, denoted by $\overline{K}_n$, is a graph for which each vertex is isolated, i.e., $deg(u) = 0$ for all $u \in V(\overline{K}_n)$. If $n \geq 3$ and we have a path P_n, then we define a *cycle*, C_n, to be a graph where $V(C_n) = V(P_n)$ and $E(C_n) = E(P_n) \cup \{\{p_n, p_1\}\}$. A *tree* is a connected graph that has no subgraphs taking the form of a cycle. A *bipartite graph*, denoted as B_{n_1,n_2}, is a graph such that $V(B_{n_1,n_2}) = V_1 \cup V_2$ where V_1 and V_2 are nonempty vertex sets, $V_1 \cap V_2 = \emptyset$, $|V_1| = n_1$, and $|V_2| = n_2$. The edges of B_{n_1,n_2} join exactly one vertex of V_1 to exactly one vertex of V_2. It follows that a *complete bipartite graph* K_{n_1,n_2} is a bipartite graph for which there exists an edge $(u, v) \in E(K_{n_1,n_2})$ for each vertex $u \in V_1$ and each vertex $v \in V_2$. Lastly, the *star graph*, denoted by S_n, is a tree that takes the form of the complete bipartite graph $K_{n,1}$. For examples of some of the graph families discussed in this section, see Figure 1.2.

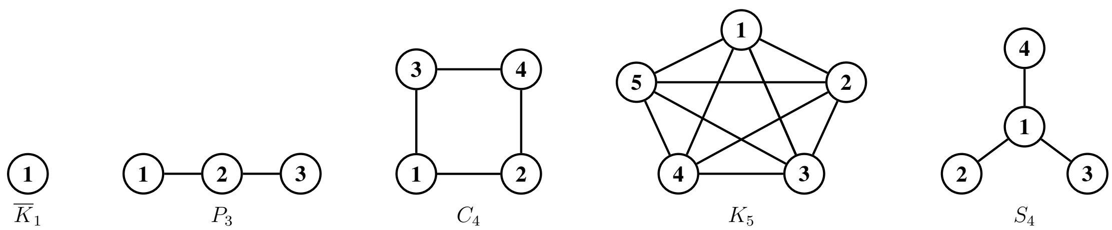


Figure 1.2: Examples of simple graph families

1.3 Introduction to Zero-forcing

The application of zero-forcing played a vital role in the work, published in 2013, done by Burgarth et al. in [4] . A group of mathematicians and computer scientists were studying the dynamics of systems on networks. They utilized zero-forcing to investigating linear and quantum controllability for systems evolving on networks. In their work, they used zero-forcing sets of graphs to determine whether dynamical systems were controllable. For further reading see [4]. Since we have noted the importance of zero-forcing by describing one of its applications in quantum controllability, we now introduce the theory behind zero-forcing.

We now summarize zero-forcing, as it is explained in [5], while introducing our own notation. Zero-forcing is a coloring game on a graph in which an initial set of vertices is colored gray while the remaining vertices are colored white. Each vertex is classified into binary categories, either gray or white. Some color change rule, where vertices have the ability to force other vertices to change their color to gray at distinct time steps, is then applied iteratively until no more vertices may be forced gray. For a graph $G \in \mathbb{G}$, we refer to the *initial coloring* of G as the set of initial gray colored vertices, $C \subseteq V(G)$. Let $t \in \mathbb{N}$ and $C^{[t]}$ represent the set of gray colored vertices at time step t, that is, after t applications of the color change rule. Note that $C^{[0]} = C$. Additionally, let $C^{[t^*]}$ denote the *final coloring* of G in which no more forcings are possible from applying the color change rule. Since G is finite, there exists a $t^* \geq 0$ such that $C^{[t^*+k]} = C^{[t^*]}$ for all $k \in \mathbb{N}$. In other words, our color change rule cannot perform any additional forcings at higher time steps. If the entire graph of G is colored gray at time step t^*, i.e., $C^{[t^*]} = V(G)$, then we say that C is a *zero-forcing set* of G. The cardinality of the smallest possible zero-forcing set is defined to be the *zero-forcing number* of G, denoted as $Z(G) = \min\{|C| : C^{[t^*]} = V(G)\}$. It is common that $Z(G)$ and $Z_{-}(G)$ are used to differentiate between the zero-forcing number of a graph using the standard and skew zero forcing rules respectively. We further demonstrate these concepts by introducing both the standard and skew zero-forcing rules.

1.3.1 Standard Zero-forcing Rule

The *standard zero-forcing rule* states that a gray vertex u will force a white vertex v to change colors if and only if v is the only white vertex in the neighborhood of u. It is important to note that only gray vertices are allowed to perform forcings in the standard zero-forcing rule. Consider an example in which we demonstrate the application of the standard zero-forcing rule to a path of order 5 in Figure 1.3. Since $C^{[4]} = C^{[5]}$, then the final coloring of P_5 is $C^{[4]}$. Furthermore, the set $\{1\}$ is a zero forcing set of P_5 since $C^{[4]} = V(P_5)$. Hence, we have that $Z(P_5) = 1$.

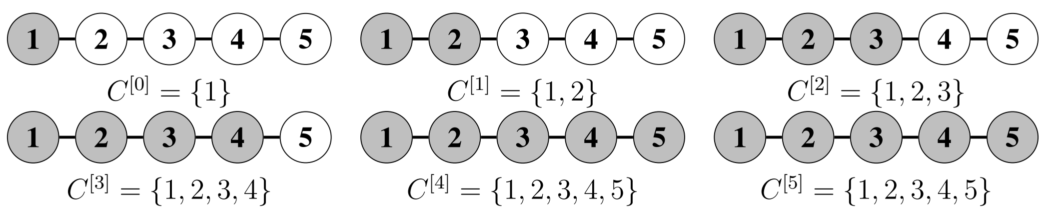


Figure 1.3: A standard zero-forcing set for P_5

When utilizing the standard zero-forcing rule, we define a *zero-forcing chain* to be a list of chronological forcings that outline the order in which vertices force other vertices. For example, the forcing chain associated with the example in Figure 1.3 is as follows

$$(1) \to (2) \to (3) \to (4) \to (5).$$

This zero-forcing chain shows that vertex 1 forces vertex 2, then vertex 2 forces vertex 3, then vertex 3 forces vertex 4, and vertex 4 forces vertex 5. Typically, we need more than one forcing chain to outline all the forcings of a simple graph. Each vertex in the zero-forcing set will have its own forcing chain. We only have one forcing chain associated with the example in Figure 1.3 because $Z(P_5) = 1$, i.e., any zero forcing set of P_5 contains only one vertex. Additionally, note that each vertex listed in a zero-forcing chain must be colored gray. This will be important when discussing the skew zero-forcing rule. If a vertex in a zero-forcing chain performs no forcings from one time step to the next, that vertex is repeated in the chain. For example, in Figure 1.4, vertex 2 cannot perform any forcings from time step 0 to time step 1. Hence, the corresponding entry in the zeor-forcing chain is $(2) \to (2)$.

Now that we have seen an example of a zero-forcing set, it is important to note that not every initial coloring of vertices equates to a zero-forcing set. For example, consider the initial coloring of P_5 in Figure 1.4. Note that the initial coloring of P_5 is the set $\{2\}$. We can see that vertex 2 cannot perform any more forcings by the standard zero-forcing rule. Hence, the final coloring of P_5 is $\{2\} \neq V(G)$ and we do not have a zero-forcing set.

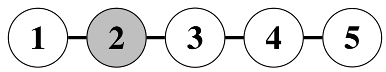


Figure 1.4: A non zero forcing set for a path graph of order 5.

We conclude this introduction to the standard zero-forcing rule by stating the standard zero-forcing numbers of various graph families found in [6].

1. For all empty graphs of order $n \in \mathbb{N}$, we have that $Z(\overline{K}_n) = n$.
2. For $n \geq 2 \in \mathbb{N}$, we have that $Z(P_n) = 1$.
3. Given a cycle of order $n \geq 4 \in \mathbb{N}$, we have that $Z(C_n) = 2$.
4. All complete graphs have $Z(K_n) = n - 1$ for $n \geq 3 \in \mathbb{N}$.
5. Given $n \geq 4 \in \mathbb{N}$, we have that $Z(S_n) = n - 2$.

Lastly, suppose that $G \in \mathbb{G}$ such that $G = \bigcup_{i=1}^{j} G_i$ for $G_1, \ldots, G_j \in \mathbb{G}$, where $j > 1 \in \mathbb{N}$. Additionally, let $\bigcap_{i=1}^{j} V(G_i) = \emptyset$ and $\bigcap_{i=1}^{j} E(G_i) = \emptyset$. It then follows that $Z(G) = \sum_{i=1}^{j} Z(G_i)$. This result holds for both the standard and skew zero-forcing rules. For example, consider the graph in Figure 1.5. In this case, we have $G = S_4 \cup P_2 \cup \hat{P}_2$ with $V(S_4) \cap V(P_2) \cap V(\hat{P}_2) = \emptyset$ and $E(S_4) \cap E(P_2) \cap E(\hat{P}_2) = \emptyset$, where $\hat{P}_2$ denotes the second path graph of order 2 that contains vertices 8 and 7. Consequently, we have that $Z(G) = Z(S_4) + Z(P_2) + Z(\hat{P}_2) = 2 + 1 + 1 = 4$.

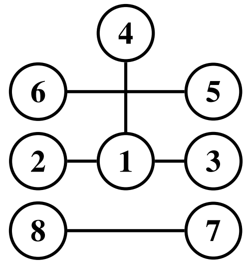
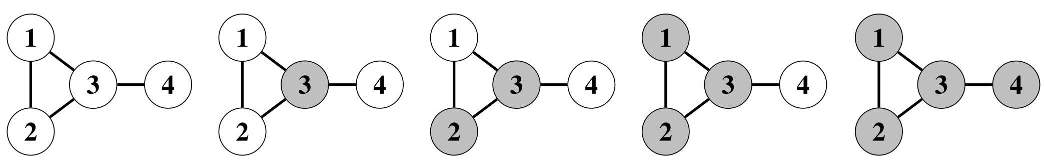


Figure 1.5: Standard zero-forcing number of the disjoint union of two simple graphs

1.3.2 Skew Zero-forcing Rule

The *skew zero-forcing rule* states that a vertex u can force a white vertex v to change its color to gray if and only if v is the only white vertex in the neighborhood of u. This color change rule is similar to the standard zero-forcing rule; however, the key difference is that white vertices can now perform forcings. Consequently, we can have zero-forcing sets that contain no vertices. For example, consider a graph $G \in \mathbb{G}$ of order 4 in Figure 1.6. The final coloring of G is $C^{[4]}$ since no more forcings are possible for any $t > 4$. It follows that since $C^{[4]} = V(G)$, then the set $\emptyset$ is a zero-forcing set of G and $Z_{-}(G) = 0$. It is important that we walk through each time step to understand how skew zero-forcing operates. We can see that at $t = 0$ no vertices are colored gray. Vertex 4 then forces vertex 3 at $t = 1$. At $t = 2$, vertex 1 forces vertex 2. Then vertex 2 forces vertex 1 at $t = 3$. Finally, at $t = 4$, vertex 3 forces vertex 4. Through this demonstration, we can see that forcings are not unique. For example, at $t = 2$, vertex 2 could have forced vertex 1 rather than vertex 1 forcing vertex 2.



$C^{[0]} = \emptyset$ $C^{[1]} = \{3\}$ $C^{[2]} = \{2, 3\}$ $C^{[3]} = \{1, 2, 3\}$ $C^{[4]} = \{1, 2, 3, 4\}$

Figure 1.6: A simple graph with skew forcing number equal to 0.

When working with the skew zero-forcing rule, the concepts of zero-forcing chains no longer apply. In Section 1.3.1 we saw that each zero-forcing chain begins with a vertex in the zero-forcing set and any vertex in the chain must be colored gray. However, in the skew zero-forcing rule, white vertices may also perform forcings. This means that at time step 0, vertices that are not in the zero-forcing set can perform forcings. Moreover, any white vertex that is not currently in the zero-forcing chain at a time step t can perform a forcing. Consequently, gray colored vertices seem to appear with no source of forcing. Hence, there is no concrete way of representing a zero-forcing chain when using the skew zero-forcing rule.

Now that the concepts associated with skew zero-forcing have been demonstrated, we conclude this introduction to zero-forcing by listing the skew zero-forcing numbers of various graph families found in [7].

1. For all empty graphs of order $n \in \mathbb{N}$, we have that $Z_{\text{-}}(\overline{K}_n) = n$.

2. For all $n \geq 2 \in \mathbb{N}$, we have that $Z_{\text{-}}(P_n) = 0$ if n is even and $Z_{\text{-}}(P_n) = 1$ if n is odd.

3. Given a cycle of order $n \geq 3 \in \mathbb{N}$, we have that $Z_{\text{-}}(C_n) = 2$ if n is even and $Z_{\text{-}}(C_n) = 1$ if n is odd.

4. All complete graphs have $Z_{\text{-}}(K_n) = n - 2$ for $n \geq 3 \in \mathbb{N}$.

5. Given $n \geq 4 \in \mathbb{N}$, we have that $Z_{\text{-}}(S_n) = n - 2$.

1.4 Introduction to Integer Programming

This section introduces the idea of integer programming—a special case of linear programming—by summarizing the concepts explained in [8]. A *linear programming* problem is defined to be a problem for which a linear function is maximized or minimized with respect to various linear constraints. These constraints may contain equalities or inequalities. In this thesis, we are concerned primarily with minimization models. Let $\mathbf{b} = (b_1, b_2, ..., b_l)^\intercal$ and $\mathbf{c} = (c_1, c_2, ..., c_k)^\intercal$ be two vectors and define the $l \times k$ matrix of real numbers

$$A = \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1k} \\ a_{21} & a_{22} & \cdots & a_{2k} \\ \vdots & \vdots & \ddots & \vdots \\ a_{l1} & a_{l2} & \cdots & a_{lk} \end{bmatrix}.$$

It follows that the *standard minimum problem*, which we denote by SMIN, takes the canonical form displayed in (1.1a) - (1.1c). We seek a vector $\mathbf{y} = (y_1, y_2, \ldots, y_k)^\intercal$ to

$$\text{minimize} \quad \mathbf{c}^\intercal \mathbf{y} \tag{1.1a}$$

$$\text{subject to} \quad A\mathbf{y} \leq \mathbf{b}, \tag{1.1b}$$

$$\mathbf{y} \geq \mathbf{0}. \tag{1.1c}$$

The function to be minimized, presented in (1.1a), is called the *objective function*. The constraints produced by equations (1.1b) and (1.1c) are called the *main* and *nonnegativity constraints* respectively. The vector $\mathbf{y}$ is a *feasible solution* provided that $\mathbf{y}$ satisfies (1.1b) and (1.1c). If the vector

$\mathbf{y}$ is a feasible solution and is minimal with respect to the objective function in (1.1a), then we say that $\mathbf{y}$ is an *optimal solution* to SMIN. The corresponding minimal value of the objective function (1.1a) is the *optimal value* of SMIN. We denote the set of all feasible and optimal solutions to SMIN by Fes(SMIN) and Opt(SMIN) respectively. Furthermore, we denote feasible and optimal solutions to SMIN as $\bar{\mathbf{y}}$ and $\mathbf{y}^*$ respectively. If the entries of A, $\mathbf{b}$, $\mathbf{c}$, and $\mathbf{y}$ are restricted to be integer values, then the model in (1.1a) - (1.1c) is called an *integer program*. Lastly, if the entries of $\mathbf{y}$ are restricted to values of 0 or 1, then the model in (1.1a) - (1.1c) is referred to as a *binary program*.

We conclude this chapter by demonstrating the concepts of linear programming discussed thus far in the example shown in equations (1.2a) - (1.2d).

$$\text{minimize} \quad 3y_1 + 4y_2 \tag{1.2a}$$

$$\text{subject to} \quad y_1 + 2y_2 \leq 10, \tag{1.2b}$$

$$-y_1 + y_2 \leq -1, \tag{1.2c}$$

$$y_1 \geq 0, y_2 \geq 0. \tag{1.2d}$$

We can see that the objective function corresponds to equation (1.2a). The main constraints are shown in equations (1.2b) - (1.2c) and the nonnegativity constraints are in equation (1.2d). Furthermore, we have that $\mathbf{b} = (10, -1)^\intercal$, $\mathbf{c} = (3, 4)^\intercal$, and

$$A = \begin{bmatrix} 1 & 2 \\ -1 & 1 \end{bmatrix}.$$

Since we are only dealing with two variables, the linear program in (1.2a) - (1.2d) can be solved by graphing the set of points that satisfy the constraints in (1.2b) - (1.2d) as seen in Figure 1.7. We then find the point of this set that minimizes the objective function in (1.2a).

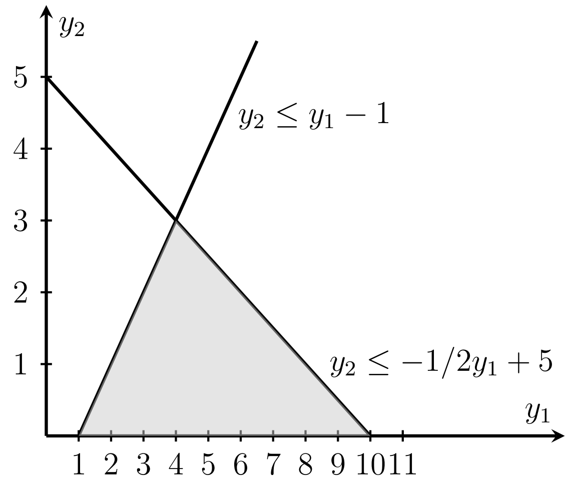


Figure 1.7: Constraint set of a simple linear program.

We can see that any point inside the shaded region or any point that falls along the edges of the shaded region in Figure 1.7 is a feasible solution. The optimal solution is the point (y_1, y_2) that is a feasible solution and minimizes the objective function in (1.2a). This occurs at the point $(1, 0)$. Therefore, we have that $y^* = (1, 0)$ and the optimal value to this example is 4. Lastly, it is important to note that optimal solutions are rarely unique. Due to the simplicity of our example in (1.2a) - (1.2d) our optimal solution was unique and took the form of a single point; however, it is common that there will be many optimal solutions to any given linear or integer program.

Chapter 2

Zero-forcing Diameter

2.1 Introducing Zero-forcing Diameter

Now that we have covered the basics of zero-forcing, we continue by investigating the relationship among different zero-forcing sets of a graph. Specifically, we are interested in quantifying the minimum intersection of two minimum zero-forcing sets of a graph with respect to its zero-forcing number. To do this, we create a new graph parameter called the *zero-forcing diameter*. First, let $S(G)$ denote the set of all standard zero-forcing sets, with minimum cardinality, of a graph $G \in \mathbb{G}$.

Definition 2.1.1. Given a graph $G \in \mathbb{G}$, we define the zero-forcing diameter of G to be

$$D(G) = Z(G) - \min_{C,C' \in S(G)} |C \cap C'|. \tag{2.1}$$

We defined the zero-forcing diameter above with respect to the standard zero-forcing rule; however, the same concept applies for the skew zero-forcing rule. We denote the standard zero-forcing diameter of G as $D(G)$. Similarly, we denote the skew zero-forcing diameter of G as

$$D_{-}(G) = Z_{-}(G) - \min_{C,C' \in S_{-}(G)} |C \cap C'|,$$

where $S_{-}(G)$ is the set of all skew zero-forcing sets of G with minimum cardinality.

We can use this new graph parameter to gain additional insight about different graph families. By calculating the zero-forcing diameter of different graphs, we can make new connections among graphs with vastly different structures. We will see that many graphs with varying structures share a common characteristic, the value of their zero-forcing diameter. Moreover, we will develop integer programming models in Chapter 3 that allow us to compute the zero-forcing diameter of all types of simple graphs. From there, we can further demonstrate different properties of common graph families.

Let us consider an example where we calculate the standard zero-forcing diameter of a simple graph. Figure 2.1 presents a graph $G \in \mathbb{G}$ along with two of its minimum standard zero-forcing sets, namely $C = \{2, 5\}$ and $C' = \{1, 3\}$. Note that $|C| = |C'| = 2$ implies that $Z(G) = 2$. It is also clear that $\min_{C,C' \in S(G)} |C \cap C'| = |\{2, 5\} \cap \{1, 3\}| = 0$. Therefore, $D(G) = 2 - 0 = 2$.

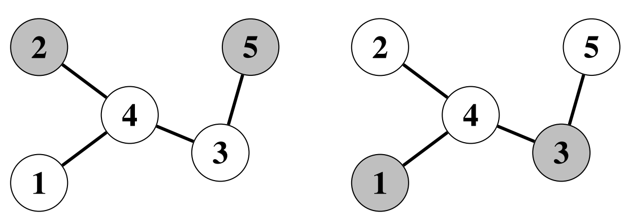


Figure 2.1: Two minimum standard zero-forcing sets of a simple graph

Similarly, we can calculate the zero-forcing diameter of the graph above with respect to the skew zero-forcing rule. Figure 2.2 shows two minimum skew zero-forcing sets, namely $C = \{1\}$ and $C' = \{2\}$, of G. Then we have $|\{1\}| = |\{2\}| = 1$, which implies that $Z_{-}(G) = 1$. Again, it is clear that $\min\limits_{C,C' \in S_{-}(G)} |C \cap C'| = |\{1\} \cap \{2\}| = 0$. Thus, $D_{-}(G) = 1 - 0 = 1$.

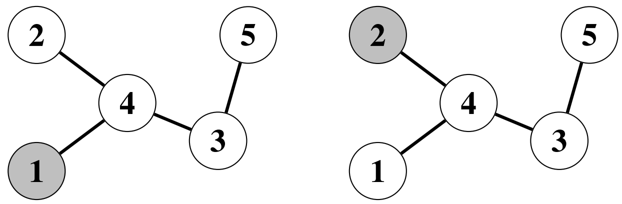


Figure 2.2: Two minimum skew zero-forcing sets of a simple graph

Similar to the zero-forcing number of a graph, the zero-forcing diameter is additive over the union of graphs with pairwise disjoint vertex and edge sets. In more rigorous terms, let $G \in \mathbb{G}$ such that $G = \bigcup\limits_{i=1}^{j} G_i$ for $G_1, \ldots, G_j \in \mathbb{G}$, where $j > 1 \in \mathbb{N}$. Additionally, let $\bigcap\limits_{i=1}^{j} V(G_i) = \emptyset$ and $\bigcap\limits_{i=1}^{j} E(G_i) = \emptyset$. It follows that $D(G) = \sum\limits_{i=1}^{j} D(G_i)$. This result holds for both the standard and skew zero-forcing rules. For example, consider the graph $G \in \mathbb{G}$ in Figure 2.3. We have that $G = P_3 \cup \overline{K}_1$. Note that $V(P_3) \cap V(\overline{K}_1) = \emptyset$ and $E(P_3) \cap E(\overline{K}_1) = \emptyset$. By Propositions 2.2.1 and 2.2.2, proven in Section 2.2, we have that $D(P_3) = 1$ and $D(\overline{K}_1) = 0$. Thus, $D(G) = 1 + 0 = 1$.

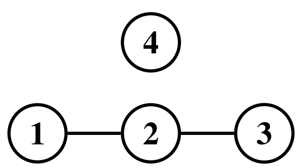


Figure 2.3: The union of two simple graphs with pairwise disjoint vertex and edge sets

Finally, we define the bounds of the zero-forcing diameter, with respect to the standard zero-forcing rule, for a simple graph in Theorem 2.1.1. It is important to note that these bounds are sharp, meaning that we can construct a graph $G \in \mathbb{G}$ such that $D(G)$ is equal to either the lower or upper bound. See Propositions 2.2.1 and 2.2.3 in Section 2.2 to see how we can construct such graphs that satisfy each extreme value of the zero-forcing diameter. Additionally, these bounds hold for the skew zero-forcing rule.

Theorem 2.1.1. *For all graphs $G \in \mathbb{G}$, we have that $0 \leq D(G) \leq \lfloor n/2 \rfloor$ where n is the order of G.*

Proof. Suppose $G \in \mathbb{G}$. By definition, $0 \leq D(G) \leq Z(G)$. Hence, $D(G) \geq 0$. By way of contradiction, suppose $D(G) > \lfloor n/2 \rfloor$, i.e., $D(G) = \lfloor n/2 \rfloor + c$ where $0 < c \leq \lfloor n/2 \rfloor$. Then $Z(G) - \min_{C,C' \in S(G)} |C \cap C'| = \lfloor n/2 \rfloor + c$ and $Z(G) \geq \lfloor n/2 \rfloor + c$. It then follows that $\min_{C,C' \in S(G)} |C \cap C'| \geq c$. Hence, $D(G) \leq \lfloor n/2 \rfloor$, which contradicts our assumption that $D(G) > \lfloor n/2 \rfloor$. Therefore, $0 \leq D(G) \leq \lfloor n/2 \rfloor$ for all $G \in \mathbb{G}$. □

2.2 Standard Zero-forcing Diameters of Common Graph Families

In this section, we present various results concerning the value of the zero-forcing diameter for common graph families with respect to the standard zero-forcing rule.

Proposition 2.2.1. *Given a graph $G \in \mathbb{G}$, we have that $D(G) = 0$ if and only if G is an empty graph.*

Proof. Suppose $G \in \mathbb{G}$ and $D(G) = 0$, i.e., $Z(G) = \min_{C,C' \in S(G)} |C \cap C'|$. This implies there is only one unique zero-forcing set of G, which is only possible if no forcings occur. Hence, our zero-forcing set must include all vertices in G. This implies that G has no edges. Thus, G is an empty graph by definition.

Conversely, suppose G is an empty graph. Since G has no edges by definition, then the only way to force all of its vertices gray is to include them in our zero-forcing set. Hence, $\min_{C,C' \in S(G)} |C \cap C'| = Z(G)$. Thus, $D(G) = 0$. Therefore, $D(G) = 0$ if and only if G is an empty graph. □

Proposition 2.2.2. *Given a path graph P_n, we have that $D(P_n) = 1 \; \forall \; n \geq 2 \in \mathbb{N}$.*

Proof. Suppose P_n is a path graph such that $n \geq 2 \in \mathbb{N}$. Recall that $Z(P_n) = 1$. We can select two minimum zero-forcing sets, C and C', of P_n that contain the leftmost and rightmost vertices of P_n respectively. It then follows that $\min_{C,C' \in S(P_n)} |C \cap C'| = 0$. Therefore, $D(G) = 1 - 0 = 1$ for all $n \geq 2 \in \mathbb{N}$. □

Proposition 2.2.3. *Given $n \in \mathbb{N}$, it is possible to construct a graph $G \in \mathbb{G}$ of order n such that $D(G) = \lfloor n/2 \rfloor$.*

Proof. We break this proof into two cases. First, suppose that n is even. Then let $G = \bigcup_{i=1}^{n/2} G_i$, where $G_1, \ldots, G_{n/2}$ are path graphs of order two with pairwise disjoint vertex and edge sets. We have that $D(G) = \sum_{i=1}^{n/2} D(G_i)$. By Proposition 2.2.2, $D(G) = \sum_{i=1}^{n/2} 1 = n/2$. Hence, $D(G) = \lfloor n/2 \rfloor$.

Now suppose that n is odd. We can now let $G = \left(\bigcup_{i=1}^{\lfloor n/2 \rfloor} G_i \right) \cup \overline{K}_1$, where $G_1, \ldots, G_{\lfloor n/2 \rfloor}$ are path graphs of order two with pairwise disjoint vertex and edge sets. Also note that the vertex in $V(\overline{K}_1)$ is not in the vertex set of any of the paths $G_1, \ldots, G_{\lfloor n/2 \rfloor}$. Then, $D(G) = \sum_{i=1}^{\lfloor n/2 \rfloor} D(G_i) + D(\overline{K}_1)$. By Propositions 2.2.2 and 2.2.1, we have that $D(G) = \sum_{i=1}^{\lfloor n/2 \rfloor} 1 + 0 = \lfloor n/2 \rfloor$. Therefore, given $n \in \mathbb{N}$, we can construct a graph $G \in \mathbb{G}$ of order n such that $D(G) = \lfloor n/2 \rfloor$. □

Proposition 2.2.4. *Given a complete graph K_n, we have that $D(K_n) = 1 \; \forall \; n \geq 3 \in \mathbb{N}$.*

Proof. Suppose K_n is a complete graph such that $n \geq 3 \in \mathbb{N}$. Recall that $Z(K_n) = n - 1$. To find $D(K_n)$, let C be a minimum zero-forcing set of K_n containing $n - 1$ arbitrary vertices in $V(K_n)$. Let C' be another minimum zero-forcing set of K_n containing the vertex in the set $V(K_n) \setminus C$ in addition to $n - 2$ arbitrary vertices in C. Since C and C' share as little vertices as possible and $|C \cap C'| = n - 2$, then $\min_{C,C' \in S(K_n)} |C \cap C'| = n - 2$. Therefore, $D(K_n) = (n - 1) - (n - 2) = 1$ for all $n \geq 3 \in \mathbb{N}$. □

Proposition 2.2.5. *Given a star graph S_n, we have that $D(S_n) = 1 \; \forall \; n \geq 4 \in \mathbb{N}$.*

Proof. Suppose S_n is a star graph such that $n \geq 4 \in \mathbb{N}$ and let $L(S_n) \subseteq V(S_n)$ be the set of leaves in the graph S_n. Note that $Z(S_n) = n - 2$. To find $D(S_n)$, let C be a minimum zero-forcing set of S_n containing $|L(S_n)| - 1$ arbitrary leaves in $L(S_n)$, i.e., $n - 2$ vertices in $V(S_n)$. Similarly, let C' be another minimum zero-forcing set of S_n containing the vertex in the set $L(S_n) \setminus C$ and $|L(S_n)| - 2$ vertices in C. Since C and C' share as little vertices as possible and $|C \cap C'| = |L(S_n)| - 2 = n - 3$, then $\min_{C,C' \in S(S_n)} |C \cap C'| = n - 3$. Therefore, $D(S_n) = (n - 2) - (n - 3) = 1$ for all $n \geq 4 \in \mathbb{N}$. □

Proposition 2.2.6. *Given a cycle graph C_n, we have that $D(C_n) = 2 \; \forall \; n \geq 4 \in \mathbb{N}$.*

Proof. Suppose C_n is a cycle graph such that $n \geq 4 \in \mathbb{N}$. Recall that $Z(C_n) = 2$. Suppose C is a minimum zero-forcing set of C_n containing two adjacent vertices in $V(C_n)$. Since $n \geq 4$, then $|V(C_n) \setminus C| \geq 2$. Then we can let C' be another minimum zero-forcing set of C_n containing two adjacent vertices in $V(C_n) \setminus C$. It then follows that $C \cap C' = \emptyset$, which implies that $\min_{C,C' \in S(C_n)} |C \cap C'| = 0$. Therefore, $D(C_n) = 2 - 0 = 2$ for all $n \geq 4 \in \mathbb{N}$. □

2.3 Skew Zero-forcing Diameters of Common Graph Families

We conclude this chapter by presenting the following results regarding the value of the zero-forcing diameter for common graph families with respect to the skew zero-forcing rule.

Proposition 2.3.1. *If P_n is a path graph of order n, then for all $n \geq 2 \in \mathbb{N}$, we have that $D_{\text{-}}(P_n) = 0$ if n even and $D_{\text{-}}(P_n) = 1$ if n is odd.*

Proof. Suppose P_n is a path graph of order n, where $n \geq 2 \in \mathbb{N}$. First, let n be even. Then $Z_{\text{-}}(P_n) = 0$. Trivially, it follows that $D_{\text{-}}(P_n) = 0$.

Now suppose that n is odd. Note that $Z_{\text{-}}(P_n) = Z(P_n) = 1$. Then by the same logic seen in the proof of Proposition 2.2.2, we have that $D_{\text{-}}(P_n) = 1$. Therefore, $D_{\text{-}}(P_n) = 0$ if n is even and $D_{\text{-}}(P_n) = 1$ if n is odd for all path graphs or order $n \geq 2 \in \mathbb{N}$. □

Proposition 2.3.2. *Given a complete graph K_n, we have that $D_{\text{-}}(K_n) = 2 \;\forall\; n \geq 4 \in \mathbb{N}$.*

Proof. Suppose K_n is a complete graph of order $n \geq 4 \in \mathbb{N}$. Then $Z_{\text{-}}(K_n) = n - 2$. To find $D_{\text{-}}(K_n)$, let C be a minimum zero-forcing set of K_n containing $n-2$ arbitrary vertices in $V(K_n)$. Let C' be another minimum zero-forcing set of K_n containing both vertices in the set $V(K_n) \setminus C$ in addition to $n-4$ arbitrary vertices in C. Since C and C' share as little vertices as possible and $|C \cap C'| = n-4$, then $\min\limits_{C,C' \in S_{\text{-}}(K_n)} |C \cap C'| = n-4$. Therefore, $D_{\text{-}}(K_n) = (n-2) - (n-4) = 2$ for all $n \geq 4 \in \mathbb{N}$. □

Proposition 2.3.3. *Given a start graph S_n, we have that $D_{\text{-}}(S_n) = 1 \;\forall\; n \geq 4 \in \mathbb{N}$.*

Proof. See the proof of Proposition 2.2.5. Since $Z_{\text{-}}(S_n) = Z(S_n) = n-2$, the proof of this result for the skew zero-forcing rule follows exactly the same logic as the proof for the result pertaining to the standard zero-forcing rule. □

Proposition 2.3.4. *If C_n is a cycle graph of order n, then for all $n \geq 3 \in \mathbb{N}$, we have that $D_{\text{-}}(C_n) = 2$ if n is even and $D_{\text{-}}(C_n) = 1$ if n is odd.*

Proof. Suppose C_n is a cycle graph of order n such that $n \geq 3 \in \mathbb{N}$. For the case where n is even, see the proof of Proposition 2.2.6. Since $Z_{\text{-}}(C_n) = Z(C_n) = 2$ when n is even, then we can use the exact same logic seen in Proposition 2.2.6 to prove that $D_{\text{-}}(C_n) = 2$ when n is even. Now let n be odd. Recall that $Z_{\text{-}}(C_n) = 1$ since n is odd. Suppose that C is a minimum zero-forcing set of C_n containing one arbitrary vertex in $V(C_n)$. Since $n \geq 3$, then $V(C_n) \setminus C$ is nonempty. So we can let C' be another minimum zero-forcing set of C_n containing one arbitrary vertex in $V(C_n) \setminus C$. It then follows that $C \cap C' = \emptyset$, which implies that $\min\limits_{C,C' \in S_{\text{-}}(C_n)} |C \cap C'| = 0$. Hence, $D_{\text{-}}(G) = 1 - 0 = 1$ when n is odd. Therefore, $D_{\text{-}}(C_n) = 2$ if n is even and $D_{\text{-}}(C_n) = 1$ if n is odd for all $n \geq 3 \in \mathbb{N}$. □

Chapter 3

Integer Programming Models for Zero-forcing Number and Diameter

3.1 Standard and Skew Zero-forcing Number Integer Programs

We first discuss computational approaches to calculating the zero-forcing number of a simple graph with respect to the standard zero-forcing rule. Since we have an understanding of the concepts regarding zero-forcing, we can now review the integer programming model introduced by Brimkov et al. in [1] to compute the zero-forcing number of a graph $G \in \mathbb{G}$. This integer program, which we call ZF (G), is displayed in (3.1a) - (3.1e). Note that every edge in $E(G)$ is replaced by two directed edges with opposite direction. A binary variable s_v indicates whether vertex v is in the zero-forcing set. An integer variable x_v, ranging in $\{0, \ldots, T\}$ indicates at which time step vertex v is forced, where T is the maximum difference between the forcing times of two vertices. Lastly, a binary variable y_e, for each directed edge $e = (u, v)$, indicates whether u forces v. The notation $\delta^-(v)$ refers to the set of edges pointing towards the vertex v, $N(u)$ denotes the neighborhood of u, and $N[u]$ refers to the closed neighborhood of u, i.e., $N[u] = N(u) \cup \{u\}$.

$$\text{minimize} \quad \sum_{v \in V(G)} s_v \tag{3.1a}$$

$$\text{subject to} \quad s_v + \sum_{e \in \delta^-(v)} y_e = 1, \quad \forall\, v \in V(G), \tag{3.1b}$$

$$x_u - x_v + (T+1)y_e \leq T, \quad \forall\, e = (u, v) \in E(G), \tag{3.1c}$$

$$x_w - x_v + (T+1)y_e \leq T, \quad \forall\, e = (u, v) \in E(G),\ \forall\, w \in N(u) \setminus \{v\}, \tag{3.1d}$$

$$\mathbf{s} \in \{0,1\}^n, \mathbf{x} \in \{0, \ldots, T\}^n, \mathbf{y} \in \{0,1\}^m \tag{3.1e}$$

The solutions to ZF (G) take the form of a vector $\mathbf{w} = \mathbf{s} \oplus \mathbf{x} \oplus \mathbf{y}$. We denote feasible and optimal solutions of ZF (G) as $\overline{\mathbf{w}} = \overline{\mathbf{s}} \oplus \overline{\mathbf{x}} \oplus \overline{\mathbf{y}}$ and $\mathbf{w}^* = \mathbf{s}^* \oplus \mathbf{x}^* \oplus \mathbf{y}^*$ respectively. Note that constraint (3.1b) indicates that vertex v is either in the zero-forcing set or eventually gets forced at a later time step. Constraint (3.1c) ensures that if vertex u forces vertex v, then the time step at which u is forced must be smaller than the time step at which v is forced. Lastly, constraint (3.1d) makes sure that if vertex u forces vertex v, then the time step at which vertex w is forced is less than the time step at which v is forced for all neighbors w of u. We now show that the optimal value of ZF (G) is equal to the zero-forcing number of G in Theorem 3.1.1, which was proven in [1].

Theorem 3.1.1. *The optimal value of* ZF (G) *is equal to the standard zero-forcing number of* G *for all* $G \in \mathbb{G}$.

Proof. Suppose $C \subseteq V(G)$ is a zero-forcing set of G and let $\mathcal{F}$ be a fixed set of corresponding zero-forcing chains. Since C is a zero-forcing set, then every vertex $v \in V(G)$ is either in C, i.e., $s_v = 1$, or is forced by some other vertex $u \in V(G)$, i.e., $y_e = 1$ where $e = (u, v)$. Hence, constraint (3.1b) must hold. Now let x_v be the time step for which $v \in V(G)$ is forced. Since a vertex cannot force any other vertices until all but one of its neighbors are forced, it follows that for every edge $e = (u, v)$ for which $y_e = 1$, u must be forced before v, meaning that $x_u < x_v$. Similarly, $x_w < x_v$ for all neighbors w of u. Thus, constraints (3.1c) - (3.1d) are satisfied. If $y_e = 0$, then constraints (3.1c) - (3.1d) hold since T is the maximum difference between the forcing times of two vertices. Therefore, constraints (3.1b) - (3.1e) are valid for any zero-forcing set C and associated set of forcing chains $\mathcal{F}$.

Conversely, let $\overline{\mathbf{w}} = \overline{\mathbf{s}} \oplus \overline{\mathbf{x}} \oplus \overline{\mathbf{y}} \in$ Fes$(\text{ZF}\,(G))$. Suppose $C \subseteq V(G)$ is the set of vertices such that $s_v = 1$. By constraint (3.1b), each vertex $v \in V(G)$ is either in C or is the head of exactly one edge $e = (u, v)$, meaning the edge connecting u and v is pointing toward v, such that $y_e = 1$. Then by constraints (3.1c) and (3.1d), there must be an integer $x_v \in \{0, \ldots, T\}$ such that $x_w + 1 \leq x_v$ for all $w \in N[u] \setminus \{v\}$. Therefore, the edges $e = (u, v)$ for which $y_e = 1$ define a chronological list of forcings. Since every vertex $v \in V(G)$ is either in C or is the head of exactly one edge $e = (u, v)$ such that $y_e = 1$, then it follows that C is a zero-forcing set. Additionally, if $\overline{\mathbf{s}}$ is minimal with respect to the objective function (3.1a), i.e., $\overline{\mathbf{w}} \in$ Opt$(\text{ZF}\,(G))$, then C is a minimum zero-forcing set and it follows that the objective value of $\text{ZF}\,(G)$ is the zero-forcing number of G. □

We conclude this section by altering the $\text{ZF}\,(G)$ integer program to define a similar model for calculating the skew zero-forcing number of a simple graph. Given $G \in \mathbb{G}$, we denote this integer program by $\text{ZF}_{-}(G)$ and define it to be $\text{ZF}\,(G)$ without constraint (3.1c). Recall that constraint (3.1c) ensures that if a vertex u forces a vertex v, then the time step at which u is forced must be smaller than the time step at which v is forced. For the skew zero-forcing rule, this condition is no longer required. Since white vertices can now perform forcings, any given vertex can perform a forcing prior to itself being forced. Hence, the time step at which u is forced does not have to be less than the stime step at which v is forced. It follows, by similar logic seen in the proof of Theorem 3.1.1, that the optimal value of $\text{ZF}_{-}(G)$ is equal to the skew zero-forcing number of G for all $G \in \mathbb{G}$.

3.2 Standard and Skew Zero-forcing Diameter Integer Programs

In this section, we develop integer programs to calculate the zero-forcing diameter of a graph $G \in \mathbb{G}$ with respect to the standard and skew zero-forcing rules. In 2020, Cameron et al. introduced the concept of the optimal diameter of a binary program in [9]. Although this concept is restricted to binary programs in [9], we can apply the same idea to the $\text{ZF}\,(G)$ integer program since we are only interested in the binary variable s. Recall that s indicates which vertices are in a zero-forcing set of the corresponding graph. We now build upon the concepts seen in [9] by defining the optimal diameter of ZF (G), which will help us develop an integer program for calculating the standard zero-forcing diameter of a simple graph. To reduce redundancy, we will work only with the standard zero-forcing rule; however, the same model and results that follow also hold for the $\mathrm{ZF}_{-}(G)$ integer program.

First, we define the Euclidean norm of a vector. From this point on, let v_i denote the i^{th} element of a vector $\mathbf{v}$. Given that v has p elements, the *Euclidean norm* of $\mathbf{v}$ is defined as $\|\mathbf{v}\| = \sqrt{v_1^2 + v_2^2 + \ldots + v_p^2}$. Then the *optimal diameter* of ZF (G) is defined as follows.

Definition 3.2.1. Given a graph $G \in \mathbb{G}$, the optimal diameter of ZF (G) is defined as

$$d(G) = \max_{\mathbf{w}^*, \boldsymbol{\omega}^* \in \mathrm{Opt}(\mathrm{ZF}(G))} \|\mathbf{s}^* - \boldsymbol{\varsigma}^*\|^2, \tag{3.2}$$

where $\mathbf{w}^* = \mathbf{s}^* \oplus \mathbf{x}^* \oplus \mathbf{y}^*$ and $\boldsymbol{\omega}^* = \boldsymbol{\varsigma}^* \oplus \boldsymbol{\xi}^* \oplus \boldsymbol{\Upsilon}^*$

It is important to note that we are maximizing the difference between $\mathbf{s}^*$ and $\boldsymbol{\varsigma}^*$ because they refer to the n-dimensional binary vectors that indicate which vertices are in two optimal zero-forcing sets, which we denote by C and C' in Definition 2.1.1, respectively. Thus, allowing us to quantify the minimum intersection of two minimum zero-forcing sets. Additionally, since $\mathbf{s}^*, \boldsymbol{\varsigma}^* \in \{0,1\}^n$, it follows that

$$d(G) = \max_{\mathbf{w}^*, \boldsymbol{\omega}^* \in \mathrm{Opt}(\mathrm{ZF}(G))} \sum_{i=1}^{n} |s_i^* - \varsigma_i^*|.$$

Furthermore, note that $d(G) = 2D(G)$, where $D(G)$ is the zero-forcing diameter in Definition 2.1.1. We formally prove this result in Corollary 3.2.3; however, we continue to explain the intuitive reasoning for why $d(G) = 2D(G)$. We can see that $s_i^* - \varsigma_i^* = 1$ if and only if $s_i^* \neq \varsigma_i^*$, i.e., vertex i is in one of the zero-forcing sets but not the other. Hence, $d(G)$ is equal to the total number of vertices that are in one zero-forcing set but not the other whereas $D(G)$ is equal to the total number of vertices that are not in both. It then follows that

$$d(G) = 2D(G) = 2\left(Z(G) - \min_{C, C' \in S(G)} |C \cap C'|\right).$$

To simplify notation, we now present the canonical form of the integer program ZF (G), which was introduced in Section 3.1, for calculating the standard zero-forcing number of a graph $G \in \mathbb{G}$. Let A be a matrix containing the coefficients on the left hand side of constraints (3.1b) - (3.1d) and let $\mathbf{b}$ represent the right hand side of constraints (3.1b) - (3.1d) in ZF (G). Then ZF (G) can be represented as follows.

$$\begin{aligned} \text{minimize} \quad & \mathbf{1}^{\intercal}\mathbf{s} && (3.3a) \\ \text{subject to} \quad & A\mathbf{w} \leq \mathbf{b}, && (3.3b) \\ & \mathbf{s} \in \{0,1\}^n, \mathbf{x} \in \{0,\dots,T\}^n, \mathbf{y} \in \{0,1\}^m && (3.3c) \end{aligned}$$

where $\mathbf{1}$ is the all ones vector of appropriate length and $\mathbf{w} = \mathbf{s} \oplus \mathbf{x} \oplus \mathbf{y}$. Recall that solutions to $\text{ZF}\,(G)$ take the form of the vector $\mathbf{w}$.

Given a graph $G \in \mathbb{G}$, we now define the following integer program, denoted by $\text{ZFD}\,(G)$, for calculating $d(G)$ with respect to the standard zero-forcing rule.

$$\begin{aligned} \text{minimize} \quad & \mathbf{1}^{\intercal}(\mathbf{s}+\boldsymbol{\varsigma}) + \epsilon\mathbf{1}^{\intercal}\mathbf{z} && (3.4a) \\ \text{subject to} \quad & A\mathbf{w} \leq \mathbf{b}, && (3.4b) \\ & A\boldsymbol{\omega} \leq \mathbf{b}, && (3.4c) \\ & \mathbf{s}+\boldsymbol{\varsigma}-\mathbf{z} \leq \mathbf{1}, && (3.4d) \\ & \mathbf{s},\boldsymbol{\varsigma},\mathbf{z} \in \{0,1\}^n;\ \mathbf{x},\boldsymbol{\xi} \in \{0,\dots,T\}^n;\ \mathbf{y},\boldsymbol{\Upsilon} \in \{0,1\}^m && (3.4e) \end{aligned}$$

Again, recall that $\mathbf{w} = \mathbf{s} \oplus \mathbf{x} \oplus \mathbf{y}$. Additionally, note that $\boldsymbol{\omega} = \boldsymbol{\varsigma} \oplus \boldsymbol{\xi} \oplus \boldsymbol{\Upsilon}$ and $\epsilon > 0$. Then the solutions to $\text{ZFD}\,(G)$ take the form of $\mathbf{q} = \mathbf{w} \oplus \boldsymbol{\omega} \oplus \mathbf{z}$. The vector $\mathbf{z} \in \{0,1\}^n$ indicates which vertices of G are in two zero-forcing sets of G, represented by $\mathbf{s}$ and $\boldsymbol{\varsigma}$. From equations (3.4b) and (3.4c), we can see that $\mathbf{w}$ and $\boldsymbol{\omega}$ are feasible solutions to $\text{ZF}\,(G)$ shown in equations (3.3a) - (3.3c). This is important since we are trying to calculate the minimum intersection of two minimum zero-forcing sets of G. We denote the feasible solutions to $\text{ZFD}\,(G)$ as $\overline{\mathbf{q}} = \overline{\mathbf{w}} \oplus \overline{\boldsymbol{\omega}} \oplus \overline{\mathbf{z}}$ where $\overline{\mathbf{w}} = \overline{\mathbf{s}} \oplus \overline{\mathbf{x}} \oplus \overline{\mathbf{y}}$ and $\overline{\boldsymbol{\omega}} = \overline{\boldsymbol{\varsigma}} \oplus \overline{\boldsymbol{\xi}} \oplus \overline{\boldsymbol{\Upsilon}}$. Similarly, the optimal solutions to $\text{ZFD}\,(G)$ take the form of $\mathbf{q}^* = \mathbf{w}^* \oplus \boldsymbol{\omega}^* \oplus \mathbf{z}^*$ where $\mathbf{w}^* = \mathbf{s}^* \oplus \mathbf{x}^* \oplus \mathbf{y}^*$ and $\boldsymbol{\omega}^* = \boldsymbol{\varsigma}^* \oplus \boldsymbol{\xi}^* \oplus \boldsymbol{\Upsilon}^*$. Now let $[n] = \{1,2,...,n\}$. Then we have the following proposition concerning the optimal solutions of $\text{ZFD}\,(G)$.

Proposition 3.2.1. *Suppose $G \in \mathbb{G}$ and let $\mathbf{q}^* \in$ Opt($\text{ZFD}\,(G)$). Then $z_i^* = 1$ if and only if $s_i^* = \varsigma_i^* = 1$ for all $i \in [n]$.*

Proof. Suppose $z_i^* = 1$ for some $i \in [n]$. By way of contradiction, also suppose either $s_i^* = 0$ or $\varsigma_i^* = 0$. Then constraint (3.4d) will be true if $z_i^* = 0$. This contradicts the fact that we have an optimal solution to $\text{ZFD}\,(G)$, i.e., the objective function (3.4a) would not be minimized. Thus, our assumption that $s_i^* = 0$ or $\varsigma_i^* = 0$ must be false. Hence, $s_i^* = \varsigma_i^* = 1$.

Conversely, let $s_i^* = \varsigma_i^* = 1$ for some $i \in [n]$. Then constraint (3.4d) holds only when $z_i^* = 1$. Therefore, given $\mathbf{q}^* \in$ Opt($\text{ZFD}\,(G)$), $z_i^* = 1$ if and only if $s_i^* = \varsigma_i^* = 1$ for all $i \in [n]$. ☐

Note that $\epsilon > 0$ is a necessary condition for the proof of Proposition 3.2.1 to hold. This condition ensures that we may obtain a feasible solution with a smaller objective value in (3.4a) by setting $z_i^* = 0$. We now prove that there exists a value of ϵ where the optimal diameter of $\mathrm{ZF}\,(G)$ can be derived from any element of Opt($\mathrm{ZFD}\,(G)$). Before doing so, we introduce the following notation. Given vectors $\mathbf{a}, \mathbf{b} \in \{0,1\}^n$, we define $\psi(\mathbf{a}, \mathbf{b})$ to be $\psi(\mathbf{a}, \mathbf{b}) = \{i \in [n] : a_i = b_i = 0\}$.

Theorem 3.2.2. *Given a graph $G \in \mathbb{G}$, there exists an $\epsilon > 0$ such that for all $\mathbf{q}^* \in$ Opt($\mathrm{ZFD}\,(G)$), we have $d(G) \leq n - \mathbf{1}^\intercal \mathbf{z}^*$.*

Proof. We break this proof into two cases. First, assuming Fes($\mathrm{ZF}\,(G)$) = Opt($\mathrm{ZF}\,(G)$). Second, assuming Fes($\mathrm{ZF}\,(G)$) $\neq$ Opt($\mathrm{ZF}\,(G)$). In the first case, it follows that there exists $k \in \mathbb{Z}$ such that for all $\overline{\mathbf{q}} \in$ Fes($\mathrm{ZFD}\,(G)$), we have $\mathbf{1}^\intercal(\overline{\mathbf{s}} + \overline{\boldsymbol{\varsigma}}) = k$. Hence, for any $\epsilon > 0$, the optimal value of $\mathrm{ZFD}\,(G)$ occurs when $\mathbf{1}^\intercal \mathbf{z}$ is minimized. Furthermore, by Proposition 3.2.1, for any $\mathbf{q}^* \in$ Opt($\mathrm{ZFD}\,(G)$), we have

$$\|\mathbf{s}^* - \boldsymbol{\varsigma}^*\|^2 = n - \mathbf{1}^\intercal \mathbf{z}^* - |\psi(\mathbf{s}^*, \boldsymbol{\varsigma}^*)| \leq n - \mathbf{1}^\intercal \mathbf{z}^*.$$

Since $\mathbf{1}^\intercal \mathbf{z}^*$ is minimized, it follows that the upper bound $n - \mathbf{1}^\intercal \mathbf{z}^*$ is maximized. Therefore,

$$d(G) \leq n - \mathbf{1}^\intercal \mathbf{z}^*.$$

In the second case, it follows that there exists minimal $\overline{\mathbf{w}}_*, \overline{\boldsymbol{\omega}}_* \in$ Fes($\mathrm{ZF}\,(G)$) $\setminus$ Opt($\mathrm{ZF}\,(G)$) such that

$$\mathbf{1}^\intercal(\overline{\mathbf{s}} + \overline{\boldsymbol{\varsigma}}) \geq \mathbf{1}^\intercal(\overline{\mathbf{s}}_* + \overline{\boldsymbol{\varsigma}}_*) > \mathbf{1}^\intercal(\mathbf{s}^* + \boldsymbol{\varsigma}^*)$$

for all $\overline{\mathbf{w}}, \overline{\boldsymbol{\omega}} \in$ Fes($\mathrm{ZF}\,(G)$) $\setminus$ Opt($\mathrm{ZF}\,(G)$) and $\mathbf{w}^*, \boldsymbol{\omega}^* \in$ Opt($\mathrm{ZF}\,(G)$). Now we can fix $\mathbf{w}^*, \boldsymbol{\omega}^* \in$ Opt($\mathrm{ZF}\,(G)$) and define

$$\epsilon = \frac{\mathbf{1}^\intercal(\overline{\mathbf{s}}_* + \overline{\boldsymbol{\varsigma}}_*) - \mathbf{1}^\intercal(\mathbf{s}^* + \boldsymbol{\varsigma}^*)}{2n}. \tag{3.5}$$

Thus, for all $\overline{\mathbf{w}}, \overline{\boldsymbol{\omega}} \in$ Fes($\mathrm{ZF}\,(G)$) $\setminus$ Opt($\mathrm{ZF}\,(G)$), it follows that

$$\mathbf{1}^\intercal(\overline{\mathbf{s}} + \overline{\boldsymbol{\varsigma}}) \geq \mathbf{1}^\intercal(\mathbf{s}^* + \boldsymbol{\varsigma}^*) + 2n\epsilon > \mathbf{1}^\intercal(\mathbf{s}^* + \boldsymbol{\varsigma}^*) + n\epsilon \geq \mathbf{1}^\intercal(\mathbf{s}^* + \boldsymbol{\varsigma}^*) + \epsilon \mathbf{1}^\intercal \mathbf{z}$$

for any $\mathbf{z} \in \{0,1\}^n$. Hence, given $\mathbf{q}^* \in$ Opt($\mathrm{ZFD}\,(G)$), we have $\mathbf{w}^* = \mathbf{s}^* \oplus \mathbf{x}^* \oplus \mathbf{y}^* \in$ Opt($\mathrm{ZF}\,(G)$) and $\boldsymbol{\omega}^* = \boldsymbol{\varsigma}^* \oplus \boldsymbol{\xi}^* \oplus \boldsymbol{\Upsilon}^* \in$ Opt($\mathrm{ZF}\,(G)$). Furthermore, minimizing the objective function in (3.4a) requires $\mathbf{1}^\intercal \mathbf{z}^*$ to be minimized. Since $\mathbf{1}^\intercal \mathbf{z}^*$ is minimized, by Proposition 3.2.1, we have that

$$\|\mathbf{s}^* - \boldsymbol{\varsigma}^*\|^2 = n - \mathbf{1}^\intercal \mathbf{z}^* - |\psi(\mathbf{s}^*, \boldsymbol{\varsigma}^*)| \leq n - \mathbf{1}^\intercal \mathbf{z}^*$$

holds, where $d(G) = \|\mathbf{s}^* - \boldsymbol{\varsigma}^*\|^2$. Therefore, there exists an $\epsilon > 0$ such that $d(G) \leq n - \mathbf{1}^\intercal \mathbf{z}^*$ for all $\mathbf{q}^* \in \text{Opt}(\text{ZFD}\,(G))$. $\square$

We conclude this chapter by proving the following corollary regarding the value of $d(G)$ for a graph $G \in \mathbb{G}$. The result in Corollary 3.2.3 allows us to compute the zero-forcing diameter, as defined in Definition 2.1.1, of G using an optimal solution of $\text{ZFD}\,(G)$.

Corollary 3.2.3. *Suppose $G \in \mathbb{G}$ and let $\|\mathbf{s}^*\|^2 = Z(G)$ for all $\mathbf{w}^* \in \text{Opt}(\text{ZF}\,(G))$. Then there exists an $\epsilon > 0$ such that $d(G) = 2D(G)$ for all $\mathbf{q}^* \in \text{Opt}(\text{ZFD}\,(G))$.*

Proof. Let $\mathbf{q}^* \in \text{Opt}(\text{ZFD}\,(G))$. Then by Theorem 3.2.2, the vectors $\mathbf{w}^* = \mathbf{s}^* \oplus \mathbf{x}^* \oplus \mathbf{y}^*$ and $\boldsymbol{\omega}^* = \boldsymbol{\varsigma}^* \oplus \boldsymbol{\xi}^* \oplus \boldsymbol{\Upsilon}^*$ are both in $\text{Opt}(\text{ZF}\,(G))$. Additionally, by Theorem 3.2.2, we have that $\mathbf{1}^\intercal \mathbf{z}^*$ is minimized. Since $\|\mathbf{s}^*\|^2 = Z(G)$ for all $\mathbf{w}^* \in \text{Opt}(\text{ZF}\,(G))$, then we have

$$\|\mathbf{s}^* - \boldsymbol{\varsigma}^*\|^2 = 2(Z(G) - \mathbf{1}^\intercal \mathbf{z}^*).$$

It follows that $d(G) = \|\mathbf{s}^* - \boldsymbol{\varsigma}^*\|^2$ since $\mathbf{1}^\intercal \mathbf{z}^*$ is minimized. Recall that $\mathbf{z}^* \in \{0,1\}^n$ indicates which vertices of G are in two minimum zero-forcing sets of G, represented by $\mathbf{s}^*$ and $\boldsymbol{\varsigma}^*$. By this definition, and since $\mathbf{1}^\intercal \mathbf{z}^*$ is minimized, it follows that $\mathbf{1}^\intercal \mathbf{z}^*$ is equivalent to $\min_{C,C' \in S(G)} |C \cap C^{'}|$ where C and $C^{'}$ are represented by $\mathbf{s}^*$ and $\boldsymbol{\varsigma}^*$ respectively. Therefore, we have that

$$d(G) = 2\left(Z(G) - \min_{C,C' \in S(G)} |C \cap C^{'}|\right) = 2D(G)$$

for all $\mathbf{q}^* \in \text{Opt}(\text{ZFD}\,(G))$. $\square$

Chapter 4

Conclusion

4.1 Conclusions

In this thesis, we began by studying preliminary information regarding graph theory, zero-forcing, and integer programming. We defined a simple graph, explained important definitions related to simple graphs, introduced different graph operations, and presented common graph families. We continued by describing zero-forcing, an iterative coloring game on a graph, and demonstrating how the standard and skew zero-forcing rules operate. Additionally, we listed the standard and skew zero-forcing numbers of common graph families. We concluded the introduction to preliminary information by providing foundational knowledge of linear and integer programming.

Then we proceeded to formulate our contributions by creating a new graph parameter called the zero-forcing diameter, which quantifies the minimum intersection of two minimum zero-forcing sets of a graph with respect to its zero-forcing number. We investigated the zero-forcing diameter with respect to both the standard and skew zero-forcing rules—finding the numerical bounds of the zero-forcing diameter for each rule. Furthermore, we calculated the value of the zero-forcing diameter of common graph families and supported these calculations with theoretical proofs. The zero-forcing diameter has allowed us to make connections between graphs with vastly different structures. We saw that the path graph, complete graph, and star graph all have the same standard zero-forcing diameter. This shows us that graph families with vastly different structures may share a common characteristic—their zero-forcing diameters.

Finally, in Chapter 3, we discussed computational approaches to zero-forcing. We started by introducing an integer programming model, denoted by $\mathrm{ZF}\,(G)$ and developed by Brimkov et al. in [1], for calculating the zero-forcing number of a graph $G \in \mathbb{G}$ with respect to the standard zero-forcing rule. We then altered $\mathrm{ZF}\,(G)$ to develop an integer program to calculate the zero-forcing number of G with respect to the skew zero-forcing rule. We continued by defining the optimal diameter of $\mathrm{ZF}\,(G)$, denoted by $d(G)$, and developed a new integer program for calculating $d(G)$. We concluded Chapter 3 by proving that our integer program does, in fact, calculate $d(G)$ correctly.

4.2 Future Research

We conclude this thesis by exploring possible avenues for future research. The first logical idea is to investigate additional zero-forcing rules. We have thoroughly explored the standard and skew zero-forcing rules; however, it would be beneficial to expand our work to cover other color change rules. One such rule, described in [10], is called the positive semidefinite zero-forcing rule. Another interesting rule is the power domination zero-forcing rule, introduced by Benson et al. in [11] for its applications in electrical networks. We could proceed by developing integer programming models to calculate both the zero-forcing number and diameter of a simple graph with respect to these color change rules. Then use those integer programs to prove additional theoretical results for specific graph families.

In closing, one last topic of future investigation could be machine learning approaches to zero-forcing. Machine learning in graph theory has been studied by many professionals in both academia and industry. For example, in [12], Wang et al. discuss automated graph machine learning approaches for solving the task of finding the best hyper-parameters and neural architecture configurations for data and tasks regarding graphs. In [12], the authors explain how difficult it is to manually develop optimal machine learning algorithms for graph-related tasks. This makes for an interesting and challenging research topic where we can work with preexisting open-source libraries for automated graph machine learning approaches. We could study these libraries and develop our own machine learning algorithms for calculating the zero-forcing number and diameter of a graph.

Bibliography

[1] Boris Brimkov, Caleb C. Fast, and Illya V. Hicks. Computational approaches for zero forcing and related problems. *European Journal of Operational Research*, 2019.

[2] Reinhard Diestel. *Graph Theory*, volume 173. Spring-Verlag Heidelberg, New York, 3 edition, 2005.

[3] Edward R. Scheinerman. *Mathematics : A Discrete Introduction*. Brooks Cole Publishing Co., 3 edition, 2013.

[4] D. Burgarth, D. D'Alessandro, L. Hogben, S. Severini, and M. Young. Zero forcing, linear and quantum controllability for systems evolving on networks. *IEEE Trans. Auto. Control*, 58, 2013.

[5] Shaun M. Fallat and Leslie Hogben. Minimum rank, maximum nullity, and zero forcing number of graphs. 2014.

[6] AIM Minimum Rank – Special Graphs Work Group. Zero forcing sets and the minimum rank of graphs. *Linear Algebra and its Applications*, 428(7):1628–1648, 2008.

[7] Nicole F. Kingsley. *Skew propagation time*. PhD thesis, Iowa State University, 2015.

[8] Thomas S. Ferguson. Linear programming: A concise introduction.

[9] Thomas R. Cameron, Sebastian Charmot, and Jonad Pulaj. Diameter polytopes of feasible binary programs. 2020.

[10] Jason Ekstrand, Craig Erickson, H. Tracy Hall, Diana Hay, Leslie Hogben, Ryan Johnson, Nicole Kingsley, Steven Osborne, Travis Peters, Jolie Roat, Arianne Ross, Darren D. Row, Nathan Warnberg, and Michael Young. Positive semidefinite zero forcing. *Linear Algebra and its Applications*, 439(7):1862–1874, 2013.

[11] Katherine F. Benson, Daniela Ferrero, Mary Flagg, Veronika Furst, Leslie Hogben, Violeta Vasilevska, and Brian Wissman. Zero forcing and power domination for graph products. 2015.

[12] Xin Wang, Ziwei Zhang, and Wenwu Zhu. Automated graph machine learning: Approaches, libraries and directions. 2022.

Academic Vita

Jakob Loedding

Email: jul914@psu.edu, jakeloedding11@gmail.com

EDUCATION

The Pennsylvania State University, Erie, PA — *Graduation: May 2022*

SCHREYER HONORS COLLEGE

- Bachelor of Science: Applied Mathematics
- Minors: Computer Science and Statistics

CMPSC 445: Applied Machine Learning in Data Science — *Aug 2021 – Present*

- Learn concepts and implementation of supervised/unsupervised algorithms like K-NN, K-means, Naïve Bayes, SVM, ANN, CNN, RNN, and NLP

IBM Coursera Course Certificate: Python Project for Data Engineering — *Aug 2021*

- Mastered skills of extracting, loading, and transforming data during intensive two-week course
- Utilized Python, Jupyter Notebooks, and Watson Studio to interact with APIs, implement web scrapping, perform calculations, and load data to various formats

WORK EXPERIENCE

Systems Engineering Intern at Northrop Grumman — *June 2021 – Aug 2021*

- Developed an automated dashboard, in Tableau, presenting the perceived and predictive utility of engineering metrics used to make program intervention decisions
- Expanded a full-stack data engineering pipeline that links data between Alteryx, SQL servers, and Tableau
- Constructed Alteryx workflows that extract data via Python code that interacts with company APIs
- Studied regression code with lag analysis performed by team members utilizing R

Student Researcher at The Pennsylvania State University, Erie, PA — *Jan 2020 – Present*

- Study graph theory, zero-forcing, rankability of data, and optimization techniques in Python
- Develop integer programs that compute optimal zero-forcing sets of a graph using CPLEX optimization
- Invent zero-forcing diameter, a graph parameter, and generalize its value for basic graphs

Teacher's Assistant/Grader at The Pennsylvania State University, Erie, PA — *Sep 2019 – Dec 2019*

- Graded homework and crafted original answer keys for a class of 30+ underclassmen under strict deadlines
- Monitored attendance sheets and assisted with proctoring in-person exams to enforce academic integrity
- Mentored and tutored struggling students to assist in academic success

ACADEMIC PROJECTS

Machine Learning Retro Video Game Music Generator – Python Application — *Aug 2021 – Nov 2021*

- Trained a dual autoencoder that utilizes K-means clustering to generate retro video game music by categories
- Web scrapped 5,000 songs as samples and implemented object-oriented data preparation scripts
- Developed a GUI in Python that allows the user to select a category for which to generate music

Schreyer Honors Option Research – RSA Encryption — *Sep 2020 – Dec 2020*

- Formulated a 14-page research paper investigating the RSA algorithm and its applications

Schreyer Honors Option Research – Riccati Differential Equation — *Jan 2020 – May 2020*

- Wrote a 7-page paper analyzing the Riccati equation and methods of calculating its solution

SKILLS

Python, R, SQL, Java, C++, Alteryx Designer, Tableau, Jupyter Notebook, Regression Analysis, Git, Data Management, CPLEX Optimization, Web Scrapping, Agile Workflow, Leadership, Collaboration

ACADEMIC AWARDS/HONORS

Most Promising Freshman in Mathematics, President's Freshman Award (maintaining a 4.0 GPA), Dean's List, Behrend Honors Certificate (completed in 2 years as opposed to the standard 4)

COMMUNITY INVOLVEMENT

Peer Tutor at The Pennsylvania State University, Erie, PA — *Sep 2021 – Present*

Member of Pi Mu Epsilon – Honorary National Mathematics Society — *Apr 2021 – Present*

Member of National Society of Leadership and Success — *Mar 2019 – Present*

Schreyer Honors Mentor — *Sep 2020 – Dec 2020*