INTEGER PROGRAMMING APPROACHES TO TWO DOMINATION RELATED PROBLEMS IN GRAPH THEORY

A THESIS SUBMITTED TO
THE GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES
OF
MIDDLE EAST TECHNICAL UNIVERSITY

BY

ÇINAR ARI

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR
THE DEGREE OF MASTER OF SCIENCE
IN
INDUSTRIAL ENGINEERING

JULY 2025

Approval of the thesis:

INTEGER PROGRAMMING APPROACHES TO TWO DOMINATION RELATED PROBLEMS IN GRAPH THEORY

submitted by **ÇINAR ARI** in partial fulfillment of the requirements for the degree of **Master of Science in Industrial Engineering Department, Middle East Technical University** by,

Prof. Dr. Naci Emre Altun
Dean, Graduate School of **Natural and Applied Sciences** _____

Prof. Dr. Zeynep Pelin Bayındır
Head of Department, **Industrial Engineering** _____

Assoc. Prof. Dr. Mustafa Kemal Tural
Supervisor, **Industrial Engineering, METU** _____

Examining Committee Members:

Prof. Dr. Meral Azizoğlu
Industrial Engineering, METU _____

Assoc. Prof. Dr. Mustafa Kemal Tural
Industrial Engineering, METU _____

Prof. Dr. Oya Ekin Karaşan
Industrial Engineering, Bilkent University _____

Prof. Dr. Sinan Gürel
Industrial Engineering, METU _____

Assist. Prof. Dr. Nader Ghaffarinasab
Industrial Engineering, METU _____

Date:04.07.2025

I hereby declare that all information in this document has been obtained and presented in accordance with academic rules and ethical conduct. I also declare that, as required by these rules and conduct, I have fully cited and referenced all material and results that are not original to this work.

Name, Surname: Çınar Arı

Signature :

ABSTRACT

INTEGER PROGRAMMING APPROACHES TO TWO DOMINATION RELATED PROBLEMS IN GRAPH THEORY

Arı, Çınar
M.S., Department of Industrial Engineering
Supervisor: Assoc. Prof. Dr. Mustafa Kemal Tural

July 2025, 67 pages

In this thesis, two domination variants; namely, paired-domination and defensive domination, are studied using integer programming techniques.

The first variant involves finding a dominating set (called a paired-dominating set) in a given simple graph which induces a subgraph that has a perfect matching. Assuming weights on the nodes and the edges, we study the problem of finding a paired-dominating set of minimum weight. For this problem, after proposing an initial integer programming formulation, we strengthen it and show that the linear programming relaxation of the latter characterizes the convex hull of the incidence vectors of paired-dominating sets in trees even though the constraint matrix is not always totally unimodular. Moreover, we initiate the study of the paired-dominating set polytope by proposing several sets of valid inequalities. The performance of the strengthened formulation and the impact of valid inequalities are analyzed via a computational study.

The second variant looks for a dominating set (called a k-defensive dominating set) in a given simple graph which can defend against all attacks to any k nodes of the graph

for a given positive integer k. A node in our dominating set can defend against at most one attacker in its closed neighborhood. Assuming weights on the nodes, we study the problem of finding a k-defensive dominating set of minimum weight. For this problem, we propose an integer programming formulation with exponentially many constraints, strengthen it via some valid inequalities and implement the strengthened formulation using a row generation procedure.

Keywords: Integer programming, Domination, Paired-domination, Defensive domination, Valid inequalities.

ÖZ

ÇİZGE TEORİSİNDEKİ İKİ BASKIN KÜME VARYANTINA TAMSAYILI PROGRAMLAMA YAKLAŞIMLARI

Arı, Çınar
Yüksek Lisans, Endüstri Mühendisliği Bölümü
Tez Yöneticisi: Doç. Dr. Mustafa Kemal Tural

Temmuz 2025 , 67 sayfa

Bu tezde, iki baskın küme varyantı olan eşli baskınlık ve defansif baskınlık, tamsayılı programlama teknikleri kullanılarak çalışılmıştır.

İlk türde, verilen bir basit çizgede, eşli baskın küme olarak adlandırılan mükemmel eşleme içeren bir altçizgeyi indükleyen bir baskın küme bulunması amaçlanmaktadır. Düğümler ve kenarlar üzerinde ağırlıklar tanımlandığı varsayılarak, minimum ağırlıklı bir eşli baskın küme bulma problemi ele alınmıştır. Bu problem için başlangıçta bir tamsayılı programlama formülasyonu önerilmiş, ardından bu formülasyon güçlendirilmiş ve elde edilen doğrusal programlama gevşetmesinin, kısıtlar matrisi her zaman tamamen unimodüler olmasa da, ağaçlar üzerindeki eşli baskın kümelerin insidans vektörlerinin konveks zarfını tanımladığı gösterilmiştir. Ayrıca, eşli baskın küme politopunun incelenmesine başlanmış ve çeşitli geçerli eşitsizlik kümeleri önerilmiştir. Güçlendirilmiş modelin performansı ve geçerli eşitsizliklerin etkisi, sayısal bir çalışma ile rastgele üretilmiş çizgeler üzerinde analiz edilmiştir.

İkinci varyant, verilen basit bir çizge ve pozitif bir tamsayı k için; k-defansif baskın

küme olarak adlandırılan, çizgenin herhangi k düğümüne yönelik tüm saldırılara karşı savunma yapabilecek bir baskın küme bulmayı amaçlamaktadır. Defansif baskın kümedeki bir düğüm, kapalı komşuluğundaki en fazla bir saldırgana karşı savunma yapabilir. Düğümler üzerinde ağırlıkların tanımlı olduğu varsayımıyla, en düşük ağırlığa sahip bir k-defansif baskın kümenin bulunması problemi incelenmiştir. Bu problem için, üstel sayıda kısıta sahip bir tamsayılı programlama formülasyonu önerilmiş, formülasyon bazı geçerli eşitsizliklerle güçlendirilmiş ve güçlendirilmiş formülasyon, satır üretimi yöntemi kullanılarak uygulanmıştır.

Anahtar Kelimeler: Tamsayılı programlama, Baskın kümeler, Eşli-baskın kümeler, Defansif baskın kümeler, Geçerli eşitsizlikler

ACKNOWLEDGMENTS

I want to thank my advisor, Mustafa Kemal Tural. I feel extremely fortunate to have him as a mentor and I deeply appreciate his support and guidance. Working with someone who share the same excitement is a big pleasure. His enthusiasm and determination has been a constant source of motivation.

I thank my examining committee members, Prof. Dr. Meral Azizoğlu, Prof. Dr. Oya Ekin Karaşan, Prof. Dr. Sinan Gürel, and Assist. Prof. Dr. Nader Ghaffarinasab for their invaluable comments and suggestions.

Finally, I would like to thank my family and friends for their love and support.

This research was funded by the Scientific and Technological Research Council of Türkiye (TÜBİTAK) through grant number 2210.

TABLE OF CONTENTS

ABSTRACT v

ÖZ vii

ACKNOWLEDGMENTS ix

TABLE OF CONTENTS x

LIST OF TABLES xii

LIST OF FIGURES xiii

LIST OF ALGORITHMS xv

LIST OF ABBREVIATIONS xvi

CHAPTERS

1 INTRODUCTION 1

1.1 Dominating Sets in Graphs 1

1.2 History of Domination 2

1.3 Applications of Domination 3

1.4 Contribution and Outline of the Thesis 4

2 PRELIMINARY CONCEPTS IN GRAPH THEORY 7

3 SOLVING THE MINIMUM WEIGHTED PAIRED-DOMINATING SET PROBLEM USING INTEGER PROGRAMMING 15

3.1 Problem Definition 15

3.2 Literature Review 16

3.3 Integer Programming and Paired-Domination in Trees 17

3.4 Valid Inequalities 34

3.5 Computational Studies 38

3.6 Conclusion 46

4 A ROW GENERATION APPROACH TO THE MINIMUM WEIGHTED K-DEFENSIVE SET PROBLEM 47

4.1 Problem Definition 47

4.2 Literature Review 49

4.3 An Integer Programming Formulation for the MW-K-DEF-DOM Problem 50

4.4 A Row Generation Approach 51

4.5 Valid Inequalities 53

4.6 Computational Studies 58

4.7 Conclusion 64

REFERENCES 65

LIST OF TABLES

TABLES

Table 3.1 Unweighted version: Solution times and average gap for three formulations across 100 Erdős–Rényi graphs for each (n, p) combination . . 42

Table 3.2 Weighted version: Solution times and average gap for three formulations across 100 Erdős–Rényi graphs for each (n, p) combination 43

Table 3.3 Unweighted version: Number of integral solutions of the LP relaxation across 100 Erdős–Rényi graphs for each (n, p) combination. 44

Table 3.4 Weighted version: Number of integral solutions of the LP relaxation across 100 Erdős–Rényi graphs for each (n, p) combination. 45

Table 4.1 Smallest k for which not all inequalities can be listed within 1 hour for different number of vertices n 58

Table 4.2 Unweighted Version: Computation statistics for various k values across (n, p) pairs 62

Table 4.3 Weighted Version: Computation statistics for various k values across (n, p) pairs 63

LIST OF FIGURES

FIGURES

Figure 1.1 Example graph $G = (V, E)$ 2

Figure 2.1 Example graph $G_1 = (V_1, E_1)$ 8

Figure 2.2 Graphs based on G_1 illustrating subgraphs 10

Figure 2.3 Example graph $G_2 = (V_2, E_2)$ 11

Figure 2.4 Examples of a star graph and a non-star tree on 5 vertices 12

Figure 2.5 Example graph $G_3 = (V_3, E_3)$ 12

Figure 3.1 Example graph $G = (V, E)$ 16

Figure 3.2 Star graph S_4 19

Figure 3.3 A tree for which the coefficient matrix of the LP relaxation of (IP-PD-2) is not totally unimodular 21

Figure 3.4 Original graph T' 26

Figure 3.5 T_I^* 26

Figure 3.6 T_J^* 26

Figure 3.7 Example fractional solution that is cut off by inequality (3.15) . . 34

Figure 3.8 Example fractional solution that is cut off by inequality (3.16) . . 35

Figure 3.9 Example fractional solution that is cut off by inequality (3.17) . . 36

Figure 3.10 Example fractional solution that is cut off by inequality (3.18) . . 37

Figure 3.11 Example fractional solution that is cut off by inequality (3.19) . . 38

Figure 4.1 Example graph S_5 to illustrate k-defensive sets 48

Figure 4.2 Example fractional solution that is cut off by inequality (4.8) . . 54

Figure 4.3 Example fractional solution that is cut off by inequality (4.9) . . 55

Figure 4.4 Example fractional solution that is cut off by inequality (4.10) . . 57

Figure 4.5 Star Graph S_7 57

LIST OF ALGORITHMS

ALGORITHMS

Algorithm 1 ConcatReduce 28

Algorithm 2 Matching 31

LIST OF ABBREVIATIONS

LP	Linear programming
IP	Integer programming
MW-PDSP	Minimum weighted paired-dominating set problem
MW-DSP	Minimum weighted dominating set problem

CHAPTER 1

INTRODUCTION

Domination in graphs is a widely studied topic, with over 5000 papers published since it is first defined [17]. The popularity of domination can be attributed to its broad applications and its strong connections to other graph-theoretic concepts. This thesis focuses on two variants of domination; namely, paired-domination and defensive domination. Before starting the discussion of these variants, it is beneficial to start with an overview of domination in graphs, including its definition, history, and applications.

1.1 Dominating Sets in Graphs

Let $G = (V, E)$ be a graph, where $V \neq \emptyset$ is its vertex set and E is its edge set. A set $D \subseteq V$ is said to be a dominating set of G if every vertex in $V \setminus D$ is adjacent to a vertex in D. A vertex is said to dominate itself and all the vertices that are adjacent. Hence, D is a dominating set of G if every vertex in V is dominated by at least one vertex in D. Every graph has a dominating set, as V is a dominating set of $G = (V, E)$. The domination number of a graph G is equal to the minimum cardinality of a dominating set in G. The minimum dominating set problem, denoted by MIN-DOM, is the problem of finding a dominating set of minimum cardinality in a given graph. The decision version of MIN-DOM is, given a graph $G = (V, E)$ and a positive integer k, determine whether G has a dominating set of size at most k.

In the graph G in Figure 1.1, the set of vertices $S_1 = \{1, 2\}$ is a dominating set because each vertex in $V \setminus S_1 = \{3, 4, 5\}$ is adjacent to some vertex in S_1. Other dominating sets include $S_2 = \{1, 3\}$, $S_3 = \{1, 2, 3\}$, $S_4 = \{4, 5\}$. On the contrary,

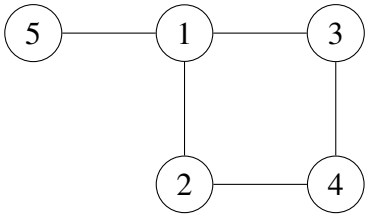


Figure 1.1: Example graph $G = (V, E)$

the set $S_5 = \{1, 5\}$ is not a dominating set because vertex $4 \in V \setminus S_5$ is not adjacent to any vertex in S_5. Similarly, $S_6 = \{3, 5\}$ is not a dominating set because vertex 2 is not dominated; and $S_7 = \{2, 3, 4\}$ is not a dominating set because vertex 5 is not dominated.

1.2 History of Domination

The origin of domination can be traced back to the Roman Empire. Due to the size of the empire, the emperor had to strategically place its troops so that all regions were protected, and could receive help from neighbors without leaving any area exposed. This problem could be modeled as a graph, where the vertices are the regions and the edges represent whether or not two regions are neighbors. The strategy Romans used to defend the empire is studied in the literature under the name *Roman domination* [6].

Other historic problems that are closely related to domination include chessboard problems, which gained popularity in the 1800s. For instance, the Queens Domination Problem is a well-known problem that asks for the minimum number of queens needed on an $n \times n$ chessboard such that every square is either occupied or attacked by a queen. This is equivalent to finding a minimum dominating set in a graph where vertices represent board squares and edges represent queen attacks [17].

The first appearance of domination in the literature was in 1958 by Claude Berge under the name *coefficient of external stability* [1]. The term *domination* was coined by Øystein Ore in 1962 in his book [24]. In the following years, research in domination rapidly gained momentum. Variants of domination have been introduced to the liter-

ature. In these variants, dominating sets that satisfy additional constraints are sought. The motivation for studying the variants has been their respective application areas, as well as connections to other domination-related problems like independent sets and irredundant sets [17]. In 1979, the decision version of MIN-DOM was shown to be NP-complete [15]. The research in domination investigates (possibly on a variant) finding bounds on the domination number, the computational complexity of the problem for different graph classes, as well as developing exact and approximate algorithms [17].

1.3 Applications of Domination

There are many practical applications of domination. Three of them are discussed in this section.

The first application is the design of communication networks. Assume the goal is to select some broadcast centers such that the entire region is able to receive the messages. This problem can be modeled as finding a dominating set in a graph where the edges exist between pairs of vertices if the message broadcasted in one endpoint can be received in the other [23]. Other variants have been proposed, where the decision maker has the power to choose the strength of the broadcast (so that there is a greater range the message is transmitted) at an increasing cost, and the goal is to find the broadcast centers that accomplish the goal with minimum cost [12], which was later defined as broadcast domination in graphs [9].

Another application of domination in graphs is the selection of representatives. Given a group of people from an organization, the goal is to select a minimum number of people as representatives such that every person knows a representative or is one themselves. This can be modeled using a graph where vertices represent individuals and edges connect those who know each other. The problem then reduces to finding a minimum dominating set in the constructed graph [17].

Domination is also used for applications where efficient coverage is required, for instance when setting up a surveillance system for an area. Vertices represent possible surveillance points to place guards (or cameras), and a placed guard can see everything in its neighborhood. The problem is to find the minimum number of guards

required for a full coverage, which is equivalent to finding a minimum dominating set.

Other application areas include facility location, school bus routing, chemistry, transporting hazardous materials, and monitoring electrical power networks. The reader is referred to [17] where other application areas are explained in detail.

Next, a brief definition of paired-domination and defensive domination, and the contributions of this thesis to the literature are given.

1.4 Contribution and Outline of the Thesis

Paired-domination is a variant of domination that is closely related to matchings in graphs. In this thesis, we study the minimum weighted paired-dominating set problem (MW-PDSP) with both vertex and edge weights, to be defined in Chapter 3. This is the first study that considers edge weights in paired-domination. Section 3.1 defines the problem studied, and Section 3.2 presents a literature review. In Section 3.3, an integer programming formulation for the MW-PDSP is proposed for the first time in the literature. Moreover, in the same section, after strengthening our initial formulation, we show that its linear programming relaxation characterizes the convex hull of the incidence vectors of paired-dominating sets in trees. This shows that MW-PDSP can be solved in polynomial time in trees even when there are weights on the edges. In Section 3.4, we initiate the study of the paired-dominating set polytope, by proposing several sets of valid inequalities. The performance of the strengthened formulation and the impact of valid inequalities are analyzed in randomly generated graphs via a computational study in Section 3.5.

Defensive domination is a variant that has applications in network security. This variant looks for a dominating set that can defend against multiple simultaneous attacks. The precise definition of the problem studied is given in Section 4.1. In Section 4.2, a literature review on defensive domination is conducted. In Section 4.3, we propose an integer programming formulation for the problem for the first time. To our knowledge, there were no algorithms available in the literature to solve the problem in general graphs. In Section 4.4, a row generation procedure based on the IP formu-

lation is developed. Three classes of valid inequalities are proposed in Section 4.5, and we test the row generation procedure on random graphs in Section 4.6.

CHAPTER 2

PRELIMINARY CONCEPTS IN GRAPH THEORY

For the sake of completeness, this chapter introduces the necessary preliminaries in graph theory that are used throughout the thesis. For a more detailed treatment of the fundamental concepts of graph theory, the reader is referred to [16].

Graphs are mathematical objects that are used to model a great amount of real-life phenomena. A graph $G = (V, E)$ consists of a set of vertices V, and a set of edges E that represent connections between pairs of vertices. In simple and undirected graphs, each edge $\{i, j\} \in E$ is an unordered pair with $i \neq j$, which implies a bidirectional connection between i and j. Furthermore, in simple graphs multiple edges between two vertices and edges between a vertex and itself are not allowed. In the scope of this thesis, unless otherwise specified, all graphs are considered to be simple and undirected.

Let $G = (V, E)$ be a simple graph with nonempty vertex set V and edge set E. The number of vertices in a graph, $|V|$, is called the order of the graph. Two vertices $i, j \in V$ are said to be adjacent if $\{i, j\} \in E$. For an edge $\{i, j\} \in E$, the vertices i and j are the endpoints of $\{i, j\}$. The vertex i (respectively, j) and edge $\{i, j\}$ are said to be incident to each other. Two distinct edges are adjacent if they are incident to a common vertex.

For a vertex $v \in V$, the open neighborhood of v, denoted by $\delta(v)$, is defined as the set of all vertices adjacent to v, i.e., $\delta(v) = \{j \in V \mid \{v, j\} \in E\}$. The closed neighborhood of v, denoted by $\Delta(v)$, is defined as $\delta(v) \cup \{v\}$. Similarly, for a subset of vertices $S \subseteq V$, the open neighborhood of S, denoted by $\delta(S)$, is the union of the open neighborhoods of its vertices, i.e., $\delta(S) = \bigcup_{v \in S} \delta(v)$. The closed neighborhood

of S, denoted by $\Delta(S)$, is the union of the closed neighborhoods of its vertices, i.e., $\Delta(S) = \bigcup_{v \in S} \Delta(v)$. Equivalently, $\Delta(S) = S \cup \delta(S)$.

For a vertex $v \in V$, $\partial(v)$ is defined to be the set of edges incident to v, i.e., $\partial(v) = \{e \in E \mid v \in e\}$. Similarly, for a subset of vertices $S \subseteq V$, the set $\partial(S)$ is defined to be the set of edges incident to at least one vertex in S, i.e., $\partial(S) = \bigcup_{v \in S} \partial(v)$.

The degree of a vertex v, denoted by $deg(v)$, is defined as the number of vertices that are adjacent to v, that is $deg(v) = |\delta(v)|$. A vertex is said to be a pendant vertex if $deg(v)$ = 1. Any vertex that is adjacent to a pendant vertex is called a support vertex. A vertex u is said to be universal if $deg(u) = |V| - 1$.

When defining a set for a graph, for instance the closed neighborhood of some vertex, if the underlying graph is not clear from context, it is denoted as a subscript of the set. For example, if there are two graphs that have been introduced, say G and H, $\delta_G(i)$ would denote the open neighborhood of vertex i in graph G and $\Delta_H(j)$ would be the closed neighborhood of j in the graph H. For conciseness, the subscript is omitted if the underlying graph is clear from context.

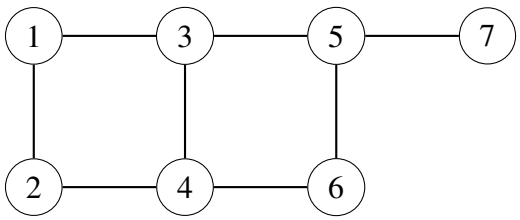


Figure 2.1: Example graph $G_1 = (V_1, E_1)$

Some of the definitions are illustrated using the graph $G_1 = (V_1, E_1)$ in Figure 2.1. The vertex set of G_1 is $V_1 = \{1, 2, 3, 4, 5, 6, 7\}$, and the edge set is $E_1 = \{\{1, 2\}, \{1, 3\}, \{2, 4\}, \{3, 4\}, \{3, 5\}, \{4, 6\}, \{5, 6\}, \{5, 7\}\}$. G_1 is a simple graph because its edges are not directed, and there are no loops or multiple edges.

We begin by providing examples of neighborhoods and incident edge sets for the vertices 1 and 5. The open neighborhood, closed neighborhood, and the incident edge sets of the vertices 1 and 5 are,

$$\delta(1) = \{2, 3\}, \qquad \Delta(1) = \{1, 2, 3\}, \qquad \partial(1) = \{\{1, 2\}, \{1, 3\}\}$$
$$\delta(5) = \{3, 6, 7\}, \qquad \Delta(5) = \{3, 5, 6, 7\}, \qquad \partial(5) = \{\{3, 5\}, \{5, 6\}, \{5, 7\}\}$$

When the vertex set $\{1, 5\}$ is considered, the open neighborhood, closed neighborhood, and incident edge set of the vertex set $\{1, 5\}$ are given by:

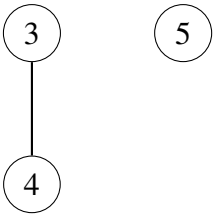
$$\delta(\{1, 5\}) = \delta(1) \cup \delta(5) = \{2, 3\} \cup \{3, 6, 7\} = \{2, 3, 6, 7\}$$
$$\Delta(\{1, 5\}) = \{1, 5\} \cup \delta(\{1, 5\}) = \{1, 2, 3, 5, 6, 7\}$$
$$\partial(\{1, 5\}) = \partial(1) \cup \partial(5) = \{\{1, 2\}, \{1, 3\}, \{3, 5\}, \{5, 6\}, \{5, 7\}\}$$

Furthermore, we have that $deg(5) = |\delta(5)| = 3$. Vertex 7 is a pendant vertex because $deg(7) = 1$. Vertex 5 is support vertex because it is adjacent to pendant vertex 7. In this graph, there are no universal vertices because no vertex is adjacent to all other vertices.

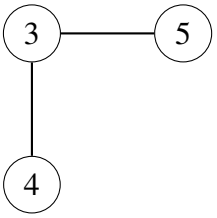
We next present further relevant definitions. A graph $H = (V', E')$ is a subgraph of $G = (V, E)$ if $V' \subseteq V$ and $E' \subseteq E$. Given a subset of vertices $S \subseteq V$, the induced subgraph on S, denoted by $G[S]$, is the subgraph whose vertex set is S and whose edge set consists of all edges in G that have both endpoints in S, that is, $G[S] = (S, E[S])$ where $E[S] = \{\{u, v\} \in E \mid u, v \in S\}$.

In Figure 2.2, we have several graphs to illustrate the definitions of subgraphs. The graph in Figure 2.2a is a subgraph of G_1 but not an induced subgraph because it misses the edge $\{3, 5\}$ which is in E_1. The graph in Figure 2.2b is an induced subgraph of G_1 as it has all the edges from G_1 that has both endpoints in the set $\{3, 4, 5\}$. Lastly, the graph in Figure 2.2c is not a subgraph of G_1 since it has the edge $\{4, 5\}$ which does not exist in the graph G_1.

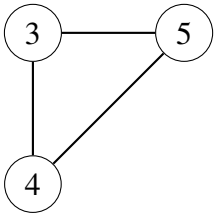
A path in $G = (V, E)$ is a sequence of vertices $P = (v_0, v_1, v_2, \ldots, v_k)$ such that $\{v_{i-1}, v_i\} \in E$ for all $i = 1, 2, \ldots, k$. A graph is connected if for every pair of vertices $i, j \in V$, there exists a path between i and j. If a graph is not connected, then



(a) Subgraph of G_1(not induced)



(b) Induced subgraph of G_1



(c) Not a subgraph of G_1

Figure 2.2: Graphs based on G_1 illustrating subgraphs

it is said to be a disconnected graph. A connected component of a graph $G = (V, E)$ is an induced subgraph $G' = (V', E')$ for some $V' \subseteq V$, such that:

(i) for every pair of vertices $u, v \in V'$, there exists a path in G' connecting u and v;

(ii) for every vertex $w \in V \setminus V'$, there exists no path in G connecting w to any vertex in V'.

Condition (ii) can be equivalently stated as: for every vertex $w \in V \setminus V'$, the set $V' \cup \{w\}$ does not satisfy condition (i). Connected components can also be referred to by their vertex sets.

A vertex $v \in V$ is called a cut vertex if its removal (together with all edges incident

to v) increases the number of connected components of G. An edge is called a bridge if its removal increases the number of connected components in G.

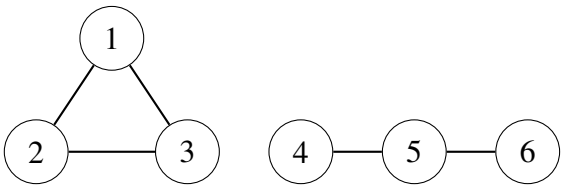


Figure 2.3: Example graph $G_2 = (V_2, E_2)$

In Figure 2.3, the graph $G_2 = (V_2, E_2)$ is not connected because there exists no path that connects vertices 3 and 4. There are two connected components in G_2, namely the induced subgraphs with vertex sets $\{1, 2, 3\}$ and $\{4, 5, 6\}$, respectively. Vertex 5 is a cut vertex because its removal increases the number of connected components in G_2. Edges $\{4, 5\}$ and $\{5, 6\}$ are bridges.

Next, some classes of graphs relevant to the next chapters are defined. A graph $T = (V, E)$ is called a tree if any of the following conditions hold:

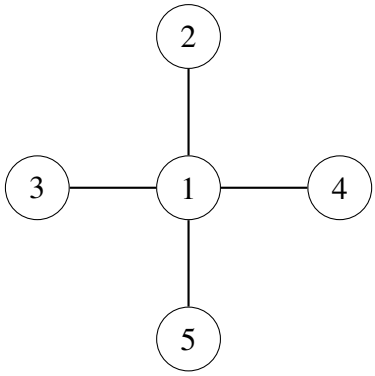
- T is connected and every edge of T is a bridge.
- T is connected and has $|E| = |V| - 1$.
- T is minimally connected (removal of any edge disconnects the graph).

Note that every tree of order $n \geq 2$ has at least two pendant vertices.

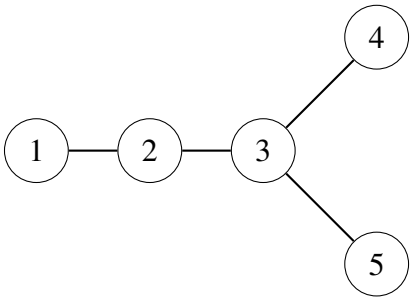
A graph $S = (V, E)$ is called a star graph if it is a tree and all but one of the vertices are pendant vertices. A star graph with n vertices is denoted as S_n. The non-pendant vertex is called the internal vertex and the pendant vertices are called outer vertices.

In Figure 2.4a, an example of a star graph with 5 vertices is provided. Figure 2.4b is another tree that is not a star graph.

A matching M in a graph $G = (V, E)$ is a subset of edges such that no two edges share a common endpoint. If $\{i, j\} \in M$ for some matching M, then we say that i and j are matched in M. If all vertices in V are matched in M, then M is said to be a perfect matching.



(a) Star graph S_5 with 5 vertices



(b) A tree

Figure 2.4: Examples of a star graph and a non-star tree on 5 vertices

A matching is said to be a perfect matching if all the vertices in V are an element of some edge in M.

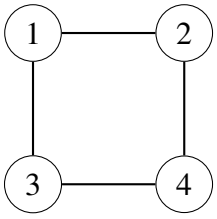


Figure 2.5: Example graph $G_3 = (V_3, E_3)$

In Figure 2.5, a graph $G_3 = (V_3, E_3)$ is provided. In G_3, the edge set $Q = \{\{1, 2\}\}$ is a matching because no two edges in Q are adjacent, however Q is not a perfect matching because vertices 3 and 4 are not matched in Q. The edge set $W = \{\{1, 2\}, \{3, 4\}\}$ is a perfect matching because no two edges in W are adjacent and all vertices are matched in W. Finally the edge set $P = \{\{1, 2\}, \{1, 3\}\}$ is not a matching because

$\{1, 2\}$ and $\{1, 3\}$ are adjacent.

CHAPTER 3

SOLVING THE MINIMUM WEIGHTED PAIRED-DOMINATING SET PROBLEM USING INTEGER PROGRAMMING

3.1 Problem Definition

Let $G = (V, E)$ be a graph. A subset S of V is a paired-dominating set if S is a dominating set and $G[S]$ has a perfect matching. Every graph without isolated vertices has a paired-dominating set. Motivated by the following application, paired-domination was introduced by Haynes and Slater [18]. Assume that each region of a museum is to be protected by a guard in the region or a guard in a neighboring region. If all the regions are protected, then the regions with the guards form a dominating set in the graph which is obtained by taking the regions as vertices and adding an edge between two vertices if the corresponding regions are neighbors. If, in addition, it is required that each guard is paired up with an adjacent guard so that they can back each other up, then the regions with the guards form a paired-dominating set.

The paired-domination number of a graph G is the minimum cardinality of a paired-dominating set of G. Let w_i be the weight associated with vertex $i \in V$ and α_e the weight associated with edge $e \in E$. If $e = \{i, j\}$, then we also use the notation α_{ij} for α_e. The minimum weighted paired-dominating set problem (MW-PDSP) is the problem of finding a dominating set S and a matching M in $G[S]$ with

$$\sum_{i \in S} w_i + \sum_{e \in M} \alpha_e$$

the minimum.

In graph G in Figure 3.1, consider the vertex set $V_1 = \{1, 2\}$. V_1 is a dominating set

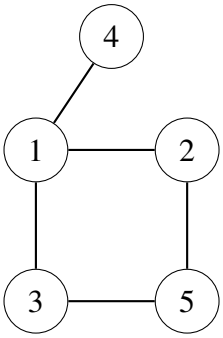


Figure 3.1: Example graph $G = (V, E)$

and $G[V_1]$ has a perfect matching $M_1 = \{\{1, 2\}\}$; hence V_1 is a paired-dominating set. The set $V_2 = \{1, 3, 4, 5\}$ is another paired-dominating set because it is a dominating set and in $G[V_2]$, $M_2 = \{\{1, 4\}, \{3, 5\}\}$ is a perfect matching.

On the contrary, the set $V_3 = \{1, 2, 3\}$ is not a paired-dominating set because the order of $G[V_3]$ is odd, which clearly cannot contain a perfect matching. $V_4 = \{1, 2, 3, 4\}$ is not a paired-dominating set either, because there are no perfect matchings in $G[V_4]$.

3.2 Literature Review

Most of the studies in the literature on paired-domination focus on the unweighted version of the MW-PDSP, i.e., $w_i = 1, \forall i \in V$ and $\alpha_e = 0, \forall e \in E$ and study the paired domination number for general or specific graphs, see e.g., [5, 14]. There are studies dealing with the weighted version of MW-PDSP, however such studies only consider weights on the vertices, i.e., $\alpha_e = 0, \forall e \in E$. In this thesis, we study the MW-PDSP in graphs where there are weights both on the vertices and the edges for the first time. Our results will continue to hold if there are no weights on the edges. It is important to note that when there are weights both on the vertices ($w_i, i \in V$) and edges ($\alpha_e, e \in E$) in the MW-PDSP, the problem can be reduced to an equivalent one with only edge weights, where the new weight α'_e of edge $e = \{i, j\}$ will be $\alpha'_e = \alpha_e + w_i + w_j$ and the new weight w'_i of vertex i is zero.

Paired-dominating sets were introduced in [18], where the authors proved that determining whether a graph has a paired-dominating set of size less than a given integer

k is NP-complete. In [4], the authors further show that this problem remains NP-complete even when restricted to bipartite graphs, chordal graphs, or split graphs. There are a number of studies in the literature investigating paired-domination in trees. The authors in [27] present a linear time algorithm to compute the paired-domination number in trees and characterize trees with equal domination and paired-domination numbers. The study in [20] characterizes the set of vertices of a tree that are contained in every, or in no, paired-dominating set of minimum cardinality. The authors in [3] study the MW-PDSP in trees where there are weights only on the vertices and propose a linear time dynamic programming style algorithm for its solution. Beyond trees, the paired-domination number can also be computed in polynomial time for some other classes of graphs, including block and interval graphs [4], strongly orderable graphs [26], and circular-arc graphs [22].

For a recent and comprehensive survey on paired-domination, including both structural and algorithmic results, we refer the reader to [8].

In this chapter, we study the minimum weighted paired-dominating set problem using integer programming techniques. To the knowledge of the author, this is the first study in the literature that considers edge weights in paired-domination and this is the first study that proposes an integer programming formulation for the MW-PDSP. After proposing an initial integer programming formulation, we make it stronger and show that the linear programming relaxation of the latter formulation solves the MW-PDSP in trees. Moreover, we propose several sets of valid inequalities for the paired dominating set polytope, and computationally investigate the strength of our formulation and the impact of the valid inequalities on random graphs.

3.3 Integer Programming and Paired-Domination in Trees

Given a graph $G = (V, E)$ and a weight w_i associated with each vertex $i \in V$, the minimum weighted dominating set problem (MW-DSP) is the problem of finding a dominating set of minimum weight. An integer programming formulation for the MW-DSP is given as follows in [2]:

$$
\begin{aligned}
\text{(IP-D)} \quad \text{minimize} \quad & \sum_{i \in V} w_i x_i \\
\text{subject to} \quad & \sum_{j \in \Delta(i)} x_j \geq 1, \quad \forall i \in V && (3.1) \\
& x_i \in \{0,1\}, \quad \forall i \in V && (3.2)
\end{aligned}
$$

Here, x_i is a binary variable that takes the value 1 if and only if vertex i is included (or selected) in the dominating set. The objective function minimizes the weight of the dominating set. Constraints (3.1) make sure that every vertex $i \in V$ is either included in the dominating set or is dominated by a vertex in the dominating set.

The formulation (IP-D) can be extended to obtain an integer programming formulation for the MW-PDSP. Keeping the x_i's in the formulation, we introduce for each edge $\{i,j\} \in E$ a binary variable y_{ij} which takes the value 1 if and only the edge $\{i,j\}$ is included in the perfect matching of $G[S]$, where $S = \{i \in V | x_i = 1\}$. An integer programming formulation for the MW-PDSP is given as follows

$$
\begin{aligned}
\text{(IP-PD-1)} \quad \text{minimize} \quad & \sum_{i \in V} w_i x_i + \sum_{\{i,j\} \in E} \alpha_{ij} y_{ij} \\
\text{subject to} \quad & \sum_{j \in \Delta(i)} x_j \geq 1, \quad \forall i \in V && (3.3) \\
& x_i = \sum_{j \in \delta(i)} y_{ij}, \quad \forall i \in V && (3.4) \\
& x_i \in \{0,1\}, \quad \forall i \in V && (3.5) \\
& y_{ij} \in \{0,1\}, \quad \forall \{i,j\} \in E && (3.6)
\end{aligned}
$$

In (IP-PD-1), Constraints (3.3) ensure that the selected vertices form a dominating set. Constraints in (3.4) make sure that x_i is equal to 1 if and only if some y_{ij} takes the value 1 for a vertex $j \in \delta(i)$ which forces x_j to take the value 1 as well. Therefore it is ensured that, each vertex selected in the dominating set is paired up with an adjacent selected vertex. By Constraint (3.4), it is also guaranteed that a selected vertex cannot be paired up with more than one adjacent selected vertex and therefore implying that the selected vertices form a paired-dominating set.

In the linear programming (LP) relaxation of (IP-PD-1), the constraints in (3.5) and (3.6) are replaced with

$$0 \leq x_i \leq 1, \quad \forall i \in V \tag{3.7}$$

$$0 \leq y_{ij} \leq 1, \quad \forall \{i, j\} \in E \tag{3.8}$$

respectively.

Let $P_1(G)$ be the polytope described by the LP relaxation of (IP-PD-1) for a given graph G, and let $F(G)$ be the set of incidence vectors of the paired dominating sets in G. We call $conv(F(G))$ - convex hull of $F(G)$ - as the paired-dominating set polytope.

Consider the tree T_1 in Figure 3.2 which is a star graph of order four with internal vertex 0 and three outer vertices $1, 2$, and 3.

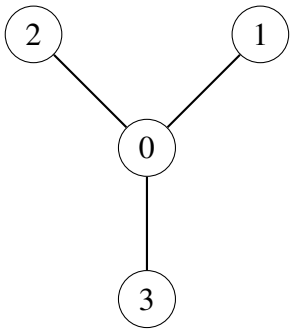


Figure 3.2: Star graph S_4

It can be shown that the following non-integral point (x', y') is an extreme point of $P_1(T_1)$ implying that $P_1(T_1)$ is not integral, that is $conv(F(T_1)) \neq P_1(T_1)$. Hence the linear programming relaxation of (IP-PD-1) does not always solve the MW-PDSP in trees.

$$\begin{pmatrix} x' \\ y' \end{pmatrix} = \begin{pmatrix} x'_0 \\ x'_1 \\ x'_2 \\ x'_3 \\ y'_{01} \\ y'_{02} \\ y'_{03} \end{pmatrix} = \begin{pmatrix} 3/4 \\ 1/4 \\ 1/4 \\ 1/4 \\ 1/4 \\ 1/4 \\ 1/4 \end{pmatrix}$$

Consider a paired-dominating set S of $G = (V, E)$. Note that any vertex not in S is dominated by a vertex in S as S is a dominating set and any vertex in S is dominated by some other vertex in S as S is a paired-dominating set. Therefore it can be concluded that if S is a paired-dominating set, then every vertex of G is dominated by some vertex in S. With this observation, we now provide a stronger formulation for the MW-PDSP.

$$\begin{aligned}
\text{(IP-PD-2)} \quad \text{minimize} \quad & \sum_{i \in V} w_i x_i + \sum_{\{i,j\} \in E} \alpha_{ij} y_{ij} \\
\text{subject to} \quad & \sum_{j \in \delta(i)} x_j \geq 1, \quad \forall i \in V && (3.9) \\
& x_i = \sum_{j \in \delta(i)} y_{ij}, \quad \forall i \in V && (3.4) \\
& x_i \in \{0, 1\}, \quad \forall i \in V && (3.5) \\
& y_{ij} \in \{0, 1\}, \quad \forall \{i, j\} \in E && (3.6)
\end{aligned}$$

The only difference of (IP-PD-2) from (IP-PD-1) is the constraint set (3.9). As the incidence vector of any paired-dominating set satisfies all the inequalities in (3.9), the formulation (IP-PD-2) is still valid for the MW-PDSP. Note that the non-integral vector (x', y') does not satisfy the inequality (3.9) when it is written for the internal vertex 0 of T_1.

Let $P_2(G)$ be the polytope described by the LP relaxation of (IP-PD-2) for a given graph G.

We first prove that for the support vertices, the corresponding x variables can be fixed.

Proposition 3.3.1. *Let $i \in V$ be a support vertex in $G = (V, E)$. Then for any $(\bar{x}, \bar{y}) \in P_2(G)$, we have that $\bar{x}_i = 1$.*

Proof. Let $j \in V$ be a pendant vertex that is adjacent to i. We have that $\bar{x}_i \geq 1$ by (3.9) written for j and $\bar{x}_i \leq 1$ by (3.7) written for i resulting in $\bar{x}_i = 1$. □

Note that Proposition 3.3.1 is stronger than stating that $\bar{x}_i = 1$ for every feasible vector $(\bar{x}, \bar{y})$ of (IP-PD-2) as $P_2(G)$ contains the feasible region of (IP-PD-2).

Does the LP relaxation of the formulation (IP-PD-2) solve the MW-PDSP in trees? In the search for an answer to this question, one can check whether the coefficient matrix of the LP relaxation of (IP-PD-2) is totally unimodular or not. When total unimodularity is checked, it can be seen that the coefficient matrix of the LP relaxation of (IP-PD-2) is totally unimodular for any tree of order less than or equal to four. However, for the following tree in Figure 3.3 which is of order five, the corresponding coefficient matrix is not totally unimodular.

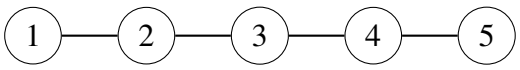


Figure 3.3: A tree for which the coefficient matrix of the LP relaxation of (IP-PD-2) is not totally unimodular

A part of the coefficient matrix of the LP relaxation of (IP-PD-2) for the tree in Figure 3.3 is

$$\left[\begin{array}{ccccc|cccc} 0 & \mathbf{1} & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ \mathbf{1} & 0 & \mathbf{1} & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & \mathbf{1} & 0 & \mathbf{1} & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & \mathbf{1} & 0 & \mathbf{1} & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & \mathbf{1} & 0 & 0 & 0 & 0 & 0 \\ \hline \mathbf{1} & 0 & 0 & 0 & 0 & \mathbf{-1} & 0 & 0 & 0 \\ 0 & \mathbf{1} & 0 & 0 & 0 & \mathbf{-1} & \mathbf{-1} & 0 & 0 \\ 0 & 0 & \mathbf{1} & 0 & 0 & 0 & \mathbf{-1} & \mathbf{-1} & 0 \\ 0 & 0 & 0 & \mathbf{1} & 0 & 0 & 0 & \mathbf{-1} & \mathbf{-1} \\ 0 & 0 & 0 & 0 & \mathbf{1} & 0 & 0 & 0 & \mathbf{-1} \end{array}\right]$$

where the first five rows correspond to the inequalities (3.9), the last five rows to the inequalities (3.4), the first five columns to the x variables, and the last four columns to the y variables. The following submatrix obtained by taking rows $2, 3, 4, 6, 7, 9, 10$ and columns $1, 2, 3, 4, 5, 6, 9$ has determinant 2.

$$\left[\begin{array}{ccccc|cc} \mathbf{1} & 0 & \mathbf{1} & 0 & 0 & 0 & 0 \\ 0 & \mathbf{1} & 0 & \mathbf{1} & 0 & 0 & 0 \\ 0 & 0 & \mathbf{1} & 0 & \mathbf{1} & 0 & 0 \\ \hline \mathbf{1} & 0 & 0 & 0 & 0 & \mathbf{-1} & 0 \\ 0 & \mathbf{1} & 0 & 0 & 0 & \mathbf{-1} & 0 \\ 0 & 0 & 0 & \mathbf{1} & 0 & 0 & \mathbf{-1} \\ 0 & 0 & 0 & 0 & \mathbf{1} & 0 & \mathbf{-1} \end{array}\right]$$

Even though the coefficient matrix of the LP relaxation of (IP-PD-2) is not always totally unimodular for trees, we will show that $P_2(T)$ is always integral for every tree T implying that the LP relaxation of (IP-PD-2) solves the MW-PDSP in trees. Before proving the main theorem, we will prove that for any star graph S, $P_2(S)$ is integral.

Proposition 3.3.2. *Let $S = (V, E)$ be a star graph with internal vertex 0 and outer vertices $1, 2, \ldots, n$. We have that $P_2(S)$ is integral.*

Proof. There are n paired-dominating sets in S which are obtained by selecting the internal vertex and any of the outer vertices. We will show that any $(\bar{x}, \bar{y}) \in P_2(S)$

can be written as a convex combination of the incidence vectors of these n paired-dominating sets. As 0 is a support vertex, by Proposition 3.3.1, we have that $\bar{x}_0 = 1$. By (3.4), we have that $\bar{x}_0 = \bar{y}_{01} + \bar{y}_{02} + \cdots + \bar{y}_{0n} = 1$. We have that $\bar{x}_i = \bar{y}_{0i}$ for any $i \in \{1, 2, \ldots, n\}$ again by (3.4) which implies that $\bar{x}_1 + \bar{x}_2 + \cdots + \bar{x}_n = 1$. We can now write $(\bar{x}, \bar{y})$ as a convex combination of the incidence vectors of the paired-dominating sets of S as

$$\begin{pmatrix} \bar{x} \\ \bar{y} \end{pmatrix} = \begin{pmatrix} \bar{x}_0 \\ \bar{x}_1 \\ \bar{x}_2 \\ \vdots \\ \bar{x}_n \\ \bar{y}_{01} \\ \bar{y}_{02} \\ \vdots \\ \bar{y}_{0n} \end{pmatrix} = \bar{x}_1 \begin{pmatrix} 1 \\ 1 \\ 0 \\ \vdots \\ 0 \\ 1 \\ 0 \\ \vdots \\ 0 \end{pmatrix} + \bar{x}_2 \begin{pmatrix} 1 \\ 0 \\ 1 \\ \vdots \\ 0 \\ 0 \\ 1 \\ \vdots \\ 0 \end{pmatrix} + \cdots + \bar{x}_n \begin{pmatrix} 1 \\ 0 \\ 0 \\ \vdots \\ 1 \\ 0 \\ 0 \\ \vdots \\ 1 \end{pmatrix}.$$

□

We will now state the main theorem.

Theorem 3.3.3. *For any tree T, $P_2(T)$ is integral.*

We will prove Theorem 3.3.3 after stating and proving some claims. Any tree of order two or three is a star graph, and by Proposition 3.3.2, for such trees the polytope described by the LP relaxation of (IP-PD-2) is integral. Assume that Theorem 3.3.3 is not true. That means that there exists a smallest counterexample $T' = (V', E')$, i.e., a tree T' with $|E(T')|$ the smallest such that $P_2(T')$ is not integral. As $P_2(T')$ is not integral, it has a non-integral extreme point $(\bar{x}, \bar{y})$. We will study some properties of the point $(\bar{x}, \bar{y})$ and then show that it's existence results in a contradiction.

Claim 3.3.4. *Let $P' \subseteq V'$ be the set of pendant vertices of T'. We have that $\bar{x}_i > 0$ for every $i \in P'$.*

Proof of Claim 3.3.4 Assume to the contrary that $\bar{x}_i = 0$ for some $i \in P'$. Because i is a pendant vertex of T', i has exactly one adjacent vertex, say j, and by Proposition

3.3.1, we have that $\bar{x}_j = 1$. Removing the vertex i and the edge $\{i, j\}$ from T', we obtain a new tree, say T'', such that the subvector of $(\bar{x}, \bar{y})$ obtained by removing $\bar{x}_i$ and $\bar{y}_{ij}$, call it $(\bar{x}', \bar{y}')$, belongs to $P_2(T'')$. As T' is the smallest tree with $P_2(T')$ non-integral, we have that $P_2(T'')$ is integral. Therefore $(\bar{x}', \bar{y}')$ can be written as a convex combination of the incidence vectors of paired-dominating sets of T'' as

$$(\bar{x}', \bar{y}') = \sum_{t=1}^{k} \alpha_t (\bar{x}^t, \bar{y}^t),$$

where $0 < \alpha_t \leq 1$ for each $t \in \{1, 2, \ldots, k\}$, $\sum_{t=1}^{k} \alpha_t = 1$, and $(\bar{x}^t, \bar{y}^t)$ is the incidence vector of some paired-dominating set of T'' for each $t \in \{1, 2, \ldots, k\}$. Note that as $\bar{x}_j = \bar{x}'_j = 1$, it should be that $\bar{x}^t_j = 1$ for each $t \in \{1, 2, \ldots, n\}$. This means that $(\bar{x}', \bar{y}')$ is a convex combination of incidence vectors of paired-dominating sets of T'' such that in each paired-dominating set the vertex j is selected. Note that any paired-dominating set of T'' in which vertex j is selected is also a paired-dominating set of T' as the vertex i is dominated by j. Therefore adding two components with zero values (for $\bar{x}_i$ and $\bar{y}_{ij}$) to each vector $(\bar{x}^t, \bar{y}^t)$, we obtain an incidence vector of a paired-dominating set of T', call it (x^t, y^t). We then have

$$(\bar{x}, \bar{y}) = \sum_{t=1}^{k} \alpha_t (x^t, y^t),$$

showing that $(\bar{x}, \bar{y})$ can be written as a convex combination of incidence vectors of paired-dominating sets of T'. This contradicts to the fact that $(\bar{x}, \bar{y})$ is an extreme point of $P_2(T')$.

Claim 3.3.5. *We have that $\bar{y}_{ij} > 0$ for all $\{i, j\} \in E'$.*

Proof of Claim 3.3.5 Assume that $\bar{y}_{ij} = 0$ for some edge $\{i, j\} \in E'$. If i or j is a pendant vertex, say vertex i, then we have $\bar{x}_i = \bar{y}_{ij} = 0$ contradicting Claim 3.3.4.

We can assume that neither vertex i nor vertex j is a pendant vertex. As every edge is a bridge, removal of edge $\{i, j\}$ results in two connected components, say $T_I = (V_I, E_I)$ and $T_J = (V_J, E_J)$, where $i \in V_I$ and $j \in V_J$. We can represent $(\bar{x}, \bar{y})$

as $\begin{pmatrix} \bar{x} \\ \bar{y} \end{pmatrix} = \begin{pmatrix} \bar{x}_I \\ \bar{x}_J \\ \bar{y}_I \\ \bar{y}_J \\ \bar{y}_{ij} \end{pmatrix}$, where $\bar{x}_I, \bar{y}_I$ and $\bar{x}_J, \bar{y}_J$ refers to the x and y variables for the vertices and edges in T_I and T_J, respectively.

Using T_I, we construct a new tree $T_I^* = (V_I^*, E_I^*)$ by adding a new vertex k_1 and a new edge $\{i, k_1\}$ to T_I. Mathematically, $V_I^* = V_I \cup \{k_1\}$ and $E_I^* = E_I \cup \{i, k_1\}$.

Moreover, we create a new vector $\begin{pmatrix} \bar{x}_I^* \\ \bar{y}_I^* \end{pmatrix}$, where we choose $\bar{x}_{k_1}^* = \bar{y}_{i,k_1}^* = 1 - \bar{x}_i$. To make sure the new vector satisfies inequality (3.4), we choose $\bar{x}_i^* = 1$. The rest of the x and y variables are chosen to be the same as $(\bar{x}, \bar{y})$.

$$\begin{pmatrix} \bar{x}_I^* \\ \bar{y}_I^* \end{pmatrix} = \begin{pmatrix} (\bar{x}_v)_{v \in V_I \setminus \{i\}} \\ \bar{x}_i^* \\ \bar{x}_{k_1}^* \\ (\bar{y}_e)_{e \in E_I} \\ \bar{y}_{i,k_1}^* \end{pmatrix}, \quad \text{where } \bar{x}_i^* = 1, \quad \bar{x}_{k_1}^* = 1 - \bar{x}_i, \quad \bar{y}_{i,k_1}^* = 1 - \bar{x}_i.$$

Similarly, we introduce a new tree $T_J^* = (V_J^*, E_J^*)$, where $V_J^* = V_J \cup \{k_2\}$ and $E_J^* = E_J \cup \{j, k_2\}$. We define $\bar{x}_{k_2}^* = \bar{y}_{j,k_2}^* = 1 - \bar{x}_j$, and we choose $\bar{x}_j^* = 1$ to obtain the vector

$$\begin{pmatrix} \bar{x}_J^* \\ \bar{y}_J^* \end{pmatrix} = \begin{pmatrix} (\bar{x}_v)_{v \in V_J \setminus \{j\}} \\ \bar{x}_j^* \\ \bar{x}_{k_2}^* \\ (\bar{y}_e)_{e \in E_J} \\ \bar{y}_{j,k_2}^* \end{pmatrix}, \qquad \text{where } \bar{x}_j^* = 1, \quad \bar{x}_{k_2}^* = 1 - \bar{x}_j, \quad \bar{y}_{j,k_2}^* = 1 - \bar{x}_j.$$

A visual representation of the original graph T', and the newly created graphs T_I^* and T_J^* can be seen in Figure 3.4, Figure 3.5, and Figure 3.6, respectively.

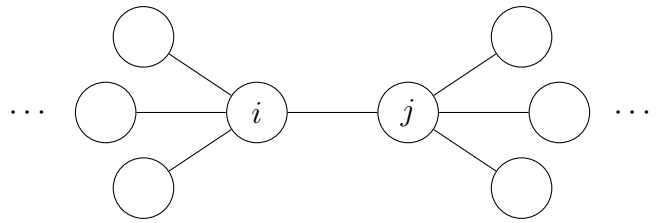


Figure 3.4: Original graph T'

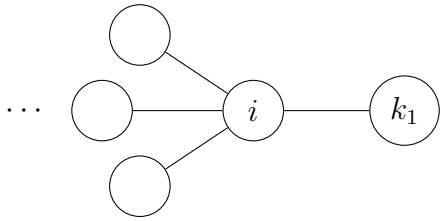


Figure 3.5: T_I^*

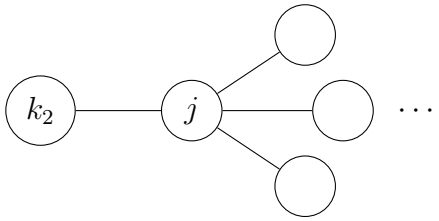


Figure 3.6: T_J^*

Lemma 1. *We have that* $\begin{pmatrix} \bar{x}_I^* \\ \bar{y}_I^* \end{pmatrix} \in P_2(T_I^*)$ *and* $\begin{pmatrix} \bar{x}_J^* \\ \bar{y}_J^* \end{pmatrix} \in P_2(T_J^*)$.

Proof of Lemma 1: We prove that $\begin{pmatrix} \bar{x}_I^* \\ \bar{y}_I^* \end{pmatrix} \in P_2(T_I^*)$ by showing that it satisfies all the inequalities defining $P_2(T_I^*)$. By our construction, $\begin{pmatrix} \bar{x}_I^* \\ \bar{y}_I^* \end{pmatrix}$ satisfies all the inequalities in (3.4), (3.7), (3.8) for all the vertices in V_I^*; and (3.9) written for vertices in $V_I \setminus \{i\}$. We need to show that the inequality (3.9) written for i and k_1 hold true. For both k_1 and i, the fact that $\bar{x}_i^* = 1$ implies that left-hand side of the inequality (3.9) is at least 1. Hence, when written for either vertex, the inequality is satisfied.

The proof for $\begin{pmatrix} \bar{x}_J^* \\ \bar{y}_J^* \end{pmatrix} \in P_2(T_J^*)$ is similar and omitted. □

Let $ext(T_I^*) = \{\nu_i : i \in N\}$ and $ext(T_J^*) = \{\omega_j : j \in M\}$ to be the sets of extreme points of $P_2(T_I^*)$ and $P_2(T_J^*)$, respectively where $N = \{1, 2, \ldots, n\}$ and $M = \{1, 2, \ldots, m\}$. T_I^* and T_J^* have a smaller number of vertices than T', therefore $P_2(T_I^*)$ and $P_2(T_J^*)$ must have integral extreme points.

By Lemma 1, $\begin{pmatrix} \bar{x}_I^* \\ \bar{y}_I^* \end{pmatrix} \in P_2(T_I^*)$, so we can write it as a convex combination of elements in $ext(T_I^*)$, that is; there exist nonnegative real numbers α_r for all $r \in N$ such that

$$\begin{pmatrix} \bar{x}_I^* \\ \bar{y}_I^* \end{pmatrix} = \sum_{r \in N} \alpha_r \nu_r, \quad \text{where} \quad \sum_{r \in N} \alpha_r = 1. \tag{3.10}$$

and similarly, as $\begin{pmatrix} \bar{x}_J^* \\ \bar{y}_J^* \end{pmatrix} \in P_2(T_J^*)$, we can write it as a convex combination of elements in $ext(T_J^*)$, that is there exists nonnegative real numbers α_p for all $p \in M$ such that

$$\begin{pmatrix} \bar{x}_J^* \\ \bar{y}_J^* \end{pmatrix} = \sum_{p \in M} \beta_p \omega_p, \quad \text{where} \quad \sum_{p \in M} \beta_p = 1. \tag{3.11}$$

We create a partition of $ext(T_I^*)$ into three sets A, B, C such that
$A = \{\nu \in ext(T_I^*) \mid y_{i,k_1} = 0\}$
$B = \{\nu \in ext(T_I^*) \mid y_{i,k_1} = 1, \exists q \in \delta_{T_I^*}(i) \setminus \{k_1\} \text{ such that } x_q = 1\}$
$C = \{\nu \in ext(T_I^*) \mid y_{i,k_1} = 1, \forall q \in \delta_{T_I^*}(i) \setminus \{k_1\}, x_q = 0\}$

and similarly, create a partition of $ext(T_J^*)$ into three sets D, E, F, such that

$D = \{\omega \in ext(T_J^*) \mid y_{j,k_2} = 0\}$
$E = \{\omega \in ext(T_J^*) \mid y_{j,k_2} = 1, \exists q \in \delta_{T_J^*}(j) \setminus \{k_2\} \text{ such that } x_q = 1\}$
$F = \{\omega \in ext(T_J^*) \mid y_{j,k_2} = 1, \forall q \in \delta_{T_J^*}(j) \setminus \{k_2\}, x_q = 0\}$

For notational convenience, we define $N_A, N_B, N_C \subseteq N$ to be the index sets of the extreme points in A, B, C, respectively; and similarly $M_D, M_E, M_F \subseteq M$ to be the index sets of the extreme points in D, E, F, respectively.

Lemma 2. *For the coefficients α_r and β_p used in Equations (3.10) and (3.11) to write the convex combinations, the following inequalities hold true:*

$$\sum_{p \in M_D} \beta_p \geq \sum_{r \in N_C} \alpha_r \tag{3.12}$$

$$\sum_{r \in N_A} \alpha_r \geq \sum_{p \in M_F} \beta_p \tag{3.13}$$

Proof of Lemma 2: We prove that inequality (3.12) holds true. Consider the variable $\bar{x}^*_{k_2}$ = 1 - $\bar{x}_j$. The x_{k_2} component of $\sum_{p \in M} \beta_p \omega_p$ has to be $1 - \bar{x}_j$. It is only the extreme points in $E \cup F$ that have nonzero x_{k_2} component, therefore we have that $\sum_{p \in M_E \cup M_F} \beta_p = 1 - \bar{x}_j$, and hence $\sum_{p \in M_D} \beta_p = \bar{x}_j$.

By Constraint (3.9) written for vertex i in T', we have that $\bar{x}_j + \sum_{q \in \delta_{T'}(i) \setminus \{j\}} \bar{x}_q \geq 1$. Subtracting $\bar{x}_j$ from both sides, we obtain $\sum_{q \in \delta_{T'}(i) \setminus \{j\}} x_q \geq 1 - \bar{x}_j$. Because the x_k values in the left-hand side of $\sum_{q \in \delta_{T'}(i) \setminus \{j\}} \bar{x}_q \geq 1 - \bar{x}_j$ do not change from $(\bar{x}, \bar{y})$ to $(\bar{x}^*_I, \bar{y}^*_I)$, we have $\sum_{q \in \delta_{T'}(i) \setminus \{j\}} \bar{x}^*_q \geq 1 - \bar{x}_j$. When the convex combination in (3.11) is written, the extreme points in F do not contribute positively to the x variable corresponding to any vertex in $\delta_{T'}(i) \setminus \{j\}$ (note that $\delta_{T'}(i) \setminus \{j\} = \delta_{T^*_J}(i) \setminus \{k_2\}$), and all positive contribution must come from the extreme points in $A \cup B$. Therefore, $\sum_{r \in N_A \cup N_B} \alpha_r \geq 1 - \bar{x}_j$. Finally, as (3.10) is a convex combination, we have $\sum_{r \in N} \alpha_r = 1$. Subtracting the inequality $\sum_{r \in N_A \cup N_B} \alpha_r \geq 1 - \bar{x}_j$, we are left with $\sum_{r \in N_C} \alpha_r \leq \bar{x}_j$. Therefore, we have $\sum_{p \in M_D} \beta_p \geq \sum_{r \in N_C} \alpha_r$, and inequality (3.12) holds true.

The proof for inequality (3.13) is similar and omitted. □

Next, we propose an algorithm that takes two extreme points (one from $ext(T^*_I)$ and one from $ext(T^*_J)$), and combine them into a vector with the same dimensions as $(\bar{x}, \bar{y})$, by dropping the vertices k_1 and k_2 and all variables related to them.

Algorithm 1: ConcatReduce

Input: Two extreme points, $\nu \in ext(T^*_I)$, $\omega \in ext(T^*_J)$

Output: A vector with the same dimension as $(\bar{x}, \bar{y})$

Concatenate ν and ω into a vector $\nu\omega$;

Remove components x_{k_1} and x_{k_2} from $\nu\omega$

Remove the component y_{i,k_1} from $\nu\omega$, and if $y_{i,k_1} = 1$, then set $x_i = 0$

Remove the component y_{j,k_2} from $\nu\omega$, and if $y_{j,k_2} = 1$, then set $x_j = 0$

Add the dimension y_{ij} to $\nu\omega$ and set $y_{ij} = 0$;

return *The resulting vector* $\nu\omega$;

Lemma 3. *Selecting a vector from the set C and pairing it with any vector from the set D, then applying Algorithm 1, yields a feasible point of $P_2(T')$. Similarly, choosing a vector from the set F and combining it with any vector from the set A using Algorithm 1 results in a feasible point of $P_2(T')$.*

Proof of Lemma 3: Let $\nu \in C$ and $\omega \in D$ be the inputs of Algorithm 1. The algorithm ensures $\nu\omega$ has the same dimensions with $\{\bar{x}, \bar{y}\}$. We will show that the output $vw \in P_2(T')$ by showing it satisfies all the constraints defining $P_2(T')$.

As $\omega \in D$, it satisfies the inequalities defining $P_2(T_J^*)$. Similarly, $\nu \in C$ implies v satisfies the inequalities defining $P_2(T_I^*)$. When Algorithm 1 outputs $\nu\omega$, it drops the x and y variables related to k_1 and k_2. The only neighbors of k variables are i and j, therefore all the constraints written for the vertices other than i and j are unchanged and remain satisfied.

We just need to show Constraints (3.9), (3.4), (3.7) and (3.8) written for i and j are satisfied. Constraints (3.7) and (3.8) written for i and j are clearly satisfied. When the algorithm adjusts $x_i = 0$ whenever $x_{k_1} = 1$, and similarly $x_j = 0$ whenever $x_{k_2} = 1$, Constraints (3.4) written for i and j also remain satisfied.

For all $\omega \in D$, we have $x_j = 1$, and x_j remains 1 after Algorithm 1 because $x_{k_2} = 0$. Furthermore, we have $y_{j,k_2} = 0$, therefore j must have a pair ℓ from the set $\ell \in \delta(j) \setminus \{k_1\}$, and $y_{j\ell} = 1$ which shows Constraint (3.9) written for j is satisfied. As $x_j = 1$, we have that vertex i is dominated and the Constraint (3.9) written for vertex i is satisfied.

Showing the statement holds true for $\nu \in A$ and $\omega \in F$ is similar and omitted. □

Lemma 4. *Taking a vector $\nu \in A \cup B$, and combining it with any vector $\omega \in E \cup F$ using Algorithm 1 results in a feasible point of $P_2(T')$.*

Proof of Lemma 4: The proof is similar to Lemma 3 and is omitted. □

Next, we will define Algorithm 2. It will output a set of tuples, where each tuple consists of a feasible point of $P_2(T')$ and a weight such that using these feasible points and weights, we can write a convex combination of $(\bar{x}, \bar{y})$.

To be used in the definition of Algorithm 2, we define the sets of tuples consisting of extreme points of $P_2(T_I^*)$ (respectively, $P_2(T_J^*)$), along with their associated weights given in Equation (3.10) (respectively, Equation (3.11)) as follows:

$$A_T = \{(\alpha_i, \nu_i) \mid i \in N_A\}, \quad B_T = \{(\alpha_i, \nu_i) \mid i \in N_B\}, \quad C_T = \{(\alpha_i, \nu_i) \mid i \in N_C\},$$
$$D_T = \{(\beta_j, \omega_j) \mid j \in M_D\}, \quad E_T = \{(\beta_j, \omega_j) \mid j \in M_E\}, \quad F_T = \{(\beta_j, \omega_j) \mid j \in M_F\}$$

Let Q be the output of Algorithm 2. Using elements of Q, we can write a convex combination of $(\bar{x}, \bar{y})$ in the following way:

$$\sum_{(\gamma_{\nu\omega}, \nu\omega) \in Q} \gamma_{\nu\omega} \cdot \nu\omega = (\bar{x}, \bar{y}), \quad \text{where} \quad \sum_{(\gamma_{\nu\omega}, \nu\omega) \in Q} \gamma_{\nu\omega} = 1. \tag{3.14}$$

We finish the proof by showing that Equation (3.14) holds true.

By Equation 3.12, we have that

$$\sum_{p \in D} \alpha_p \geq \sum_{r \in C} \beta_r$$

When Algorithm 2 is initiated, in each iteration, $\min\{\alpha_\nu, \beta_\omega\}$ is subtracted from both sides of the inequality. The right hand side will reach zero first due to Lemma 2, which ensures that the while loop always finds an element with nonzero β in D_T if there is an element with nonzero α in C_T. Similarly, the second while loop always finds an element with nonzero α in A_T if there is an element with nonzero β in F_T due to Lemma 2. Moreover, all the concatenations in the first two while loops result in feasible points of $P_2(T')$ by Lemma 3.

After the first two while loops terminate, the sum of α in C_T and β in F_T is zero and the sum of α in $A_T \cup B_T$ is equal to the sum of β in $C_T \cup D_T$. In each iteration, $\min\{\alpha_v, \beta_w\}$ is removed from both sides, which means that both sides reach zero at the same time. In the third while loop, all concatenations result in feasible points of $P_2(T')$ by Lemma 4.

When the third while loop ends, in all the sets $A_T, B_T, C_T, D_T, E_T, F_T$, all tuples have α and β values zero, and all tuples in the output Q contain feasible points

Algorithm 2: Matching

Input: $A_T, B_T, C_T, D_T, E_T, F_T$

Output: A set Q with tuples of weights and elements of $P_2(T')$ that is a convex combination of $(\bar{x}, \bar{y})$

Initialize $Q \leftarrow \emptyset$;

while *there exists a tuple* $(\alpha_p, \nu_p) \in C_T$ *with* $\alpha_p \geq 0$ **do**

  Pick the tuple $(\beta_q, \omega_q) \in D_T$ with the largest β_q;

  $\nu\omega \leftarrow$ ConcatReduce(ν_p, ω_q);

  Add $\big(\min\{\alpha_p, \beta_q\}, \nu\omega\big)$ to Q;

  Update (α_p, ν_p) in C_T: $\alpha_p \leftarrow \alpha_p - \min\{\alpha_p, \beta_q\}$;

  Update (β_ω, ω) in D_T: $\beta_q \leftarrow \beta_q - \min\{\alpha_p, \beta_q\}$;

end

while *there exists a tuple* $(\beta_q, \omega_q) \in F_T$ *with* $\beta_q \neq 0$ **do**

  Pick the tuple $(\alpha_p, \nu_p) \in A_T$ with the largest α_p;

  $\nu\omega \leftarrow$ ConcatReduce(ν_p, ω_q);

  Add $\big(\min\{\alpha_p, \beta_q\}, \nu\omega\big)$ to Q;

  Update (α_p, ν_p) in A_T: $\alpha_p \leftarrow \alpha_p - \min\{\alpha_p, \beta_q\}$;

  Update (β_q, ω_q) in F_T: $\beta_q \leftarrow \beta_q - \min\{\alpha_p, \beta_q\}$;

end

// At this point, there should be no more tuples with nonzero α or β in C_T and F_T

while *there exists a tuple* $(\alpha_p, \nu_p) \in A_T \cup B_T$ *with* $\alpha_p \geq 0$ **do**

  Pick the tuple $(\beta_q, \omega_q) \in D_T \cup E_T$ with the largest β_q;

  $\nu\omega \leftarrow$ ConcatReduce(ν_p, ω_q);

  Add $\big(\min\{\alpha_\nu, \beta_\omega\}, \nu\omega\big)$ to Q;

  Update (α_p, ν_p) in $A_T \cup B_T$: $\alpha_p \leftarrow \alpha_p - \min\{\alpha_p, \beta_q\}$;

  Update (β_q, ω_q) in $D_T \cup E_T$: $\beta_q \leftarrow \beta_q - \min\{\alpha_p, \beta_q\}$;

end

return Q;

$\nu\omega$ of $P_2(T')$ with a corresponding weight $\gamma_{\nu\omega}$ such that $\sum_{(\gamma_{\nu\omega},\nu\omega)\in Q}\gamma_{\nu\omega} = 1$, and $\sum_{(\gamma_{\nu\omega},\nu\omega)\in Q}\gamma_{\nu\omega}\cdot\nu\omega = (\bar{x},\bar{y})$, which shows that $(\bar{x},\bar{y})$ can be written as a convex combination of elements in $P_2(T')$. This contradicts that $(\bar{x},\bar{y})$ is an extreme point of $P_2(T')$.

Claim 3.3.6. *We have that $\bar{x}_i > 0$ for all $i \in V'$.*

Proof of Claim 3.3.6 Directly follows from Claim 3.3.5, as $\bar{y}_{ij} > 0$ for all $e \in E'$ implies $\bar{x}_i > 0$ for all $i \in V'$.

Claim 3.3.7. *We have that $\bar{y}_{ij} < 1$ for all $\{i,j\} \in E'$.*

Proof of Claim 3.3.7 Assume that $\bar{y}_{ij} = 1$ for some edge $\{i,j\}$. As T' is not a star graph, at least one of i and j is not a pendant vertex. This means that one of these two vertices, say i, is adjacent to at least one other vertex, say k. This implies by (3.4) that $\bar{y}_{ik} = 0$ contradicting Claim 3.3.5.

Claim 3.3.8. *If i is neither a pendant nor a support vertex and $\sum_{j\in\delta(i)}\bar{x}_j = 1$, then we have that $\bar{x}_i < 1$.*

Proof of Claim 3.3.8 We know by Claim 3.3.5 that $\bar{y}_{ij} > 0$ for all $\{i,j\} \in E'$. Assume to the contrary that i is neither a support nor a pendant vertex, $\sum_{j\in\delta(i)}\bar{x}_j = 1$, and $\bar{x}_i = 1$. By (3.4), we have that $\bar{x}_j \geq \bar{y}_{ij}$ for every $j \in \delta(i)$ and the inequality is strict if j is not a pendant vertex. This implies that $\sum_{j\in\delta(i)}\bar{x}_j \geq \sum_{j\in\delta(i)}\bar{y}_{ij}$ and the inequality is strict if at least one vertex in $\delta(i)$ is not a pendant vertex. As $1 = \bar{x}_i = \sum_{j\in\delta(i)}\bar{y}_{ij}$ and $\sum_{j\in\delta(i)}\bar{x}_j = 1$, we have that $\sum_{j\in\delta(i)}\bar{x}_i = \sum_{j\in\delta(i)}\bar{y}_{ij}$ which implies that all the vertices in $\delta(i)$ are pendant and hence T' is a star graph. This contradicts to the fact that T' cannot be a star graph.

Claim 3.3.9. *If i is a support vertex, then we have that $\sum_{j\in\delta(i)}x_j > 1$.*

Proof of Claim 3.3.9 Let i be a support vertex and for the sake of contradiction, assume that $\sum_{j\in\delta(i)}\bar{x}_j = 1$. By Proposition 3.3.2, T' cannot be a star graph. This implies that $\exists m, v \in V'$ such that $\{i,m\},\{m,v\} \in E'$. By Claim 3.3.5, we have that $\bar{y}_{mv} > 0$. Furthermore, $\sum_{j\in\delta(i)}\bar{x}_j > \bar{x}_i + \bar{y}_{mv} > 1$, and hence, the claim holds true.

Proof of Theorem 3.3.3 We are now ready to prove Theorem 3.3.3. Let P' and S' be the set of pendant and support vertices of T', respectively. Let us denote by I' the set $V' \setminus (P' \cup S')$. We rewrite the inequalities describing $P_2(T')$ as

$$\sum_{j \in \delta(i)} x_j \geq 1, \quad \forall\, i \in S' \tag{3.9a}$$

$$\sum_{j \in \delta(i)} x_j \geq 1, \quad \forall\, i \in I' \tag{3.9b}$$

$$x_i = \sum_{j \in \delta(i)} y_{ij}, \quad \forall\, i \in V' \tag{3.4}$$

$$x_i \geq 0, \qquad \forall\, i \in V' \tag{3.7a}$$

$$x_i \leq 1, \qquad \forall\, i \in P' \tag{3.7b}$$

$$x_i = 1, \qquad \forall\, i \in S' \tag{3.7c}$$

$$x_i \leq 1, \qquad \forall\, i \in I' \tag{3.7d}$$

$$y_{ij} \geq 0, \qquad \forall\, \{i,j\} \in E' \tag{3.8a}$$

$$y_{ij} \leq 1, \qquad \forall\, \{i,j\} \in E' \tag{3.8b}$$

Note that the inequality (3.9) written for a pendant vertex is implied by (3.7c) and is therefore not needed. As $(\bar{x}, \bar{y})$ is an extreme point of $P_2(T')$, it has to satisfy $|V'| + |E'|$ many linearly independent inequalities defining $P_2(T')$ at equality.

Let us now count the maximum number of inequalities that $(\bar{x}, \bar{y})$ may satisfy. By Claim 3.3.9, $(\bar{x}, \bar{y})$ does not satisfy any of the inequalities in (3.9a) at equality. Assume that $(\bar{x}, \bar{y})$ satisfies k inequalities of (3.9b) at equality. This implies by Claim 3.3.8 that $(\bar{x}, \bar{y})$ satisfies at most $|I'| - k$ many inequalities of (3.7d) at equality. $(\bar{x}, \bar{y})$ satisfies all the ($|V'|$ many) equalities in (3.4). $(\bar{x}, \bar{y})$ does not satisfy any of the inequalities in (3.7a), (3.7b), (3.8a), and (3.8b) at equality. Finally $(\bar{x}, \bar{y})$ satisfies $|S'|$ many inequalities due to (3.7c). In total, the number of linearly independent inequalities satisfied by $(\bar{x}, \bar{y})$ at equality is at most $k + |I'| - k + |V'| + |S'| = |V'| + |V'| - |P'|$. As each tree has at least two pendant vertices, this number is at most $2|V'| - 2$ which is strictly less than $|V'| + |E'| = 2|V'| - 1$. This shows that such a non-integral extreme point $(\bar{x}, \bar{y})$ cannot exist and completes the proof.

3.4 Valid Inequalities

In this section, we propose some valid inequalities to strengthen our formulation. Adding valid inequalities to an IP formulation makes it stronger and may lead to a reduction in solution times. For each proposed class of inequalities, a proof of validity as well as a fractional point in the LP relaxation that is cut off is provided. This fractional point is not cut off by any of the previous inequalities which shows that the new class of inequalities is not dominated by the previous ones.

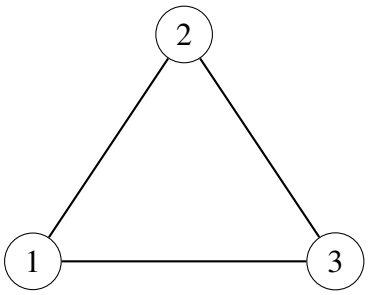
Proposition 3.4.1. *Let $G = (V, E)$ be a graph,and $v \in V$. Then, the following inequality is valid for $conv(F(G))$:*

$$\sum_{e \in \partial(\delta(v))} y_e \geq 1 \tag{3.15}$$

Proof: Let P be a paired-dominating set of G, and let M be a perfect matching in $G[P]$. As P is a paired-dominating set, for the vertex v, there must exist a vertex $w \in \delta(v)$ such that $w \in P$. This implies that there exists an edge $e \in M$ that is incident to w, and hence $\sum_{e \in \partial(\delta(v))} y_e \geq 1$.

□

In Figure 3.7, a fractional solution cut off by inequality 3.15 is provided. When the inequality is written for vertex 1, the inequality becomes $y_{12} + y_{13} + y_{23} \geq 1$ and cuts off the solution.



$$\begin{pmatrix} y_{12} \\ y_{23} \\ y_{13} \end{pmatrix} = \begin{pmatrix} 0.25 \\ 0.25 \\ 0.25 \end{pmatrix}$$

Figure 3.7: Example fractional solution that is cut off by inequality (3.15)

Proposition 3.4.2. *Let $G = (V, E)$ be a graph. Define the following sets E_D and E_N, where $E_D = \{e = \{u, w\} \in E \mid \forall v \in V, v \in \delta(u)$ or $v \in \delta(w)\}$ and $E_N = E \setminus E_D$.*

Then, the following inequality is valid for $conv(F(G))$*:*

$$2\sum_{e\in E_D} y_e + \sum_{e\in E_N} y_e \geq 2 \tag{3.16}$$

Proof: Let P be a paired dominating set of G, and let M be a perfect matching in $G[P]$. If there exists an edge $e_d \in E_D$ such that $e_d \in M$, then the inequality is valid. Otherwise, assume there exists no such edge e_d. From the definition of the set E_N, no single edge $e_n = \{u, v\} \in E_N$ is a dominating set of G. Therefore, there must exist at least two edges $e_1, e_2 \in E_N$ such that $e_1, e_2 \in M$.

□

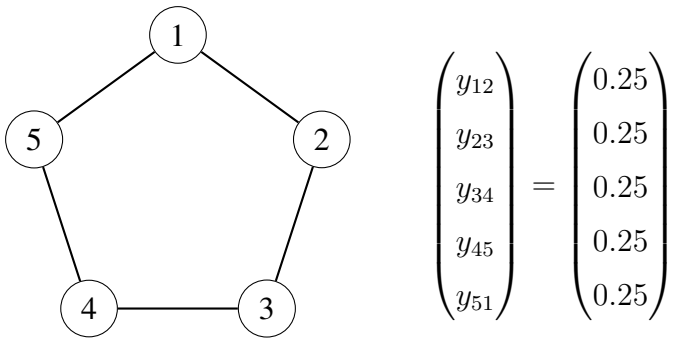


Figure 3.8: Example fractional solution that is cut off by inequality (3.16)

In Figure 3.8, none of the edges dominate the entire graph by themselves, so all the edges in the graph are in E_D. Inequality (3.16) becomes $y_{12}+y_{23}+y_{34}+y_{45}+y_{15} \geq 2$ and cuts off the given fractional solution.

Proposition 3.4.3. *Let* $G = (V, E)$ *be a graph,* $v \in V$*, and* $\{i, j\} \in E$ *such that* $i \notin \delta(v)$ *and* $j \notin \delta(v)$*.*

Then, the following inequality is valid for $conv(F(G))$*:*

$$(1 - y_{ij}) + \sum_{u\in\partial(\delta(v))\setminus\partial(\{i,j\})} y_u \geq 1. \tag{3.17}$$

Proof: Let $P \subseteq V$ be a paired-dominating set, and M be a perfect matching in G[P]. If $\{i, j\} \notin M$, then inequality (3.17) is satisfied. Assume $\{i, j\} \in M$. Because $i \notin \delta(v)$ and $j \notin \delta(v)$, to dominate v, there must exist a vertex $u \in \delta(v)$ such that $u \in P$, and there must exist be an edge $e \in M$ such that u is an endpoint of e.

Because $\{i, j\} \in M$ and M is a matching, it follows that i and j are not endpoints of e. The set $\partial(\delta(v)) \setminus \partial(\{i, j\}$ is precisely those edges that dominate v, and do not have i or j as endpoints. Therefore, at least one edge in $\partial(\delta(v)) \setminus \partial(\{i, j\}$ must be an element of M, and hence inequality (3.17) is satisfied. □

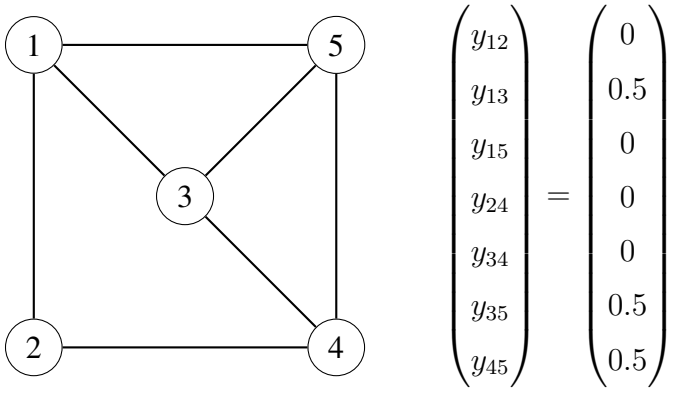


Figure 3.9: Example fractional solution that is cut off by inequality (3.17)

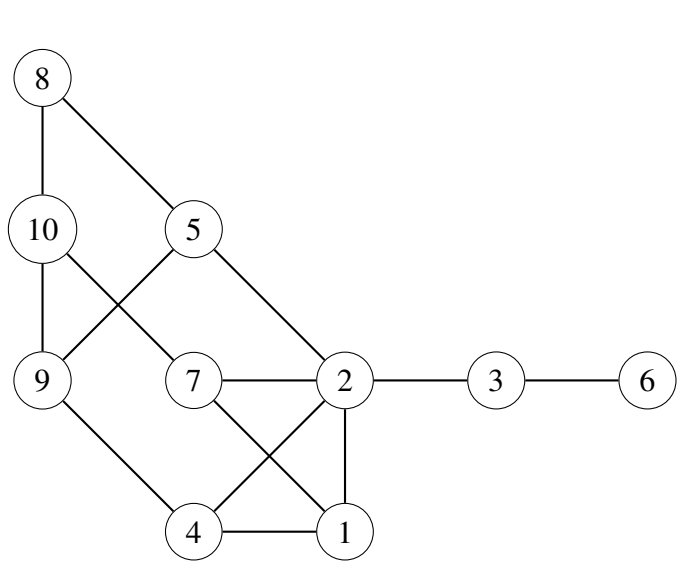
In Figure 3.9, choosing the edge $\{i, j\}$ as $\{3, 5\}$ and writing the inequality for vertex 2, the inequality becomes $(1 - y_{35}) + y_{12} + y_{24} \geq 1$ and cuts off the fractional solution given.

Proposition 3.4.4. *Let $G = (V, E)$ be a graph, and $w \in V$. For $k \geq 0$, let there be vertices $V' = \{v_1, v_2, \ldots, v_{2k+1}, v_{2k+2}\} \subseteq V \setminus \Delta(w)$ such that $\{v_1, v_2\}, \{v_2, v_3\}, \ldots, \{v_{2k+1}, v_{2k+2}\} \in E$. Define the set $E' = \{\{v_1, v_2\}, \{v_2, v_3\}, \ldots, \{v_{2k+1}, v_{2k+2}\}\}$. The following inequality is valid for $conv(F(G))$:*

$$\sum_{i \in \partial(\delta(w)) \setminus \partial(V')} y_i \geq (\sum_{j \in E'} y_j) - k \tag{3.18}$$

Proof: Let $P \subseteq V$ be a paired-dominating set, and M be a perfect matching in G[P]. If $|E' \cap M| \leq k$, then the inequality holds as the left hand side is always nonnegative. Assume instead that $|E' \cap M| > k$. Because M is a matching, consecutive edges $\{v_i, v_{i+1}\}$ and $\{v_{i+1}, v_{i+2}\}$ cannot appear together in the matching. Therefore, to have $|E' \cap M| > k$, it must be the case that $\{v_1, v_2\}, \{v_3, v_4\}, \ldots, \{v_{2k+1}, v_{2k+2}\} \in M$, and $\{v_2, v_3\}, \{v_4, v_5\}, \ldots, \{v_{2k}, v_{2k+1}\} \notin M$, and $|E' \cap M| = k+1$. Because w is not dominated by any of the edges in E', there must be an edge $e \in M \setminus E'$ such that $e \in \partial(\delta(w))$. Furthermore, $e \notin \partial(V')$ because $\{v_1, v_2\}, \{v_3, v_4\}, \ldots, \{v_{2k+1}, v_{2k+2}\} \in$

M. The left hand side of the inequality is precisely such edges, which ensures $\sum_{i \in \partial(\delta(w)) \setminus \partial(V')} y_i \geq 1$, and hence inequality (3.18) is satisfied. □



$$\begin{pmatrix} y_{12} \\ y_{14} \\ y_{17} \\ y_{23} \\ y_{24} \\ y_{25} \\ y_{27} \\ y_{36} \\ y_{49} \\ y_{58} \\ y_{59} \\ y_{7,10} \\ y_{8,10} \\ y_{9,10} \end{pmatrix} = \begin{pmatrix} 0.5 \\ 0.5 \\ 0 \\ 0 \\ 0 \\ 0.5 \\ 0 \\ 1 \\ 0.5 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0.5 \end{pmatrix}$$

Figure 3.10: Example fractional solution that is cut off by inequality (3.18)

Choosing k=1 and the path as $\{2, 1\}, \{1, 4\}$, and $\{4, 9\}$ and writing the inequality for vertex 8, the inequality becomes $y_{8,10} + y_{58} + y_{7,10} \geq y_{12} + y_{14} + y_{49} - 1$ and cuts off the fractional solution given in Figure 3.10.

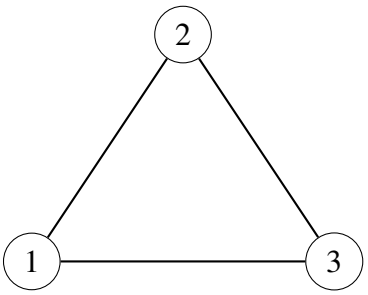
Remark: Inequality (3.18) reduces to inequality (3.17) for k=0.

Proposition 3.4.5. *Let $G = (V, E)$ be a graph, and let $S \subseteq V$ be a subset of vertices such that $|S|$ is odd. Then, the following inequality is valid for $conv(F(G))$:*

$$\sum_{i \in S, j \in S} y_{i,j} \leq \frac{|S| - 1}{2} \tag{3.19}$$

Proof: Inequalities (3.19) are valid for the matching polytope, and because the selected edges form a matching, they are also valid for the paired-dominating set polytope. □

In Figure 3.11, writing the inequality for the vertices $1, 2, 3$, we have $y_{12} + y_{13} + y_{23} \leq 1$ which cuts the given fractional solution off.



$$\begin{pmatrix} y_{12} \\ y_{23} \\ y_{13} \end{pmatrix} = \begin{pmatrix} 0.5 \\ 0.5 \\ 0.5 \end{pmatrix}$$

Figure 3.11: Example fractional solution that is cut off by inequality (3.19)

Remark: Inequalities (3.19) are polynomial-time separable, as shown in [25].

3.5 Computational Studies

In this section, we test the performance of our formulation and the effect of proposed valid inequalities on random graphs. We are interested in how fast our formulation is at solving the problem, and the number of integral solutions coming from the LP relaxation of our formulation for randomly generated graphs. Additionally, we are interested in how the addition of the proposed valid inequalities into the formulation affect these metrics.

Hardware and Software used: The experiments were conducted on a machine equipped with an Intel Xeon E-2246G CPU running at 3.60 GHz and 16 GB of RAM. We use CPLEX 22.1.1 to solve the integer and linear programs, and we solve the integer programs with presolve enabled.

Firstly, we look at the solution times of MW-PDSP instances on different graph sizes and densities. We solve the problem instances using three versions of (IP-PD-2); (IP-PD-2) without any valid inequalities, (IP-PD-2) with all valid inequalities except (3.17), (3.18) and (3.19) which is denoted by (IP-PD-3), and with all valid inequalities except (3.17) and (3.18) which is denoted by (IP-PD-4).

In (IP-PD-3) and (IP-PD-4), we decided not to include the inequalities (3.17) and (3.18) because there are a very large number of them and we do not have a fast sep-

aration algorithm. Preliminary experiments show that adding all inequalities (3.17) (which is a subset of inequalities in (3.18)) even for $k = 0$ causes the solver to run out of memory in relatively small instances. On the other hand, although there are exponentially many inequalities in (3.19), these inequalities are well-known in the literature to be polynomial-time separable, as shown in [25]. We implement the algorithm from [21] as a user cut callback in CPLEX, and investigate whether spending the time to find inequalities of type (3.19) improves the solution time.

For each $|V|$ and p pair where $|V| = n \in \{10, 20, 30, 40, 50, 75, 100, 150, 200, 300, 400, 500, 1000\}$, and density $p \in \{0.1, 0.25, 0.5, 0.75\}$, we create 5 random Erdős–Rényi graphs, and check that they are connected. We consider two cases; the weighted version where each edge is assigned a random weight distributed uniformly in the interval $[-1, 1]$, and the cardinality version where all edges and vertices have weight 1. For the weighted version, we decided to put weights only on the edges, since any instance of MW-PDSP involving both edge and vertex weights or only vertex weights can be transformed into an equivalent one with only edge weights, and putting weights only on the edges allows us to maintain a uniform distribution.

We set a timelimit of 3600 seconds for each instance, and report the number of instances an optimal solution is reached, average solution time for the instances where the optimal solution is reached and the average integrality gap for the instances which are not solved to optimality within the timelimit. The results can be seen in Table 3.1 and Table 3.2 for the unweighted and weighted cases, respectively.

In the unweighted version, for small to moderate instances until $n = 100$, all three of the formulations are able to solve the problem effectively. Interestingly, for $n = 100$ and $p = 0.25$, the solution time for (IP-PD-4) is higher than one might expect. When we check the instances, in one instance, the callback routine finds many inequalities to add, and slows the program down. For instances with $n = 150$ and $n = 200$, (IP-PD-2) is the fastest meanwhile (IP-PD-4) is unable to solve some instances in sparse graphs. Starting in $n = 300$, we see some instances that are not solved to optimality by any formulation. For large instances, (IP-PD-2) has the best gap in most cases, but there are instances where (IP-PD-3) performs the best; for example in $n = 300, p = 0.25$ where (IP-PD-2) is the only one that solves any instances

optimally and has a better gap in the instances it cannot solve. For large graphs with $p = 0.75$, while all three are able to solve all instances, (IP-PD-4) is the fastest. For instances with $n = 1000$, (IP-PD-3) runs out of memory in all instances except the ones with density $p = 0.1$, and (IP-PD-4) runs out of memory in all instances. In general, the formulations are much better at solving MW-PDSP when the input is a dense graph compared to sparse ones.

In the weighted version, (IP-PD-2) has the best solution time - it is able to solve all instances in less than 20 seconds. Similar to the unweighted case, (IP-PD-3) and (IP-PD-4) faces a memory constraint in $n = 1000$. All three formulations perform significantly better in the weighted version compared to the unweighted version in large graphs. Unlike the unweighted version, we solve the instances faster on sparse graphs compared to dense graphs.

Next, we investigate the number of integral solutions coming from the LP relaxation of our formulation, and how adding the proposed valid inequalities affect this count. We create 100 Erdős–Rényi graphs for each $|V| = n \in \{5, 10, 15, 20, 30, 40, 50, 75, 100\}$, and density $p \in \{0.1, 0.25, 0.5, 0.75\}$ pair, and ensure that they are connected. Similar to the analysis of solution times, we consider both the weighted and unweighted instances of the problem. Starting with formulation (IP-PD-1), we solve the LP relaxation for each instance and record the number of integral solutions obtained. We then incrementally strengthen the formulation, first by transitioning to (IP-PD-2), and subsequently by adding the proposed valid inequalities one at a time in the order presented in the thesis to observe the improvement made by each inequality. It should be noted that the results are sequence-dependent, that is adding the inequalities in a different order might change the impact of each inequality. The results are shown in Tables 3.3 and 3.4 for the unweighted and the weighted versions, respectively.

In both the unweighted and weighted versions, we see that the improved formulation from (IP-PD-1) to (IP-PD-2) significantly improves the number of integral solutions. The impact of the inequalities are different for the weighted and unweighted versions.

In the unweighted version, the most effective inequalities appear to be (3.15), (3.17), and (3.19). In the unweighted version, we can observe that all the proposed inequal-

ities make a difference, for example, for the instances with $n = 15$, $p = 0.25$, we see an increase in the number of integral solutions with the addition of each proposed inequality. The number of integral solutions goes from 10 to 71 with the addition of all inequalities.

In the weighted version, we perform exceptionally well when we add all the inequalities. Inequality (3.19) proves to be the most effective, whereas the remaining inequalities yield relatively minor improvements.

In both the unweighted and weighted versions of the problem, for all the instances with density $p = 0.75$, we obtain no fractional solutions from the LP relaxation after adding all the inequalities. The number of integral solutions is generally lower for sparse graph instances compared to dense graph instances, especially for the unweighted instances.

Table 3.1: Unweighted version: Solution times and average gap for three formulations across 100 Erdős–Rényi graphs for each (n, p) combination

		(IP-PD-2)			(IP-PD-3)			(IP-PD-4)		
n	p	#OPT	Time(s)	Gap	#OPT	Time(s)	Gap	#OPT	Time(s)	Gap
10	0.1	5	0.01	-	5	0.01	-	5	0.01	-
	0.25	5	0.01	-	5	0.01	-	5	0.01	-
	0.5	5	0.01	-	5	0.01	-	5	0.01	-
	0.75	5	0.01	-	5	0.01	-	5	0.01	-
20	0.1	5	0.01	-	5	0.01	-	5	0.01	-
	0.25	5	0.01	-	5	0.01	-	5	0.01	-
	0.5	5	0.01	-	5	0.01	-	5	0.01	-
	0.75	5	0.01	-	5	0.01	-	5	0.01	-
30	0.1	5	0.01	-	5	0.01	-	5	0.01	-
	0.25	5	0.02	-	5	0.02	-	5	0.02	-
	0.5	5	0.03	-	5	0.02	-	5	0.05	-
	0.75	5	0.01	-	5	0.02	-	5	0.02	-
40	0.1	5	0.02	-	5	0.01	-	5	0.06	-
	0.25	5	0.02	-	5	0.03	-	5	0.38	-
	0.5	5	0.01	-	5	0.05	-	5	0.03	-
	0.75	5	0.02	-	5	0.04	-	5	0.75	-
50	0.1	5	0.02	-	5	0.05	-	5	0.46	-
	0.25	5	0.09	-	5	0.10	-	5	0.99	-
	0.5	5	0.02	-	5	0.05	-	5	0.07	-
	0.75	5	0.03	-	5	0.08	-	5	0.07	-
75	0.1	5	0.10	-	5	0.19	-	5	19.17	-
	0.25	5	0.21	-	5	0.19	-	5	33.10	-
	0.5	5	0.04	-	5	0.17	-	5	0.23	-
	0.75	5	0.08	-	5	0.21	-	5	0.20	-
100	0.1	5	0.36	-	5	0.51	-	5	27.78	-
	0.25	5	0.70	-	5	4.93	-	5	238.98	-
	0.5	5	0.07	-	5	0.41	-	5	0.60	-
	0.75	5	0.17	-	5	0.49	-	5	0.45	-
150	0.1	5	10.26	-	5	15.20	-	2	2202.34	13.05%
	0.25	5	9.27	-	5	36.42	-	2	1976.19	20.86%
	0.5	5	0.42	-	5	1.91	-	5	3.03	-
	0.75	5	1.52	-	5	1.69	-	5	1.62	-
200	0.1	5	465.8	-	5	907.49	-	0	-	31.73%
	0.25	5	36.5	-	5	288.91	-	0	-	39.70%
	0.5	5	1.41	-	5	4.34	-	5	3.61	-
	0.75	5	3.42	-	5	4.03	-	5	3.85	-
300	0.1	0	-	28.2%	0	-	31.89%	0	-	43.58%
	0.25	0	-	41.3%	2	3306.04	25.44%	0	-	57.01%
	0.5	3	405.25	22.6%	0	-	33.33%	0	-	33.33%
	0.75	5	20.75	-	5	15.28	-	5	13.61	-
400	0.1	0	-	37.69%	0	-	42.59%	0		51.22%
	0.25	0	-	49.03%	0	-	55.20%	0		56.61%
	0.5	0	-	61.26%	0	-	44.60%	0		64.34%
	0.75	5	79.16	-	5	41.08	-	5	33.12	
500	0.1	0	-	44.36%	0	-	50.91%	0	-	57.17%
	0.25	0	-	51.07%	0	-	56.63%	0	-	59.56%
	0.5	0	-	64.55%	0	-	51.37%	0	-	65.34%
	0.75	5	223.11	-	5	99.92	-	5	92.10	-
1000	0.1	0	-	57.90%	0	-	60.87%	OOM		
	0.25	0	-	66.41%	OOM			OOM		
	0.5	0	-	66.52%	OOM			OOM		
	0.75	0	-	65.96%	OOM			OOM		

Table 3.2: Weighted version: Solution times and average gap for three formulations across 100 Erdős–Rényi graphs for each (n, p) combination

		(IP-PD-2)			(IP-PD-3)			(IP-PD-4)		
n	p	#OPT	Time(s)	Gap	#OPT	Time(s)	Gap	#OPT	Time(s)	Gap
10	0.1	5	0.15	-	5	0.01	-	5	0.01	-
	0.25	5	0.04	-	5	0.01	-	5	0.01	-
	0.5	5	0.03	-	5	0.01	-	5	0.01	-
	0.75	5	0.05	-	5	0.01	-	5	0.01	-
20	0.1	5	0.02	-	5	0.01	-	5	0.01	-
	0.25	5	0.02	-	5	0.01	-	5	0.01	-
	0.5	5	0.03	-	5	0.01	-	5	0.01	-
	0.75	5	0.04	-	5	0.01	-	5	0.01	-
30	0.1	5	0.01	-	5	0.01	-	5	0.01	-
	0.25	5	0.02	-	5	0.01	-	5	0.01	-
	0.5	5	0.02	-	5	0.02	-	5	0.02	-
	0.75	5	0.03	-	5	0.03	-	5	0.03	-
40	0.1	5	0.01	-	5	0.01	-	5	0.01	-
	0.25	5	0.05	-	5	0.02	-	5	0.02	-
	0.5	5	0.06	-	5	0.03	-	5	0.06	-
	0.75	5	0.03	-	5	0.05	-	5	0.06	-
50	0.1	5	0.01	-	5	0.01	-	5	0.01	-
	0.25	5	0.02	-	5	0.02	-	5	0.02	-
	0.5	5	0.03	-	5	0.05	-	5	0.08	-
	0.75	5	0.04	-	5	0.09	-	5	0.09	-
75	0.1	5	0.03	-	5	0.03	-	5	0.06	-
	0.25	5	0.04	-	5	0.05	-	5	0.15	-
	0.5	5	0.16	-	5	0.20	-	5	2.07	-
	0.75	5	0.30	-	5	0.34	-	5	2.75	-
100	0.1	5	0.03	-	5	0.03	-	5	0.05	-
	0.25	5	0.03	-	5	0.11	-	5	0.14	-
	0.5	5	0.05	-	5	0.29	-	5	0.32	-
	0.75	5	0.08	-	5	0.51	-	5	0.55	-
150	0.1	5	0.04	-	5	0.07	-	5	0.11	-
	0.25	5	0.07	-	5	0.33	-	5	0.51	-
	0.5	5	0.13	-	5	0.98	-	5	1.13	-
	0.75	5	0.17	-	5	1.83	-	5	1.90	-
200	0.1	5	0.06	-	5	0.16	-	5	0.51	-
	0.25	5	0.10	-	5	0.70	-	5	0.84	-
	0.5	5	0.22	-	5	2.36	-	5	2.64	-
	0.75	5	0.31	-	5	4.22	-	5	4.56	-
300	0.1	5	0.11	-	5	0.59	-	5	1.10	-
	0.25	5	0.21	-	5	2.26	-	5	2.79	-
	0.5	5	0.55	-	5	7.65	-	5	8.98	-
	0.75	5	0.91	-	5	14.43	-	5	15.62	-
400	0.1	5	0.19	-	5	1.05	-	5	2.39	-
	0.25	5	0.37	-	5	5.36	-	5	5.77	-
	0.5	5	1.14	-	5	20.29	-	5	24.15	-
	0.75	5	1.73	-	5	37.18	-	5	42.45	-
500	0.1	5	0.27	-	5	2.02	-	5	3.24	-
	0.25	5	0.76	-	5	11.09	-	5	13.63	-
	0.5	5	1.66	-	5	35.25	-	5	34.96	-
	0.75	5	2.76	-	5	85.29	-	5	95.18	-
1000	0.1	5	1.37	-	5	18.89	-	5	25.43	-
	0.25	5	4.37	-	5	222.67	-	5	311.68	-
	0.5	5	9.33	-	OOM			OOM		
	0.75	5	18.34	-	OOM			OOM		

Table 3.3: Unweighted version: Number of integral solutions of the LP relaxation across 100 Erdős–Rényi graphs for each (n, p) combination.

n	p	(IP-PD-1)	(IP-PD-2)	(IP-PD-2)+ (3.15)	(IP-PD-2)+ (3.15),(3.16)	(IP-PD-2)+ (3.15),(3.16),(3.17)	(IP-PD-2)+ (3.15),(3.16),(3.17),(3.19)
5	0.1	39	97	97	97	99	100
	0.25	37	88	87	87	89	100
	0.5	12	68	75	75	77	100
	0.75	1	26	84	84	84	100
10	0.1	6	77	82	83	87	96
	0.25	6	54	64	73	80	92
	0.5	18	35	68	87	88	95
	0.75	1	3	91	91	91	100
15	0.1	2	76	81	81	81	91
	0.25	10	44	55	60	68	71
	0.5	5	16	47	91	93	93
	0.75	0	0	87	87	87	100
20	0.1	0	71	72	72	72	84
	0.25	0	13	20	31	31	31
	0.5	1	9	24	77	77	77
	0.75	0	0	99	99	99	100
30	0.1	0	54	58	58	59	68
	0.25	0	1	3	5	5	5
	0.5	0	0	1	56	71	71
	0.75	0	0	100	100	100	100
40	0.1	0	20	23	24	24	24
	0.25	0	0	0	0	1	1
	0.5	0	0	1	53	56	56
	0.75	0	0	100	100	100	100
60	0.1	0	13	14	14	14	14
	0.25	0	0	0	0	0	1
	0.5	0	0	1	41	41	41
	0.75	0	0	100	100	100	100
80	0.1	0	0	0	0	0	0
	0.25	0	0	0	0	0	1
	0.5	0	0	1	21	21	21
	0.75	0	0	100	100	100	100
100	0.1	0	0	0	0	0	0
	0.25	0	0	0	0	0	0
	0.5	0	0	0	19	19	23
	0.75	0	0	100	100	100	100

Table 3.4: Weighted version: Number of integral solutions of the LP relaxation across 100 Erdős–Rényi graphs for each (n, p) combination.

n	p	(IP-PD-1)	(IP-PD-2)	(IP-PD-2)+ (3.15)	(IP-PD-2)+ (3.15),(3.16)	(IP-PD-2)+ (3.15),(3.16),(3.17)	(IP-PD-2)+ (3.15),(3.16),(3.17),(3.19)
5	0.1	53	99	99	100	100	100
	0.25	74	98	98	99	99	100
	0.5	85	96	96	96	99	100
	0.75	88	91	91	91	92	100
10	0.1	55	95	95	95	96	100
	0.25	73	96	96	96	99	100
	0.5	92	93	93	93	93	100
	0.75	83	83	83	83	83	100
15	0.1	54	99	100	100	100	100
	0.25	84	92	93	93	93	100
	0.5	73	73	73	73	73	100
	0.75	54	54	54	54	54	100
20	0.1	43	94	94	94	96	99
	0.25	82	89	89	89	89	100
	0.5	66	66	66	66	66	100
	0.75	72	72	72	72	72	100
30	0.1	54	93	94	94	98	100
	0.25	76	76	76	76	76	100
	0.5	62	62	62	62	62	100
	0.75	68	68	68	68	68	100
40	0.1	73	90	90	90	90	100
	0.25	49	49	49	49	49	100
	0.5	60	60	60	60	60	100
	0.75	78	78	78	78	78	100
60	0.1	77	77	77	77	77	100
	0.25	51	51	51	51	51	100
	0.5	66	66	66	66	66	100
	0.75	59	59	59	59	59	100
80	0.1	77	77	77	77	77	100
	0.25	46	46	46	46	46	100
	0.5	65	65	65	65	65	100
	0.75	66	66	66	66	66	100
100	0.1	63	63	63	63	63	100
	0.25	47	47	47	47	47	100
	0.5	62	62	62	62	62	100
	0.75	60	60	60	60	60	100

3.6 Conclusion

In this thesis, we present an integer programming formulation for MW-PDSP for the first time and initiate the study of the paired-dominating set polytope. We give an inequality description of the paired-dominating set polytope when the underlying graph is a tree. We propose several sets of valid inequalities, and the computational study of random graphs show that our formulation and valid inequalities are effective at solving MW-PDSP, particularly in dense instances.

For graph classes where the unweighted version of MW-PDSP is known to be solvable in polynomial time, for example for block graphs and circular-arc graphs, characterization of the paired-dominating set polytope could be a future research direction. Additionally, one could search for a linear-time algorithm for the MW-PDSP in trees when there are both vertex and edge weights.

CHAPTER 4

A ROW GENERATION APPROACH TO THE MINIMUM WEIGHTED K-DEFENSIVE SET PROBLEM

An important application area of domination is network security, where the dominating set is often considered to be a placement of police officers or surveillance cameras. With the motivation to model network security problems, defensive domination was introduced by Farley and Proskurowski in [13]. Defensive domination aims to defend against multiple simultaneous attacks to the nodes of a network. The next two sections present a formal definition of the problem followed by a comprehensive literature review.

4.1 Problem Definition

For a graph $G = (V, E)$, an *attack* $A \subseteq V$ is any subset of vertices, and each element of A is called an *attacker*. A *k-attack* is an attack with $|A| \leq k$. Given an attack A, and a set of defenders $D \subseteq V$, we say that D *defends against* A if there exists a surjective map $f\colon D \to A$ such that for each defender $d \in D$, $f(d) \in \Delta(d)$. D is said to be a *k-defensive set* (or a k-defensive dominating set) if it can defend against every k-attack.

In plain words, for every possible attack of size at most k, there needs to be an assignment of defenders such that: (i) each defender defends against at most one attacker, (ii) defenders can only defend a vertex in their closed neighborhood.

Note that if a defender at some vertex j is defending (is assigned to) some other vertex, then j is not considered to be defended by the defender at j.

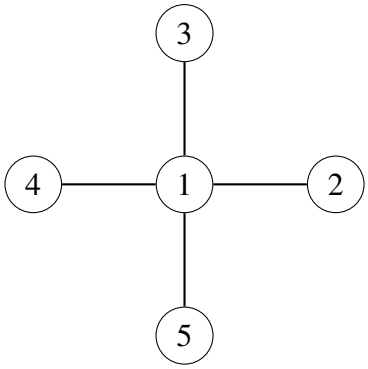


Figure 4.1: Example graph S_5 to illustrate k-defensive sets

Consider the star graph of order 5 in Figure 4.1, and assume $k = 2$. Let $V_1 = \{1, 2, 3\}$. Is V_1 a k-defensive (since k=2, 2-defensive) set? The answer is no, because it is not possible to defend against the attack $\{4, 5\}$. Within V_1, the only vertex that can defend vertices 4 and 5 is vertex 1. However, since vertex 1 can be assigned to defend only one of them, whichever of 4 or 5 it is not assigned to will remain undefended. Another candidate could be the set $V_2 = \{2, 3, 4\}$. Clearly this is also not 2-defensive because any attack that includes vertex 5 cannot be defended against.

Consider the vertex set $V_3 = \{1, 2, 3, 4\}$. We enumerate all the cases to show that V_3 is a 2-defensive set. If the attack does not contain vertex 5, then each vertex in the attack defends itself. If the attack is $\{1, 5\}$, vertex 1 defends vertex 5, and an outer vertex, say 2, defends vertex 1. Otherwise, the attack is on vertex 5 and an outer vertex, without loss of generality say $\{2, 5\}$. Then vertex 2 defends itself and vertex 1 defends vertex 5.

Another 2-defensive set is $V_4 = \{2, 3, 4, 5\}$. Both V_3 and V_4 are also 3-defensive and 4-defensive sets of the graph in Figure 4.1, but they are not 5-defensive sets since we need at least 5 defenders to defend an attack on 5 vertices.

Given a graph G, and an integer $k \geq 1$, the problem of finding a k-defensive set with minimum cardinality in G is called as the MIN-K-DEF-DOM problem. For $k = 1$, this problem is equivalent to finding a minimum cardinality dominating set in G, which is known to be NP-hard. The problem is only of independent interest when $k \geq 2$. In this chapter of the thesis, we will focus on a generalization of MIN-K-DEF-DOM, called MW-K-DEF-DOM problem, where the goal is to find the k-defensive set of

minimum weight, where the weight of a k-defensive set D is the sum of the weights of the vertices in D.

4.2 Literature Review

Defensive domination was introduced in [13], due to its applications in network security. In the same paper, the authors give a polynomial-time algorithm (with respect to both instance size and k) that finds a k-defensive set with minimum cardinality in trees. Despite its close relation to network security and domination, defensive domination has a limited literature. In [7], the authors introduce a generalization of MIN-K-DEF-DOM problem, where they allow attackers and defenders to be multisets, and apply it to a variant of cops-and-robber game.

In [11], the authors show that when k is treated as a fixed parameter, the decision version of MIN-K-DEF-DOM is NP-complete for any positive integer k even when the input is a split graph. On the contrary, when both k and the instance size are part of the input, they prove that the problem is not in NP by showing it is co-NP-hard to determine whether a given set is a k-defensive set or not. In this context, treated as a fixed parameter means analyzing the problem with respect to a fixed k and studying how the complexity scales with the instance size in the input graph. In contrast, when k is a part of the input, the computational complexity is evaluated in terms of both k and instance size. Furthermore, they propose algorithms that run in linear time in the instance size and k for paths, cycles, co-chain graphs, and threshold graphs.

In [10], the authors develop a polynomial-time algorithm (even in k) to solve MIN-K-DEF-DOM for interval graphs. In [19], for a fixed k, it is shown that the decision version of MIN-K-DEF-DOM is NP-complete even when restricted to bipartite graphs. The authors further show that the problem is APX-complete for bounded degree graphs.

In this chapter of the thesis, we consider the MW-K-DEF-DOM problem for the first time. In Section 4.3, we propose an integer programming formulation for the MW-K-DEF-DOM problem. This formulation has exponentially many inequalities, and to tackle this, a row generation approach based on the IP formulation is developed in

Section 4.4. Several sets of valid inequalities are given in Section 4.5, and we test our formulation on random graphs in Section 4.6.

4.3 An Integer Programming Formulation for the MW-K-DEF-DOM Problem

In this section, we propose an integer programming formulation for the MW-K-DEF-DOM problem. Before we state the IP formulation, we present a result from [10] that will be helpful in our formulation:

Theorem 4.3.1. *(Observation 2 in [10]) Let $G = (V, E)$ be a graph. $D \subseteq V$ is k-defensive if and only if $|\Delta(A) \cap D| \geq |A|$ for every set A with at most k vertices such that $G^2[A]$ is connected.*

For a graph $G = (V, E)$, the graph G^2 is the graph with the same vertex set as G, and there is an edge between two distinct vertices i and j in G^2 if and only if the distance between i and j in G is at most 2.

Define $\mathcal{A}_k$ to be the set of attacks A of size at most k such that $G^2[A]$ is connected. Mathematically, $\mathcal{A}_k = \{A \subseteq V \mid |A| \leq k,\ G^2[A] \text{ is connected}\}$.

Given a graph $G = (V, E)$, a positive integer k, and a weight w_i for each vertex $i \in V$, an integer programming formulation for the MW-K-DEF-DOM problem is given as follows:

$$\text{(IP-DEF-DOM)} \qquad \min \sum_{i \in V} w_i x_i \tag{4.1}$$

s.t.

$$\sum_{i \in \Delta(A)} x_i \geq |A|, \quad \forall A \in \mathcal{A}_k \tag{4.2}$$

$$x_i \in \{0, 1\}, \qquad \forall i \in V \tag{4.3}$$

Here, x_i is a binary variable that takes the value 1 if and only if vertex i is included in the k-defensive set. The objective function (4.1) aims to minimize the total weight

of the selected vertices. Constraints (4.2) come directly from Theorem 4.3.1 and they collectively ensure all k-attacks can be defended.

It is worth noting that the Constraints (4.2) are not required in (IP-DEF-DOM) for attacks A for which $G^2[A]$ is disconnected, however including them would also result in a correct formulation. One can show that given an attack A with $G^2[A]$ disconnected, Constraint (4.2) written for attack A can be obtained by summing Constraints (4.2) written for the vertex sets of connected components in $G^2[A]$.

(IP-DEF-DOM) has $O(|V|)$ many variables and has $O(|V|^k)$ many constraints. This means an exponential number of constraints in k. In our preliminary experiments, we have seen that the computer runs out of memory even in small instances. To tackle this, we propose a row generation procedure based on (IP-DEF-DOM) in the next section.

4.4 A Row Generation Approach

In this section, we propose a row generation approach based on (IP-DEF-DOM). Due to the exponential number of constraints of type (4.2) with respect to k, it is impractical to use (IP-DEF-DOM) directly on instances with many attackers. In the row generation procedure, rows corresponding to Constraints (4.2) will be generated only when they are violated. Therefore, we define the separation problem of Constraints (4.2), which we call FIND-ATTACK-GIVEN-DEFENSE: Given $D \subseteq V$ and the corresponding incidence vector $\bar{x}$ of D, try to find an inequality of the form (4.2) that cuts off $\bar{x}$, or conclude that $\bar{x}$ is a feasible solution to (IP-DEF-DOM). Equivalently, this problem can also be interpreted as, given a set of defenders D, either find an attack A that cannot be defended by D, or conclude that no such attack exists, i.e., D is a k-defensive set and $\bar{x}$ is a feasible solution of (IP-DEF-DOM). FIND-ATTACK-GIVEN-DEFENSE has already been studied in the literature before. In [11], the authors show that FIND-ATTACK-GIVEN-DEFENSE is co-NP-hard when k is a part of the input. However, no algorithm to solve FIND-ATTACK-GIVEN-DEFENSE has been developed yet.

Given a graph $G = (V, E)$ and a set D with the incidence vector $\bar{x}$, we next propose

an integer programming formulation with the following decision variables: y_i takes the value 1 if and only if vertex i is included in the attack, and z_j takes the value 1 if and only if a vertex in the closed neighborhood of vertex j is included in the attack.

$$
\begin{aligned}
(Separation - IP) \qquad \min \quad & \sum_{i \in V} \bar{x}_i z_i - \sum_{i \in V} y_i && (4.4)\\
\text{s.t.} \quad & \sum_{i \in V} y_i \leq k && (4.5)\\
& y_i \leq z_j, \quad \forall i \in V, \forall j \in \Delta(i) && (4.6)\\
& y_i, z_i \in \{0,1\}, \quad \forall i \in V && (4.7)
\end{aligned}
$$

The following theorem states the relation between (Separation-IP) and FIND-ATTACK-GIVEN-DEFENSE:

Theorem 4.4.1. *Let $G = (V, E)$ be a graph. Given $D \subseteq V$ and the corresponding incidence vector $\bar{x}$ of D, there exists an inequality of the form (4.2) that cuts off $\bar{x}$ if and only if (Separation-IP) has a negative optimal objective function value.*

Proof: Let $G = (V, E)$ be a graph and $D \subseteq V$ be the input to (Separation-IP) with the incidence vector $\bar{x}$.

Assume there exists an inequality of the form (4.2) written for $A \subseteq V, |A| \leq k$ that cuts off $\bar{x}$. That is, $\sum_{i \in \Delta(A)} \bar{x} < |A|$. We will construct a solution (y^*, z^*) to (Separation-IP) with a negative objective function value, which will imply that the optimal objective function value is also negative. Choose $y_i^* = 1 \quad \forall i \in A$ and $z_j^* = 1$ for all vertices j for which there exist a vertex i in its closed neighborhood with $y_i^* = 1$. This satisfies all the constraints of (Separation-IP). Furthermore, $\sum_{i \in V} \bar{x}_i z_i^* = \sum_{i \in \Delta(A)} \bar{x}_i z_i^* = \sum_{i \in \Delta(A)} \bar{x}_i < |A|$. Because $\sum_{i \in V} y_i^* = |A|$, we have that $\sum_{i \in V} \bar{x}_i z_i - \sum_{i \in V} y_i < 0$.

Now, we prove the converse. Assume there exists an optimal solution (y^*, z^*) with negative objective function value for (Separation-LP). Therefore, $\sum_{i \in V} \bar{x}_i z_i^* - \sum_{i \in V} y_i^* < 0$. Let $U = \{i \in V \mid y_i^* = 1\}$. We will show that the inequality (4.2) written for U cuts off $\bar{x}$, that is $\sum_{i \in \Delta(U)} \bar{x}_i < |U|$. We have that $\sum_{i \in V} y_i^* = \sum_{i \in U} y_i^* = |U|$,

and because (y^*, z^*) satisfies Constraint (4.5), $|U| \leq k$. Furthermore, as (y^*, z^*) satisfies Constraint (4.6), we have that $\sum_{i \in V} \bar{x}_i z_i^* \geq \sum_{i \in \Delta(U)} \bar{x}_i z_i^* = \sum_{i \in \Delta(U)} \bar{x}_i$. Finally, we conclude that $0 > \sum_{i \in V} \bar{x}_i z_i^* - \sum_{i \in V} y_i^* \geq \sum_{i \in \Delta(U)} \bar{x}_i - |U|$, and hence, $\sum_{i \in \Delta(U)} \bar{x}_i < |U|$. □

Using (Separation-IP), our row generation procedure is as follows. We start with an incomplete formulation of (IP-DEF-DOM), that lacks some (possibly all) constraints of type (4.2) within a branch-and-bound framework. At each node of the tree, whenever an integral feasible solution is found, we invoke (Separation - IP) to find the violated constraints of type (4.2). This process ensures that we have an optimal solution that is feasible for (IP-DEF-DOM).

4.5 Valid Inequalities

In this section, we propose some valid inequalities to strengthen (IP-DEF-DOM). Adding valid inequalities to an IP formulation makes it stronger and may lead to a reduction in solution times. Within the row generation framework, including valid inequalities initially may decrease the number of constraints of type (4.2) added during the separation process.

For each proposed class of inequalities, a proof of validity as well as a fractional point of the LP relaxation that is cut off is provided. This fractional point is not cut off by any of the previous inequalities which shows that the new class of inequalities is not dominated by the previous ones.

Proposition 4.5.1. *Let $G = (V, E)$ be a graph and $w \in V$ be a cut vertex. Upon the removal of w, let G split into $m > 1$ connected components with vertex sets $V_1, V_2, \ldots, V_m$. Define $S \subseteq \{1, 2, \ldots, m\}$ to be a maximal subset satisfying the following condition:*

$$|V_i| + |V_j| \leq k \quad \forall i, j \in S,\ i \neq j.$$

Then, the following inequality is valid:

$$x_w + \sum_{v \in \bigcup_{i \in S} V_i} x_v \geq \sum_{i \in S} |V_i| \tag{4.8}$$

Proof: Let D be a k-defensive set of G. For the sake of contradiction, assume that the inequality does not hold. That means there exists two vertices $u, v \in \{w\} \cup \left(\bigcup_{i \in S} V_i\right)$ such that $u, v \notin D$.

At least one of u, v, without loss of generality say u, is in V_i for some $i \in S$. If v is also in V_i or $v = w$, the attack on V_i cannot be defended because $|\Delta(V_i) \cap D| < |V_i|$. Otherwise, $v \in V_j$ for some $j \in S, i \neq j$. This time, the attack on V_i and V_j together cannot be defended because $|\Delta(V_i \cup V_j) \cap D| < |V_i| + |V_j|$ (It should be noted that for any $i, j \in S$, it is possible to attack V_i and V_j simultaneously as they satisfy $|V_i| + |V_j| \leq k$).

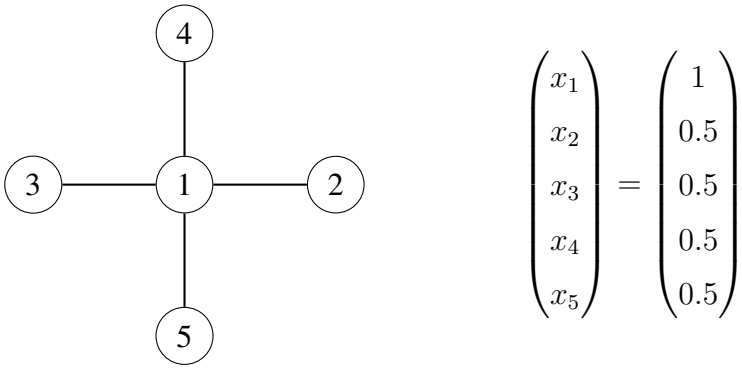


Figure 4.2: Example fractional solution that is cut off by inequality (4.8)

In Figure 4.2, let $k = 2$. We write inequality 4.8 for the cut vertex 1. Each connected component satisfies the condition in Proposition 4.5.1, and the inequality becomes $x_1 + x_2 + x_3 + x_4 + x_5 \geq 4$ and cuts the solution off.

Proposition 4.5.2. *Let $G = (V, E)$ be a graph. Let $B = \{v \in V \mid |V \setminus \Delta(v)| \geq k\}$. If $V \setminus B$ is not a k-defensive set, then the following inequality is valid:*

$$\sum_{v \in V} x_v \geq k + 1 \tag{4.9}$$

Proof: Assume $V \setminus B$ is not a k-defensive set. Then, for any k-defensive set $D \subseteq V$, there exists a vertex $v \in B$ such that $v \in D$. As $v \in B$, $|V \setminus \Delta(v)| \geq k$. Let

$w_1, w_2, \ldots, w_k \in V \setminus \Delta(v)$. By Theorem 4.3.1, because D is a k-defensive set, we have that $\Delta(\{w_1, w_2, \ldots, w_k\}) \cap D \geq k$. Furthermore, as $v \notin \Delta(\{w_1, w_2, \ldots, w_k\})$, we conclude that $\sum_{v \in V} x_v \geq k + 1$. □

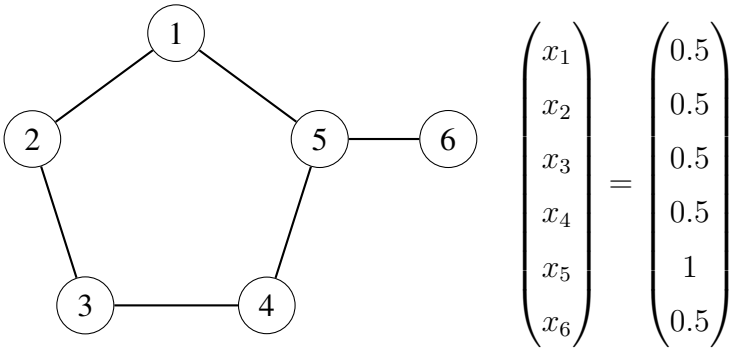


Figure 4.3: Example fractional solution that is cut off by inequality (4.9)

In the graph in Figure 4.3, let $k = 3$. We have that $B = \{1, 2, 3, 4, 6\}$. Since $V \setminus B$ is not a k-defensive set, inequality (4.9), which is $x_1 + x_2 + x_3 + x_4 + x_5 + x_6 \geq 4$ is valid and cuts the given fractional solution off.

The next set of valid inequalities we propose originates from two inequalities of the form $\sum_{i \in V'} x_i \geq |V'|$, for some $V' \subseteq V$, each of which is valid individually. We show that we can sum them and reduce the coefficient of some variables in the summed inequality, provided some additional conditions are satisfied.

Proposition 4.5.3. *Let $G = (V, E)$ be a graph, and $V_1, V_2 \subseteq V$ such that $\Delta(V_1) \cap \Delta(V_2) = \{z\}$ for some $z \in V$ and assume the inequalities $\sum_{i \in \Delta(V_1)} x_i \geq |V_1|$ and $\sum_{i \in \Delta(V_2)} x_i \geq |V_2|$ are both valid. Consider the induced subgraphs $G[\Delta(V_1) \setminus \{z\}]$ and $G[\Delta(V_2) \setminus \{z\}]$. Let γ_1 (respectively γ_2) denote the number of vertices in the connected component of $G[\Delta(V_1) \setminus \{z\}]$ (respectively $G[\Delta(V_2) \setminus \{z\}]$) that contains the largest number of vertices from V_1 (respectively V_2). If $k \geq \gamma_1 + \gamma_2$, the following inequality is valid:*

$$\sum_{i \in \Delta(V_1) \cup \Delta(V_2)} x_i \geq |V_1| + |V_2| \tag{4.10}$$

Proof:

Let $G = (V, E)$ be a graph and $D \subseteq V$ be a k-defensive set of G. By the assumption

that $\sum_{i\in\Delta(V_1)} x_i \geq |V_1|$ and $\sum_{i\in\Delta(V_2)} x_i \geq |V_2|$ are valid, we have that $|\Delta(V_1)\cap D| \geq |V_1|$ and $|\Delta(V_2)\cap D| \geq |V_2|$.

If $z \notin D$, then $(\Delta(V_1)\cap D)\cap(\Delta(V_2)\cap D) = \emptyset$, and therefore $\sum_{i\in\Delta(V_2)} x_i + \sum_{i\in\Delta(V_1)} x_i \geq |V_1| + |V_2|$ and inequality (4.10) is valid.

Assume $z \in D$. Then, $|(\Delta(V_1)\cap D)\setminus\{z\}| \geq |V_1| - 1$ and similarly, $|(\Delta(V_2)\cap D)\setminus\{z\}| \geq |V_2| - 1$. Furthermore, $(\Delta(V_2)\setminus\{z\})\cap(\Delta(V_1)\setminus\{z\}) = \emptyset$, therefore $\sum_{i\in\Delta(V_1)\cup\Delta(V_2)} x_i \geq |V_1| + |V_2| - 1$. To show that inequality (4.10) is valid, we will use proof by contradiction. Assume $|(\Delta(V_1)\cap D)\setminus\{z\}| = |V_1| - 1$ and $|(\Delta(V_2)\cap D)\setminus\{z\}| = |V_2| - 1$.

Let $p \geq 1$ be the number of connected components of the induced subgraph $G[\Delta(V_1)\setminus\{z\}]$. Label their vertex-sets by $X_1, X_2, \ldots, X_p$. For each $i = 1, 2, \ldots, p$, define $\alpha(X_i)$ to be the number of vertices from V_1 in the component X_i, that is $\alpha(X_i) = |X_i\cap V_1|$. Similarly, let $r \geq 1$ be the number of connected components of the induced subgraph $G[\Delta(V_2)\setminus\{z\}]$. Label their vertex-sets by $Y_1, Y_2, \ldots, Y_r$. For each $i = 1, 2, \ldots, r$, define $\beta(Y_i) = |Y_i\cap V_2|$, i.e., $\beta(Y_i)$ is the number of vertices from V_2 in the component Y_i.

By the pigeonhole principle, there must exist a connected component in $G[\Delta(V_1)\setminus\{z\}]$, without loss of generality call it X_1, such that $|(\Delta(X_1)\setminus\{z\})\cap D| < \alpha(X_1)$. We will attack all the vertices in $X_1\cap V_1$, and $|(\Delta(X_1)\setminus\{z\})\cap D| < \alpha(X_1)$ implies that z will be forced to defend a vertex in $X_1\cap V_1$. Similarly, there must exists a connected component in connected component in $G[\Delta(V_2)\setminus\{z\}]$, without loss of generality call it Y_1, such that $|(\Delta(Y_1)\setminus\{z\})\cap D| < \beta(Y_1)$. Attacking X_1 and Y_1 simultaneously (we have enough attackers for such an attack because of the assumption that $k \geq \gamma_1+\gamma_2 \geq \alpha(X_1) + \beta(Y_1)$) creates an attack that cannot be defended, and we conclude that the inequality $\sum_{i\in\Delta(V_1)\cup\Delta(V_2)} x_i \geq |V_1| + |V_2|$ is valid. □

In the graph in Figure 4.4, let $k = 3$. Using the same notation from Proposition 4.5.3, we choose $V_1 = \{4, 7\}$ and $V_2 = \{5, 6\}$, and hence $\Delta(V_1)\cap\Delta(V_2) = \{1\}$. Checking the induced subgraphs $G[\Delta(V_1)\setminus\{1\}]$ and $G[\Delta(V_2)\setminus\{1\}]$, we have that $\gamma_1 = 1$ and $\gamma_2 = 2$. Inequality (4.10) written for V_1 and V_2 becomes $x_1+x_3+x_4+x_5+x_6+x_7 \geq 4$ and cuts the solution off.

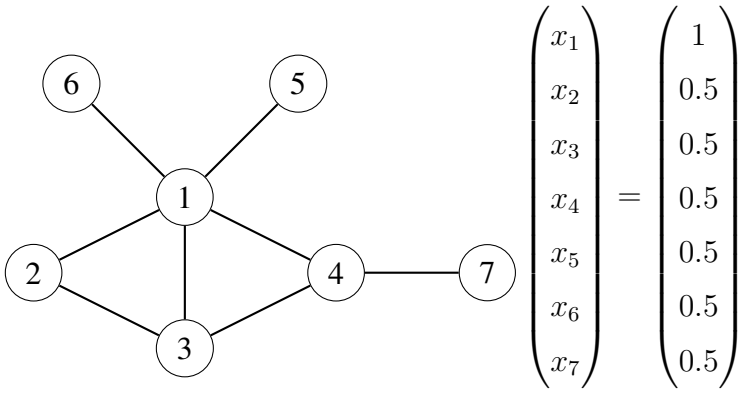


Figure 4.4: Example fractional solution that is cut off by inequality (4.10)

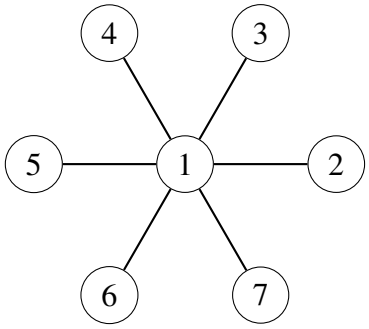


Figure 4.5: Star Graph S_7

Remark: The procedure in Proposition 4.5.3 can be used repeatedly to aggregate inequalities. For example, consider S_7, the star graph of order 7, drawn in Figure 4.5. Let $k = 2$. Using the same notation from Proposition 4.5.3, choose $V_1 = \{2, 3\}$ and $V_2 = \{4, 5\}$. The inequalities $\sum_{i\in\Delta(V_1)} x_i \geq |V_1|$ and $\sum_{i\in\Delta(V_2)} x_i \geq |V_2|$ are valid because they are the same as inequalities (4.2) written for V_1 and V_2. $\Delta(V_1)\cap\Delta(V_2) = \{1\}$, and $\gamma_1 = \gamma_2 = 1$, therefore the inequality $\sum_{i\in\Delta(V_1)\cup\Delta(V_2)} x_i \geq |V_1|+|V_2|$ is valid and stronger than the sum of $\sum_{i\in\Delta(V_1)} x_i \geq |V_1|$ and $\sum_{i\in\Delta(V_2)} x_i \geq |V_2|$.

Now we will use Proposition 4.5.3 again, by choosing $V_1 \cup V_2$ as the first vertex set, and $V_3 = \{6, 7\}$ as the second vertex set. This time, $\sum_{i\in\Delta(V_1)\cup\Delta(V_2)} x_i \geq |V_1| + |V_2|$ comes from the initial use of Proposition 4.5.3 on V_1 and V_2, and $\sum_{i\in\Delta(V_3)} x_i \geq |V_3|$ is inequality (4.2) written for V_3. After checking the necessary conditions, the inequality becomes $\sum_{i\in\Delta(V_1)\cup\Delta(V_2)\cup\Delta(V_3)} x_i \geq |V_1| + |V_2| + |V_3|$. It should be pointed out that this inequality cannot be written directly by using Proposition 4.5.3, but can be written

after using it recursively.

4.6 Computational Studies

In this section, we test the performance of our IP formulation and the row generation procedure on random graphs.

Hardware and Software used: All the experiments were conducted on a machine equipped with an Intel Xeon E-2246G CPU running at 3.60 GHz and 16 GB of RAM. We use CPLEX 22.1.1 to solve the integer programs with presolve enabled.

In our preliminary experiments, we have tried to use (IP-DEF-DOM) directly, that is by initially generating all the inequalities in the formulation. Due to the exponential number of inequalities in k, the formulation gets impractical quickly with the increase of k, even for relatively small graphs. In Table 4.1, for each instance size, where instance size is specified by the number of vertices n, we provide the smallest k value for which it is not possible to list all inequalities within a time limit of 1 hour. This smallest k value is the same across all densities, so a single k value is reported per instance size.

Table 4.1: Smallest k for which not all inequalities can be listed within 1 hour for different number of vertices n

n	Smallest k
10	-
20	-
30	10
40	10
50	5
75	4
100	4
150	4
200	3

Next, we test the row generation approach. (IP-DEF-DOM) is implemented as an integer program in CPLEX 22.1.1, and we implement (Separation - IP) as a lazy constraint callback. We investigate the solution times of MW-K-DEF-DOM instances on Erdős–Rényi graphs with varying order (n) and density (p) values. We generate 5 Erdős–Rényi graphs for each (n, p) pair. For each instance, we apply our row generation algorithm across all considered k values. Initially, we add all of Constraints (4.2) for $k = 1$ and $k = 2$, as well as all the valid inequalities except (4.10). We exclude inequalities (4.10) due to the large number of such inequalities and the absence of a fast separation algorithm. The lazy constraint callback is not used for instances where $k = 2$, because all the constraints are initially present from (IP-DEF-DOM). We consider two cases: a weighted version in which each vertex has a random weight drawn uniformly from $[0, 1]$, and a cardinality version where all vertices have unit weight. Negative weights are not considered because any vertex with a negative weight will always be included in every k-defensive set.

We set a time limit of 3600 seconds for each instance, and for every (n, p, k) combination, we report the number of instances in which an optimal solution was obtained within the time limit under the column "#Opt", the average solution time over the instances where an optimal solution is obtained under the column "T(s)", the average number of times the callback was invoked under the column "#Cb", and in column "TInCb(s)", we report the average total time spent generating inequalities in the callback.

The results for the unweighted and weighted versions can be found in Tables 4.2 and 4.3, respectively.

There are a lot of instances that are solved to optimality using the row generation procedure for which it is not possible to use (IP-DEF-DOM) directly by initially generating all inequalities. This clearly shows the effectiveness of the row generation procedure.

In the unweighted version, all the instances of order up to 40 were solved successfully within the time limit. Across most n, k pairs, the solution times and the number of optimally solved instances do not change linearly. Increasing p from 0.1 to 0.25 increases the solution times and decreases the number of optimally solved instances.

However, increasing p to 0.5 and then to 0.75 reduces the solution times and increases the number of optimally solved instances. For example, consider $n = 75$ and $k = 4$. For $p = 0.1$, four of the instances are solved to optimality with an average time of 2070 seconds, but for $p = 0.25$, none of the instances are solved to optimality. However, for $p = 0.5$ and $p = 0.75$, we are able to obtain optimal solutions to all the instances, with a better performance in $p = 0.75$ compared to $p = 0.5$. Increasing k tends to make the problem harder, but especially on dense instances, the problem with $k = 10$ is relatively easier. Consider $n = 150$ and $p = 0.75$. Each increase of k from 2 to 4 increases the solution time and decreases the number of optimally solved instances, and for $k = 4$ and $k = 5$, none of the instances are solved to optimality. However, after increasing k from 5 to 10, we are able to solve all the instances to optimality, with an average of only 80 seconds. One possible explanation is that, for dense graphs, most of the inequalities of type (4.2) are implied by others, because the closed neighborhood of 10 vertices is often the vertex set itself, which may not be the case for only 5 vertices.

In the weighted version, we perform much better compared to the unweighted one. One explanation of this could be the inherent symmetry of the problem; there exists many k-defensive sets of minimum cardinality which makes it harder to fathom nodes in the branch and bound tree in the unweighted case. When the vertices have weights, this symmetry is broken and the performance improves. All the instances of order up to 100 were solved successfully within the time limit. For $n = 150$ and $n = 200$, only instances that have density $p = 0.1$ were not able to be solved. Our formulation was effective at solving all instances with density other than 0.1. The solution times as well as the number of times the callback is invoked decreases as the graph density increases. For example, when $n = 150$ and $k = 4$, each increase in density reduced the solution time and the number of times the callback is invoked by at least one third. As expected, increasing the number of vertices in the instances leads to higher solution times and more callbacks are invoked. Finally, when k is increased, there is a big increase in difficulty, especially for sparse instances; but for dense instances solution times show no significant variation. For example, for $n = 200$, in less dense instances the solution times greatly increase with the increase in k, and in the case of $p = 0.1$, the number of instances solved to optimality decreases each time k is

increased. However, for $p = 0.8$, the solution times remain constant. In general, the formulation performs worst at the density $p = 0.1$ for the weighted version.

For both the weighted and unweighted versions, the row generation procedure has a similar performance as the subproblem of finding a violated inequality is independent of the vertex weights. When n or k increases, there are more inequalities in (IP-DEF-DOM) and therefore the lazy constraint callback is usually invoked more times on the average. The exception to this is for instances with $p = 0.75$ and $k = 10$, for which only a few inequalities need to be generated. When n increases, time is spent per callback as well as total time spent in the callback is increases. For a (n, p, k) combination where not all instances are solved to optimality, optimally solved instances generally have slightly fewer number of rows generated compared to the non-optimal instances. Although not explicitly reported in the tables, for all (n, p, k) combinations, more than 99% of the time, invoking the callback results in a successful addition of an inequality.

Table 4.2: Unweighted Version: Computation statistics for various k values across (n, p) pairs

		$k=2$		$k=3$				$k=4$				$k=5$				$k=10$			
n	p	#Opt	T(s)	#Opt	T(s)	#Cb	TInCb(s)	#Opt	T(s)	#Cb	TInCb(s)	#Opt	T(s)	#Cb	TInCb(s)	#Opt	T(s)	#Cb	TInCb(s)
10	0.10	5	0.04	5	0.04	3.00	0.04	5	0.05	3.80	0.05	5	0.06	3.60	0.05	5	0.03	2.00	0.02
	0.25	5	0.04	5	0.09	5.20	0.08	5	0.13	8.60	0.13	5	0.09	6.00	0.09	5	0.03	2.00	0.02
	0.50	5	0.06	5	0.04	2.60	0.04	5	0.12	5.80	0.11	5	0.07	2.60	0.06	5	0.03	2.00	0.02
	0.75	5	0.02	5	0.05	2.80	0.04	5	0.04	2.60	0.04	5	0.05	2.40	0.05	5	0.03	2.00	0.03
20	0.10	5	0.04	5	0.15	8.80	0.14	5	0.25	15.80	0.24	5	0.23	14.00	0.22	5	0.32	15.00	0.31
	0.25	5	0.06	5	0.59	28.60	0.56	5	0.64	32.00	0.62	5	1.29	35.40	1.26	5	0.71	20.40	0.70
	0.50	5	0.06	5	0.55	25.20	0.54	5	0.59	23.40	0.58	5	0.73	23.00	0.72	5	0.26	5.80	0.25
	0.75	5	0.07	5	0.24	9.40	0.23	5	0.44	16.00	0.43	5	0.41	14.20	0.41	5	0.09	2.20	0.08
30	0.10	5	0.05	5	0.42	21.40	0.42	5	1.02	48.80	1.02	5	1.13	43.80	1.10	5	3.43	78.40	3.39
	0.25	5	0.13	5	1.86	67.80	1.74	5	3.16	115.20	3.01	5	5.45	129.00	5.27	5	3.09	74.00	3.00
	0.50	5	0.10	5	0.68	20.80	0.64	5	1.72	49.80	1.67	5	2.94	66.80	2.86	5	1.17	20.60	1.15
	0.75	5	0.09	5	0.09	1.80	0.07	5	0.33	7.00	0.31	5	0.44	8.60	0.43	5	0.43	6.20	0.41
40	0.10	5	0.11	5	1.59	62.60	1.49	5	2.88	111.00	2.77	5	8.60	183.40	8.29	5	18.59	291.20	17.51
	0.25	5	0.41	5	5.63	140.00	4.69	5	21.69	514.00	18.33	5	41.64	637.40	34.58	5	73.26	742.60	37.07
	0.50	5	0.42	5	5.52	115.80	5.16	5	16.58	333.80	15.88	5	7.92	125.80	7.63	5	8.28	116.80	8.08
	0.75	5	0.17	5	0.53	8.40	0.49	5	0.26	3.80	0.24	5	0.25	3.20	0.22	5	1.60	15.20	1.55
50	0.10	5	0.26	5	6.82	185.20	6.12	5	16.94	446.80	13.91	5	74.99	826.80	63.70	5	126.62	1466.20	80.75
	0.25	5	1.51	5	35.72	375.00	16.31	5	150.70	1232.40	53.39	5	600.83	2513.00	192.44	4	1308.16	2406.00*	148.85
	0.50	5	0.79	5	26.63	361.20	22.31	5	51.94	719.80	44.34	5	92.30	974.20	82.98	5	18.71	193.40	17.09
	0.75	5	0.26	5	1.34	15.40	1.29	5	0.93	10.80	0.88	5	0.66	5.80	0.63	5	4.74	34.20	4.58
75	0.10	5	5.32	5	498.05	881.80	46.14	4	2070.24	3268.20†	154.98	1	896.88	4924.60‡	477.62	0	-	9109.60	567.51
	0.25	5	89.45	5	2047.12	1518.40	130.44	0	-	4864.60	575.56	0	-	5101.20	1126.71	0	-	8587.00	1653.82
	0.50	5	8.90	5	123.42	390.40	45.48	5	335.12	1594.00	180.49	5	841.53	2535.60	399.98	5	221.95	1353.40	183.81
	0.75	5	0.82	5	24.57	150.40	23.30	5	10.44	63.60	9.93	5	7.04	32.40	6.67	5	0.33	1.20	0.22
100	0.10	5	361.34	0	-	2595.00	199.33	0	-	6281.00	418.99	0	-	6598.40	850.46	0	-	9706.80	842.81
	0.25	5	1811.41	0	-	1625.80	463.09	0	-	3937.20	1157.20	0	-	5635.60	1871.50	0	-	5978.40	2672.01
	0.50	5	42.62	5	1356.75	2752.60	527.09	1	1883.35	5512.80§	1869.47	0	-	5789.00	2072.74	0	-	4126.00	1843.83
	0.75	5	5.80	5	297.97	972.00	253.02	5	231.32	792.00	207.23	5	76.39	232.80	71.47	5	1.59	4.40	1.30
150	0.10	0	-	0	-	1387.20	333.22	0	-	2784.00	877.96	0	-	3291.60	895.83	0	-	5513.20	968.44
	0.25	0	-	0	-	719.20	318.92	0	-	2276.00	988.80	0	-	3074.40	1611.09	0	-	4031.40	2638.91
	0.50	1	3580.99	0	-	736.60	388.16	0	-	3161.80	1584.57	0	-	4205.40	2095.91	0	-	4842.40	2796.46
	0.75	5	74.33	5	2425.04	2499.40	1393.47	0	-	4456.00	2501.87	0	-	4368.00	2590.09	5	80.63	117.00	73.66
200	0.10	0	-	0	-	548.80	152.61	0	-	972.20	282.33	0	-	1317.80	458.20	0	-	2249.80	1403.05
	0.25	0	-	0	-	326.80	141.50	0	-	1349.00	677.22	1	3565.40	1735.40	993.20	0	-	2371.40	2174.52
	0.5	0	-	0	-	981.20	752.55	0	-	1952.00	1553.01	0	-	3287.60	2446.84	0	-	3144.40	2849.04
	0.75	5	727.01	0	-	1334.20	1306.02	0	-	2521.60	2497.27	0	-	2661.80	2665.67	2	2970.48	2294.00¶	2567.64

As a footnote, we give the number of callbacks invoked averaged over only optimally solved instances, for (n, p, k) combinations where not all instances are solved to optimality.

*1844.75 when averaged over the instances solved to optimality

†1673 when averaged over the instances solved to optimality

‡3001.75 when averaged over the instances solved to optimality

§2370 when averaged over the instances solved to optimality

¶2054.5 when averaged over the instances solved to optimality

Table 4.3: Weighted Version: Computation statistics for various k values across (n, p) pairs

		$k=2$		$k=3$				$k=4$				$k=5$				$k=10$			
n	p	#Opt	T(s)	#Opt	T(s)	#Cb	TInCb(s)	#Opt	T(s)	#Cb	TInCb(s)	#Opt	T(s)	#Cb	TInCb(s)	#Opt	T(s)	#Cb	TInCb(s)
10	0.10	5	0.05	5	0.04	2.60	0.04	5	0.05	2.80	0.04	5	0.05	2.20	0.05	5	0.03	2.00	0.03
	0.25	5	0.05	5	0.06	4.20	0.06	5	0.08	5.40	0.08	5	0.08	5.00	0.07	5	0.03	2.00	0.02
	0.50	5	0.02	5	0.03	1.60	0.02	5	0.05	3.00	0.05	5	0.04	2.00	0.03	5	0.04	2.00	0.04
	0.75	5	0.02	5	0.06	3.20	0.05	5	0.04	2.00	0.03	5	0.05	2.00	0.04	5	0.03	2.00	0.03
20	0.10	5	0.03	5	0.09	5.20	0.08	5	0.16	9.40	0.15	5	0.22	11.80	0.21	5	0.20	8.60	0.19
	0.25	5	0.05	5	0.18	8.40	0.17	5	0.25	11.60	0.24	5	0.39	14.80	0.39	5	0.24	7.20	0.24
	0.50	5	0.04	5	0.26	10.00	0.26	5	0.21	8.60	0.20	5	0.25	7.40	0.24	5	0.11	3.00	0.11
	0.75	5	0.04	5	0.07	2.60	0.06	5	0.10	3.80	0.10	5	0.11	3.60	0.10	5	0.10	2.00	0.09
30	0.10	5	0.05	5	0.25	13.00	0.22	5	0.72	33.20	0.70	5	0.93	37.60	0.90	5	1.61	38.20	1.58
	0.25	5	0.11	5	0.81	27.60	0.77	5	1.09	39.40	1.03	5	1.74	35.40	1.70	5	0.55	12.80	0.53
	0.50	5	0.07	5	0.42	11.80	0.40	5	0.48	13.20	0.45	5	0.42	8.60	0.40	5	0.37	7.60	0.35
	0.75	5	0.08	5	0.10	2.20	0.09	5	0.07	1.40	0.05	5	0.12	2.20	0.10	5	0.23	3.40	0.21
40	0.10	5	0.16	5	0.87	32.40	0.84	5	1.96	77.80	1.89	5	3.75	93.40	3.66	5	5.14	93.80	5.04
	0.25	5	0.18	5	1.33	32.00	1.20	5	2.34	62.00	2.16	5	3.72	55.00	3.50	5	2.11	41.60	2.06
	0.50	5	0.11	5	0.57	11.20	0.52	5	0.59	11.60	0.54	5	0.72	10.80	0.68	5	0.41	5.80	0.38
	0.75	5	0.16	5	0.25	3.80	0.23	5	0.14	2.00	0.12	5	0.16	1.60	0.14	5	0.43	4.40	0.40
50	0.10	5	0.12	5	2.01	52.40	1.88	5	3.62	108.40	3.44	5	12.39	160.20	12.02	5	15.11	252.80	14.34
	0.25	5	0.41	5	1.83	33.80	1.56	5	3.56	75.80	3.18	5	5.94	68.00	5.62	5	2.96	48.20	2.83
	0.50	5	0.20	5	1.07	14.80	0.99	5	1.18	18.60	1.10	5	1.64	17.40	1.56	5	0.50	6.00	0.46
	0.75	5	0.14	5	0.86	9.60	0.82	5	0.41	4.60	0.38	5	0.43	3.80	0.39	5	0.41	3.60	0.38
75	0.10	5	0.75	5	9.67	136.00	8.25	5	29.45	430.20	20.29	5	85.70	666.00	63.43	5	125.66	850.00	52.80
	0.25	5	0.70	5	6.31	64.40	5.61	5	9.99	118.40	8.87	5	19.10	155.00	17.66	5	9.41	98.00	8.95
	0.50	5	0.61	5	3.19	23.40	2.89	5	3.64	29.20	3.40	5	3.48	20.40	3.26	5	1.72	11.40	1.57
	0.75	5	0.45	5	1.49	8.00	1.34	5	1.20	6.80	1.06	5	1.49	6.40	1.35	5	0.42	1.80	0.30
100	0.10	5	1.25	5	18.82	176.20	14.42	5	47.95	489.00	32.65	5	110.56	613.80	72.66	5	502.98	1520.20	143.18
	0.25	5	0.98	5	12.79	81.80	10.97	5	19.83	142.60	16.83	5	34.42	181.80	30.48	5	29.10	167.60	27.14
	0.50	5	1.11	5	4.79	21.20	4.22	5	7.14	34.40	6.55	5	7.21	28.80	6.71	5	2.49	9.60	2.18
	0.75	5	0.69	5	2.95	9.80	2.57	5	2.38	7.80	2.02	5	3.14	9.20	2.80	5	1.65	4.60	1.37
150	0.10	5	7.04	5	132.95	291.40	41.08	5	840.61	955.40	120.22	4	1591.06	1683.40 [‖]	234.07	1	1116.81	3171.20 [**]	768.47
	0.25	5	3.66	5	48.25	135.20	33.70	5	96.31	294.60	68.58	5	98.08	280.00	74.57	5	72.05	179.40	62.93
	0.50	5	3.62	5	17.74	36.40	14.80	5	22.63	50.00	19.90	5	19.09	39.40	17.04	5	7.69	13.20	6.44
	0.75	5	2.16	5	10.94	16.80	9.49	5	8.12	12.20	6.79	5	5.06	6.80	3.98	5	5.38	6.80	4.26
200	0.10	5	76.55	4	441.31	400.60	85.18	3	1017.67	1463.80 [††]	308.23	2	2128.25	2195.60 [‡‡]	501.22	0	-	3719.20	1870.61
	0.25	5	7.71	5	65.77	117.00	45.94	5	112.71	205.40	79.31	5	176.98	310.40	128.46	5	215.14	292.40	188.63
	0.50	5	5.54	5	36.49	45.20	30.85	5	43.66	55.40	38.22	5	43.59	53.80	38.50	5	18.02	17.20	14.61
	0.75	5	3.77	5	10.95	8.40	8.16	5	14.39	11.60	11.29	5	11.77	8.80	8.75	5	6.94	4.20	4.56

As a footnote, we give the number of callbacks invoked averaged over only optimally solved instances, for (n, p, k) combinations where not all instances are solved to optimality.

[‖] 1308 when averaged over the instances solved to optimality

[**] 961 when averaged over the instances solved to optimality

[††] 710.5 when averaged over the instances solved to optimality

[‡‡] 1199.5 when averaged over the instances solved to optimality

4.7 Conclusion

In this thesis, we study defensive domination with vertex weights for the first time. We propose an integer programming formulation for the MW-K-DEF-DOM problem. Due to the large number of constraints in this formulation, we develop a row generation procedure, by considering the separation problem of the inequalities in our formulation. We propose several sets of valid inequalities to strengthen the formulation, and test the row generation procedure on randomly generated graphs.

As future work, developing a way to incorporate inequalities (4.10) into the formulation to strengthen the formulation could be promising. Additionally, proposing new valid inequalities, possibly restricted to certain graph classes, could be interesting as well.

REFERENCES

[1] C. Berge. *Theory of Graphs and its Applications*, volume 2 of *Collection Universitaire de Mathématiques*. Dunod, Paris, 1958.

[2] M. Bouchakour and A. R. Mahjoub. One-node cutsets and the dominating set polytope. *Discrete Mathematics*, 165-166:101–123, 1997.

[3] L. Chen, C. Lu, and Z. Zeng. Hardness results and approximation algorithms for (weighted) paired-domination in graphs. *Theoretical Computer Science*, 410(47):5063–5071, 2009.

[4] L. Chen, C. Lu, and Z. Zeng. Labelling algorithms for paired-domination problems in block and interval graphs. *Journal of Combinatorial Optimization*, 19(4):457–470, 2010.

[5] T. Cheng, L. Kang, and C. Ng. Paired domination on interval and circular-arc graphs. *Discrete Applied Mathematics*, 155(16):2077–2086, 2007.

[6] E. J. Cockayne, P. A. Dreyer, S. M. Hedetniemi, and S. T. Hedetniemi. Roman domination in graphs. *Discrete Mathematics*, 278(1):11–22, 2004.

[7] D. Dereniowski, T. Gavenčiak, and J. Kratochvíl. Cops, a fast robber and defensive domination on interval graphs. *Theoretical Computer Science*, 2019.

[8] W. J. Desormeaux, T. W. Haynes, and M. A. Henning. Paired domination in graphs. In *Topics in Domination in Graphs*, volume 64, pages 31–77. Springer Cham, 2020.

[9] J. E. Dunbar, D. J. Erwin, T. W. Haynes, S. M. Hedetniemi, and S. T. Hedetniemi. Broadcasts in graphs. *Discrete Applied Mathematics*, 154(1):59–75, 2006.

[10] T. Ekim, A. Farley, A. Proskurowski, and M. Shalom. Defensive domination in proper interval graphs. *Discrete Applied Mathematics*, 331:59–69, 05 2023.

[11] T. Ekim, A. M. Farley, and A. Proskurowski. The complexity of the defensive domination problem in special graph classes. *Discrete Mathematics*, 343(2):111665, 2020.

[12] D. J. Erwin. Dominating broadcasts in graphs. *Bulletin of the Institute of Combinatorics and its Applications*, 42:89–105, 2004.

[13] A. M. Farley and A. Proskurowski. Defensive domination. *Congressus Numerantium*, 168:97–107, 2004.

[14] O. Favaron and M. A. Henning. Paired-domination in claw-free cubic graphs. *Graphs and Combinatorics*, 20(4):447–456, 2004.

[15] M. R. Garey and D. S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W.H. Freeman and Company, New York, 1979.

[16] J. M. Harris, J. L. Hirst, and M. J. Mossinghoff. *Combinatorics and Graph Theory*. Undergraduate Texts in Mathematics. Springer, New York, NY, 2008.

[17] T. W. Haynes, S. T. Hedetniemi, and M. A. Henning. *Domination in Graphs: Core Concepts*. Springer Monographs in Mathematics. Springer, Cham, 2023.

[18] T. W. Haynes and P. J. Slater. Paired-domination in graphs. *Networks*, 32(3):199–206, 1998.

[19] M. A. Henning, A. Pandey, and V. Tripathi. Approximation algorithm and hardness results for defensive domination in graphs. In *International Workshop on Combinatorial Algorithms (IWOCA)*, volume 12931 of *Lecture Notes in Computer Science*, pages 261–273, 2021.

[20] M. A. Henning and M. D. Plummer. Vertices contained in all or in no minimum paired-dominating set of a tree. *Journal of Combinatorial Optimization*, 10(3):283–294, 2005.

[21] A. Letchford, G. Reinelt, and D. O. Theis. A faster exact separation algorithm for blossom inequalities. In *Lecture Notes in Computer Science*, volume 3064, pages 196–205, 2004.

[22] C.-C. Lin and H.-L. Tu. A linear-time algorithm for paired-domination on circular-arc graphs. *Theoretical Computer Science*, 591:99–105, 2015.

[23] C. L. Liu. *Introduction to Combinatorial Mathematics*. McGraw-Hill, New York, 1968.

[24] O. Ore. *Theory of Graphs*, volume 38 of *American Mathematical Society Publications*. American Mathematical Society, Providence, RI, 1962.

[25] M. W. Padberg and M. R. Rao. Odd minimum cut-sets and b-matchings. *Mathematics of Operations Research*, 7(1):67–80, 1982.

[26] D. Pradhan and B. S. Panda. Computing a minimum paired-domination set in strongly orderable graphs. *Discrete Applied Mathematics*, 253:37–50, 2019.

[27] H. Qiao, L. Kang, M. Cardei, and D. Du. Paired-domination of trees. *Journal of Global Optimization*, 25(1):43–54, 2003.