Universidade Estadual de Campinas
Instituto de Computação

Andrei de Almeida Sampaio Braga

An Eternal Domination Problem: Graph Classes, Solving Methods, and Practical Standpoint

Um Problema de Dominação Eterna: Classes de Grafos, Métodos de Resolução e Perspectiva Prática

CAMPINAS
2019

Andrei de Almeida Sampaio Braga

An Eternal Domination Problem: Graph Classes, Solving Methods, and Practical Standpoint

Um Problema de Dominação Eterna: Classes de Grafos, Métodos de Resolução e Perspectiva Prática

Tese apresentada ao Instituto de Computação da Universidade Estadual de Campinas como parte dos requisitos para a obtenção do título de Doutor em Ciência da Computação.

Dissertation presented to the Institute of Computing of the University of Campinas in partial fulfillment of the requirements for the degree of Doctor in Computer Science.

Supervisor/Orientador: Prof. Dr. Cid Carvalho de Souza
Co-supervisor/Coorientador: Prof. Dr. Orlando Lee

Este exemplar corresponde à versão final da Tese defendida por Andrei de Almeida Sampaio Braga e orientada pelo Prof. Dr. Cid Carvalho de Souza.

CAMPINAS
2019

Ficha catalográfica
Universidade Estadual de Campinas
Biblioteca do Instituto de Matemática, Estatística e Computação Científica
Silvania Renata de Jesus Ribeiro - CRB 8/6592

B73e

Braga, Andrei de Almeida Sampaio, 1986-
An eternal domination problem : graph classes, solving methods, and practical standpoint / Andrei de Almeida Sampaio Braga. – Campinas, SP : [s.n.], 2019.

Orientador: Cid Carvalho de Souza.
Coorientador: Orlando Lee.
Tese (doutorado) – Universidade Estadual de Campinas, Instituto de Computação.

1. Otimização combinatória. 2. Programação inteira. 3. Teoria dos grafos. 4. Algoritmos. 5. Conjunto dominante eterno. I. Souza, Cid Carvalho de, 1963-. II. Lee, Orlando, 1969-. III. Universidade Estadual de Campinas. Instituto de Computação. IV. Título.

Informações para Biblioteca Digital

Título em outro idioma: Um problema de dominação eterna : classes de grafos, métodos de resolução e perspectiva prática
Palavras-chave em inglês:
Combinatorial optimization
Integer programming
Graph theory
Algorithms
Eternal dominating set
Área de concentração: Ciência da Computação
Titulação: Doutor em Ciência da Computação
Banca examinadora:
Cid Carvalho de Souza [Orientador]
Daniel Morgato Martin
Mário César San Felice
Cláudio Leonardo Lucchesi
Eduardo Candido Xavier
Data de defesa: 05-12-2019
Programa de Pós-Graduação: Ciência da Computação

Identificação e informações acadêmicas do(a) aluno(a)
- ORCID do autor: https://orcid.org/0000-0001-6872-496X
- Currículo Lattes do autor: http://lattes.cnpq.br/0868229988425889

Universidade Estadual de Campinas
Instituto de Computação

Andrei de Almeida Sampaio Braga

An Eternal Domination Problem: Graph Classes, Solving Methods, and Practical Standpoint

Um Problema de Dominação Eterna: Classes de Grafos, Métodos de Resolução e Perspectiva Prática

Campinas, 05 de dezembro de 2019

Acknowledgements

I want to express deep gratitude to my advisors, Prof. Cid C. de Souza and Prof. Orlando Lee. I am sincerely thankful for their guidance, dedication, and patience throughout my doctoral work. I am especially grateful to Prof. Cid C. de Souza, from whom I have been learning since my master's course.

I also want to thank my friends and colleagues from the Institute of Computing of the University of Campinas, particularly the students of the Laboratory of Optimization and Combinatorics. I enjoyed their company during insightful discussions and relaxing conversations. I take the opportunity to acknowledge the commitment and support from the staff of the Institute of Computing's graduate program.

I truly thank the following funding agencies. This study was financed in part by the *Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - Brasil* (CAPES) - Finance Code 001. In addition, this study was partly funded by the *Conselho Nacional de Desenvolvimento Científico e Tecnológico - Brasil* (CNPq) - Grant 141964/2013-8.

I owe special thanks to Daniel Moraes, a friend that I met in class and with whom I have been debating work- and life-related issues since then. His partnership was decisive to several aspects of my development.

My most heartfelt thanks go to my wife, Alice. Without her continuous encouragement and support, I would not have reached my goals. We have been sharing love for our dachshund, Frida, an adorable source of pleasant and restless moments.

I cannot thank enough my family, notably my parents, José and Erika. Put simply, I am forever indebted to them for always looking after, advising, and aiding me.

Above all, I thank God for every step I took in this unique journey.

Resumo

O problema do conjunto dominante m-eterno é um problema de otimização em grafos que tem sido muito estudado nos últimos anos e para o qual se têm listado aplicações em vários domínios. O objetivo é determinar o número mínimo de guardas que consigam defender eternamente ataques nos vértices de um grafo; denominamos este número o *índice de dominação m-eterna* do grafo. Nesta tese, estudamos o problema do conjunto dominante m-eterno: lidamos com aspectos de natureza teórica e prática e abordamos o problema restrito a classes específicas de grafos e no caso geral.

Examinamos o problema do conjunto dominante m-eterno com respeito a duas classes de grafos: os grafos de Cayley e os conhecidos grafos de intervalo próprios. Primeiramente, mostramos ser inválido um resultado sobre os grafos de Cayley presente na literatura, provamos que o resultado é válido para uma subclasse destes grafos e apresentamos outros achados. Em segundo lugar, fazemos descobertas em relação aos grafos de intervalo próprios, incluindo que, para estes grafos, o índice de dominação m-eterna é igual à cardinalidade máxima de um conjunto independente e, por consequência, o índice de dominação m-eterna pode ser computado em tempo linear.

Tratamos de uma questão que é fundamental para aplicações práticas do problema do conjunto dominante m-eterno, mas que tem recebido relativamente pouca atenção. Para tanto, introduzimos dois métodos heurísticos, nos quais formulamos e resolvemos modelos de programação inteira e por restrições para computar limitantes ao índice de dominação m-eterna. Realizamos um vasto experimento para analisar o desempenho destes métodos. Neste processo, geramos um *benchmark* contendo 750 instâncias e efetuamos uma avaliação prática de limitantes ao índice de dominação m-eterna disponíveis na literatura.

Por fim, propomos e implementamos um algoritmo exato para o problema do conjunto dominante m-eterno e contribuímos para o entendimento da sua complexidade: provamos que a versão de decisão do problema é NP-difícil. Pelo que temos conhecimento, o algoritmo proposto foi o primeiro método exato a ser desenvolvido e implementado para o problema do conjunto dominante m-eterno.

Abstract

The m-eternal dominating set problem is a graph-protection optimization problem that has been an active research topic in the recent years and reported to have applications in various domains. It asks for the minimum number of guards that can eternally defend attacks on the vertices of a graph; this number is called the *m-eternal domination number* of the graph. In this thesis, we study the m-eternal dominating set problem by dealing with aspects of theoretical and practical nature and tackling the problem restricted to specific classes of graphs and in the general case.

We examine the m-eternal dominating set problem for two classes of graphs: Cayley graphs and the well-known proper interval graphs. First, we disprove a published result on the m-eternal domination number of Cayley graphs, show that the result is valid for a subclass of these graphs, and report further findings. Secondly, we present several discoveries regarding proper interval graphs, including that, for these graphs, the m-eternal domination number equals the maximum size of an independent set and, as a consequence, the m-eternal domination number can be computed in linear time.

We address an issue that is fundamental to practical applications of the m-eternal dominating set problem but that has received relatively little attention. To this end, we introduce two heuristic methods, in which we propose and solve integer and constraint programming models to compute bounds on the m-eternal domination number. By performing an extensive experiment to validate the features of these methods, we generate a 750-instance benchmark and carry out a practical evaluation of bounds for the m-eternal domination number available in the literature.

Finally, we propose and implement an exact algorithm for the m-eternal dominating set problem and contribute to the knowledge on its complexity: we prove that the decision version of the problem is NP-hard. As far as we know, the proposed algorithm was the first developed and implemented exact method for the m-eternal dominating set problem.

List of Figures

1.1 A graph satisfying $\gamma^{\infty}_{m,1} = 2q + 1 > q + 4 = \gamma^{\infty}_{m,m}$ for $q > 3$. 14
1.2 A Hasse diagram on a partially ordered set representation of bounds on the m-eternal domination number. 16

2.1 Compositions of symmetries for D_6. 23
2.2 The Cayley graph G_1 and the sets of vertices D_1 and hD_1. 23
2.3 The Cayley graph G_2 and the dominating sets D_2 and D_2h. 24
2.4 A Cayley graph G such that $\gamma(G) = 3$ and $\sigma_m(G) = 4$. 26

3.1 A graph G and a graph $G^+[A, S]$. 33
3.2 Placements of guards in C_1 and C_2 on a graph G and its subgraph $G[B \cup S]$. 36
3.3 Application of Theorem 3.3 for the proper interval graph of Figure 3.1(a). . 37
3.4 A proper interval graph G and the subgraph G_1. 38

4.1 A graph having a neocolonization of weight 3, a neocolonization setup of weight 4, and a 5,m-EDSC. 45
4.2 A graph having an efficient strategy of defense for 6 guards that is derived from a neocolonization setup. 47
4.3 A graph for which a larger dominating set may yield a less-weight neocolonization setup. 51
4.4 A graph having an 8-weight neocolonization setup that can be improved into a 7-weight one. 51
4.5 Number of instances per intervals of number of γ^{∞}_{m}-size dominating sets. . 55
4.6 Intervals of time to compute γ^{∞}_{m} per intervals of number of γ^{∞}_{m}-size dominating sets. 55

List of Tables

2.1 Labels of vertices of G. 25

4.1 Gap (H1-UB − H1-LB) per order intervals. 56
4.2 Gap (H1-UB − H1-LB) and H1-UB origin per density intervals. 56
4.3 Gap (H1-UB − H1-LB) per intervals of number of γ_m^∞-size dominating sets. 56
4.4 Gap (H1-UB − H1-LB) per intervals of time to compute γ_m^∞. 56
4.5 Gap (H2-UB − H2-LB) per order intervals. 68
4.6 Gap (H2-UB − H2-LB) and H2-UB origin per density intervals. 68
4.7 Gap (H2-UB − H2-LB) per intervals of number of γ_m^∞-size dominating sets. 68
4.8 Gap (H2-UB − H2-LB) per intervals of time to compute γ_m^∞. 68
4.9 Gap (H1-UB − H1-LB) and H1-UB origin for graphs with larger order and density at most 20%. 70
4.10 Gap (H2-UB − H2-LB) and H2-UB origin for graphs with larger order and density at most 20%. 70

Contents

1 Introduction **12**

1.1 The m-eternal dominating set problem 13

1.2 Related work 14

1.3 Organization of the thesis and main contributions 17

1.4 Dissimilarities in terminology and notation 18

2 The m-Eternal Dominating Set Problem for Cayley Graphs **20**

2.1 Introduction 21

2.2 Groups D_6, $\mathbb{Z}_3$ and $D_6 \times \mathbb{Z}_3$ 22

2.3 The proof by Goddard et al. 23

2.4 A counterexample 25

2.5 Computational testing 27

2.5.1 Data 27

2.5.2 Results 27

2.6 Conclusion 28

3 The m-Eternal Dominating Set Problem for Proper Interval Graphs **30**

3.1 Introduction 31

3.2 Definitions and terminology 32

3.3 A lower bound for $\gamma^{\infty}_{m,m}$ 34

3.4 The values $\gamma^{\infty}_{m,m}$ and $\gamma^{\infty}_{m,1}$ for proper interval graphs 37

4 Practical Standpoint for the m-Eternal Dominating Set Problem **41**

4.1 Introduction 42

4.2 Preliminaries 43

4.3 Neocolonization-setup efficient strategy of defense and bound 45

4.4 First heuristic method 48

4.4.1 Minimum connected dominating set upper bound 48

4.4.2 Minimum and maximum minimum dominating set lower bounds . . 49

4.4.3 Dominating-set neocolonization-setup upper bound 50

4.4.4 Time-limited exact-algorithm lower bound 52

4.5 Results – Part 1 54

4.5.1 Instances 54

4.5.2 Results 54

4.6 Second heuristic method 56

4.6.1 Minimum-weight neocolonization-setup integer program 57

4.6.2 Restricted branch-and-price 59

4.7 Results – Part 2 66

4.7.1 Time limit for the restricted branch-and-price 66

4.7.2 Results 67
4.8 Results – Part 3 68
4.8.1 Instances 68
4.8.2 Results 69
4.9 Conclusion and future work 70

5 Conclusion **74**

Chapter 1

Introduction

The m-eternal dominating set problem is a graph-protection optimization problem in which one asks for the minimum number of guards that can eternally defend attacks on the vertices of a graph. Alternatively, it can be viewed as a pursuit-evasion two-player combinatorial game where, in each turn, a team of pursuers must counter the intrusion of an evader. The problem has been an active research topic in the recent years (for a survey on the subject, see the work of Klostermeyer and Mynhardt [28]) and reported to have applications in various domains, such as military policies [31], network security [1], and multi-robot patrolling [16].

The m-eternal dominating set problem restricted to specific classes of graphs has been the focus of several studies [12, 20, 21, 23]. However, there is still a lot of progress to be made in this direction of research. Such studies determined bounds on or even found the exact value of the optimum of the problem for the considered classes of graphs. Other works produced bounds on the optimum of the problem in the general case [9, 20, 25, 26]. Nevertheless, relatively little attention was given to an important issue, which is fundamental to practical applications of the problem: to provide an efficient strategy of defense for the guards. To our knowledge, experimental investigations have not been communicated before for the m-eternal dominating set problem. Finally, the computational complexity of the problem is largely unknown.

In this thesis, we disprove a published result on the m-eternal dominating set problem and tackle the problem restricted to two specific classes of graphs. Moreover, we address the issue of providing an efficient strategy of defense for the guards and perform an extensive experimental study on the m-eternal dominating set problem. Lastly, we propose and implement an exact algorithm for the m-eternal dominating set problem and prove a result on its computational complexity.

Terminology. The non-basic terms used in this text without being specified represent standard concepts from Graph Theory, Computational Complexity, and Mathematical Programming. For the definitions of these concepts, the unfamiliar reader is pointed to, respectively, the textbooks of West [35], Garey and Johnson [19], and Wolsey [36].

1.1 The m-eternal dominating set problem

Consider the following problem of graph protection. Place guards on some vertices of a graph G. An attack to G occurs at a vertex, and, to defend an attack at v, the guards must move so that one guard occupies v. In this movement, a guard can stay at the same vertex or go to an adjacent one. We are asked to determine the minimum number of guards that can eternally defend G, namely, that can eternally perform the task of defending any attack to G.

We denote this minimum number of guards by $\gamma^{\infty}_{x,y}(G)$. This notation enables us to refer to different versions of the problem above, which are characterized by the values of x and y. If $x =$ "m", then all guards can move during a defense; if $x = p \in \mathbb{N}$, $p \geq 1$, then at most p guards are allowed to move. Similarly, if $y =$ "m", then a vertex can be occupied by an arbitrary number of guards; if $y = p \in \mathbb{N}$, $p \geq 1$, then at most p guards are allowed to be positioned at a vertex. As simple examples, for the 5-vertex path P_5 and the 6-vertex cycle C_6, it holds that $\gamma^{\infty}_{1,1}(P_5) = 3 = \gamma^{\infty}_{m,1}(P_5)$ and $\gamma^{\infty}_{1,1}(C_6) = 3 > 2 = \gamma^{\infty}_{m,1}(C_6)$.

The problem of determining $\gamma^{\infty}_{m,1}(G)$ is called the *m-eternal dominating set problem* – alternatively, the *m-eternal domination problem* – and has its origin back in Constantine's approach to defend the Roman Empire, known as Roman domination [28]. Accordingly, $\gamma^{\infty}_{m,1}(G)$ is denominated the *m-eternal domination number* of G. Consider guards *eternally defending* G, that is, eternally performing the task of defending any attack to G. Note that these guards must occupy, initially and after responding to each attack, a dominating set of G (a set S of vertices of G such that every vertex of G that is not in S has a neighbor in S). The problem's name captures the idea of a set of vertices that can be eternally modified yet remaining a dominating set.

An important result regarding $\gamma^{\infty}_{m,1}$ distinguished this parameter from $\gamma^{\infty}_{m,m}$. Allowing multiple guards to occupy the same vertex can only lessen the minimum number of guards required to eternally defend a graph. Hence, $\gamma^{\infty}_{1,m} \leq \gamma^{\infty}_{1,1}$ and $\gamma^{\infty}_{m,m} \leq \gamma^{\infty}_{m,1}$. Burger et al. [8] showed that $\gamma^{\infty}_{1,m} = \gamma^{\infty}_{1,1}$, that is, the lessening above does not occur when at most one guard can move during a defense. Goddard et al. [20] conjectured that, analogously, $\gamma^{\infty}_{m,m} = \gamma^{\infty}_{m,1}$, and other authors [9, 23] mentioned the question if this equality was true. Later, Finbow et al. [13] proved that the difference between these last parameters can be arbitrarily large.

Based on our information, despite being cited earlier [27], the proof by Finbow et al. had not been published until recently [13]. For not having access to their work, we independently developed the alternative proof below. Our demonstration was inspired by a construction presented by Fomin et al. [14].

Theorem 1.1. *For a positive integer p, there exists a graph G such that $\gamma^{\infty}_{m,1}(G) - \gamma^{\infty}_{m,m}(G) \geq p$.*

Proof. Define G to be the graph of Figure 1.1 with $q = p + 3$. For $i = 1, 2, \ldots, q$, let the *petal* H_i of G be the subgraph of G induced by the vertices $v_{i,1}, v_{i,2}, \ldots, v_{i,7}$. Moreover, set $S_{i,a} = \{ v_{i,2}, v_{i,6}, v_{i,7} \}$, $S_{i,b} = \{ v_{i,3}, v_{i,5}, v_{i,7} \}$, $S_{i,c} = \{ v_{i,4}, v_{i,5}, v_{i,6} \}$, $S_{i,d} = \{ v_{i,1}, v_{i,6}, v_{i,7} \}$. Below, we prove that $\gamma^{\infty}_{m,1}(G) \geq 2q + 1$ and $\gamma^{\infty}_{m,m}(G) \leq q + 4$; it follows that $\gamma^{\infty}_{m,1}(G) - \gamma^{\infty}_{m,m}(G) \geq p$.

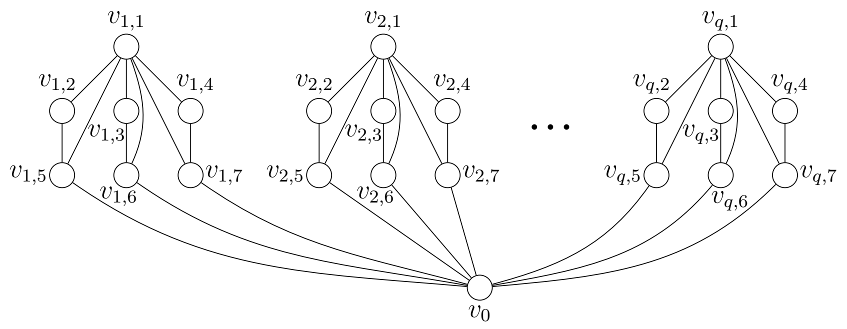


Figure 1.1: A graph satisfying $\gamma^{\infty}_{m,1} = 2q + 1 > q + 4 = \gamma^{\infty}_{m,m}$ for $q > 3$.

Suppose $\gamma^{\infty}_{m,1}(G) \leq 2q$, and consider $2q$ guards eternally defending G with at most one guard per vertex. After the defense of an attack at v_0, for some petal H_i of G, at most one guard is placed on the vertices of H_i. If the next attack occurs at $v_{i,2}$, then, after the defense of this attack, $v_{i,1}$, $v_{i,3}$, and $v_{i,4}$ are unoccupied and at most one of $v_{i,5}$, $v_{i,6}$, and $v_{i,7}$ is occupied. As a consequence, for $v = v_{i,3}$ or $v = v_{i,4}$, if the next attack happens at v, then it cannot be defended, which is a contradiction. Thus, $\gamma^{\infty}_{m,1}(G) \geq 2q + 1$.

We now show that, if multiple guards can occupy a vertex, then $q + 4$ guards can eternally defend G; hence $\gamma^{\infty}_{m,m}(G) \leq q + 4$. Place $q + 4$ guards on G: one guard on $v_{i,1}$ for $i = 1, 2, \ldots, q$ and four guards on v_0. Assume that the first attack to G occurs at a vertex v of a petal H_i of G, and let $v \in S$ with $S \in \{ S_{i,a}, S_{i,b}, S_{i,c}, S_{i,d} \}$. To defend this attack, the guard on $v_{i,1}$ and two guards on v_0 move to occupy the vertices of S. While the next attack happens at a vertex of H_i, the guards on H_i move to occupy the vertices of an appropriate set $S \in \{ S_{i,a}, S_{i,b}, S_{i,c}, S_{i,d} \}$. When the next attack occurs at a vertex of a different petal H_j of G, the guards on H_i return to their initial positions on G and the remaining guards move in a way analogous to the defense of the first attack to G. The movement of the latter guards initiates a new instance of the process above, which is repeated for every maximal sequence of attacks on vertices of a particular petal of G. $\square$

1.2 Related work

Next, we review the literature on the m-eternal dominating set problem – for additional bibliography, the reader is referred to the thorough survey of Klostermeyer and Mynhardt [28]. In this section, the following notation is adopted. For a graph G, we denote by $V(G)$ and $\bar{G}$, respectively, the vertex set of G and the complement of G. Given $S \subseteq V(G)$, we write $G[S]$ for the subgraph of G induced by S. For simplicity, we employ γ^{∞}_{m} in place of $\gamma^{\infty}_{m,1}$ to designate the m-eternal domination number.

The first studied instance of $\gamma^{\infty}_{x,y}$ was $\gamma^{\infty}_{1,1}$. Burger et al. [8] introduced this parameter and established several initial results. Given a graph G, a *clique cover* of G is a set of cliques Q_1, Q_2, $\ldots$, Q_k of G such that $\bigcup_{i=1}^{k} Q_i = V(G)$. Let $\alpha(G)$ and $\theta(G)$ be, respectively, the maximum size of an independent set of G and the minimum size of a clique cover of G. Theorem 1.2 presents a lower and an upper bound on $\gamma^{\infty}_{1,1}(G)$ proved by the authors.

Theorem 1.2 ([8]). *For a graph G, $\alpha(G) \leq \gamma_{1,1}^{\infty}(G) \leq \theta(G)$.*

Goddard et al. [20] published a seminal paper on γ_m^{∞}. Define $\gamma(G)$ to be the minimum size of a dominating set of a graph G. Theorems 1.3 and 1.4 give bounds on $\gamma_m^{\infty}(G)$ provided in the mentioned paper, with Theorems 1.3 and 1.2 combining nicely.

Theorem 1.3 ([20]). *For a graph G, $\gamma(G) \leq \gamma_m^{\infty}(G) \leq \alpha(G)$.*

Theorem 1.4 ([20]). *For a graph G, $\gamma_m^{\infty}(G) \leq 2\gamma(G)$.*

For a connected graph G, consider $\gamma_c(G)$ to be the minimum size of a connected dominating set of G. As defined by Goddard et al. [20], a *neocolonization* of a graph G is a partition Z of $V(G)$, say $Z = \{V_1, V_2, \ldots, V_k\}$, such that, for every $V_i \in Z$, $G[V_i]$ is connected. The *weight* of Z is given by $\sum_{V_i \in Z} w(V_i)$, where $w(V_i) = 1$ if V_i is a clique and $w(V_i) = \gamma_c(G[V_i]) + 1$ otherwise. Denote by $\theta_C(G)$ the minimum weight of a neocolonization of G – one can see that $\theta_C(G) \leq \theta(G)$ by observing that a neocolonization of G of weight $\theta(G)$ can be obtained from a minimum clique cover of G. Theorems 1.5 and 1.6 exhibit upper bounds on γ_m^{∞} also proved by Goddard et al. [20].

Theorem 1.5 ([20]). *For a graph G, $\gamma_m^{\infty}(G) \leq \theta_C(G)$.*

Theorem 1.6 ([20]). *For a connected graph G, $\gamma_m^{\infty}(G) \leq \theta_C(G) \leq \gamma_c(G) + 1$.*

As reported by Klostermeyer and Mynhardt [28], the same authors [25] obtained an upper bound on $\gamma_m^{\infty}(G)$ for a connected graph G with at least two vertices. Chambers et al. [9] achieved bounds on γ_m^{∞} somewhat different from the ones listed so far. Let $\tau(G)$ be the minimum size of a vertex cover of a graph G. Theorems 1.7-1.9 put forward the results just mentioned.

Theorem 1.7 ([25]). *For a connected graph G with at least two vertices, $\gamma_m^{\infty}(G) \leq 2\tau(G)$.*

Theorem 1.8 ([9]). *For a connected graph G of order n, $\gamma_m^{\infty}(G) \leq \lceil n/2 \rceil$.*

Theorem 1.9 ([9]). *For a graph G of order n, $\gamma_m^{\infty}(G) + \gamma_m^{\infty}(\bar{G}) \leq n + 1$.*

Above, we state bounds on the m-eternal domination number imposing no restriction or essentially just connectivity on the input graph – Figure 1.2 displays a Hasse diagram [4] on a representation of these bounds as a partially ordered set. Klostermeyer and MacGillivray [23], Klostermeyer and Mynhardt [26], and Chambers et al. [9] investigated restrictions on the input graph under which equality would hold in such bounds. Klostermeyer and MacGillivray showed that, for trees, $\gamma_m^{\infty} = \theta_C$; Klostermeyer and Mynhardt characterized connected graphs for which $\gamma_m^{\infty} = 2\tau$; and Chambers et al. specified $\mathcal{G}$ such that, for a graph G of order n, $\gamma_m^{\infty}(G) + \gamma_m^{\infty}(\bar{G}) = n + 1$ if and only if G or $\bar{G} \in \mathcal{G}$.

In a paper published during the work of the present thesis, we proved that, for proper interval graphs, $\gamma_m^{\infty} = \alpha$ (see Chapter 3). This result can be extended as in Theorem 1.10. The facts that $\alpha = \theta$ for perfect graphs [35] and proper interval graphs are perfect [35] imply the third equality below. This equality combined with our initial result, Theorem 1.5, and $\theta_C \leq \theta$ builds the remainder of the theorem.

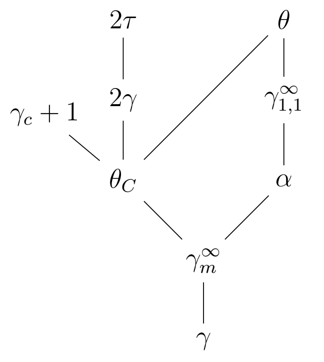


Figure 1.2: *(Adaptation of a figure due to Goddard et al. [20].)* A Hasse diagram [4] on a partially ordered set representation of bounds on the m-eternal domination number.

Theorem 1.10. *For a proper interval graph G, $\gamma_m^{\infty}(G) = \theta_C(G) = \alpha(G) = \theta(G)$.*

Recently, Rinemberg and Soulignac [33] revealed that, for interval graphs, $\gamma_m^{\infty} = \theta_C$. Take the interval graph given by the k-leaf star $K_{1,k}$ with $k \geq 3$. Note that $\gamma_m^{\infty}(K_{1,k}) = 2 < k = \alpha(K_{1,k}) = \theta(K_{1,k})$. Hence, Rinemberg and Soulignac's result cannot be extended to generalize Theorem 1.10 for all interval graphs. Nevertheless, their proof can be particularized to demonstrate Theorem 1.10 [33] and, therefore, to arrive at our result on proper interval graphs.

Goddard et al. [20] determined the exact value of γ_m^{∞} for the classes of graphs indicated in Theorem 1.11. Henning et al. [22] provided the upper bound $7n/16$ on γ_m^{∞} for cubic bipartite graphs of order n. Klostermeyer and Mynhardt showed that, while $\gamma_m^{\infty} \leq \tau$ for connected graphs with minimum degree at least 2 [25], $\tau \leq \gamma_m^{\infty}$ for trees with at least two vertices [26]. Klostermeyer and Mynhardt characterized the trees with at least two vertices satisfying $\gamma_m^{\infty} = \tau$ [26], and the same was done for the bounds γ, α, 2γ, $\gamma_c + 1$ [24], and 2τ [26].

Theorem 1.11 ([20])**.**

(a) For a complete graph G, $\gamma_m^{\infty}(G) = 1$.

(b) For a complete bipartite graph G with at least two edges, $\gamma_m^{\infty}(G) = 2$.

(c) For the path P_n of order n, $\gamma_m^{\infty}(P_n) = \lceil n/2 \rceil$.

(d) For the cycle C_n of order n, $\gamma_m^{\infty}(C_n) = \lceil n/3 \rceil$.

A *grid graph* consists of the Cartesian product $P_m \square P_n$ [35] of two paths P_m and P_n. Several authors [3, 12, 21, 34] studied γ_m^{∞} for grid graphs. Some of their findings are listed in Theorem 1.12.

Theorem 1.12.

(a) [21] For $n \geq 2$, $\gamma_m^{\infty}(P_2 \square P_n) = \lceil 2n/3 \rceil$.

(b) [21] For $2 \leq n \leq 8$, $\gamma_m^{\infty}(P_3 \square P_n) = n$;
[21] for $n \geq 9$, $\gamma_m^{\infty}(P_3 \square P_n) \leq \lceil 8n/9 \rceil$;
[12] for $n > 11$, $1 + \lceil 4n/5 \rceil \leq \gamma_m^{\infty}(P_3 \square P_n) \leq 2 + \lceil 4n/5 \rceil$.

(c) *[3] For* $n \geq 1$, $n \notin \{2, 6\}$, $\gamma_m^\infty(P_4 \square P_n) = 2\lceil (n+1)/2 \rceil$;
[3] $\gamma_m^\infty(P_4 \square P_2) = 3$ *and* $\gamma_m^\infty(P_4 \square P_6) = 7$.

(d) *[34] For* $1 \leq n \leq 12$, $n \notin \{9, 10\}$, $\gamma_m^\infty(P_5 \square P_n) = \lfloor (6n+9)/5 \rfloor$ *and, for* $n \in \{9, 10\}$, $\gamma_m^\infty(P_5 \square P_n) = \lfloor (6n+11)/5 \rfloor$;
[34] for $n > 12$, $\lfloor (6n+9)/5 \rfloor \leq \gamma_m^\infty(P_5 \square P_n) \leq \lfloor (4n+4)/3 \rfloor$.

(e) *[3] For* $n \geq 1$, $\lfloor 10(n+1)/7 \rfloor \leq \gamma_m^\infty(P_6 \square P_n) \leq \lceil 8n/5 + 8 \rceil$;
[3] $\gamma_m^\infty(P_6 \square P_6) = 10$.

(f) *[3]* $13 \leq \gamma_m^\infty(P_7 \square P_7) \leq 14$.

Contributions were made on solving the m-eternal dominating set problem for specific classes of graphs. Klostermeyer and MacGillivray [23] explained how to solve the problem for trees in linear time. In our paper on proper interval graphs mentioned above, we showed that, for these graphs, γ_m^∞ can be computed in linear time. Rinemberg and Soulignac [33] extended our result by presenting a linear-time algorithm to determine γ_m^∞ for interval graphs.

Early in the work of this thesis, we designed and implemented an exact algorithm for the m-eternal dominating set problem. We were inspired by a configuration-graph approach developed by Fomin et al. [15] to a closely related problem, namely, the eternal vertex cover problem [15]. Finbow et al. [13] and Bard et al. [2] also proposed configuration-graph algorithms for the m-eternal dominating set problem, whereas Klostermeyer et al. [29] proceeded similarly for an eviction variant of the problem. Furthermore, the ideas of Bard et al. [2] were the basis of an algorithm implemented by Křišťan [30].

The problem of computing $\gamma_{m,m}^\infty$ was shown to be a special case of a spy game [11], and the decision version of the game in this special case was proved to be NP-hard [10]. The decision version of the eternal vertex cover problem was proved to be NP-hard [15]. The decision versions of other problems similar to the m-eternal dominating set problem, given by two-player combinatorial games, were proved to be PSPACE-hard [16, 17, 32] or PSPACE-complete [18]. However, from what we know of works preceding ours, complexity results regarding the NP and PSPACE classes have not been reported before for the decision version of the m-eternal dominating set problem.

1.3 Organization of the thesis and main contributions

This thesis is organized as a collection of papers. Chapters 2-4 reproduce papers published or submitted for publication during this doctoral work – following the rules of the University of Campinas, the original texts are reproduced without modifications except for layout and formatting adjustments. Chapter 5 closes the document with final considerations and directions for future research.

Below, we outline the main contributions of the thesis, which are presented in the following three chapters. These chapters contain prologues that summarize the contents of and provide technical information on the corresponding papers. In addition, they have separate bibliography sections that list the references within the respective manuscripts.

While reviewing the literature on the m-eternal dominating set problem, we found an error: a result published by Goddard et al. [20] on Cayley graphs was not valid. We arrived at this conclusion when investigating a statement by the same authors [20]: that the result was probably true for any vertex-transitive graph. In Chapter 2, we disprove the invalid result, show that it holds for a subclass of the Cayley graphs, and report further findings on the m-eternal domination number of these graphs.

In Chapter 3, we tackle the m-eternal dominating set problem for the well-studied class of proper interval graphs: we show that, for these graphs, the m-eternal domination number equals the maximum size of an independent set and, as a consequence, the m-eternal dominating set problem can be solved in linear time. We also prove that, for proper interval graphs, there is no advantage in allowing multiple guards to occupy the same vertex and there is a straightforward strategy of defense for the smallest possible number of guards. To obtain the results above, we establish a lower bound on $\gamma^{\infty}_{m,m}$ that may be useful to deal with other classes of graphs as well.

In Chapter 4, we address an issue that is fundamental to practical applications of the m-eternal dominating set problem but that has received relatively little attention. The issue is to obtain not only a good-quality upper bound on the m-eternal domination number but also an associated efficient strategy of defense. We introduce two heuristic methods, in which we propose and solve integer and constraint programming models to compute bounds on the m-eternal domination number. By performing an extensive experiment to validate the features of these methods, we generate a 750-instance benchmark and carry out a practical evaluation of bounds for the m-eternal domination number available in the literature.

Finally, also in Chapter 4, we propose an exact algorithm for the m-eternal dominating set problem (see Section 4.4.4) and contribute to the knowledge on its complexity: we prove that the decision version of the problem is NP-hard. As far as we know, the proposed algorithm was the first developed and implemented exact method for the m-eternal dominating set problem. This algorithm was initially used in the aforementioned study of Cayley graphs. Later, it was executed with a time limit in the heuristic methods cited above.

1.4 Dissimilarities in terminology and notation

In the three papers presented in Chapters 2-4, we give definitions in terms of dominating sets for the m-eternal dominating set problem. These definitions vary from one chapter to another reflecting the change on our understanding of how to best put the problem in dominating-set terms. In Chapters 2-4, we also make use of different terminology and notation. Below, we comment on these dissimilarities.

In the paper of Chapter 2, we disprove a result published by Goddard et al. [20]. We follow their terminology and notation, where *eternal m-security number* and σ_m are equivalent, respectively, to m-eternal domination number and $\gamma^{\infty}_{m,1}$. Moreover, we define the problem of determining σ_m – i.e., the m-eternal dominating set problem – as the problem of calculating the minimum size of an *eternal m-secure set*, namely, a dominating set from which, for any sequence of vertices, one can construct a corresponding sequence of dominating sets satisfying certain conditions. This definition is in accordance with the language used by Goddard et al. [20] and is a simple rewriting of a definition given by Klostermeyer and MacGillivray [23].

The preceding definition of the m-eternal dominating set problem is somehow problematic because it demands clarification on its sequence of vertices, which models a sequence of attacks. It should be clarified that the attacks are chosen and revealed by an attacker as the guards perform the defenses. The problem is not the same if the sequence of attacks is revealed in advance [28]. In the paper of Chapter 3, we proceed differently: we state the problem as to compute the minimum size of an *eternal dominating set*, a recursively defined set that recasts a concept introduced by Chambers et al. [9].

Furthermore, in Chapter 3, we deal simultaneously with the numbers $\gamma^{\infty}_{m,1}$ and $\gamma^{\infty}_{m,m}$. Within this context, one probably associates more easily the problem of determining $\gamma^{\infty}_{m,1}$ with the term *eternal dominating set problem* than with the term m-eternal dominating set problem. Accordingly, we employ the former term in place of the latter although the eternal dominating set problem more frequently means the problem of finding $\gamma^{\infty}_{1,1}$ [28].

Lastly, in the paper of Chapter 4, we aim at explicitly representing in terms of dominating sets a *strategy of defense* that k guards can follow to eternally defend a graph. To this end, we define a *k,m-eternal dominating set collection*, which is an explicit representation of all placements where the k guards may be initially and after responding to an attack. As a consequence of Theorem 4.1 (see Chapter 4), the m-eternal dominating set problem can be put as the problem of calculating the minimum positive integer k such that the input graph has a k,m-eternal dominating set collection. For convenience, we write γ^{∞}_{m} instead of $\gamma^{\infty}_{m,1}$ for the m-eternal domination number.

Chapter 2

A Note on the Paper "Eternal Security in Graphs" by Goddard, Hedetniemi, and Hedetniemi (2005)

Goddard et al. [20] stated that the m-eternal domination number and the minimum size of a dominating set were equal for Cayley graphs. In addition, they claimed that the same was probably true for all vertex-transitive graphs. However, while investigating the claim on the latter graphs, we found that the statement on the former graphs was not valid.

In this chapter, we disprove the invalid statement of Goddard et al. [20] and show that it holds for a subclass of the Cayley graphs. Moreover, we study the difference between the m-eternal domination number and the minimum size of a dominating set for Cayley graphs. To this end, we compute these parameters for a large number of instances from an existing Cayley graph repository. We show that the difference between the parameters can be indefinitely increased for disconnected Cayley graphs and leave open the question of whether it can be greater than 1 if connectivity is enforced.

The subsequent text is a reproduction of a paper published in the *Journal of Combinatorial Mathematics and Combinatorial Computing* [7] and co-authored by Cid C. de Souza[1] and Orlando Lee[1]. In this work, to compute the m-eternal domination number of Cayley graphs, we use our exact algorithm for the m-eternal dominating set problem (see Section 1.3). Before proceeding, the reader is invited to recall the notes on the terminology and notation employed in this chapter's paper (see Section 1.4).

Abstract

In the paper "Eternal security in graphs" by Goddard, Hedetniemi and Hedetniemi (2005, [4]), the authors claimed that, for any Cayley graph, the *eternal m-security number* equals the minimum cardinality of a *dominating set*. However, the equality is false. In this note, we present a counterexample and comment on the eternal m-security number for Cayley graphs.

[1]Institute of Computing, University of Campinas, Brazil.

2.1 Introduction

Goddard, Hedetniemi, and Hedetniemi [4] studied the problem of determining the *eternal 1-security number* of a graph. They defined this problem as finding the minimum cardinality of an *eternal 1-secure set* of the graph. This cardinality was first considered by Burger et al. [2].

The same authors also investigated a related problem: determining the *eternal m-security number* of a graph, denoted by σ_m (some authors denote the eternal m-security number by γ_m^∞). This problem consists in finding the minimum cardinality of an *eternal m-secure set* of the graph. For a better understanding, we state the problem in a formal way. To this end, given a simple graph $G = (V, E)$, we first define the concepts of a *shift* and of a *dominating set*.

Take two sets of vertices $A, B \subseteq V$. A shift from A to B is a bijective function $f : A \to B$ such that, if $f(u) = v$, then $u = v$ or $uv \in E$. Notice that there is a shift from A to B only if $|A| = |B|$.

A set $D \subseteq V$ is a dominating set of G if, for each $v \in (V \setminus D)$, there is a vertex $u \in D$ such that $uv \in E$. We denote the minimum cardinality of a dominating set of G by $\gamma(G)$.

A dominating set $D_0 \subseteq V$ is an eternal m-secure set of G if, for any sequence of vertices $v_1, v_2, \ldots \in V$, one can construct a sequence of dominating sets $D_1, D_2, \ldots$ of G such that, for $i = 1, 2, \ldots$:

1. There is a shift from D_{i-1} to D_i;
2. $v_i \in D_i$.

The problem of determining σ_m admits the following interpretation. Consider guards placed on the vertices of a graph with at most one guard per vertex. Suppose that an attack occurs at a vertex. To defend the attack, one guard must move from an adjacent vertex to the attacked one, unless it already had a guard. The other guards may move to prepare to defend a next attack. The problem is to find the minimum number of guards so that attacks can be defended indefinitely. The computation of the eternal 1-security number of a graph corresponds to a version of this problem in which only one guard can move per defense. We shall use this interpretation in later arguments since it is more intuitive, although not strictly formal.

In Goddard et al. [4], the authors established the value of σ_m for graphs from several classes. They also presented bounds on σ_m for general graphs. One class studied by the authors is that of Cayley graphs, for which they stated Theorem 2.1 reproduced below. However, we found that this result is not valid. Prior to write down the theorem and exhibit the counterexample, we recall the definition of a Cayley graph.

A Cayley graph is a simple graph $G = (V, E)$ defined as follows. Consider a *group* Γ and a set C of elements of Γ satisfying:

(i) C does not contain the identity of Γ;

(ii) If $x \in C$, then $x^{-1} \in C$ (x^{-1} is the inverse of element x in Γ).

The Cayley graph $G = CG(\Gamma, C)$ of Γ with respect to C is such that $V = \Gamma$ and $xy \in E$ if and only if $x = hy, h \in C$. For connecting the vertices of G, the elements of C and C

itself are called *connectors* and *connecting set*. One can prove that G is connected if and only if C *generates* Γ.

The theorem presented in Goddard et al. [4] follows. We disprove it by showing a Cayley graph for which $\gamma < \sigma_m$.

Theorem 2.1 (Goddard et al. [4, Theorem 10])**.** *For any Cayley graph G, $\gamma(G) = \sigma_m(G)$.*

This paper is organized as follows. In the next section, we briefly describe the groups used for constructing the graph that invalidates Theorem 2.1. In Section 2.3, we discuss the proof given by Goddard et al. [4]. In Section 2.4, we present the counterexample that we encountered. In Section 2.5, we comment on computational tests we carried out with Cayley graphs in an attempt to establish the exact relation between γ and σ_m for this class. Finally, in Section 2.6, we make some final remarks.

2.2 Groups D_6, $\mathbb{Z}_3$ and $D_6 \times \mathbb{Z}_3$

In this section, we define the groups D_6, $\mathbb{Z}_3$ and $D_6 \times \mathbb{Z}_3$. For more details on these groups, see, for example, the textbook by Dummit and Foote [3].

A regular polygon of n sides have $2n$ symmetries: n rotations and n reflections. Because of this number, the set of its symmetries is denoted by D_{2n}. The set D_{2n} under the operation of composition is a group. This group is called the *dihedral group* and, with some abuse of notation, is also denoted only by D_{2n}.

The elements of the dihedral group D_{2n}, in multiplicative notation, are given by

$$D_{2n} = \{1, r, r^2, \ldots, r^{n-1}, s, sr, sr^2, \ldots, sr^{n-1}\},$$

with $1, r, r^2, \ldots, r^{n-1}$ corresponding to the n rotations and $s, sr, sr^2, \ldots, sr^{n-1}$ corresponding to the n reflections of the polygon. This group admits the following *presentation*:

$$D_{2n} = \langle r, s \mid r^n = s^2 = 1, rs = sr^{-1} \rangle.$$

From this presentation, one can obtain the result of operations on elements.

The group D_6 is the defined by the symmetries of a triangle. To facilitate the arguments made further in this paper, we display the multiplication table of this group in Figure 2.1.

Now, the set of possible remainders after dividing an integer by 3 is $\mathbb{Z}_3 = \{0, 1, 2\}$. Under the operation of addition modulo 3, this set is a group. This group is an instance of the *cyclic group* and is denoted $\mathbb{Z}_3$, like the set itself.

The groups D_6 and $\mathbb{Z}_3$ can be used to form a new group through their *direct product*. The direct product of D_6 and $\mathbb{Z}_3$ is denoted $D_6 \times \mathbb{Z}_3$ and is defined as follows.

The set of elements of the group $D_6 \times \mathbb{Z}_3$ is given by the Cartesian product of the sets

	1	**r**	$\mathbf{r^2}$	**s**	**sr**	$\mathbf{sr^2}$
1	1	r	r^2	s	sr	sr^2
r	r	r^2	1	sr^2	s	sr
$\mathbf{r^2}$	r^2	1	r	sr	sr^2	s
s	s	sr	sr^2	1	r	r^2
sr	sr	sr^2	s	r^2	1	r
$\mathbf{sr^2}$	sr^2	s	sr	r	r^2	1

Figure 2.1: Compositions of symmetries for D_6

D_6 and $\mathbb{Z}_3$ – also denoted $D_6 \times \mathbb{Z}_3$. We have that

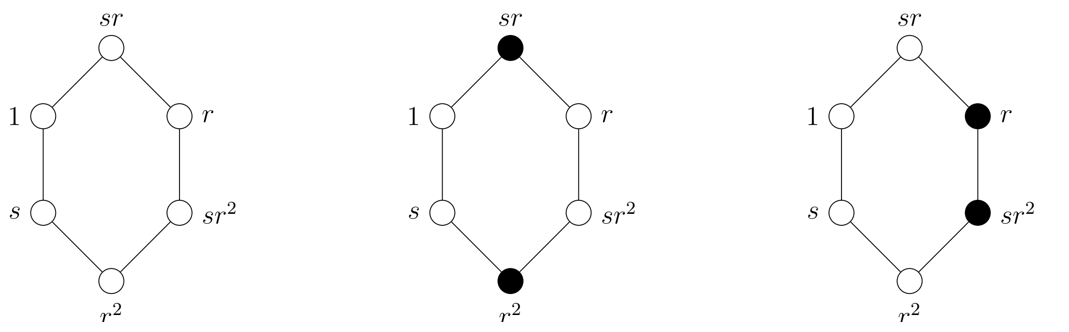
$$\begin{aligned} D_6 \times \mathbb{Z}_3 = \{&(1,0),(1,1),(1,2),(r,0),(r,1),(r,2),\\ &(r^2,0),(r^2,1),(r^2,2),(s,0),(s,1),(s,2),\\ &(sr,0),(sr,1),(sr,2),(sr^2,0),(sr^2,1),(sr^2,2)\}. \end{aligned}$$

The operation of the group occurs componentwise: the result for the first half of the elements is shown in Figure 2.1 and the result for the second half is given by the addition modulo 3.

2.3 The proof by Goddard et al. [4]

We use the group D_6 to exhibit a flaw in the proof by Goddard et al. [4] to Theorem 2.1. The group D_6 is non-abelian (non-commutative) and it is its non-commutative property that allows us to reach our goal.

First of all, let us define a Cayley graph $G_1 = CG(D_6, C_1)$ of this group. For that, we choose the connecting set $C_1 = \{s, sr\}$. From the table in Figure 2.1, we have that G_1 is the graph depicted in Figure 2.2(a).



(a) G_1 (b) D_1 (black vertices) (c) hD_1 (black vertices)

Figure 2.2: The Cayley graph G_1 and the sets of vertices D_1 and hD_1

Now, let us follow the proof given by Goddard et al. [4]. Consider the dominating set $D_1 = \{r^2, sr\}$ pictured in Figure 2.2(b). Suppose an attack occurs at r. The only vertex in D_1 adjacent to r is sr. As they are adjacent, for some $h \in C_1$, $r = h(sr)$. By Figure 2.1, $h = s$.

Let xA and Ax, for an element $x \in D_6$ and a set $A = \{a_1, a_2, \ldots, a_k\} \subseteq D_6$, stand for

$$xA = \{xa_1, xa_2, \ldots, xa_k\} \text{ and}$$
$$Ax = \{a_1x, a_2x, \ldots, a_kx\} \text{ respectively.}$$

Goddard et al. [4] claimed that

$$hD_1 = sD_1 = \{sr^2, s(sr)\} = \{sr^2, r\}$$

is a dominating set. However, as it is clear from Figure 2.2(c), this is not true.

We raise the possibility that the authors mistakenly considered the mapping

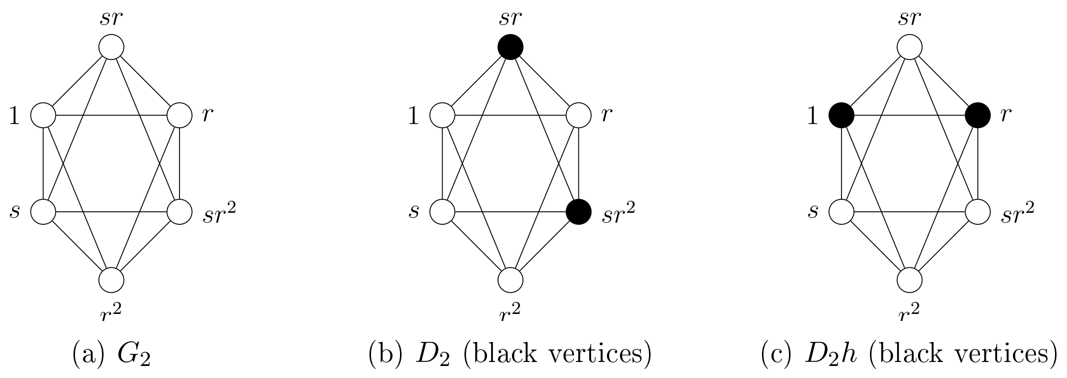
$$f(v) = hv, \text{ for each vertex } v \text{ of } G_1 \tag{2.1}$$

an automorphism of G_1. Instead, by a known result for Cayley graphs [1], it is the mapping

$$f(v) = vh, \text{ for each vertex } v \text{ of } G_1 \tag{2.2}$$

which is an automorphism of G_1.

We also observe that, in general, a mapping of the form (2.2) does not correspond to a shift from one dominating set to another. As an example, let us consider applying a mapping of the form (2.2) to a dominating set of the Cayley graph $G_2 = CG(D_6, C_2)$ defined by the connecting set $C_2 = \{s, sr, r, r^2\}$ – see a drawing of G_2 in Figure 2.3(a). The dominating set is $D_2 = \{sr, sr^2\}$ – shown in Figure 2.3(b). Let us choose $h = sr^2$. The outcome is: sr is mapped to the adjacent vertex r, but sr^2 is mapped to the non-adjacent vertex 1 – the resulting dominating set is shown in Figure 2.3(c).



(a) G_2 (b) D_2 (black vertices) (c) D_2h (black vertices)

Figure 2.3: The Cayley graph G_2 and the dominating sets D_2 and D_2h

At last, we point out that the proof by Goddard et al. [4] and their result are valid for a subclass of Cayley graphs. This subclass consists of graphs defined as follows.

A Cayley graph G is *obtainable from an abelian group* if there is an abelian group Γ and a connecting set C such that $G = CG(\Gamma, C)$. Note there may also be a non-abelian group Γ' and a connecting set C' satisfying $G = CG(\Gamma', C')$. For graphs defined in this way, mappings of the forms (2.1) and (2.2) coincide (because the elements commute). For this reason, the following theorem holds.

Theorem 2.2. *For any Cayley graph G obtainable from an abelian group, $\gamma(G) = \sigma_m(G)$.*

2.4 A counterexample

We present a Cayley graph G such that $\gamma(G) < \sigma_m(G)$. The construction of G follows immediately. In the sequel, we prove our claim.

We construct $G = CG(D_6 \times \mathbb{Z}_3, C)$ from the group $D_6 \times \mathbb{Z}_3$. For simplicity, we label the vertices of G as $v_1, v_2, \ldots, v_{18}$ according to the correspondence shown in Table 2.1. We choose the connecting set

$$C = \{(s,1),(s,2),(sr,1),(sr,2),(r,1),(r^2,2)\}$$

to provide the edges of G. The graph is pictured in Figure 2.4.

v_1	v_2	v_3	v_4	v_5	v_6
$(1,0)$	$(1,1)$	$(1,2)$	$(r,0)$	$(r,1)$	$(r,2)$
v_7	v_8	v_9	v_{10}	v_{11}	v_{12}
$(r^2,0)$	$(r^2,1)$	$(r^2,2)$	$(s,0)$	$(s,1)$	$(s,2)$
v_{13}	v_{14}	v_{15}	v_{16}	v_{17}	v_{18}
$(sr,0)$	$(sr,1)$	$(sr,2)$	$(sr^2,0)$	$(sr^2,1)$	$(sr^2,2)$

Table 2.1: Labels of vertices of G

We prove our claim in Theorem 2.6. Before doing so, we provide three lemmas. In the first one, we observe that $\gamma(G) = 3$. In the second lemma, we show that vertex v_1 is contained in only one minimum dominating set. In the last lemma, we argue that no vertex is contained in more than one minimum dominating set. Finally, after the latter, we exhibit all minimum dominating sets of G.

Lemma 2.3. *It holds that $\gamma(G) = 3$.*

Proof. As G is 6-regular, two vertices dominate at most 14 vertices. Since G has 18 vertices, we have that $\gamma(G) > 2$.

Observing Figure 2.4, we can see the set $\{v_1, v_6, v_8\}$ is a dominating set. Therefore, $\gamma(G) = 3$. □

Lemma 2.4. *The vertex v_1 is contained in only one minimum dominating set.*

Proof. Let us construct a minimum dominating set D that contains v_1. By Lemma 2.3, we have that D has three vertices. So, our task is to choose two more vertices.

Since D contains v_1, we can see in Figure 2.4 that 7 vertices are already dominated by D: the vertices with thick border. Also in Figure 2.4, we can see that every vertex except

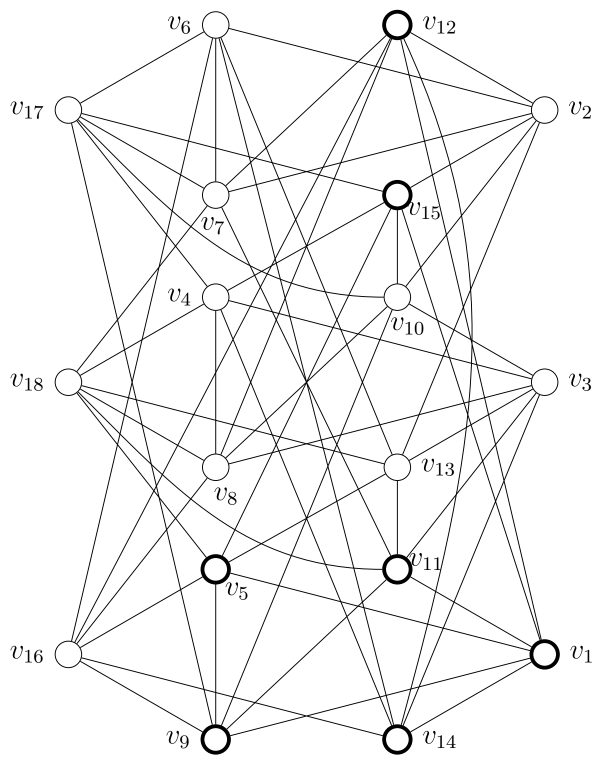


Figure 2.4: A Cayley graph G such that $\gamma(G) = 3$ and $\sigma_m(G) = 4$

v_6 and v_8 is adjacent to at least two already dominated vertices. Thus, two such vertices dominate at most 10 more vertices (summing up 17 vertices). Therefore, to construct D, we must choose at least one of v_6 and v_8.

Suppose we choose v_6. By Figure 2.4, we can see the only way of choosing one more vertex and dominating all vertices not yet dominated, is by selecting v_8. Then, suppose we pick v_8. By Figure 2.4, we can see v_6 must also be chosen for us to end up with a minimum dominating set. Hence $D = \{v_1, v_6, v_8\}$. □

Lemma 2.5. *No vertex of G is contained in more than one minimum dominating set.*

Proof. Suppose some vertex v_i is contained in two different minimum dominating sets $D_1 = \{v_i, v_j, v_k\}$ and $D_2 = \{v_i, v_l, v_m\}$. Since G is vertex-transitive (by a known result for Cayley graphs [1]), there is an automorphism α mapping v_i to v_1. But, then, $D_1' = \{v_1, \alpha(v_j), \alpha(v_k)\}$ and $D_2' = \{v_1, \alpha(v_l), \alpha(v_m)\}$ are two different dominating sets containing v_1, which contradicts Lemma 2.4. □

We can state, from Figure 2.4, that the following 6 sets are dominating sets of G:

$$\begin{gathered}\{v_1, v_6, v_8\}, \{v_3, v_5, v_7\}, \{v_2, v_4, v_9\}, \\ \{v_{11}, v_{15}, v_{16}\}, \{v_{10}, v_{14}, v_{18}\}, \{v_{12}, v_{13}, v_{17}\}.\end{gathered} \tag{2.3}$$

By Lemma 2.5, these are the only minimum dominating sets.

Theorem 2.6. *It holds that $\gamma(G) < \sigma_m(G)$.*

Proof. To prove the theorem, we consider three guards defending the vertices of G. First of all, it is obvious that, to defend the vertices of a graph, guards must always be placed on vertices that form a dominating set, because an attack at an undominated vertex cannot be defended. In the case of three guards defending the vertices of G, they must always be placed on a minimum dominating set of G.

Consider then an attack at vertex v_1. By Lemma 2.4, $D_1 = \{v_1, v_6, v_8\}$ is the only minimum dominating set containing v_1. This fact implies that, after defending an attack at v_1, guards must be placed on the vertices of D_1.

Consider now an attack at vertex v_{13}. As listed in (2.3), $D_2 = \{v_{12}, v_{13}, v_{17}\}$ is the only minimum dominating set containing v_{13}. As a consequence, after defending an attack at v_{13}, guards must be placed on the vertices of D_2. However, as we can see from Figure 2.4, guards cannot move from vertices v_1, v_6 and v_8 to v_{12}, v_{13} and v_{17}, because v_{13} and v_{17} are adjacent to v_6 but to neither v_1 nor v_8.

Hence, there is a sequence of three attacks that cannot be defended by the three guards. So, at least four guards are needed to defend the vertices of G. Therefore, $\sigma_m(G) \geq 4 > 3 = \gamma(G)$ and the theorem is proved. $\square$

2.5 Computational testing

The graph presented in the previous section was found through extensive computational testing. In this section, we describe some elements of this experiment and report relevant information.

2.5.1 Data

We searched 7871 Cayley graphs of non-abelian groups of order up to 31 and of order 33 catalogued by Royle [5]. These graphs have degree (recall that Cayley graphs are regular) less than half of the number of vertices. We denote them by *set 1*.

We also searched 7871 Cayley graphs which are the complements of the graphs of set 1. These are graphs having degree greater or equal than half of the number of vertices. We refer to them as *set 2*.

2.5.2 Results

One interesting outcome of our experiments is that, for almost all graphs, we found that $\gamma = \sigma_m$. Just for 61 out of 7871, i.e., 0.77% of them, we obtained a different result. Another relevant fact is that, in these cases, the result was always that $\gamma + 1 = \sigma_m$. We also noted that for all graphs of set 2, $\gamma = \sigma_m$.

Motivated by the above findings, we searched for a graph for which $\gamma + 1 < \sigma_m$. However, as determining σ_m becomes much more time-consuming as the graphs get bigger, an exhaustive search over a huge number of Cayley graphs of non-abelian groups rapidly becomes impractical. We then moved to the strategy of generating specific graphs to attain our goal.

One successful example is the graph G_2 consisting of two disconnected copies of the graph G presented in Section 2.4. This graph is a Cayley graph of the group $(D_6 \times \mathbb{Z}_3) \times \mathbb{Z}_2$. One can see that $\gamma(G_2) = 6 < 8 = \sigma_m(G_2)$. Moreover, it is possible to generate the graphs

- G_4 consisting of four disconnected copies of G;
- G_8 consisting of eight disconnected copies of G;
- and so forth.

For $G_i, i = 4, 8, \ldots, \sigma_m(G_i) - \gamma(G_i) = i$.

It is worth emphasizing, however, that G_i is not a connected graph. We could not discover a connected Cayley graph such that $\gamma + 1 < \sigma_m$.

2.6 Conclusion

We disproved a result by Goddard et al. [4] on the eternal m-security number of Cayley graphs. We did this by presenting a Cayley graph for which $\gamma < \sigma_m$. We remarked, however, that the result of Goddard *et al.* is valid for a large subclass of Cayley graphs: the Cayley graphs obtainable from abelian groups.

We also determined computationally the value of σ_m for 7871 Cayley graphs of non-abelian groups. For almost all of them, we got that γ is indeed equal to σ_m. For the remaining graphs, we found that $\gamma + 1 = \sigma_m$. We leave open the question of whether there exists a connected Cayley graph having $\gamma + 1 < \sigma_m$.

Acknowledgements

The authors would like to thank the anonymous referee for carefully reading this paper and making helpful suggestions.

This work was funded by grants 477692/2012-5, 141964/2013-8, 302804/ 2010-2 and 303947/2008-0 from Conselho Nacional de Desenvolvimento Científico e Tecnológico (CNPq).

Bibliography

[1] A. Bondy and U. Murty. *Graph Theory.* Graduate Texts in Mathematics. Springer, 2008.

[2] A. Burger, E. Cockayne, W. Gründlingh, C. Mynhardt, J. van Vuuren, and W. Winterbach. Infinite order domination in graphs. *Journal of Combinatorial Mathematics and Combinatorial Computing*, 50:179–194, 2004.

[3] D. Dummit and R. Foote. *Abstract Algebra.* Wiley, 2003.

[4] W. Goddard, S. Hedetniemi, and S. Hedetniemi. Eternal security in graphs. *Journal of Combinatorial Mathematics and Combinatorial Computing*, 52:169–180, 2005.

[5] G. Royle. Cayley graphs, 1998. [Online; accessed 30-November-2013]. Available at: http://school.maths.uwa.edu.au/~gordon/remote/cayley/index.html.

Chapter 3

The Eternal Dominating Set Problem for Proper Interval Graphs

---

In what follows, we consider the m-eternal dominating set problem for the well-studied class of proper interval graphs. As our main result, we prove that the m-eternal domination number equals the maximum size of an independent set and, consequently, the m-eternal dominating set problem can be solved in linear time for proper interval graphs. In addition, we show that, for these graphs, there is no advantage in allowing multiple guards to occupy the same vertex and there is a straightforward strategy of defense for the smallest possible number of guards. Finally, we provide a lower bound on $\gamma^{\infty}_{m,m}$ that may also be useful for dealing with classes of graphs other than the proper interval graphs.

The subsequent text is a reproduction of a paper published in *Information Processing Letters* [5] and co-authored by Cid C. de Souza[1] and Orlando Lee[1]. Rinemberg and Soulignac [33] have recently communicated a closely related work on the m-eternal dominating set problem for interval graphs (see Section 1.2). Before proceeding, the reader is invited to recall the notes on the terminology and notation employed in this chapter's paper (see Section 1.4).

Abstract

In this paper, we solve the Eternal Dominating Set problem for proper interval graphs. We prove that, in this case, the optimal value of the problem equals the largest size of an independent set. As a consequence, we show that the problem can be solved in linear time for such graphs. To obtain the result, we first consider another problem in which a vertex can be occupied by an arbitrary number of guards. We then derive a lower bound on the optimal value of this latter problem, and prove that, for proper interval graphs, it is the same as the optimum of the first problem.

[1]Institute of Computing, University of Campinas, Brazil.

3.1 Introduction

In this paper, we deal with the following problem of guards defending attacks at the vertices of a graph. Initially, the guards are placed on some of the vertices of the graph. Then, to defend an attack at a vertex v, guards must move so that, after their movement, one guard occupies v. During a defense, a guard can only move to a neighboring vertex or stay at the same vertex. Thus, for guards to be able to defend every possible attack, the set of vertices initially occupied must constitute a dominating set. This also holds when the defense of any next attack is considered. The problem that we treat is to determine the smallest number of guards such that the guards can defend any sequence of attacks at the vertices of the graph. Since this sequence is not known in advance, we can think of it as infinite, and say that the guards defend the graph *eternally*.

Different versions of this problem have been studied. Each such version is characterized by two parameters: (i) x, the number of guards that can move during a defense, and (ii) y, the number of guards that can occupy a vertex initially or after a defense. Accordingly, we denote by $\gamma^{\infty}_{x,y}$ the optimal value of the version in which the following holds. If $x =$ "m", then all guards can move during a defense; if $x = p \in \mathbb{N}$, $p \geq 1$, at most p guards are allowed to move. Similarly, if $y =$ "m", then a vertex can be occupied by an arbitrary number of guards; if $y = p \in \mathbb{N}$, $p \geq 1$, at most p guards are allowed to be positioned at a vertex. As a simple example, for the 6-cycle C_6, it holds that $\gamma^{\infty}_{1,1}(C_6) = 3$, and $\gamma^{\infty}_{m,1}(C_6) = 2$.

The first-posed version of the problem was the one referred by $\gamma^{\infty}_{1,1}$. This version was investigated by Burger et al. [2]. Later, Goddard, Hedetniemi, and Hedetniemi [6], proposed the $\gamma^{\infty}_{m,1}$ version. Since then, the values $\gamma^{\infty}_{1,1}$, $\gamma^{\infty}_{m,1}$, and $\gamma^{\infty}_{m,m}$ were studied in several papers. We point the reader to the survey by Klostermeyer and Mynhardt [9] for further references.

In this paper, we consider the problem in $\gamma^{\infty}_{m,1}$ and $\gamma^{\infty}_{m,m}$ versions. We call the first version the Eternal Dominating Set problem, or the EDSP for short. In this setting, the placement of the guards can be simply described by a set of vertices (there is at most one guard per vertex). The term above is an attempt to capture the idea of a set of vertices that can be eternally modified, yet remaining a dominating set.

One interesting aspect of the versions of the problem is how the respective optimal values relate. For example, one can see that $\gamma^{\infty}_{1,2}(G) \leq \gamma^{\infty}_{1,1}(G)$ for any graph G. Furthermore, a stronger result by Burger et al. [2] establishes that $\gamma^{\infty}_{1,2}(G) = \gamma^{\infty}_{1,1}(G)$ for any graph G. One can also see that $\gamma^{\infty}_{m,m}(G) \leq \gamma^{\infty}_{m,1}(G)$ for any graph G. In the same vein of the equality above, Goddard et al. [6] conjectured that $\gamma^{\infty}_{m,m}(G) = \gamma^{\infty}_{m,1}(G)$ for any graph G. In addition, Chambers, Kinnersley, and Prince [3] published results for $\gamma^{\infty}_{m,m}$ and reported that they did not know examples which would make these results invalid for $\gamma^{\infty}_{m,1}$. Klostermeyer and MacGillivray [8] mentioned the conjecture as well. However, Finbow et al. [5] refuted it.

It is worth noting that $\gamma^{\infty}_{1,1}$ was already established for some classes of graphs. For example, Burger et al. [2] proved that, for any graph G, $\alpha(G) \leq \gamma^{\infty}_{1,1}(G) \leq \theta(G)$, where $\alpha(G)$ denotes the largest size of an independent set of G and $\theta(G)$ is the smallest size of a clique cover of G. As a consequence, for any perfect graph G, it holds that $\alpha(G) =$

$\gamma^{\infty}_{1,1}(G) = \theta(G)$. Also, when G is a circular-arc graph, Regan [10] showed that $\gamma^{\infty}_{1,1}(G) = \theta(G)$.

The main contribution of the present work is solving the EDSP for the well-studied class of proper interval graphs. In the subsequent sections, we prove that, for any such graph G,

$$\gamma^{\infty}_{m,m}(G) = \gamma^{\infty}_{m,1}(G) = \alpha(G).$$

As a consequence, we show that, in this case, the EDSP can be solved in linear time.

Other contributions of this work are the following. First, for the EDSP in general, to describe a defense strategy for the guards may be a non-trivial task, because, ultimately, one has to draw the movements of the guards for any sequence of attacks. We show that, for proper interval graphs, there is a straightforward strategy for the smallest possible number of guards. Moreover, to achieve the results above, we first derive a lower bound for $\gamma^{\infty}_{m,m}$. This bound applies to proper interval graphs, but may be useful for other interesting classes of graphs as well.

The rest of this paper is structured as follows. In the next section, we define $\gamma^{\infty}_{m,m}$ and $\gamma^{\infty}_{m,1}$ more formally, and present other necessary concepts. In Section 3.3, we prove the lower bound for $\gamma^{\infty}_{m,m}$ cited above. At last, in Section 3.4, we focus on proper interval graphs.

3.2 Definitions and terminology

In this section, we accumulate the definitions stated in this paper. All concepts that appear in this text and that are not defined below are standard. The graph theoretical concepts can be found, for instance, in the book by Bondy and Murty [1]. The notions of a 1-secure dominating multiset and of a k-secure dominating multiset are derived from definitions given by Chambers et al. [3].

The graphs treated throughout this paper are undirected and simple. Below, we consider a graph G, whose sets of vertices and edges are denoted by $V(G)$ and $E(G)$, respectively.

Consider two multisets A, $B \subseteq V(G)$. We construct the sets A^* and B^*, associated to them, as follows. Define A^d as the set of all distinct elements of A. For every $a \in A^d$, let $o(a)$ be the number of occurrences of a in A. Take $P(a)$ as the set of the $o(a)$ ordered pairs $\langle a, 1\rangle$, $\langle a, 2\rangle$, …, $\langle a, o(a)\rangle$. Then, $A^* = \bigcup_{a \in A^d} P(a)$. Construct B^* analogously. Notice that $|A| = |A^*|$, and $|B| = |B^*|$.

For the next concept, we consider a function f that relates A to B. Since A and B are multisets (thus not necessarily sets) and a function is defined over sets, A and B cannot be directly used to define f. We employ the sets A^* and B^* to this purpose.

A *shift* from A to B is a bijective function $f : A^* \to B^*$ such that, if $f(\langle u, p\rangle) = \langle v, q\rangle$, then $u = v$ or $uv \in E(G)$. Observe that there is a shift from A to B only if $|A| = |B|$.

A multiset $D \subseteq V(G)$ is a *dominating multiset* of G if, for each $v \in (V(G) \setminus D)$, there is a vertex $u \in D$ such that $uv \in E(G)$. A dominating multiset $D \subseteq V(G)$ is a 1-*secure dominating multiset* of G if, for every $v \in V(G)$, there is a dominating multiset D' of G

such that:

1. There is a shift from D to D';
2. $v \in D'$.

For $k \in \mathbb{N},\ k > 1$, a dominating multiset $D \subseteq V(G)$ is a *k-secure dominating multiset* of G if, for every $v \in V(G)$, there is a $(k-1)$-secure dominating multiset D' of G such that:

1. There is a shift from D to D';
2. $v \in D'$.

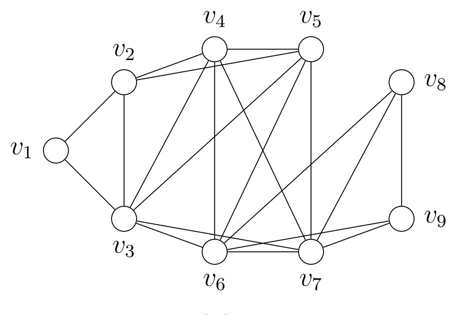
A dominating multiset $D \subseteq V(G)$ is an *eternal dominating multiset* of G if it is a k-secure dominating multiset of G for each $k \in \mathbb{N},\ k \geq 1$. The value $\gamma^{\infty}_{m,m}(G)$ denotes the smallest size of an eternal dominating multiset of G.

A *dominating set* D of G can be defined as a dominating multiset of G that is a set. We denote the smallest size of a dominating set of G by $\gamma(G)$. A *1-secure dominating set*, a *k-secure dominating set* for $k \in \mathbb{N},\ k > 1$, and an *eternal dominating set* of G are defined in the same way their multiset counterparts are with the restriction that all multisets involved are sets. The value $\gamma^{\infty}_{m,1}(G)$ denotes the smallest size of an eternal dominating set of G.

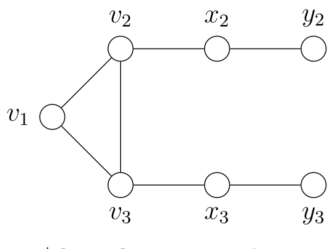
Therefore, the following inequality chain holds:

$$\gamma(G) \leq \gamma^{\infty}_{m,m}(G) \leq \gamma^{\infty}_{m,1}(G). \tag{3.1}$$

A set $S \subseteq V(G)$ is a *separator* of G if there is a partition $\{A, B\}$ of $V(G) \setminus S$ such that, for every $a \in A,\ b \in B,\ ab \notin E(G)$. As an example, for the graph in Figure 3.1(a), $S = \{v_4, v_5, v_6, v_7\}$ is a separator, since $A = \{v_1, v_2, v_3\}$ and $B = \{v_8, v_9\}$ satisfy the requirements above.



(a) G



(b) $G^{+}[A, S]$ for $A = \{v_1, v_2, v_3\}$ and $S = \{v_4, v_5, v_6, v_7\}$

Figure 3.1: A graph G and a graph $G^{+}[A, S]$.

Given a set $X \subseteq V(G)$, we denote by $G[X]$ the subgraph of G induced by X. Given disjoint sets $A, S \subseteq V(G)$, we construct the set $N(A, S)$ and the graph $G^{+}[A, S]$ as follows. Set $N(A, S) = \{v \in A\colon\ v \text{ is adjacent to a vertex in } S\}$. To construct $G^{+}[A, S]$, define it initially as a copy of $G[A]$. Then, for every $v_i \in N(A, S)$, add to $G^{+}[A, S]$ new vertices x_i, y_i and new edges $v_i x_i, x_i y_i$. For instance, regarding the example of the preceding paragraph, $N(A, S) = \{v_2, v_3\}$ and $G^{+}[A, S]$ is as shown in Figure 3.1(b).

As already stated in Section 3.1, we denote the smallest size of a clique cover of G by $\theta(G)$, and the largest size of an independent set of G by $\alpha(G)$.

3.3 A lower bound for $\gamma^{\infty}_{m,m}$

Consider a graph G, and pairwise disjoint sets $A, S, B \subseteq V(G)$ such that: (i) $A \cup S \cup B = V(G)$, (ii) for every $a \in A, b \in B$, $ab \notin E(G)$, and (iii) S is a clique. In this section, we prove the following lower bound for $\gamma^{\infty}_{m,m}(G)$:

$$\gamma^{\infty}_{m,m}(G) \geq \gamma^{\infty}_{m,m}(G[B \cup S]) + p,$$

where p is a nonnegative value later introduced in Lemma 3.1.

If A and B above are not empty, then S is a separator that is a clique. In this case, the bound can be informally described as follows: the value $\gamma^{\infty}_{m,m}$ for the graph induced by the vertices in one of the separated parts and the vertices of the separator is less or equal to the value $\gamma^{\infty}_{m,m}$ for the whole graph. We note, however, that, if S is not a clique, this is not necessarily true. Take, for example, $G = C_6$, $A = \{v_1\}$, $S = \{v_2, v_6\}$, and $B = \{v_3, v_4, v_5\}$, where $V(C_6) = \{v_1, \ldots, v_6\}$, and $v_iv_j \in E(C_6)$ if, and only if, $(i+1) \equiv j \pmod 6$. It holds that $G[B \cup S] = P_5$, the 5-path, and that $\gamma^{\infty}_{m,m}(G) = \gamma^{\infty}_{m,m}(C_6) = 2$ and $\gamma^{\infty}_{m,m}(G[B \cup S]) = \gamma^{\infty}_{m,m}(P_5) = 3$.

We begin with a helpful lemma and then arrive at the main theorem. Before stating the lemma, we notice that, for disjoint sets of vertices A and S of a graph G, $\gamma(G^+[A,S]) \geq |N(A,S)|$. This is because, for each $v_i \in N(A,S)$, there are vertices x_i and y_i in $G^+[A,S]$ and, for both x_i and y_i to be dominated, at least one of them must be in the dominating set, since y_i has degree one and is connected solely to x_i (see Figure 3.1).

Lemma 3.1. *Consider a graph G, and pairwise disjoint sets $A, S, B \subseteq V(G)$ such that $A \cup S \cup B = V(G)$, and, for every $a \in A, b \in B$, $ab \notin E(G)$. Suppose that $D \subseteq V(G)$ is a dominating set of G. If $\gamma(G^+[A,S]) \geq |N(A,S)| + p$, for some $p \geq 0$, then $|D \cap A| \geq p$.*

Proof. Suppose that $|D \cap A| = q < p$, $q \geq 0$. Define D' as the set of all vertices in $D \cap A$ along with all vertices x_i as in the construction of $G^+[A,S]$. Below, we argue that D' is a dominating set of $G^+[A,S]$. However, since $|D'| = q + |N(A,S)| < p + |N(A,S)| \leq \gamma(G^+[A,S])$, this is a contradiction.

Take a vertex $v_i \in A \setminus D'$ that is not adjacent to a vertex of $D' \cap A$ in $G^+[A,S]$. Since $(D \cap A) = (D' \cap A)$,

1. v_i is not in D and
2. v_i is not adjacent to a vertex of $D \cap A$ in G.

Because D is a dominating set of G and, for every $a \in A, b \in B$, $ab \notin E(G)$, v_i is adjacent to a vertex of $D \cap S$ in G. Thus, $v_i \in N(A,S)$. By the constructions of $G^+[A,S]$ and D', v_i is adjacent to $x_i \in D'$ in $G^+[A,S]$. Moreover, each vertex y_i as in the construction of $G^+[A,S]$ is adjacent to $x_i \in D'$. □

In the following theorem, we use the more intuitive language of guards defending a graph rather than the formal definition of an eternal dominating multiset given in Section

3.2. We also use, though, the knowledge that the set of the vertices occupied by guards must always constitute a dominating set.

Theorem 3.2. *Consider a graph G, and pairwise disjoint sets $A, S, B \subseteq V(G)$ such that $A \cup S \cup B = V(G)$, and, for every $a \in A, b \in B$, $ab \notin E(G)$. If $\gamma(G^+[A,S]) \geq |N(A,S)| + p$, for some $p \geq 0$, and S is a clique, then $\gamma^\infty_{m,m}(G) \geq \gamma^\infty_{m,m}(G[B \cup S]) + p$.*

Proof. If $S = \emptyset$, G consists of the two subgraphs $G[A]$ and $G[B]$ with no connection between them. Also, $N(A,S) = \emptyset$ and $G^+[A,S] = G[A]$. Then,

$$\begin{aligned}\gamma^\infty_{m,m}(G) &= \gamma^\infty_{m,m}(G[B]) + \gamma^\infty_{m,m}(G[A]) \geq \gamma^\infty_{m,m}(G[B \cup S]) + \gamma(G^+[A,S]) \\ &\geq \gamma^\infty_{m,m}(G[B \cup S]) + p.\end{aligned}$$

Therefore, assume that $S \neq \emptyset$.

Suppose that $\gamma^\infty_{m,m}(G) = k < \gamma^\infty_{m,m}(G[B \cup S]) + p$, $k \geq 0$. Then, k guards can eternally defend attacks at the vertices of G. Denote by C_1 the set of k such guards.

Let O be the set of the vertices of G occupied by the guards in C_1 initially or after the defense of an attack at a vertex of G. We have that O is a dominating set of G. By Lemma 3.1, $|O \cap A| \geq p$. Hence, there are, initially and after the defense of each attack, at least p of the guards in C_1 on the vertices in A. Thus, we can set $k = p + q$, $q \geq 0$; $q < \gamma^\infty_{m,m}(G[B \cup S])$.

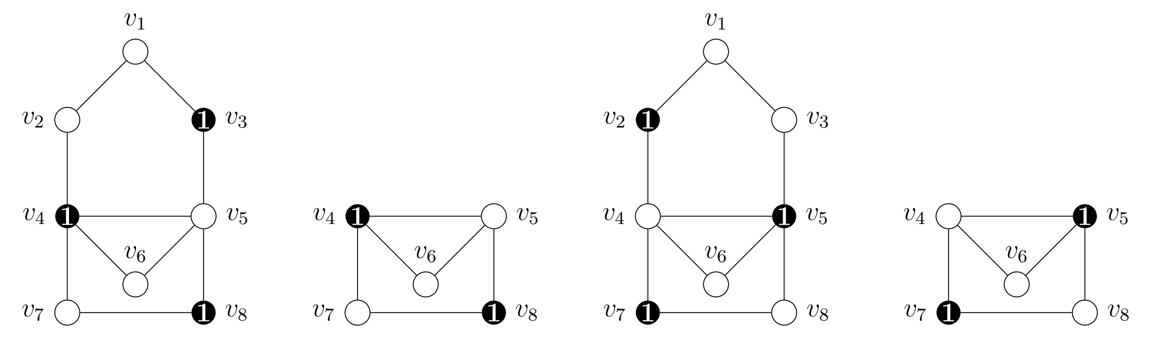
Take a sequence of attacks at the vertices of $G[B \cup S]$. This is also a sequence of attacks at the vertices of G. Thus, the k guards in C_1 moving through G can defend it. We argue that q guards moving only through $G[B \cup S]$ can defend it too; we denote by C_2 the set of these q guards. Doing so, we conclude that $\gamma^\infty_{m,m}(G[B \cup S]) \leq q$, which is a contradiction.

Consider a sequence of movements made by the guards in C_1 to defend the sequence of attacks. We describe a sequence of movements made by the guards in C_2 using the former.

Let BS_1, $|BS_1| = r$, $r \geq 0$, be the set of the guards in C_1 initially placed on the vertices in $B \cup S$ and $p + s$, $s \geq 0$, be the number of guards in C_1 initially placed on the vertices in A. Since $A \cup S \cup B = V(G)$ and A, S, and B are pairwise disjoint sets, $r + s = q$. Choose r guards in C_2 (recall that $|C_2| = q$) and call BS_2 the set of them. Set $E_2 = C_2 \setminus BS_2$ – the guards in E_2 will act like *extra guards* ($|E_2| = s$). Create a bijective function $f : BS_1 \to BS_2$ to make a correspondence between guards in the two sets. The initial position of each guard g_2 in C_2 at a vertex of $G[B \cup S]$ is as follows:

- If $g_2 \in BS_2$, place $g_2 = f(g_1)$ at the same vertex occupied by g_1;
- If $g_2 \in E_2$, place g_2 at an arbitrary vertex in S (recall that $S \neq \emptyset$).

In Figure 3.2(b), it is shown an example of the initial placement of the guards in C_2 which is based on the example of the initial placement of the guards in C_1 presented in Figure 3.2(a). For the graph in Figure 3.2(a), $\gamma^\infty_{m,m} = 3$ ($\gamma(G) = 3$, $\gamma^\infty_{m,m}(G[A \cup S]) = 2$, and $\gamma^\infty_{m,m}(G[B]) = 1$). In this case, $r = 2$, $p = 1$, $s = 0$ (there are no extra guards), and $q = 2$.



(a) Initial placement of the guards in C_1 (b) Initial placement of the guards in C_2 (c) The guards in C_1 after an attack at v_2 (d) The guards in C_2 after an attack at v_2

Figure 3.2: Placements of the guards in C_1 and in C_2 (one guard on each black vertex) on the vertices of a graph G and its subgraph $G[B \cup S]$, respectively – $A = \{v_1, v_2, v_3\}$, $S = \{v_4, v_5, v_6\}$, and $B = \{v_7, v_8\}$.

Now, consider the movements made by every guard g_1 in C_1 during the defense of an attack. Suppose that g_1 goes from a vertex u to a vertex w, $u = w$ meaning g_1 does not move at all. If $u \neq w$, g_1 goes through the edge uw. Proceed as follows considering the guards in the order below.

1. For every g_1 such that $u, w \in B \cup S$ and $u \neq w$, move the guard $g_2 = f(g_1)$ from u to w through the edge uw.
2. For every g_1 such that $u, w \in B \cup S$ and $u = w$, do nothing.
3. If $u \in B \cup S$ and $w \in A$ (thus $u \neq w$), since, for every $a \in A, b \in B$, $ab \notin E(G)$, then $u \in S$. For every such g_1, keep the guard $g_2 = f(g_1)$ positioned at u. Furthermore, set $BS_1 = BS_1 \setminus \{g_1\}$, $BS_2 = BS_2 \setminus \{g_2\}$ and $E_2 = E_2 \cup \{g_2\}$. As a consequence, g_2 is no longer associated to g_1, and becomes an extra guard.
4. If $u \in A$ and $w \in B \cup S$ (thus $u \neq w$), since, for every $a \in A, b \in B$, $ab \notin E(G)$, then $w \in S$. For every such g_1, move an arbitrary guard $g_2 \in E_2$ – an extra guard – from the vertex it occupies in S to w. Moreover, set $BS_1 = BS_1 \cup \{g_1\}$, $BS_2 = BS_2 \cup \{g_2\}$ and $E_2 = E_2 \setminus \{g_2\}$. In addition, let $g_2 = f(g_1)$. Observe that, since S is a clique, the movement of g_2 is valid.
5. For every g_1 such that $u, w \in A$, do nothing.

The existence of an extra guard g_2 in item 4 is ensured by every guard g_1 in item 3 being considered before every guard g_1 in item 4. Since, initially and after the defense of each attack, at least p of the guards in C_1 are on the vertices in A, after considering the guards in item 5, we have that $|BS_1| \leq q$. Moreover, because the only item in which $|BS_1|$ decreases (item 3) precedes the only one in which $|BS_1|$ increases (item 4), it follows that $|BS_1|$ is never greater than q. This implies that there will be enough extra guards.

The importance of the order of items 3 and 4 is illustrated with Figures 3.2(c) and 3.2(d). If an attack occurs at vertex v_2 of the graph of Figure 3.2(a), the guards in C_1 have to move as follows: the guard on v_4 goes to v_2, the guard on v_3 goes to v_5, and the guard on v_8 either goes to v_7 or stays at v_8 – the placement in which v_7 is occupied is pictured in Figure 3.2(c). In the order above, the movement of the first of these guards is considered before the movement of the second one. Because of this, the guard on v_4 in Figure 3.2(b) first becomes an extra guard and then, when an extra guard is required to occupy v_5, it goes to v_5. The resulting placement is shown in Figure 3.2(d).

We can see that, initially and after the defense of each attack, the vertices in $B \cup S$ that are occupied by the guards in C_1 are also occupied by the guards in C_2. Additionally, the guards in C_2 move only through edges and vertices of $G[B \cup S]$. Hence, our claim is proved. □

3.4 The values $\gamma^\infty_{m,m}$ and $\gamma^\infty_{m,1}$ for proper interval graphs

We now determine $\gamma^\infty_{m,m}(G)$ and $\gamma^\infty_{m,1}(G)$ for a proper interval graph G by deriving a relation that applies to both values: $\gamma^\infty_{m,m}(G) = \alpha(G)$ and $\gamma^\infty_{m,1}(G) = \alpha(G)$. Another important connection obviously follows: $\gamma^\infty_{m,m}(G) = \gamma^\infty_{m,1}(G)$.

Our first step is, using Theorem 3.2 from Section 3.3, to prove that $\gamma^\infty_{m,m}(G) = \alpha(G)$ for any proper interval graph G. We employ, in the process, a known characterization of proper interval graphs. This characterization is found in the literature in different forms. We choose to state it, in Theorem 3.3, as Figueiredo, Meidanis, and de Mello [4] do, a convenient language for us. Our result is given in Theorem 3.4.

Theorem 3.3. *A graph G is a proper interval graph if, and only if, its vertices can be linearly ordered so that, for each maximal clique Q of G, the vertices of Q occur consecutively in this ordering.*

An application of Theorem 3.3 for the proper interval graph of Figure 3.1(a) is illustrated in Figure 3.3. In this figure, each maximal clique of the graph is represented by a curve, and its vertices are the vertices between the ends of the curve, inclusive. We can see that the ordering $\langle v_1, v_2, \ldots, v_9\rangle$ satisfies the condition stated in the theorem.

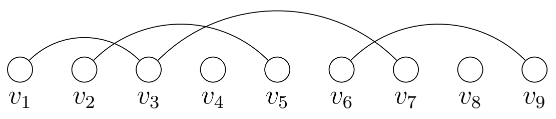


Figure 3.3: Application of Theorem 3.3 for the proper interval graph of Figure 3.1(a).

Theorem 3.4. *If G is a proper interval graph, then $\gamma^\infty_{m,m}(G) = \alpha(G)$.*

Proof. The proof is by induction on $|V(G)|$. The result is obviously valid for a graph with no vertices ($\gamma^\infty_{m,m} = \alpha = 0$). Suppose that it also holds for all proper interval graphs with less than n vertices and take G as a proper interval graph on $n \geq 1$ vertices.

Since G is a proper interval graph, Theorem 3.3 holds for G. Let $V(G) = \{v_1,\ v_2,\ \ldots,\ v_n\}$ and consider an ordering $\langle v_1, v_2, \ldots, v_n\rangle$ of the vertices of G as in the theorem. Notice that there cannot be two distinct maximal cliques $Q' = \{v_i, v_{i+1}, \ldots, v_{i+p}\}$ and $Q'' = \{v_j, v_{j+1}, \ldots, v_{j+q}\}$ of G such that $i = j$ or $(i + p) = (j + q)$. Otherwise, $Q' \subset Q''$ or $Q'' \subset Q'$ – either way, a contradiction. See the illustration in Figure 3.3.

Now, take the maximal clique $Q_1 = \{v_1, v_2, \ldots, v_k\}$ (Q_1 is the only maximal clique containing v_1). Consider, also, among all maximal cliques of G intersecting Q_1, the maximal clique $Q_2 = \{v_i, v_{i+1}, \ldots, v_{i+p}\}$ such that $(i+p)$ is the largest possible index – in Figure 3.3, $Q_1 = \{v_1, v_2, v_3\}$, and $Q_2 = \{v_3, v_4, \ldots, v_7\}$. Note that, because, for every edge uv of a graph, u and v are together in one of its maximal cliques, the following holds:

1. v_1 has no neighbor outside Q_1;
2. All neighbors of the vertices of Q_1 that are not in Q_1 are in Q_2.

Let G_1 denote $G[V(G) \setminus Q_1]$. Considering v_1, $Q_1 \setminus \{v_1\}$ and $Q_2 \setminus Q_1$, we have the situation pictured in Figure 3.4. Since every induced subgraph of a proper interval graph is a proper interval graph, G_1 is a proper interval graph. Hence, by induction hypothesis, $\gamma^{\infty}_{m,m}(G_1) = \alpha(G_1)$.

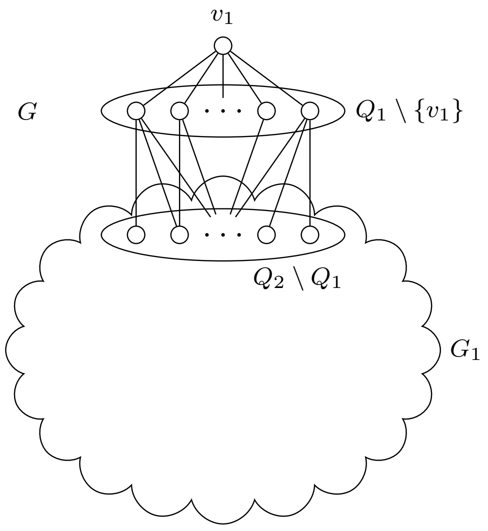


Figure 3.4: A proper interval graph G and the subgraph G_1.

To prove that $\gamma^{\infty}_{m,m}(G) = \alpha(G)$, we first show that $\alpha(G) = \alpha(G_1) + 1$. Suppose that there is an independent set I of G such that $|I| > \alpha(G_1) + 1$. Since Q_1 is a clique, $|I \cap Q_1| \leq 1$. Then, the set $I' = I \setminus Q_1$ is an independent set of G_1 with $|I'| > \alpha(G_1)$, which is a contradiction. Thus, $\alpha(G) \leq \alpha(G_1) + 1$. Additionally, by the item 1 above, we can construct an independent set of G of size $\alpha(G_1) + 1$ by adding v_1 to a independent set of G_1 of largest size. Hence, our claim is true.

Secondly, we argue that $\gamma^{\infty}_{m,m}(G) = \gamma^{\infty}_{m,m}(G_1) + 1$. Let $A = Q_1$, $S = Q_2 \setminus Q_1$, and $B = V(G) \setminus (Q_1 \cup Q_2)$. Note that $A, S, B \subseteq V(G)$ are pairwise disjoint sets such that $A \cup S \cup B = V(G)$. By the item 2 above, for every $a \in A, b \in B$, $ab \notin E(G)$. By the item 1 above and because A is a clique, $\gamma(G^+[A, S]) = |N(A, S)| + 1$. Given S is a clique, Theorem 3.2 implies that $\gamma^{\infty}_{m,m}(G) \geq \gamma^{\infty}_{m,m}(G[B \cup S]) + 1$. Since $B \cup S = V(G) \setminus Q_1$, it follows that $\gamma^{\infty}_{m,m}(G) \geq \gamma^{\infty}_{m,m}(G_1) + 1$.

We can also see that $\gamma^{\infty}_{m,m}(G_1) + 1$ guards can eternally defend attacks at the vertices of G: it is possible to maintain $\gamma^{\infty}_{m,m}(G_1)$ guards defending the attacks at the vertices of G_1 and one guard independently defending the attacks at the vertices in Q_1. Thus, $\gamma^{\infty}_{m,m}(G) = \gamma^{\infty}_{m,m}(G_1) + 1$.

Therefore, $\gamma^{\infty}_{m,m}(G) = \gamma^{\infty}_{m,m}(G_1) + 1 = \alpha(G_1) + 1 = \alpha(G)$, and the theorem is proved. $\square$

As our second move, we combine the inequality on the right in (3.1) – Section 3.2, Theorem 3.5, by Goddard et al. [6], and the theorem above to prove that, for any proper interval graph G, $\gamma^{\infty}_{m,1}(G) = \alpha(G)$ and $\gamma^{\infty}_{m,m}(G) = \gamma^{\infty}_{m,1}(G)$. In the sequel, we reproduce the theorem by Goddard et al. [6] and present our outcome.

Theorem 3.5 (Goddard et al. [6, Theorem 13])**.** *For any graph G, $\gamma^{\infty}_{m,1}(G) \leq \alpha(G)$.*

Theorem 3.6. *If G is a proper interval graph, then $\gamma^{\infty}_{m,m}(G) = \gamma^{\infty}_{m,1}(G) = \alpha(G)$.*

Proof. By the inequality on the right in (3.1) and Theorem 3.5, we have the following inequality chain:

$$\gamma^{\infty}_{m,m}(G) \leq \gamma^{\infty}_{m,1}(G) \leq \alpha(G).$$

By Theorem 3.4, it collapses and we reach our goal. $\square$

Notice that Theorem 3.6 does not hold for interval graphs in general. For example, if $K_{1,k}$ is the star with k leaves and $k \geq 3$, we have that $\gamma^{\infty}_{m,1}(K_{1,k}) = 2 < \alpha(K_{1,k}) = k$. It is also worth noting that there are graphs which are not proper interval graphs and for which $\gamma^{\infty}_{m,1} = \alpha$. This is the case for the graph G resulting from removing edge v_2x_2 and adding edge v_3x_2 in the graph of Figure 3.1(b) (observe that G is an interval graph, although not a proper one). Moreover, there are graphs that are not interval graphs for which $\gamma^{\infty}_{m,1} = \alpha$. The 4-cycle C_4 is an instance where the latter equation holds.

Finally, we present two important implications of Theorem 3.6. First, since α can be determined in linear time for proper interval graphs [7], the EDSP can be solved in linear time for such graphs. Second, because any proper interval graph G is a perfect graph, we have that $\gamma^{\infty}_{m,1}(G) = \theta(G)$. This provides us with a straightforward defense strategy [6] for the smallest possible number of guards. As the vertices of G can be partitioned into $\theta(G)$ cliques, it is enough to have one guard to defend the vertices of each of them.

Acknowledgements

The authors would like to thank Professor Célia P. de Mello for suggesting the investigation of the EDSP for proper interval graphs, and the anonymous reviewers for their careful reading of the paper and valuable comments.

This work was funded by grants 477692/2012-5, 141964/2013-8, 302804/ 2010-2, and 303947/2008-0 from *Conselho Nacional de Desenvolvimento Científico e Tecnológico* (CNPq, Brazil).

Bibliography

[1] A. Bondy and U. Murty. *Graph Theory.* Graduate Texts in Mathematics. Springer, 2008.

[2] A. Burger, E. Cockayne, W. Gründlingh, C. Mynhardt, J. van Vuuren, and W. Winterbach. Infinite order domination in graphs. *Journal of Combinatorial Mathematics and Combinatorial Computing*, 50:179–194, 2004.

[3] E. Chambers, B. Kinnersley, and N. Prince. Mobile eternal security in graphs. Unpublished. [Online; accessed 21-August-2014]. Available at: `http://staff.imsa.edu/~nprince/researchfiles/MES.pdf`, 2006.

[4] C. de Figueiredo, J. Meidanis, and C. de Mello. A linear-time algorithm for proper interval graph recognition. *Information Processing Letters*, 56(3):179–184, 1995.

[5] S. Finbow, S. Gaspers, M.-E. Messinger, and P. Ottaway. Eternal domination. Unpublished. Cited in [9], 2010.

[6] W. Goddard, S. Hedetniemi, and S. Hedetniemi. Eternal security in graphs. *Journal of Combinatorial Mathematics and Combinatorial Computing*, 52:169–180, 2005.

[7] M. Golumbic. *Algorithmic Graph Theory and Perfect Graphs: Second Edition*. Annals of Discrete Mathematics. Elsevier, 2004.

[8] W. Klostermeyer and G. MacGillivray. Eternal dominating sets in graphs. *Journal of Combinatorial Mathematics and Combinatorial Computing*, 68:97–111, 2009.

[9] W. Klostermeyer and C. Mynhardt. Protecting a graph with mobile guards. *arXiv e-prints*, 2014. arXiv:1407.5228v1.

[10] F. Regan. *Dynamic Variants of Domination and Independence in Graphs*. Graduate thesis, Rheinischen Friedrich-Wilhelms University, Bonn, 2007.

Chapter 4

Practical Standpoint for m-Eternal Domination

Next, our attention is turned to heuristic methods suited for practical applications of the m-eternal dominating set problem. We propose heuristic methods that run in a reasonable time, produce a good-quality upper bound k on the m-eternal domination number, and output a structure from which one can derive an efficient strategy that k guards can follow to eternally defend the input graph. These features are validated through extensive experimentation. As far as we know, neither implementations gathering such features nor experiments reaching such extension have been reported before for the m-eternal dominating set problem.

In the methods above, we employ integer and constraint programming models to compute several bounds on the m-eternal domination number. One of these models is, based on our information, the first integer program for the problem of determining a minimum-weight neocolonization of a graph. By performing the experimental study of this work, we generate a 750-instance benchmark and carry out a practical evaluation of bounds for the m-eternal domination number available in the literature. As a contribution to the knowledge on the complexity of the m-eternal dominating set problem, we prove that its decision version is NP-hard.

The subsequent text is a reproduction of a paper submitted to *International Transactions in Operational Research* and co-authored by Márcio F. Reis[1], Cid C. de Souza[1], and Orlando Lee[1]. A preliminary version of this paper was presented at the *13th Cologne-Twente Workshop on Graphs & Combinatorial Optimization* [6]. Our exact algorithm for the m-eternal dominating set problem (which is mentioned in Section 1.3) is described in Section 4.4.4.

Abstract

We address an issue that is fundamental to practical applications of the *m-eternal dominating set problem* but that has received relatively little attention. Our goal is to

[1]Institute of Computing, University of Campinas, Brazil.

obtain not only a good-quality upper bound on the *m-eternal domination number* but also an associated *efficient strategy of defense*. To this end, we introduce two heuristic methods, in which we propose and solve integer and constraint programming models to compute bounds on the m-eternal domination number. We perform an extensive experiment to validate the features of these methods; as a consequence, a benchmark of 750 instances is generated. Finally, we prove that the decision version of the m-eternal dominating set problem is NP-hard.

4.1 Introduction

In a cyber-physical system (CPS), computational elements control physical processes in a highly integrated manner. Examples of CPSs are modern electric power, water distribution, transportation, and health-care systems [23]. As CPSs scale, they get more vulnerable to attacks, which range from data measurement failures to cyber or physical attacks. Since such systems are increasingly larger and crucial to our society, we are progressively in need of efficient surveillance mechanisms for them.

The efficient surveillance of a CPS may be viewed as a practical application of graph protection, a class of problems that includes the following one (for a thorough survey on graph protection, see the work of Klostermeyer and Mynhardt [27]). Place guards on some vertices of a graph G. An attack to G occurs at a vertex, and, to defend an attack at v, the guards must move so that one guard occupies v. In this movement, each guard can stay at the same vertex or go to an adjacent one. At all times, there must be at most one guard per vertex. We are asked to determine the minimum number of guards that can *eternally defend* G, namely, that can eternally perform the task of defending any attack to G. This number is denoted by $\gamma_m^\infty(G)$.

The problem above is called the *m-eternal dominating set problem* (m-EDSP) – alternatively, the *m-eternal domination problem* – and has its origin back in Constantine's approach to defend the Roman Empire, known as Roman domination [27]. Accordingly, $\gamma_m^\infty(G)$ is denominated the *m-eternal domination number* of G. Consider guards *eternally defending* G, that is, eternally performing the task of defending any attack to G. Note that these guards must occupy, initially and after responding to each attack, a dominating set of G (a set S of vertices of G such that every vertex of G that is not in S has a neighbor in S). The problem's name captures the idea of a set of vertices that can be eternally modified yet remaining a dominating set (the prefix "m" distinguishes the problem from a version in which at most one guard can move during the defense of an attack [27]). Later, we define in terms of dominating sets a *strategy of defense* of G for k guards, which is a strategy that k guards can follow to eternally defend G.

In this paper, we address an issue that is fundamental to practical applications of the m-EDSP but that has been given relatively little attention in the literature. Works on the m-EDSP usually provide bounds on $\gamma_m^\infty(G)$ for any graph G [8, 18, 25, 26] or determine bounds on or even find the exact value of $\gamma_m^\infty(G)$ for G restricted to specific classes of graphs [7, 10, 18, 19, 24]. The issue we deal with is to produce not only a good-quality upper bound, say of value k, on $\gamma_m^\infty(G)$ for a generic graph G but also an *efficient* strategy of defense of G for k guards – we translate such a strategy of defense into a definition with polynomial-time and polynomial-space requirements. To our knowledge, a similar goal was only pursued by Abbas et al. [1].

As our main contribution, we propose two heuristic methods suited for practical applications of the m-EDSP. The methods use integer and constraint programming techniques to compute bounds on $\gamma_m^\infty(G)$ for an input graph G. We show that they

(i) run in a reasonable time,

(ii) produce a good-quality upper bound, say of value k, on $\gamma_m^\infty(G)$, and

(iii) output a structure from which one can derive an efficient strategy of defense of G for k guards,

with the first two features being validated through extensive experimentation. Based on our information, neither implementations gathering such features nor experiments reaching such extension have been reported before for the m-EDSP.

As a secondary contribution, we add to the knowledge on the complexity of the m-EDSP: we show that its decision version – where one asks if $\gamma_m^\infty(G) \leq k$ for a graph G and a positive integer k – is NP-hard. Similar problems, given by two-player combinatorial games, have been proven PSPACE-hard [12, 13, 14, 28]. If the m-EDSP is also proven PSPACE-hard, then its decision version will unlikely be in NP and our result will likely establish the closest relationship between the problem and the NP class.

The remainder of this paper is organized as follows. In the next section, we introduce assumptions and definitions considered throughout the text and prove the complexity result above. In Section 4.3, we provide demonstrations used to attest Feature (iii) and to explain the correctness of upper bounds employed in the heuristic methods proposed for the m-EDSP. We describe these methods in Sections 4.4 and 4.6, present computational results for them in Sections 4.5 and 4.7, and discuss their limitations in Section 4.8. Finally, in Section 4.9, we draw conclusions and comment on future work.

4.2 Preliminaries

All non-basic concepts used in this text without being specified are standard (for the respective mathematical programming and graph-related definitions, see the books by Wolsey [32] and West [31]). All graphs that we deal with are finite and, except for the ones constructed in Section 4.4.4, simple. The graphs not stated as directed are undirected. Throughout this paper, we employ the following terminology and notation, for which we assume a graph G and a directed graph H. The key definitions are illustrated at the end of the section.

Each of the symbols $|S|$ for a set S; $V(G)$ and $E(G)$; $G[S]$ for $S \subseteq V(G)$; $d(v)$, $N(v)$, and $N[v]$ for a vertex v of G; $V(H)$; and $N^+(v)$ and $N^-(v)$ for a vertex v of H has the same meaning as in the notation used by West [31]. For $S \subseteq V(G)$, we set $N(S) = \bigcup_{v \in S} N(v)$ and $N[S] = \bigcup_{v \in S} N[v]$. A neighborhood symbol is written with a subscript (e.g., $N_G(v)$ and $N_H^+(v)$) when the underlying graph needs to be explicitly stated. We denote the minimum size of a dominating set of G by $\gamma(G)$ and, assuming that G is connected, the minimum size of a connected dominating set of G by $\gamma_c(G)$.

Let a partition Z of $V(G)$, say $Z = \{\, V_1, V_2, \ldots, V_k \,\}$, be such that, for each $V_i \in Z$, $G[V_i]$ is connected. Consider an ordered pair $\langle Z, Y\rangle$ having Y as a set composed of a connected dominating set of $G[V_i]$ for every $V_i \in Z$ that is not a clique. We call Z a *neocolonization* of G and $\langle Z, Y\rangle$ a *neocolonization setup* of G; we denote (in an abuse of notation) the element of Y corresponding to $V_i \in Z$ by $Y(V_i)$. The *weight* $w(Z)$ of Z is given by $\sum_{V_i \in Z} w(V_i)$, where $w(V_i) = 1$ if V_i is a clique and $w(V_i) = \gamma_c(G[V_i]) + 1$ otherwise. The *weight* $w(\langle Z, Y\rangle)$ of $\langle Z, Y\rangle$ is given by the sum of 1 for every clique part of Z and of $\min\{|Y(V_i)| + 1, |V_i|\}$ for each non-clique part V_i of Z. Notice that $w(\langle Z, Y\rangle) \geq w(Z)$. We denote by $\theta_C(G)$ the minimum weight of a neocolonization setup of G – equivalently, the minimum weight of a neocolonization of G.

Assume that $\mathcal{D}$ is the set of all dominating sets of G. We say that $D_i \in \mathcal{D}$ *shifts to* $D_j \in \mathcal{D}$ when there exists a bijective function $f\colon D_i \to D_j$ such that, if $f(u) = v$, then $u = v$ or $uv \in E(G)$; f is called a *shift* from D_i to D_j (note that f exists only if $|D_i| = |D_j|$). For $D_i \in \mathcal{D}$ and $v \in V(G)$, we define $\mathcal{N}(D_i, v) = \{\, D_j \in \mathcal{D} : v \in D_j$ and D_i shifts to $D_j \,\}$. Consider guards placed on the vertices of $D_i \in \mathcal{D}$. A shift from D_i to $D_j \in \mathcal{D}$ models these guards moving to the vertices of D_j to defend an attack to G. Moreover, $\mathcal{N}(D_i, v)$ gives all dominating sets that these guards may occupy after responding to an attack at $v \in V(G)$.

Suppose a nonempty collection $\mathcal{C}$ of k-size dominating sets of G such that, for every $D_i \in \mathcal{C}$ and every $v \in V(G)$, $\mathcal{N}(D_i, v) \cap \mathcal{C} \neq \emptyset$. Take M to be a method that, given $D_i \in \mathcal{C}$ and $v \in V(G)$, returns $D_j \in \mathcal{N}(D_i, v) \cap \mathcal{C}$ and a shift from D_i to D_j. Let I_M be the input data for M. The existence of $\mathcal{C}$ implies $\gamma_m^\infty(G) \leq k$: k guards can eternally defend G by starting placed on the vertices of a dominating set $D^0 \in \mathcal{C}$ and responding to an i-th attack, say at v^i, by moving from $D^{i-1} \in \mathcal{C}$ to $D^i \in \mathcal{N}(D^{i-1}, v^i) \cap \mathcal{C}$ according to a shift f^i. The triple $\langle \mathcal{C}, M, I_M\rangle$ specifies such a defense: D^i and f^i are the dominating set and the shift returned by M for D^{i-1} and v^i. We refer to $\mathcal{C}$ as a *k,m-eternal-dominating-set collection* (k,m-EDSC) – when convenient, we write m-EDSC consisting of k-size dominating sets or just m-EDSC – of G, to M as a *defense method* of $\mathcal{C}$, and to $\langle \mathcal{C}, M, I_M\rangle$ as a *strategy of defense* of G for k guards. We also refer to $\langle \mathcal{C}, M, I_M\rangle$ as an *efficient strategy of defense* of G for k guards if, with respect to $|V(G)|$,

1. $\mathcal{C}$ is composed of a polynomial number of dominating sets,
2. I_M can be represented in polynomial length, and
3. M can be executed in polynomial time.

Consider k guards eternally defending G and all dominating sets that they may occupy initially and after responding to an attack. These dominating sets form a k,m-EDSC of G. Therefore, $\gamma_m^\infty(G) \leq k$ implies that G has a k,m-EDSC. This implication and the converse one above culminate in the following theorem.

Theorem 4.1. *For a graph G, $\gamma_m^\infty(G) \leq k$ if and only if G has a k,m-EDSC.*

Next, we prove that the decision version of the m-EDSP is NP-hard. To this end, we employ an inequality established by Goddard et al. [18]: $\gamma_m^\infty(G) \leq \gamma_c(G) + 1$ for any connected graph G.

Theorem 4.2. *The decision version of the m-EDSP is NP-hard.*

Proof. We provide a polynomial-time reduction from a classical NP-complete problem: the vertex cover problem [17]. Call G the input graph of this problem, and build an extension G' of G as follows. Add new vertices u, w, and y and new edges uw, uy, and, for each $v \in V(G)$, uv. Moreover, for every $v_iv_j \in E(G)$, add a new vertex x_{ij} and new edges v_ix_{ij} and v_jx_{ij}; denote by X the set of all vertices x_{ij}. For a positive integer r, we show that G has a vertex cover of size at most r if and only if $\gamma_m^\infty(G') \leq r + 2$.

Suppose that Q is a vertex cover of G of size at most r. The set $Q \cup \{u\}$ is a connected dominating set of G' of size at most $r + 1$. Hence, by the aforementioned inequality, $\gamma_m^\infty(G') \leq r + 2$. Now, consider $r + 2$ guards eternally defending G'. After the defense of an attack at w, there must be a guard on w and a guard on y or u. Thus, $V(G') \setminus \{u, w, y\}$ has at most r occupied vertices; let S be the set of these vertices. Notice that every vertex of X must be dominated by a vertex of S. As a consequence, we can derive a vertex cover Q of G from S with $|Q| \leq |S| \leq r$. □

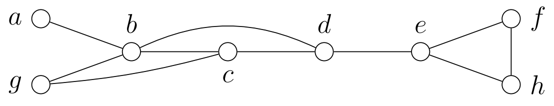


Figure 4.1: A graph having a neocolonization of weight 3, a neocolonization setup of weight 4, and a 5,m-EDSC.

Examples. As stated earlier, we illustrate the key definitions given in this section. Let G be the graph of Figure 4.1, $V_1 = \{a,\ b,\ c,\ d,\ g\}$, $V_2 = \{e,\ f,\ h\}$, $D = \{b,\ c\}$, and, for $v \in \{a,\ f,\ g,\ h\}$, $D_v = \{b,\ c,\ d,\ e,\ v\}$. We form a neocolonization Z and a neocolonization setup $\langle Z, Y\rangle$ of G by setting $Z = \{V_1,\ V_2\}$ and $Y = \{D\}$ – the set V_1 is the sole non-clique part of Z and $Y(V_1) = D$. The weights $w(Z)$ and $w(\langle Z, Y\rangle)$ are 3 and 4 respectively. Guards placed on the dominating set D_a may defend an attack at f by moving along the path $ab \ldots f$ to reach the dominating set D_f. This movement is described by the shift $m\colon D_a \to D_f$ specified as $m(a) = b$, $m(b) = c$, $m(c) = d$, $m(d) = e$, and $m(e) = f$. Note that $D_f \in \mathcal{N}(D_a, f)$. The collection $\{D_a,\ D_f,\ D_g,\ D_h\}$ is a 5,m-EDSC of G. Thus, by Theorem 4.1, $\gamma_m^\infty(G) \leq 5$.

4.3 Neocolonization-setup efficient strategy of defense and bound

We rely on the same demonstrations to provide the following insights into each m-eternal domination number upper bound used in the proposed heuristic methods: how to derive an associated efficient strategy of defense and why the bound is valid. These demonstrations are given below.

Theorem 4.3. *From a neocolonization setup $\langle Z, Y\rangle$ of a graph G, one can derive an efficient strategy of defense $\langle \mathcal{C}, M, I_M\rangle$ of G for $w(\langle Z, Y\rangle)$ guards.*

Proof. Transform $\langle Z, Y\rangle$ as follows assuming $Z = \{V_1, V_2, \ldots, V_k\}$. For each non-clique part V_r of Z such that $Y(V_r) = V_r$, find a spanning tree T of $G[V_r]$ and set $Y(V_r) = Y(V_r) \setminus \{v\}$, where v is a leaf of T; note that $Y(V_r)$ remains a connected dominating set of $G[V_r]$. The resulting neocolonization setup weighs the same as before and has $Y(V_r) \subsetneq V_r$ for every non-clique part V_r of Z.

Let S be the set formed of a vertex $s_r \in V_r$ for each clique part V_r of Z and of all vertices of the connected dominating set $Y(V_r)$ along with a vertex $s_r \in V_r \setminus Y(V_r)$ for every non-clique part V_r of Z. For $v \in V(G) \setminus S$, define $S_v = S \setminus \{s_r\} \cup \{v\}$ if v belongs to $V_r \in Z$. Because $|S| = w(\langle Z, Y\rangle)$, Z is a partition of $V(G)$, and, for each $V_r \in Z$, $S \cap V_r$ is a dominating set of $G[V_r]$, S is a $w(\langle Z, Y\rangle)$-size dominating set of G. Analogously, for every $v \in V(G) \setminus S$, S_v is a $w(\langle Z, Y\rangle)$-size dominating set of G. Set $\mathcal{C} = \{S\} \cup \{S_v : v \in V(G) \setminus S\}$ – see the end of the proof for an instance of $\mathcal{C}$.

Algorithm 4.1 Method $M^{\dagger}$ (a defense method of an m-EDSC)

† This method is used in the proof of Theorem 4.3, where its input items are defined.

```
Input: $G$; $\langle Z, Y\rangle$; $\mathcal{C}$; for each $V_r \in Z$, $s_r$; a dominating set $D_i \in \mathcal{C}$; and a vertex $x \in V(G)$.
Output: A dominating set $D_j \in \mathcal{N}(D_i, x) \cap \mathcal{C}$ and a shift $f$ from $D_i$ to $D_j$.
1: If $x \in D_i$, then set $D_j = D_i$ and $f(t) = t$, $\forall t \in D_i$; output $D_j$ and $f$; and stop.
2: Determine $V_p \in Z$ with $x \in V_p$ and $w$ with $D_i \cap V_p \setminus (S \setminus \{s_p\}) = \{w\}$, and set $x' = x$.
3: If $V_p$ is a clique, then set $f(w) = x'$.
4: Else:
5:     Compute a path $wu_1 \ldots u_m x'$ of $G[V_p]$ with $u_1, \ldots, u_m \in Y(V_p)$.
6:     Set $f(w) = u_1$, $f(u_l) = u_{l+1}$ for $l \in \{1, \ldots, m-1\}$, and $f(u_m) = x'$.
7: If $D_i = S_v$ with $v$ in $V_q \in Z$ and $V_p \neq V_q$, then set $x' = s_q$, $V_p = V_q$, and $w = v$, and go to 3.
8: Set $f(t) = t$ for each unmapped $t \in D_i$ and $D_j = S$ if $x \in S$ and $D_j = S_x$ otherwise.
9: Output $D_j$ and $f$.
```

Define M as in Algorithm 4.1, which corresponds to the following guard approach – see below for examples on this approach. Place guards on the vertices of a dominating set of $\mathcal{C}$. Next, these guards defend an attack at $x \in V(G)$ by performing at most two movements and reaching a dominating set of $\mathcal{C}$ that contains x.

If x is occupied, then the guards do not move. Otherwise, the first movement occurs. Let x belong to $V_p \in Z$. Since the guards start on a dominating set of $\mathcal{C}$, they occupy exactly one vertex of V_p if V_p is a clique and of $V_p \setminus Y(V_p)$ otherwise; call w this vertex. Also, the guards occupy all vertices of $Y(V_p)$ in case V_p is not a clique. In this case, because $Y(V_p)$ is a connected dominating set of $G[V_p]$, there is a path P of $G[V_p]$ with $P = wu_1 \ldots u_m x$, $u_1, \ldots, u_m \in Y(V_p)$, and all vertices of P but x occupied. If V_p is a clique, then the guard on w moves to x; else, each guard on P moves one vertex further towards x along the path.

The following three cases give the dominating set the guards occupy after executing the first movement – note that, by the description above, $w, x \in V_p$; since, initially, x is unoccupied and all vertices of $Y(V_p)$ are occupied if V_p is not a clique, $x \in S$ implies $x = s_p$; and $x \neq w$. If they start at S, then $w = s_p$, $x \notin S$, and they finish at S_x. If they begin at S_v for $v \in V(G) \setminus S$ such that $v \in V_p$, then $w = v$ and they end at S when $x \in S$ (thus $x = s_p$) and at S_x otherwise. If they initiate at S_v for $v \in V(G) \setminus S$ with v in $V_q \in Z$ and $V_p \neq V_q$, then $w = s_p$, $x \notin S$, and they terminate at $S \setminus \{ s_p, s_q \} \cup \{ x, v \}$. Only in the last case, where the ending dominating set is not in $\mathcal{C}$, the guards do the second movement, which is the same as the first except for x, V_p, and w replaced with s_q, V_q, and v respectively. By this movement, the guards reach S_x.

The output of M implies that, for every $D_i \in \mathcal{C}$ and every $x \in V(G)$, $\mathcal{N}(D_i, x) \cap \mathcal{C} \neq \emptyset$. Hence, $\mathcal{C}$ is an m-EDSC of G consisting of $w(\langle Z, Y\rangle)$-size dominating sets; the method M is a defense method of $\mathcal{C}$. The size of $\mathcal{C}$ equals $(1 + |V(G) \setminus S|) \in O(|V(G)|)$, the input I_M of M can be represented in $O(|V(G)|^2)$ length, and each step of M runs at most twice and can be executed in $O(|V(G)|^2)$ time. Therefore, $\langle \mathcal{C}, M, I_M\rangle$ is an efficient strategy of defense of G for $w(\langle Z, Y\rangle)$ guards.

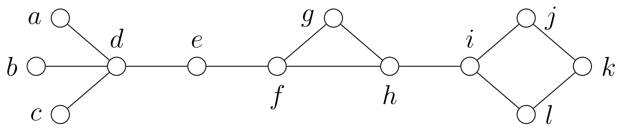


Figure 4.2: A graph having an efficient strategy of defense for 6 guards that is derived from a neocolonization setup.

Examples. In the argumentation above, assume that G is the graph of Figure 4.2. Moreover, consider that $\langle Z, Y\rangle$ is the neocolonization setup $\langle \{ V_1, V_2, V_3 \}, \{ D', D'' \} \rangle$, where $V_1 = \{ a, b, c, d, e \}$, $V_2 = \{ f, g, h \}$, $V_3 = \{ i, j, k, l \}$, $D' = \{ d \}$, and $D'' = \{ i, j \}$ – it holds that $Y(V_1) = D'$, $Y(V_3) = D''$, and $w(\langle Z, Y\rangle) = 6$. Finally, suppose that $s_1 = a$, $s_2 = g$, and $s_3 = l$.

As a consequence, $\mathcal{C}$ consists of the seven 6-size dominating sets of G given by $S = \{ a, d, g, i, j, l \}$, $S_b = S \setminus \{ a \} \cup \{ b \}$, $S_c = S \setminus \{ a \} \cup \{ c \}$, $S_e = S \setminus \{ a \} \cup \{ e \}$, $S_f = S \setminus \{ g \} \cup \{ f \}$, $S_h = S \setminus \{ g \} \cup \{ h \}$, and $S_k = S \setminus \{ l \} \cup \{ k \}$. In addition, the guard approach corresponding to M works as in the following examples; we write D_i and D_j, respectively, for the initial and final placements of the guards. If $D_i = S$ and $x = f$, then the guard on g moves to f and $D_j = S_f$ ($V_p = V_2$ and $w = g$). If $D_i = S$ and $x = k$, then each guard on path $lijk$ moves towards k and $D_j = S_k$ ($V_p = V_3$ and $w = l$). If $D_i = S_b$ and $x = a$, then each guard on path bda moves towards a and $D_j = S$ ($V_p = V_1$ and $w = b$). If $D_i = S_b$ and $x = c$, then each guard on path bdc moves towards c and $D_j = S_c$ ($V_p = V_1$ and $w = b$). If $D_i = S_b$ and $x = k$, then each guard on path $lijk$ moves towards k, each guard on path bda moves towards a, and $D_j = S_k$ ($V_p = V_3$, $w = l$, $V_q = V_1$, $v = b$, and $s_q = a$). □

Theorem 4.4. *For a neocolonization setup $\langle Z, Y\rangle$ of a graph G, $\gamma_m^\infty(G) \leq w(\langle Z, Y\rangle)$.*

Proof. For simplicity, let $w = w(\langle Z, Y\rangle)$. By Theorem 4.3, there is a strategy of defense $\langle \mathcal{C}, M, I_M\rangle$ of G for w guards. Because $\mathcal{C}$ is a w,m-EDSC of G, Theorem 4.1 implies $\gamma_m^\infty(G) \leq w$. □

4.4 First heuristic method

Our first heuristic method (Heur 1) consists in computing several upper and lower bounds on $\gamma_m^\infty(G_{\text{H1}})$ for an input graph G_{H1}. The method is outlined in Algorithm 4.2 : below, we define the bounds in the list of Step 2 and detail their computation.

Algorithm 4.2 First heuristic method

```
Input: A connected graph $G_{\text{H1}}$.
Output: An upper bound H1-UB and a lower bound H1-LB on $\gamma_m^\infty(G_{\text{H1}})$ and a structure H1-S.
1: Set $u = \infty$ and $l = -\infty$.
2: For each bound $B$ on $\gamma_m^\infty(G_{\text{H1}})$ in the list UB-MCD, LB-MDS, LB-MMD, UB-DNS, LB-TLE, do:
3:     Compute $B$ and, in the upper bound case, obtain a by-product $p$; call $b$ the value of $B$.
4:     If $B$ is an upper bound on $\gamma_m^\infty(G_{\text{H1}})$ and $b < u$, then set $u = b$ and $s = p$.
5:     If $B$ is a lower bound on $\gamma_m^\infty(G_{\text{H1}})$ and $b > l$, then set $l = b$.
6:     If $u = l$, then go to 7.
7: Set H1-UB = $u$, H1-LB = $l$, and H1-S = $s$.
8: Output H1-UB, H1-LB, and H1-S.
```

In what follows, we also give an approach to derive, from the structure H1-S outputted by Heur 1, an efficient strategy of defense of G_{H1} for H1-UB guards. This approach is presented in two cases, at Sections 4.4.1 and 4.4.3, corresponding to the two possible assignments to H1-S.

For a graph G, since guards can only move to neighboring vertices during the defense of an attack to G, it holds that $\gamma_m^\infty(G) = \gamma_m^\infty(C_1) + \gamma_m^\infty(C_2) + \ldots + \gamma_m^\infty(C_k)$, where C_1, C_2, …, C_k are the components of G. Hence, one can assume, without loss of generality, that the input graph to the m-EDSP is connected. We consider this assumption for Heur 1, as stated in the input to Algorithm 4.2.

4.4.1 Minimum connected dominating set upper bound

The minimum connected dominating set upper bound (UB-MCD) is given by $\gamma_c(G_{\text{H1}}) + 1$. A result by Goddard et al. [18, Theorem 14] stated earlier (for Theorem 4.2) implies $\gamma_m^\infty(G_{\text{H1}}) \leq \gamma_c(G_{\text{H1}}) + 1$. The following is an alternative proof to this inequality. From a minimum connected dominating set D of G_{H1}, derive the neocolonization setup $\langle Z_D, Y_D\rangle$, where $Z_D = \{ N[D] \} = \{ V(G_{\text{H1}}) \}$ and $Y_D = \emptyset$ if $V(G_{\text{H1}})$ is a clique and $Y_D = \{ D \}$ if not. By Theorem 4.4, $\gamma_m^\infty(G_{\text{H1}}) \leq w(\langle Z_D, Y_D\rangle)$. Since $w(\langle Z_D, Y_D\rangle) = 1 = \gamma_c(G_{\text{H1}})$ in case $V(G_{\text{H1}})$ is a clique and $w(\langle Z_D, Y_D\rangle) = \gamma_c(G_{\text{H1}}) + 1$ otherwise, $\gamma_m^\infty(G_{\text{H1}}) \leq \gamma_c(G_{\text{H1}}) + 1$. We refer to $\langle Z_D, Y_D\rangle$ as *the neocolonization setup associated with* UB-MCD.

To compute UB-MCD, we determine as follows a minimum connected dominating set D of G_{H1}. First, we find a maximum-leaf spanning tree T of G_{H1} using a flow-based mixed integer program proposed by Reis et al. [30] – this program provides a competitive and simple-to-implement exact method for the problem of finding a maximum-leaf spanning tree of a connected graph. Next, we choose D to consist of all non-leaf vertices of T if $|V(T)| \neq 2$ and to contain any of the two vertices of T otherwise; to see that $|D| = \gamma_c(G_{\text{H1}})$, observe that, for a smaller connected dominating set D' of G_{H1}, one could obtain a spanning tree $T_{D'}$ of $G_{\mathrm{H1}}[D']$ and enlarge $T_{D'}$ into a spanning tree T' of G_{H1} having more leaves in comparison with T.

In case H1-UB is set the value of UB-MCD (see Algorithm 4.2), H1-S is assigned D. In this case, to derive an efficient strategy of defense of G_{H1} for H1-UB guards from H1-S, one can first build the neocolonization setup $\langle Z_D, Y_D\rangle$ from D as above and then construct an efficient strategy of defense of G_{H1} for $w(\langle Z_D, Y_D\rangle)$ guards from $\langle Z_D, Y_D\rangle$ as in the proof of Theorem 4.3.

4.4.2 Minimum and maximum minimum dominating set lower bounds

Consider $\gamma_m^\infty(G_{\mathrm{H1}})$ guards eternally defending G_{H1}. After responding to an attack, these guards must occupy the vertices of a dominating set of G_{H1} (recall the introduction of the m-EDSP in Section 4.1). Therefore, $\gamma_m^\infty(G_{\mathrm{H1}}) \geq \gamma(G_{\mathrm{H1}})$. The value of the minimum dominating set lower bound (LB-MDS) is $\gamma(G_{\mathrm{H1}})$.

For each $v \in V(G_{\mathrm{H1}})$, if the attack above occurs at v, then the guards must respond to it by moving to a dominating set of G_{H1} containing v. Provided $\gamma_v(G_{\mathrm{H1}})$ denotes the minimum size of such a dominating set, it follows that $\gamma_m^\infty(G_{\mathrm{H1}}) \geq \gamma_v(G_{\mathrm{H1}})$. As a consequence, $\gamma_m^\infty(G_{\mathrm{H1}})$ is at least $\max_{v\in V(G_{\mathrm{H1}})} \gamma_v(G_{\mathrm{H1}})$, which is the value of the maximum minimum dominating set lower bound (LB-MMD). LB-MMD improves on LB-MDS but to a limited extent: one can show that $\max_{v\in V(G_{\mathrm{H1}})} \gamma_v(G_{\mathrm{H1}}) \leq \gamma(G_{\mathrm{H1}}) + 1$.

To compute LB-MDS, we solve a standard integer program to find the minimum size of a dominating set (MDS-IP):

$$\min\left\{\sum_{v\in V(G_{\mathrm{H1}})} x_v : \sum_{u\in N[v]} x_u \geq 1,\ \forall v \in V(G_{\mathrm{H1}});\ x_v \in \{0,1\},\ \forall v \in V(G_{\mathrm{H1}})\right\}$$

To determine LB-MMD, we run a procedure A in which a version MDS-IP(v) of MDS-IP for $v \in V(G_{\mathrm{H1}})$, a variable m, and a set $S \subseteq V(G_{\mathrm{H1}})$ play the following roles. MDS-IP(v) calculates $\gamma_v(G_{\mathrm{H1}})$: it results from adding $x_v = 1$ to MDS-IP and thus forcing the feasible solutions to represent the dominating sets of G_{H1} containing v. The variable m tracks the maximum $\gamma_v(G_{\mathrm{H1}})$ already calculated in the procedure. The set S filters the vertices that may render an improvement to m, that is, for each $v \in V(G_{\mathrm{H1}}) \setminus S$, $\gamma_v(G_{\mathrm{H1}}) \leq m$.

We now present Procedure A; LB-MMD is given by the final value of m. First, set $m = \gamma(G_{\mathrm{H1}})$ and $S = V(G_{\mathrm{H1}})$. Next, remove from S every vertex v with the following property, denoted by P: v belongs to a dominating set of G_{H1} corresponding to an optimal solution found when solving MDS-IP – note that $\gamma_v(G_{\mathrm{H1}}) = \gamma(G_{\mathrm{H1}})$. While $S \neq \emptyset$, extract u from S, optimize MDS-IP(u) to obtain $\gamma_u(G_{\mathrm{H1}})$, and set $m = \gamma_u(G_{\mathrm{H1}})$ in case $\gamma_u(G_{\mathrm{H1}}) > m$. Also, if $\gamma_u(G_{\mathrm{H1}}) = \gamma(G_{\mathrm{H1}})$, then remove from S each vertex v having Property P(u), which is Property P defined for MDS-IP(u) – notice that $\gamma_v(G_{\mathrm{H1}}) = \gamma(G_{\mathrm{H1}})$.

4.4.3 Dominating-set neocolonization-setup upper bound

The dominating-set neocolonization-setup upper bound (UB-DNS) results from an attempt to improve on UB-MCD: it is given by the weight of a neocolonization setup $\langle Z^*, Y^* \rangle$ of G_{H1} derived from a dominating set that is not restricted to be connected. Theorem 4.4 implies the validity of the bound.

In case H1-UB is given the value of UB-DNS (see Algorithm 4.2), H1-S is assigned $\langle Z^*, Y^* \rangle$. In this case, to derive an efficient strategy of defense of G_{H1} for H1-UB guards from H1-S, one can build an efficient strategy of defense of G_{H1} for $w(\langle Z^*, Y^* \rangle)$ guards from $\langle Z^*, Y^* \rangle$ as in the proof of Theorem 4.3.

The neocolonization setup $\langle Z^*, Y^* \rangle$ is computed as follows. First, we construct a collection $\mathcal{D}$ of dominating sets of G_{H1}. Next, for each $D \in \mathcal{D}$, we build a neocolonization setup $\langle Z_D, Y_D \rangle$ from D, run an improvement procedure for $\langle Z_D, Y_D \rangle$, and assign $\langle Z_D, Y_D \rangle$ to $\langle Z^*, Y^* \rangle$ if $\langle Z^*, Y^* \rangle$ is still unset or $w(\langle Z_D, Y_D \rangle) < w(\langle Z^*, Y^* \rangle)$.

To build $\langle Z_D, Y_D \rangle$ from D for each $D \in \mathcal{D}$, we employ the following approach – see the end of this subsection for an illustration of this approach. Sort the components of $G_{\mathrm{H1}}[D]$ in non-increasing order of number of vertices; assume the components are sorted as C_1, C_2, …, C_p. Form p corresponding sets: $V_1 = N[V(C_1)]$ and, for $i = 2, \ldots, p$, $V_i = N[V(C_i)] \setminus (V_1 \cup V_2 \cup \ldots \cup V_{i-1})$. Find which of these sets are not cliques; say V_{i_1}, V_{i_2}, …, V_{i_q}. Finally, let $\langle Z_D, Y_D \rangle = \langle \{ V_1, V_2, \ldots, V_p \}, \{ V(C_{i_1}), V(C_{i_2}), \ldots, V(C_{i_q}) \} \rangle$.

In order to improve $\langle Z_D, Y_D \rangle$, we consider a result on reducing its weight – see the provided example of applying this result. For $v \in V(G_{\mathrm{H1}}) \setminus D$, denote by P_v the set of every non-clique part V_i of Z_D such that the component C_i of $G_{\mathrm{H1}}[D]$ has a neighbor of v and $V(C_i) \subsetneq V_i$. Choose $u \in V(G_{\mathrm{H1}}) \setminus D$; let u belong to $V_j \in Z_D$. Transform $\langle Z_D, Y_D \rangle$ by setting

$$Z_D = \begin{cases} Z_D \setminus P_u \cup \left\{ \bigcup_{V_i \in P_u} V_i \right\} & \text{in case } V_j \in P_u, \\ Z_D \setminus (P_u \cup \{ V_j \}) \cup \left(\left\{ \bigcup_{V_i \in P_u} V_i \cup \{ u \} \right\} \cup \{ V_j \setminus \{ u \} \} \right) & \text{otherwise;} \end{cases}$$

$$Y_D = Y_D \setminus \{ V(C_i) : V_i \in P_u \} \cup \left\{ \bigcup_{V_i \in P_u} V(C_i) \cup \{ u \} \right\}.$$

If $|P_u| \geq 3$, then $w(\langle Z_D, Y_D \rangle)$ is reduced by $|P_u| - 2$.

We run the following simple improvement procedure for $\langle Z_D, Y_D \rangle$ – see below for an execution of this procedure. Define $v \in V(G_{\mathrm{H1}}) \setminus D$ to be an *improving vertex* if $|P_v| \geq 3$. While there are improving vertices, select $v^* \in \arg\max_{v \in V(G_{\mathrm{H1}}) \setminus D} |P_v|$, apply the transformation above to $\langle Z_D, Y_D \rangle$ for v^*, and update P_v for each $v \in V(G_{\mathrm{H1}}) \setminus D$ so that the newly formed non-clique part of Z_D is removed from further consideration.

The constructed collection $\mathcal{D}$ consists of an s-size dominating set of G_{H1} for every s in a size range R. The motivation to add a larger dominating set D to $\mathcal{D}$ is that it may yield a less-weight neocolonization setup $\langle Z_D, Y_D \rangle$. As an example, for the graph of Figure 4.3, while the 4-size dominating set $\{ a, e, h, i \}$ renders a 7-weight neocolonization setup, the 5-size dominating set $\{ b, e, f, h, i \}$ gives origin to a 6-weight one. We choose $R =$

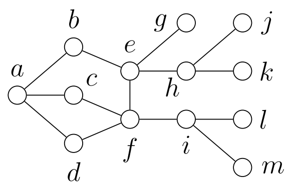


Figure 4.3: A graph for which a larger dominating set may yield a less-weight neocolonization setup.

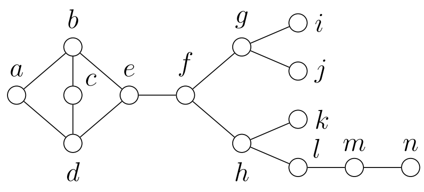


Figure 4.4: A graph having an 8-weight neocolonization setup that can be improved into a 7-weight one.

$[\gamma(G_{\mathrm{H1}}), \gamma_c(G_{\mathrm{H1}})]$ – this range seems reasonable for our aim: to produce neocolonization setups weighing less than $\gamma_c(G_{\mathrm{H1}}) + 1$, the value of UB-MCD.

We arrange $\mathcal{D}$ to be formed of the dominating sets computed through the mixed integer programs below. Our intention is to obtain dominating sets inducing subgraphs having large components. In a pursuit of this intention, we determine dominating sets inducing subgraphs having maximum vertex-degree sum (equivalently, having maximum number of edges).

First, we solve the following program, in which we use x variables to select vertices that form a dominating set of G_{H1} and y variables to count the vertex degrees of the subgraph induced by this dominating set. At an optimal solution, the x variables encode a minimum dominating set of G_{H1} inducing a subgraph having maximum vertex-degree sum.

$$\begin{array}{lll}
\min & |V(G_{\mathrm{H1}})| \displaystyle\sum_{v \in V(G_{\mathrm{H1}})} x_v - \sum_{v \in V(G_{\mathrm{H1}})} y_v & \\
\text{s.t.:} & & \\
& \displaystyle\sum_{u \in N[v]} x_u \geq 1, & \forall v \in V(G_{\mathrm{H1}}) \\
& y_v \leq d(v)\ x_v, & \forall v \in V(G_{\mathrm{H1}}) \\
& y_v \leq \displaystyle\sum_{u \in N(v)} x_u, & \forall v \in V(G_{\mathrm{H1}}) \\
& x_v \in \{0,1\},\ y_v \geq 0, & \forall v \in V(G_{\mathrm{H1}})
\end{array}$$

Next, we optimize modified versions of this program. For each $s \in [\gamma(G_{\mathrm{H1}})+1, \gamma_c(G_{\mathrm{H1}})]$, we consider the version that results from removing the term $|V(G_{\mathrm{H1}})| \sum_{v \in V(G_{\mathrm{H1}})} x_v$ from the objective function and adding the constraint $\sum_{v \in V(G_{\mathrm{H1}})} x_v \leq s$. In this version, an optimal solution corresponds to an s-size dominating set of G_{H1} inducing a subgraph having maximum vertex-degree sum.

Examples. We illustrate two steps of the computation of UB-DNS: the construction of a neocolonization setup $\langle Z_D, Y_D \rangle$ from a dominating set D of the collection $\mathcal{D}$ and the execution of the improvement procedure for $\langle Z_D, Y_D \rangle$. To this end, assume that G_{H1} is the graph of Figure 4.4 and D is $\{\, b,\ e,\ g,\ h,\ m \,\}$.

In the approach to build $\langle Z_D, Y_D\rangle$ from D, suppose that the components of $G_{\mathrm{H1}}[D]$ are sorted as C_1, C_2, C_3, C_4 such that $V(C_1) = \{\, b,\ e \,\}$, $V(C_2) = \{\, g \,\}$, $V(C_3) = \{\, h \,\}$, and $V(C_4) = \{\, m \,\}$. We obtain $\langle Z_D, Y_D\rangle = \langle\{\, V_1,\ V_2,\ V_3,\ V_4 \,\},\ \{\, V(C_1),\ V(C_2),\ V(C_3) \,\}\rangle$, where $V_1 = \{\, a,\ b,\ c,\ d,\ e,\ f \,\}$, $V_2 = \{\, g,\ i,\ j \,\}$, $V_3 = \{\, h,\ k,\ l \,\}$, and $V_4 = \{\, m,\ n \,\}$. Note that $w(\langle Z_D, Y_D\rangle) = 3 + 2 + 2 + 1 = 8$. The improvement procedure applies the transformation to $\langle Z_D, Y_D\rangle$ once, for vertex f. In this case, $P_f = \{\, V_1,\ V_2,\ V_3 \,\}$. It results that $Z_D = \{\, (V_1 \cup V_2 \cup V_3),\ V_4 \,\} = \{\, \{\, a,\ b,\ \ldots,\ l \,\},\ \{\, m,\ n \,\} \,\}$ and $Y_D = \{\, V(C_1) \cup V(C_2) \cup V(C_3) \cup \{\, f \,\} \,\} = \{\, b,\ e,\ f,\ g,\ h \,\}$. Now, $w(\langle Z_D, Y_D\rangle) = 6 + 1 = 7$.

4.4.4 Time-limited exact-algorithm lower bound

The time-limited exact-algorithm lower bound (LB-TLE) is drawn from a time-limited version of Algorithm 4.3, an exact algorithm for the m-EDSP developed by us inspired on the work of Fomin et al. [11] for a closely related problem. Below, we further describe this algorithm and derive LB-TLE.

Algorithm 4.3 Exact algorithm for the m-EDSP

```
Input: A graph $G$.
Output: $\gamma_m^\infty(G)$ and an $m$-EDSC of $G$.
1: Compute $\gamma(G)$.
2: For every integer $k$ from $\gamma(G)$ to $|V(G)|$, do:
3:     If $\gamma_m^\infty(G) \le k$, then output $k$ and a $k$,$m$-EDSC of $G$, and stop – it follows that $\gamma_m^\infty(G) = k$.
```

In Step 3 of Algorithm 4.3, we use the fact that $\gamma_m^\infty(G) \le k$ if and only if G has a k,m-EDSC (see Theorem 4.1). First, we obtain a graph $\mathcal{G}_k$ with the following property, denoted by P_1: in case $V(\mathcal{G}_k)$ is nonempty, $V(\mathcal{G}_k)$ is a k,m-EDSC of G; otherwise, G does not have a k,m-EDSC. Then, we check if $\gamma_m^\infty(G) \le k$ by testing if $V(\mathcal{G}_k) \ne \emptyset$ and, in case the condition is true, output k and $V(\mathcal{G}_k)$. We detail next how $\mathcal{G}_k$ is obtained and why Property P_1 is valid.

We begin by constructing a graph $\mathcal{G}_k^0$: $V(\mathcal{G}_k^0)$ consists of all k-size dominating sets of G and, for $D_i, D_j \in V(\mathcal{G}_k^0)$, $E(\mathcal{G}_k^0)$ contains D_iD_j if and only if D_i shifts to D_j – because D_i shifts to itself and, if D_i shifts to D_j, then D_j shifts to D_i, this graph has a loop for each vertex and is defined to be undirected. To assemble $V(\mathcal{G}_k^0)$, we employ constraint programming to search for all feasible solutions to constraints

$$\sum_{v \in V(G)} x_v = k;\ \sum_{u \in N[v]} x_u \ge 1,\ \forall v \in V(G);\ x_v \in \{0, 1\},\ \forall v \in V(G).$$

To decide if $D_iD_j \in E(\mathcal{G}_k^0)$, we test if there is a k-unit flow in a directed graph where, for every $u \in D_i$ and $v \in D_j$, at most one unit of flow can pass through all arcs uw such that $w \in N[u] \cap D_j$ and through all arcs wv such that $w \in N[v] \cap D_i$. Provided $n = |V(G)|$, one can build $V(\mathcal{G}_k^0)$ in $O(2^n n^2)$ time and determine $E(\mathcal{G}_k^0)$ in $((|V(\mathcal{G}_k^0)|(|V(\mathcal{G}_k^0)| - 1)/2)\ n^{O(1)}) \in O(4^n n^{O(1)})$ time.

Next, we assign to $\mathcal{G}_k$ the last graph obtained in the following process. For every graph $\mathcal{G}_k^i$ below, let a vertex of $\mathcal{G}_k^i$ be *unsafe* if it corresponds to a dominating set D of G such that $\mathcal{N}(D, v) \cap V(\mathcal{G}_k^i) = \emptyset$ for some $v \in V(G)$. Set $i = 0$. While $V(\mathcal{G}_k^i) \ne \emptyset$ and $\mathcal{G}_k^i$ has unsafe vertices, find all unsafe vertices of $\mathcal{G}_k^i$, obtain $\mathcal{G}_k^{i+1}$ by removing them from $\mathcal{G}_k^i$, and increment i by 1. This loop can be executed in $O(8^n n)$ time: it performs at most $|V(\mathcal{G}_k^0)|$ iterations, which can each be run in $O(|E(\mathcal{G}_k^0)|n + |V(\mathcal{G}_k^0)|))$ time.

We now show that $\mathcal{G}_k$ has the following property, named P$_2$: every k,m-EDSC of G is a subset of $V(\mathcal{G}_k)$. By induction on i, we prove that each graph $\mathcal{G}_k^i$ above has Property P$_2(i)$, which is Property P$_2$ defined for $\mathcal{G}_k^i$. Since $V(\mathcal{G}_k^0)$ contains all k-size dominating sets of G, $\mathcal{G}_k^0$ has Property P$_2(0)$. Suppose that $\mathcal{G}_k^i$ has Property P$_2(i)$. The property implies that the dominating sets of G corresponding to the unsafe vertices of $\mathcal{G}_k^i$ cannot belong to a k,m-EDSC of G. Hence, $\mathcal{G}_k^{i+1}$ has Property P$_2(i{+}1)$.

Finally, we prove that $\mathcal{G}_k$ has Property P$_1$. If $V(\mathcal{G}_k) \neq \emptyset$, then, by its construction, $\mathcal{G}_k$ has no unsafe vertices and, therefore, $V(\mathcal{G}_k)$ is a k,m-EDSC of G. Else, Property P$_2$ implies that G has no k,m-EDSC.

The correctness of Algorithm 4.3 is based on the fundamental lower bound $\gamma(G)$ (see Section 4.4.2) and upper bound $|V(G)|$ on $\gamma_m^\infty(G)$. Its running time is dominated by the duration of Step 2's loop, which can be executed in $O(8^n n^{O(1)})$ time – recall the partial complexities discussed above. The subsequent characteristic of the algorithm contrasts with Feature (iii) of the heuristic methods proposed in this work. If we also output $\mathcal{G}_k$ in Step 3, then one can derive as follows, from the return of Algorithm 4.3, a strategy of defense of G for $\gamma_m^\infty(G)$ guards that is not necessarily efficient. Define $\mathcal{C}$ to be the m-EDSC $V(\mathcal{G}_k)$; since the output of the algorithm occurs for $k = \gamma_m^\infty(G)$, this m-EDSC consists of $\gamma_m^\infty(G)$-size dominating sets. Develop an $O(|V(\mathcal{G}_k)| + n^{O(1)})$-time defense method M of $\mathcal{C}$ such that, for $D_i \in \mathcal{C}$ and $v \in V(G)$, M consults $\mathcal{G}_k$ to find $D_j \in \mathcal{N}(D_i, v) \cap \mathcal{C}$ and employs the flow test mentioned in the construction of $\mathcal{G}_k^0$ to determine a shift from D_i to D_j. Set I_M to be an appropriate $O(|V(\mathcal{G}_k)| + n^2)$-length input to M. The triple $\langle \mathcal{C}, M, I_M \rangle$ is a strategy of defense of G for $\gamma_m^\infty(G)$ guards. Because $|V(\mathcal{G}_k)|$ may be exponential in n, it is not necessarily efficient.

To obtain LB-TLE, we modify Algorithm 4.3 to output a lower bound on $\gamma_m^\infty(G)$. We allow the algorithm to be interrupted due to time limit when performing Step 3. Assume such an interruption occurs, and suppose k at its current value. In case it was checked that $\gamma_m^\infty(G) \leq k$, the algorithm completes normally. If it was checked that $\gamma_m^\infty(G) > k$ or this checking was not finished, then the algorithm outputs $k + 1$ or k respectively and stops (no k,m-EDSC of G is provided).

Consider this modified version of Algorithm 4.3 applied to G_{H1} and optimized as below; LB-TLE is the lower bound on $\gamma_m^\infty(G_{\mathrm{H1}})$ outputted. The following optimizations rely on two observations and on lower and upper bounds on $\gamma_m^\infty(G_{\mathrm{H1}})$, namely, l and u, known at the time of the computation of LB-TLE (see Algorithm 4.2). In Step 2, one can replace $\gamma(G)$ with any lower bound on $\gamma_m^\infty(G)$. We remove Step 1 and initialize k to l – when computing LB-TLE, we know that l is greater than or equal to the value of LB-MDS (see Algorithm 4.2) and thus $l \geq \gamma(G_{\mathrm{H1}})$. If $\gamma_m^\infty(G) \leq p$ for some integer p, then the best lower bound on $\gamma_m^\infty(G)$ is determined up to the end of Step 2 loop's iteration for $k = p - 1$, when one concludes that $\gamma_m^\infty(G) \geq p$ in case the algorithm does not stop. We replace $|V(G)|$ with $u - 1$ in Step 2 and add a new step after the loop that simply outputs u.

4.5 Results – Part 1

In this section, we present computational results for Heur 1. We implemented Heur 1 in `C/C++` using IBM CPLEX 12.5.1 [22] to solve all involved integer programs except that for UB-MCD, for which we employed Gurobi 5.6.3 [20]. The solution search done for LB-TLE with constraint programming (see Section 4.4.4) was performed via IBM CP 12.5.1 [21]. Moreover, we set 5 minutes to be the time limit for LB-TLE. All runs were made on an Intel Quad-Core 3.4GHz machine with 8 GB of RAM.

4.5.1 Instances

Our instance set contains 600 Erdős-Rényi random graphs forced to be connected and divided into 20 groups of 30 graphs. Each group is characterized by a pair of intervals, one restricting the order of the graphs and the other, their density. The order intervals are $[10, 20]$, $[21, 30]$, $[31, 40]$, and $[41, 50]$, while the density intervals are $(0\%, 20\%]$, $(20\%, 40\%]$, $(40\%, 60\%]$, $(60\%, 80\%]$, and $(80\%, 100\%]$.

We evaluate if this instance set is representative by analyzing if it is heterogeneous with respect to the following aspects: for each graph G, $t(G)$, namely, the time taken by Algorithm 4.3 to compute $\gamma_m^\infty(G)$, and $p(G)$, that is, the number of dominating sets of G of size $\gamma_m^\infty(G)$. Having distinct instances with respect to these aspects, we expect to experiment on cases where it ranges from easy to difficult to solve the m-EDSP and to obtain an m-EDSC consisting of a polynomial number of small dominating sets.

Our analysis is carried out through Figures 4.5 and 4.6. The former figure shows for how many graphs p lies, respectively, in intervals $P_1 = [1, 100]$, $P_2 = [101, 500]$, $P_3 = [501, 1000]$, $P_4 = [1001, 5000]$, $P_5 = [5001, 15000]$, and $P_6 = [15001, 187140]$ – the maximum p found being 187140. The latter figure presents, for graphs with p in each interval from P_1 to P_6, the percentage of cases where t belongs, respectively, to intervals $T_1 = (0\text{s}, 1\text{s}]$, $T_2 = (1\text{s}, 10\text{s}]$, $T_3 = (10\text{s}, 1\text{m}]$, $T_4 = (1\text{m}, 10\text{m}]$, $T_5 = (10\text{m}, 30\text{m}]$, $T_6 = (30\text{m}, 1\text{h}]$, and $T_7 = (1\text{h}, 2\text{h}]$ – s, m, and h meaning seconds, minutes, and hours – or is greater than the limit of 2 hours (>2h).

By Figures 4.5 and 4.6, we see that the instances vary considerably with respect to p and t. Notice that, for 42.5% of the graphs, there are at least 1000, and, for 6.3% of the graphs, there are at least 15000 dominating sets of size γ_m^∞. Since the time complexity of Algorithm 4.3 grows with this number, t has larger values in the last three columns and the largest values in the last column of Figure 4.6.

4.5.2 Results

We validate Features (i) and (ii) (see Section 4.1) of Heur 1 by measuring the outputted gap, i.e., H1-UB − H1-LB (see Algorithm 4.2), and the running time. We also discuss how the instance aspects listed in Section 4.5.1 influence the gap value.

The optimum was obtained for 61% of the instances and the gap was at most 1 in 97% of the cases. Moreover, the gap was at most 2 for all but one test, and the overall maximum gap was 3. As for the running time, it was at most 1 second for 72.5% of the

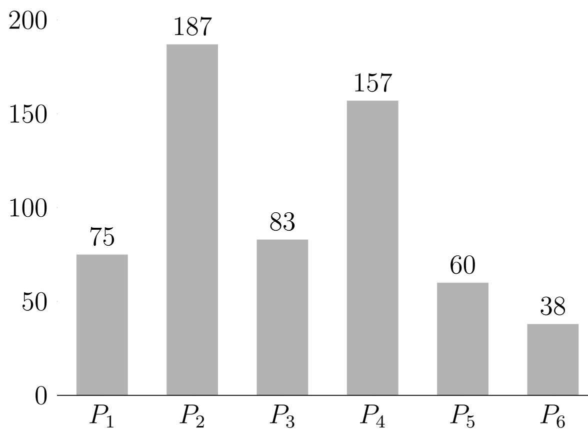


Figure 4.5: Number of instances per intervals of number (p) of γ_m^∞-size dominating sets.

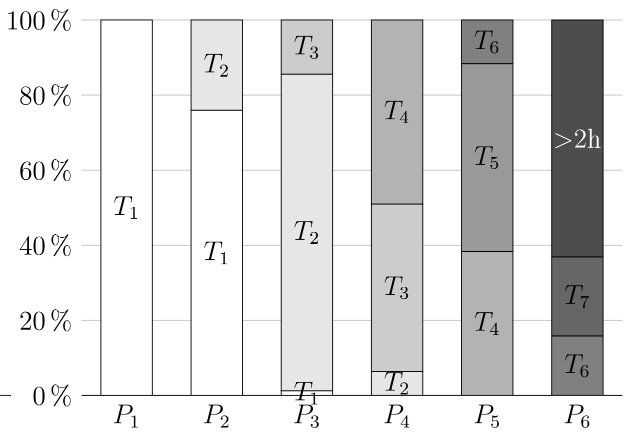


Figure 4.6: Intervals of time (t) to compute γ_m^∞ per intervals of number (p) of γ_m^∞-size dominating sets.

instances and at most 18.67 seconds – the average running time – in 90% of the cases. The overall maximum time the method took to execute was 308.32 seconds.

It is worth noting that LB-TLE was in general the most time-consuming bound. When Heur 1 executed in more than 1 second and computed LB-TLE (recall Algorithm 4.2), this computation represented on average 86.6% of the running time. Nevertheless, LB-TLE improved the gap for 12.17% of all the graphs and for 42.50% of the graphs of density at most 20%, which, as seen below, form an important instance class.

In Tables 4.1 to 4.4, we report how the obtained gap values distribute over each order, density, p, and t interval (see Section 4.5.1), respectively. For each order interval, for all but one p interval, and for all but two t intervals, the picture is practically the same: for at least 96% of the instances, the gap is at most 1. One explanation for the distinction of intervals P_6, T_6, and >2h is that it is simply caused by the reduced number of tests in these intervals, especially in T_6, which make a significant percentage result from a few cases of gap greater than 1. We mention that, for intervals P_6, T_6, and >2h, these cases correspond to graphs of density at most 20%, and that, as observed next, very low densities substantially impact the quality of H1-UB.

Among the density intervals, there is a clear contrast: for intervals $(20\%, 40\%]$ to $(80\%, 100\%]$, the gap is greater than 1 for at most 0.83% of the graphs, while, for interval $(0\%, 20\%]$, the gap is greater than 1 in 14.17% of the cases. This contrast can be related to the quality of UB-MCD. In the last two columns of Table 4.2, we count, for UB-MCD and UB-DNS, for how many graphs each of them is the origin of H1-UB. By these columns, we see that UB-MCD is the main responsible for the value of H1-UB. However, UB-MCD is not as efficacious in the cases of density at most 20% as it is in the other ones. For interval $(0\%, 20\%]$, UB-DNS improves on UB-MCD but does not prevent a quality loss of H1-UB.

From the discussion above, we conclude that, for a graph G, one can produce a good-quality upper bound on $\gamma_m^\infty(G)$ at a small time cost by first computing the weight of a particular neocolonization setup of G, namely, the one associated with UB-MCD (see

	Gap			
	0	1	2	3
$[10, 20]$	101	43	6	0
$[21, 30]$	91	55	4	0
$[31, 40]$	86	60	3	1
$[41, 50]$	88	58	4	0

Table 4.1: Gap (H1-UB − H1-LB) per order intervals.

	Gap				H1-UB		
	0	1	2	3	(UB-)	MCD	DNS
(0%, 20%]	39	64	16	1		96	24
(20%, 40%]	83	36	1	0		117	3
(40%, 60%]	83	37	0	0		119	1
(60%, 80%]	90	30	0	0		120	0
(80%, 100%]	71	49	0	0		119	1

Table 4.2: Gap (H1-UB − H1-LB) and H1-UB origin per density intervals.

	Gap			
	0	1	2	3
P_1	43	30	2	0
P_2	89	94	4	0
P_3	47	34	2	0
P_4	118	35	4	0
P_5	43	15	2	0
P_6	26	8	3	1

Table 4.3: Gap (H1-UB − H1-LB) per intervals of number (p) of γ_m^∞-size dominating sets.

	Gap			
	0	1	2	3
T_1	113	99	6	0
T_2	66	56	3	0
T_3	55	25	2	0
T_4	74	23	3	0
T_5	25	5	0	0
T_6	12	0	1	0
T_7	7	1	0	0
>2h	14	7	2	1

Table 4.4: Gap (H1-UB − H1-LB) per intervals of time (t) to compute γ_m^∞.

Section 4.4.1), and next calculating the potentially improving weights of other specific neocolonization setups of G, that is, the ones considered for UB-DNS (see Section 4.4.3). It is thus natural to investigate if it pays off moving further in the direction of determining the minimum weight of a neocolonization setup of G. We perform this movement in our second heuristic method.

4.6 Second heuristic method

Our second heuristic method (Heur 2) extends the first one by computing an additional upper bound on the m-eternal domination number of the input graph, namely, the minimum-weight neocolonization-setup upper bound (UB-MNS). Heur 2 is summarized in Algorithm 4.4 ; we assume without loss of generality a connected input graph to Heur 1 (see the introduction of Section 4.4) and do the same for Heur 2.

Algorithm 4.4 Second heuristic method

```
Input: A connected graph $G_{H2}$.
Output: An upper bound H2-UB and a lower bound H2-LB on $\gamma_m^\infty(G_{H2})$ and a structure H2-S.
1: Steps 1 to 8 of Algorithm 4.2 with the list of Step 2 changed to UB-MCD, LB-MDS, LB-
   MMD, UB-DNS, UB-MNS, LB-TLE and $G_{H1}$, H1-UB, H1-LB, and H1-S replaced with,
   respectively, $G_{H2}$, H2-UB, H2-LB, and H2-S.
```

In Section 4.6.1, we propose an integer program to determine the minimum weight of a neocolonization setup of G_{H2}. We obtain a solution s to this program and a by-product $B(s)$ using the restricted branch-and-price method described in Section 4.6.2. The value of UB-MNS is the cost $c(s)$ of s. Also in Section 4.6.1, we specify $B(s)$ and explain how one can build from $B(s)$ a neocolonization setup $\langle Z, Y\rangle$ of G_{H2} such that $w(\langle Z, Y\rangle) \leq c(s)$. This inequality combined with Theorem 4.4 implies $\gamma_m^\infty(G_{\mathrm{H2}}) \leq c(s)$ and, therefore, the validity of UB-MNS.

We show in Section 4.4 that one can derive, from the structure H1-S outputted by Heur 1, an efficient strategy of defense of G_{H1} for H1-UB guards. To prove an analogous result for Heur 2, it remains to consider the following assignment to H2-S. In case H2-UB is set the value of UB-MNS (see Algorithm 4.4), H2-S is assigned $B(s)$. In this case, by first building a neocolonization setup $\langle Z, Y\rangle$ of G_{H2} from $B(s)$ as mentioned above and next constructing an efficient strategy of defense of G_{H2} for $w(\langle Z, Y\rangle)$ guards from $\langle Z, Y\rangle$ as in the proof of Theorem 4.3, one can derive from H2-S an efficient strategy of defense of G_{H2} for H2-UB guards – this number of guards is actually at most H2-UB since $w(\langle Z, Y\rangle) \leq c(s)$.

4.6.1 Minimum-weight neocolonization-setup integer program

Consider the weight of a forest F of G_{H2} to be as follows. Let T be a tree of G_{H2}. The *peripheral vertices* of T form a subset of the set of all 0-degree and 1-degree vertices of T. The *weight* w_T of T is the number of non-peripheral vertices of T added by 1. Suppose that T_1, T_2, …, T_k are the *maximal trees* of F, that is, the components of F. The *peripheral vertices* of F are the peripheral vertices of T_1, T_2, …, T_k all together. The *weight* w_F of F is given by $\sum_{i=1}^{k} w_{T_i}$; observe that w_F equals the number of non-peripheral vertices of F added by k.

Furthermore, consider the following correspondence between forests of G_{H2} and neocolonization setups of induced subgraphs of G_{H2}. Take a forest F of G_{H2} having maximal trees T_1, T_2, …, T_k. Let $S = V(F)$, and build the neocolonization setup $\langle Z, Y\rangle$ of $G_{\mathrm{H2}}[S]$: assume $V_i = V(T_i)$ for $i = 1, 2, \ldots, k$, set $Z = \{ V_1, V_2, \ldots, V_k \}$, and, for each non-clique part V_i of Z, assign to $Y(V_i)$ the set of all non-peripheral vertices of T_i – one can show that these vertices form a connected dominating set of $G_{\mathrm{H2}}[V_i]$ if V_i is a non-clique part of Z. We call $\langle Z, Y\rangle$ *the neocolonization setup corresponding to* F. Conversely, choose a neocolonization setup $\langle Z, Y\rangle$ of an induced subgraph of G_{H2}. For every $V_i \in Z$, construct a spanning tree T_i of $G_{\mathrm{H2}}[V_i]$ such that, in case V_i is not a clique, $V_i \setminus Y(V_i)$ gives all peripheral vertices of T_i – one can always meet this condition because $Y(V_i)$ is a connected dominating set of $G_{\mathrm{H2}}[V_i]$. Define F to be the forest composed of the constructed trees. We refer to F as *a forest of* G_{H2} *corresponding to* $\langle Z, Y\rangle$. Notice that $w(\langle Z, Y\rangle) \leq w_F$ in both alternatives above.

The integer program below models the problem of calculating the minimum weight $\theta_C(G_{\mathrm{H2}})$ of a neocolonization setup of G_{H2}. In this program, we determine a neocolonization setup $\langle Z, Y\rangle$ of G_{H2} as follows. Next, $\mathcal{C}$ and $\mathcal{F}$ denote, respectively, the set of all cliques and of all forests of G_{H2}. We use x variables to select some clique parts of Z. Moreover, we employ a single y variable to establish the remainder of $\langle Z, Y\rangle$, namely, $\langle Z', Y\rangle$, where Z' is the set of all remaining parts of Z – note that $\langle Z', Y\rangle$ is a neocolonization setup of the subgraph of G_{H2} induced by $\bigcup_{V_i \in Z'} V_i$; if $y_F = 1$, then $\langle Z', Y \rangle$ is the neocolonization setup corresponding to the forest F.

$$\min \qquad z = \sum_{C \in \mathcal{C}} x_C + \sum_{F \in \mathcal{F}} w_F y_F \tag{4.1}$$

s.t.:

$$\sum_{C \in \mathcal{C}: v \in C} x_C + t_v = 1, \qquad \forall v \in V(G_{\mathrm{H2}}) \tag{4.2}$$

$$\sum_{F \in \mathcal{F}: v \in V(F)} y_F = t_v, \qquad \forall v \in V(G_{\mathrm{H2}}) \tag{4.3}$$

$$\sum_{F \in \mathcal{F}} y_F \leq 1 \tag{4.4}$$

$$x_C,\ y_F,\ t_v \in \{0, 1\}, \qquad \forall C \in \mathcal{C},\ \forall F \in \mathcal{F},\ \forall v \in V(G_{\mathrm{H2}}) \tag{4.5}$$

Constraint (4.4) ensures that there is at most one nonzero y variable. Assume that x_{C_1}, x_{C_2}, ..., x_{C_k} and y_F are, respectively, the x and y variables having value 1. For each $v \in V(G_{\mathrm{H2}})$, if the variable t_v is assigned 0, then constraints (4.2) state that v belongs to exactly one of the cliques C_1, C_2, ..., C_k. In case $t_v = 1$, constraints (4.3) say that v is a vertex of the forest F. Constraints (4.2) and (4.3) combined impose that $\{ C_1, C_2, \ldots, C_k, V(F) \}$ is a partition of $V(G_{\mathrm{H2}})$.

To see that the optimum z^* of program (4.1)-(4.5) equals $\theta_C(G_{\mathrm{H2}})$, observe the following. The cost of a solution to the program is an upper bound on the weight of the corresponding neocolonization setup of G_{H2}. Thus, $z^* \geq \theta_C(G_{\mathrm{H2}})$. Given a neocolonization setup $\langle Z, Y \rangle$ of G_{H2}, transform $\langle Z, Y \rangle$ as in the beginning of the proof of Theorem 4.3 : $w(\langle Z, Y \rangle)$ remains the same and, for each non-clique part V_i of Z, $Y(V_i) \subsetneq V_i$. Take a solution s to program (4.1)-(4.5) corresponding to $\langle Z, Y \rangle$ in which all clique parts of Z are determined by x variables. The cost of s equals $w(\langle Z, Y \rangle)$. Hence, $\theta_C(G_{\mathrm{H2}}) \geq z^*$ and, therefore, $\theta_C(G_{\mathrm{H2}}) = z^*$.

Modify program (4.1)-(4.5) by defining $\mathcal{C}$ to be the set of only the maximal cliques of G_{H2} and by turning constraints (4.2) into $\geq$ inequalities. Call the *minimum-weight neocolonization-setup integer program* (MWNS-IP) the resulting program:

$$\min \qquad z = \sum_{C \in \mathcal{C}} x_C + \sum_{F \in \mathcal{F}} w_F y_F \tag{4.6}$$

s.t.:

$$\sum_{C \in \mathcal{C}: v \in C} x_C + t_v \geq 1, \qquad \forall v \in V(G_{\mathrm{H2}}) \tag{4.7}$$

$$\text{Constraints (4.3)-(4.5)} \tag{4.8}$$

Notice that, if C_1, C_2, ..., C_k and F are, accordingly, the cliques and the forest selected in a solution to this program, then $\{ C_1, C_2, \ldots, C_k, V(F) \}$ is not necessarily a partition of $V(G_{\mathrm{H2}})$. However, such a partition may be obtained by removing from the cliques C_1, C_2, ..., C_k each vertex that belongs to F and keeping in a single clique among C_1, C_2,

…, C_k every vertex that is outside F. One can conclude that program (4.1)-(4.5) has the same optimum as the MWNS-IP, which contains a potentially much smaller number of x variables. We choose the MWNS-IP to compute $\theta_C(G_{\mathrm{H2}})$ – our choice is inspired by a formulation due to Mehrotra and Trick [29] for the problem of finding a minimum coloring of a graph.

We now specify $B(s)$ (see the introduction of Section 4.6) for a solution s to the MWNS-IP and show how to derive from $B(s)$ a neocolonization setup $\langle Z, Y\rangle$ of G_{H2} having weight less than or equal to the cost of s. Suppose that x_{C_1}, x_{C_2}, …, x_{C_p} and y_F are, respectively, the x and y variables with value 1 in s. We have that $B(s) = \{\, C_1, C_2, \ldots, C_p, V(F), P(F)\,\}$, where $P(F)$ is the set of all peripheral vertices of the forest F. To derive $\langle Z, Y\rangle$, proceed as follows. Set $V_1 = C_1 \setminus V(F)$ and, for $i = 2, \ldots, p$, $V_i = C_i \setminus (V(F) \cup C_1 \cup C_2 \cup \ldots \cup C_{i-1})$. Find the components G_1, G_2, …, G_q of $G_{\mathrm{H2}}[V(F) \setminus P(F)]$. For $i = 1, 2, \ldots, q$, compute $P_i = \{\, v \in P(F) : \exists u \in V(G_i),\ uv \in E(G)\,\}$. Let $V_{p+1} = V(G_1) \cup P_1$ and, for $i = 2, \ldots, q$, $V_{p+i} = V(G_i) \cup P_i \setminus (P_1 \cup P_2 \cup \ldots \cup P_{i-1})$. Assign $Z = \{\, V_1, V_2, \ldots, V_{p+q}\,\}$, and, for $i = 1, 2, \ldots, q$, add $V(G_i)$ to Y if V_{p+i} is not a clique of G_{H2}. It can be argued that $w(\langle Z, Y\rangle)$ is at most the cost of s.

The MWNS-IP has relatively few constraints. Nevertheless, it may have tremendously many x variables – even with $\mathcal{C}$ containing only the maximal cliques of G_{H2}, $|\mathcal{C}|$ may be exponential in $|V(G_{\mathrm{H2}})|$ – and y variables. Its variables must not, therefore, be handled all together in an explicit manner. To deal with the MWNS-IP, we use a branch-and-price method [32], which is a method that employs the column generation technique within a branch-and-bound framework.

4.6.2 Restricted branch-and-price

We apply a restricted branch-and-price method to the MWNS-IP. This method is restricted because the solving process can be prematurely stopped due to time limit or as a consequence of the employed branching rules (specified in Section 4.6.2.5) not applying to the current node of the enumeration tree. Below, we present the main details of the method, which was developed using SCIP [16], a full-fledged constraint and mixed integer programming solver that supports the implementation of column generation for integer programs.

Terminology and notation. Our method is a linear-relaxation-based branch-and-price. Given a node of the enumeration tree, denote by IP the integer program of the node and by LP the linear relaxation of the IP. We call *clique* and *forest variables*, respectively, the x and y variables of the IP (see (4.6)-(4.8)). We write *clique* and *forest columns*, analogously, for the columns corresponding to the x and y variables in the matrix representation of the IP. The same nomenclature is used for the LP.

To solve the LP, we begin by optimizing the *restricted LP* (RLP), namely, a feasible restricted version of the LP that has all constraints, all t variables, and a number of clique and forest variables of the LP. The optimal solution obtained to the RLP is also optimal to the LP if the reduced cost of every remaining clique and forest variable of the LP is non-negative. While this condition is not met, we add at least one column corresponding to a variable with negative reduced cost to the RLP and reoptimize it.

The *master problem* (MP) stands for the linear relaxation of the integer program of the current node of the enumeration tree. The *restricted master problem* (RMP) designates the most recently solved restricted version of the MP; u^*_{RMP} denotes the optimal dual solution computed for the RMP. The *clique* and *forest pricing problems* are, respectively, the problems of determining the minimum reduced cost of a clique and of a forest variable of the MP.

4.6.2.1 Pricing strategy

We test if the optimal solution obtained to the RMP is also optimal to the MP by checking if $\bar{c}^*_{cliq}$ and $\bar{c}^*_{for}$ are non-negative, where $\bar{c}^*_{cliq}$ and $\bar{c}^*_{for}$ denote, respectively, the minimum reduced cost of a clique and of a forest variable. To check if $\bar{c}^*_{cliq} \geq 0$, we run a greedy method that tries to find a clique variable with negative reduced cost and, only when this method fails, we solve the clique pricing problem. To check if $\bar{c}^*_{for} \geq 0$, we proceed analogously.

Furthermore, we begin by checking if $\bar{c}^*_{cliq} \geq 0$ and, only in case this condition holds, we move on for checking if $\bar{c}^*_{for} \geq 0$. The motivation to this approach is twofold. First, in our early experiments, the time to solve the forest pricing problem was significantly greater in comparison with the clique pricing problem. Secondly, as seen later in Section 4.6.2.4, as a consequence of this approach, we can set an interesting lower bound on the optimum of the MP.

4.6.2.2 Clique-column generation

The reduced cost of a clique variable x_C is given by $1 - \sum_{v \in C} \check{\theta}_v$, where C is the corresponding clique of G_{H2} and $\check{\theta}_v$ for $v \in V(G_{\mathrm{H2}})$ are the values in u^*_{RMP} (see the introduction of Section 4.6.2) of the non-negative dual variables θ_v for $v \in V(G_{\mathrm{H2}})$ associated with constraints (4.7) of the MP. To solve the clique pricing problem, we define the weight of a vertex $v \in V(G_{\mathrm{H2}})$ to be $\check{\theta}_v$, consider the weight of a clique of G_{H2} to be the sum of the weights of its vertices, and determine the maximum weight of a clique of G_{H2}. For this purpose, we use the Tclique algorithm, a combinatorial branch-and-bound by Borndörfer and Kormos [6] that, in a version of limited number of enumeration nodes, has been successfully applied for heuristically separating clique-based cuts [3, 4, 5]. An implementation of Tclique is already included in SCIP.

Before solving the clique pricing problem, we run a greedy method M that tries to build a clique C of G_{H2} such that the reduced cost of the clique variable x_C is negative. The method M initializes C to be empty, sorts the vertices of G_{H2} in non-increasing order with respect to $\check{\theta}$ values, and, following this order, inserts into C every vertex of G_{H2} that is adjacent to each one previously inserted.

If M succeeds, i.e., $\sum_{v \in C} \check{\theta}_v > 1$, then the corresponding clique column is added to the RMP. Else, we run Tclique. In this case, among the cliques of weight greater than 1 found during Tclique's execution, the clique columns corresponding to the ones of highest weights are added to the RMP. The number of added columns is a function of the number of vertices of G_{H2} and the number of initial clique columns, which are discussed in Section 4.6.2.7.

We code the clique-column generation above by adapting an implementation provided in SCIP. This implementation works in an analogous manner within the context of a branch-and-price method to find a minimum coloring of a graph using an integer program proposed by Mehrotra and Trick [29].

4.6.2.3 Forest-column generation

Inspired by the flow-based mixed integer program introduced by Reis et al. [30] to find a maximum-leaf spanning tree of a connected graph (this program is mentioned in Section 4.4.1), we propose a flow-based mixed integer program (FPP-MIP) to solve the forest pricing problem. To this end, we write the reduced cost of a forest variable y_F as below. Assume that the corresponding forest F of G_{H2} has k maximal trees. Let $O(F) = \{v \in V(G_{\mathrm{H2}}) : v \text{ is outside } F\}$ and $P(F) = \{v \in V(F) : v \text{ is a peripheral vertex of } F\}$. Consider μ_v for $v \in V(G_{\mathrm{H2}})$ and λ to be, respectively, the dual variables associated with constraints (4.3) and constraint (4.4) of the MP. Suppose that $\check{\mu}_v$ for $v \in V(G_{\mathrm{H2}})$ and $\check{\lambda}$ are, accordingly, the values in u^*_{RMP} (see the introduction of Section 4.6.2) of μ_v for $v \in V(G_{\mathrm{H2}})$ and λ. We write the reduced cost of y_F as

$$\begin{aligned} w_F - \sum_{v\in V(F)} \check{\mu}_v - \check{\lambda} &= k + |V(F)| - |P(F)| - \sum_{v\in V(F)} \check{\mu}_v - \check{\lambda} \\ &= k + |V(G_{\mathrm{H2}})| - |O(F)| - |P(F)| - \sum_{v\in V(F)} \check{\mu}_v - \check{\lambda} \\ &= k + \sum_{v\in O(F)} \check{\mu}_v - |O(F)| - |P(F)| + |V(G_{\mathrm{H2}})| - \sum_{v\in V(G_{\mathrm{H2}})} \check{\mu}_v - \check{\lambda}. \quad (4.9) \end{aligned}$$

Convert G_{H2} into a directed graph by replacing each edge $uv \in E(G_{\mathrm{H2}})$ with arcs uv and vu. Extend the resulting graph into a directed graph H by adding new vertices r, o, and p, and new arcs ro, pr, and rv, ov, and vp for every $v \in V(G_{\mathrm{H2}})$. Set $b_r = |V(G_{\mathrm{H2}})|$, $b_o = b_p = 0$, and $b_v = -1$ for each $v \in V(G_{\mathrm{H2}})$. Regarding the supply/demand of a vertex v of H as b_v, we can describe a flow in H as follows. An amount of at least $|V(G_{\mathrm{H2}})|$ leaves r and reaches all vertices in $V(G_{\mathrm{H2}})$ by passing through arcs rv for $v \in V(G_{\mathrm{H2}})$ or going via vertex o. One unit of this amount is consumed by each vertex in $V(G_{\mathrm{H2}})$. The remainder returns to r by continuing to arcs vp for $v \in V(G_{\mathrm{H2}})$ and falling into arc pr.

The FPP-MIP is given below. In this program, we employ f variables to determine a flow in H: for each arc uv of H, f_{uv} settles the amount in uv. From this flow, one can derive a forest F of G_{H2} as follows. For each $v \in V(G_{\mathrm{H2}})$, consider v to be outside F if and only if $f_{ov} = 1$ and v to be a peripheral vertex of F if and only if $f_{vp} = 1$ – in case $f_{ov} = f_{vp} = 0$, v is a non-peripheral vertex of F. For each $v \in V(G_{\mathrm{H2}})$ with $f_{ov} = 0$ and $f_{rv} > 0$, build a tree of F from v and every $t \in V(G_{\mathrm{H2}})$ satisfying the subsequent conditions: $f_{ot} = 0$ and there is a directed path P in H from v to t such that, for each arc uw of P, $f_{uw} > 0$, and, for each vertex u of P, u is not yet added to a tree of F and $f_{ru} = 0$ if $u \neq v$.

Moreover, we use $\sum_{v\in V(G_{\mathrm{H2}})} g_{rv}$ to count the number of maximal trees of F. Also, we set $C = |V(G_{\mathrm{H2}})| - \sum_{v \in V(G_{\mathrm{H2}})} \check{\mu}_v - \check{\lambda}$ and write, in the objective function, the reduced cost of the variable y_F (see (4.9)). Finally, we denote by M the maximum flow in H going out from a vertex in $V(G_{\mathrm{H2}})$: $M = 2(|V(G_{\mathrm{H2}})| - 1)$.

$$\min \quad \sum_{v \in V(G_{\mathrm{H2}})} g_{rv} + \sum_{v \in V(G_{\mathrm{H2}})} (\check{\mu}_v - 1) f_{ov} - \sum_{v \in V(G_{\mathrm{H2}})} f_{vp} + C$$

s.t.:

$$\sum_{u \in N_H^+(v)} f_{vu} - \sum_{u \in N_H^-(v)} f_{uv} = b_v, \qquad \forall v \in V(H) \tag{4.10}$$

$$\sum_{u \in N_{G_{\mathrm{H2}}}(v)} f_{vu} + M(f_{ov} + f_{vp}) \leq M, \qquad \forall v \in V(G_{\mathrm{H2}}) \tag{4.11}$$

$$f_{rv} \leq (M+1) g_{rv}, \qquad \forall v \in V(G_{\mathrm{H2}}) \tag{4.12}$$

$$f_{ro},\ f_{pr},\ f_{rv} \geq 0, \qquad \forall v \in V(G_{\mathrm{H2}}) \tag{4.13}$$

$$f_{uv},\ f_{vu} \geq 0, \qquad \forall uv \in E(G_{\mathrm{H2}}) \tag{4.14}$$

$$f_{ov},\ f_{vp},\ g_{rv} \in \{0,1\}, \qquad \forall v \in V(G_{\mathrm{H2}}) \tag{4.15}$$

Constraints (4.10) impose flow conservation in H. Constraints (4.11) state that, for every $v \in V(G_{\mathrm{H2}})$, if $f_{ov} = 1$ or $f_{vp} = 1$, then v sends flow to no other node of H or sends flow only to p. Constraints (4.12) relate, for each $v \in V(G_{\mathrm{H2}})$, the flow in arc rv to the tree-counting variable g_{rv}.

To try to optimize the FPP-MIP faster, we add to it the following constraints. These constraints reduce the set of forest columns to be generated for the MWNS-IP without compromising optimality – if a forest F of G_{H2} is such that $y_F = 1$ in an optimal solution to the MWNS-IP, then one can obtain an optimal solution to the MWNS-IP in which no forest variable has value 1 or one can transform F into a forest F' of G_{H2} such that $y_{F'} = 1$ in an optimal solution to the MWNS-IP and F' corresponds to a solution to constraints (4.10)-(4.20).

$$f_{rv} \geq 4 g_{rv}, \qquad \forall v \in V(G_{\mathrm{H2}}) \tag{4.16}$$

$$f_{ov} + f_{vp} \geq 1, \qquad \forall v \in V(G_{\mathrm{H2}}),\ |N_{G_{\mathrm{H2}}}(v)| = 1 \tag{4.17}$$

$$f_{ou} \leq f_{ov}, \qquad \forall v \in V(G_{\mathrm{H2}}),\ N_{G_{\mathrm{H2}}}(v) = \{u\} \tag{4.18}$$

$$f_{vp} = 0, \qquad \forall v \in V(G_{\mathrm{H2}}),\ u \in N_{G_{\mathrm{H2}}}(v),\ |N_{G_{\mathrm{H2}}}(u)| = 1 \tag{4.19}$$

$$f_{ou} \leq f_{ov} + f_{vp}, \qquad \forall v \in V(G_{\mathrm{H2}}),\ |N_{G_{\mathrm{H2}}}(v)| > 1,\ \forall u \in N_{G_{\mathrm{H2}}}(v) \tag{4.20}$$

Derive as above a forest F of G_{H2} from the flow in H determined by a solution to the FPP-MIP. Constraints (4.16) forbid the maximal trees of F to all have fewer than three vertices. Constraints (4.17) impose that every 1-degree vertex of G_{H2} either is not in F or is a peripheral vertex of F. Constraints (4.18) state that, if a 1-degree vertex of G_{H2} is in F, then its sole neighbor is also in F. Constraints (4.19) say that every vertex of G_{H2} that has a 1-degree neighbor either is not in F or is a non-peripheral vertex of F. Constraints (4.20) imply that, if a vertex of G_{H2} has degree greater than 1 and is a non-peripheral vertex of F, then each of its neighbors is in F.

Prior to solving the forest pricing problem, we execute three greedy methods that aim to build a forest of G_{H2} such that the associated forest variable has negative reduced cost. These methods, denoted by M_A, M_B, and M_C, rely on three sets of vertices, given by $V_1 = \{ v \in V(G_{\mathrm{H2}}) : \check{\mu}_v > 1 \}$, $V_2 = \{ v \in V(G_{\mathrm{H2}}) : 0 < \check{\mu}_v \leq 1 \}$, and $V_0 = V(G_{\mathrm{H2}}) \setminus (V_1 \cup V_2)$.

Suppose that we are constructing a forest F of G_{H2}. For a vertex $v \in V(G_{\mathrm{H2}})$, assume that we set v to be a vertex of F (non-peripheral or peripheral vertex) if $v \in V_1$, a peripheral vertex of F if $v \in V_2$, and a vertex outside F if $v \in V_0$. Regarding the FPP-MIP, this setting is equivalent to the assignment $f_{ov} = 0$ and $f_{vp} = 0$ or $f_{ov} = 0$ and $f_{vp} = 1$ if $v \in V_1$, $f_{ov} = 0$ and $f_{vp} = 1$ if $v \in V_2$, and $f_{ov} = 1$ and $f_{vp} = 0$ if $v \in V_0$. The term $(\check{\mu}_v - 1) f_{ov} - f_{vp}$ of the objective function of the FPP-MIP encodes the *outside-peripheral contribution* of v to the reduced cost of the forest variable y_F. Observe that, under the assignment above, this contribution is less than or equal to 0. The greedy method M_A is based on this observation: the method constructs a forest F of G_{H2} out of the vertices of V_1 seeking to minimize w_F and then tries to include the vertices of V_2 as additional peripheral vertices of the existing trees of F. An improvement step performed in M_A is to remove from F every tree T such that $w_T - \sum_{v \in V(T)} \check{\mu}_v > 0$.

In some of our early experiments, we detected a negative-reduced-cost forest variable y_F with the corresponding forest F of G_{H2} having the following layout, denoted by L: several non-peripheral vertices of F contained in V_2 and several peripheral vertices of F contained in V_1. According to the outside-peripheral contribution of a vertex $v \in V_2$ to the reduced cost of y_F, it is not advantageous to set v to be a non-peripheral vertex of F. However, this setting may pay off when it enables some vertices of V_1 to be added as peripheral vertices of F. The greedy methods M_B and M_C try to generate a forest F of G_{H2} satisfying the layout L above: in M_B, F is constructed as in M_A with sets V_1 and V_2 interchanged; in M_C, the non-peripheral vertices of F come from V_2 and the peripheral vertices of F are both chosen from V_2 and formed of every vertex of V_1 having a neighbor in V_2.

We execute M_C, M_A, and M_B obeying this order and running the next method only when the previous one fails. If any of them succeeds, then the corresponding forest column is added to the RMP. Else, we solve the FPP-MIP. In this case, if the optimum of the FPP-MIP is negative, then we add to the RMP the forest columns derived from the solutions of lowest costs obtained while solving the problem. The number of added columns is a function of the number of vertices of G_{H2} and the number of initial forest columns, a subject addressed in Section 4.6.2.7.

In the process above, we may add to the RMP some forest columns derived from solutions to the FPP-MIP that have non-negative cost. The forest variables corresponding to these columns have non-negative reduced cost and, therefore, do not help improving the optimum of the RMP. Nevertheless, in case one of these variables has later negative reduced cost, we may spare a round of pricing variables and thus of possibly optimizing the FPP-MIP, a task found to be computationally expensive in our preliminary experiments.

4.6.2.4 Lower bound on the optimum of the master problem

Denote by z^*_{RMP} the optimum of the RMP. Multiply every constraint of the MP by -1 times the value in u^*_{RMP} (see the introduction of Section 4.6.2) of the corresponding dual variable – this operation preserves the sign of constraint (4.4) and reverses the sign of constraints (4.7). Add all resulting constraints to the MP's solution-cost equation

$$\sum_{C \in \mathcal{C}} x_C + \sum_{F \in \mathcal{F}} w_F y_F = z_{MP}.$$

It follows that

$$z_{MP} \;\geq\; z^*_{RMP} + \sum_{v \in V(G_{\mathrm{H2}})} \bar{c}(t_v)\; t_v + \sum_{C \in \mathcal{C}} \bar{c}(x_C)\; x_C + \sum_{F \in \mathcal{F}} \bar{c}(y_F)\; y_F, \tag{4.21}$$

where, for each $\pi \in \{\, t_v \;\forall v \in V(G_{\mathrm{H2}}),\, x_C \;\forall C \in \mathcal{C},\, y_F \;\forall F \in \mathcal{F}\,\}$, $\bar{c}(\pi)$ is the reduced cost of the variable π.

When testing if the optimal solution obtained to the RMP is also optimal to the MP, suppose that we proceed to check if the minimum reduced cost $\bar{c}^*_{for}$ of a forest variable is non-negative. In this case, because of our pricing strategy (see Section 4.6.2.1), $\bar{c}(x_C) \geq 0$ for each $C \in \mathcal{C}$; moreover, since the RMP contains all t variables, $\bar{c}(t_v) \geq 0$ for each $v \in V(G_{\mathrm{H2}})$. Within this context, we can rewrite inequality (4.21) as

$$z_{MP} \;\geq\; z^*_{RMP} + \sum_{F \in \mathcal{F}} \bar{c}(y_F)\; y_F \;\geq\; z^*_{RMP} + \bar{c}^*_{for} \sum_{F \in \mathcal{F}} y_F. \tag{4.22}$$

Assume that $\bar{c}^*_{for}$ is negative, and multiply this value by constraint (4.4) – thus, reversing the sign of the constraint – of the MP. By combining the resulting constraint with inequality chain (4.22), we conclude that

$$z_{MP} \geq z^*_{RMP} + \bar{c}^*_{for}.$$

Since this inequality holds for the cost z_{MP} of any solution to the MP, we arrive at a lower bound on the optimum z^*_{MP} of the MP, which is employed in our method:

$$z^*_{MP} \geq z^*_{RMP} + \bar{c}^*_{for}.$$

This lower bound is based on a lower bound highlighted by Desrosiers and Lübbecke [9] for the optimum of a linear program being solved using column generation.

4.6.2.5 Branching strategy

We employ a branching strategy that may cause our method to stop without a proven optimal solution to the MWNS-IP – in Section 4.7.2, we discuss how assuredly solving the MWNS-IP would affect the performance of Heur 2. Before describing this strategy, in which only t variables are branched on, we point out a difficulty of branching on a clique variable; a similar complication stems from branching on a forest variable.

Suppose that we branch on a clique variable x_C – two nodes are created in the enumeration tree, one where x_C is fixed at 0 and other where x_C is fixed at 1. In each node of the enumeration tree where x_C is fixed, we have the task of preventing the clique column corresponding to x_C to be generated again. It seems complex to perform this task when x_C is fixed at 0: this conclusion is drawn from arguments analogous to the ones presented by Mehrotra and Trick [29, Section 4] when considering branching rules for a branch-and-price method to find a minimum coloring of a graph.

As stated above, we branch only on t variables. Denote by s^* the optimal solution found for the MP. For each $v \in V(G_{\text{H2}})$, define $\check{t}_v$ to be the value in s^* of the variable t_v. If $\check{t}_v$ is fractional for some $v \in V(G_{\text{H2}})$, then let $u \in \arg\max_{v \in V(G_{\text{H2}})} \min\{\check{t}_v,\ 1 - \check{t}_v\}$. Else, among every vertex v of G_{H2} such that t_v is a non-fixed variable, take u for a maximum-degree vertex. We branch on the variable t_u; in case $\check{t}_u$ is integral, s^* is still an optimal linear relaxation solution in one of the two nodes created in the enumeration tree.

It may happen that each t variable is already fixed in the current node of the enumeration tree and, after being processed, the node cannot be fathomed. In this case, it would remain to solve two independent problems: a minimum clique-cover problem on $G_{\text{H2}}[V_0]$ and a minimum-weight spanning-forest problem on $G_{\text{H2}}[V_1]$, where $V_0 = \{\, v \in V(G_{\text{H2}}) : t_v \text{ is fixed at } 0 \,\}$ and $V_1 = V(G_{\text{H2}}) \setminus V_0$. We, however, just interrupt the solving process instead.

4.6.2.6 Enumeration-node selection

We use an enumeration-node selection strategy that aims to process first the nodes that are more promising in containing optimal solutions to the MWNS-IP. This strategy is the so-called *best estimate search* [2] and is implemented through the corresponding SCIP plugin [16].

4.6.2.7 Initial solutions and columns

Based on preliminary experiments, we decided to add, as initial solutions to our method, the neocolonization setup corresponding to bound UB-MCD (see Section 4.4.1) and the neocolonization setup $\langle Z^*, Y^* \rangle$ corresponding to bound UB-DNS (see Section 4.4.3) in case Z^* is formed of only non-clique parts. Before setting the method in motion, we do the following as well.

We add a solution s in which all forest variables have value 0. To produce s, we compute a coloring of the complement of G_{H2} seeking to minimize the number of used colors. As a consequence, we obtain a partition $\{\, C_1, C_2, \ldots, C_k \,\}$ of $V(G_{\text{H2}})$ into cliques. The nonzero variables of s are the clique variables $x_{C'_1}, x_{C'_2}, \ldots, x_{C'_k}$, where, for $i = 1, 2, \ldots, k$, C'_i is a maximal clique of G_{H2} that includes C_i.

To compute the coloring above, we employ an implementation provided in SCIP that generates an initial solution in the context of a branch-and-price method to find a minimum coloring of a graph by solving an integer program due to Mehrotra and Trick [29]. In this implementation, a coloring is first constructed by a greedy method. Tabucol – a tabu search algorithm that, although proposed more than 30 years ago, seems to remain popular, probably for being simple and easy to implement and for providing good-quality solutions at a reasonable computational effort [15] – is next run successive times to try to achieve a coloring that uses fewer colors until an improving coloring cannot be found.

We also add a solution s in which all clique variables have value 0. To this end, we solve a minimum-weight spanning-forest problem: a version of the FPP-MIP (see Section 4.6.2.3) with the variable f_{ov} fixed at 0 for every $v \in V(G_{\text{H2}})$. The nonzero variables of s are all t variables and the forest variable y_F, where F is the obtained spanning forest of G_{H2}.

Aiming to create a diversified initial forest column set, we add forest columns that are in some sense complementary to the ones involved in the preceding solutions:

- For each vertex v of G_{H2} with degree at least 2, a forest column corresponding to the tree with an edge vu for each $u \in N(v)$;
- For each edge uv of G_{H2} such that $N(u) \nsubseteq N[v]$ and $N(v) \nsubseteq N[u]$, a forest column corresponding to the tree with an edge uw for each $w \in N(u)$ and an edge vw for each $w \in N(v) \setminus N[u]$.

4.7 Results – Part 2

We now report on computational tests for Heur 2. All bounds already employed in Heur 1 were computed as described in Section 4.5. The calculation of UB-MNS was implemented as follows: whereas the general restricted branch-and-price method was coded in `C/C++` using SCIP 3.2.0 [16], the FPP-MIP (see Section 4.6.2.3) and the linear relaxations were solved through IBM CPLEX 12.5.1 [22]. The instance set and the machine used to carry out the tests were the same as in the computational evaluation of Heur 1 (see Section 4.5).

4.7.1 Time limit for the restricted branch-and-price

We experimented on different time limits for the restricted branch-and-price method used to calculate bound UB-MNS (see Section 4.6). A description of our final experiment follows.

First, we executed Heur 2 for all instances stopping it immediately before the computation of UB-MNS (see Algorithm 4.4). There were 275 unsolved instances. Next, we applied to these unsolved instances a 5-minute and a 1-hour time-limit version of the restricted branch-and-price method. In every run, the best neocolonization setup found by both versions was the same. However, while the 5-minute time-limit version proved this neocolonization setup to have minimum weight in 84.36% of the cases, the 1-hour counterpart did so for 95.27% of the tests.

Recall that the purpose of Heur 2 is to produce, within a reasonable amount of time, a good-quality upper bound on the m-eternal domination number of the input graph. When computing UB-MNS, our goal is not to prove that an obtained neocolonization setup has minimum weight, but just to find one of weight as small as possible. For this reason, we settled on the 5-minute time limit for the restricted branch-and-price method.

4.7.2 Results

As done for Heur 1, we validate Features (i) and (ii) (see Section 4.1) of Heur 2 by assessing the reached gap, that is, H2-UB − H2-LB (see Algorithm 4.4), and the running time. This assessment is performed in comparison with the results for Heur 1, which are given in Section 4.5.2. We make use of the same instance aspects considered before.

With respect to the gap, we can state the following. The results for Heur 2 discussed next are presented in Tables 4.5 to 4.8, where we show how the gap values obtained by the method distribute over each order, density, p, and t interval (see Section 4.5.1), respectively. Additionally, in the last three columns of Table 4.6, we count for how many times each of the bounds UB-MCD, UB-DNS, and UB-MNS is the origin of H2-UB.

For all order, p, and t intervals but T_5, T_6, and T_7, Heur 2 improved on the results of Heur 1. Accordingly, for all order, p, and t intervals except P_6, T_6, and >2h, the percentage of cases of gap at most 1 increased from at least 96% to at least 98.67%. The exceptions of intervals P_6, T_6, and >2h can be explained as in Section 4.5.2.

As for the density intervals, the results were improved on for intervals $(0\%, 20\%]$ and $(20\%, 40\%]$. As a consequence, the contrast between interval $(0\%, 20\%]$ and the other ones was significantly diminished. The percentage of cases of gap greater than 1 was reduced from at most 0.83% to 0% for intervals $(20\%, 40\%]$ to $(80\%, 100\%]$ and from 14.17% to 3.33% for interval $(0\%, 20\%]$. The impact of UB-MNS on the improvement for interval $(0\%, 20\%]$ can be quantified through the percentage of tests for which each employed upper bound was the origin of H2-UB: 69.17% for UB-MCD, 14.17% for UB-DNS, and 16.66% for UB-MNS.

In general, the gap found by Heur 2 was better. The percentage of instances that were solved to optimality increased from 61% to 62.5%. Moreover, the gap was at most 1 in 99.33% instead of in 97% of the tests, and, in the four remaining cases, the gap was 2.

In regard to the running time, the results were worse in general. The percentage of runs of at most 1 second decreased from 72.50% to 63.66%. Furthermore, the average running time rose from 18.67 to 52.77 seconds, and the proportion of running time values not greater than the mean went from 90% to 83.17%. The maximum running time grew from 308.32 to 608.32 seconds even though Heur 2 took at most 308.32 seconds to execute for 93.33% of the instances.

Based on the analysis above, we draw the following conclusions. By employing Heur 2, one can achieve bold gap results – overall maximum gap 2 and gap at most 1 for 99.33% of the tests – but has to afford larger running times. By using Heur 1, one can still obtain good gap results at a smaller time cost.

Lastly, we make a relevant investigation. Consider a version Heur 2* of Heur 2 in which the MWNS-IP is solved to optimality and the value of UB-MNS is the minimum weight $\theta_C(G_{\text{H2}})$ of a neocolonization setup of the input graph G_{H2} – recall that, in Heur 2 (see Section 4.6), a restricted branch-and-price method is applied to the MWNS-IP and the value of UB-MNS is greater than or equal to $\theta_C(G_{\text{H2}})$. We examine how Heur 2* would perform in comparison with Heur 2.

We report that, for only 7 of the 600 instances, the value of UB-MNS was computed and not proved to equal $\theta_C(G_{\text{H2}})$. Hence, Heur 2* might improve on H2-UB for just

	Gap		
	0	1	2
$[10, 20]$	106	43	1
$[21, 30]$	94	55	1
$[31, 40]$	86	62	2
$[41, 50]$	89	61	0

Table 4.5: Gap (H2-UB − H2-LB) per order intervals.

	Gap			H2-UB			
	0	1	2	(UB-)	MCD	DNS	MNS
(0%, 20%]	46	70	4		83	17	20
(20%, 40%]	85	35	0		114	3	3
(40%, 60%]	83	37	0		119	1	0
(60%, 80%]	90	30	0		120	0	0
(80%, 100%]	71	49	0		119	1	0

Table 4.6: Gap (H2-UB − H2-LB) and H2-UB origin per density intervals.

	Gap		
	0	1	2
P_1	43	32	0
P_2	91	95	1
P_3	49	33	1
P_4	121	36	0
P_5	44	16	0
P_6	27	9	2

Table 4.7: Gap (H2-UB − H2-LB) per intervals of number (p) of γ_m^∞-size dominating sets.

	Gap		
	0	1	2
T_1	115	102	1
T_2	69	55	1
T_3	57	25	0
T_4	75	25	0
T_5	25	5	0
T_6	12	0	1
T_7	7	1	0
>2h	15	8	1

Table 4.8: Gap (H2-UB − H2-LB) per intervals of time (t) to compute γ_m^∞.

1.17% of the instances. Moreover, Heur 2* would likely run in substantially more time in general compared with Heur 2, which could be prohibitive to using Heur 2* for practical applications of the m-EDSP.

4.8 Results – Part 3

In order to discuss the limitations of Heur 1 and Heur 2, we analyze results obtained using instances for which the methods were likely to perform worse. Heur 1 and Heur 2 were executed under the settings stated in Sections 4.5 and 4.7 with the following change: to ensure reasonable running times for the methods, the sum of the times for computing UB-MCD and UB-DNS was limited to 5 minutes. The computational environment was the same as before (see Section 4.5).

4.8.1 Instances

The instances consist of 150 Erdős-Rényi random graphs forced to be connected, restricted to have density at most 20%, and separated into groups g_1, g_2, ..., g_5. For $i = 1, 2, \ldots, 5$, group g_i contains 30 graphs with order in the interval $[51 + 10(i-1), 60 + 10(i-1)]$. Compared with the instances presented in Section 4.5.1, the current instances are given by graphs that are larger in order and have their density concentrated into the interval $(0\%, 20\%]$.

The previous results for Heur 1 and Heur 2 (see Sections 4.5.2 and 4.7.2) imply that the upper bounds computed by the methods are likely to succeed the least for graphs of density at most 20%. Furthermore, the problems solved by Heur 1 and Heur 2 are likely to demand more computational effort for graphs of larger order. In this case, the methods are expected to run in more time and to obtain worse-quality values for the bounds computed with time limit – the values obtained for these bounds being more distant from the values attainable without the fixed time limits. Therefore, Heur 1 and Heur 2 are likely to perform worse for the current instances.

4.8.2 Results

As before, we analyze the achieved gap, i.e., H1-UB − H1-LB for Heur 1 and H2-UB − H2-LB for Heur 2 (see Algorithms 4.2 and 4.4), and the running time. In what follows, this analysis is done by comparing the results for the current instance set, namely, the graphs described in Section 4.8.1, with the results for the previous instance set, that is, the graphs introduced in Section 4.5.1.

The gap values outputted by Heur 1 and Heur 2 for the current instance set are displayed, respectively, in Tables 4.9 and 4.10. The first columns of these tables give the number of instances per gap value. The last columns of the former and of the latter table present the number of cases in which each employed upper bound is the origin of H1-UB and H2-UB, accordingly.

Compared with the results for the previous instance set, the gap and the run time obtained by Heur 1 and Heur 2 for the current instance set were substantially worse in general. Regarding the gap, even in comparison with only the instances of density at most 20% of the previous instance set, the results of the methods for the current instances were worse. It is worth noting that the comparison of results between Heur 1 and Heur 2 is considerably less advantageous to Heur 2 when done for the current instance set. The fact that UB-MNS surpassed the other upper bounds for only 1.33% of the current instances explains this loss of advantage.

To illustrate the performance deterioration of the methods, we address the results of Heur 1 for the current in relation to the previous instance set. As for the gap, the percentages of instances that were solved to optimality and of cases of gap at most 1 decreased from 61% to 26% and from 97% to 82%, respectively. The maximum gap rose from 3 to 6, there being cases of each gap in the range $[0, 6]$. As for the running time, the percentage of executions of at most 1 second dropped from 72.5% to only 2.66%, and the average running time increased from 18.67 to 254.69 seconds. The proportion of running time values not greater than the mean reduced from 90% to 27.33%, and the maximum running time went from 308.32 to 621.49 seconds.

Recall that the bounds computed with time limit are UB-MCD, UB-DNS, UB-MNS, and LB-TLE. With respect to the values obtained for these bounds being more distant from the values attainable without the time limits, we report the following related results. We compare the impact of larger time limits for the previous and for the current instance set. For UB-MCD, UB-DNS, and UB-MNS, the influence of a 1-hour time limit would be null for both instance sets: in every test, each of these upper bounds was already at

	Gap							H1-UB		
	0	1	2	3	4	5	6	(UB-)	MCD	DNS
# of instances	39	84	15	5	3	3	1		147	3

Table 4.9: Gap (H1-UB − H1-LB) and H1-UB origin for graphs with larger order and density at most 20%.

	Gap							H2-UB			
	0	1	2	3	4	5	6	(UB-)	MCD	DNS	MNS
# of instances	39	84	16	5	3	2	1		146	2	2

Table 4.10: Gap (H2-UB − H2-LB) and H2-UB origin for graphs with larger order and density at most 20%.

its minimum or would not be improved. In contrast, for LB-TLE, the effect of a 2-hour time limit would be greater for the current instance set: the bound would be improved, accordingly, in 0.36% and in 14.03% of the cases in which it was computed.

From the results above, we conclude that Heur 1 and Heur 2 performed significantly worse for the current instance set. In addition, these results put light on a limitation of our approach: for instances given by graphs of large enough order, the gap outputted by Heur 1 and Heur 2 is likely to not be efficacious as a consequence of worse-quality values derived from the time-limited parts of the methods.

4.9 Conclusion and future work

In this paper, we addressed an issue that is fundamental to practical applications of the m-eternal dominating set problem but that has received relatively little attention. Our goal was to obtain not only a good-quality upper bound on the m-eternal domination number but also an associated efficient strategy of defense. To this end, we introduced two heuristic methods, in which we proposed and solved integer and constraint programming models to compute bounds on the m-eternal domination number. We performed an extensive experiment to validate the features of these methods. As a consequence, we carried out a practical evaluation of bounds for the m-eternal domination number available in the literature. Furthermore, we generated a benchmark that has 750 instances and varies with respect to the best known solution: there are cases of proven optimal solution, of small bound gap, and of larger bound gaps. Finally, we proved that the decision version of the m-eternal dominating set problem is NP-hard.

As a future work, we aim to develop improved heuristic methods as follows. Each currently used upper bound is derived from the weight of a neocolonization setup, which is calculated considering only defenses executed by guards moving along paths. By also employing upper bounds drawn from defenses involving cyclic movements, we expect to achieve better results. For an illustration, take the n-vertex hole H_n with $n \geq 5$. The minimum weight of a neocolonization setup of H_n is $\lceil n/2 \rceil$. By moving together either clockwise or counterclockwise during each defense, guards in less number, namely, $\lceil n/3 \rceil$, can eternally defend H_n.

Acknowledgements

This work was funded by grants 141964/2013-8, 311373/2015-1, 425340/2016-3, and 306454/2018-1 from *Conselho Nacional de Desenvolvimento Científico e Tecnológico* (CNPq, Brazil) and by grant 2015/11937-9 from *Fundação de Amparo à Pesquisa do Estado de São Paulo* (FAPESP, Brazil).

Bibliography

[1] W. Abbas, S. Bhatia, and X. Koutsoukos. Guarding networks through heterogeneous mobile guards. In *Proceedings of the 2015 American Control Conference*, pages 3428–3433, 2015.

[2] T. Achterberg. *Constraint Integer Programming.* PhD thesis, Technische Universität Berlin, Berlin, 2007.

[3] M. Bergner, M. Lübbecke, and J. Witt. A branch-price-and-cut algorithm for packing cuts in undirected graphs. *Journal of Experimental Algorithmics*, 21:1.2:1–1.2:16, 2016.

[4] A. Bley, N. Boland, C. Fricke, and G. Froyland. A strengthened formulation and cutting planes for the open pit mine production scheduling problem. *Computers & Operations Research*, 37(9):1641–1647, 2010.

[5] N. Boland, A. Bley, C. Fricke, G. Froyland, and R. Sotirov. Clique-based facets for the precedence constrained knapsack problem. *Mathematical Programming*, 133(1): 481–511, 2012.

[6] R. Borndörfer and Z. Kormos. An algorithm for maximum cliques. Unpublished working paper, Konrad-Zuse-Zentrum für Informationstechnik Berlin, 1997.

[7] A. Braga, C. de Souza, and O. Lee. The eternal dominating set problem for proper interval graphs. *Information Processing Letters*, 115(6–8):582–587, 2015.

[8] E. Chambers, B. Kinnersley, and N. Prince. Mobile eternal security in graphs. Manuscript, 2006.

[9] J. Desrosiers and M. Lübbecke. A primer in column generation. In *Column Generation*, pages 1–32. Springer, 2005.

[10] S. Finbow, M.-E. Messinger, and M. van Bommel. Eternal domination on $3 \times n$ grid graphs. *Australasian Journal of Combinatorics*, 61(2):156–174, 2015.

[11] F. Fomin, S. Gaspers, P. Golovach, D. Kratsch, and S. Saurabh. Parameterized algorithm for eternal vertex cover. *Information Processing Letters*, 110(16):702–706, 2010.

[12] F. Fomin, P. Golovach, and D. Lokshtanov. Guard games on graphs: Keep the intruder out! *Theoretical Computer Science*, 412(46):6484–6497, 2011.

[13] F. Fomin, P. Golovach, and P. Prałat. Cops and robber with constraints. *SIAM Journal on Discrete Mathematics*, 26(2):571–590, 2012.

[14] F. Fomin, F. Giroire, A. Jean-Marie, D. Mazauric, and N. Nisse. To satisfy impatient web surfers is hard. *Theoretical Computer Science*, 526:1–17, 2014.

[15] P. Galinier and A. Hertz. A survey of local search methods for graph coloring. *Computers & Operations Research*, 33(9):2547–2562, 2006.

[16] G. Gamrath, T. Fischer, T. Gally, A. Gleixner, G. Hendel, T. Koch, S. Maher, M. Miltenberger, B. Müller, M. Pfetsch, C. Puchert, D. Rehfeldt, S. Schenker, R. Schwarz, F. Serrano, Y. Shinano, S. Vigerske, D. Weninger, M. Winkler, J. Witt, and J. Witzig. The SCIP optimization suite 3.2. Technical Report 15-60, ZIB, Berlin, 2016.

[17] M. Garey and D. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W. H. Freeman, 1979.

[18] W. Goddard, S. Hedetniemi, and S. Hedetniemi. Eternal security in graphs. *Journal of Combinatorial Mathematics and Combinatorial Computing*, 52:169–180, 2005.

[19] J. Goldwasser, W. Klostermeyer, and C. Mynhardt. Eternal protection in grid graphs. *Utilitas Mathematica*, 91:47–64, 2013.

[20] Gurobi Optimization. Gurobi Optimizer. `https://www.gurobi.com`.

[21] IBM. IBM CPLEX CP Optimizer. `https://www.ibm.com/analytics/cplex-cp-optimizer`.

[22] IBM. IBM CPLEX Optimizer. `https://www.ibm.com/analytics/cplex-optimizer`.

[23] S. Khaitan and J. McCalley. Design techniques and applications of cyberphysical systems: A survey. *IEEE Systems Journal*, 9(2):350–365, 2015.

[24] W. Klostermeyer and G. MacGillivray. Eternal dominating sets in graphs. *Journal of Combinatorial Mathematics and Combinatorial Computing*, 68:97–111, 2009.

[25] W. Klostermeyer and C. Mynhardt. Graphs with equal eternal vertex cover and eternal domination numbers. *Discrete Mathematics*, 311(14):1371–1379, 2011.

[26] W. Klostermeyer and C. Mynhardt. Vertex covers and eternal dominating sets. *Discrete Applied Mathematics*, 160(7–8):1183–1190, 2012.

[27] W. Klostermeyer and C. Mynhardt. Protecting a graph with mobile guards. *Applicable Analysis and Discrete Mathematics*, 10(1):1–29, 2016.

[28] M. Mamino. On the computational complexity of a game of cops and robbers. *Theoretical Computer Science*, 477:48–56, 2013.

[29] A. Mehrotra and M. Trick. A column generation approach for graph coloring. *INFORMS Journal on Computing*, 8(4):344–354, 1996.

[30] M. Reis, O. Lee, and F. Usberti. Flow-based formulation for the maximum leaf spanning tree problem. *Electronic Notes in Discrete Mathematics*, 50:205–210, 2015.

[31] D. West. *Introduction to Graph Theory.* Prentice Hall, 2001.

[32] L. Wolsey. *Integer Programming.* Wiley, 1998.

Chapter 5

Conclusion

In this work, we disproved a result on the m-eternal dominating set problem and examined the problem for two specific classes of graphs: Cayley graphs and proper interval graphs. Additionally, we addressed an issue that is fundamental to practical applications of the m-eternal dominating set problem but that has been given relatively little attention: to provide an efficient strategy of defense for the guards. Furthermore, we developed and extensively experimented heuristic methods for the m-eternal dominating set problem. Finally, we proposed and implemented an exact algorithm for the m-eternal dominating set problem and proved that its decision version is NP-hard.

Goddard et al. [20] published that, for Cayley graphs, the m-eternal domination number $\gamma_{m,1}^{\infty}$ was equal to the minimum size γ of a dominating set. In our study of Cayley graphs, we disproved this result and established its validity for a subclass of these graphs. Moreover, we performed an experimental analysis of the parameters $\gamma_{m,1}^{\infty}$ and γ for Cayley graphs. We showed that the difference between these parameters can be indefinitely increased for disconnected Cayley graphs, but left open the question of whether it can be greater than 1 if connectivity is enforced. Future work includes going further into this open question, checking if the equality $\gamma_{m,1}^{\infty} = \gamma$ holds for other classes of vertex-transitive graphs, and investigating if $\gamma_{m,1}^{\infty}$ can be computed in polynomial time for Cayley graphs.

In our research on proper interval graphs, we demonstrated several results on the m-eternal dominating set problem restricted to these graphs: the m-eternal domination number equals the maximum size of an independent set, the m-eternal domination number can be computed in linear time, there is no advantage in allowing multiple guards to occupy the same vertex, and there is a straightforward strategy of defense for the smallest possible number of guards. In addition, we gave a lower bound on $\gamma_{m,m}^{\infty}$ that may also be useful when delving into other classes of graphs. Recently, Rinemberg and Soulignac [33] dealt with the m-eternal dominating set problem for the broader class of interval graphs. It seems promising to continue studying the problem for related classes of graphs, such as the proper circular-arc and circular-arc graphs.

Using integer and constraint programming, we developed heuristic methods suited for practical applications of the m-eternal dominating set problem. Through extensive experimentation, these methods were shown to run in a reasonable time, produce a good-quality upper bound on the m-eternal domination number, and output a structure from which one can derive an associated efficient strategy of defense. To our knowledge, neither implementations gathering such features nor experiments reaching such extension have been reported before for the m-eternal dominating set problem. It is worth noting that we generated a 750-instance benchmark, performed a practical evaluation of bounds for the m-eternal domination number available in the literature, and, as far as we know, presented the first integer program for the problem of determining a minimum-weight neocolonization of a graph.

Attempts to improve the heuristic methods above should consider the following. The heuristic methods consist in computing several upper and lower bounds on the m-eternal domination number. Each of the upper bounds is derived from the weight of a neocolonization setup, which is calculated taking into account only defenses executed by guards moving along paths. Upper bounds drawn from defenses involving cyclic movements may require fewer guards and, therefore, their use may narrow the bound gap outputted by the heuristic methods.

Based on our information, we first implemented an exact algorithm for the m-eternal dominating set problem. We also developed a time-limited counterpart that provides a lower bound on the m-eternal domination number. A future enhancement to the presented algorithms is to employ parallelization to reduce execution times. Within the main loop of the algorithms, one can parallelize the processes of determining the edges of the initial dominating-set graph and finding the unsafe vertices of the subsequent dominating-set graphs.

To advance the knowledge on the complexity of the m-eternal dominating set problem, a next step is to further understand the relationship between the problem and the complexity classes NP and PSPACE. For this purpose, the following questions should be answered – the latter question is mentioned by Klostermeyer and Mynhardt [28]. Is it possible to describe a solution to the m-eternal dominating set problem so that one can check if the solution is indeed valid within time polynomial in the order of the input graph? Given a graph G of order n, is there a polynomial function $f(n)$ such that, if guards can defend any sequence of $f(n)$ attacks to G, then they can eternally defend G?

Bibliography

[1] W. Abbas, S. Bhatia, and X. Koutsoukos. Guarding networks through heterogeneous mobile guards. In *Proceedings of the 2015 American Control Conference*, pages 3428–3433, 2015.

[2] S. Bard, C. Duffy, M. Edwards, G. MacGillivray, and F. Yang. Eternal domination in split graphs. *Journal of Combinatorial Mathematics and Combinatorial Computing*, 101:121–130, 2017.

[3] I. Beaton, S. Finbow, and J. MacDonald. Eternal domination numbers of $4 \times n$ grid graphs. *Journal of Combinatorial Mathematics and Combinatorial Computing*, 85: 33–48, 2013.

[4] G. Birkhoff. *Lattice Theory.* American Mathematical Society, 1948.

[5] A. Braga, C. de Souza, and O. Lee. The eternal dominating set problem for proper interval graphs. *Information Processing Letters*, 115(6–8):582–587, 2015.

[6] A. Braga, C. de Souza, and O. Lee. Computing bounds for eternal domination. In *Book of Abstracts of the 13th Cologne-Twente Workshop on Graphs & Combinatorial Optimization*, pages 177–180, 2015.

[7] A. Braga, C. de Souza, and O. Lee. A note on the paper "Eternal Security in Graphs" by Goddard, Hedetniemi, and Hedetniemi (2005). *Journal of Combinatorial Mathematics and Combinatorial Computing*, 96:13–22, 2016.

[8] A. Burger, E. Cockayne, W. Gründlingh, C. Mynhardt, J. van Vuuren, and W. Winterbach. Infinite order domination in graphs. *Journal of Combinatorial Mathematics and Combinatorial Computing*, 50:179–194, 2004.

[9] E. Chambers, B. Kinnersley, and N. Prince. Mobile eternal security in graphs. Manuscript, 2006.

[10] N. Cohen, N. Martins, F. Mc Inerney, N. Nisse, S. Pérennes, and R. Sampaio. Spy-game on graphs: Complexity and simple topologies. *Theoretical Computer Science*, 725:1–15, 2018.

[11] N. Cohen, F. Mc Inerney, N. Nisse, and S. Pérennes. Study of a combinatorial game in graphs through linear programming. *Algorithmica*, pages 1–33, 2018.

[12] S. Finbow, M.-E. Messinger, and M. van Bommel. Eternal domination on $3 \times n$ grid graphs. *Australasian Journal of Combinatorics*, 61(2):156–174, 2015.

[13] S. Finbow, S. Gaspers, M.-E. Messinger, and P. Ottaway. A note on the eternal dominating set problem. *International Journal of Game Theory*, 47(2):543–555, 2018.

[14] F. Fomin, S. Gaspers, P. Golovach, D. Kratsch, and S. Saurabh. Eternal vertex cover. Manuscript, 2010.

[15] F. Fomin, S. Gaspers, P. Golovach, D. Kratsch, and S. Saurabh. Parameterized algorithm for eternal vertex cover. *Information Processing Letters*, 110(16):702–706, 2010.

[16] F. Fomin, P. Golovach, and D. Lokshtanov. Guard games on graphs: Keep the intruder out! *Theoretical Computer Science*, 412(46):6484–6497, 2011.

[17] F. Fomin, P. Golovach, and P. Prałat. Cops and robber with constraints. *SIAM Journal on Discrete Mathematics*, 26(2):571–590, 2012.

[18] F. Fomin, F. Giroire, A. Jean-Marie, D. Mazauric, and N. Nisse. To satisfy impatient web surfers is hard. *Theoretical Computer Science*, 526:1–17, 2014.

[19] M. Garey and D. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness.* W. H. Freeman, 1979.

[20] W. Goddard, S. Hedetniemi, and S. Hedetniemi. Eternal security in graphs. *Journal of Combinatorial Mathematics and Combinatorial Computing*, 52:169–180, 2005.

[21] J. Goldwasser, W. Klostermeyer, and C. Mynhardt. Eternal protection in grid graphs. *Utilitas Mathematica*, 91:47–64, 2013.

[22] M. Henning, W. Klostermeyer, and G. MacGillivray. Bounds for the m-eternal domination number of a graph. *Contributions to Discrete Mathematics*, 12(2):91–103, 2017.

[23] W. Klostermeyer and G. MacGillivray. Eternal dominating sets in graphs. *Journal of Combinatorial Mathematics and Combinatorial Computing*, 68:97–111, 2009.

[24] W. Klostermeyer and G. MacGillivray. Eternal domination in trees. *Journal of Combinatorial Mathematics and Combinatorial Computing*, 91:31–50, 2014.

[25] W. Klostermeyer and C. Mynhardt. Graphs with equal eternal vertex cover and eternal domination numbers. *Discrete Mathematics*, 311(14):1371–1379, 2011.

[26] W. Klostermeyer and C. Mynhardt. Vertex covers and eternal dominating sets. *Discrete Applied Mathematics*, 160(7–8):1183–1190, 2012.

[27] W. Klostermeyer and C. Mynhardt. Protecting a graph with mobile guards. *arXiv e-prints*, 2014. arXiv:1407.5228v1.

[28] W. Klostermeyer and C. Mynhardt. Protecting a graph with mobile guards. *Applicable Analysis and Discrete Mathematics*, 10(1):1–29, 2016.

[29] W. Klostermeyer, M. Lawrence, and G. MacGillivray. Dynamic dominating sets: the eviction model for eternal domination. *Journal of Combinatorial Mathematics and Combinatorial Computing*, 97:247–269, 2016.

[30] J. Křišťan. Eternal domination on special graph classes. Bachelor's thesis, Czech Technical University in Prague, Prague, 2018.

[31] I. Lamprou, R. Martin, and S. Schewe. Eternally dominating large grids. *Theoretical Computer Science (in press)*, 2018.

[32] M. Mamino. On the computational complexity of a game of cops and robbers. *Theoretical Computer Science*, 477:48–56, 2013.

[33] M. Rinemberg and F. Soulignac. The eternal dominating set problem for interval graphs. *Information Processing Letters*, 146:27–29, 2019.

[34] C. van Bommel and M. van Bommel. Eternal domination numbers of $5 \times n$ grid graphs. *Journal of Combinatorial Mathematics and Combinatorial Computing*, 97: 83–102, 2016.

[35] D. West. *Introduction to Graph Theory.* Prentice Hall, 2001.

[36] L. Wolsey. *Integer Programming.* Wiley, 1998.