FACULTY
OF INFORMATION
TECHNOLOGY
CTU IN PRAGUE

ASSIGNMENT OF BACHELOR'S THESIS

Title:	Eternal domination on special graph classes
Student:	Jan Matyáš Křišťan
Supervisor:	RNDr. Tomáš Valla, Ph.D.
Study Programme:	Informatics
Study Branch:	Computer Science
Department:	Department of Theoretical Computer Science
Validity:	Until the end of summer semester 2018/19

Instructions

Two players play a game on a graph G: first player controls a set of guards on vertices and second player keeps attacking vertices.
After each attack the first player must move the guards along the edges of G such that the attack is supressed.
If the graph can be guarded forever, we say it is eternally dominated and the task usually is to establish the minimum number of guards needed.
This problem has recently received lot of attention in the international community due to its relation with graph parameters like chromatic, domination, clique covering number, and others.
The goals of the thesis are:
1) to survey past results in the field
2) to implement several previously known efficient algorithms for certain graph classes
3) try to attack the problem of establishing eternal domination number for several graph classes like cactus graph and others

References

Will be provided by the supervisor.

doc. Ing. Jan Janoušek, Ph.D.
Head of Department

doc. RNDr. Ing. Marcel Jiřina, Ph.D.
Dean

Prague January 25, 2018

FACULTY
OF INFORMATION
TECHNOLOGY
CTU IN PRAGUE

Bachelor's thesis

Eternal domination on special graph classes

Jan Matyáš Křišťan

Department of Theoretical Computer Science
Supervisor: RNDr. Tomáš Valla, Ph.D

May 14, 2018

Acknowledgements

First and foremost, I thank my supervisor Tomáš Valla for his enthusiasm and interest and for introducing me to many interesting topics. I also thank all my friends for listening to me talk about something they had no interest in and often did not make sense. Namely I thank Josef Erik Sedláček, Tung Anh Vu, Jan Uhlík, Samuel Křišťan, Miroslav Sochor, Václav Blažej, Martin Bobek, Honza Vu and Michal Cvach either for their helpful discussions or their support during the studies. I also thank my family for supporting me during the studies.

Declaration

I hereby declare that the presented thesis is my own work and that I have cited all sources of information in accordance with the Guideline for adhering to ethical principles when elaborating an academic final thesis.

I acknowledge that my thesis is subject to the rights and obligations stipulated by the Act No. 121/2000 Coll., the Copyright Act, as amended, in particular that the Czech Technical University in Prague has the right to conclude a license agreement on the utilization of this thesis as school work under the provisions of Article 60(1) of the Act.

In Prague on May 14, 2018 ....

Citation of this thesis

Křišťan, Jan Matyáš. *Eternal domination on special graph classes.* Bachelor's thesis. Czech Technical University in Prague, Faculty of Information Technology, 2018.

Abstract

In this thesis, we study the m-eternal domination problem. Given graph G, guards are placed on vertices of G. Then vertices are subject to sequential attacks. After each attack, a guard must move into the attacked vertex. At most one guard is allowed to occupy any vertex. We denote the minimum number of guards, that can defend G indefinitely as $\gamma_m^\infty(G)$.

We consider cactus graphs G, such that every edge in G is on a cycle of size $3k + 1$ for some $k \in \mathbb{N}$. We show that for every such G on n vertices, $\gamma_m^\infty(G) = 1 + (n - 1)/3$.

We introduce the m-eternal guard configuration problem, being the same as the m-eternal domination problem, except it allows multiple guards on single vertex. We denote the minimum number of required guards in G as $\Gamma_m^\infty(G)$. We present a linear algorithm for computing $\Gamma_m^\infty(G)$ in cactus graphs, where every articulation is in two blocks. Moreover, we design a linear-time algorithm for computing γ_m^∞ in clique trees. We include a C++ implementation of these algorithms, together with an exponential algorithm for both problems in general graphs.

Keywords eternal domination, graph protection, cactus graph, clique tree, combinatorial game, eternal security

Abstrakt

V této práci studujeme problém věčné dominace grafu, který je známý pod názvem m-eternal domination problem. Je zadán graf G a na vrcholy G umístíme ochránce. Následně jsou na vrcholy postupně vedeny útoky. Po každém útoku se musí nějaký ochránce přesunout na ohrožený vrchol. Každý vrchol smí okupovat nejvýše jeden ochránce. Nejmenší počet ochráců, který ochrání G, značíme $\gamma_m^\infty(G)$.

Zabýváme se kaktusovými grafy G takovými, že každá hrana v G je na cyklu o velikosti $3k+1$ pro nějaké $k \in \mathbb{N}$. Ukazujeme, že pro každé takové G na n vrcholech platí $\gamma_m^\infty(G) = 1 + (n-1)/3$.

Představujeme problém m-eternal guard configuration, který je stejný jako m-eternal domination problem, ale povoluje více ochránců na jednom vrcholu. Nejmenší nutný počet ochránců pro graf G označujeme jako $\Gamma_m^\infty(G)$. Popisujeme lineární algoritmus pro výpočet $\Gamma_m^\infty(G)$ v kaktusových grafech, kde každá artikulace je ve dvou blocích. Navíc předkládáme lineární algoritmus pro výpočet $\gamma_m^\infty(G)$ v klikových stromech. Přikládáme implementaci v C++ těchto algoritmů spolu s exponenciálním algoritmem, který řeší oba problémy v obecných grafech.

Klíčová slova věčná dominace, eternal domination, bránění grafu, kaktusový graf, klikový strom, kombinatorická hra, věčné zabezpečení

Contents

List of Figures

Chapter 1

Introduction

The problem which we explore in this paper can be described as a combinatorial game played on a graph. The first player controls a set of guards, which he initially places on vertices of the graph. The second player repeatedly chooses one vertex, which he attacks. The first player must defend against the attack by moving one of his guards to the attacked vertex. During his turn, he can move each of his guards past at most one edge. While one of the guards must move to the attacked vertex, the others can take a different position in order to prepare for future attacks. If the configuration of guards can defend against any infinite sequence of attacks, we say that the configuration is eternally dominating. We are interested in finding the smallest such configuration.

We study two different models in this text. The first model is commonly referred to as the m-eternal domination problem and allows only one guard on a vertex at a time. Guards can therefore be understood as a set of vertices D. If it is possible for the guards to defend against any infinite sequence of attacks with D as its configuration, D is an m-eternal dominating set. The second studied model relaxes this condition and permits multiple guards on one vertex. We refer to this model as the m-eternal guard configuration. Some other texts also refer to the m-eternal domination problem as the eternal secure set problem or “all-guards move” model [1].

Both of those models can, for example, be used to model a strategy of soldiers defending a city against quick guerilla attacks, if we assume that each attack can be defended before another one appears. Alternatively, the guards can represent a set of firefighters, extinguishing fires appearing throughout a city.

We present the definitions used in this text. All graphs $G = (V, E)$ in this text are undirected, unless stated otherwise. $V(G)$ is the set of vertices of G. $E(G)$ is the set of edges of G. $N(v)$ denotes the set of neighbors of $v \in V$, also $N[v] = N(v) \cup \{v\}$. We say that $D \subseteq V$ is a *dominating set* if every vertex not in D has some neighbor in D. The size of the smallest dominating set on G is denoted by $\gamma(G)$.

The m-eternal domination problem is concerned with finding the smallest possible m-eternal dominating set, which can be defined as follows. A set of vertices $D \subseteq V$ is an *m-eternal dominating set* on $G = (V, E)$ if these following conditions are satisfied:

- D is a dominating set.
- For every $v \in V \setminus D$, there exists D' such that D' is an m-eternal dominating set, $v \in D'$ and there exists some bijection $f : D \to D'$ which satisfies that for every $u \in D : f(u) \in N[u]$.

The size of the smallest m-eternal dominating set on G is denoted by $\gamma_m^\infty(G)$.

We present a definition of the m-eternal guard configuration, the second studied model in this text. Let P be a finite set of guards, then an m-eternal guard configuration on G is a function $g : P \to V$ satisfying the following:

- $g(P)$ is a dominating set.
- For every $v \in V \setminus g(P)$, there exists some $g' : P \to V$ such that g' is an m-eternal guard configuration, $g^{-1}(v) \in P$ and for every $p \in P$, $g'(p) \in N[g(p)]$.

We denote the size of the smallest set P, such that there is an m-eternal guard configuration for P on G, as $\Gamma_m^\infty(G)$.

$G[U]$ is the subgraph of G induced by the set of vertices $U \subseteq V$. We say that a vertex is *protected* in respect to some $U \subseteq V$ if it has some neighbor in U or is itself in U. $C(G)$ is the set of all cycles in G. A *block* or a *biconnected component* of graph G is a maximal biconnected subgraph of G.

Leaf vertex is a vertex with degree 1. A cycle in G is a *leaf cycle* if exactly one of its vertices has degree greater than 2. Similarly, we say that induced subgraph H of G is a *leaf clique*, if for every $v \in V(G) \setminus V(H)$, graph induced by $V(H) \cup \{v\}$ is not a clique and exactly one vertex in H has degree greater than $|V(H)| - 1$. P_n denotes a path on n edges, therefore on $n - 1$ vertices.

Cactus is a graph that is connected and its every edge lies on at most one cycle. An equivalent definition is that it is connected and any two cycles have at most one vertex in common. *Clique tree* is a graph in which every biconnected component is a clique. For example, every tree is a clique tree. *Noose* is a graph, which is a cycle with a set of pairwise disjoint cliques, such that each of them shares exactly one vertex with the cycle.

The m-eternal domination problem was first introduced by Goddard et. al. [2]. A lot of research has focused on finding bounds of γ_m^∞ in different conditions or graph classes. Goddard et al. [2] determine γ_m^∞ exactly for paths, cycles, complete graphs and complete bipartite graphs. We list the results in a brief form below:

- $\gamma_m^\infty(P_n) = \lceil n/2 \rceil$

- $\gamma_m^\infty(C_n) = \lceil n/3 \rceil$
- $\gamma_m^\infty(K_n) = 1$
- $\gamma_m^\infty(K_{m,n}) = 2$

Another often studied model, often referred to as eternal domination, allows moving only one guard during one turn. The minimal number of guards required to defend a graph in this model is denoted by γ^∞. Goddard, Hedetniemi and Hedetniemi [2] prove one set of bounds on γ_m^∞. Together with bounds proved by Burger et. al. [3] we get the following chain of inequalities.

Theorem 1 (Goddard et. al. [2, 3]). *For any graph G,*

$$\gamma(G) \leq \gamma_m^\infty(G) \leq \alpha(G) \leq \gamma^\infty(G) \leq \theta(G).$$

Here $\alpha(G)$ denotes the size of the maximum independent set in G and $\theta(G)$ denotes the clique covering number of G. The clique covering number of G is defined as the minimum number of cliques in G required to cover the vertex set of G.

Braga, de Souza and Lee [4] show that $\gamma_m^\infty(G) = \alpha(G)$ in all proper-interval graphs. As the problem of finding the maximum independent set in an interval graph on n vertices can be solved in time $\mathcal{O}(n \log n)$, or $\mathcal{O}(n)$ in the case endpoints of the intervals are sorted [5], we can compute $\gamma_m^\infty(G)$ efficiently on proper interval graphs.

Henning, Klostermeyer and MacGillivray [6] explore the relationship between the minimum degree of a graph, denoted by δ, and γ_m^∞. The authors prove the following theorem

Theorem 2 (Honning, Klostermeyet, MacGillivray [6]). *If G is a connected graph with $\delta(G) \geq 2$ of order $n \neq 4$, then $\gamma_m^\infty(G) \leq \lfloor (n-1)/2 \rfloor$, and this bound is tight.*

Finbow, Messinger and van Bommel [7], prove the following result about m-eternal domination of $3 \times n$ grids

Theorem 3 (Finbow, Messinger, van Bommel [7]). *For $n \geq 2$,*

$$\gamma_m^\infty(P_3 \square P_n) \leq \lceil 6n/7 \rceil + \begin{cases} 1 & \textit{if } n \equiv 7, 8, 14 \textit{ or } 15 \pmod{21} \\ 0 & \textit{otherwise} \end{cases}$$

Here $G \square H$ denotes the Cartesian product of graphs G and H.

Van Bommel and van Bommel [8] prove the following results for $5 \times n$ grids:

Theorem 4 (van Bommel, van Bommel [8]).

$$\lfloor \frac{6n+9}{5} \rfloor \leq \gamma_m^\infty(P_5 \square P_n) \leq \lfloor \frac{4n+4}{3} \rfloor$$

Concerning algorithmic research, there is a description of a linear algorithm for computing γ_m^∞ in trees presented by Klostermeyer and MacGillivray [9]. There is also a description of a brute-force algorithm presented by Bard et. al. [10], which we use in the Chapter 3. This algorithm solves the problem in exponential time and space.

In this text, we show that for a special subset of cactus graphs, whose every edge lies on a cycle of size $3k+1$, the m-eternal domination number can be computed directly from the sizes of the cycles. We also present a linear algorithm for computing the minimum number of required guards in the m-eternal guard configuration model in a restricted subclass of cacti. We require that every articulation is contained in exactly two blocks.

Our main results are summarized in the following theorems.

Theorem 6. *Let $G = (V, E)$ be a cactus, whose every edge lies on a C_{3k+1}. Let n be the number of vertices of G. Then $\gamma_m^\infty(G) = \gamma(G) = 1 + (n-1)/3$.*

Theorem 14. *Let G be some cactus graph on n vertices and m edges, with each articulation contained in two biconnected components. Then there exists an algorithm that computes $\Gamma_m^\infty(G)$ and runs in time $\mathcal{O}(n+m)$.*

In Chapter 2 in Section 2.1, we present a theorem, which gives us an upper bound on γ_m^∞ in graphs with some articulation by describing a general strategy. Then we show a direct computation of γ_m^∞ in cactus graphs, which have every edge on a cycle of size $3k+1$ for some $k \in \mathbb{N}$. We also show a direct computation of γ_m^∞ in noose graphs.

Next we consider the m-eternal guard configuration problem and present an algorithm computing $\Gamma_m^\infty(G)$ in cactus graphs, in which every articulation is in exactly two blocks. We provide a pseudo-code of the algorithm.

In Section 2.2, we show an extension of the linear algorithm by Klostermeyer and MacGillivray [9] from trees to clique trees. We present a pseudo-code of the algorithm.

Lastly, in Chapter 3 we provide an overview of the implemented algorithms. In Section 3.1 we describe the idea of the exponential brute-force algorithm including optimizations.

Chapter 2

Our results

2.1 Cactus graphs

Regarding the m-eternal guard configuration model and m-eternal domination model, we make a simple observation. Because every m-eternal guard configuration must induce a dominating set, we derive the following result.

Observation 1. *For every graph G, $\gamma(G) \leq \Gamma_m^\infty(G)$ and $\gamma(G) \leq \gamma_m^\infty(G)$.*

Also, because every strategy used in the m-eternal domination model can be applied in the m-eternal guard configuration, the following holds.

Observation 2. *For every graph G, $\Gamma_m^\infty(G) \leq \gamma_m^\infty(G)$.*

In the following text, we make use block-cut trees, which are described by Frank Harary [11] under the name block-cutpoint trees. It is a tree representation of biconnected components of a graph G. We define block-cut tree of graph G as a graph $BC(G) = (A \cup B, E')$, where A is the set of articulations in G and B is the set of biconnected components in G. A vertex $a \in A$ is connected by an edge to some $b \in B$ if and only if a lies in b in G.

We will show that every cactus can be built by attaching leaf cycles and leaf vertices. By *attaching* a C_n to a vertex v, we mean adding some P_{n-2} and connected both of its end vertices to v by edges.

Lemma 3. *Every cactus can be constructed in the following manner: We start with either a single vertex or a cycle and then extend it by following operations.*

- *Attach a leaf to one vertex.*
- *Attach a cycle to a single vertex.*

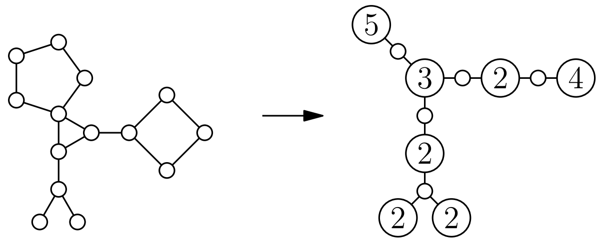

Figure 2.1: On the left is a cactus graph G, on the right its block-cut tree. Vertices representing biconnected components are labeled with their size, the unlabeled vertices represent articulations

Proof. Let G be any cactus graph. We will create a sequence of graphs $H_1, ..., H_k$, such that $H_1 = G$. If $i > 1$ and H_{i-1} is neither a single vertex nor a cycle, let H_i be H_{i-1} with either a leaf vertex removed or a leaf cycle removed. Removing a leaf cycle means removing all its vertices, except the one which has degree greater than 2.

In both those cases, H_i will not contain any new cycles and remain connected, therefore it will be a cactus.

We can observe that there is always a leaf vertex or a leaf cycle. Consider the block-cut tree $BC(G)$ for G. Because G is a cactus, every block in G is either a cycle or a pair of vertices connected by an edge. Every leaf in $BC(G)$ is some block in G. Such a leaf is either a leaf cycle or a pair of vertices connected by an edge, one of which is a leaf vertex.

Therefore we can reduce the whole cactus to a vertex or a cycle. By reversing the sequence, we obtain a sequence of graphs $I_1, ..., I_k$ starting with a single vertex or a cycle and ending with G. Every I_i with $i \geq 2$ is constructed from I_{i-1} by attaching a leaf vertex or a leaf cycle. $\square$

The following text concerns m-eternal domination in graphs containing some articulation. In this context, we will say that we partition the vertices of G into two subset H and I, such that $H \cup I = V(G)$ and $H \cap I = \emptyset$. We say that H and I are two partitions of G. By *partitioning by articulation* v, we mean splitting the vertices of G into two partitions H and I, such that H contains v and a strict subset of neighbors of v.

Another concept which we introduce are *restricted* edges. At one turn, only one of the restricted edges may be used by a guard to move through. We will use the introduced concepts in the statement of the following theorem.

Theorem 4. *Let G be a graph with articulation v. Let us partition the vertices of G by the articulation v into H and I. Let I' be the copy of $G[I]$ with added set R of restricted edges between every pair of neighbors of v.*

Let C be the set of all minimum m-eternal dominating sets on I', such that at most one guard passes through R during any move. Let D be the set

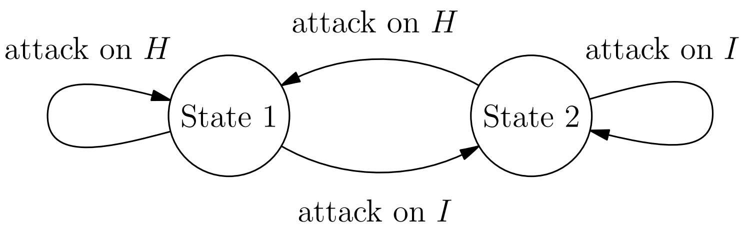

Figure 2.2: In State 1, v has to be occupied and I has a configuration of guards from C. In State 2, v may not be occupied and I has a configuration of guards from D.

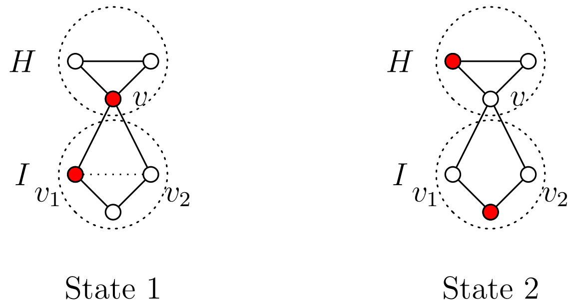

Figure 2.3: Partitioning of the graph into sets of vertices H and I, with possible guard configurations in each of the states. The edge $\{v_1, v_2\}$ in state 1 is the restricted edge in I'.

of all dominating sets on $G[I]$*. Let* G *be such a graph, that* $D \cap C \neq \emptyset$*, then* $\gamma_m^\infty(G) \leq \gamma_m^\infty(G[H]) + \gamma_m^\infty(I')$*.*

Proof. Let us illustrate with the state machine in Figure 2.2. We can start in either of its states. Also see Figure 2.3 for an example of two possible guard configurations corresponding to the two states.

Let us describe state 1, that is, v is occupied and guards on I are in any configuration of C. We will show how to simulate a guard passing through some edge $e = \{v_1, v_2\} \in R$, as that is the only difference between I' and $G[I]$. Without loss of generality, assume the movement is from v_1 to v_2. By moving the guard from v to v_2 and from v_1 to v, we simulate a guard passing through e. Therefore, we can guard $G[I]$ with the set of configurations C, assuming that v is occupied.

In case of attack on one of vertices in I, guards on H will not move and we will move from one configuration in C to another one in C. Therefore, we will remain in state 1.

In case of attack on a vertex in H, the guards in H may leave v unoccupied, therefore the guards in I may no longer use the edges in R. Therefore, the guards on I will move into some configuration in $D \cap C$, so that it does not require edges in R to be dominating. We move into state 2.

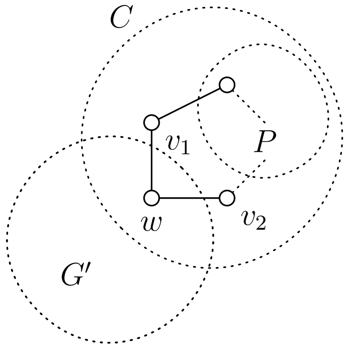

Figure 2.4: Vertices of the cycle removed in the inductive step

Now, let us describe state 2. Guards on I must be in some configuration in $D \cap C$. The strategy for $G[H]$ is used to defend against attacks on H, while the guards on I remain in the same configuration.

In case of attack on a vertex in I, the guards on I move into any configuration in C. Also, we will suppose an attack on v to force a guard on H to move to v. We will move into state 1.

We have described a strategy for G which will keep the guards of I and H separated and is m-eternally dominating, while using the minimum number of guards on both I' and $G[H]$, therefore it holds that $\gamma_m^\infty(G) \leq \gamma_m^\infty(G[H]) + \gamma_m^\infty(I')$. □

Lemma 5. *Let G be a cactus whose every edge lies on a C_{3k+1}. Let n be the number of vertices of G. Then $\gamma(G) = 1 + (n-1)/3$.*

Proof. Let $\beta(G) = 1 + (|V(G)| - 1)/3$.

First, let us show that $\gamma(G) \leq \beta(G)$. $|C(G)|$ is the number of C_{3k+1} in G. We will use induction on $|C(G)|$.

If $|C(G)| = 1$, then $G \cong C_{3k+1}$ and it holds that $\gamma(C_{3k+1}) = k + 1 = 1 + (|V(C_{3k+1})| - 1)/3 = \beta(C_{3k+1})$.

Let us show that $\gamma(G') \leq \beta(G')$ implies $\gamma(G) \leq \beta(G)$, where G' is G with $3k$ vertices of a leaf C_{3k+1} removed, thus with one less C_{3k+1}. By removing the leaf C_{3k+1}, we mean removing all the vertices of the C_{3k+1}, except the articulation connecting the C_{3k+1} to the rest of G.

Let C be the removed leaf C_{3k+1}. Let w be the vertex in C which is common to G' and C. Next, let v_1 and v_2 be vertices on C which are incident to w and let P be the remaining set of vertices on C. It holds that $G' = G \setminus (\{v_1, v_2\} \cup P)$. P will induce a subgraph P_{3k-3}. This notation is displayed in Figure 2.4. It holds that $C(G) = C(G') \cup C$.

By the induction hypothesis, there exists a dominating set on G' with size at most $\beta(G')$. Let D be such a dominating set.

Let us denote the vertices of P as $P = \{p_1, p_2, \ldots, p_{3k-2}\}$ in such a way, that the edges in $G[P]$ are $\{p_i, p_{i+1}\}$ for all $i \in \{1, \ldots, 3k-3\}$. Let $D_P =$

$\{p_1, p_4, \ldots, p_{3k-2}\}$ be the subset of the vertices of P. It is clear that D_P is a dominating set on $G[P]$. It holds that $|D_P| = k = (|V(C_{3k+1})| - 1)/3$. Vertices $D \cup D_P$ form a dominating set on G, because all the vertices possibly not dominated by D must be in P or are v_1 and v_2. Both v_1 and v_2 are dominated, because in D_P, we placed guards on both endpoints of P.

The inequality $|D| \leq \beta(G')$ implies $|D \cup D_P| \leq \beta(G') + k \leq \beta(G') + (|V(C_{3k+1})|-1)/3 = 1+(|V(G')|-1+|V(C_{3k+1})|-1)/3 = 1+(|V(G)|-1)/3 = \beta(G)$, therefore $\gamma(G) \leq \beta(G)$

Now, let us show that $\gamma(G) \geq \beta(G)$. By Lemma 3, we can build G from a single C_{3k+1} by attaching a new C_{3k+1} one by one. Let $G_1, ..., G_p$ be a sequence of graphs, where $G_1 \cong C_{3k+1}$ and $G_p = G$. Also, for every i, $1 < i \leq p$, it holds that G_i is G_{i-1} with one C_{3k+1} attached. We will use the same notation as above for vertices of the leaf C_{3k+1} removed in every G_i, $i \in \{2, ..., p\}$. Therefore, $G_{i-1} = G_i \setminus (\{v_1, v_2\} \cup P)$. Let C be the cycle removed from G_i.

We will show a proof by contradiction: let there exist some cactus G, whose every edge lies on a C_{3k+1} such that $\gamma(G) < \beta(G)$. Then, there is a smallest i such that $\gamma(G_i) < \beta(G_i)$. For $i = 1$, this can not hold, as $\gamma(C_{3k+1}) = k + 1 = (|V(C_{3k+1})| - 1)/3 + 1 = \beta(C_{3k+1})$.

Therefore $i > 1$. Then G_i has $3k$ more vertices than G_{i-1}. Let us denote the vertices of the new C_{3k+1} in G_i the same way as above, that is w, v_1, v_2 and P. Let the new C_{3k+1} be C. Let D be a dominating set on G_i with size $\gamma(G_i) \leq \beta(G) - 1$. For every vertex in G_i to be dominated, it must be true that C is dominated. Dominating C requires at least $k + 1$ vertices. Let us consider $D_C = D \cap (C \setminus \{w\})$. It holds that $|D_C| \geq k$, as at least $k + 1$ guards are required to dominate C_{3k+1} and one of those can occupy w. It holds that $D \setminus D_C$ must be a dominating set on $G_i \setminus (C \setminus \{w\}) = G_{i-1}$. It also holds that $|D \setminus D_C| \leq \beta(G_i) - 1 - k = \beta(G_i) - 1 - (|V(C)| - 1)/3 = (|V(G_i)| - 1 - |V(C)| + 1)/3 = (|V(G_{i-1})| - 1)/3 - 1 = \beta(G_{i-1}) - 1$, thus $\gamma(G_{i-1}) < \beta(G_{i-1})$. This is a contradiction with the assumption that i is the smallest possible such that $\gamma(G_i) < \beta(G_i)$. □

Theorem 6. *Let $G = (V, E)$ be a cactus, whose every edge lies on a C_{3k+1}. Let n be the number of vertices of G. Then $\gamma_m^\infty(G) = \gamma(G) = 1 + (n - 1)/3$.*

Proof. Let $\beta(G) = 1 + (|V(G)| - 1)/3$.

It holds that $\gamma(G) \leq \gamma_m^\infty(G)$. By Lemma 5 it holds that $\gamma(G) = \beta(G) \leq \gamma_m^\infty(G)$.

Let us show that $\gamma_m^\infty(G) \leq \beta(G)$ by induction on $|C(G)|$. If $|C(G)| = 1$, then $G \cong C_{3k+1}$, therefore it holds that $\gamma_m^\infty(G) = \gamma_m^\infty(C_{3k+1}) = k + 1 = \beta(G)$.

We will show that $\gamma_m^\infty(G') \leq \beta(G')$ implies $\gamma_m^\infty(G) \leq \beta(G)$, where G' is G with the leaf C_{3k+1} removed, except its articulation shared with the rest of G. Let C be the C_{3k+1} removed in G. Let us denote the vertices of C in the same way as in Lemma 5, that is w, v_1, v_2 and P. The notation is displayed in Figure 2.4. We can see that $G' = G \setminus (\{v_1, v_2\} \cup P)$.

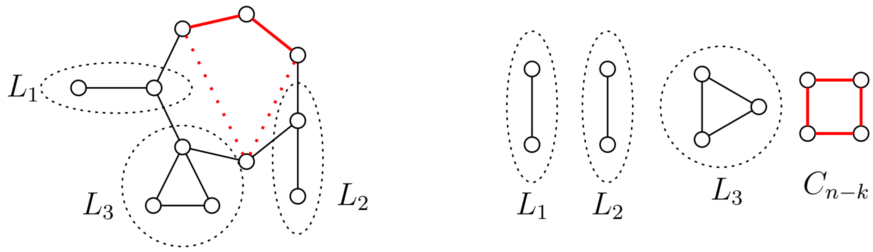

Figure 2.5: On the left is an example of a noose graph with cycle of size $n = 7$ and attached cliques L_1, L_2 and L_3. On the right is a graph, whose strategy we claim is equivalent to the strategy of the noose on the left. The red edges in the noose on the left are the edges, which will be guarded as a cycle of size $n - k$, with the dashed edges on being part of the noose.

Let I be C with w removed. We will show that partitioning of vertices of G by the articulation w into $V(G')$ and $V(I)$ satisfies the condition of Theorem 4. In this case, G' is the induced subgraph of G containing the articulation and I is the induced subgraph of G, such that $V(I) = V(C) \setminus w$, therefore $V(I) \cup V(G') = V(G)$ and $V(I) \cap V(G') = \emptyset$.

Let I' be I with the added restricted edge $\{v_1, v_2\}$, as those are the two vertices in I that are incident to the articulation w in G. It holds that $I' \cong C_{3k}$. Also $I \cong P_{3k-1}$. Because $I' \cong C_{3k}$, there will be exactly 3 m-eternal dominating sets on I' of size k and one of those sets is a dominating set on P_{3k-1}. Therefore, the condition that the intersection of the set of all m-eternal dominating sets on I' and the set of all dominating sets on I is not empty, is satisfied. As I' will have only one restricted edge, the condition that only one restricted edge may be used during a move is satisfied.

By Theorem 4, $\gamma_m^\infty(G) \leq \gamma_m^\infty(I') + \gamma_m^\infty(G') = \gamma_m^\infty(C_{3k}) + \gamma_m^\infty(G')$ $\leq (|V(C_{3k})|)/3 + \beta(G') = (|V(C_{3k})|)/3 + (|V(G)| - 1)/3 + 1 = (|V(C_{3k})| + |V(G')| - 1)/3 + 1 = (|V(G)| - 1)/3 + 1 = \beta(G)$. $\square$

Theorem 7. *Let G be a noose graph. Then $\gamma_m^\infty(G) = \lceil (n-k)/3 \rceil + k$, where n is the number of vertices on the cycle and k is the number of vertices with exactly one clique attached.*

Proof. Any noose graph G' can be created by placing an initial cycle and then repeatedly attaching cliques to its vertices. Let C' be the cycle of G' of size n', and let $L_1, \dots, L_{k'}$ be the cliques attached to C'. By attaching a clique of size ℓ to $v \in C'$, we mean adding a disjoint clique of size $\ell - 1$ and connecting its vertices to v. See Figure 2.5 for illustration.

This way of constructing any G' will allow us to do a proof by induction on the number of cliques attached. The proof will work for a starting cycle of any size. We will show, that the optimal guarding strategy of any G', which is a noose, is equivalent to separately guarding $L_1, \dots, L_{k'}$ and a cycle of size $n' - k'$. In case $n' = k'$, the strategy for the cycle will be omitted.

Let $\beta(G') = \lceil (n' - k')/3 \rceil + k'$. First let us show that $\gamma_m^\infty(G') \le \beta(G')$ for every noose graph G'.

Consider the case $k' = n'$. The strategy of placing one guard on every clique is eternally dominating for the whole G'. Therefore, $\gamma_m^\infty(G') \le \beta(G') = k'$.

Consider the case $k' = 0$, it holds that $G' \cong C'_n$, therefore $\gamma_m^\infty(G') = \lceil n'/3 \rceil = \beta(G')$.

For cases $0 < k' < n'$, we show a proof by induction on the number of cliques attached to the cycle. Let H be a noose graph consisting of cycle C_p and cliques $K_1, \ldots, K_q$. We claim that $(\gamma_m^\infty(H') \le \lceil (p-(q-1))/3 \rceil + (q-1)$ and $(q-1) < p)$ implies $\gamma_m^\infty(H) \le \lceil (p-q)/3 \rceil + q$, where H' is H with one less clique attached to the cycle. Let K_q be the clique missing in H'. We also claim that if the strategy of H' is equivalent to guarding all its cliques separately and all vertices of $H' \setminus (K_1 \cup \cdots \cup K_{q-1})$ as a cycle of size $p-(q-1)$, then the same will be true for H, with the size of the cycle being $p - q$ and the set of cliques being $K_1, \ldots K_q$.

In the base case $q = 1$, it holds that $H' \cong C_p$, therefore $\gamma_m^\infty(H') \le \lceil p/3 \rceil$. It holds that H' is guarded by the strategy of a cycle.

By the inductive hypothesis, H' is guarded as disjoint cliques and a cycle. Let B' a cycle C_{p-q+1}, the strategy of guarding this cycle is equivalent to guarding the vertices of $H' \setminus (K_1 \cup \cdots \cup K_{q-1})$. Now we show how to extend the strategy on H' to H. Let v' be the vertex on H' to which we will attach K_q. Because all cliques of H' are guarded independently, it suffices to extend the strategy on B', when we attach a clique to some $v \in B'$. Let B be B' with the new clique K_q attached to v. The new strategy on B will be applied on H.

Let us now apply Theorem 4 to get an upper bound on $\gamma_m^\infty(B)$. We can see that v is an articulation in B connecting the subgraphs K_q and B'. Let I be the vertices of $B \setminus K_q$. Note that $I \cong P_{p-q-1}$. Now I' will be $G[I]$ with the restricted edges R added. The set R contains only a single edge which connects the two vertices of degree one in I. Therefore, I' is a cycle of size $p-q$. Because I' is a cycle, there must a minimum m-eternal dominating set on I' dominating $G[I]$ without the restricted edges. From Theorem 4 follows that $\gamma_m^\infty(B) \le \gamma_m^\infty(K') + \gamma_m^\infty(I') = \gamma_m^\infty(K') + \gamma_m^\infty(C_{p-q}) = 1 + \lceil (p-q)/3 \rceil$. The strategy used will be that which is described in Theorem 4, therefore the strategy will separately guard K' and C_{p-q}. We already assumed that $K_1, \ldots, K_{q-1}$ are guarded separately and we have shown that K_q will be guarded separately as well by applying Theorem 4. Also, it holds, that $H \setminus (K_1 \cup \cdots \cup K_q)$ will be guarded as C_{p-q}. Therefore, $\gamma_m^\infty(H) \le \gamma_m^\infty(C_{p-q}) + q \le \lceil (p-q)/3 \rceil + q$.

This shows that for any noose graph G', it holds that $\gamma_m^\infty(G') \le \beta(G')$, therefore also for G, it holds that $\gamma_m^\infty(G) \le \lceil (n-k)/3 \rceil$.

Now let us show that $\gamma_m^\infty(G) \ge \lceil (n-k)/3 \rceil + k$.

Let $K_1, \ldots, K_k$ be the cliques of G. Let $C = G \setminus (K_1 \cup \cdots \cup K_k)$. Let C' be the cycle of G, which shares exactly one vertex with each $K_1, \ldots, K_k$. Let

$\beta(G) = \lceil (n-k)/3 \rceil + k$.

In case $k = 0$, therefore $G \cong C_n$, it holds that $\gamma_m^\infty(G) = \lceil n/3 \rceil = \beta(G)$. In case of every vertex of the cycle having a clique attached to it, the fact that no guard can dominate two cliques at once implies $\gamma(G) \geq k$, which implies $\gamma_m^\infty(G) \geq k = \beta(G)$.

Let us consider other cases. For contradiction, let G be any noose such that $\gamma_m^\infty(G) < \beta(G)$, therefore $\lceil (n-k)/3 \rceil + k - 1$ guards is enough to guard G.

We can see that each $K_1, \ldots, K_k$ must be always occupied by at least one guard. Therefore at least k guards will always occupy $K_1, \ldots, K_k$. That leaves $\lceil (n-k)/3 \rceil - 1$ guards to defend C. Now suppose some attack on C. The guard on some K_i can move outside of K_i only if some other guard takes his place in the same turn. This allows the defender to move guards along paths induced by vertices of $C' \cap (K_1 \cup \cdots \cup K_k)$. Let P be some induced connected component of $C' \cap (K_1 \cup \cdots \cup K_k)$ in G. Observe, that if the guard on one end of P moves outside of P into C, the guards must move along P to keep every K_i which shares a vertex with P occupied. Therefore, either one guard moves from C into P or some K_i was occupied by more than one guard and the additional guard moves into P during this move. We can see that the only use for more than one guard on some K_i placed on P is to move it into P to keep P occupied.

There will be at most $\lceil (n-k)/3 \rceil - 1$ guards on C. While attacking only C, the defending strategy of the guards will be equivalent to defending a cycle without the vertices $C' \cap (K_1 \cup \cdots \cup K_k)$, therefore a C_{n-k}. The guards moving along some P, with one guard from C entering P and other leaving P, is equivalent to one guard moving across an edge in C_{n-k}. In case some K_i was occupied by more than one guard and the guard moves into C at some later time, the move is equivalent to adding a guard to one endpoint of an edge. Moving the guard back into some K_j such that K_j would be occupied by two guards is equivalent to removing the guard from the strategy, only for it to be placed back into the strategy later. We map this move to keeping the guard on the vertex from which it would disappear, which may only give the defender an advantage. Note that this would allow more than one guard on the vertex.

Therefore, a defending strategy against the attacks on C with only $\lceil (n-k)/3 \rceil - 1$ guards would give us a strategy defending C_{n-k} with only $\lceil (n-k)/3 \rceil - 1$ guards in the eternal guard configuration model. This implies $\Gamma_m^\infty(C_{n-k}) \leq \lceil (n-k)/3 \rceil - 1$, which is a contradiction, as $\Gamma_m^\infty(C_{n-k}) \geq \gamma(C_{n-k}) = \lceil (n-k)/3 \rceil$.

□

Now we consider the m-eternal guard configuration problem, therefore from now on, we allow multiple guards on one vertex. We now present a

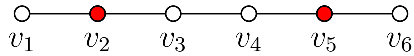

Figure 2.6: Example of a configuration of guards in the alternating pattern on P_5 for $k = 1$.

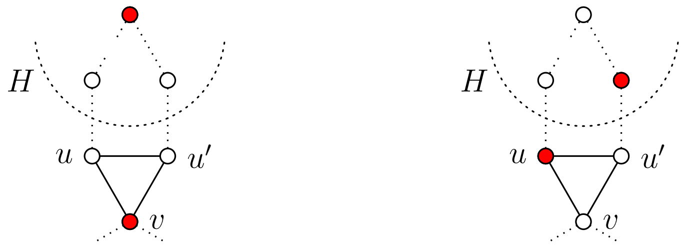

Figure 2.7: Situation in graph G when performing Reduction 1. H is the set of $3k - 3$ vertices of the leaf C_{3k} that are removed during the reduction. The added edge is $\{u, u'\}$. On the left is the configuration of guards in case v is occupied, on the right is the case when u is occupied.

collection of reductions, which will allow us to compute Γ_m^∞ in cactus graphs, whose every articulation lies in exactly two blocks.

Reduction 1. *Replace a single leaf C_{3k}, where $k \geq 2$, with K_3.*

Reduction 2. *Replace a single C_{3k+1} with K_1.*

Reduction 3. *Replace a single C_{3k+2} with K_2.*

Reduction 4. *Let H be some leaf clique which does not share any vertex with any induced cycle of size more than 3. Then remove H.*

Reduction 5. *Let H be some leaf clique which shares exactly one vertex only with exactly one cycle and the size of that cycle is more than 3.*

First we describe a general way, in which we can place guards on induced paths. Let P_{3n-1} be a path on $3n$ vertices. Let $V(P_{3n-1}) = \{v_1, v_2, \ldots, v_{3n}\}$, so that the edges of the P_{3n-1} are $\{v_i, v_{i+1}\}$ for every $1 \leq i < 3n$. We say that we keep the guards on P_{3n-1} in the *alternating pattern*, if the guards are placed on the set of vertices $\{v_{k+1}, v_{k+4}, v_{k+7}, ..., v_{k+3n-2}\}$ for some $0 \leq k \leq 2$. An example is displayed in Figure 2.6.

Lemma 8. *Let G be a cactus. Let C be a leaf C_{3k} on G. Let G' be G after application of Reduction 1 with C. Then G' is a cactus and $\Gamma_m^\infty(G) = \Gamma_m^\infty(G') + k - 1$.*

Proof. Let H be the induced subgraph of G which we removed when we replaced C with K_3.

Figure 2.8: Situation in graph G in case of Reduction 2. On the left is one possible configuration when v is occupied. On the right is the required configuration when v stops being occupied.

First we show that $\Gamma_m^\infty(G) \geq \Gamma_m^\infty(G') + k - 1$. H is some path on $3k-3$ vertices, with at most two of its vertices dominated from $G[V(G')]$. Therefore, we need to add guards to dominate at least $3k-5$ vertices, which requires at least $\lceil (3k-5)/3 \rceil = k-1$ guards.

Let us now show that $\Gamma_m^\infty(G) \leq \Gamma_m^\infty(G') + k - 1$. Any optimal strategy on G' can be easily extended to G by adding $k-1$ guards to H. We keep the guards on H in the alternating pattern at all times.

Let K be the new K_3 in G'. Because K has to be dominated at all times, at least one guard must be placed on it. Now we describe the movement of the $k-1$ guards on H in G in accordance to the movement of the guard placed on K in G'.

Let v be the vertex on K which is not incident to H. Let u and u' be the vertices on K incident to H. The notation is displayed in Figure 2.7. If the guard on K moves to v, we move the guards on H such that none of them is incident to K.

Without loss of generality, if a guard moves to u, we move the guards on H such that one of them is incident to u'. If the strategy on G' requires a guard to pass from u to u', we simulate that by having one guard from H move to u' and from u to H.

Now suppose an attack comes on H. We move the guards along H to repel the attack. Depending on the resulting configuration of guards on H, we suppose some attack on one of vertices v, u or u'. If none of the guards on H is incident to u or u', suppose an attack on v to force a guard moving there, so u and u' are still dominated. Without loss of generality, if one of the guards is incident to u, the vertex in H incident to u' would be left unprotected. Therefore, we suppose an attack on u' in G', to force a guard to move there. Now, if the strategy on G' requires a guard to pass through $\{u, u'\}$, we are able to simulate that, because the neighbor of u in H is occupied. $\square$

Lemma 9. *Let G be a cactus. Let C be a leaf C_{3k+1}. Let G' be G after application of Reduction 2 with C. Then G' is a cactus and $\Gamma_m^\infty(G) = \Gamma_m^\infty(G') + k$.*

Proof. Let H be the induced subgraph of G which we removed when we replaced C with K_1. Let v be the vertex of K_1. The notation is in Figure 2.8.

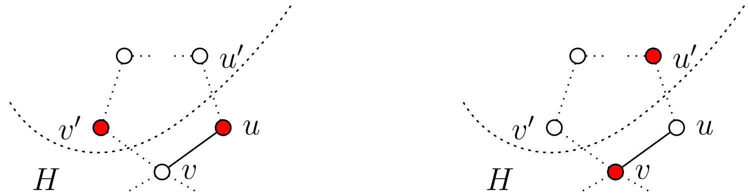

Figure 2.9: Situation in graph G in case of Reduction 3. On the left is the configuration after an attack on H that forced a guard to move into v'. On the right is a situation after an attack that forced a guard to move in to u'.

We show that $\Gamma_m^\infty(G) \geq \Gamma_m^\infty(G')+k$, as H is some path on $3k$ vertices, with at most 2 of those dominated from G'. Thus we need at least $\lceil (3k-2)/3 \rceil = k$ additional guards to protect it.

We now show that $\Gamma_m^\infty(G) \leq \Gamma_m^\infty(G')+k$. We extend an optimal strategy on G'. We place k guards on H. We keep the guards on H in the alternating pattern at all times and move them depending on the presence of the guard on v. If the guard moves away from v, we move the guards on H so that none of them is incident to v. In case the guard moves to v, we may leave the guards on H as they are.

Now suppose an attack on H. If the guards are forced to move so that one of them is incident to v, we suppose an attack on v to move a guard there. Now, if there comes another attack so that the guard is forced to move from v to H, we made sure to have a guard incident to v in H, so we move all the guards on H in the direction of v, to keep v occupied and to keep the eternally dominating configuration on H.

□

Lemma 10. *Let G be a cactus. Let C be a leaf C_{3k+2}. Let G' be G after application of Reduction 3 with C. Then G' is a cactus and $\Gamma_m^\infty(G) = \Gamma_m^\infty(G')+k$.*

Proof. Let H be the induced subgraph of G which we removed when we replaced C with C_3. Let K be the new K_2 in G'.

We show that $\Gamma_m^\infty(G) \geq \Gamma_m^\infty(G')+k$, as H is some path on $3k$ vertices and we have at most two of those dominated from G', therefore we need at least $\lceil (3k-2)/3 \rceil = k$ vertices to dominate it.

We show that $\Gamma_m^\infty(G) \leq \Gamma_m^\infty(G')+k$ by extending an optimal strategy on G'. We place k guards on H and keep them in the alternating pattern at all times.

Let u be the leaf vertex in K and v be the other vertex on K. Let $v' \in H$ be the vertex incident to v and $u' \in H$ be the vertex incident to u. The notation is displayed in Figure 2.9.

There must always be at least one guard on K. Suppose there is some attack on vertices of G'. Then we move the guards on H such that none of them is incident to u or v.

Now suppose an attack on H. If a guard on H is forced to move into v', we suppose an attack on u in G', to make sure that u is occupied. If after this comes an attack on H such that the guard on u must move to u', we move the guard on v' into v and move the rest of guards in G' as if there was an attack on v, that required the guard on u to move into v.

Similarly, if some attack on H requires a guard on H to move into u', we suppose an attack on v in G'. Therefore if u' is occupied, v will also be occupied. Suppose some attack on H requires a guard to move from v to v', in the previous moves we made sure that u' is also occupied. Therefore we move the guard on v to v' and at the same time move the guard from u' into u and also move the rest of guards in G' as if some attack on v required the guard on u to move into v.

□

Lemma 11. *Let G' be G after applying Reduction 4 with H being the leaf clique that is removed. Then G' is a cactus and $\Gamma_m^\infty(G) = \Gamma_m^\infty(G') + 1$.*

Proof. If H does not share any vertex with any induced cycle of size more than 4, it must share exactly one vertex with some clique of size 2 or 3. We will show $\Gamma_m^\infty(G') \leq \Gamma_m^\infty(G) - 1$, which implies $\Gamma_m^\infty(G) \geq \Gamma_m^\infty(G') + 1$. As H is a clique, there must be always at least one guard on H to defend it. If we assume, that there is always exactly one guard, we may remove H along with the guard and same strategy will defend G'. Suppose there was another guard on H. Because one guard suffices to defend against any attacks on H, the only use of the second guard was to move outside of H in case of an attack somewhere else. We may as well place the guard on the clique, which shares one vertex with H and the strategy will be the same.

Also, any strategy on G' can be easily extended to G by placing one guard on the newly added H. Therefore $\Gamma_m^\infty(G) \leq \Gamma_m^\infty(G') + 1$.

□

Lemma 12. *Let G' be G after applying Reduction 5, with H being the leaf clique. Then G' is a cactus and $\Gamma_m^\infty(G) = \Gamma_m^\infty(G') + 1$.*

Proof. First, we show that $\Gamma_m^\infty(G) \leq \Gamma_m^\infty(G') + 1$. Let u and v be the vertices incident to H in $G \setminus H$. Let w be the vertex in H incident to u and v. We will use Theorem 4. Let us partition the vertices of G into $V(H)$ and $V(G')$, with $V(H)$ being the set containing the articulation w and $\{u, v\}$ being the restricted edge added to $G[V(G')]$. Because there is only one restricted edge, the condition that at most one restricted edge is used is always satisfied and therefore $G[V(G')] = G'$. The notation is displayed in Figure 2.10.

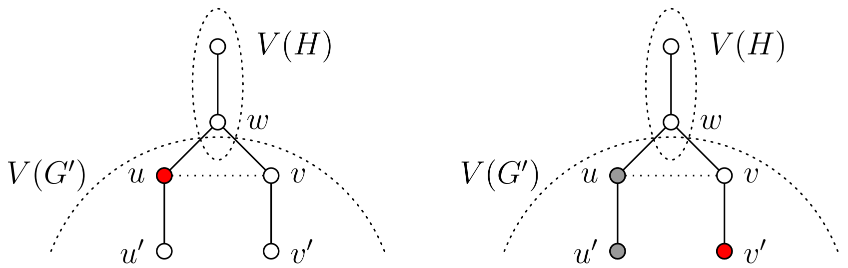

Figure 2.10: Displayed is an example of the graph G. The red vertices are occupied by a guard, the dashed $\{u, v\}$ edge is the added restricted edge in $G[V(G')]$. On the left is the situation, where we suppose there is no m-eternal dominating configuration on G' inducing a dominating set without $\{u, v\}$. On the right is the situation after an attack on v', with at least one of the gray vertices being occupied.

We need to show that there is at least one eternally dominating configuration on G', which induces a dominating set even without the edge $\{u, v\}$. Let C be the cycle in G sharing one vertex with H and let C' be the cycle in G' such that $V(C') = V(C) \setminus V(H)$. We assumed that the size of C is at least 4, therefore, C' has size at least 3.

For contradiction, suppose there is not an m-eternally dominating configuration in G', such that it induces a dominating set without the edge $\{u, v\}$. Then every m-eternally dominating configuration, without loss of generality, has u occupied and $N[v]$ contains no guard. Let v' be the neighbor of v, such that $v' \neq u$ and let u' be the neighbor of u, such that $u' \neq v$. It is possible, that $v' = u'$, in case the size of C' is 3. Now suppose some attack on v'. The guard occupying u must stay in $N[u]$, therefore u will still be dominated, and because v' has to be occupied, v will also be dominated, even without the $\{u, v\}$ edge.

To show $\Gamma_m^\infty(G) \geq \Gamma_m^\infty(G')+1$, which is equivalent to $\Gamma_m^\infty(G') \leq \Gamma_m^\infty(G)-1$, we adapt the strategy on G for G' while removing one guard. At least one guard must always occupy H to defend it. Suppose exactly one guard always occupies H. Without loss of generality, the guard can move from H to v only if some guard moves from u to H. That is equivalent to a guard moving across the edge $\{u, v\}$. Suppose another guard occupies H. Its only possible use is to move into v or u in case of an attack. We can place this guard on v, where it can perform the same movements. When we remove H from G, we can also remove the one guard which always had to occupy it.

□

Lemma 13. *Let G be a graph. Then we can construct its block-cut tree $BC(G)$ in linear time.*

Proof. Using Tarjan's algorithm [12], we can find the biconnected components of a graph in linear time. After running this algorithm, we will have every vertex of G labeled with the set of its biconnected components. Finding all articulations now consists of iterating through all vertices of G and checking, which are in more than one biconnected component. Now we can simply create the vertices of the block-cut tree from the set of articulations and biconnected components, and connect each articulation to the biconnected components, which it is a part of. The number of biconnected components is bounded by the number of edges in G and the number of articulations is lower than the number of vertices in G. Let n be the number of vertices in G and m be the number of edges. Then the block-cut tree will have at most $\mathcal{O}(n + m)$ vertices, and because it is a tree, also $\mathcal{O}(n + m)$ edges. Constructing each vertex and edge is done in constant time, therefore the whole construction is done in $\mathcal{O}(n + m)$. □

We present a description of a polynomial algorithm, which computes $\Gamma_m^\infty(G)$ in cactus graphs, in which every articulation is in two blocks. We follow a general description with a detailed pseudo-code. The general algorithm is as follows.

1. Construct block-cut tree BC of the input graph G.
2. Try to apply one of the Reductions 1, 2, 3, 4 or 5 on BC.
3. If one the reductions was applied, then let G' be G after application of the reduction and let p be the appropriate number of guards that was removed from G by applying the reduction. Recursively apply the algorithm on G' and return $\Gamma_m^\infty(G') + p$.
4. If none of the reductions were applied, then $G \cong C_n$, therefore return $\Gamma_m^\infty(C_n)$.

We present a pseudo-code implementation of the algorithm, such that its running time is linear with the size of the input. We use an iterative approach, which is easier to implement and also easier to analyze.

Algorithm 1 m-Eternal guard configuration in a cactus

```
procedure m-EGC-Cactus-Graph(G)
    ▷ Size −1 means that the vertex is an articulation
    BC = (V′, E′, size, deg) ← the block-cut tree of G
    ▷ Set the number of deleted cliques of every articulation to zero
    for v ∈ V′ do
        if size(v) = −1 then
            cliques(v) = 0
        end if
    end for
    stack ← ∅
    for v ∈ V′ do
        if deg(v) ≤ 1 then
            push v into stack
        end if
    end for
    g ← 0                                              ▷ The resulting Γ_m^∞(G)
    while stack ≠ ∅ do
        pop from stack into v
        ▷ Pick an undeleted neighbor of v
        u ← the only neighbor of v, such that deg(u) > 0.
        del ← false
        if size(v) ≠ −1 then                           ▷ A block
            (g, stack, size, clique, del) ←
                    block(u, v, g, stack, size, clique, del)
        else if size(v) = −1 then                      ▷ An articulation
            (g, stack, size, clique, del) ←
                    articulation(u, v, g, stack, size, clique, del)
        end if
        if del = true then
            deg(u) ← deg(u) − 1
            deg(v) ← 0                                 ▷ Deletion of v
            if deg(u) = 1 then
                push u into stack
            end if
        end if
    end while
    return g
end procedure
```

Algorithm 1 m-Eternal guard configuration in a cactus

```
procedure BLOCK(u, v, g, stack, size, clique, del)
    if size(v) > 3 then                                    ▷ Cycle of size > 3
        (g, stack, size, del) ← LEAF-CYCLE(v, g, stack, size, del)
    else if 2 ≥ size(v) ≥ 3 then                           ▷ Reduction 4 or 5
        if deg(v) = 0 then                                 ▷ The block was reduced to K_n
            g ← g + 1
        else
            clique(u) ← clique(u) + 1
        end if
        del ← true
    else if size(v) = 1 then                               ▷ A leftover vertex from reduced block
        if deg(v) = 0 then                                 ▷ The block was reduced to a single vertex
            g ← g + 1
        end if
        del ← true
    else if size(v) = 0 then                               ▷ A completely reduced block
        del ← true
    end if
    return (g, stack, size, clique, del)
end procedure
procedure LEAF-CYCLE(v, g, stack, size, del)
    if size(v) mod 3 = 0 then                              ▷ Reduction 1
        g ← g + size(v)/3 − 1
        size(v) ← 3
        push v into stack
    else if size(v) mod 3 = 1 then                         ▷ Reduction 2
        g ← g + (size(v) − 1)/3
        if deg(v) = 0 then                                 ▷ v is not a leaf cycle, but a disjoint cycle
            g ← g + 1
        end if
        del ← true
    else if size(v) mod 3 = 2 then                         ▷ Reduction 3
        g ← g + (size(v) − 2)/3
        size(v) ← 2
        push v into stack
    end if
    return (g, stack, size, del)
end procedure
```

Algorithm 1 m-Eternal guard configuration in a cactus

76: **procedure** ARTICULATION($u, v, g, stack, size, clique, del$)
77:     **if** $clique(v) = 1$ **then** ▷ Part of a noose
78:         $size(u) \leftarrow size(u) - 1$
79:         $g \leftarrow g + 1$
80:         $del \leftarrow true$
81:     **else if** $clique(v) = 0$ **then** ▷ Was adjacent to a C_{3k+1} cycle
82:         $del \leftarrow true$
83:     **end if**
84:     **return** ($g, stack, size, clique, del$)
85: **end procedure**

Theorem 14. *Let G be some cactus graph on n vertices and m edges, with each articulation contained in two biconnected components. Then Algorithm 1 correctly computes $\Gamma_m^\infty(G)$ and runs in time $\mathcal{O}(n + m)$.*

Proof. First we show correctness of Algorithm 1. We perform the reductions in the while loop at line 17. At the start of every iteration, we process one leaf vertex in BC. In case the last iteration deleted a block vertex from BC, the next vertex processed must be the articulation which became a leaf. This is ensured because we use a stack to hold the leaves and after every deletion, at most one vertex may become a leaf. Also, this ensures that BC is a valid block-cut tree whenever we perform a reduction on a block vertex in BC.

Let v be the leaf vertex currently processed in the loop. Consider the case where v is some leaf cycle of size more than 3, therefore $size(v) > 3$. We simulate one of the Reductions 1, 2 or 3 in LEAF-CYCLE at line 57. It suffices to decrease the size of the cycle in case of Reductions 1 or 3. In case of Reduction 2, we simply delete the block, except the articulation contained in it. In case v is a disjoint cycle and does not have any articulation incident to it which would be processed later, we increase Γ_m^∞ immediately and correctly compute Γ_m^∞ for the disjoint cycle.

Consider the case where v is a leaf clique, therefore $2 \leq size(v) \leq 3$. We simulate Reduction 4 or 5 in Procedure BLOCK at line 37. In both reductions, we remove the whole clique and decrease the size of the incident block by one. We mark the incident articulation u as having been part of the reduction, so that when we process u, we increase Γ_m^∞ by one and decrease the size of the incident block by one.

We have to check for the case where v is a disjoint clique and does not have any articulation incident to it. In that case, we increase Γ_m^∞ right away.

We finish the reduction in the next iteration, when u becomes a leaf. It will be processed by Procedure ARTICULATION at line 76. In cases of both reductions, we now increase Γ_m^∞ by one and decrease the size of the incident block, therefore removing the articulation from it.

This concludes the simulations of the reductions. In the algorithm, we have to consider more cases that can appear as a consequence of previous reductions. One is when a block has size 1, therefore its size was decreased by some other reductions. This block is either an isolated vertex or it contains a single vertex, which was previously an articulation and therefore is contained in some other block, which will be processed later. This is taken care of at line 47.

Another case is that some block was completely deleted in previous reductions, therefore has size zero. This case is taken care of at line 58.

This concludes the analysis of correctness of the algorithm. Now we show that Algorithm 1 runs in time $\mathcal{O}(n+m)$.

By Lemma 13, we can construct BC in linear time, therefore line 3 will run in time $\mathcal{O}(n+m)$. We can modify the construction of BC, so that we receive every block vertex in BC labeled with the size of the block. This can still be performed in linear time.

Note that the number of vertices of BC is bounded $2n$, therefore the same holds for the number of edges in BC. Therefore $|V(BC)| = \mathcal{O}(n)$ and $|E(BC)| = \mathcal{O}(n)$.

Now consider the while loop at line 17. We claim that every vertex in BC is processed at most twice in the loop and every iteration takes constant time. Let v be the currently processed vertex. In case it is a block of size at most 3 or an articulation, it will be deleted at the end of the iteration. In case it is a block of size more than 3, it will be processed at line 37. In that case, it becomes a block of size at most 3 and will be deleted in the next iteration.

To show that every iteration takes constant time, first consider the way we pick an undeleted neighbor of v on line 20. This can be done by iterating over all edges incident to v. Because every vertex in BC is processed at most twice, we iterate over every edge at most four times. The rest of operations in the while loop at 17 clearly takes constant time. Therefore the whole algorithm runs in time $\mathcal{O}(n+m)$.

□

Figure 2.11: On the left is an example of a graph G. On the right is the graph G after application of Reduction 6.

2.2 Clique trees

Klostermeyer and MacGillivray [9] describe a linear algorithm for solving the m-eternal domination problem on trees. We show an extension, which allows its use on clique trees. It is based on a set of reductions, which we are able to execute in linear time.

We say that two cliques in a graph G are *incident* if they share exactly one vertex.

Reduction 6. *Remove a leaf clique which is incident to only one other clique.*

Reduction 7. *Let x be a vertex incident to more than 2 leaf vertices which is also incident to at most one clique of size greater than 2. Remove all leaves incident to x.*

Reduction 8. *Let v be some vertex which is contained in more than one leaf clique and lies on at most one non-leaf clique. Remove all edges on leaf cliques that v is contained in, except those edges which are incident to v.*

Repeated applications of these reductions will end with G being either a K_n or a star, both of which are trivial to solve.

Lemma 15. *If G' is the result of applying Reduction 6 to G, then G' is a clique tree and $\gamma_m^\infty(G) = \gamma_m^\infty(G') + 1$.*

Proof. The proof is adapted from the proof by Klostermeyer and MacGillivray [9] of their Lemma 21 in the cited work. See Figure 2.11 for illustration.

It is clear that G' is a clique tree. No new clique or a cycle could have been created.

Let H be the leaf clique removed from G. At least one guard must always occupy H to defend against attacks on H and one guard suffices. Therefore, in an optimal strategy of G, one guard always occupies H. After removing H, we may remove one guard with it. Therefore $\gamma_m^\infty(G') \leq \gamma_m^\infty(G) - 1$ implies $\gamma_m^\infty(G) \geq \gamma_m^\infty(G) + 1$.

Also, any strategy of G' can be extended to G by adding one guard to H. Therefore $\gamma_m^\infty(G) \leq \gamma_m^\infty(G') + 1$. □

□

Figure 2.12: On the left is an example of a graph G, with the encircled vertices being the leaf vertices removed during application of Reduction 7. On the right is G after the application of Reduction 7.

Lemma 16. *If G' is the result of applying Reduction 7 to G, then G' is a clique tree and $\gamma_m^\infty(G) = \gamma_m^\infty(G') + 1$.*

Proof. The proof is adapted from the proof by Klostermeyer and MacGillivray [9] of their Lemma 20 in the cited work. See Figure 2.12 for illustration.

Let H be the clique of size at least 2 that x lies on. Let $\ell_1, \ell_2, ..., \ell_k, k \geq 2$ be the leaves adjacent to x such that none of them is in H. In order to defend against the sequence of attacks $\ell_1, \ell_2, ..., \ell_k$, there must always be one guard on one of the leaves $\ell_1, \ell_2, ..., \ell_k$ and one on x, and also those two guards suffice. Thus, in a minimum m-eternal dominating set, there are two guards that defend these leaves. After removing $\ell_1, \ell_2, ..., \ell_k$ and the guard which was required to move to the leaves in case of an attack, the same strategy eternally defends G'. Therefore $\gamma_m^\infty(G') \leq \gamma_m^\infty(G) - 1$, which implies $\gamma_m^\infty(G) \geq \gamma_m^\infty(G') + 1$.

Now we show that $\gamma_m^\infty(G) \leq \gamma_m^\infty(G') + 1$. In any strategy on G', there must be a guard on H to defend against attacks on H. We place the one additional guard on ℓ_1 and suppose an attack on x in G' to force a guard moving there.

Now suppose there is an attack on $y \in H \setminus \{x\}$. The guard on the occupied leaf will move to x to protect the leaves $\ell_1, \ell_2, ..., \ell_k$, while the other guard on H moves to y. Suppose there is an attack on $y' \in \{\ell_1, \ell_2, ..., \ell_k\}$. One of two cases is possible. In the first case is one of the other leaves $\ell_1, \ldots, \ell_k$ occupied, therefore we made sure that x is also occupied. Let ℓ_p be the occupied leaf. We move the guard on ℓ_p to x and the guard on x to y'. In the second case, none of the leaves $\ell_1, \ldots, \ell_k$ is occupied. In that case, the additional guard has moved to x. We move the guard on x to y' and suppose an attack on x in G' to make sure that x is occupied.

□

Lemma 17. *If G' is the result of applying Reduction 8 to G, then G' is a clique tree and $\gamma_m^\infty(G) = \gamma_m^\infty(G')$.*

Proof. We want to show that those removed edges do not need to be used in an optimal strategy. Let $L_1, \ldots, L_k$ be the cliques from which we removed the edges. Let R be the set of the removed edges. Let v be the vertex shared by

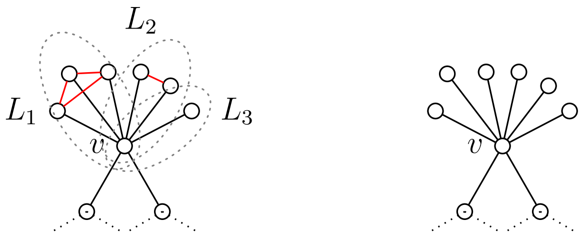

Figure 2.13: On the left is an example of a graph G, with the red edges being the set R removed during application of Reduction 8. The leaf cliques that contain v are L_1, L_2 and L_3. On the right is G after the application of Reduction 8.

$L_1, \ldots, L_k$. We claim that it is sufficient to keep a guard on v at all times and occupy at most one of $L_1, \ldots, L_k$, namely the one that was last attacked. See Figure 2.13 for illustration.

For contradiction, suppose that in every optimal strategy, at least one edge $\{r_1, r_2\} \in R$ is used. For a guard to pass through $\{r_1, r_2\}$, he had to be on r_1 or r_2, therefore was placed on the leaf clique. As all the other leaf cliques has to be dominated, either there is a guard on v or on every leaf clique.

Suppose that there is a guard on every leaf clique. That requires at least $k \geq 2$ guards. Moving one of the guards on $L_1, \ldots, L_k$ to v instead keeps all $L_1, \ldots, L_k$ protected while using only 2 guards. This is at least as good as the previous configuration, therefore we can assume that at least one optimal strategy uses it. We will show that this configuration is still eternally dominating.

Without loss of generality, we can replace the move from r_1 to r_2 by a move from v to r_2 and move the other guard on $L_1, \ldots, L_k$ to v. Therefore, we can assume that at least one of the optimal strategies on G keeps v occupied and does not use the edges in R.

$\square$

We now present the description of the algorithm, which uses the reductions. The algorithm is as follows.

1. Construct block-cut tree BC of the input graph G
2. Try to apply one of the Reductions 6, 7 or 8 on BC.
3. If one the reductions was applied, then let G' be G after application of the reduction and let p be the appropriate number of guards that was removed from G by applying the reduction. Recursively apply the algorithm on G' and return $\gamma_m^\infty(G') + p$.
4. If none of the reductions were applied, then G is a star or a K_n. Return $\gamma_m^\infty(G)$ directly.

We provide a pseudo-code implementation the algorithm, such that its running time is linear with the size of the input. We use an iterative approach, which is easier to implement and also easier to analyze.

Algorithm 2 m-Eternal domination number in clique trees

```
procedure m-EDN-BLOCK-GRAPH(G)
    ▷ Size −1 means that the vertex is an articulation
    BC = (V′, E′, size, deg) ← the block-cut tree of G
    ▷ Set the number of deleted cliques of every articulation to zero
    for v ∈ V′ do
        if size(v) = −1 then
            cliques(v) = 0
        end if
    end for
    stack ← ∅
    for v ∈ V′ do
        if deg(v) ≤ 1 then
            push v into stack
        end if
    end for
    edn ← 0
    while stack ≠ ∅ do
        pop from stack into v
        u ← the only neighbor of v such that deg(u) > 0.
        if size(v) ≥ 1 then                              ▷ An undeleted clique
            edn ← CLIQUE(edn, v, u)
        else if size(v) = −1 then                        ▷ An articulation
            edn ← ARTICULATION(edn, v, u)
        end if
        if deg(v) > 0 then
            deg(u) ← deg(u) − 1
            deg(v) ← 0                                   ▷ Deletion of v
        end if
        if deg(u) = 1 then
            push u into stack
        end if
    end while
    return edn
end procedure
```

Algorithm 2 m-Eternal domination number in clique trees

```
35: procedure Clique(edn, u, v)
36:     if size(v) > 1 then                              ▷ An undeleted clique
37:         if deg(v) = 0 then                           ▷ An isolated K_n
38:             edn ← edn + 1
39:         else                                         ▷ A leaf clique
40:             cliques(u) ← cliques(u) + 1
41:         end if
42:     else if size(v) = 1 then
43:         if deg(v) = 0 then                           ▷ An isolated vertex
44:             edn ← edn + 1
45:         end if
46:         ▷ Otherwise v must be contained in another
47:         ▷ block and will be reduced there
48:     end if
49:     return edn
50: end procedure
51: procedure Articulation(edn, u, v)
52:     if cliques(v) = 1 then                           ▷ Reduction 6
53:         size(u) ← size(u) − 1
54:         edn ← edn + 1
55:     else if cliques(v) > 1 then                      ▷ Reductions 8 and 7
56:         edn ← edn + 1
57:     end if
58:     ▷ Otherwise v was part of a block reduced to size 0 or 1
59:     ▷ by some previous reductions
60:     return edn
61: end procedure
```

Theorem 18. *Let G be a clique tree on n vertices and m edges, then Algorithm 2 correctly computes $\gamma_m^\infty(G)$ and runs in time $\mathcal{O}(n+m)$.*

Proof. First we show the correctness of the algorithm. After constructing BC for G, we repeatedly pick a leaf from BC and after simulating a reduction on it, we delete it. This is done in the loop at line 17. Let v be the currently processed leaf of BC. In case v is a block, we want to perform either Reduction 6 or Reductions 8 and 7. Let u be the articulation incident to v in BC. Which reduction we want to execute depends on the number of blocks that u is contained in. Therefore we keep a track of how many blocks incident to u in BC we deleted and postpone the reductions to the point in time, when u is processed.

It is guaranteed that after every block containing some articulation u is processed, u will be the next processed vertex in the loop at line 17. This is ensured by using a stack to keep our leaf vertices. We insert vertices into the stack as soon as they became a leaf, therefore at most one vertex is inserted after deletion of some vertex. This is done at line 25.

Suppose we now process some articulation v in BC. In case it was in one deleted clique, we want to perform Reduction 6. This is done in Procedure ARTICULATION at line 52. It suffices to increase the resulting γ_m^∞ by one and decrease the size of the incident block, from which the articulation was removed during Reduction 6.

In case v was in more than one deleted clique, we want to perform Reduction 8 and subsequently Reduction 7. As the cliques are already deleted, we simply increase γ_m^∞ by one and continue.

When none of the reductions can be applied, BC represents either a single clique or a star. Let v be the currently processed vertex of BC. In case BC represents a single clique, then v has no incident articulation in BC. This is checked at line 37. In this case, we increase γ_m^∞ by one.

Suppose that we reduced BC so that it represents a star. Let G' be the graph represented by BC. We will show that the algorithm runs in such a way, that it correctly outputs $\Gamma_m^\infty(G')$ as 2. First, the leaf vertices of G' will be removed one by one at line 36. After this is applied, the resulting graph is a single clique of size one. Therefore the resulting $\Gamma_m^\infty(G')$ is 2.

In the course of the algorithm, there can appear other cases we have to take care of. Let v be a processed block vertex in BC. Suppose the size of v is one, therefore its size was decreased by previous reductions. If it is not an isolated vertex, it must be contained in some other clique and will be processed with it. This is checked at line 42. In case the size of v is zero, all of its vertices were deleted in the previous reductions. Any reductions will now be skipped and v will be simply deleted.

Suppose the case is that we process some articulation v in BC, which is marked as not being a part of any deleted cliques. That can happen only if the articulation was contained in some blocks, which were previously reduced to size 1 or 0. In both cases, v is not an articulation. Therefore we simply delete v and continue.

This concludes the proof of correctness of the algorithm. We now show that the algorithm runs in time $\mathcal{O}(n+m)$. By Lemma 13, the construction of BC at 3 will take time $\mathcal{O}(n+m)$. The loop at line 17 processes every vertex exactly once, as it is always deleted at the end of every iteration.

Consider the way we pick an undeleted neighbor of the processed vertex v at line 19. This is achieved by iteration over all edges incident to v. As every vertex in BC is processed once, every edge will be iterated over exactly twice. Note that the size of BC is linear in the size of G. Therefore, the whole algorithm runs in time $\mathcal{O}(n+m)$. □

Chapter 3

Implementation

3.1 Brute-force algorithm

We include implementation of an exact algorithm for finding either all minimum m-eternal dominating sets or minimum m-eternal guard configurations, therefore also computing γ_m^∞ or Γ_m^∞. The idea is described by Bard et. al. [10]. It consists of creating a configuration graph, whose vertices are possible placements of guards on the input graph, while edges represent the fact that one placement may be turned into another during one turn.

More formally, let $D_k(G) = (V', E')$ be the directed configuration graph for G and some $k \in \mathbb{N}$, where $V' \subset \mathbb{Z}^{|V(G)|}$. Let us order the vertices of G as $V = \{v_1, v_2, ..., v_n\}$. Then every $I \in V'$, where $V' = (p_1, p_2, ..., p_n)$ represents some configuration, where for every $i \leq n$, there is p_i guards placed on v_i. We require that for every $I \in V'$, it holds that $\sum_{i=1}^{n} p_i = k$. Let $I, J \in V'$, then $(I, J) \in E'$ if and only if the set of guards represented by I can move into the configuration represented by J.

For $D_k(G)$ to represent a valid strategy, every $I \in V'$ must be an eternally dominating configuration. That is true if and only if, for every $I \in V'$, the union of the endpoints of edges starting in v equals to $V(G) \setminus I$. That is, in case of an attack on any unoccupied vertex, we are able to move into some configuration which defends against this attack, and is also able to respond to any attack.

It is therefore true that $\gamma_m^\infty(G) \leq k$ if and only if the corresponding configuration graph is not empty. The general idea is to build $D_k(G)$ with all possible configurations of k guards on $V(G)$ and iteratively remove those vertices in $D_k(G)$, which do not represent an eternally dominating configuration. This solution is exponential in time and space, as the number of vertices of $D_k(G)$ is exponential in the size of G.

To make sure this approach produces a valid strategy for the m-eternal domination model, we restrict every configuration to have at most one guard on every vertex.

We present a polynomial algorithm which decides whether one configuration can move into another during one move.

Lemma 19. *Let $D_k(G) = (V', E')$ be the configuration digraph for $G = (V, E)$ and k. Let $I, J \in V'$. Then deciding whether $(I, J) \in E'$ can be done in polynomial time.*

Proof. We use a max flow algorithm on an auxiliary network to decide whether $(I, J) \in E'$. Let $T = (V_T, E_T)$ be the auxiliary network, such that $V_T = V_1 \cup V_2 \cup \{s, t\}$. The sets V_1 and V_2 are copies of $V(G)$. We connect $v_1 \in V_1$ to $v_2 \in V_2$ by an arc if and only if $v_2 \in N[v_1]$ in G. We connect s to every $v_1 \in V_1$ and every $v_2 \in V_2$ to t. We construct the capacities of the arcs as follows. Let $L \in V'$, then $L(v)$ is the number of guards on $v \in V(G)$ in the configuration represented by L. Let $c : E_T \to \mathbb{Z}$ be the capacity function defined as $c(v, u) = \begin{cases} \infty & \text{if } v \in V_1 \text{ and } u \in V_2 \\ I(u) & \text{if } v = s \text{ and } u \in V_1 \\ J(v) & \text{if } v \in V_2 \text{ and } u = t \end{cases}$

Observe that a flow moving from some $v_1 \in V_1$ to $v_2 \in V_2$ represents guards moving across an edge between those vertices. The size of the flow going from every $v_1 \in V_1$ is at most the number of guards on $v_1 \in V(G)$ in the configuration I. Similarly, the size of flow going from every $v_2 \in V_2$ is at most the number of guards on $v_2 \in V(G)$ in the configuration J. We can move from configuration I into J in one turn if and only if the size of the maximum flow from s to t is equal to k. That is, every guard in I was moved into some other position in J. □

In our implementation, we choose the Ford Fulkerson algorithm to compute the maximum flow. The fact that its worst case running time is $\mathcal{O}(|E_N|f)$ [13], where f is the size of the resulting flow, is suitable for this use. Let n be that number of vertices of G and m be the number of edges of G. We can see that $|E_N| = 3n + m$, therefore the total running time is $\mathcal{O}((n + m)k)$.

Also regarding the implementation, one basic optimization is checking all configurations in V' and discarding those, which do not induce a dominating set on G, as those can not be a part of any valid strategy.

3.1.1 Heuristic speed-up

We employ a heuristic, which in practice provides a significant speed-up against the naive brute-force, while still guaranteeing correctness of the result. The idea is based on the observation, that for many graphs, an optimal strategy with k guards requires a significantly lower number of configurations, than is the number of all dominating sets on G of size k.

Consider for example a graph G consisting of k disjoint P_1. While $\gamma_m^\infty(G) = \gamma(G) = k$ and the number of m-eternal dominating sets is 2^k, just two different dominating sets suffice to defend against all attacks.

While building the edges of the configuration graph $D_k(G)$, we periodically check if the graph already contains some valid strategy and we return the result if it does. Because the construction of vertices of $D_k(G)$ is implemented in an iterative way, many similar configurations are placed next to each other in the array holding all the configurations. To increase the chances of configurations, which would produce a valid strategy, being processed close to each other, we randomly shuffle the array of vertices.

3.2 Cactus graphs

We implement our algorithm for cactus graphs, in which every articulation is in two blocks. First, we construct the block-cut tree, as described in Lemma 13. Then we perform the check if the input graph is of correct class. To do that, we use the following

Observation 20. *Graph G is a cactus if and only if it is connected and every block is either a cycle or a pair of vertices connected by an edge*

This holds as if any edge was lying on two cycles, that edge would be part of some block which is not a cycle. Also, every articulation is in exactly two blocks if and only if the degree of every articulation in the block-cut tree of the input graph is 2.

The implementation of the algorithm is described Algorithm 1.

Chapter 4

Future work

4.1 Open problems

There are many interesting problems regarding the m-eternal domination problem still unanswered. The complexity of the decision variant of the m-eternal domination problem is still largely unknown. The approach of creating the configuration graph with all possible placements of guards as its vertices shows that the problem is solvable in exponential time and space, therefore it belongs to EXPTIME. It is not clear whether the problem lies in PSPACE. Our experiments hint at the fact, that the size of the minimum possible configuration graph, such that it represents a valid strategy with optimal number of guards, is often small. It is therefore an interesting question, whether we can bound the size of the minimum optimal configuration graph by some function of the size of the input. Bounding the size of the minimum optimal configuration graph by some polynomial of the input size would imply that the m-eternal domination problem lies in PSPACE.

The natural extension of the algorithm from cactus graphs is to the more general case of graphs with treewidth equal to 2. It is an interesting question, whether we can design an algorithm, whose running time would be parameterized by the treewidth of the input graph.

It is also interesting to show for which graph classes is γ^∞ equal to Γ_m^∞ and which conditions are either necessary or sufficient, so that this equality holds.

Bibliography

[1] Klostermeyer, W. F.; Mynhardt, C. M. Protecting a Graph with Mobile Guards. *ArXiv e-prints*, July 2014, 1407.5228.

[2] Goddard, W.; Hedetniemi, S. M.; Hedetniemi, S. T. Eternal Security in Graphs. *J. Combin. Math. Combin. Comput.*, volume 52, 2005: pp. 169–180. Available from: http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.94.2592

[3] Burger, A. P.; Cockayne, E. J.; Gründlingh, W. R; Mynhardt, C. M.; van Vuuren, J. H.; Winterbach, W. Infinite order domination in graphs. *J. Combin. Math. Combin. Comput.*, volume 50, 2004: pp. 179–194.

[4] Braga, A.; de Souza, C. C.; Lee, O. The Eternal Dominating Set problem for proper interval graphs. *Information Processing Letters*, volume 115, no. 6, 2015: pp. 582 – 587, ISSN 0020-0190, doi:https://doi.org/10.1016/j.ipl.2015.02.004. Available from: http://www.sciencedirect.com/science/article/pii/S0020019015000216

[5] Gupta, U. I.; Lee, D. T.; Leung, J. Y.-T. Efficient algorithms for interval graphs and circular arc graphs. *Networks*, volume 12, no. 4: pp. 459–467, doi:10.1002/net.3230120410, https://onlinelibrary.wiley.com/doi/pdf/10.1002/net.3230120410. Available from: https://onlinelibrary.wiley.com/doi/abs/10.1002/net.3230120410

[6] Henning, M.; Klostermeyer, W.; MacGillivray, G. Bounds for the m-Eternal Domination Number of a Graph. *Contributions to Discrete Mathematics*, volume 12, no. 2, 2017, ISSN 1715-0868. Available from: http://cdm.ucalgary.ca/cdm/index.php/cdm/article/view/538

[7] Finbow, S.; Messinger, M.-E.; van Bommel, M. F. Eternal domination on $3 \times n$ grid graphs. *Australasian J. Combinatorics*, volume 61, 2015: pp. 156–174.

[8] van Bommel, C. M.; van Bommel, M. F. Eternal Domination Numbers of 5 × n Grid Graphs. *Journal of Combinatorial Mathematics and Combinatorial Computing*, to appear. Available from: http://people.stfx.ca/mvanbomm/publicat/eternal5xn.pdf

[9] Klostermeyer, W. F.; MacGillivray, G. Eternal Dominating Sets in Graphs. *Journal of Combinatorial Mathematics and Combinatorial Computing*, volume 68, February 2009: pp. 97–111. Available from: https://www.researchgate.net/publication/283935997_Eternal_dominating_sets_in_graphs

[10] Bard, S.; Duffy, C.; Edwards, M.; Macgillivray, G.; Yang, F. Eternal Domination in Split Graphs. volume 101, May 2017: pp. 121–130. Available from: https://www.mathstat.dal.ca/~cduffy/edom.pdf

[11] Harary, F. *Graph Theory*. Addison-Wesley Publishing Company, Inc., 1969.

[12] Tarjan, R. Depth-First Search and Linear Graph Algorithms. *SIAM Journal on Computing*, volume 1, no. 2, 1972: pp. 146–160, doi: 10.1137/0201010, https://doi.org/10.1137/0201010. Available from: https://doi.org/10.1137/0201010

[13] Cormen, T. H.; Leiserson, C. E.; Rivest, R. L.; Stein, C. *Introduction to Algorithms*. MIT Press, third edition, 2009.

Appendix A

Contents of the enclosed media

```
readme.txt ..................... the file with media contents description
implementation ............................... implementation sources
  Makefile ............. the Makefile used to build the implementation
thesis ................. the directory of LaTeX source codes of the thesis
  Makefile ............. the Makefile used to build the thesis PDF file
BP_Kristan_Jan_Matyas_2018.pdf ....... the thesis text in PDF format
```