Mark van den Bergh

The Game of Cops and Robbers on Geometric Graphs

Master thesis

Supervisors:
dr. T. Müller (UU), dr. F.M. Spieksma

Date master exam: September 21, 2017

Mathematisch Instituut, Universiteit Leiden

The Game of Cops and Robbers on Geometric Graphs

Mark van den Bergh

September 8, 2017

Supervisors:
dr. T. Müller (UU)
dr. F.M. Spieksma

Mathematisch Instituut
Universiteit Leiden

Abstract

The Game of Cops and Robbers is played by two players on a graph, the one player controlling a set of cops and the other controlling a robber. The goal of the cops is to capture the robber in a finite number of moves, whereas the robber wants to evade the cops indefinitely. The central question we ask is: how many cops are needed on a given graph for them to be able to win the game? This amount is called the cop number of a graph.

In this thesis, we give an extensive overview of results that give an upper bound on the cop number for graphs of a given type, like unit disk graphs and string graphs, as well as some examples of these graphs having a high cop number. In this overview, we will see that the results all greatly depend on the statement or proof method of one of two theorems published by Aigner and Fromme [1].

In addition to a survey of the existing work, we explore whether the two main theorems can be used to further sharpen these results. Leading to mostly negative answers, we conjecture that a different approach is necessary in order to book further progress.

Preface

In 1983, a board game called *Scotland Yard* won the prestiguous *Spiel des Jahres* award for being the best board game of the year according to a panel of critics. In this game, named after the headquarters of the London police, an elusive criminal called *Mr. X* tries to avoid capture by a number of police officers who chase him across a map of London. The goal of the policemen is to capture Mr. X within a set number of turns, whereas Mr. X wins the game by evading the officers during this time.

The mathematical field of *extremal combinatorics* deals with the question of how large or small some set or structure needs to be, if it is to adhere to certain requirements. One might for example wonder how many people can attend a party without having a set of three people who either all have met before or are complete strangers to each other. The answer to this question turns out to be five, a proof of which may be found on a page dedicated to *Ramsey's Theorem* [3].

In this thesis, we will explore a game that ties the gameplay of Scotland Yard and the mathematical field of extremal combinatorics together, known as the *Game of Cops and Robbers*. In this game, a set of cops hunt a robber moving along the vertices of some given graph. The cops win the game by capturing the robber in finite time, while the robber wins if he can move without being captured forever. For a given graph, we then ask the question: *how many cops are sufficient and necessary to always capture the robber whatever his strategy?* By trying to find this cop number, we delve into the field of extremal combinatorics introduced above.

Before this small preface comes to an end, some acknowledgements are in place. I would like to thank my kind supervisors Tobias Müller and Floske Spieksma for allowing me great freedom in exploring the given subject, as well as for the fruitful yet always "gezellige" meetings, lacking a suitable English adjective. Furthermore, I would like to thank my loving parents, especially my father for listening to my endless ramblings on the subject and his aid in manufacturing a couple of the many figures in this text.

Contents

1 Introduction

In the field of extremal combinatorics, one asks how large or small a set or structure can or needs to be if it must satisfy certain properties. An example of such a question is: how large can a graph be, if it may not contain a clique or independent set of n vertices for some natural number n? This question in particular gave rise to the subfield of *Ramsey theory*.

Another question drawing closer to the subject we will study in this thesis, concerns the game of tic-tac-toe. Stated loosely: if we fix the diameter or our playing board (the original 3×3 board having a diameter of 3), in what dimension should we play the game of tic-tac-toe, if we always want to have a winner? The answer lies in the *Hales-Jewett Theorem* [22].

In this thesis, we will consider a different game, called the *Game of Cops and Robbers*. This game belongs to the larger class of so-called *pursuit-evasion games*. Other examples of games belonging to this class are the Angel Problem [14] and the Homicidal Chauffeur Problem [24]. The game involving only one cop and one robber was independently introduced by Quillot [35] and Nowakowski and Winkler [33]. Aigner and Fromme introduced a version of the game in which one robber plays against multiple cops [1].

The game in the form we will study here, is played by two players, one controlling a single robber and one controlling a set of $k \geqslant 1$ cops. The robber and every one of the cops start the game by occupying a vertex in a graph. Then, every turn, the robber and the cops may move along an edge of the graph to a neighboring vertex. If a cop manages to land on the vertex currently occupied by the robber, the cops have won. For the robber to win, he must indefinitely avoid capture. Throughout, both players have full information on each other's whereabouts.

The central question posed in this field of study is the following: *given some graph G, how many cops are necessary and sufficient for the cops to always win against the robber?* This amount of cops is called the *cop number*.

Unsurprisingly, to determine the cop number of any arbitrary given graph turns out to be hard. Therefore, it is interesting whether one is able to determine or at least bound the cop number for some set or class of graphs with a certain structure, by smartly exploiting this structure. It is, for example, not hard to show that any tree has cop number 1: a single cop can always win the game on a tree, whatever strategy the robber follows. A deeper result of this type was published by Aigner and Fromme [1], stating that the cop number of any planar graph is at most 3. Results in the same spirit can be found for other classes of graphs. We will focus here on two classes of so-called geometric graphs, being the *unit disk graphs* and *string graphs*.

The goal of this thesis is twofold. First, the thesis aims to give an extensive overview of the existing results on the Game of Cops and Robbers on the geometric graph classes discussed above, as well as some auxilliary results and some results on some other graph classes. The proofs of these results have been taken from various papers and have oftentimes been elaborated, slightly restructured and/or enhanced with figures. In viewing this collection of results on the different types of graphs, we will see that all statements on some upper or lower bound on the cop number of a graph class are extensions or rephrasings of two main theorems, being Proposition 4.1.3 and Theorem 5.1.1.

Second, we will investigate whether these two theorems can be of any further use in application to the classes of unit disk graphs and string graphs. The results of this investigation, with Example 4.2.3, Proposition 5.3.2 and Theorem 5.4.2 being the key installments, are exploratory and mostly negative, showing that some intuitive routes to take unfortunately do not yield stronger results than previously known.

The structure of the thesis is as follows. In Chapter 2, we more formally introduce the Game of Cops and Robbers, as well as the different graph classes we will encounter. The main results of the chapter are Theorem 2.5.8 and Theorem 2.6.5, which provide classifications for graphs having cop number 1 and graphs having cop number $k > 1$, respectively.

In Chapter 3, we will look at the game from a computational point of view, providing two algorithms to determine the cop number of a given graph. Furthermore, we make the statement that "determining the cop number of any given graph is hard" precise in Theorem 3.2.4 and we list some complexity results for several geometric graph classes.

Chapter 4 gives an overview of different results providing upper bounds on the cop number for some graphs. All these results are in some way based on (the proof of) Proposition 4.1.3. Using this as starting point, we will find that planar graphs have cop number at most 3 in Theorem 4.1.6, that unit disk graphs have cop number at most 9 in Theorem 4.2.4 and that string graphs have cop number at most 15 in Theorem 4.3.8. Moreover, we will state an upper bound for general graphs based on the treewidth of the graph.

Then, in Chapter 5, we move in the "opposite" direction compared to the previous chapter, searching for graphs with a high cop number. All constructions made in this chapter are based on the statement of Theorem 5.1.1. With this theorem, we will show that a graph without any restrictions imposed on it, can have an arbitrarily high cop number. Furthermore, we will find a unit disk graph having cop number 3, and we will show that Theorem 5.1.1 cannot be applied to find a unit disk graph with higher cop number. Finally, we explore the possibilities of applying the theorem to string graphs.

2 Background

In this first chapter, we will introduce some of the basic concepts that will be used throughout the thesis, as well as some standard classification results. In Section 2.1, we discuss some notational issues. Section 2.2 is a small interlude on several classes of geometric graphs and the relations between them. In Section 2.3, we somewhat formally establish the rules of the Game of Cops and Robbers and define what we mean by the 'cop number' of a graph, which we compute for some simple graph types in Section 2.4. In Section 2.5 we will then give an elegant classification of all graphs having cop number 1, followed by a slightly less elegant classification of graphs having cop number $k > 1$ in Section 2.6.

2.1 Notation and basic concepts

Before we delve into the rules of the game that we will be studying, we establish some notation that will be used throughout the thesis. We will often use the natural numbers, which we define as

$$\mathbb{N} = \{1, 2, 3, \ldots\}.$$

We write $\mathbb{N}_0 = \mathbb{N} \cup \{0\}$ if we also want to include zero. For any $n \in \mathbb{N}$, we write $[n] = \{1, \ldots, n\}$ as shorthand notation.

Unless stated otherwise, we will always be considering finite graphs $G = (V, E)$, where V is the set of *vertices* or *nodes* of G and $E \subseteq \binom{V}{2}$ is the set of *edges*, using the notation $\binom{V}{2} = \{W \subseteq V : |W| = 2\}$. We sometimes use $V(G)$ and $E(G)$ to denote the vertex resp. edge set of a graph G. Unless stated otherwise, all graphs considered are *simple*, that is, contain no loops or multiple edges. Moreover, for the sake of notational simplicity, we will always consider graphs to be unique up to isomorphism.

For any vertex $v \in V(G)$, we write $N(v)$ for the *neighborhood* of v, which is the set of all vertices connected to v. In set notation, this is

$$N(v) = \{u \in V(G) : uv \in E(G)\}.$$

We write $N_G(v)$ if we need to make explicit that we are considering the neighborhood of v in G. Furthermore, for convenience, we will employ the notation $N[v] = N(v) \cup \{v\}$ for the *closed neighborhood* of v, that is, the neighborhood of v and v itself. If $W \subseteq V$ is a set of vertices, we write $N[W] = \bigcup_{v \in W} N[v]$.

For some $v \in V(G)$, we denote by $d(v)$ or $d_G(v)$ the *degree* of v in G, which is the amount of neighboring vertices of v in G. In short, $d(v) = |N(v)|$. For a graph G, we write $\delta(G) = \min_{v \in V} d(G)$ for the *minimum degree* in G and $\Delta(G) = \max_{v \in V} d(G)$ for the *maximum degree* in G.

Another useful notion is that of a path. A *path* in a graph G is a sequence of vertices $P = (v_1, v_2, \ldots, v_k)$ with $v_i \in V$ such that $v_i v_{i+1} \in E$ for all $i \in [k-1]$. We call $k-1$ the *length* of the path P. If $k > 3$ and $v_k = v_1$, i.e., if the path ends at the same vertex as where it begins, it is called a *cycle*. For a graph G, we denote by $g(G)$ the *girth* of G, which is defined as the length of the shortest cycle in G. If G does not contain any cycles, we set $g(G) = \infty$. Unless stated otherwise, we will always consider *connected* graphs in which every pair of vertices is connected by at least one path.

We conclude this section with the definition of some special types of graphs. By K_n we denote the *complete* graph on n vertices, that is, the unique graph having n vertices of which every pair is connected by an edge. Furthermore, we call a graph G *bipartite* if its vertex set V can be split into two disjoint sets U and W such that $V = U \cup W$ and no

edges exist between two vertices in U or between two vertices in W. By $K_{n,m}$ we denote the complete bipartite graph having vertex set $V = U \cup W$ with $|U| = n$ and $|W| = m$ and all edges between U and W being present.

Having discussed some notational issues and basic concepts from graph theory, we continue on a slight detour to elaborate on some classes of so-called geometric graphs.

2.2 Geometric graphs

Throughout this thesis, we will often use the geometrical structure of the graphs we consider. It is therefore useful to define what we mean by the concept of a drawing of a graph.

Definition 2.2.1. Let G be a graph with vertices $V = \{v_1, \dots, v_n\}$ and edges $E = \{e_1, \dots, e_m\}$. A *drawing* of G is a map $\phi\colon V \to \mathbb{R}^2$ and a set of continuous curves $\gamma_1, \dots, \gamma_m\colon [0,1] \to \mathbb{R}^2$ such that for $e_i = v_j v_k \in E(G)$ we have either $\gamma_i(0) = \phi(v_j)$ and $\gamma_i(1) = \phi(v_k)$ or $\gamma_i(0) = \phi(v_k)$ and $\gamma_i(1) = \phi(v_j)$.

In any drawing of a graph G, we say that two of the curves γ_i and γ_j representing the edges e_i and e_j have a *crossing*, if there are some $t_1, t_2 \in (0,1)$ such that $\gamma_i(t_1) = \gamma_j(t_2)$. In other words, a crossing is some intersection point of two curves which is not a starting nor ending point of both curves. Slightly abusing the terminology introduced, we often say that edges e_i and e_j cross without explicitly referencing the corresponding curves in the drawing. Now, the following definition is intuitive.

Definition 2.2.2. A graph G is called *planar*, if there exists a drawing of G with no crossings.

It is well-known that the complete graph K_5 is non-planar, as is $K_{3,3}$. We will find that the planarity of a graph will play some role in our analysis.

Besides the geometrical structure of a graph itself in some drawing of it, some graphs contain more structure in the form of some different representation. Before we delve deeper into the definitions of these classes of graphs, we introduce the following concept.

Definition 2.2.3. Let $\mathcal{A} = \{A_1, \dots, A_n\}$ be a collection of sets $A_k \subseteq \mathbb{R}^2$. The *intersection graph* of $\mathcal{A}$, denoted by $G_{\mathcal{A}}$, is defined by $V(G_{\mathcal{A}}) = \mathcal{A}$ and

$$E(G_{\mathcal{A}}) = \{\{A_i, A_j\} \in \binom{V}{2} : i \neq j \text{ and } A_i \cap A_j \neq \varnothing\}.$$

In words, an intersection graph models some collection of subsets as $\mathbb{R}^2$ as vertices in a graph that are connected if and only if the associated sets in $\mathbb{R}^2$ intersect. To define a geometric graph, we reverse this procedure.

Definition 2.2.4. Let $G = (V, E)$ be a graph with $V = \{v_1, \dots, v_n\}$ for some $n \in \mathbb{N}$. We say G is *geometric* if there exists a collection $\mathcal{A} = \{A_1, \dots, A_n\}$ of sets $A_k \subseteq \mathbb{R}^2$ such that $G = G_{\mathcal{A}}$.

A graph is thus called a geometric graph if it is the intersection graph of some collection of sets in the plane. If these sets are of a particular form, we obtain a more restriced class of graphs. The following two examples are of interest.

Definition 2.2.5. Let $G = (V, E)$ be a graph with $V = \{v_1, \dots, v_n\}$ for some $n \in \mathbb{N}$.

(i) If there exist $c_1, \dots, c_n$ such that G is the intersection graph of the collection $\mathcal{C} = \{C_1, \dots, C_n\}$ of unit disks $C_k = \{x \in \mathbb{R}^2 : \|x - c_k\|_2 \leqslant 1\}$ centered at c_k, then G is a *unit disk graph*. The collection $\mathcal{C}$ is called the unit disk representation of G.

(ii) If there exist continuous curves $\gamma_1, \dots, \gamma_n \colon [0, 1] \to \mathbb{R}^2$ in the plane such that G is the intersection graph of these curves, then G is called a *string graph*. The collection of curves $\{\gamma_1, \dots, \gamma_n\}$ is called the string representation of G.

Throughout, it is sometimes more convenient to use an alternative definition of unit disk graphs. Note that in the definition above, $C_i \cap C_j \neq \varnothing$ holds if and only if $\|c_i - c_j\|_2 \leqslant 2$. This, in turn, holds if and only if c_i is contained in the disk of radius 2 centered at c_j and c_j is contained in the disk of radius 2 centered at c_i. Therefore, after scaling, a unit disk graph may also be seen as a graph for which the vertices can be mapped into the plane in such a way that two vertices v and w are connected precisely if the image of v is contained in the unit disk centered around the image of w and vice versa.

Now, we wonder what relation there exists between the graph classes introduced above. We immediately obtain the following result.

Proposition 2.2.6. *Every unit disk graph is a string graph.*

Proof. Let G be a unit disk graph with unit disk representation $\mathcal{C} = \{C_1, \dots, C_n\}$, having centers at $c_1, \dots, c_n \in \mathbb{R}^2$. Define for $k \in [n]$ the curve $\gamma_k \colon [0, 1] \to \mathbb{R}^2$ by $\gamma_k(t) = (c_k + \cos(2\pi t), c_k + \sin(2\pi t))$. Note that γ_k desribes the boundary of the disk C_k. Therefore, it is clear that $C_i, C_j \in \mathcal{C}$ intersect if and only if γ_i and γ_j do. Hence, G is the intersection graph of the set $\{\gamma_1, \dots, \gamma_n\}$ and thus a string graph. □

The converse statement does not hold, i.e., the set of unit disk graphs is a proper subset of the set of string graphs.

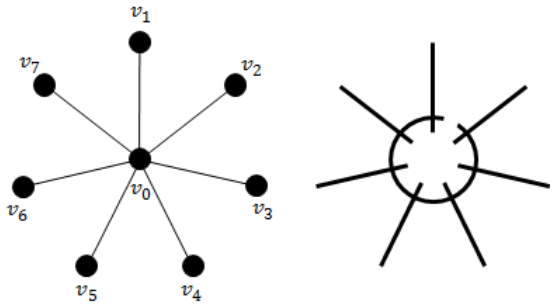

Figure 2.2.1: A drawing and a string representation of $K_{1,7}$.

Proposition 2.2.7. *The graph $K_{1,7}$ is a string graph, but not a unit disk graph.*

Proof. See Figure 2.2.1 for a string representation of $K_{1,7}$. Now, write $V = \{v_0\} \cup \{v_1, \dots, v_7\}$ and suppose $K_{1,7}$ has some unit disk representation $\mathcal{C} = \{C_0, C_1, \dots, C_7\}$ with centers at $c_0, c_1, \dots, c_7$. Assume w.l.o.g. that $c_0 = (0, 0)$. As there are seven centers to be placed in the plane, there must be two centers, say c_i and c_j with $i, j \in [7]$, such that $\angle c_i c_0 c_j \leqslant \frac{2\pi}{7}$.

Consider this triangle $\triangle c_0 c_i c_j$ as depicted in Figure 2.2.2, where $\alpha = \angle c_i c_0 c_j$. As the disks C_i and C_j do not overlap, we have that $z > 2$. Now, as $\alpha \leqslant \frac{2\pi}{7} < \frac{\pi}{3}$, we have

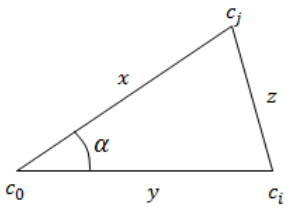

Figure 2.2.2: The triangle $\triangle c_0 c_i c_j$.

that $\sin\alpha < \sin\frac{\pi}{3} = \frac{\sqrt{3}}{2}$, so $\frac{z}{\sin\alpha} > \frac{4}{\sqrt{3}}$. Therefore, by the law of sines, also $\frac{x}{\sin\angle c_0 c_i c_j} > \frac{4}{\sqrt{3}}$. As the disks C_0 and C_j intersect, we have that $x \leqslant 2$. Therefore, we conclude that $\sin\angle c_0 c_i c_j < \frac{\sqrt{3}}{2}$, so that $\angle c_0 c_i c_j < \frac{\pi}{3}$. Similarly, we derive that $\angle c_0 c_j c_i < \frac{\pi}{3}$. But then the sum of the angles in $\triangle c_0 c_i c_j$ adds up to strictly less than π, which is a contradiction. □

Furthermore, note that the set of string graphs is in turn a proper subset of the set of all graphs, which the following result shows.

Proposition 2.2.8 ([15]). *Define the graph G by subdividing every edge of K_5 into two, see also Figure 2.2.3. Then G is not a string graph.*

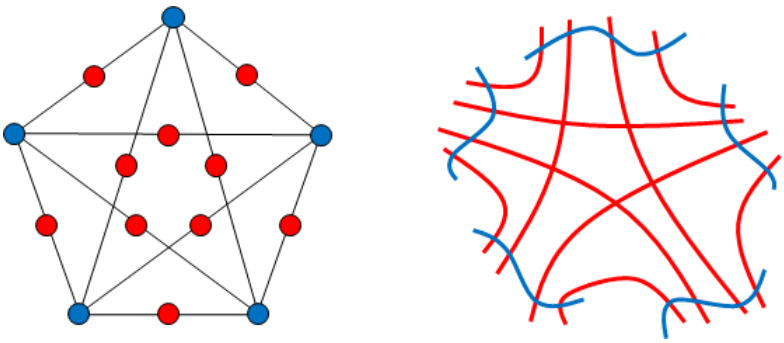

Figure 2.2.3: The graph G and an erroneous string representation.

Proof. Let $V_1 = \{v_1, \ldots, v_5\}$ in $V(G)$ be the 'original' vertices of K_5, colored blue in the picture, and let $V_2 = \{v_6, \ldots, v_{15}\}$ be the nodes added by the subdivision of the edges, labeled red in the figure. Let $\{\gamma_1, \ldots, \gamma_{15}\}$ be some string representation of G, supposing that such a representation exists.

Suppose that $\gamma_i(0) = x_i$ for all $i \in \{1, \ldots, 5\}$. Now, we shrink the strings $\gamma_1, \ldots, \gamma_5$ in a continuous way until $\gamma_i(t) = x_i$ for all $i \in \{1, \ldots, 5\}$ and $t \in [0, 1]$. Noting that none of the vertices in V_1 are connected, we find that none of these strings should intersect among themselves, so by shrinking them to distinct single points we do not add or remove intersections between these strings.

We do, however, add intersections among the strings $\gamma_6, \ldots, \gamma_{15}$, as in every set of four strings that intersected one of the strings γ_i, $i \in \{1, \ldots, 5\}$, in our original representation, these four strings now also intersect each other. Now, note that by these strings, every one of the five points $x_1, \ldots, x_5$ is connected to every other point. As such, the resulting representation is a drawing of K_5. However, as no two vertices in V_2 are adjacent, no two strings from $\{\gamma_6, \ldots, \gamma_{15}\}$ may intersect in our original string representation. As we

have only added intersections to this representation in the points $x_1, \ldots, x_5$ and as K_5 is not planar, this gives a contradiction. Hence, the required string representation does not exist. □

What about the planarity of the geometric graphs in Definition 2.2.5? The class of planar graphs turns out to be a proper subset of the class of string graphs.

Proposition 2.2.9 ([15])**.** *Every planar graph is a string graph.*

Proof. Let G be some planar graph and fix some drawing ϕ, $\gamma_1, \ldots, \gamma_m$ of G with no crossings. We may then define for every vertex v_i a curve γ_i' which covers the first half of every γ_i starting in $\phi(v_i)$. See Figure 2.2.4 for an example. By the planarity of our drawing, two such γ_i' intersect if and only if the two corresponding vertices were connected by some γ_j in the drawing. □

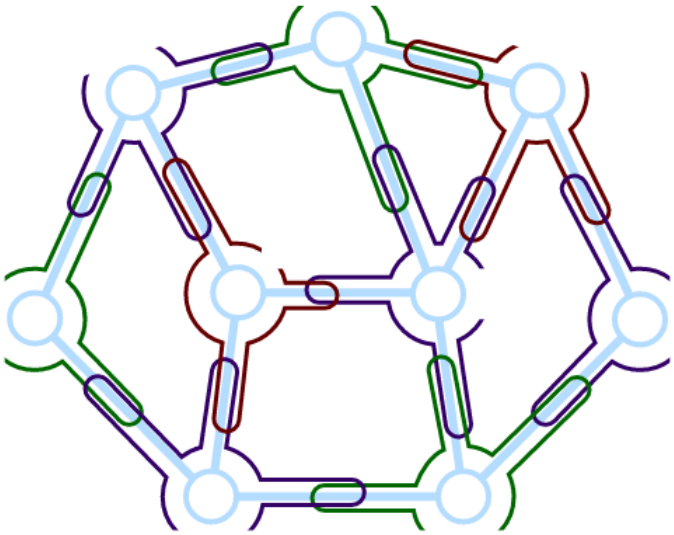

Figure 2.2.4: A string representation of a planar graph [16].

Proposition 2.2.10. *The graph K_5 is a string graph, but not planar.*

Proof. See Figure 2.2.5 for a string representation of K_5. The fact that K_5 is non-planar is well-known and can be viewed as part of Kuratowski's Theorem [30]. □

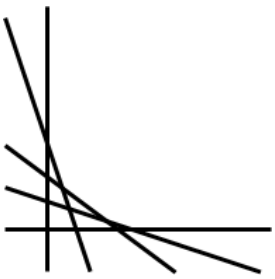

Figure 2.2.5: A string representation of K_5.

It even turns out that every planar graph can be represented as the intersection graph of a set of straight line segments in the plane [11], a fact which was known as Scheinerman's conjecture until it was proven several years ago.

Finally, we note that the class of unit disk graphs is not contained in the graph of planar graphs, nor vice versa. Indeed, $K_{1,7}$ is planar, but not a unit disk graph by virtue of Proposition 2.2.7. The other way around, K_5 is clearly a unit disk graph, as it can be viewed as the intersection graph of five unit disks having a center in $[0, \frac{1}{10}]^2$.

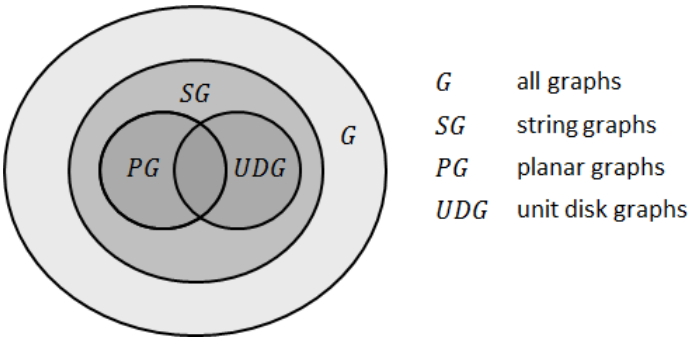

Figure 2.2.6: The relations between the different classes of graphs.

The results of our small tour through the world of geometric graphs are depicted in the Venn diagram in Figure 2.2.6. We now continue with a true introduction to the subject of study.

2.3 Rules of the game

The Game of Cops and Robbers is a two-player pursuit-evasion game played on any graph G. One player controls a single robber and the other controls a set of k cops, each of which occupy some vertex in the graph. During the game, both players always have complete information on the position(s) of the other player. The goal of the cops is to capture the robber, while the goal of the robber is to avoid capture for as long as possible. In line with tradition, we will often refer to the robber as a male figure, whereas the cops are considered female.

In the first round, round 0, every cop must be placed on some vertex in the graph as starting position. Multiple cops may occupy the same vertex at the same time. Next, the robber must pick a starting vertex in the graph. Now, the players take turns, starting with the player controlling the cops. During the cops' turn, each cop may move to an adjacent vertex via some edge originating from the current position. Then, during the robber's turn, he may also move to an adjacent vertex in the graph.

While we have used the verb 'may' in the previous sentences, this is sometimes replaced by 'must'. In fact, there is often made a distinction between the form of the game *without passes*, in which all cops and the robber must always travel along an edge every turn, and the game *with passes allowed*, in which the cops and robber are allowed to 'pass' and stay on the same vertex for a turn. The former is also known in the literature as the *active* form of the game, the latter as the *passive* version. By adding loops to each vertex in the graph, an instance of the game without passes can be transformed into an instance of the game with passes allowed, as a cop or robber can now decide to stay on the same vertex by travelling along a loop. Apart from these loops, multiple edges between vertices play no role. In the sequel, we will assume that we are playing the game with passes allowed on a simple graph, i.e., a graph without loops or multiple edges.

Now, the cops win the game if some cop manages to occupy the same vertex as the robber at the end of her turn. We then say that the robber has been caught. Conversely, if the robber manages to infinitely stay out of the hands of the cops, we say that the robber wins the game. A natural question to ask now is how many cops are needed to capture the robber on a given graph. This question leads to the following graph parameter and embodies the main questions of this thesis.

Definition 2.3.1. Let G be a graph. We define the *cop number* of G, written $c(G)$, as the minimum amount of cops that can win the Game of Cops and Robbers on G. In other words, $c(G)$ is the minimum number of cops necessary to be able to capture the robber in finite time.

Before turning to compute the cop number of some simple classes of graphs in the next section, we introduce one more intuitive piece of notation. We write R to denote the position of the robber in the graph, or the robber in general. Similarly, we let C denote the set containing the positions of the cops in the graph, or the set of cops themselves.

2.4 Some examples

Being faced with this new graph parameter $c(G)$, we compute its value for some simple graphs. In doing so, we will see that the concept of the so-called "robber territory", the part of the graph in which the robber can freely move without being caught by the cops, plays an important role. This will also be the case in later proofs.

Example 2.4.1. Let G be a tree. We claim that $c(G) = 1$. To prove this claim, we need to find a strategy that allows the cop to capture the robber no matter what the robber does. In round 0, the cop may choose some vertex in G at random. The robber must now pick some vertex in $V(G)\backslash N[C]$ to start in or he is captured immediately in the first turn of the cop.

Now, note that by removing C and all incident edges from the tree, the graph would fall apart into connected components (possibly one component if C was a leaf). Let G_R be the connected component in which R is situated. There must be an edge connecting C to G_R, as otherwise G_R would have been disconnected from the rest of the graph in the first place. During her turn, the cop moves along this edge to some vertex in G_R.

By this move, the component in which the robber can move is reduced in size by one: the vertex to which the cop moved is now off-limits. Furthermore, it cannot increase in size, as this would imply that there is another route to the original vertex in which the cop started, which would give a cycle. Now, no matter where the robber goes during his turn, this process can be repeated in every next turn of the cop, reducing the size of the component in which the robber can freely move by one each turn. As the graph is finite, this implies that at some point the size of the "robber territory" becomes zero, i.e., the robber has been caught.

Example 2.4.2. Let G be a cycle containing at least 4 vertices. Then $c(G) = 2$. To prove the validity of this statement, we need to not only construct a winning strategy for two cops, but also a winning strategy for the robber against one cop. We start with the latter. Let C be the vertex chosen by the cop in round 0. Now, the robber chooses some vertex $R \in V(G)\backslash N[C]$, which is possible because the graph has at least 4 vertices and $|N[C]| = 3$ no matter what C is, as every node in the cycle has degree 2. Next, in every turn of the cop, she moves either clockwise or counterclockwise through the cycle, or passes. If the cop passes, the robber passes as well. Otherwise, the robber moves in the same direction as the cop did. By doing so, the distance between the cop and the

robber after the robber's move always remains the same and as such the robber is never caught.

When the game is played by two cops, the robber is less lucky. In round 0, let the cops choose two distinct vertices v and w such that $v \notin N[w]$, which is again possible as the graph has at least four vertices. Similar to the previous example, the removal of v and w would cause the graph to fall apart in two components, one of which is the component where the robber is situated. Every turn, by moving both cops into this component, the size of the "robber territory" is reduced by two. Therefore, the robber is eventually caught.

Example 2.4.3. Let $G = K_n$ be the complete graph on n vertices. It is not hard to see that then $c(G) = 1$. Indeed, every vertex in G is connected to every other vertex, so $N[C] = V(G)$ no matter which vertex C is the starting vertex. Therefore, whichever vertex the robber picks to start at in round 0, the cop will always capture him in her first move.

Example 2.4.4. Let $G = K_{n,m}$ be a complete bipartite graph with $n, m \geqslant 2$. Then $c(G) = 2$. Again, first consider the game of the robber playing against one cop. For convenience, we write $V(G) = U \cup W$ with $|U| = n$ and $|W| = m$. Suppose that the cop chooses a vertex $C \in U$ to start at in round 0. The robber then chooses $R \in U$ different from C to start in, which is possible as $n \geqslant 2$ by assumption. In the first turn of the cop, she moves to some vertex $C' \in W$. In reaction, the robber must then move to some $R' \in W$ different from C', which is again possible as $m \geqslant 2$ and all nodes in U are connected to all nodes in W. This process repeats, moving both the cop and robber back and forth between U and W without them ever meeting. Noting that both players may also start in W, we conclude that the robber wins against one cop.

When faced with two cops, however, the robber quickly loses. In round 0, the first cop picks some vertex in U while the second cop moves to occupy some vertex in W. Now $N[C] = V(G)$, so wherever the robber starts the game, he is captured in the first round.

2.5 Dismantlability

Of course, we are interested in determining the cop number for more graphs than those discussed in the previous section. Does there exist a classification of the graphs having cop number k for some given $k \in \mathbb{N}$? Later on, we will see that this problem is very hard for $k \geqslant 2$. However, there does turn out to be an elegant way of describing the graphs having cop number 1, which is the main focus of this section. Throughout, we will call any graph G having cop number $c(G) = 1$ *cop-win*.

For one cop to be able to win on a graph, she needs to be able to somehow 'close in' on the robber. Central to her strategy will be the concept of a *corner* in a graph. Intuitively, a corner is some vertex from which the robber cannot easily escape, as there is another vertex from which the cop can guard all exits of the corner. Formally, it is defined as follows.

Definition 2.5.1. Let G be a graph and let $v \in V(G)$. Then v is called a *corner* in G if there exists some $w \in V(G)$ such that $N[v] \subseteq N[w]$. We then say that w *covers* v in G.

We will show that corners are useless for the robber, in the sense that any winning strategy for the robber never enters any corner. This is not surprising, as being in a corner seems to be dangerous for the robber, having little ways to escape. In fact, we will first prove an even stronger statement, of which our result will then be a corollary. To formulate this statement, we require two more definitions.

Definition 2.5.2. Let G and H be graphs. A map

$$f\colon (V(G), E(G)) \to (V(H), E(H))$$

is called a *graph retraction* if $uv \in E(G)$ implies that either $f(u)f(v) \in E(H)$ or $f(u) = f(v)$ for any pair $u, v \in V(G)$.

Definition 2.5.3. Let G be a graph. A subgraph H of G is called a *retract* of G if there exists some graph retraction $f\colon G \to H$ such that $f(v) = v$ for every $v \in V(H)$.

Theorem 2.5.4 ([6])**.** *Let G be a graph and let H be a retract of G. Then $c(H) \leqslant c(G)$.*

Proof. Let $f\colon G \to H$ be a graph retraction mapping $V(H)$ to itself and suppose $c(G) = k$. We will construct a strategy for k cops to win against the robber on H. To do so, we consider two coupled games of Cops and Robbers, both being played by k cops and one robber, one being played on G and the other on H.

In round 0, the cops choose a set of vertices C in G to start from. For the game in H, we let the cops start on $f[C]$. Now, the robber first chooses a vertex R to start from in H. Then the robber in G starts in the same vertex. During the game, every turn, first the cops in G move. Then, if a cop moved from v to w in G, we let the image of this cop move from $f(v)$ to $f(w)$ in H, which is possible as f is a graph retraction. Next, the robber in H moves however he wants, after which the robber in G copies this move, which is possible as H is a subgraph of G.

As $c(G) = k$, the game must arrive at a point in which the robber in G can make one more move before he is caught by the cops. At this point, we have that $N_G[R] \subseteq N_G[C]$. Hence, there is a cop on some vertex $v \in V(G)$ such that $vR \in E(G)$ and for any w with $wR \in E(G)$ there is a cop on a vertex $w' \in V(G)$ such that $ww' \in E(G)$. Then also $f(v)f(R) = f(v)R \in E(H)$ and $f(w)f(w') \in E(H)$, so $N_H[f(R)] = N_H[R] \subseteq N_H[f(C)]$. Therefore, the robber in H is also caught after his next move, regardless of what he does. This shows that k cops win in H, so $c(H) \leqslant k$. □

Corollary 2.5.5. *Let G be a graph and let H be a retract of G. If G is cop-win, then H is also cop-win.*

Corollary 2.5.6. *Let G be a graph, let $v \in V(G)$ be a corner of G and let H be the subgraph of G formed by removing v and all incident edges from G. If G is cop-win, then H is also cop-win.*

Proof. Suppose w covers v in G. Define $f\colon G \to H$ by

$$f(u) = \begin{cases} u & \text{for } u \neq v, \\ w & \text{for } u = v. \end{cases}$$

Consider some edge $uu' \in E(G)$. If $u \neq v$ and $u' \neq v$, then $f(u)f(u') = uu' \in E(H)$. If $u = v$ and $u' = w$, then $f(u) = w = f(u')$. If $u = v$ and $u' \neq w$, then we note that as $N_G[v] \subseteq N_G[w]$ by assumption, the fact that $vu' \in E(G)$ implies that $wu' \in E(G)$. Hence $f(u)f(u') = wu' \in E(G)$. This proves that f is a graph retraction. Furthermore, it is clear that $f(u) = u$ for all $u \in V(G)\backslash\{v\} = V(H)$. Therefore, H is a retract of G, so H is cop-win. □

Using Theorem 2.5.4, we can now give a nice classification of cop-win graphs, which was already published in 1984 in one of the first papers on the subject of cops and robbers [1]. It is based upon the following concept.

Definition 2.5.7. Let G be a graph. Set $G_0 = G$. We say that G is *dismantlable* if there exists a sequence of graphs $G_1, \ldots, G_n$ such that

(i) For all $i \in [n]$, G_i is a subgraph of G_{i-1} formed by removing a corner and all incident edges from G_{i-1};

(ii) $G_n = K_1$.

Theorem 2.5.8 ([1])**.** *Let G be a graph. Then G is cop-win if and only if G is dismantlable.*

Proof. "⇒": Suppose that G is cop-win. First, note that G must contain a corner. Indeed, consider the point in the game at which the robber has one more move before being caught by the cop. As the cop can capture the robber whatever he does during his turn, we find that $N[R] \subseteq N[C]$, so C covers R and R is a corner.

We continue by induction on the amount of vertices n in G. For the base case, $n = 1$, we find that $G = K_1$ so that dismantlability trivially holds. Next, suppose that any cop-win graph on n vertices is dismantlable and consider some cop-win graph G on $n+1$ vertices. By the previous reasoning, G must contain some corner v. By Corollary 2.5.6, the graph H formed by deleting v and all incident edges from G is also cop-win. As H is a graph on n vertices, by the induction hypothesis we conclude that H is dismantlable. Hence G itself is dismantlable.

"⇐": We again proceed by induction on the amount of vertices n. For $n = 1$, we have that $G = K_1$, so naturally G is cop-win. As induction hypothesis, assume that any dismantlable graph on n vertices is cop-win. Consider a dismantlable graph G on $n + 1$ vertices. By the dismantlability, G must have some corner v such that the graph H formed by deleting v and all incident edges is dismantlable. As H is a graph on n vertices, we also have by the induction hypothesis that H is cop-win.

Now, just like in the proof of Theorem 2.5.4, we will consider two games played in parallel, one on G and one on H. Every turn, the cop in H first makes a move, after which the cop in G copies this move. Then, the robber in G makes a move, which is copied by the robber in H, except for when the robber moves to or from v. In this case, the robber in H moves to or from its covering vertex w instead, which is possible as $N[v] \subseteq N[w]$.

As H is cop-win, at some point in time, the robber is captured there. If on that moment $R = C \neq v$ in G, then the robber in G is captured, as well. If $R = v$ in G, we have that $C = w$. Hence, the robber will be captured in G in the next turn. □

To conclude this section, we apply the statement of Theorem 2.5.8 to the examples given in the previous section. To start, note that any tree is dismantlable, as any leaf is a corner and may therefore be removed. By repeating this process, we may continue to remove leaves until we are left with a single vertex. It therefore immediately follows that any tree is cop-win.

Similarly, any vertex in a complete graph K_n is a corner which is covered by every other vertex in the graph. We may therefore remove all the vertices save one in random order, showing dismantlability. It follows that every complete graph is cop-win.

In contrast, a cycle of length at least four does not contain any corners. Suppose the vertices are labeled $v_1, \ldots, v_n$ with $n \geqslant 4$. For any vertex v_i, $i \in [n]$, we then have that $N[v_i] = \{v_{i-1}, v_i, v_{i+1}\}$, with indices taken modulo n if necessary. It is therefore clear that $N[v_j] \subseteq N[v_i]$ does not hold for any distinct $i, j \in [n]$. Hence by Theorem 2.5.8, cycles are not cop-win, which is in accordance with the result from the previous section.

Similarly, any complete bipartite graph $K_{n,m}$ with $n, m \geqslant 2$ does not contain any corners. Writing $V(K_{n,m}) = U \cup W$ with $|U| = n$ and $|W| = m$, we have that $N[u] = \{u\} \cup W$ for any $u \in U$ and $N[w] = U \cup \{w\}$ for any $w \in W$. By considering cases, we again conclude that $N[v] \subseteq N[v']$ cannot hold for distinct $v, v' \in V(K_{n,m})$, so $K_{n,m}$ is not dismantlable and therefore not cop-win.

2.6 Graph products and k-cop-win graphs

The classification of graphs having cop number at most k, also known as k-cop-win graphs, is much harder and somewhat less elegant than the characterisation in the previous paragraph. It depends heavily on the use of so-called graph products. Therefore, let us start with two definitions.

Definition 2.6.1. Let G and H be graphs. We denote the *categorical product* (also called *direct product*) of G and H by $G \times H$. We define $V(G \times H) = V(G) \times V(H)$ and set $(a, b)(c, d) \in E(G \times H)$ if and only if $ac \in E(G)$ and $bd \in E(H)$.

Definition 2.6.2. Let G and H be graphs. We denote the *strong product* of G and H by $G \boxtimes H$. We define $V(G \boxtimes H) = V(G) \times V(H)$ and set $(a, b)(c, d) \in E(G \boxtimes H)$ if and only if one of the following holds:

- $a = c$ and $bd \in E(H)$; or
- $b = d$ and $ac \in E(G)$; or
- $ac \in E(G)$ and $bd \in E(H)$.

In words, in a graph product, we find that every vertex represents a pair of vertices, one from G and one from H. In the categorical product, two pairs of vertices are connected in the product precisely if both components are connected in G and H, respectively. In the strong product, two pairs are also connected if both components are connected in the original graphs, as well as if one of the components is equal and the other has a connection. This is all best clarified by an example.

Example 2.6.3. Consider the two graphs G and H in Figure 2.6.1, with $G = K_3$ and H a path of length 4. The categorical product $G \times H$ and the strong product $G \boxtimes H$ are shown in Figures 2.6.2 and 2.6.3, respectively. We see that in $G \times H$, two vertices are connected if their first coordinate differs and the second coordinate differ by exactly one, which precisely represent the criteria to be connected in K_3 and a path, respectively. In $G \boxtimes H$, two vertices are also connected if either coordinate is equal.

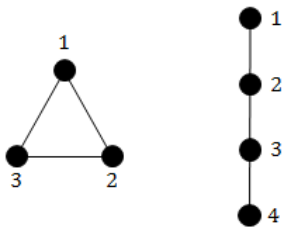

Figure 2.6.1: The graphs G and H.

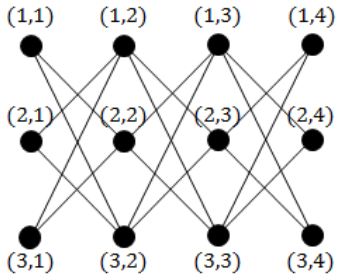

Figure 2.6.2: The categorical product $G \times H$.

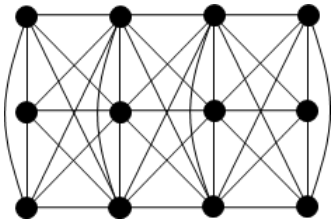

Figure 2.6.3: The strong product $G \boxtimes H$.

The following property of the products comes in handy.

Lemma 2.6.4. *The categorical and strong products are associative, i.e., it holds that*

$$(G \times H) \times J = G \times (H \times J) \quad \textit{and} \quad (G \boxtimes H) \boxtimes J = G \boxtimes (H \boxtimes J)$$

for any graphs G, H and J, with equality up to isomorphism.

Proof. We prove the statement for the categorical product; the proof for the strong product is similar. By definition, $V((G \times H) \times J)$ consists precisely of all vertices having a label of the form $((a, b), c)$ with $a \in V(G)$, $b \in V(H)$ and $c \in V(J)$. Similarly, the vertex sex $V(G \times (H \times J))$ contains exactly all vertices labeled $(a, (b, c))$ with $a \in V(G)$, $b \in V(H)$ and $c \in V(J)$. Therefore, by dropping the inner brackets, we see that the vertex sets of both graphs coincide.

Now, let $(a, b, c)(d, e, f) \in E((G \times H) \times J)$. We then have that $(a, b)(d, e) \in E(G \times H)$ and $cf \in E(J)$ by the definition of the categorical product. Applying the definition once more, we see that $ad \in E(G)$ and $be \in E(H)$ must hold. Then also $(b, c)(e, f) \in E(H \times J)$, so $(a, b, c)(d, e, f) \in E(G \times (H \times J))$. Completely analogously, we can show that any edge in $E(G \times (H \times J))$ also belongs to $E((G \times H) \times J)$, which shows that the graphs are indeed equal. □

The lemma allows us to generalise both products to more than two graphs in the obvious way. For graphs $G_1, G_2, \ldots, G_n$, we may thus speak of

$$\prod_{i=1}^{n} G_i = G_1 \times G_2 \times \ldots \times G_n$$

and

$$\boxtimes_{i=1}^{n} G_i = G_1 \boxtimes G_2 \boxtimes \ldots \boxtimes G_n.$$

When all factors are equal, say $G_i = G$ for all $i \in [n]$, it is convenient to speak of the power of a graph, which is denoted by

$$G^n = \prod_{i=1}^{n} G = \underbrace{G \times G \times \ldots \times G}_{n \text{ times}}$$

and

$$G_{\boxtimes}^n = \bigboxtimes_{i=1}^{n} G = \underbrace{G \boxtimes G \boxtimes \ldots \boxtimes G}_{n \text{ times}}.$$

Now, noting that we are playing the game with passes allowed so that any cop may decide to pass for a round, remark that we may represent the position of k cops in a graph G by considering one cop in the k-th power of the strong product $G_{\boxtimes}^k$. Indeed, in any one round, all of the k cops may stay where they are or move to an adjacent vertex of G. In $G_{\boxtimes}^k$, this is modelled by connecting two vertices $(v_1, \ldots, v_k)$ and $(w_1, \ldots, w_k)$ precisely if either $v_i = w_i$ or v_i is connected to w_i for all $i \in [k]$. The cop being on the vertex $(v_1, \ldots, v_k)$ in $G_{\boxtimes}^k$ thus represents the i-th cop being on vertex v_i in G. Note that if we would want to play the version of the game without passes, in which each cop must move every turn, we may simply replace $G_{\boxtimes}^k$ by G^k.

The cops now together being represented as one cop occupying a single vertex, it would be intuitive and convenient if G was k-cop-win if and only if $G_{\boxtimes}^k$ is cop-win, which can be checked by checking dismantlability. Unfortunately, it is not that simple. Indeed, if we would let the cop in $G_{\boxtimes}^k$ play against a robber in this graph, this single robber would actually correspond to k robbers in the original graph G. Hence, $G_{\boxtimes}^k$ is cop-win if and only if k cops can capture k robbers in G in finite time, which is an entirely different game.

Therefore, we need to still consider the robber as occupying a vertex in G and not in $G_{\boxtimes}^k$. To do so, we construct a sequence of relations on the set $V(G) \times V(G_{\boxtimes}^k)$, described in [13]. For $v \in V(G)$ and $w \in V(G_{\boxtimes}^k)$, we say that $v \leqslant_0 w$ if the vertex w represents at least one of the cops occupying v, i.e., $w = (v_1, \ldots, v_k)$ with $v_i = v$ for at least one $i \in [k]$. For $i \in \mathbb{N}$, we inductively define that $v \leqslant_i w$ if and only if for every $v' \in N_G[v]$ there is some $w' \in N_{G_{\boxtimes}^k}[w]$ such that $v' \leqslant_j w'$ for some $j < i$.

Intuitively, this means that if $v \leqslant_i w$, then if the robber is on v and the cops are on w, the robber can be caught in at most i rounds. Indeed, if the robber moves to some v', then the cops can move to some w' so that $v' \leqslant_j w'$ for some smaller number j. Iterating this process, eventually $R \leqslant_0 C$ must hold and the robber is captured.

Note that the relations $\leqslant_i$ form a non-decreasing sequence: for $j < i$, we have that $\leqslant_j \subseteq \leqslant_i$. To show this, suppose that $v \leqslant_j w$ for some $v \in V(G)$, $w \in V(G_{\boxtimes}^k)$ and let $i > j$. By the definition of $\leqslant_j$, for every $v' \in N_G[v]$ we can find a $w' \in N_{G_{\boxtimes}^k}[w]$ such that $v' \leqslant_\ell w'$ for some $\ell < j$. Noting that $j < i$, the definition for $v \leqslant_i w$ is immediately satisfied.

Now, as $V(G) \times V(G_{\boxtimes}^k)$ is finite (it contains at most $|V(G)|^{k+1}$ elements), there must be some $M \in \mathbb{N}$ such that $\leqslant_M = \leqslant_{M+1}$ holds. We define $\preceq = \leqslant_M$. Intuitively, $v \preceq w$ for some $v \in V(G)$ and $w \in V(G_{\boxtimes}^k)$ precisely if the cops can capture the robber in finite time if the robber starts from v and the cops start in w. We therefore arrive at the following statement.

Theorem 2.6.5 ([13])**.** *Let G be a graph. Then $c(G) \leqslant k$ if and only if there exists some $w \in V(G_{\boxtimes}^k)$ such that $v \preceq w$ for every $v \in V(G)$.*

Proof. "⇒": First, we claim that if $v \not\preceq w$ for some $v \in V(G)$, $w \in V(G^k_{\boxtimes})$, then there must exist a $v' \in N_G[v]$ such that $v' \not\preceq w'$ for all $w' \in N_{G^k_{\boxtimes}}[w]$. Suppose not, i.e., let $v \not\preceq w$ and suppose that for all $v' \in N_G[v]$ there is some $w' \in N_{G^k_{\boxtimes}}[w]$ such that $v' \preceq w'$. By definition, this means that $v' \leqslant_M w'$ for $M \in \mathbb{N}$ defined above. But then $v \leqslant_{M+1} w$ holds, which is a contradiction. This proves the claim.

Now, assume that $c(G) \leqslant k$, but that for all $w \in V(G^k_{\boxtimes})$ there is some $v \in V(G)$ such that $v \not\preceq w$. We will prove by induction that the robber can then always escape, so that $c(G) > k$, a contradiction. Let $w_0 \in V(G^k_{\boxtimes})$ be the chosen starting position of the cop. By assumption, there is some $v \in V(G)$ such that $v \not\preceq w_0$. By the earlier claim, there is then some $v_0 \in N_G[v]$ such that $v_0 \not\preceq w'$ for all $w' \in N_{G^k_{\boxtimes}}[w_0]$. The robber picks v_0 to start.

Wherever the cop moves during her turns, say to $w_1 \in V(G^k_{\boxtimes})$, we thus have that $v_0 \not\preceq w_1$. The robber can then find a $v_1 \in N_G[v_0]$ such that $v_1 \not\preceq w'$ for all $w' \in N_{G^k_{\boxtimes}}[w_1]$ to move to. By inductively continuing in this fashion, the robber can make sure that $R \not\preceq C$ holds at any point during the game. Noting that this implies that $R \not\leqslant_0 C$, the robber indefinitely avoids capture.

"⇐": Claim: if $R \leqslant_i C$ for some $i \leqslant M$, then the cop can capture the robber in at most i moves. The proof goes by induction to i. The base case $R \leqslant_0 C$ is obvious: the robber is immediately caught. Hence suppose that the statement holds for some $i < M$ and assume that $R \leqslant_{i+1} C$. By the definition of $\leqslant_{i+1}$, to whichever vertex $R' \in N_G[R]$ the robber moves, there is some $C' \in N_{G^k_{\boxtimes}}[C]$ such that $R' \leqslant_j C'$ for some $j < i+1$. By the induction hypothesis, the robber is then caught by the cop after at most $j < i+1$ moves and we are done.

Suppose now that there exists some $w \in V(G^k_{\boxtimes})$ such that $v \preceq w$ for all $v \in V(G)$. The cop chooses $C = w$ as her starting position. Then, no matter on which vertex $R \in V(G)$ the robber starts, always $R \preceq C$ will hold. Hence by definition $R \leqslant_M C$, so the robber is always caught in at most M rounds. □

As an illustration of the use of the theorem, we apply it to cycles, which were already proven to have cop number 2 in Example 2.4.2.

Example 2.6.6. Let G be a cycle of n vertices. In the previous section, we have proven that $c(G) > 1$ using the result of Theorem 2.5.8. Now we will re-establish the result that $c(G) = 2$ from Example 2.4.2 by applying Theorem 2.6.5 to show that $c(G) \leqslant 2$.

Consider the graph $G^2_{\boxtimes}$ which is partly drawn in Figure 2.6.4; note that the vertices on the boundary are connected with each other in the same pattern, which has been left out because of practical reasons. First, we find that

$$\leqslant_0 = \{(v, (w, u)) \in V(G) \times V(G^2_{\boxtimes}) \mid v = w \text{ or } v = u\}.$$

In other words, for every $(w, u) \in V(G^2_{\boxtimes})$ we have that $v \leqslant_0 (w, u)$ for all $v \in V(G)$ numbered equal to the row or the column that (w, u) is in. Next, the definition yields that $v \leqslant_1 (w, u)$ if and only if for every $v' \in N_G[v] = \{v-1, v, v+1\}$ we can find some $(w', u') \in N_{G^2_{\boxtimes}}[(u, w)] = \{w-1, w, w+1\} \times \{u-1, u, u+1\}$ such that $v' = w'$ or $v' = w'$, where addition is taken modulo n. We find that this boils down to

$$v \leqslant_1 (w, u) \Leftrightarrow v \in \{w-1, w, w+1\} \text{ or } v \in \{u-1, u, u+1\}.$$

Iterating this procedure, we obtain for $i = 2, 3, \ldots$ that

$$v \leqslant_i (w, u) \Leftrightarrow v \in \{w-i, \ldots, w, \ldots, w+i\} \text{ or } v \in \{u-i, \ldots, u, \ldots, u+i\}.$$

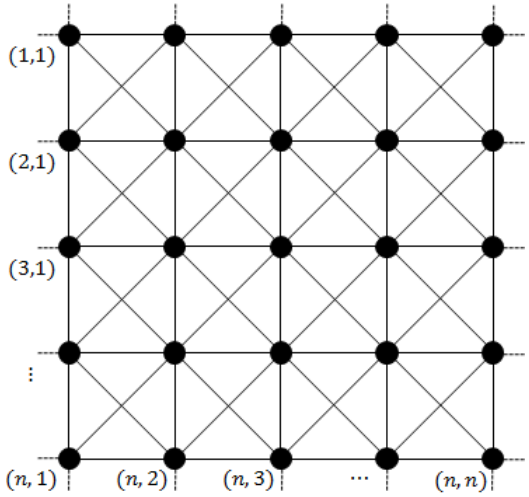

Figure 2.6.4: A partial representation of the graph $G^2_{\boxtimes}$.

Hence, for $M = \lfloor \frac{n}{2} \rfloor$, it follows that $v \leqslant_M (w, u)$ for all $v \in V(G)$ and $(w, u) \in V(G^2_{\boxtimes})$ so that $\leqslant_M = \leqslant_{M+1}$. Therefore, $v \preceq (w, u)$ for all v and (w, u). The requirement of Theorem 2.6.5 is as such certainly satisfied, which yields that $c(G) \leqslant 2$. Combined with the results from the previous section, we once more conclude that $c(G) = 2$.

Though Theorem 2.6.5 provides an algorithmic way of checking whether a graph is k-cop-win, the construction of $\preceq$ is in general tedious. In the next chapter, we will indeed see that the problem of deciding whether the cop number of a graph is at most k is computationally hard.

3 Computational aspects

In this section, we consider the computational complexity of the game of cops and robbers, the treatment of which somewhat closely follows Section 5 of [9]. Given a graph G and some natural number k, how hard is it to determine whether $c(G) \leqslant k$? Unsurprisingly, for graphs in general, this problem turns out to be very hard indeed, if k is allowed to be part of the input.

Before we start, we shortly recall the complexity classes that we will encounter. For a detailed overview, see, e.g., [5]. A decision problem is said to be in the class P if it is polynomially solvable, that is, there exists some algorithm with runtime polynomial in the size of the input, that solves the problem. A problem is contained in the class NP, if it is solvable in polynomial time by a non-deterministic Turing machine, or, equivalently, if the validity of a solution of the problem can be checked in polynomial time. A problem is said to be in the class EXPTIME if there exists an algorithm which solves the problem running in time exponential in the size of the input. It is clear that $\mathrm{P} \subseteq \mathrm{NP} \subseteq \mathrm{EXPTIME}$, but it is unknown whether these inclusions are strict.

Finally, we say a decision problem A reduces to a decision problem B if for any instance I of the problem A, we can algorithmically construct in polynomial time an instance I' of B such that I is a yes-instance of A if and only if I' is a yes-instance of B. Now, a problem A is said to be NP-hard if any decision problem in the class NP can be reduced to A. If A itself is also contained in NP, then we say that A is NP-complete. Similar definitions hold for the terms EXPTIME-hard and EXPTIME-complete. Note, to show that a problem A is NP-hard, it is sufficient to reduce another NP-hard problem B to A.

Having shortly introduced these terms, we turn to an overview of the subsections. In Section 3.1, we will discuss two algorithms that decide whether a given graph has cop number at most equal to k if k is not part of the input. In Section 3.2, we prove that this problem becomes NP-complete if k is allowed to be part of the input. Finally, in Section 3.3, we assess the situation for the classes of geometric graphs introduced in the previous chapter.

3.1 Two algorithms for k fixed

To be able to speak about the complexity of the Game of Cops and Robbers, we first need to model this game as a decision problem, having only a 'yes' or 'no' answer. An intuitive definition is the following.

k-COP NUMBER: Given as input some $k \in \mathbb{N}$ and a graph G, does $c(G) \leqslant k$ hold?

It is clear that an instance of this problem is given by a pair (k, G) of a natural number k and a graph G and that the outcome is indeed either 'yes' or 'no'. Unfortunately, we will see in the next section that this decision problem is hard: for $k \geqslant 2$ it belongs to the class of EXPTIME-complete problems. However, if we fix our natural number k beforehand and do not allow it to be part of the input, we can design somewhat efficient algorithms to answer the question posed. In this section, we will therefore turn our attention to the following decision problem.

k-FIXED COP NUMBER: For fixed $k \in \mathbb{N}$, given as input a graph G, does $c(G) \leqslant k$ hold?

We will prove that this question belongs to the class P of relatively easy problems. To do so, we will explicitly give an algorithm that solves our problem in polynomial time, which is a straightforward implementation of Theorem 2.6.5.

```
1: function SOLVE FIXED COP NUMBER
2:     ⩽_0 ← ∅
3:     for all w ∈ V(G^k_⊠), v ∈ V(G), i ∈ [k] do
4:         if (w)_i = v then
5:             ⩽_0 ← ⩽_0 ∪(v, w)
6:         end if
7:     end for
8:     i ← 0
9:     do
10:        i ← i + 1, ⩽_i ← ∅
11:        for all v ∈ V(G), w ∈ V(G^k_⊠) do
12:            if v ⩽_{i−1} w or ∃v' ∈ N_G[v], w' ∈ N_{G^k_⊠}[w], j < i : v' ⩽_j w' then
13:                ⩽_i ← ⩽_i ∪(v, w)
14:            end if
15:        end for
16:    while ⩽_i ≠ ⩽_{i−1}
17:    if ∃w ∈ V(G^k_⊠) ∀v ∈ V(G) : v ⩽_i w then
18:        return c(G) ⩽ k
19:    else
20:        return c(G) > k
21:    end if
22: end function
```

It follows directly from Theorem 2.6.5 that the algorithm is correct. Furthermore, as the $\leqslant_i$ form a non-decreasing sequence and $|\leqslant_i|$ is bounded from above by $|V(G) \times V(G^k_{\boxtimes})|$, the do-while loop in lines 9–16 is guaranteed to terminate after a finite number of iterations. Therefore, it only remains to determine the complexity of the algorithm.

Theorem 3.1.1. *The function* SOLVE FIXED COP NUMBER *runs in time complexity* $O(n^{3k+3}k)$*, where* $n = |V(G)|$ *is the amount of vertices in the input graph* G*.*

Proof. To speed up things, we do some preprocessing before diving into the algorithm itself. In particular, we compute $N_G[v]$ for all $v \in V(G)$, which may be done in $O(n^2)$ or less, depending on the way in which G is stored. Next, we compute $N_{G^k_{\boxtimes}}(u)$ for all $u \in V(G^k_{\boxtimes})$. For every of these n^k different u, we need to check for n^k vertices whether they are neighbors. This is done by checking n neighbor lists. Therefore, this computation can be done in time $O(n^{2k+1})$, which is therefore the runtime of the preprocessing phase.

Now, we analyse the algorithm itself. Lines 2 and lines 4–6 may be executed in constant time. Therefore, the for-loop in lines 3–7 has runtime of order $O(n^{k+1}k)$. Line 8 again takes constant time, as well as lines 10 and 13. If the relation $\leqslant_i$ is stored in a $n \times n^k$-matrix, for example, checking whether $v \leqslant_{i-1} w$ in line 12 can also be done in constant time. To check the second criterion in line 12, we need runtime $O(n^{k+1}k)$ in the worst case. The for loop in line 11 is executed n^{k+1} times, so lines 10–15 have a total runtime of $O(n^{2k+2}k)$. As mentioned above, $|\leqslant_i|$ increases by at least one every time the do-while loop in lines 9–16 is executed. As it is bounded by n^{k+1}, the loop has total complexity $O(n^{3k+3}k)$.

Finally, in line 17, we need to check at most n^{k+1} values, so lines 17–21 have runtime $O(n^{k+1})$. Therefore, the algorithm has a whole runs in time $O(n^{3k+3}k)$. □

We conclude that the problem k-FIXED COP NUMBER is indeed in P. However, the runtime of the given algoritm is none too good. Can we design a more efficient algorithm? The answer turns out to be yes: a better algorithm was published in [8]. It is based on

the following alternative classification result for k-cop-win graphs, slightly different from the one discussed in the previous chapter. Slightly abusing notation, for a graph G and some $u = (u_1, \dots, u_k) \in V(G^k_{\boxtimes})$, we will write $N_G[u] = \bigcup_{i=1}^{k} N_G[u_i]$.

Theorem 3.1.2 ([8]). *Let $k \in \mathbb{N}$ and let G be a graph. Then $c(G) > k$ if and only if there exists a mapping $f\colon V(G^k_{\boxtimes}) \to 2^{V(G)}$ having the following two properties.*

(1) *For every $u \in V(G^k_{\boxtimes})$,*
$$\varnothing \neq f(u) \subseteq V(G) \backslash N_G[u].$$

(2) *For every $uv \in E(G^k_{\boxtimes})$,*
$$f(u) \subseteq N_G[f(v)].$$

Proof. "⇒". Suppose that $c(G) > k$, i.e., that the robber has a winning strategy on G when playing against k cops. Define $f\colon V(G^k_{\boxtimes}) \to 2^{V(G)}$ by setting $f(u) \subseteq V(G)$ to be the set of vertices such that, if the cops start from u, the robber can start from any $r \in f(u)$ and win the game. We check that this f has the properties (1) and (2).

(1) Let $u \in V(G^k_{\boxtimes})$ be arbitrary. Since R has a winning strategy regardless of where the cops begin, $f(u) \neq \varnothing$ must hold, which proves the first part of (1). To prove the second part, let $r \in f(u)$ be arbitrary. If $r \in N_G[u]$, then C can capture the robber immediately or in the next round, so $r \notin f(u)$ by construction. Hence $r \notin N_G[u]$, so $r \in V(G) \backslash N_G[u]$.

(2) Let $uv \in E(G^k_{\boxtimes})$ be arbitrary, and let $z \in f(u)$. By the definition of f, R can win if he is in z and C is in u. Now, in the next round, C may move from u to v. As R still has a winning strategy, there must be some $z' \in V(G)$, perhaps equal to z, to which R can now move to avoid capture. By definition, $z' \in f(v)$. Hence either $z \in f(v)$ or $z' \in f(v)$ and therefore $z \in N_G[f(v)]$ holds, so we have proven that $f(u) \subseteq N_G[f(v)]$.

"⇐". Assume now that there exists some $f\colon V(G^k_{\boxtimes}) \to 2^{V(G)}$ which has properties (1) and (2). We will construct a winning strategy for the robber. Let $u^0 = (u^0_i)_{i=1}^k \in V(G^k_{\boxtimes})$ be the vector containing the initial positions of the k cops. In round 0, the robber starts the game on some vertex $r^0 \in f(u^0)$, which must exist as $f(u^0) \neq \varnothing$ by (1). Since also by (1) we have that $f(u^0) \subseteq V(G) \backslash N_G[u^0]$, we see that the cops cannot move to r^0 in one step, so the robber is safe from capture in round 1.

We proceed by induction: suppose that for all $t \leqslant m$, the robber can move to some $r^t \in f(u^t)$ so that he avoids capture in the next (t^{th}) round. Consider round $m+1$ in which $R = r^m \in f(u^m)$ and C moves from u^m to u^{m+1}. As this move is legal, $u^m u^{m+1} \in E(G^k_{\boxtimes})$ must hold. Therefore, by (2), we conclude that $f(u^m) \subseteq N_G[f(u^{m+1})]$. Hence, as $R = r^m \in f(u^m)$, by definition we have that $r^m \in N_G[f(u^{m+1})] = \bigcup_{v \in f(u^{m+1})} N_G[v]$. Therefore, we conclude that $r^m \in N_G[v]$ for some $v \in f(u^{m+1})$ so that the robber may move from r^m to this $v =: r^{m+1} \in f(u^{m+1})$ to avoid capture in the next round. □

The idea for the algorithm is to try and construct a mapping $f\colon V(G^k_{\boxtimes}) \to 2^{V(G)}$ that has properties (1) and (2) or to conclude that such an f cannot exist. This can be done in the following manner.

```
1: function SOLVE FIXED COP NUMBER II
2:     for all u ∈ V(G^k_⊠) do
3:         f(u) ← V(G)\N_G[u]
4:     end for
```

```
5:    Initialise queue: Q ← V(G^k_⊠)
6:    while Q ≠ ∅ do
7:        Pop u from Q
8:        for all u′ ∈ N_{G^k_⊠}[u] do
9:            f(u′) ← f(u′) ∩ N_G[f(u)]
10:           if f(u′) was changed then
11:               Push u′ to Q
12:           end if
13:       end for
14:   end while
15:   if ∃u ∈ V(G^k_⊠) : f(u) = ∅ then
16:       return c(G) ⩽ k
17:   else
18:       return c(G) > k
19:   end if
20: end function
```

Here, to pop an element from Q means to remove and consider the first element of the queue. Similarly, pushing some u to Q adds this element to the end of the queue. We check that the algorithm SOLVE FIXED COP NUMBER II is correct and indeed runs in polynomial time.

Theorem 3.1.3 ([8]). *For an input graph G, the function* SOLVE FIXED COP NUMBER II *correctly determines whether $c(G) \leqslant k$ in finite time.*

Proof. By Theorem 3.1.2, we only need to prove that our algorithm succesfully checks whether there exists a map $f\colon V(G^k_{\boxtimes}) \to 2^{V(G)}$ having properties (1) and (2) in a finite number of iterations. By property (1), we have $f(u) \subseteq V(G) \setminus N_G[u]$ for all $u \in V(G^k_{\boxtimes})$, so $f(u)$ will contain precisely all elements that can possibly be part of it after the initialisation in lines 2–4.

Now, we remove an element of $f(u)$ only if we find out that this is necessary to do so in order for f to satisfy property (2). By induction, when considering an element $u \in V(G^k_{\boxtimes})$, we may assume that $f(u)$ contains the most elements possible. Now, by property (2), we have that $f(u') \subseteq N_G[f(u)]$ must hold for all u' adjacent to u in $G^k_{\boxtimes}$. Therefore, by intersecting $f(u')$ with $N_G[f(u)]$ for all neighbors u' of u, we remove nothing more than necessary.

Now, if $f(u')$ was decreased in size by this operation, the value of f for neighbors of u' might need to be changed as well for property (2) to hold, so we add u' back to the queue to check it again. Now, once we reach line 14, we have checked that $f(u') = f(u') \cap N_G[f(u)]$ for all $u \in V(G^k_{\boxtimes})$ and $u' \in N_G[f(u)]$, so indeed $f(u') \subseteq N_G[f(u)]$ holds for all $u'u \in E(G^k_{\boxtimes})$ and property (2) is satisfied. As the inclusion of property (1) was satisfied by the initialisation, we only need to check whether $f(u) = \varnothing$ for any $u \in V(G^k_{\boxtimes})$, which is done in lines 15–19.

Remains to prove that the algorithm always terminates. After initialisation, Q contains a finite number of elements, as does $f(u)$ for all $u \in V(G^k_{\boxtimes})$. Now, note that whenever an element is added to Q in line 11, we have decreased the size of the set $f(u')$ for some $u' \in V(G^k_{\boxtimes})$ by at least one. Therefore, line 11 can only be executed a finite number of times. Hence, as we remove an element of Q in every iteration of line 7, the queue must become empty after a finite number of iterations, after which the algorithm terminates. □

Theorem 3.1.4 ([8]). *The function* SOLVE FIXED COP NUMBER II *has a running time of $O(n^{2k+3})$, where $n = |V(G)|$ is the amount of vertices in the input graph G.*

Proof. We again do the same preprocessing as for the previous algorithm SOLVE FIXED COP NUMBER, which has runtime $O(n^{2k+1})$. As for the algorithm itself, line 3 boils down to taking the intersection of two sets of size at most n and is executed n^k times for a total of time $O(n^{k+1})$. Likewise, lines 5 and 15 run in $O(n^k)$. Remains to determine the runtime of the while loop in lines 6-14.

First, consider the runtime of a single execution of the loop. Lines 7 and 11 are executed in constant time. To compute the neighborhood in line 9, we take at most n unions over sets of size at most n, resulting in a runtime of $O(n^2)$. We then take the intersection of two sets of size at most n in time $O(n)$. The for loop in line 8 is executed at most n^k times, so lines 8–13 have runtime $O(n^{k+2})$. Remains to determine how often the while loop is executed. In line 7, we always remove an element from Q. In lines 9–11, if we add an element to Q, we have also removed at least one element from $f(u')$ for some $u' \in V(G^k_{\boxtimes})$. Therefore, in every iteration of the loop, the size of $Q \cup \bigcup_{u \in V(G^k_{\boxtimes})} f(u)$ decreases by at least one. Hence, the loop is executed at most $|Q \cup \bigcup_{u \in V(G^k_{\boxtimes})} f(u)| = n^k + n^k \cdot n = O(n^{k+1})$ times.

We conclude that lines 6–14 have a total runtime of $O(n^{k+2} \cdot n^{k+1}) = O(n^{2k+3})$. As this is the bottleneck of the algorithm, this finishes the proof. □

Hence, this second algorithm has a slightly better runtime than the first. However, the exponent of $2k+3$ is still quite horrendous. Indeed, for a graph G with only 100 vertices, to determine whether $c(G) \leqslant 3$ takes the order of 10^{16} operations, which would take years to complete on the average computer. Hence, although the results in this section show that our problem is solvable in polynomial time, in practice, the algorithms are not useful for computations.

This concludes the treatment of k fixed, but what if k is allowed to be part of the input? In particular, note that in this case it may e.g. be a function of the amount of nodes n in the graph.

3.2 Complexity for k part of input

Now we allow the integer k to be part of the input. Hence, we will consider the problem k-COP NUMBER as defined in the previous section. Whereas the problem k-FIXED COP NUMBER was solvable in polynomial time, we will see that this problem is severely more difficult. We will prove that k-COP NUMBER is NP-hard by extensively using the following lemma, the proof of which may be found in [17].

Lemma 3.2.1 ([17])**.** *Let $m, n, r \in \mathbb{N}$. If*

$$m \geqslant 2n(r+1)\frac{(n(r+1)-1)^6 - 1}{(n(r+1)-1)^2 - 1},$$

then there exists a graph $H = H(m, n, r)$ such that

(i) H is bipartite with vertex sets X and Y such that $|X| = |Y| = nm$.

(ii) We may write $X = U_1 \sqcup U_2 \sqcup \ldots \sqcup U_n$ and $Y = W_1 \sqcup W_2 \sqcup \ldots \sqcup W_n$ such that $|U_i| = |W_i| = m$. We employ the following notation: H_{ij} is the subgraph of H induced by $U_i \cup W_j$ and $d_{ij}(u)$ is the degree of a vertex $u \in U_i \cup W_j$ in H_{ij}.

(iii) Given sets X and Y with $|X| = |Y| = nm$, the graph H can be algorithmically constructed in time $O(rmn^2)$.

(iv) For all $i, j \in [n]$, for all $u \in V(H_{ij})$ we have $r - 1 \leqslant d_{ij}(u) \leqslant r + 1$.

(v) For all $u \in V(H)$ we have $d(u) \leqslant n(r + 1)$.

(vi) The graph H has girth at least 6.

To prove NP-hardness, we need a reduction from another problem, which we introduce after two definitions.

Definition 3.2.2. Let $G = (V, E)$ be a graph. A set $W \subseteq V$ is called *dominating* if $\bigcup_{w \in W} N[w] = V$.

Definition 3.2.3. Let $G = (V, E)$ be a graph. The *domination number* of G, denoted by $\gamma(G)$, is defined by

$$\gamma(G) = \min\{|W| : W \text{ is dominating}\}.$$

The following question is now a natural one to ask.

DOMINATION: Given $k \in \mathbb{N}$, $k \geqslant 2$ and a graph G, is it true that $\gamma(G) \leqslant k$?

Note that DOMINATION is NP-hard, as proven in [20]. We now have all ingredients for our proof.

Theorem 3.2.4 ([17])**.** *The problem k-COP NUMBER is NP-hard for $k \geqslant 2$.*

Proof. Let $G = (V, E)$ be a graph with vertices labeled $v_1, \ldots, v_n$ and let $k \geqslant 2$. We will construct a graph G' such that $\gamma(G) \leqslant k$ if and only if $c(G') \leqslant k$. First, set $r = k + 2$ and

$$m = \left\lceil 2n(r + 1)\frac{(n(r + 1) - 1)^6 - 1}{(n(r + 1) - 1)^2 - 1} \right\rceil.$$

For every vertex v_i of G, we now add two sets of m vertices each, which we call U_i and W_i. Each of these new vertices is connected with an edge to every vertex in $N[v_i]$. See also Figure 3.2.1. We now set $X = \bigcup_{i=1}^{n} U_i$ and $Y = \bigcup_{i=1}^{n} W_i$. By Lemma 3.2.1.iii, we may algorithmically construct a graph $H = H(m, n, r)$ on the vertex set $X \cup Y$ with the properties as described in the lemma. We call the total resulting graph on the vertices $V(G) \cup X \cup Y$ our candidate G'.

Our claim is that $\gamma(G) \leqslant k$ if and only if $c(G') \leqslant k$. First, suppose $\gamma(G) \leqslant k$ and let $Z \subseteq V(G)$ with $|Z| \leqslant k$ be a dominating set. In round 0, place one cop in every vertex of Z. By the definition of a dominating set, the robber cannot be placed on a vertex in G in round 0, because every such vertex is occupied by or adjacent to a cop and therefore the robber would be caught in round 1. Furthermore, by the definition of the sets U_i and W_i, the robber can also not be placed in one of these sets, as every vertex in these sets is also connected to a cop and therefore capture would again be imminent in the first round. Therefore, the cops win and indeed $c(G') \leqslant k$.

Now, assume that $\gamma(G) > k$. We will show that the robber can always evade k cops in G'. First, let $C \subseteq V(G')$ be an arbitrary set of vertices occupied by the k cops in round 0. We claim that there must be some vertex $v_i \in V(G)$ which is not occupied by or connected to a cop. Suppose not, then define

$$Z = \{v_i \in V(G) : v_i \in C \text{ or } C \cap (U_i \cup W_i) \neq \varnothing\},$$

i.e., Z contains all vertices v_i for which either v_i itself is occupied by a cop or a cop is present in $U_i \cup W_i$. Noting that every vertex in $U_i \cup W_i$ is adjacent to every vertex in $N[v_i]$ by definition, we find that $\bigcup_{c \in C} N[c] \cap V(G) = \bigcup_{z \in Z} N[z] \cap V(G)$. Therefore, Z

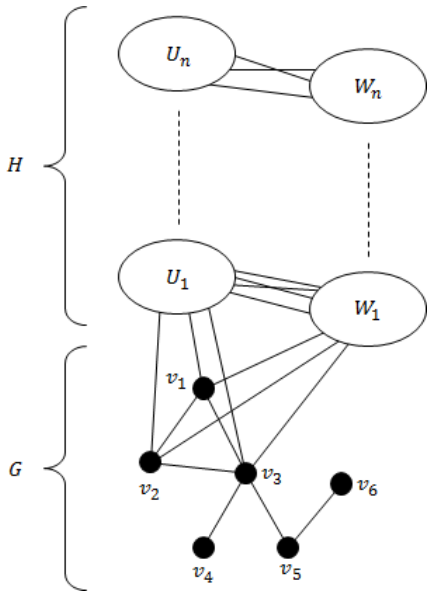

Figure 3.2.1: An overview of the graph G'.

is dominating, which provides a contradiction as $|Z| \leqslant k$ by construction. Hence indeed a v_i exists as claimed, which we will from now on call a 'safe' vertex.

Again, by definition of the sets U_i, we must conclude that no cop in G is adjacent to any vertex in U_i; if so, she would also be adjacent to v_i which is not true. As H is bipartite by construction, no cops in X are adjacent to any vertex in U_i. Finally, by Lemma 3.2.1.v, every cop in Y can be connected to at most $n(r+1)$ vertices in U_i, so in total there are at most $kn(r+1)$ vertices in U_i which are adjacent to a cop.

Now, one may check by a straightforward computation that

$$m = \left\lceil 2n(r+1)\frac{(n(r+1)-1)^6 - 1}{(n(r+1)-1)^2 - 1} \right\rceil > kn(r+1).$$

Noting that $|U_i| = m$ by construction, we therefore conclude that there must be a 'safe' vertex $u \in U_i$ which is not connected to a cop. We put the robber on this vertex u in round 0 so that he survives at least the first move of the cops.

We now proceed by induction to the number of rounds t, for which the base case $t = 0$ has just been checked. Suppose as induction hypothesis that the robber is positioned on a safe vertex $u \in X \cup Y$ at time t, after which the cops make a move. Assume without loss of generality (by symmetry) that $u \in X$.

If no cop is adjacent to u after the move, at the start of round $t+1$, the robber simply passes and survives for another turn. Otherwise, by the reasoning above, there must exist some safe vertex $v_j \in V(G)$. Now, consider the set W_j. By Lemma 3.2.1.iv, the vertex u has at least $r-1$ neighbors in W_j. Note that $r = k+2$, so u is connected to at least $k+1$ vertices in W_j. Furthermore, by Lemma 3.2.1.vi, the girth of H is at least 6, so any cop positioned in X can only be adjacent to at most one neighbor of u in W_j, as otherwise there would be a cycle of length 4. Finally, as v_j is safe by assumption, no cop in G is connected to W_j by construction. Therefore, there must exist a safe vertex $w \in W_j$. We let the robber move to this vertex in round $t+1$, which completes the proof. □

For years, it has been unknown whether the problem is in fact contained in the class NP (i.e., whether it is NP-complete). Recently, it has been shown that this is probably not the case, as our problem is in fact not only NP-hard, but even EXPTIME-complete. We omit the laborious proof which is given in [27].

Theorem 3.2.5 ([27])**.** *The problem k-*COP NUMBER *is EXPTIME-complete for $k \geqslant 2$.*

This settles the computational issues for general graphs, but what if we restrict ourselves to some smaller graph class?

3.3 Computational problems on geometric graphs

It turns out that if we restrict ourselves to some class of geometric graphs such as unit disk graphs or string graphs, the problem of determining whether $c(G) \leqslant k$ becomes much easier in terms of complexity. To make this statement precise, we define three more decision problems.

k-PLANAR COP NUMBER: Given as input some $k \in \mathbb{N}$ and a planar graph G, does $c(G) \leqslant k$ hold?

k-UDG COP NUMBER: Given as input some $k \in \mathbb{N}$ and a unit disk graph G, does $c(G) \leqslant k$ hold?

k-STRING COP NUMBER: Given as input some $k \in \mathbb{N}$ and a string graph G, does $c(G) \leqslant k$ hold?

The complexity of these problems is assesed in the following theorem. However, for the proof of this theorem, we need some more background on these graph classes, so we defer its treatment to Section 4.3.

Theorem 3.3.1. *The decision problems k-*PLANAR COP NUMBER*, k-*UDG COP NUMBER *and k-*STRING COP NUMBER *belong to P.*

Can the above result be put into practice if we are faced with some graph of which it is not known beforehand that it belongs to one of our graph classes? This is partly the case. It has been long known that recognizing planar graphs is easy; in fact, there are multiple algorithms that determine whether a graph is planar in linear time, one of which can be found in [23]. Therefore, when given some input graph G, it may be efficient to first try and see if it is planar, knowing that the problem of determining its cop number can then be solved in polynomial time.

For unit disk graphs and string graphs, the results are less favorable. For both graph classes, to determine whether a given graph belongs to the class is NP-complete [26, 29, 37]. Therefore, the statements regarding these classes in Theorem 3.3.1 cannot be easily put into practice if it is not known a priori that the given graph belongs to the class.

4 Upper bounds

So far, the graphs we have encountered have quite a low cop number. This might make us wonder whether graphs with high cop number actually exist. If we impose no restrictions on the graphs considered, it turns out that we can indeed construct graphs with a higher cop number. In fact, we will see that graphs can have an arbitrarily high cop number, but not until we reach Section 5.2.

In this chapter, we wonder whether this is also possible if we restrict our search to a class of geometric graphs. The answer turns out to be no: for some restricted classes of graphs, it is possible to determine a finite upper bound on the cop number of graphs in that class.

Hence, the question central to this section is the following: given a graph G beloning to some graph class, can we show that $c(G) \leqslant k$ must always hold for some fixed $k \in \mathbb{N}$? In the Sections 4.1, 4.2 and 4.3 we will show that the cop number of planar graphs, unit disk graphs and string graphs respectively are indeed always bounded from above by some constant k. In Section 4.4, we will consider one other upper bound on the cop number of a graph linked to the treewidth, a parameter which measures in some sense how "tree-like" a graph is.

Throughout the discussion of these theorems, we will see that while the details are different, every proof is structured in a very similar way. We start out by showing that some sort of structure in the graph — a path, for example — may be guarded by a limited number of cops so that the robber can no longer cross this guarded path of the graph. Next, by carefully picking the locations in the graph to guard, we enclose the robber in a secluded part of the graph, which we will call the "robber territory", just like in Section 2.4. By slowly moving the cops inwards, we then reduce the robber territory step by step until the robber has nowhere left to go.

4.1 Planar graphs

The main result of this section is the perhaps surprising fact that every planar graph has cop number at most 3. This statement was already proven in one of the first articles published in the field of pursuit-evasion games [1]. As foretold in the introduction of this chapter, the proof starts by showing that some structure in the graph may be guarded by a limited number of cops. We first formalise what we mean by guarding.

Definition 4.1.1. Let G be a graph and let $H = (V, E)$ with $V \subseteq V(G)$ and $E = \binom{V}{2} \cap E(G)$ be an induced subgraph. We say that H is *k-guardable* if k cops have a strategy so that after some finite number of moves, the cops can always remain in H for the remainder of the game in such a way that if the robber moves into H, he can be caught immediately or in the next round.

This definition will also be used in the subsequent sections. Now, the structure that we consider in this section is that of a shortest path, which is defined as follows.

Definition 4.1.2. Let G be a graph and let $P = (v_0, \ldots, v_k)$ be a path of $k+1$ vertices and k edges starting at $v_0 \in V(G)$ and ending at $v_k \in V(G)$. We say P is a *shortest path* connecting v_0 and v_k in G if every other path starting at v_0 and ending at v_k has length at least k. In this case, we say the *distance* between v_0 and v_k in G is k. Notation: $d_G(v_0, v_k) = k$.

The first step towards proving the upper bound for planar graphs is now the following.

Proposition 4.1.3 ([1]). *Let G be a graph and let $v, w \in V(G)$. Let P be a shortest path connecting v and w in G. Then P is 1-guardable.*

Proof. Suppose P has length k for some $k \in \mathbb{N}$ and write $P = (v_0, \dots, v_k)$. For notational convenience, we set $v_{-1} := v_0$ and $v_{k+1} := v_k$ wherever they occur. Define the map $f\colon V(G) \to \{v_0, \dots, v_k\}$ by $f(v) = v_{d(v,v_0)}$ if $d(v, v_0) \leqslant k$ and $f(v) = v_k$ otherwise. Hence, f maps all vertices in G to the path P in such a way that the distance to v_0 remains the same (or becomes k if it was larger than k). The inverse image $f^{-1}[\{v_i\}]$ therefore represents all vertices in G having distance i to v_0 for $i = 0, \dots, k-1$, and $f^{-1}[\{v_k\}]$ contains the vertices having distance at least k.

Now, let the cop start on vertex v_0 and suppose the robber starts on $v \in f^{-1}[\{v_i\}]$ for some $i = 0, \dots, k$. Note that whenever the robber moves from such a vertex $v \in f^{-1}[\{v_i\}]$, he must always end up on some $w \in f^{-1}[\{v_{i-1}\}] \cup f^{-1}[\{v_i\}] \cup f^{-1}[\{v_{i+1}\}]$ by the definition of distance. Therefore, the image of the robber $f(v) = v_i$ moves to a vertex in $\{v_{i-1}, v_i, v_{i+1}\}$. Thus $f(v)$ represents the movement of the robber restricted to P, which may be caught by C in finite time as the cop number of a path is 1 (a path is a tree).

Hence, $C = f(R)$ will hold after a finite number of rounds. Now, whenever the robber makes a move, the cop can move in such a way that $C = f(R)$ will again hold after her move. Therefore, if the robber moves onto P at some point during the game, we have by the definition of f that $f(R) = R$, so after one move of the cop $C = f(R) = R$ holds and the robber is caught. □

Note that it is crucially important that the path in the previous proposition is a shortest one. If we consider a non-shortest path in a graph, the statement does not necessarily hold: consider G being any cycle of size $n \geqslant 4$. We may view all vertices in the cycle as belonging to a single path $(v_1, \dots, v_n)$, where v_1 and v_n are adjacent. It is clear that this path is not a shortest path in G, as it has length n whereas the path (v_1, v_n) has length $1 < n$. Furthermore, we know by the result of Example 2.4.2 that our path is not 1-guardable, as $c(G) = 2 > 1$.

However, we can apply the result of Proposition 4.1.3 by splitting G into two paths $(v_1, \dots, v_{\lfloor n/2 \rfloor})$ and $(v_{\lfloor n/2 \rfloor + 1}, \dots, v_n)$. These two paths are both shortest paths in the cycle, so they can be guarded by 1 cop each. As the paths together contain all vertices in G, this again shows that $c(G) \leqslant 2$. We summarize this idea in the following corollary.

Corollary 4.1.4. *Let G be a graph. Then $c(G)$ is at most equal to the minimum number of shortest paths that together contain all vertices of G.*

Note that this bound is not tight. Indeed, consider the complete graph K_n for arbitrary $n \in \mathbb{N}$. In Example 2.4.3, we have seen that $c(K_n) = 1$. However, the only shortest paths in K_n are the paths consisting of a single edge between two vertices. Therefore, to cover all vertices in K_n by shortest paths, we would need $\lceil \frac{n}{2} \rceil$ paths.

Back to the proof of the main theorem of this section. The final trick which we need for this proof is so-called *circular inversion*. So far, we have always drawn graphs in the plane $\mathbb{R}^2$. However, we may just as well make a drawing of a graph on another surface, such as a sphere or torus. In the formal Definition 2.2.1 of a drawing, we then simply replace the codomain of ϕ and the γ_i by the surface of interest. The following fact applies.

Lemma 4.1.5. *Let G be a graph and let $S^2 = \{x \in \mathbb{R}^3 : \|x\|_2 = 1\}$ be the unit sphere. Then G is planar if and only if there is some drawing of G on S^2 without crossings.*

Proof. Note that a disk of any radius is homeomorphic to the unit sphere with a single point removed, which is easily shown using polar coordinates. Therefore, a drawing of G with no crossings in the plane can be continuously morphed into a drawing on S^2. Vice versa, for any drawing of G on the S^2, there must be one point on S^2 which is not contained in the image of the drawing. We may thus remove this point from S^2 without problems, so that the remaining surface can be continuously mapped onto e.g. the unit disk in $\mathbb{R}^2$. □

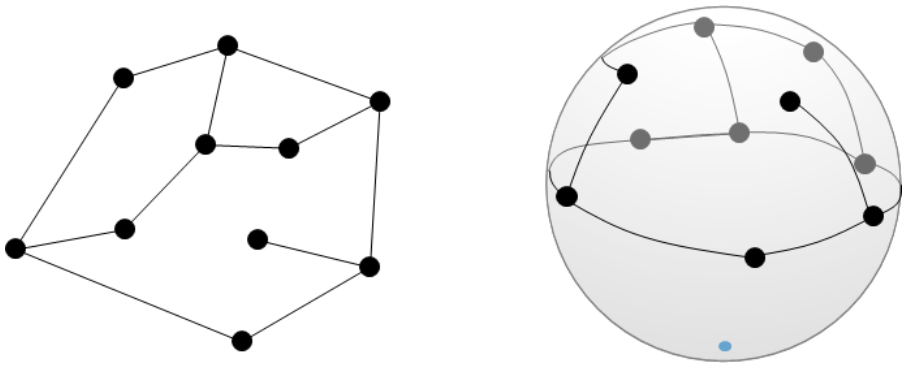

Figure 4.1.1: A drawing of some graph G in the plane and on S^2 with the blue point removed.

Figure 4.1.1 illustrates the working of the lemma. Now, note that by removing a different point from S^2 when morphing the drawing on the sphere back to a drawing in the plane, we might end up with a vastly different picture as the picture with which we started, as shown by Figure 4.1.2. As the graph is in some way "turned inside-out" by this operation (in this example picture, the 6-cycle which encompasses the graph in the first drawing is pushed inside after the transformation, and vice versa for the 4-cycle), this explains the name of circular inversion.

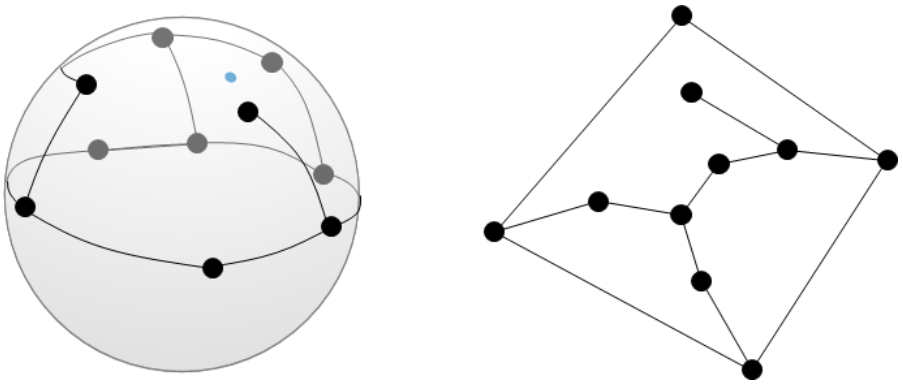

Figure 4.1.2: Morphing to a different drawing in the plane, removing the blue point in the interior of the 4-cycle from S^2.

We now proceed to the core of the proof of the main result of this section. The following notation will be useful: for a path $P = (v_1, \dots, v_n)$, we let $V(P) = \{v_1, \dots, v_n\}$ be the set of vertices of which P consists. Furthermore, for $i, j \in [n]$ with $i \leqslant j$ we write $P(v_i, v_j) = (v_i, v_{i+1}, \dots, v_{j-1}, v_j)$ for the part of the path P going from vertex v_i to v_j. Finally, we write $P_1 \cup P_2$ for the concatenation of two paths P_1 and P_2.

Theorem 4.1.6 ([1]). *For any planar graph G, it holds that $c(G) \leqslant 3$.*

Proof. Fix a drawing of G and consider three cops playing on the graph. We recall: the core idea of the proof is to determine at the start of each round i the "robber territory" $R_i \subseteq V(G)$ as the set of vertices in which the robber can move freely without being caught. We will show that for any $i \in \mathbb{N}_0$, there is some finite amount of moves $j = j(i) \in \mathbb{N}$ for which $R_{i+j} \subsetneq R_i$. By induction, it then follows that the robber territory R_i can be reduced to $\varnothing$ in finite time, so that the robber is caught by the cops.

In the proof, we will distinguish between two situations (a) and (b). We will show that we can start the game in situation (a) and, inductively, that if we are in either situation (a) or (b), we will always again end up in one of the two situations after a finite amount of moves, having reduced the size of the robber territory along the way. First, we will give a description of the cases.

Situation (a). Some cop is on a vertex $v \in V(G)$, such that removing v and all incident edges from G makes it fall apart into connected components $G_1, \ldots, G_m$, $m \in \mathbb{N}$, where $m = 1$ means that actually the graph does not really fall apart at all. We set $R_i = V(G_j)$ with G_j the component that R is in. See Figure 4.1.3.

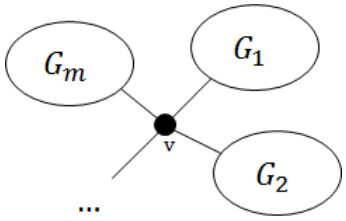

Figure 4.1.3: A sketch of situation (a).

Situation (b). Let P_1 and P_2 be two paths connecting some vertices $v, w \in V(G)$, which are disjoint except for v and w. As G is planar, we can speak of G being divided into $V(P_1 \cup P_2)$, an "interior region" consisting of all vertices lying within the closed curve formed by $P_1 \cup P_2$ and an "exterior region" containing all vertices lying outside the curve, which we call W. See Figure 4.1.4 for a sketch. By Lemma 4.1.5, we may suppose w.l.o.g. that R is in this exterior region. Furthermore, P_1 should be a shortest path between v and w in $P_1 \cup P_2 \cup W$ and P_2 should be shortest among all paths in $P_1 \cup P_2 \cup W$ between v and w that are disjoint from P_1 except for v and w. A cop C_1 guards P_1 and a cop C_2 guards P_2 in the sense of Proposition 4.1.3. We set $R_i = W$.

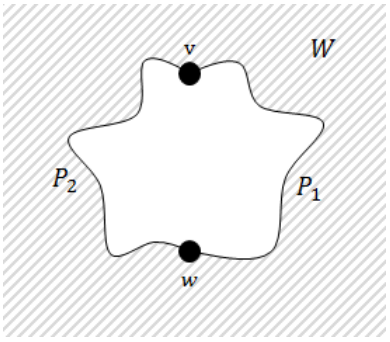

Figure 4.1.4: A sketch of situation (b).

At the start of the game, in round 0, the cops pick some single vertex $C \in V(G)$ on which they all start. Consequently, the robber picks some starting vertex $R \in V(G)$. We are now in situation (a) with R_0 the component of G with C removed that R is in.

Suppose we are in situation (a) at the start of round $i \in \mathbb{N}_0$. Assume first that $|N_G(v) \cap R_i| = 1$, i.e., that v has only one neighbor in the robber terrirory R_i, w say. If $R = w$, then the robber is caught after one move of the cop. Otherwise, by moving to w, the cop can ensure that $R_{i+1} \subsetneq R_i$, no matter what the robber does; he cannot pass the cop on w while moving to v as this would require a second neighbor of v in R_i. We are then back in situation (a), having reduced the size of the robber territory.

Next, assume that there are at least two neighbors, w_1 and w_2, say, of v in R_i. Let P be a shortest path connecting w_1 and w_2 in R_i. After a finite amount of moves j during which the cop on v remains stationary (so the robber territory does not increase), one of the other two "free" cops can guard P as in Proposition 4.1.3. We are then in situation (b) with $P_1 = P$ and $P_2 = \{w_1, v, w_2\}$ or vice versa if $w_1w_2 \in E(G)$. Furthermore, $R_{i+j} \subseteq R_i \backslash V(P) \subsetneq R_i$.

Suppose now we are in situation (b) at the start of round $i \in \mathbb{N}$. Assume first that there are no paths in $R_i \cup P_1 \cup P_2$ connecting v and w except for P_1 and P_2. Then if $P_1 \cup P_2$ is removed, R_i must fall apart into connected components $H_1, \dots, H_m$, $m \in \mathbb{N}$ which are all connected to $P_1 \cup P_2$ via a single edge in R_i, see also Figure 4.1.5. Say that $R \in H_k$, $k \in [m]$, and that H_k is connected to $u \in V(P_1 \cup P_2)$. After a finite number of moves j, the free third cop can position herself on u so that we are back in situation (a) with $R_{i+j} = H_k \subseteq R_i$. Note that this is not always a strict inclusion as $k = m = 1$ might hold, but that being back in situation (a) will always result in the robber territory being reduced as shown above.

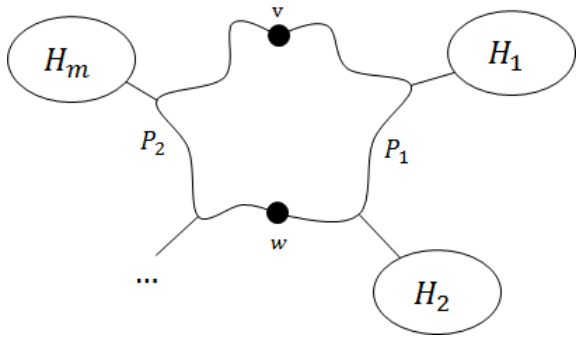

Figure 4.1.5: Situation (b) with no other paths than P_1 and P_2.

Assume next that there are other paths in $P_1 \cup P_2 \cup W$ than P_1 and P_2 that connect v and w and let $P = (v, p_1, \dots, p_r, w)$, $r \in \mathbb{N}$ be shortest among them. Note that P may be longer than P_1 and/or P_2. Write $P_1 = (v, p_1^1, \dots, p_k^1, w)$ and $P_2 = (v, p_1^2, \dots, p_\ell^2, w)$ with $k \in \mathbb{N}_0$, $\ell \in \mathbb{N}$ and first assume that $p_1 \neq p_1^2$. Let $v = p_0 = p_0^1$ and $N = \max\{a \in [k] \mid p_a = p_a^1\}$ and set $u = p_N$; in words: u is the last common vertex of P and P_1 when walking from v. Analogously, set $M = \min\{a > N \mid p_a \in V(P_1 \cup P_2)\}$ and let $x = p_M$ be the first vertex after u on which P agrees with either P_1 or P_2, setting $x = w$ if M is not well-defined. See Figure 4.1.6 for the different situations that can occur.

We look at what happens in the different cases. First suppose that $x \in V(P_1)$, so we are in case (i) or (ii); start with case (i). Suppose the robber is on a vertex lying in the outer area labeled A in the figure. By assumption, $P_1(u, x)$ is a shortest path connecting u and x in $V(P_1) \cup V(P_2) \cup W \supseteq V(P_1(u, x)) \cup V(P(u, x)) \cup A$. Furthermore, $P(u, x)$ is a

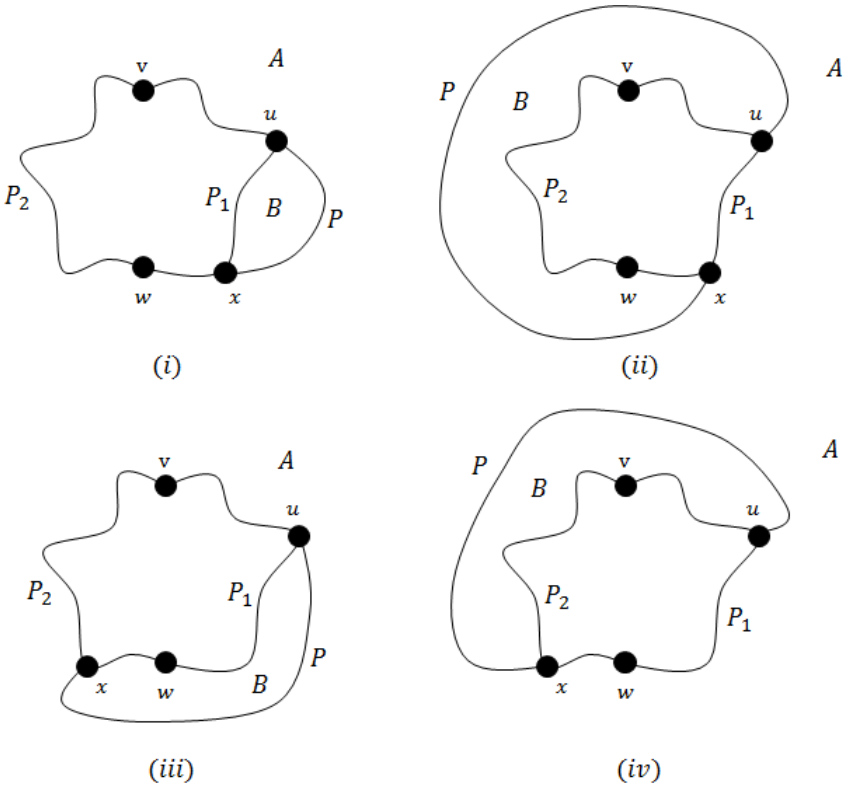

Figure 4.1.6: The different situations if $p_1 \neq p_1^2$ or $m = 0$.

shortest path connecting u and x in $V(P_1) \cup V(P_2) \cup W \supseteq V(P_1(u,x)) \cup V(P(u,x)) \cup A$ amongst all paths that are disjoint from $P_1(u,x)$. As we started in situation (b), the path P_1 is guarded by a cop, so also $P_1(u,x)$ is. Note that this cop cannot move to guard P herself, as P might not be a shortest path in the whole vertex set V, and the robber might escape R_i during the move of the cop. However, by Proposition 4.1.3, $P(v,w)$ can be guarded by the third remaining free cop in a finite number of moves j. Therefore, after these j moves, we are back in situation (b) with $R_{i+j} = A \subsetneq R_i$.

If the robber is in B, the argument is similar. We again consider $P = P_1(v,u) \cup P(u,x) \cup P_1(x,w)$ and consider the set $U := V(P_2(v,w)) \cup V(P_3(v,w)) \cup W$. If P_2 is a shortest path connecting v and w in W, then P_3 is by definition a shortest path connecting v and w in U disjoint from P_2. The path P_2 is already guarded by a cop and the free cop can move to guard P_3 in a finite number of moves j, after which we are back in situation (b) with $R_{i+j} = B \subsetneq R_i$. Conversely, if P_3 is a shortest path connecting v and w in U, then P_2 is by assumption a shortest path connecting v and w in U disjoint from P_3. Therefore, by the guarding argument before, we can again arrive in situation (b) with $R_{i+j} = B \subsetneq R_i$.

For case (ii), we reason in the same way. If the robber is in A, then $P_1(u,x)$ is a shortest path connecting u and x and $P_3(u,x)$ is a shortest path connecting u and x amongst all paths disjoint from P_1, so after a finite amount of time both paths are guarded by a cop and the robber territory is decreased to A. If the robber is in B, then either $P_3 := P_1(v,u) \cup P(u,x) \cup P_1(x,w)$ or $P_2(v,w)$ is a shortest path in $V(P_2) \cup V(P_3) \cup B$ and P_2 is a shortest path amongst those disjoint from P_3 or vice versa, so the same argument applies.

As all cases from here are similar, we quickly cover them. Suppose now $x \in V(P_2)$. In case (iii), if the robber is in A, then $P_2(v,x)$ is a shortest path connecting v and x bordering A and $P_1(v,u) \cup P(u,x)$ is a shortest path disjoint from $P_2(v,x)$, so the robber territory becomes A. If the robber is in B, then $P_1(u,w)$ is a shortest path and $P(u,x) \cup P_2(x,w)$ is a shortest disjoint path, so the robber territory becomes B. In case (iv), if R is in A, then $P_1(u,w)$ is shortest and $P(u,x) \cup P_2(x,w)$ is shortest disjoint. If R is in B, then either $P_2(v,x)$ or $P_1(v,u) \cup P(u,x)$ is shortest and the other is shortest amongst all paths disjoint from the one. Wherever the robber is, his territory is thus decreased in finite time.

Finally, we consider the case where $p_1 \neq p_1^1$. For a sketch, one can look at Figure 4.1.6 but with the labels of P_1 and P_2 swapped. In case (i), if the robber is in A, then $P_1(v,w)$ is a shortest path and $P_2(v,u) \cup P(u,x) \cup P_2(x,w)$ is shortest amongst all disjoint paths. If R is in B, then $P_2(u,x)$ is shortest and $P(u,x)$ is shortest disjoint. In case (ii), if R is in A, then $P_2(u,x)$ is shortest and $P(u,x)$ is shortest disjoint. If R is in B, then $P_1(v,w)$ is shortest and $P_2(v,u) \cup P(u,x) \cup P_1(x,w)$ is shortest amongst all disjoint paths. In case (iii), if R is in A, then $P_1(v,x)$ is shortest and $P_2(v,u) \cup P(u,x)$ is shortest disjoint. If R is in B, then either $P_2(u,w)$ is shortest and $P(u,x) \cup P_1(x,w)$ is shortest disjoint or vice versa. In case (iv), if R is in A, then $P_1(u,w)$ is shortest and $P(u,x) \cup P_2(x,w)$ is shortest disjoint. If R is in B, then either $P_1(v,u) \cup P(u,x)$ is shortest and $P_2(v,x)$ is shortest disjoint or vice versa. In all cases, using a guardability argument relying on Proposition 4.1.3, it follows that the robber territory decreases after a finite time, at which we are back in situation (b). This completes the proof. □

Though some case analysis was required, this proof is a clear example of how a strategy of guarding locations in the graph and gradually reducing the movement space of the robber works out. In the next sections, we will see that this idea forms the key part of several other proofs.

4.2 Unit disk graphs

We proceed to prove a generalisation of Theorem 4.1.6 for unit disk graphs, which was published in [7]. Before we start, we make one useful observation. Let G be a unit disk graph with $|V(G)| = n$ and let $\mathcal{C} = \{C_1, \dots, C_n\}$ be its unit disk representation as per Definition 2.2.5, with centers $c_1, \dots, c_n$. We can define a drawing of G by setting $\phi(v_i) = c_i$ for $i \in [n]$ and $\gamma_i(t) = tc_j + (1-t)c_k$ for $e_i = v_j v_k \in E(G)$. In words: G is drawn in the plane by taking the centers of the unit disks as vertices and connecting two centers by a straight line segment if the vertices are connected in G. We will refer to this drawing as the *canonical* drawing of G. Equipped with this information, we show that paths in a unit disk graph can still be guarded in some way, but more cops may be necessary.

Definition 4.2.1. Let G be a graph and let $H = (V, E)$ with $V \subseteq V(G)$ and $E = \binom{V}{2} \cap E(G)$ be an induced subgraph. Fix a drawing of G. We say that H is *k-patrollable* if the cops have a strategy so that after some finite number of moves, the cops guard H as in Definition 4.1.1 and, moreover, the robber cannot traverse an edge crossing some edge in $E(H)$ in the drawing of G without being caught in the next round at the latest.

Proposition 4.2.2 ([7])**.** *Let G be a unit disk graph and let $v, w \in V(G)$. Let P be a shortest path connecting v and w in G. Then P is 3-patrollable.*

Proof. Throughout the proof, we always consider the canonical drawing of G. Moreover, we will use the notation $B_r(x)$ for $x \in \mathbb{R}^2$ and $r \in (0, \infty)$ to denote the disk $\{y \in \mathbb{R}^2 : \|x - y\|_2 \leqslant r\}$. Furthermore, we use the alternative definition of a unit disk graph introduced below Definition 2.2.5.

Label the cops as C, C^- and C^+ and write $P = (v_1, \dots, v_k)$ for some $k \in \mathbb{N}$. We will first show that after a finite number of moves, C can guard P using the strategy from Proposition 4.1.3 and if $C = v_i$, always $C^- = v_{i-1}$ and $C^+ = v_{i+1}$ will hold, where we set $v_0 = v_1$ and $v_{k+1} = v_k$ for convenience. While this may sound trivial, there is some subtlety in having the cops C^- and C^+ acquiring their correct positions.

Start by letting the cops all move to some single vertex in G. Next, the cops move as one to guard P after a finite number of moves, which is possible by Proposition 4.1.3. Let $C = v_i$. If C remains stationary during some round, C^+ can move to v_{i+1} and C^- to v_{i-1} and the cops have assumed their intended positions. If C moves to v_{i-1}, then C^- moves along while C^+ remains stationary, putting C^+ into the right position. Similarly, we can put C^- in the right position if C moves to v_{i+1}. Noting that the path is of finite length, C must move in both directions or remain stationary after a finite amount of moves, after which we are in the desired situation. Now, the cops will always remain in this formation for the remainder of the game.

Hence, suppose that $C_t = v_i$, $C_t^+ = v_{i+1}$ and $C_t^- = v_{i-1}$ at time t. Suppose it is the robber's turn to move, so that we may denote the robber's position at this time by R_{t-1} (assuming that $t \geqslant 1$ w.l.o.g.). We will show that the cops effectively patrol P in this position. As C guards P by the above, we only need to check that R cannot cross P. Reasoning by contradiction, suppose the robber crosses P during his move to R_t.

First, note that $R_{t-1} \notin B_1(C_t)$ must hold. Suppose not, then $R_{t-1}C_t \in E(G)$. If $C_{t-1} = C_t$, then the cop C could have caught the robber in her previous move, which is in contradiction with the fact that the robber is crossing P without being caught. Otherwise, we have that either $C_{t-1}^+ = C_t$ or $C_{t-1}^- = C_t$, so that either C^+ or C^- could have caught the robber in the previous round. So indeed $R_{t-1} \notin B_1(C_t)$.

Furthermore, we observe that $R_{t-1} \notin B_1(v_{i-2}) \cup B_1(v_{i+2})$ (if $i - 2, i + 2 \in [k]$). Indeed, suppose $R_{t-1} \in B_1(v_{i-2})$. Then $R_{t-1}v_{i-2} \in E(G)$, so the robber can move onto P

without immediately being caught by C in the next round, as $v_{i-2}v_i \notin E(G)$ because P is a shortest path. Contradiction with the fact that C guards P. By similar reasoning, we conclude that also $R_{t-1} \notin B_1(v_{i+2})$ and that, moreover, $R_{t-1} \notin B_1(v_j)$ for $j \in [k]$ with $|i-j| > 2$. Therefore, the robber must cross P between v_{i-2} and v_{i+2}. See also Figure 4.2.1.

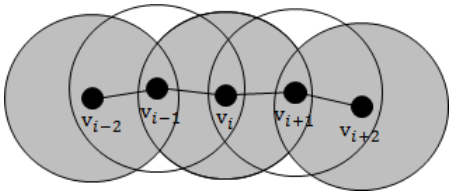

Figure 4.2.1: Part of G in which R must cross P. R_t does not lie in the gray area.

Without loss of generality, assume that R crosses P between v_i and v_{i+2}. Note that $R_t \notin B_1(v_i) \cup B_1(v_{i+1})$, as then either C or C^+ could catch the robber in her next move. Suppose first that $R_{t-1}R_t$ crosses v_iv_{i+1} as in Figure 4.2.2 and consider the triangle $\triangle R_{t-1}v_iv_{i+1}$. By the above reasoning, $\|v_i - v_{i+1}\|_2 \leqslant 1$ and $\|R_{t-1} - v_i\|_2 > 1$. Now, suppose $\angle v_iR_{t-1}v_{i+1} \geqslant \frac{\pi}{2}$. By the law of cosines, we then have that

$$\begin{aligned}\|v_i - v_{i+1}\|_2^2 &= \|v_i - R_{t-1}\|_2^2 + \|v_{i+1} - R_{t-1}\|_2^2 \\ &\quad - 2\|v_i - R_{t-1}\|_2\|v_{i+1} - R_{t-1}\|_2 \cos(\angle v_iR_{t-1}v_{i+1}) \\ &\geqslant \|v_i - R_{t-1}\|_2^2 + \|v_{i+1} - R_{t-1}\|_2^2 \\ &\geqslant \|v_i - R_{t-1}\|_2^2,\end{aligned}$$

so $1 \geqslant \|v_i - v_{i+1}\|_2 \geqslant \|v_i - R_{t-1}\|_2 > 1$, a contradiction. Therefore, $\angle v_iR_{t-1}v_{i+1} < \frac{\pi}{2}$ must hold. Considering the triangle $\triangle R_tv_iv_{i+1}$ and noting that $\|R_t - v_{i+1}\|_2 > 1$, we may again apply the law of cosines to conclude that also $\angle v_iR_tv_{i+1} < \frac{\pi}{2}$ holds.

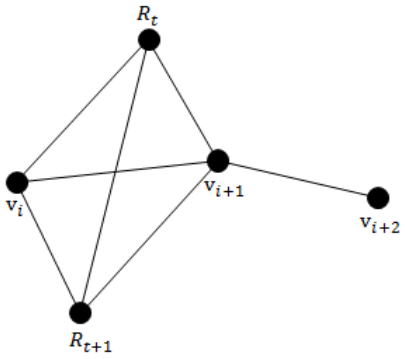

Figure 4.2.2: Vertices and edges involved in the crossing of R.

As the angles of a quadrilateral add up to 2π, we derive that $\angle R_{t-1}v_iR_t > \frac{\pi}{2}$ or $\angle R_{t-1}v_{i+1}R_t > \frac{\pi}{2}$. We can thus apply the cosine rule to either of the two triangles $\triangle R_{t-1}v_iR_t$ or $\triangle R_{t-1}v_{i+1}R_t$ to prove that $\|R_{t-1} - R_t\|_2 > 1$ must hold, which is in contradiction with $R_{t-1}R_t \in E(G)$. Hence the robber cannot cross v_iv_{i+1}. Replacing v_i by v_{i+2}, we can repeat this argument to show that the robber also cannot cross $v_{i+1}v_{i+2}$ safely. This concludes the argument. □

In a unit disk graph, three cops can thus always prevent the robber from crossing a

shortest path between two vertices. The following example shows that three cops are sometimes indeed necessary.

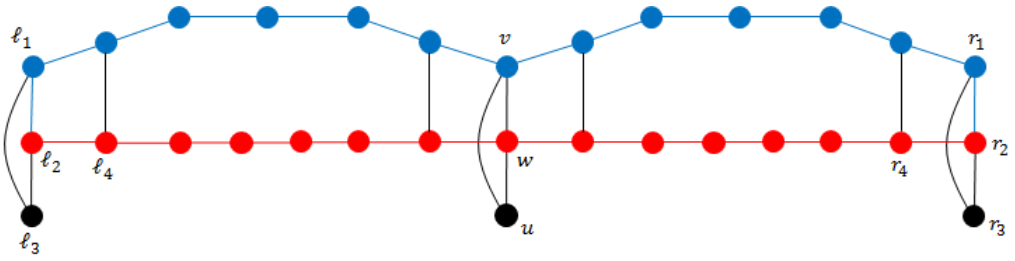

Figure 4.2.3: Three cops are necessary to patrol the red path.

Example 4.2.3. We will construct a unit disk graph G using the alternative definition, specifying a set of points in the plane representing vertices that are connected if and only if the distance between two points is at most 1. We define points at: $(0,0)$, $(0,\pm 0.49)$, $(\pm 0.9,0)$, $(\pm 0.9,0.9)$, $(\pm 1.8,0)$, $(\pm 1.8,1.3)$, $(\pm 2.4,0)$, $(\pm 2.7,1.3)$, $(\pm 3,0)$, $(\pm 3.6,0)$, $(\pm 3.6,1.3)$, $(\pm 4.5,0)$, $(\pm 4.5,0.9)$, $(\pm 5.4,0)$, $(\pm 5.4,0.49)$ and $(\pm 5.4,-0.49)$. Simple calculations using Pythagoras' theorem show that the drawing in Figure 4.2.3 accurately represents the resulting graph. Note that both the red and blue paths are shortest between ℓ_2 and r_2, having length 14.

Consider two cops playing a game against the robber on G. We will show that the two cops are not sufficient to patrol the red path. Suppose the robber ends his move on the vertex v. After the next move of the cops, one cop, say C_1, must occupy the vertex w as otherwise the robber can move to u without being caught.

Now, suppose the second cop C_2 is in the left half of the graph. The robber may then move to the right along the blue path. When he reaches the vertex r_1, the cop C_1 can only move to end her next turn on r_4 at the furthest. As C_2 started to the left of C_1, neither she can have come further. Hence, no cop is on r_2 at the start of the next turn of the robber, so he may cross to r_3 without being caught. If C_2 started on the right half of the graph, the robber could have escaped to ℓ_3 via an entirely symmetrical argument. Hence indeed three cops are necessary to patrol the red path.

It is now simple to prove the intended generalisation of the result from the previous section for unit disk graphs.

Theorem 4.2.4 ([7])**.** *For any unit disk graph G, it holds that $c(G) \leqslant 9$.*

Proof. Consider the canonical drawing of G and let nine cops play on the graph. We divide the cops into three teams of 3 cops each. Now, the proof of Theorem 4.1.6 may be copied verbatim, replacing every instance of a shortest path being guarded by a single cop by the path in question being patrolled by a team of three cops, applying Proposition 4.2.2 insead of Proposition 4.1.3. Noting that at most three paths need to be guarded at the same time, we see that $3 \cdot 3 = 9$ cops indeed suffice. □

4.3 String graphs

For string graphs, Theorem 4.1.6 can also be generalised, but now we need even more cops. The underlying idea is still the same, however: guard some structure in the graph

and gradually reduce the territory of the robber. In this section, we closely follow Sections 4 and 5 of [21], starting with the following statement extending Proposition 4.1.3 to the structure of interest.

Proposition 4.3.1 ([21]). *Let G be a graph and let $v, w \in V(G)$. Let P be a shortest path connecting v and w in G. Then $N[P] := N[V(P)]$ is 5-guardable.*

Proof. Write $P = (v_1, \dots, v_k)$ for $v_i \in V(G)$, $i \in [k]$, $k \in \mathbb{N}$. Analogously to the first part of the proof of Proposition 4.2.2, we can show that the five cops can guarantee that after a finite amount of moves, one cop (C_1 say) guards P and the other cops move in such a way that if $C_1 = v_i$, then $C_2 = v_{i-1}$, $C_3 = v_{i-2}$, $C_4 = v_{i+1}$ and $C_5 = v_{i+2}$. We again adopt the convention that $v_{-1} = v_0 = v_1$ and $v_k = v_{k+1} = v_{k+2}$. We claim that the cops moving in this formation guard $N[P]$.

Let R end his turn on some vertex $w \in V(G)$ which is adjacent to some $u \in N(P)$ and suppose $uv_i \in E$. Then C_1 must end her next move on one of the vertices in $\{v_{i-2}, v_{i-1}, v_i, v_{i+1}, v_{i+2}\}$, as otherwise the robber could move to v_i in his next two turns, not immediately being caught by C_1 afterwards, which is in contradiction with C_1 guarding P. Therefore, some cop C_j with $j \in \{2, 3, 4, 5\}$ must end her move on v_i, which prevents R from moving to u without being caught by C_j. □

In the literature, a structure akin to the shortest path and its neighborhood is sometimes called a *minimum distance caterpillar* or *mdc* for short [12]. In what follows, we will work out the details of our proof of reducing the robber territory. These details now heavily depend on a string representation of our graph. The following notion will therefore be useful.

Definition 4.3.2. Let G be a string graph and fix some string representation of G. Let $P = (v_1, \dots, v_k)$ be some shortest path between $v_1, v_k \in V(G)$, $k \in \mathbb{N}$. Fix $a \in \gamma_{v_1}$ and $b \in \gamma_{v_k}$. Let $\phi \subseteq \bigcup_{i=1}^{k} \gamma_{v_i}$ be a curve in the plane from a to b such that for every $i \in [k]$ the set $\gamma_{v_i} \cap \phi$ is connected and these $\gamma_{v_i} \cap \phi$ are ordered in ϕ in the same manner as the v_i in P. Then ϕ is called a *shortest curve* of P, and P is said to be the path associated with ϕ.

We can now speak of guarding shortest curves.

Definition 4.3.3. Let ϕ be a shortest curve in some string representation of a string graph G and let P be the path associated with ϕ. Then ϕ is said to be *k-guardable* if $N[P]$ is k-guardable in the sense of Definition 4.1.1.

The following is then a corollary of Proposition 4.3.1.

Corollary 4.3.4 ([21]). *Let ϕ be a shortest curve in some string representation of a string graph G. Then ϕ is 5-guardable.*

Note that once the five cops move in such a way that they guard ϕ, the robber is prevented from ever entering a vertex represented by a string intersecting ϕ without being caught in the next round. Indeed, any string intersecting ϕ represents either a vertex on P or adjacent to P, i.e., a vertex in $N[P]$. Therefore, intuitively, the robber cannot "cross" ϕ in the string representation of our graph and shortest curves may thus be used as "borders" for the robber territory. We turn to the details.

To start, we need some topological terms. Let $\mathcal{C} = \{\gamma_1, \dots, \gamma_n\}$, $n \in \mathbb{N}$ be a finite connection of curves. By the *faces* of $\mathbb{R}^2 \backslash \mathcal{C}$, we mean the connected components of $\mathbb{R}^2$ that emerge when the curves in $\mathcal{C}$ are removed. These faces are open sets in $\mathbb{R}^2$. For any

set $X \subseteq \mathbb{R}^2$, we write $\overline{X}$ for its *closure*, $\text{int}(X)$ for its *interior* and $\partial X = \overline{X} \setminus \text{int}(X)$ for its *border*. If $F \subseteq \mathbb{R}^2$ is a face, we call $\overline{F}$ a *closed face*. Furthermore, for $F_1, \ldots, F_n$ faces, $n \in \mathbb{N}$, we call $\overline{\bigcup_{k=1}^{n} F_k}$ a *region*.

A curve $\gamma : [0,1] \to \mathbb{R}^2$ is called *simple* if $\gamma(t_1) \neq \gamma(t_2)$ for all $t_1, t_2 \in [0,1]$. If we have that $\gamma(t_1) = \gamma(t_2)$ only for $t_1 = 0$ and $t_2 = 1$, then γ is called a *simple closed curve*, also called a *Jordan curve*. We call two curves γ_1 and γ_2 between two points $a, b \in \mathbb{R}^2$ *internally disjoint* if $\gamma_1 \cap \gamma_2 = \{a, b\}$. For any curve γ in a collection of curves $\mathcal{C}$, a *segment* of γ is an arc-connected subcurve of γ of maximal length that does not intersect any of the other curves in $\mathcal{C}$.

Next, consider a subset $X \subseteq \mathbb{R}^2$ and a string graph G. We say a vertex $v \in V(G)$ is *contained in* X if $\gamma_v \subseteq \text{int}(X)$, where γ_v is the curve corresponding to v in the string representation of G. By G_X, we denote the subgraph of G induced by the vertices contained in X.

Finally, we end this enumeration of topological concepts by stating the elementary fact that in the plane $\mathbb{R}^2$, one cannot enter nor leave the area encircled by a closed curve without crossing the curve. This fact is formalized as the *Jordan Curve Theorem*.

Theorem 4.3.5 ([39]). *Let ϕ: $[0,1] \to \mathbb{R}^2$ be a simple closed curve. Then $R^2 \setminus \phi([0,1])$ consists of exactly two connected components, one of which is bounded and one of which is unbounded, of which $\phi([0,1])$ is the boundary. The bounded component is called the* interior *of ϕ and the unbounded component is called the* exterior.

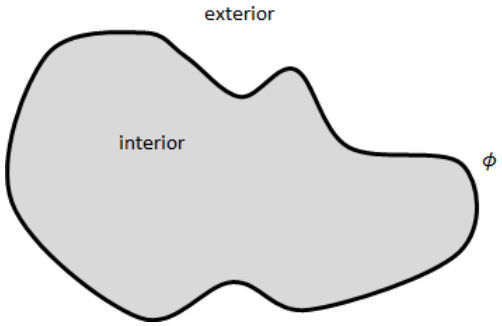

Figure 4.3.1: The curve ϕ divides $\mathbb{R}^2$ in an interior and an exterior region.

The statement of the Jordan Curve Theorem is demonstrated in Figure 4.3.1. Though the statement is intuitive, the known proofs of the theorem either require a solid background in (algebraic) topology or some extensive analysis, so we refer to the literature instead of displaying it here; see [39], for example.

We now phrase a result of Corollary 4.3.4 using the introduced terminology. Let $\gamma_1, \gamma_2 \subseteq \mathbb{R}^2$ be two internally disjoint curves from $a \in \mathbb{R}^2$ to $b \in \mathbb{R}^2$. Suppose that the string that contains the robber does not intersect $\gamma_1 \cup \gamma_2$. Let F be the closed face of $\mathbb{R}^2 \setminus (\gamma_1 \cup \gamma_2)$ containing the string with the robber. Let D be the connected component of G_F that contains the robber. If γ_1 and γ_2 are shortest curves relative to D, then these curves can be guarded by 10 cops in total in such a way that the robber is confined to D.

This observation will be crucial in our proof. Moreover, we need the following technical topological result.

Lemma 4.3.6 ([21]). *Let $a, b \in \mathbb{R}^2$, $a \neq b$. Suppose that $\gamma_1, \gamma_2 \subseteq \mathbb{R}^2$ are internally disjoint simple curves from a to b. Let F be a closed face of $\mathbb{R}^2 \setminus (\gamma_1 \cup \gamma_2)$, i.e., the*

closure of a face of $\mathbb{R}^2\backslash(\gamma_1 \cup \gamma_2)$*, and let* $\gamma_3 \subseteq F$ *be a simple curve from* a *to* b *for which* $\gamma_3 \cap \text{int}(F) \neq \varnothing$*. Then every face of* $F\backslash(\gamma_1 \cup \gamma_2 \cup \gamma_3)$ *is bounded by two simple internally disjoint curves* γ_i' *and* γ_3' *with* $\gamma_i' \subseteq \gamma_i$ *for some* $i \in \{1, 2\}$ *and* $\gamma_3' \subseteq \gamma_3$*.*

Proof. Without loss of generality, assume that F is the inner face of $\mathbb{R}^2\backslash(\gamma_1 \cup \gamma_2)$; otherwise, apply Lemma 4.1.5. Let D be an arbitrary face of $F\backslash(\gamma_1 \cup \gamma_2 \cup \gamma_3)$. By definition, D is open and arc-connected, so ∂D is a simple closed curve.

We claim that $\partial D \subseteq \gamma_i \cup \gamma_3$ for some $i \in \{1, 2\}$. To prove the claim, we need three ingredients. First, note that $\partial D \subseteq \gamma_j$ does not hold for $j \in \{1, 2, 3\}$. If this was the case, then γ_j could not be a simple curve, as it would be a curve connecting two points a and b with a closed curve being part of it.

Second, note that there must exist some $c \in (\partial D \cap \gamma_3)\backslash(\gamma_1 \cup \gamma_2)$. Otherwise, $\partial D \cap \gamma_3 \subseteq \gamma_1 \cup \gamma_2$ would hold and noting that $\partial D \subseteq \gamma_1 \cup \gamma_2 \cup \gamma_3$, we could conlude that $\partial D \subseteq \gamma_1 \cup \gamma_2$, so $\partial D = \gamma_1 \cup \gamma_2$. It then follows that $\overline{D} = F$, which is in contradiction with the assumption that $\text{int}(F) \cap \gamma_3 \neq \varnothing$.

Third, note that there cannot exist both a $c_1 \in (\partial D \cap \gamma_1)\backslash(\gamma_2 \cup \gamma_3)$ and a $c_2 \in (\partial D \cap \gamma_2)\backslash(\gamma_1 \cup \gamma_3)$. If these would both exist, then we could find a curve γ' in D connecting c_1 and c_2. Then $\gamma' \cap \gamma_3 = \varnothing$, but γ' separates a from b in F whereas γ_3 connects a and b in F, which is impossible by the Jordan Curve Theorem. See also Figure 4.3.2. The claim follows.

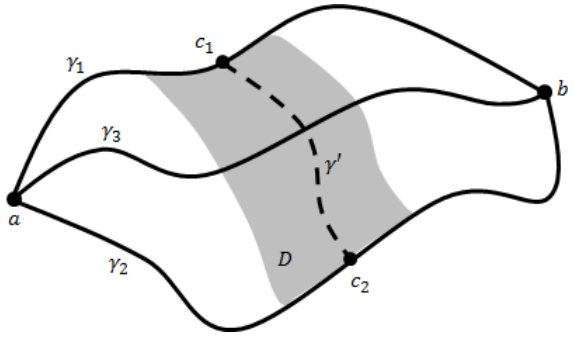

Figure 4.3.2: The path γ' that can be drawn if c_1 and c_2 both exist.

Assume now without loss of generality that c_1 like above exists, so that $\partial D \subseteq \gamma_1 \cup \gamma_3$. Define $\gamma_1' = \overline{(\partial D \cap \gamma_1)\backslash\gamma_3}$. Let d_1 be the first point on γ_1' when going from a to b via γ_1 and let d_2 be the last such point. Suppose for the sake of contradiction that γ_1' is not connected and let $d_3 \neq d_1, d_2$ be an endpoint of one segment of γ_1'. Then $d_3 \in \gamma_3$ must hold, as $\partial D \subseteq \gamma_1 \cup \gamma_3$ and d_3 is an endpoint of γ_1' whilst $\gamma_1' \subseteq \gamma_1$. But then γ_3 cannot be a simple curve as illustrated by Figure 4.3.3, as it would contain a closed curve as part of it. Therefore, γ_1' is connected.

It follows that also $\gamma_3' = \overline{\partial D\backslash\gamma_1} \subseteq \gamma_3$ must be connected, so the curves γ_1' and γ_3' provide what we are looking for. □

We introduce one more important concept. Let G be a string graph and let $R \subseteq \mathbb{R}^2$ be a region in the string representation $\mathcal{C}$ of G. By $G|_R$, pronounced as G *restricted to* R, we denote the intersection graph of $\mathcal{C} \cap R$. We write $V|_R$, $E|_R$ and $\mathcal{C}|_R$ for the vertex set, edge set and string representation of the restricted graph. As the following example shows, single vertices in G may be "split" into multiple vertices in $V|_R$ (but the converse cannot happen). If $v \in V$ gives rise to $(v_i)_i \subseteq V|_R$, we call v_i the *shards* of v.

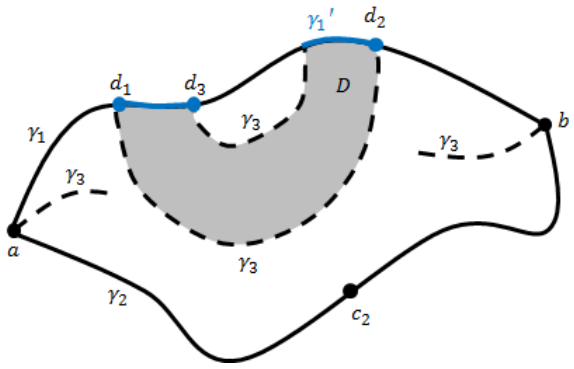

Figure 4.3.3: If γ_1' is not connected, then γ_3 is not simple.

Example 4.3.7. Consider G to be a path of three vertices with string representation $\mathcal{C}$ as shown in Figure 4.3.4. Let R be the region formed by the infinite outer face. In $\mathcal{C} \cap R$, the string γ_3 is split into two strings $\gamma_{3,1}$ and $\gamma_{3,2}$, as a segment is taken from the middle of the string. Therefore, in the resulting restricted graph $G|_R$, the original vertex 3 is split into two shards 3,1 and 3,2.

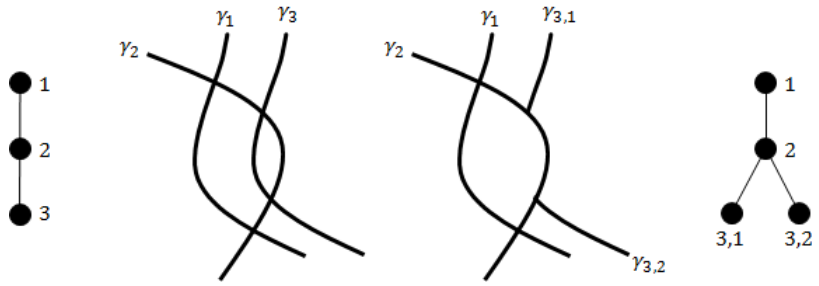

Figure 4.3.4: The graph G, its string representation $\mathcal{C}$, the restriction $\mathcal{C} \cap R$ and its intersection graph $G|_R$.

We discuss two properties of restricted graphs. If R is some region in a string representation $\mathcal{C}$ of a string graph G and ϕ is a shortest curve in G, then any subcurve $\phi' \subseteq R$ of ϕ is a shortest curve in $G|_R$. Indeed, if P is the path associated with ϕ in G, then this path can be transformed into a path P' associated with ϕ' by keeping all vertices which were unchanged during the restriction and replacing any vertices by one of their shards if necessary. Noting that the segments in R remain unaltered by the restriction, the string representation of P' will contain equally many intersections as that of P, implying that P' is of the same length. As no other path is shortened by the restriction, P' is a shortest path so ϕ' is indeed a shortest curve.

Furthermore, if S is a region such that the robber is in G_S and the cops have a strategy for capturing the robber in $G|_S$ while confining him to G_S for the rest of the game, then the cops can also capture the robber in G. The strategy used reminds of the "shadow strategies" extensively employed in the first chapter: the cops in G mirror the moves of the cop in $G|_S$. If a cop would move to a shard in $G|_S$, then the corresponding cop moves to the original vertex in G, which is by construction always a legal move. In this fashion, R is confined to G_S for the rest of the game and eventually captured.

Now, we are finally ready to prove an upper bound on the cop number of string graphs.

Theorem 4.3.8 ([21]). *For any string graph G, it holds that $c(G) \leqslant 15$.*

Proof. Fix a string representation of G and consider 15 cops playing on the graph. We mimic the proof of Theorem 4.1.6, but now focus on the topology of the string representation instead of that of a drawing of G. We will use the results derived earlier in this section to show, inductively, that the area in which the robber can move can be reduced to nothing.

To make things a little easier along the way, we will remove vertices and their incident edges from the graph if they are no longer relevant, i.e., if they are not currently guarded by the cops and the robber can no longer reach them. Note that if the cops can win on this reduced graph, then they can also win on the original graph, as the removed vertices could only be of use for the cops.

Before we truly start, we introduce some more notation. Denote by B the vertices in the structures that are guarded by the cops at any current time. As we will only guard shortest curves and single vertices, by Corollary 4.3.4, if the robber moves to $N[B]$, he is captured in the next round at the latest. Let $A \subseteq V(G)$ denote the current "robber territory", i.e., the vertices in the connected component of our current graph with $N[B]$ removed. We write $D = N[B] \cap N[A]$ and let s be the number of segments present in the string representation of the current graph.

The induction will be on $s + |B|$; in every step, either the number of segments in the string representation of the graph or the size of the robber territory decreases. It is clear that if either value reaches zero, the robber is caught. As in the proof of Theorem 4.1.6, we distinguish between two situations

Situation (1). $|B| = 1$; one cop currently guards the single vertex $b \in B$.

Situation (2). $|B| \geqslant 2$; 10 cops guard two shortest curves ϕ_1, ϕ_2 in A between points $a, b \in \mathbb{R}^2$ such that $\phi_1 \cup \phi_2$ is a simple closed curve, and $G = G|_F$ holds with F the closed face in $\mathbb{R}^2 \backslash (\phi_1 \cup \phi_2)$ containing the string with the robber γ_R. Note that if the cops guard these curves ϕ_1 and ϕ_2, then $\phi_1 \cap \gamma_R = \phi_2 \cap \gamma_R = \varnothing$ must indeed hold, as otherwise the robber can be caught immediately and we are done.

It is clear that the cops can begin the game in situation (1) by letting all 15 cops start on some randomly chosen vertex in G. We will now show that, starting in either situation (1) or (2), we can return to either of the situations, having reduced either s or $|A|$ by at least one. The case analysis is very similar to that in the proof of Theorem 4.1.6 and, as such, we quickly move over some details at times.

Assume first we are in situation (1). Suppose that $D = \{d\}$ consists of a single vertex. The cop that is now guarding b can then move to d, so that A is reduced in size by at least one. We reduce the graph to the subgraph induced by $A \cup \{d\}$ for this new set A and are back in sutation (1).

Next suppose that $D = \{d_1, \ldots, d_k\}$ for some $k > 1$. Define the points $a_i \in \mathbb{R}^2$ by $a_i = \gamma_b \cap \gamma_{d_i}$ for $i \in [k]$. Let ϕ_1 be a shortest curve between some a_i and a_j not using b, $i \neq j$, which always exists as otherwise $|D| = 1$ would hold, and let $\phi_2 \subseteq \gamma_b$ be the subcurve between a_i and a_j. See also Figure 4.3.5.

Note that $\phi_1 \cup \phi_2$ is a simple cycle. Moreover, ϕ_2 can be guarded by a single cop on b and ϕ_1 can be guarded by 5 cops by Corollary 4.3.4 as it is a shortest curve. Note that these cops are indeed available as we are currently only using one cop to guard b. Now, let F be the closed face of $\mathbb{R}^2 \backslash (\phi_1 \cup \phi_2)$ containing γ_R. Again, γ_R is indeed fully contained in such a face, because otherwise a cop could immediately catch the robber. Consider the graph defined by $G' = G|_F$. Note that we may without loss of generality continue by playing the game on G', as a winning strategy for the cops on $G|_F$ induces

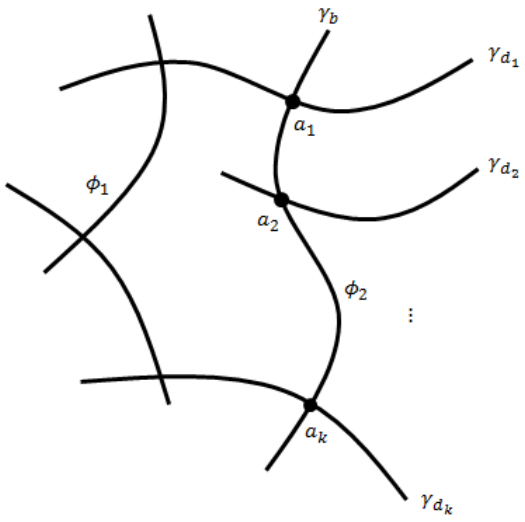

Figure 4.3.5: Situation (1) with $|D| > 1$.

a winning strategy for the cops on G. Therefore, we are now in situation (2), having reduced the size of the robber territory.

Next consider situation (2). We again distinguish two cases. First, suppose that there does not exist a shortest curve in the string representation of the vertices in A that connects the two points a and b that intersects $\text{int}(F)$. See Figure 4.3.6 for a sketch of the situation (assuming w.l.o.g. that F is finite). There must then exist some single vertex $v \in B \cup D$ that separates B and A. One of the free cops may then move to and guard v. Let $G' = G((V \backslash B) \cup \{v\})$. We are then back in situation (1), having reduced the amount of strings in the graph by at least one.

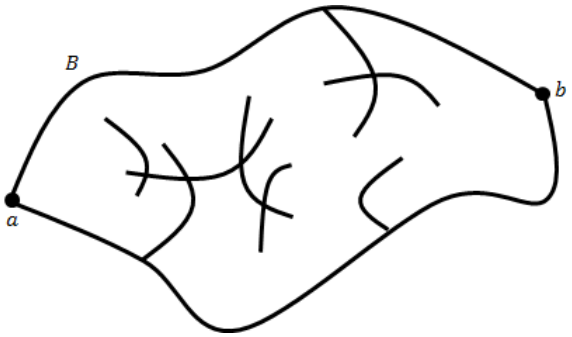

Figure 4.3.6: Situation (2) with no internal shortest curves between a and b.

Finally, suppose that there do exist shortest curves of the required form and let ϕ_3 be such a curve. Use the five free cops to guard ϕ_3 as per Corollary 4.3.4. Let F' be the closed face of $\mathbb{R}^2 \backslash (\phi_1 \cup \phi_2 \cup \phi_3)$ containing γ_R. By Lemma 4.3.6, $\partial F' \subseteq \phi_1' \cup \phi_3'$ must hold for some $\phi_i' \subseteq \phi_i$, $i \in \{1, 2\}$, $\phi_3' \subseteq \phi_3$. Moreover, $\phi_1' \cup \phi_3'$ forms a simple cycle. Note that ϕ_i' and ϕ_3' may be guarded as subcurves of the currently guarded ϕ_i and ϕ_3 respectively. Therefore, defining $G' = G|_{F'}$, we are back in situation (2), having reduced the amount of segments in the graph by at least one. This concludes the proof. □

As a conclusion to this section, we note that the upper bounds proven in this and the

previous sections may also be used to bound the complexity of the decision problems regarding the game of cops and robbers on geometric graphs introduced in Section 3.3. We illustrate this by proving Theorem 3.3.1.

Proof of Theorem 3.3.1. Consider the problem k-PLANAR COP NUMBER. If the input is $k \in \{1, 2\}$, we may apply an algorithm from Section 3.1 to solve the problem in polynomial time. If $k > 2$, then the answer is always 'yes' by virtue of Theorem 4.1.6, which may be returned in constant time. This shows that our problem is indeed in P.

For the problems k-UDG COP NUMBER and k-STRING COP NUMBER, the same reasoning applies, using the upper bound proven in Theorem 4.2.4 or Theorem 4.3.8, respectively: if $k \geqslant 9$ or $k \geqslant 15$ holds, the answer is always 'yes', and otherwise the problem can be solved in polynomial time. Hence also these problems are in P. □

4.4 Treewidth

We conclude this chapter with yet another bound based on the guardability of shortest paths, but which does not depend on one or the other graph class. For this bound, we need the notion of treewidth.

Definition 4.4.1. Let G be a graph. A *tree decomposition* of G is a pair $(T, \mathcal{W})$ such that T is a tree and $\mathcal{W} = \{W_x \mid x \in V(T)\}$ is a collection of subsets of $V(G)$ such that the following three properties hold:

(1) $\bigcup_{x \in V(T)} W_x = V(G)$,

(2) For every $uv \in E(G)$ there is some $x \in V(T)$ such that $u, v \in W_x$,

(3) For every $v \in V(G)$, the set $\{x \in V(T) \mid v \in W_x\}$ induces a subtree of T.

The elements of $\mathcal{W}$ are called *bags*. The *width* of a tree decomposition $(T, \mathcal{W})$ is

$$\max\{|W_x| - 1 : x \in V(T)\}.$$

The *treewidth* of G, denoted $\mathrm{tw}(G)$, is the minimum width among all tree decompositions of G.

To gain some feeling for this new graph parameter, we determine the treewidth for two simple types of graphs.

Example 4.4.2. Let G be a tree with $|V(G)| \geqslant 2$. We claim that $\mathrm{tw}(G) = 1$. First note that for any tree decomposition $(T, \mathcal{W})$ of G, as G contains at least one edge, by property (2), there must be at least one bag in $\mathcal{W}$ having two or more elements. Therefore, $\mathrm{tw}(G) \geqslant 1$. To prove that in fact $\mathrm{tw}(G) = 1$, it thus suffices to find a tree decomposition of G having width 2.

Root G at some arbitrary vertex $r \in V(G)$. We start the construction of our tree decomposition $(T, \mathcal{W})$ by defining a bag $W_r = \{r\}$ and a corresponding node in T, which we will call t_r. Now, starting in this root r, we visit every vertex in the tree using a depth-first search. Whenever we traverse an edge $e = rv \in E(G)$ connecting the root r to a vertex $v \in V(G)$ for the first time, we add a bag $W_e = \{r, v\}$ to $\mathcal{W}$ and a node t_e to T, which we connect to W_r.

Whenever we traverse an edge $e = vw \in E(G)$ connecting a vertex v having distance i to r to a vertex w having distance $i + 1$ to r for the first time, $i > 0$, we add a bag $W_e = \{v, w\}$ to $\mathcal{W}$ and a node t_e to T. Note that because G is a tree, there must be a

unique edge $f = uv \in E(G)$ that connects v to some vertex $u \in V(G)$ having distance $i-1$ to r. As this edge has already been traversed in our depth-first search, the bag W_f and corresponding node t_f have already been added to $\mathcal{W}$ and T, respectively. We can therefore connect t_e to t_f in T.

As our tree G is finite, the depth-first search will end after a finite number of iterations, at which point we have constructed a pair $(T, \mathcal{W})$. Remains to check that this is indeed a tree decomposition of G having width 2. First, note that T is a tree by construction, as every node $t_e \in V(T)$ representing an edge $e \in E(G)$ is only connected to exactly one node representing an edge at a smaller distance to r (or to the node representing r itself), so that no cycles can be formed.

Next, we check the three required properties. As G is a connected tree, to visit every vertex in G using depth-first search, we must also traverse every edge. Therefore, a bag of the form v, w is added to $\mathcal{W}$ for every edge $vw \in E(G)$, so properties (1) and (2) are satisfied. For property (3), note that for the root vertex r, we have that r is contained precisely in the bags W_r, and W_e for all $e = rv \in E(G)$ for some $v \in V(G)$. Every corresponding node t_e is connected to t_r by construction and these nodes are not connected among themselves, so property (3) is satisfied for r. Similarly, for any $v \in V(G)$ unequal to r, we have that $v \in W_e$ precisely for one edge $wv \in E(G)$ connecting v to a vertex w closer to r, and $v \in W_f$ for all edges $vu \in E(G)$ connecting v to a vertex u further from r in G. As $t_e t_f \in E(T)$ for all these f and no other edges are present between these nodes, property (3) is indeed also satisfied for $v \neq r$.

Finally, note that every bag has size either 1 (the bag W_r) or 2 (every other bag), so the width of the tree decomposition is indeed 2. This completes the proof that $\mathrm{tw}(G) = 1$.

Example 4.4.3. For the complete graph on $n \in \mathbb{N}$ vertices, we have $\mathrm{tw}(K_n) = n - 1$. To prove this, we first claim that for a tree T and any collection of subtrees $\mathcal{T}$ of T in which every pair of subtrees has at least one vertex in common, there must be a vertex in T that is part of every subtree in $\mathcal{T}$. Indeed, root T at some arbitrary node. Then root every subtree in $\mathcal{T}$ on the node having minimum distance to the root of T. We see that any subtree in $\mathcal{T}$ must then intersect the root of the subtree which is rooted on the level furthest from the root of T, which thus turns out to be a common vertex for all subtrees in $\mathcal{T}$.

Now, let $(T, \mathcal{W})$ be some tree decomposition of K_n. Consider the subgraphs of T induced by the sets $\{x \in V(T) \mid v \in W_x\}$ for every $v \in V(K_n)$. Note that each of these subgraphs is a tree by property (3). Furthermore, every two of these subgraphs have a common node in T by property (2). Therefore, by the claim proven above, there is some vertex $x \in V(T)$ which is contained in each of the subtrees, so $v \in W_x$ holds for all $v \in V(G)$ and hence $W_x = V(G)$. The width of the composition is therefore at least $|W_x| - 1 = n - 1$.

To complete the proof of K_n having treewidth $n - 1$, it thus suffices to find a tree decomposition of K_n with width $n - 1$. It is easily checked that the decomposition $(T, \mathcal{W})$ defined by $T = K_1$, a single vertex, and $W = V(G)$ being the corresponding bag does the job.

The treewidth thus gives some indication of how "tree-like" a graph is. As trees have cop number 1, we therefore expect graphs to have a lower cop number if the treewidth is lower. This is indeed the case, as is shown by the following upper bound, proven in [25].

Theorem 4.4.4 ([25])**.** *Let G be a graph. Then $c(G) \leqslant \mathrm{tw}(G)/2 + 1$.*

Proof. Consider a tree decomposition $(T, \mathcal{W})$ of G with minimal width. Throughout, for a bag $W \in \mathcal{W}$, by t_W we denote the node of T corresponding to W. The idea of the

proof much like what we have seen before: we let the cops guard a part of the graph so that the robber is confined to his own "territory" and proceed to reduce the size of this territory. In this installment, the cops will be guarding all vertices in some bag in $\mathcal{W}$, which will confine the robber to the component of $T \backslash t_W$ in which he is currently situated.

At the start of the game, select a bag $W \in \mathcal{W}$ with maximum size. Let $W = \{v_1, \ldots, v_k\}$, $k \in \mathbb{N}$. For $i \in \{1, \ldots, \lfloor k/2 \rfloor\}$, let a cop guard a shortest path between the vertices v_{2i-1} and v_{2i}. If k is odd, another cop is positioned to simply sit on v_k. In this way, the robber is prevented from entering any vertex in W by

$$\lceil |W|/2 \rceil = \lceil (\mathrm{tw}(G) + 1)/2 \rceil \leqslant \mathrm{tw}(G)/2 + 1$$

cops. If $W = V(G)$, we are done. Otherwise, we claim that by the cops guarding W, the robber is restricted to the vertices in the bags associated with one of the components T' of $T \backslash t_W$. First, note that the robber cannot move to a vertex in W. Therefore, to move from the one component of $T \backslash t_W$ to another, say from T_1 to T_2, the robber must move from some vertex v in a bag associated with some node in T_1 to a vertex w in a bag associated with a vertex in T_2. By property (2) of a tree decomposition, v and w must together be included in some bag, X say. However, by property (3), if this bag X is in T_1, this implies that $w \in W$, as w must be included in all bags in T on the path from X to the bag containing w in T_2 in T, and W is situated on this path by construction. Similarly, if X is in T_2, then $v \in W$ must hold. Neither can be true, which proves the claim.

Let W' be the bag in T' adjacent to W. Next, we claim that $W \cap W' \neq \varnothing$. Suppose $W \cap W' = \varnothing$. Divide the tree T' into two components T_W and $T_{W'}$ by removing the edge between t_W and $t_{W'}$. By property (3), none of the vertices in W can then be contained in a bag in $T_{W'}$ and vice versa. Hence, even none of the vertices in a bag in T_W can be contained in a bag in $T_{W'}$ and vice versa. Therefore, by property (2), none of the vertices in the bags in T_W are connected to any vertex in a bag in $T_{W'}$, so T' is disconnected, which is a contradiction. Hence the claim holds.

Furthermore, by similar reasoning as before, again using property (3), the robber cannot exit T' without passing through a vertex in $W \cap W'$. Therefore, as long as $W \cap W'$ remains guarded, R is restricted to T'. We will show that the cops can move from guarding W to guarding W' while always guarding $W \cap W'$.

We consider the cops one by one and distinguish three situations. First, if the cop is currently guarding a path with both endpoints in $W \backslash W'$ or is sitting on a vertex in $W \backslash W'$, she moves to guard a shortest path between two as of yet unguarded vertices in $W' \backslash W$ or to sit on one such vertex if only one remains. Note that this movement does not have any consequences for the guarding of $W \cap W'$.

Second, suppose the cop guards a shortest path between some vertices $v \in W \cap W'$ and $w \in W \backslash W'$ or sits on a vertex $v \in W \cap W'$. The cop then moves to v along the shortest path she is guarding (if she is not already there). Note that during this move, the distance from the cop to the vertex v decreases every turn, and as the cop started out guarding the path between v and w using the strategy from Proposition 4.1.3, she will never be further from v as this strategy would tell her to be. Therefore, the vertex v remains guarded during this move.

Next, she selects some unguarded vertex $w' \in W \backslash W'$ and starts moving to guard a shortest path between v and w'. Note that during this movement, the cop is again always at least as close to v as the path-guarding strategy of Proposition 4.1.3 dictates as she is moving along the path to acquire this position, so v is guarded during the whole manoeuvre.

Third, if the cop is currently guarding a path between two vertices in $W \cap W'$, she keeps doing so. Throughout, if all vertices in W' are at some point guarded while there are still cops in W left, the remaining cops simply move to occupy some vertex in W'. Hence, at the end of this migration, all cops are in W' and every vertex in W' is guarded.

The robber is now restricted to some strict subgraph of T', hence his "territory" has decreased. Therefore, after repeating this process a finite amount of times, the robber is caught. Noting that no bag takes more than the initial $\mathrm{tw}(G)/2 + 1$ cops to guard as we started with the bag with the largest size, we are done. □

Though we determined the treewidth of two simple graph classes in the examples above, in general, it is unfortunately hard to determine the treewidth of any arbitrary given graph. Considering the results of Chapter 2, this is not surprising. To formalize this statement, we introduce the following decision problem.

k-TREEWIDTH: Given as input some $k \in \mathbb{N}$ and a graph G, does $\mathrm{tw}(G) \leqslant k$ hold?

The result on the complexity of this problem is then as follows, the proof of which we omit here but can be found in [4].

Theorem 4.4.5 ([4])**.** *The problem k-*TREEWIDTH *is NP-complete for $k \geqslant 2$.*

Furthermore, the bound in Theorem 4.4.4 does not tell us anything about the cop number of the geometric graphs discussed here. Indeed, K_n is a unit disk graph and therewith a string graph for any $n \in \mathbb{N}$. Therefore, the treewidth of the class of unit disk graphs as well as that of the class of string graphs is unbounded.

5 Graphs with high cop number

In the previous chapter, we developed some upper bounds on the cop number for graphs. In this chapter, we will work from the other way: can we explicitly construct some graphs with a high cop number? As already mentioned before, for general graphs, the answer turns out to be yes. In Section 5.2, we will prove that if no restrictions are imposed, graphs with an arbitrarily high cop number exist.

For the geometric graph classes considered, as already indicated by the upper bounds found in the previous chapter, it is much harder to find examples of graphs having a high cop number. In Sections 5.3 and 5.4, we will discuss an example of a unit disk graph and a string graph having cop number 3, as well as the difficulties encountered when searching for a graph with higher cop number.

Just like how in the previous chapter every proof was based on more or less the same strategy for the cops, in this chapter we will see that every graph with a high cop number that we encounter is constructed using one and the same theorem. This theorem was published in [1], and we start with its treatment in Section 5.1.

5.1 Main theorem

As stated in the introduction of the section, we begin with a theorem that will prove to be a very useful (and perhaps the only known) tool in finding graphs with a high cop number later on. Recall that $g(G)$ denotes the girth of a graph G and $\delta(G)$ its minimum degree, as discussed in Section 2.1. We follow an adaptation of the proof outlined in [9].

Theorem 5.1.1 ([1], [9])**.** *Let G be a graph. If $g(G) \geqslant 5$, then $c(G) \geqslant \delta(G)$.*

Proof. The intuition of the proof is to show that if there are no 3-cycles or 4-cycles in G, any one cop can only guard at most one neighbor of the robber at any given time. Therefore, if every vertex has more neighbors than cops, the robber can always escape, no matter where he is.

For convenience, we will write $d = \delta(G)$. Let $d - 1$ cops play the game on G (for $d = 1$, the statement is trivially true). We need to show that the robber has a winning strategy. Let $C \subseteq V(G)$ be the initial position of the cops in round 0. We will prove that C cannot be a dominating set, i.e., that there exists some vertex $v \in V(G) \backslash N[C]$ such that the robber is safe on v in the first round.

The argument goes by contradiction: assume that C is dominating. Let $w \in V(G) \backslash C$ be arbitrary; note that such a w exists because $|C| = d - 1 = \delta(G) - 1 < |V(G)|$. Let $X = N(w) \cap C$ be the neighbors of w lying in C and let $Y = N(w) \backslash C$ be the neighbors of w outside of C. Note that $X \cup Y = N(w)$ and $X \cap Y = \varnothing$. Writing $|X| = x$ and $|Y| = y$, we therefore find

$$d \leqslant |N(w)| = |X \cup Y| = |X| + |Y| = x + y.$$

Next, we make three observations, clarified by Figure 5.1.1. First, as C is a dominating set by assumption, every vertex in Y must be connected to some vertex in C. Second, no vertex in X is connected to a vertex in Y, as this would together with the vertex w result in a triangle, which is impossible as $g(G) \geqslant 5$. Third, no two distinct vertices in Y can have a common neighbor, as this would similarly give a 4-cycle, which is also impossible.

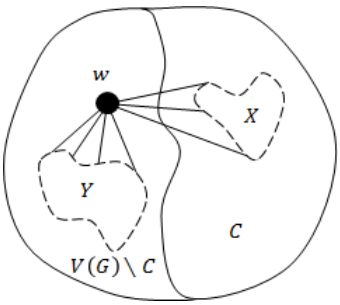

Figure 5.1.1: A sketch of G.

From these three statements, we conclude that every vertex in Y must be connected to a unique vertex in $C \backslash X$. Hence

$$d - 1 = |C| = |X| + |C \backslash X| \geqslant |X| + |Y| = x + y \geqslant d,$$

a contradiction. So C is not dominating and there is some safe vertex for R in round 0.

The rest of the proof now easily follows by induction. Assume that the robber ends round t on a vertex not adjacent to any cop, which is true for $t = 0$ by the above reasoning. The robber is then safe during the cops' next move. Now, note that after this move in round $t + 1$, each of the cops cannot be adjacent to more than one vertex in $N(R)$. Indeed, if this would be the case, G would contain a triangle or 4-cycle. Therefore, as there are only $d - 1$ cops playing, only $d - 1$ of the vertices in $N(R)$ are occupied by or adjacent to a cop when it is the robber's turn to move in round $t + 1$. Now, $\delta(G) = d > d - 1$, so there is at least one vertex in $N(R)$ which is not occupied by or adjacent to any cop. By moving there, the robber is safe for another round. □

Example 5.1.2. Consider a cycle of length at least 5. The condition of Theorem 5.1.1 is then trivially satisfied, so the cop number of the cycle is at least equal to its minimum degree. Every vertex having two neighbors, we thus conclude that $c(G) \geqslant 2$, which we already found by direct reasoning in Example 2.4.2.

While the above example is rather uninteresting, in the next sections we will see that the statement of the theorem is actually very powerful, also for more complicated graphs.

5.2 General graphs

In this section, we will answer the question whether graphs with a high cop number actually exist. As one would expect after having seen the theorem in the previous section, the answer is yes. To find graphs with arbitrarily high cop number, we need some construction that yields graphs with girth at least five and any minimum degree. Here, we will discuss two such constructions.

The first construction is based on a theorem from [1]. To explain it in detail, we need the notion of graph colorings.

Definition 5.2.1. Let $G = (V, E)$ be a graph and $k \in \mathbb{N}$. A *(proper) k-coloring* of G is a map $\chi \colon V \to [k]$ such that if $vw \in E$, then $\chi(v) \neq \chi(w)$. If a proper k-coloring exists, then G is said to be *k-colorable*.

Theorem 5.2.2 ([1])**.** *Let $k \in \mathbb{N}$. Then there exists a graph G such that $c(G) \geqslant k$.*

Proof. We will prove by induction on k that we can construct a 3-colorable graph G_k with $\delta(G_k) = \Delta(G_k) = k$ and $g(G_k) \geqslant 5$ for every k. The 3-colorability will be necessary to show that our graph has no 3-cycles nor 4-cycles. For $k = 1$, we can take K_2. For $k = 2$, we can use any cycle of length at least 5, which are indeed 3-colorable: assign the colors 1 and 2 alternatingly to each vertex, coloring the last vertex in the cycle with the third color if the length of the cycle is odd.

Now, assume inductively that we have constructed a graph G_k with the desired properties. We design G_{k+1} as follows: take four copies $G_k^1, \ldots, G_k^4$ of G_k. Color each of the copies in the same way with three colors. Next, connect every vertex having color 1 in G_k^1 to its copy in G_k^2, every vertex having color 2 to its copy in G_k^3 and every vertex having color 3 to its copy in G_k^4. The other vertices in the other copies are connected in a likewise manner, shown in Figure 5.2.1.

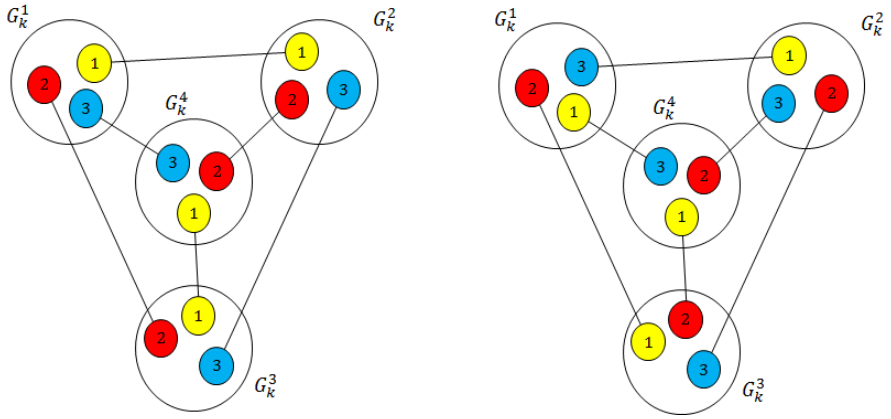

Figure 5.2.1: A sketch of the construction of G_{k+1} before and after swapping colors.

Next, switch the colors 1 and 3 in G_k^1, swap the colors 2 and 3 in G_k^2 and exchange the colors 1 and 2 in G_k^3. Note that the resulting coloring is indeed a proper 3-coloring of the graph G_{k+1}. Furthermore, exactly one neighbor has been added to every one of the vertices in a copy of G_k, so $\delta(G_{k+1}) = \Delta(G_{k+1}) = k + 1$. Finally, note that the addition of the edges between the copies of G_k does not introduce any cycles of length less than 6, because the vertices with the same color are not connected among themselves, so $g(G_{k+1}) \geqslant 5$ also still holds.

Hence, a graph G_k with the required properties indeed exists for every $k \in \mathbb{N}$. A direct application of Theorem 5.1.1 consequently shows that $c(G_k) \geqslant k$ for every k, which proves the theorem. □

This concludes the first construction showing that there exist graphs with arbitrarily high cop number. The second construction, based on the treatment in Section 1 of [9], slightly strengthens this result, showing that we may in fact require our graph with high cop number to be *bipartite*. The construction is based on the concept of a projective plane; we start with a definition.

Definition 5.2.3. A *projective plane* $P = (S, L)$ consists of a set of points S and a set of lines L such that the following three properties hold:

(1) For every pair of distinct points $p, q \in S$, there is exactly one line $\ell \in L$ incident with p and q.

(2) For every pair of distinct lines $\ell, k \in L$, there is exactly one point $p \in S$ incident with ℓ and k.

(3) There are four points in S such that no line in L is indicent with more than two of them.

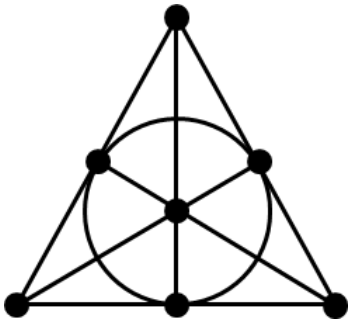

Figure 5.2.2: The Fano plane.

Example 5.2.4. An example of a projective plane is the *Fano plane* shown in Figure 5.2.2. The plane consists of seven points connected by seven lines, drawn as either line segments or a circle in the figure. It is easily seen that the first two properties of the definition of a projective plane are satisfied. To check the validity of property (3), consider the set of four points formed by the three angles of the triangle and the center point and note that no line intersects more than two of these points.

By an algebraic construction, the details of which we will not dwell upon here, one can construct from any finite field of q elements a projective plane of order q. From this construction, some nice properties of the planes readily follow. We collect these in the following lemma, the proof of which may be found in, e.g., [10].

Lemma 5.2.5 ([10])**.** *Let $P = (S, L)$ be a projective plane. There exists some $q \in \mathbb{N}$ such that the following hold:*

(i) Any point in S lies on exactly $q + 1$ lines in L.

(ii) Any line in L intersects exactly $q + 1$ points from L.

(iii) There are $q^2 + q + 1$ lines, i.e., $|L| = q^2 + q + 1$.

We call q the order *of P.*

The Fano plane in the example above has order $q = 2$. Note that, as finite fields exist for every q a prime power and there are infinitely many prime numbers, it follows that there are infinitely many projective planes, of arbitrarily high order. Remains to construct for every projective plane a corresponding graph having high cop number.

Theorem 5.2.6 ([9],[34])**.** *Let $k \in \mathbb{N}$. Then there exists a bipartite graph G such that $c(G) \geqslant k$.*

Proof. Let q be the smallest prime power greater than or equal to k and let $P = (S, L)$ be a projective plane of order q. We define a bipartite graph $G = (V_1 \cup V_2, E)$ by setting $V_1 = S$, $V_2 = L$ and

$$E = \{p\ell \in S \times L \mid p \text{ and } \ell \text{ are incident}\}.$$

By construction, G is indeed bipartite. Moreover, by Lemma 5.2.5, $d(v) = q+1$ for every $v \in V_1 \cup V_2$, so $\delta(G) = q+1$. Furthermore, note that G does not contain cycles of length 4. Indeed, such a cycle of the form $(p_1, \ell_1, p_2, \ell_2)$ with $p_1, p_2 \in S$, $\ell_1, \ell_2 \in L$ would imply that both ℓ_1 and ℓ_2 are incident with p_1 and p_2, which contradicts both properties (1) and (2) of a projective plane. Hence, G being bipartite, $g(G) \geqslant 6$. By Theorem 5.1.1, it therefore follows that $c(G) \geqslant q+1$. The statment follows. □

Note that in the proof of Theorem 5.2.6 it is enough to show that our constructed graph G has cop number at least $q+1$. In fact, we can even show that equality holds, following a proof in [34], which presents a winning strategy for $q+1$ cops.

To this end, let $q+1$ cops play the game on G, starting by occupying a set of points $C_1, \ldots, C_{q+1} \in V_1$. Suppose first that the robber chooses to occupy some line $w \in V_2$ in round 0. By Lemma 5.2.5.ii, R has $q+1$ neighbors $v_1, \ldots, v_{q+1} \in V_1$. By property (1) of a projective plane, each of these points v_i has an adjacent line $w_i \in V_2$ which is also adjacent to C_i (viewing v_i and C_i as a pair of distinct points which must share an incident line, which we call w_i). Therefore, every cop may move from C_i to w_i in round 1 so that every v_i is adjacent to C_i after this move, i.e., $N[R] \subseteq N[C]$ and the robber is caught in the next round.

The other option is that the robber occupies a point $v \in V_1$ at the start of the game. Again by property (1) of projective planes, there is some line $w_1 \in V_2$ connected to both v and C_1 so that C_1 may move here in round 1. The robber must then move to a line $w \in V_2$, at which point we are back in the previous situation but with one cop already in position. Hence indeed $c(G) = q+1$ holds.

We have thus found that graphs with arbitrarily high cop number indeed exist. But is the cop number bounded in some way in terms of the size of the graph? This question gives rise to perhaps the deepest conjecture in the field of cops and robbers, known as *Meyniel's conjecture*. The conjecture is mentioned in a paper by Frankl [18], where it is contributed to Henri Meyniel. The statement is as follows.

Conjecture 5.2.7 ([18])**.** *Let $c(n)$ be the maximal cop number among all connected graphs of n vertices. Then $c(n) = O(\sqrt{n})$.*

Note that the graphs constructed from the projective planes in the proof of Theorem 5.2.6 indeed adhere to the conjecture: every such graph has $2(q^2+q+1)$ vertices and cop number equal to $q+1$, so indeed $c(G) = O(\sqrt{|V(G)|})$ holds. While examples as these reinforce the conjecture, they do not provide a proof. Indeed, the conjecture remains wide open. However, progress is gradually made. The best bound found so far is given in the following theorem and was proven independently by three sets of authors.

Theorem 5.2.8 ([19, 31, 38])**.** *Let $c(n)$ be the maximal cop number among all connected graphs of n vertices. Then*

$$c(n) \leqslant O\left(\frac{n}{2^{(1-o(1))\sqrt{\log_2 n}}}\right).$$

All proofs of the theorem use the probabilistic method, a useful tool in combinatorics to show bounds on asymptotic behaviour.

5.3 Unit disk graphs

Whereas for general graphs it is possible to construct examples with arbitrarily high cop number, we have seen that this is not the case for the class of unit disk graphs, which

always have a cop number no higher than 9. In this section, we explore how high we can actually get. We start with an instance of a unit disk graph having cop number 3, courtesy of [7].

Example 5.3.1 ([7])**.** The graph G in question has 1440 vertices. We describe its unit disk representation $\mathcal{C}$, which uniquely determines G. It is easier to use polar coordinates $(r : \theta)$ rather than Cartesian coordinates (x, y) to define the centres of the unit disks, where r is the distance from the origin and θ the angle in degrees. Furthermore, we work with the alternative definition of unit disk graphs.

For every $\theta \in [360]$, we place a unit disk centered at $(55 : \theta)$ and one centered at $(57 : \theta + \frac{1}{2})$, forming two circles of 360 vertices each. Inside the so-formed annulus, vertices are placed at $(55 : 2\theta)$, $(56.35 : 2\theta + 0.5)$, $(55.85 : 2\theta + 1)$ and $(56 : 2\theta + 1.5)$ for $\theta \in [180]$, accounting for another 720 vertices. Part of G formed in this way is shown in Figure 5.3.1.

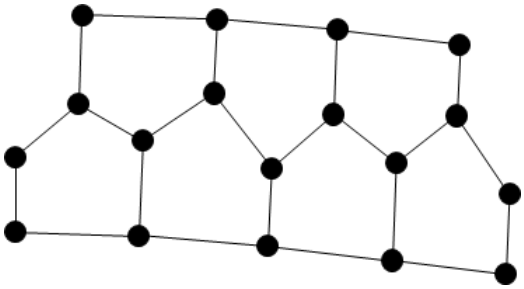

Figure 5.3.1: Part of G.

It takes some calculations to show that the graph shown above is indeed the one described by the given representation. Using the triangles shown in Figure 5.3.2, for example, we find by the law of cosines that two adjacent unit disk centres on the outer circle of radius 57 are at distance

$$c = \sqrt{a^2 + b^2 - 2ab\cos\theta} \approx 0.995$$

from each other, so the corresponding vertices are connected. Likewise, two unit centers on the inner circle of radius 55 are a distance of 0.960 apart. Similar computations show that other pairs of vertices are either connected or disconnected as shown.

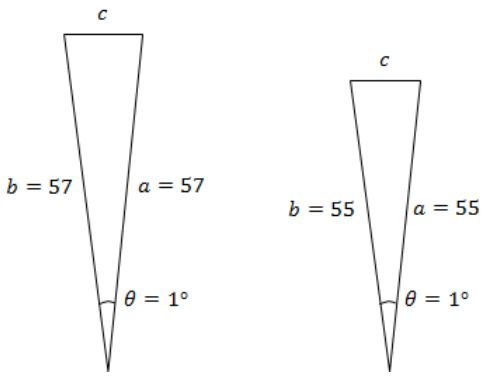

Figure 5.3.2: Triangles used in the calculation.

Now, it is clear that G has girth 5 (consisting of pentagons) and all vertices in G have degree 3. Therefore, by Theorem 5.1.1, we find that $c(G) \geqslant 3$. Furthermore, as the

drawing given in Figure 5.3.1 is also planar, Theorem 4.1.6 yields that $c(G) \leqslant 3$. We thus conclude that $c(G) = 3$.

The next logical step in search of a unit disk graph with higher cop number would be to seek a unit disk graph with girth at least 5 and minimum degree 4. However, unfortunately, such a graph does not exist.

Proposition 5.3.2. *There does not exist a unit disk graph G with $g(G) \geqslant 5$ and $\delta(G) \geqslant 4$.*

Proof. The reasoning will be similar to the proof of Proposition 4.2.1. Let G be a unit disk graph with $g(G) \geqslant 5$ and $\delta(G) \geqslant 4$. By Theorem 5.1.1, we find that $c(G) \geqslant 4$. Therefore, by Theorem 4.1.6, G is non-planar. In particular, the canonincal drawing of G contains a crossing. Consider such a crossing, sketched in Figure 5.3.3.

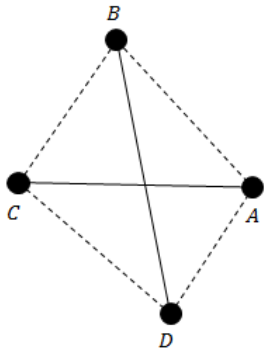

Figure 5.3.3: Sketch of a crossing in the canonical drawing of G.

Note that both diagonals of the quadrilateral shown must be of length at most 1, as we are looking at the canonincal drawing. Suppose for the sake of contradiction that all sides of the quadrilateral are of length more than 1 and consider e.g. the triangle $\triangle ABC$. Assume that $\angle ABC \geqslant \frac{\pi}{2}$. By the law of cosines it then follows that

$$1 \geqslant |AC|^2 = |AB|^2 + |BC|^2 - 2|AB||BC| \cos \angle ABC > 1 + 1 = 2,$$

which is nonsense. Therefore, $\angle ABC < \frac{\pi}{2}$. By symmetry, it then follows that the other three angles of the quadrilateral are also smaller than $\frac{\pi}{2}$. But then the four angles add up to less than 2π, which is a contradiction.

Hence, there must be at least one side of the quadrilateral which is of length at most 1. Suppose without loss of generality that $|AB| \leqslant 1$. Now, if there is another side of the quadrilateral that has length at most 1, we have a triangle or 4-cycle in the graph and we are done. Hence, suppose all other sides are larger than 1. By the reasoning above, at least one of the angles $\angle ABC$ and $\angle BAD$ must be larger than $\frac{\pi}{2}$. Say $\angle ABC \geqslant \frac{\pi}{2}$. Then, again using the law of cosines, we find that

$$1 \geqslant |AC|^2 = |AB|^2 + |BC|^2 - 2|AB||BC| \cos \angle ABC > 1,$$

which is again a contradiction. Hence there must be another side of the quadrilateral which is of length at most 1. But then we have either a triangle or 4-cycle in our graph G, which contradicts the fact that $g(G) \geqslant 5$. □

As Theorem 5.1.1 does not provide an "if and only if" statement, we cannot conclude that any unit disk graph must have $c(G) \leqslant 3$. However, as there is little more in the field of lower bounds for the cop number of a graph, the search for a unit disk graph with $c(G) \geqslant 4$ grinds to a halt here.

5.4 String graphs

Note that the graph in Example 5.3.1 from the previous section is a unit disk graph and therewith a string graph by Proposition 2.2.6. Therefore, this example provides us with a string graph having cop number three. With the upper bound on string graphs established at 15 by Theorem 4.3.8, this gives a considerable gap. Unfortunately, in this section we will see that it is not easy to find a string graph with cop number four or larger to reduce this gap from the bottom side.

Inspired by the statement of Theorem 5.1.1, it is intuitive to try and search for a string graph having girth at least 5 and minimum degree 4. Simply adding edges to the graph from Example 5.3.1 quickly leads to difficulties in finding a suitable string representation for the graph. Therefore, we turn to another candidate, which is the so-called *Robertson graph*.

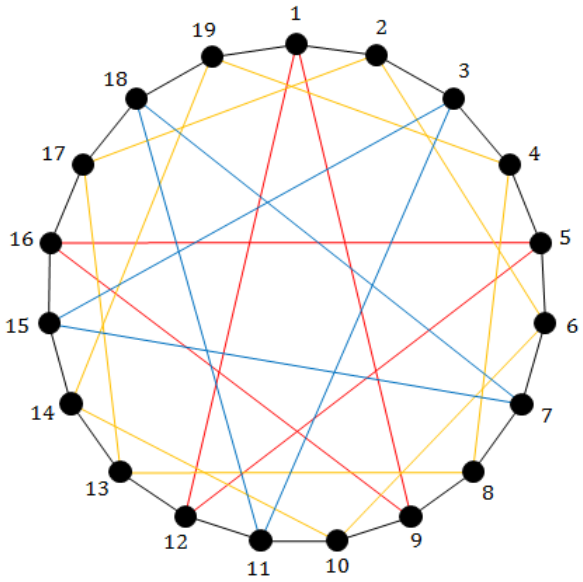

Figure 5.4.1: A drawing of the Robertson graph, adapted from [28].

Example 5.4.1. The Robertson graph [36], depicted in Figure 5.4.1, is a graph on 19 vertices which has girth 5 and in which every vertex has degree 4. It is the smallest graph having these properties, i.e., any graph with 18 or less vertices is lacking at least one of the two. By Theorem 5.1.1, the Robertson graph has cop number at least 4.

Moreover, we can construct an explicit winning strategy for 4 cops, showing that in fact the Robertson graph has cop number exactly equal to 4. Indeed, let four cops begin on the vertices 2, 5, 8 and 14. The situation is shown in Figure 5.4.2, where the positions of the cops are shown in blue and vertices connected to a cop are colored gray. It is clear that the robber must then choose to start on either vertex 11 or 18 to not be caught in the first round. We check both possibilities.

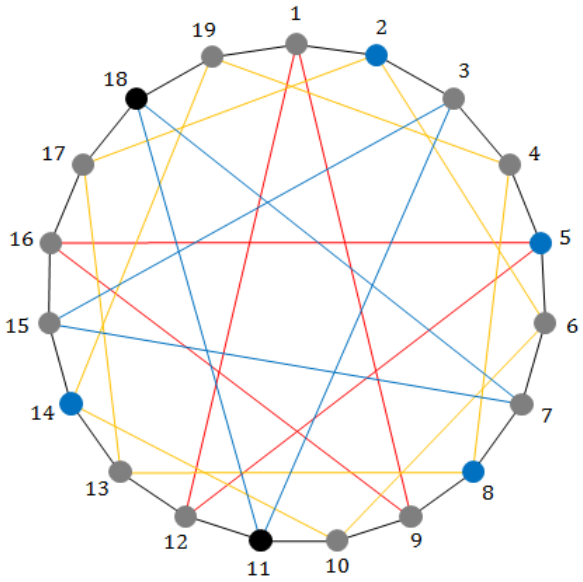

Figure 5.4.2: The four cops occupy their positions in the Robertson graph.

First suppose the robber chooses to start on vertex 11. We then move the cop standing on vertex 5 to occupy vertex 12, leaving all other cops in place. The resulting situation is shown in Figure 5.4.3, in which the vertex occupied by the robber is colored red. Now, if the robber would remain on vertex 11 in the next round, he would be caught by the cop now on vertex 12. Therefore, the only possible move to avoid capture is to move to vertex 18.

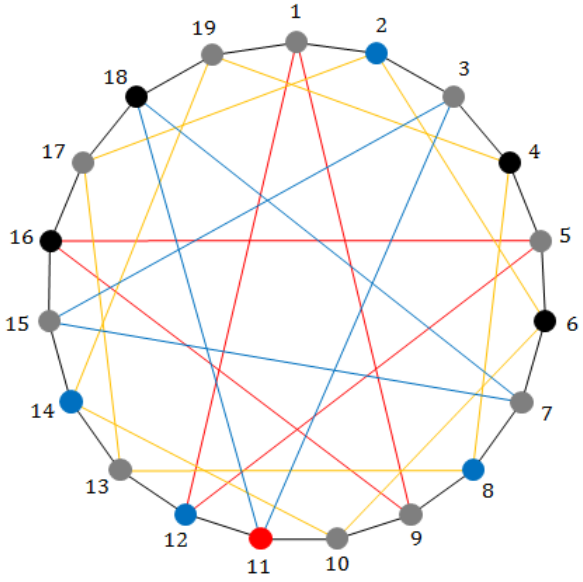

Figure 5.4.3: The situation after one move of the cops.

On the cops' turn, now the cop occupying vertex 2 moves to vertex 17, while the ohter

cops remain stationary. The result is shown in Figure 5.4.4. The robber now has nowhere left to go and is captured by the cops in the next round.

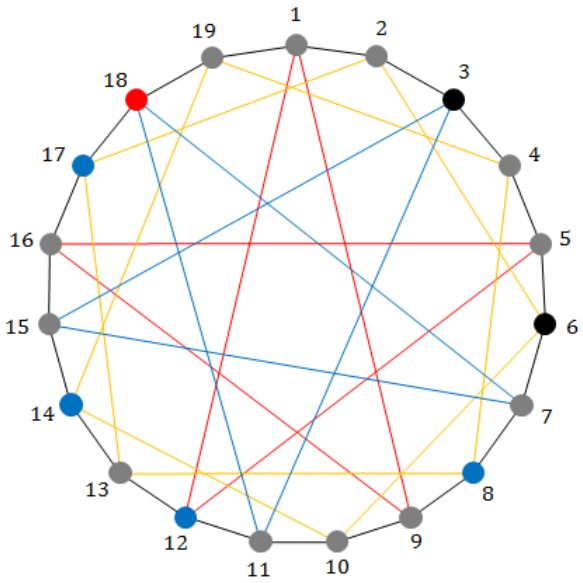

Figure 5.4.4: The robber has no safe place to go.

Next suppose the robber starts on vertex 18. Then in the first move of the cops, we move the cop from vertex 2 to vertex 17, resulting in Figure 5.4.5. The robber is then forced to move to vertex 11.

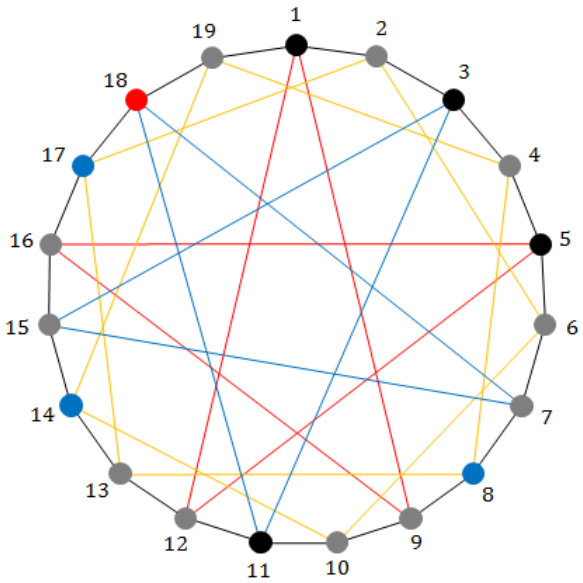

Figure 5.4.5: The situation if the robber started on vertex 18.

Consequently, the cop occupying vertex 8 moves to stand on vertex 4, while the other cops do not move. The result is the situation depicted in Figure 5.4.6, from which it

becomes clear that also in this situation the robber cannot move to a safe vertex. If the robber remains stationary during his turn, the cop on vertex 14 can move to vertex 10, capturing the robber in the next turn. This shows that indeed four cops are sufficient to always capture the robber in the Robertson graph, fixing its cop number at 4.

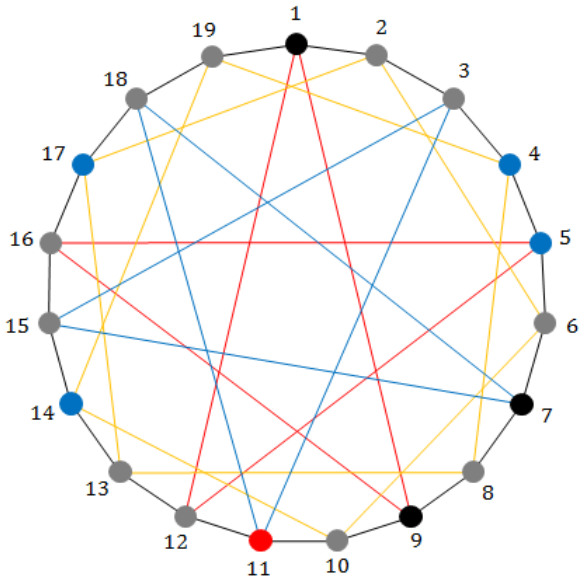

Figure 5.4.6: Again, the robber faces imminent capture.

If the Robertson graph were a string graph, it would provide us with a suitable example somewhat closing the gap between the example in the previous paragraph exhibiting a cop number of 3 and the currently best known upper bound of 15. However, in this section we conjecture that it is not.

Conjecture 5.4.2. *The Robertson graph as defined in Example 5.4.1 is not a string graph.*

We will sketch an idea that could perhaps be extended to a full proof. Our idea is to use the Jordan Curve Theorem (Theorem 4.3.5) to show that there is no way of configuring the 19 strings that should represent the Robertson graph without two strings crossing that do not represent adjacent edges in the graph. To do so, we add strings or intersections of strings to the representation one by one. Often, there are different non-equivalent ways to add a string, in which case we simply choose one of the different options. We continue adding strings in this way until we discover that two strings still need to intersect somewhere but are separated by a set of strings that they cannot cross forming a Jordan curve. When this occurs, we backtrack until the last decision we made, and try another choice in that situation.

Note that this process fixes the order in which the strings are added. However, if a string representation of the Robertson graph exists, this representation can be constructed by adding the strings to the representation in any order. Therefore, fixing the order in which the strings are added is allowed, so, if we exhaust all possibilities during the process without coming to a complete string representation, we prove that such a representation does not exist.

In trying out all the different available options, it turns out that using circular inversion (Lemma 4.1.5) can greatly reduce the amount of possibilities to be considered. However,

the amount of choices available is still very large. We exhibit our idea by showing the details of some first possible steps.

We start by noting that in the Robertson graph, which we will call G from now on, the vertices $(3, 11, 18, 7, 15)$ form a cycle, C say. Therefore, in any string representation of G, these strings need to intersect in this order. We may thus start by drawing these five strings as shown in Figure 5.4.7.a.

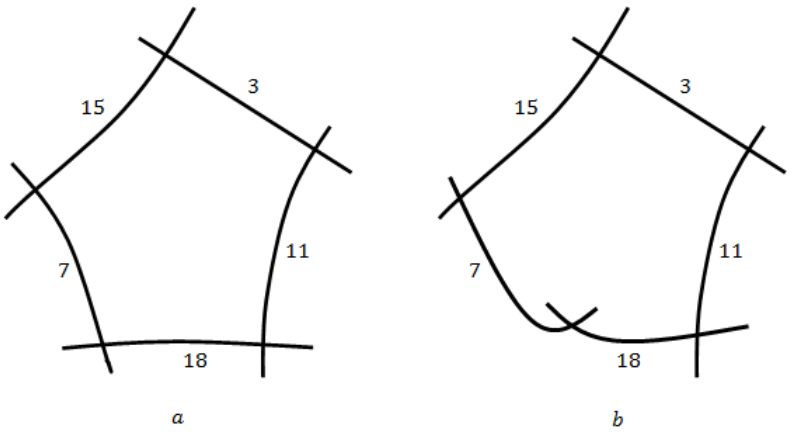

Figure 5.4.7: The cycle $C = (3, 11, 18, 7, 15)$.

However, note that the direction in which the strings intersect each other matters. The string representation shown in Figure 5.4.7.b also depicts the strings representing the vertices in C, but is not topologically equivalent to Figure 5.4.7.a. This fact already faces us with a multitude of choices for drawing the string representation of C. We start by checking the option sketched in Figure 5.4.7.a.

Now, note that vertex 1 is not adjacent to any of the vertices in C. Therefore, in a string representation of G, string 1 should not intersect any of the strings in C. By the Jordan Curve Theorem, it must therefore lie entirely in the interior or in the exterior of the simple closed curve formed by the strings representing C. By circular inversion, we may w.l.o.g. assume that string 1 lies in the interior.

Next, consider string 2. As vertex 2 is adjacent to both vertices 1 and 3, this string should intersect with both strings 1 and 3 in the string representation of G. Note that there is only one distinct way of drawing string 2 in this manner without creating unnecessary intersections. The resulting configuration is shown in Figure 5.4.8.

Next, consider vertex 9. Like vertex 1, this vertex is not connected to any of the vertices in C. Furthermore, it is connected to vertex 1. Therefore, the string representing vertex 9 must intersect 1, but may not intersect any of the strings representing C. In the current configuration, there are two non-equivalent choices to add the intersection between strings 1 and 9: either to the left or to the right of the intersection between strings 1 and 2. We choose to add string 9 to the right of string 2, and save the other choice for checking later. See Figure 5.4.9.

Next, consider string 19, which must intersect strings 1 and 18, because vertex 19 is adjacent to vertices 1 and 18 in G. There are three different places in which string 19 can intersect string 1, divided by strings 2 and 9. Moreover, the leftmost and rightmost segments of string 1 can be intersected in two directions. We choose to add string 19 as shown in Figure 5.4.10.

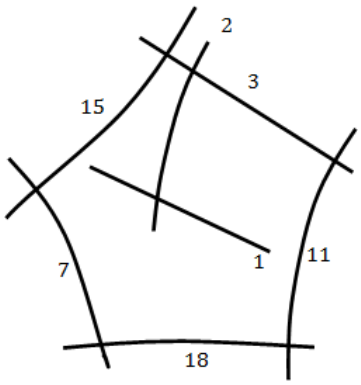

Figure 5.4.8: Strings 1 and 2 have been added.

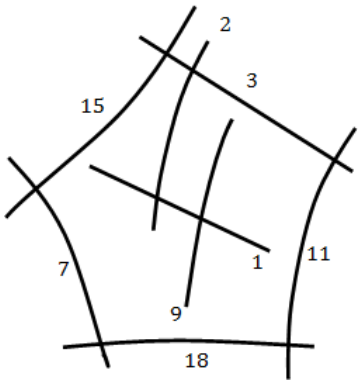

Figure 5.4.9: One choice for adding string 9.

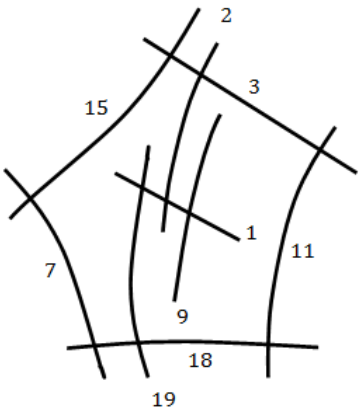

Figure 5.4.10: One choice for adding string 19.

Next, we consider string 8, which needs to intersect strings 7 and 9. In the current configuration, string 9 can be intersected in two different spots: either above or below the intersection with string 1. For both options, string 7 needs to extend around string 19 and cross string 18 again to enter the interior of C. See Figure 5.4.11 for the example choice that we will pursue.

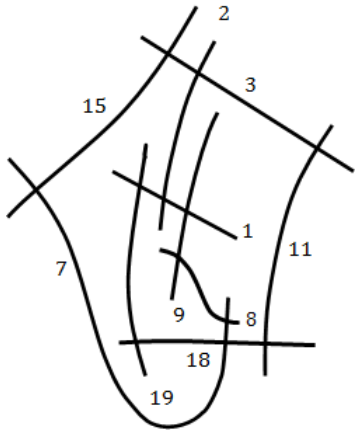

Figure 5.4.11: One choice for adding string 8.

Now, note that as we have created another intersection between strings 7 and 18, we may remove the first intersection to create more freedom. However, in doing so, we actually stumble upon a situation which we could have reached if we chose to start from the situation sketched in Figure 5.4.7.b instead of 5.4.7.a. Therefore, if we check this situation now, we may remove the situation in Figure 5.4.7.b from the list of possible choices that needs to be checked.

Furthermore, we note that instead of extending string 7 through string 18 in order to add string 8, we could also have looped string 9 around string 2 and extended it through string 1 to create a place where strings 8 and 9 could intersect. Again, by doing so, if we remove the formerly placed intersection between strings 1 and 9, we reach a situation which we already saw before, where the intersection between strings 1 and 9 would have been placed to the left of string 2. This option may then be removed from the list of options that still is to be considered.

We next consider string 10, which needs to intersect both strings 9 and 11. In the current configuration, there are only three non-equivalent choices: we place the intersection between strings 9 and 10 either above or below the intersection between strings 1 and 9, and, if placed above, the intersection can be done in two directions. We pick one of the options, drawn in Figure 5.4.12, and continue.

Next, we consider string 16, which needs to intersect the strings 9 and 15. To achieve this, we can either extend string 9 through 1 (like discussed before) or string 15 through 3. We choose the latter and pick the top segment of string 9 for the intersection, as illustrated by Figure 5.4.13.

Now, consider string 12, which needs to intersect the strings 1 and 11. Note that in the current situation, the only place where this string can be added without extending 1, is in the area where the rightmost segment of string 1 lives, colored blue in Figure 5.4.13. Indeed, the rest of string 1 is shielded from string 11 by the strings 16, 15, 7 and 8, which all cannot be passed by strings 1 nor 11. One of the two choices for the orientation of string 12 is shown in Figure 5.4.14.

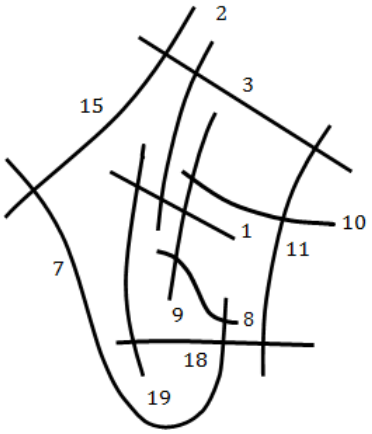

Figure 5.4.12: One choice for adding string 10.

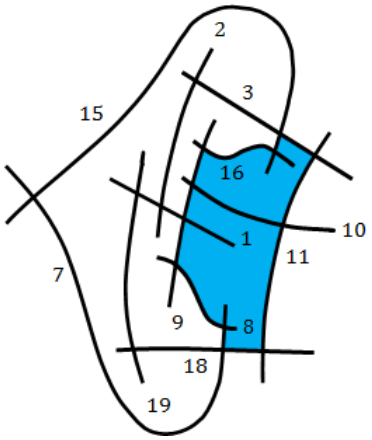

Figure 5.4.13: One choice for adding string 16. String 12 must live in the blue area.

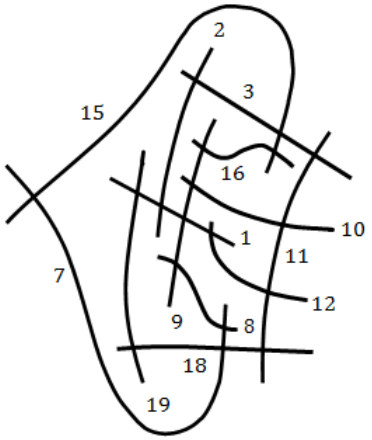

Figure 5.4.14: One choice for adding string 12.

Next, consider string 6, which must intersect the strings 2 and 7. In the current situation, this completely fixes the intersection points between string 6 and 2 and between string 7 and 2. The result is shown in Figure 5.4.15.a. Now, note that string 6 must also intersect string 10. For this, there are multiple possibilities, either in the interior or the exterior of C. We choose an option in the exterior, shown in Figure 5.4.15.b. Note that, by circular inversion, this option is equivalent to the option where string 10 passes around C around the top insead of around the bottom.

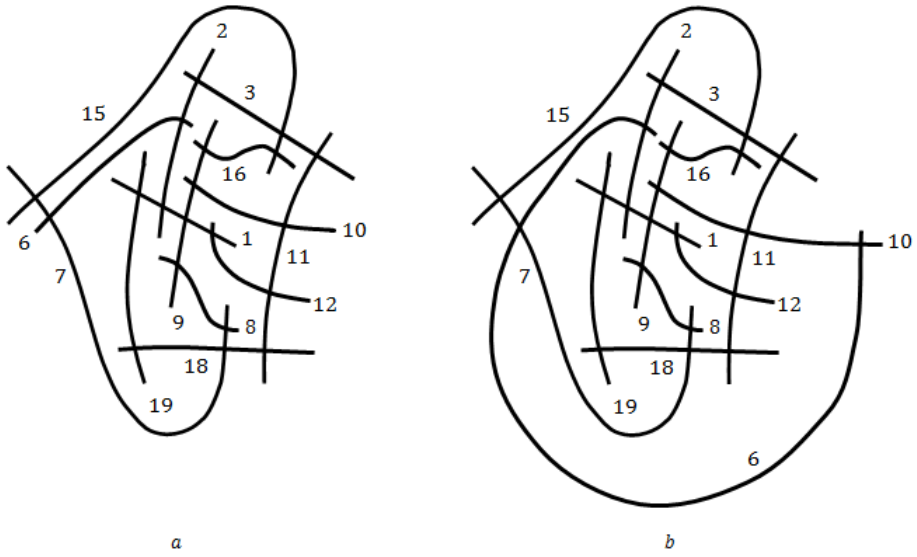

Figure 5.4.15: Adding string 6.

Now, consider string 4 which must connect strings 3 and 19. There are quite a few possibilities to achieve this. One option is to extend string 3 through string 2, which is shown in Figure 5.4.16.a. Next, string 5 must intersect string 4 somewhere, for example like in Figure 5.4.16.b. However, we now face ourselves with a problem. By the Jordan Curve Theorem, string 5 is confined to the area shaded blue in Figure 5.4.16.b, and string 16 is confined to the area shaded red in this figure. The intersection between strings 5 and 16 can therefore not be added in this configuration.

Having stumbled upon an impossibility, we trace back our steps to the last moment in which we made a choice. In this case, this was the placement of string 5. However, one easily checks that no matter where string 5 intersects string 4, the same problem occurs. Therefore, we go back further, reaching the choice of the placement of string 4. The next step would now be to try a different placement of this string and check whether this solves the problem. We continue in this fashion until we have tried all possible ways of drawing string 4, after which we backtrack further to the decision made before. In this manner, we systematically check all possible ways of adding all strings.

This concludes the sketch of our idea for a proof of the conjecture. Seeing that the process generates hundreds of situations to be checked, it would perhaps be convenient to try and use a computer to do this checking. As the application of circular inversion and checking whether we are repeating another situation reduce the amount of work to be done greatly, it would seem wise to implement these elements into the program to be used.

Besides the proof method detailed here, one could also try and search for a proof using a vastly different approach. This could lean, for example, on results known for drawings

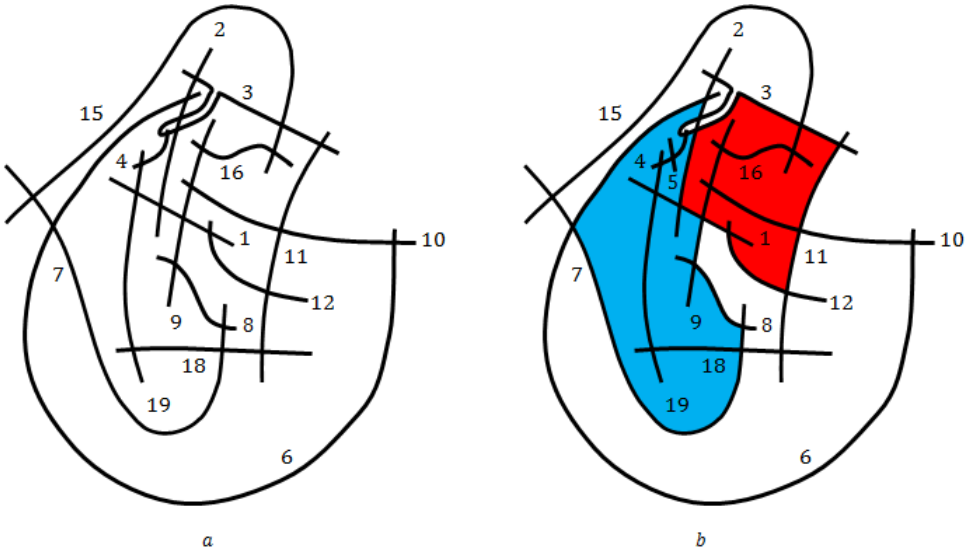

Figure 5.4.16: Adding strings 4 and 5.

of graphs. An application of the crossing number inequality [2] has unfortunately not shown succesful so far. Another direction in which one might look for a solution is that of (algebraic) topology. Perhaps a theorem from this field yields surprising insights on the matter.

If the conjecture indeed turns out to be true, it shows that perhaps the most convenient graph having girth 5 and minimum degree 4 is not an example of a string graph. However, it does not prove that such a graph does not exist. As the proof method outlined in this section heavily relies on the structure of the Robertson graph, it may be hard to generalise.

We conclude by noting that even the proof of such a stronger statement would not disprove the existence of a string graph with cop number 4 or higher; it merely rules out the use of the "beaten track", Theorem 5.1.1. Having little more to rely on to prove lower bounds on the cop number, one would need to develop vastly different methods to make progress.

6 Conclusions and further research

In this thesis, we give a detailed overview of a variety of results on the Game of Cops and Robbers. In particular, we categorise several results proving the existence of an upper bound on the cop number for certain types of geometric graphs, as well as some results giving examples of geometric graphs having a high cop number. In this categorization, it becomes clear that these results are all based on the statement or the proof of one of two main theorems.

For the first type of results, the core theorem is Proposition 4.1.3, stating that a shortest path in a graph can be guarded by a single cop. By repeatedly using this statement to gradually decrease set of vertices in the graph in which the robber can freely move, a handful of cops can capture the robber without fail. By proving that some other structure in a graph can be guarded or otherwise defended by a set number of cops and mimicking this argument, upper bounds for several types of graphs follow.

The construction of the graphs with a high cop number all rely on the statement of Theorem 5.1.1, which proves that a graph having girth at least five must have cop number at least equal to its minimum degree. Using this theorem, one can show that, if no restrictions are imposed, graphs with arbitrarily high cop number exist. Furthermore, it yields some examples of geometric graphs with a higher cop number.

In an attempt to extend the known results, we explore whether the two main theorems stated above can be used further. Unfortunately, it turns out that this is not straightforward. In Example 4.2.3, we show that the existing upper bound for unit disk graphs cannot be sharpened using only Proposition 4.1.3 and its direct corollaries. Similarly, Theorem 5.3.2 shows that Theorem 5.1.1 cannot yield a unit disk graph having a higher cop number than known so far and in Section 5.4, we conjecture that the same holds for string graphs.

These results all being negative seems to suggest that perhaps a vastly different approach is necessary to book much progress. For future research, it would therefore be very interesting to see if one could find other ways than results based on Proposition 4.1.3 and Theorem 5.1.1 to prove upper or lower bounds on the cop number for a given class of graphs. This would require thinking outside of the box sketched by the proofs of Aigner and Fromme [1].

To close off, while not having been the main focus of study in this thesis, we would like to mention two more active areas within the research field of the Game of Cops and Robbers as interesting directions for further research. First, there is the study of asymptotic behaviour of the cop number. In this area, Meyniel's conjecture [18] mentioned in Section 5.2 stands as general statement to prove or disprove. Second, work is also being done on the cop number of random graphs, for example in [7] and [32].

References

[1] M. Aigner, M. Fromme, A game of cops and robbers, Discrete Applied Mathematics 8, 1984, p. 1–11.

[2] M. Ajtai, V. Chvátal, M.M. Newborn, E. Szemerédi, , Crossing-free subgraphs, Theory and practice of combinatorics, North-Holland Mathematics Studies 60, 1982, p. 9-12.

[3] Anonymous authors, Ramsey's Theorem, Wikipedia, `https://en.wikipedia.org/wiki/Ramsey%27s_theorem`, retrieved 01/09/2017.

[4] S. Arnborg, D. Corneil, A. Proskurowski, Complexity of finding embeddings in a k-tree, SIAM Journal on Matrix Analysis and Applications, 8 (2), 1987, p. 277-284.

[5] S. Arora, B. Boaz, Computational Complexity: A Modern Approach, 2009, Cambridge.

[6] A. Berarducci, B. Intriglia, On the cop number of a graph, Advances in Applied Mathematics 14, 1993, p. 389–403.

[7] A. Beveridge, A. Dudek, A. Frieze, T. Müller, Cops and robbers on geometric graphs, Combinatorics, Probability and Computing, 21 (6), 2012, p. 816–834.

[8] A. Bonato, E. Chiniforooshan, Pursuit and Evasion from a Distance: Algorithms and Bounds, Proceedings of Workshop on Analytic Algorithms and Combinatorics ANALCO'09, 2009.

[9] A. Bonato, R.J. Nowakowski, The Game of Cops and Robbers on Graphs, American Mathematical Society, 2011.

[10] P.J. Cameron, Combinatorics: Topics, Techniques, Algorithms, Cambridge University Press, 1995.

[11] J. Chalopin, D. Gonalves, Every planar graph is the intersection graph of segments in the plane, ACM Symposium on Theory of Computing, 2009.

[12] E. Chiniforooshan, A better bound for the cop number of general graphs, Journal of Graph Theory 58, 2008, p. 45–48.

[13] N.E. Clarke, G. MacGillivray, Characterizations of k-copwin graphs, Discrete Mathematics 312, 2012, p. 1421-1425

[14] J. H. Conway, The angel problem, Games of No Chance, MSRI Publications 29, 1996, p. 3-12.

[15] G. Ehrlich, S. Even, R.E. Tarjan, Intersection graphs of curves in the plane, Journal of Combinatorial Theory, 21 (1), 1976, p. 8-20.

[16] Picture by David Eppstein — Own work, Public Domain, `https://commons.wikimedia.org/w/index.php?curid=8265700`, retrieved 11/05/2017.

[17] F.V. Fomin, P.A. Golovach, J. Kratochvìl, N. Nisse, Pursuing fast robber in graphs, Theoretical computer science 411, 2010, p. 1167–1181.

[18] P. Frankl, Cops and robbers in graphs with large girth and Cayley graphs, Discrete Applied Mathematics 17, 1987, p. 301–305.

[19] A. Frieze, M. Krivelevich, P. Loh, Variations on Cops and Robbers, Journal of Graph Theory 69, 2010, p. 383-402.

[20] M.R. Garey, D.S. Johnson, Computers and Intractability: A Guide to the Theory of NP-completeness, W.H. Freeman and Company, 1979, p. 190.

[21] T. Gavenčiak, P. Gordinowicz, V. Jelínek, P. Klavík, J. Kratochvíl, Cops and Robbers on Intersection Graphs, ISAAC 2015.

[22] A.W. Hales, R.I. Jewett, Regularity and positional games, Transactions of the American Mathematical Society 106, 1963, p. 222-229.

[23] J. Hopcroft, R.E. Tarjan, Efficient planarity testing, Journal of the Association for Computing Machinery, 21 (4), 1974, p. 549-568.

[24] R. Isaacs, Games of Pursuit, RAND Corporation, 1951.

[25] G. Joret, M. Kamiński, D.O. Theis, The cops and robber game on graphs with forbidden (induced) subgraphs, Contributions to Discrete Mathematics 5 (2), 2010, p. 40-51.

[26] R.J. Kang, T. Müller, Sphere and dot product representations of graphs, Discrete and Computational Geometry, 47 (3), 2012, p. 548–568.

[27] W.B. Kinnersly, Cops and Robbers is EXPTIME-complete, Journal of Combinatorial Theory Series B 111, 2015, p. 201–220.

[28] Picture by Koko90 — Own work, `https://commons.wikimedia.org/wiki/File:Robertson_graph_hamiltonian.svg`, retrieved 04/08/2017.

[29] J. Kratochvil, String Graphs. II. Recognizing string graphs is NP-Hard, Journal of Combinatorial Theory, Series B, 52 (1), 1991, p. 67-78.

[30] K. Kuratowski, Sur le problème des courbes gauches en topologie, Fund. Math 15, 1930, p. 271–283.

[31] L. Lu, X. Peng, On Meyniels conjecture of the cop number, Journal of Graph Theory 71 (2), 2012, p. 192–205.

[32] T. Łuczak, P. Prałat, Chasing robbers on random graphs: Zig-Zag theorem, Random Structures and Algorithms 37, 2010, p. 516–524.

[33] R.J. Nowakowski, P. Winkler, Vertex-to-vertex pursuit in a graph, Discrete Mathematics 43, 1983, p. 235–239.

[34] P. Prałat, When does a random graph have constant cop number?, Australasian Journal of Combinatorics 46, 2010, p. 285–296.

[35] A. Quillot, Jeux et pointes fixes sur les graphes, Thèse de 3ème cycle, Université de Paris VI, 1978, p. 131–145.

[36] N. Robertson, The smallest graph of girth 5 and valency 4, Bulletin of the American Mathematical Society 70 (6), 1964, p. 824–825.

[37] M. Schaefer, E. Sedgwick, D. Štefankovič, Recognizing string graphs in NP, Journal of Computer and System Sciences, 67 (2), 2003, p. 365-380.

[38] A. Scott, B. Sudakov, A new bound for the cops and robbers problem, SIAM Journal of Discrete Mathematics 25, 2011, p. 1438-1442.

[39] H. Tverberg, A proof of the Jordan curve theorem, Bulletin of the London Mathematical Society 12 (1), 1980, p. 34-38.