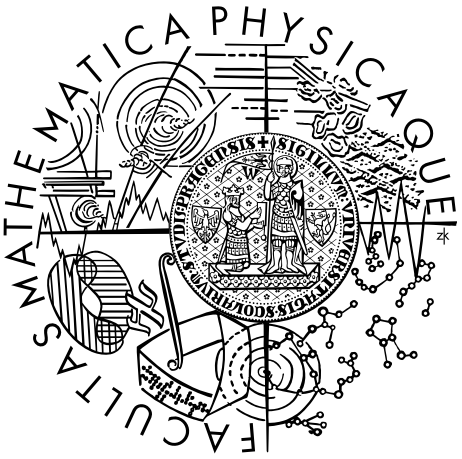
Charles University in Prague
Faculty of Mathematics and Physics

DOCTORAL THESIS



Tomáš Gavenčiak

Structural and complexity questions of graph theory

Department of Applied Mathematics

Supervisor of the thesis: prof. RNDr. Jan Kratochvíl, CSc.
Study programme: Computer Science (P1801)
Study branch: Discrete Models and Algorithms

Prague 2016

I declare that I carried out this doctoral thesis independently, and only with the cited sources, literature and other professional sources.

I understand that my work relates to the rights and obligations under the Act No. 121/2000 Sb., the Copyright Act, as amended, in particular the fact that the Charles University in Prague has the right to conclude a license agreement on the use of this work as a school work pursuant to Section 60 subsection 1 of the Copyright Act.

In ..... date ..... signature of the author

Title: Structural and complexity questions of graph theory

Title (Czech): Studium strukturálních a složitostních otázek teorie grafů

Author: Tomáš Gavenčiak

Department: Department of Applied Mathematics

Supervisor: prof. RNDr. Jan Kratochvíl, CSc., Department of Applied Mathematics

Abstract: The original *cops and robber game*, proposed in 1983 by Nowakowski and Winkler, and independently by Quilliot, is a two-player combinatorial pursuit game on a graph. Many related games have been introduced and studied since then, both with a purely game theoretic motivation and for their applications in graph theory and connections with graph width parameters.

We give an overview of the field and present three particular results: We show bounds on the required number of cops for various connected intersection graph classes, most notably we show that on connected string graphs 15 cops always win. We show the game of cops and fast robber to be polynomially decidable on interval graphs and as a tool we generalise the problem of defensive domination and show a polynomial algorithm for interval graphs. Finally we propose and examine generalisations of tree-depth, marshals and robber games, and minors to hypergraphs and hypergraph pairs.

Keywords: combinatorial games, cops and robber game, intersection graphs, computational complexity, tree-depth

Preface

Within this thesis, I present my work and results in the area of combinatorial pursuit games. The main focus is on the research problems I significantly contributed to. At the same time I aim to give the reader an overview of the area and provide context for the problems. Most of the scientific material is already published in the form of articles but this thesis is written in the form of a monograph to be more approachable.

I had the opportunity to work with different aspects of combinatorial game theory: Structural properties in the case of the original *cops and robber game* on geometric graphs, showing several strong bounds. Computational complexity in the case of the *fast robber game* and a special domination problem on interval graphs, showing a polynomial algorithm for both. And finally proposing and studying extensions of the graph parameter *tree-depth*, related games and minors to the world of hypergraphs.

The area of combinatorial games is vast and still growing in many directions, and the related graph theory even more so. In these areas, while many of the problems and research questions are only loosely coupled with one another within their areas, many tools, approaches, the general problem structure and the mindset are common. This is also the case of my research in the area and I hope to give you, the kind reader, a glimpse of it with this work.

Acknowledgements

My work on this thesis and other research projects was made possible by the great environment and team at my home department, as well as my supportive colleagues abroad. First and foremost, I greatly appreciate the guidance, support, inspiration and motivation of my adviser prof. Jan Kratochvíl. Then I would like to thank my teachers for the knowledge and ideas they gave me; my colleagues for joint work on interesting research projects and many insightful discussions; and also my students for reminding me of any gaps in my understanding. To my family and friends I owe deep gratitude for the energy and moral support, necessary for any kind of creative work.

Author's contribution

The material in the thesis has been published in several papers and some are still in preparation. The following is a list of publications related to the thesis with my significant contribution.

- T. Gavenčiak, V. Jelínek, P. Klavík, J. Kratochvíl:
  Cops and Robbers on Intersection Graphs,
  published in proceedings of ISAAC 2013 [GJKK13]
  Results of Sections 2.4, 2.5 and 2.6.

- T. Gavenčiak, P. Gordinowicz, V. Jelínek, P. Klavík, J. Kratochvíl:
  Cops and Robbers on String Graphs,
  published in proceedings of ISAAC 2015 [GGJ$^+$15]
  Results of Sections 2.6 and 2.7.

- T. Gavenčiak:
  Catching a fast robber on interval graphs,
  published in proceedings of TAMC 2011 [Gav11]
  Announcement of results of Section 3.4.

- D. Dereniowski, T. Gavenčiak, J. Kratochvíl:
  Cops, a fast robber and defensive domination on interval graphs,
  submitted to JGT (2015) [DGK15]
  Detailed proofs of Section 3.4 and results of Section 3.3. Introduction of $\mathcal{A}$-defensive domination.

- I. Adler, T. Gavenčiak, T. Klimošová:
  Hypertree-depth and minors in hypergraphs,
  published in TCS 2012 [AGK12]
  Results of Sections 4.3, 4.4 and 4.6. Introduction of hypertree-depth on hypergraphs and hypergraph pairs. Introduction of hypergraph (and hypergraph-pair) minors of Sections 4.3.4 and 4.4.4.

Contents

Preface **1**

Table of contents **3**

1 Pursuit games on graphs **5**
1.1 Notation and preliminaries 5
1.1.1 Graph theory 6
1.1.2 Combinatorial game theory 8
1.2 Intersection graph classes 10
1.2.1 Interval and chordal graphs 11
1.2.2 Circle and circular arc, function and IFA graphs 12
1.2.3 String graphs and related classes 14
1.2.4 General assumptions and remarks 15
1.3 Game variants 16
1.3.1 Pursuit game framework 16
1.4 Notable results 18

2 Cops and robber games on intersection graph classes **19**
2.1 Game of Cops and robber 20
2.2 History and notable results 21
2.2.1 One cop game 21
2.2.2 Meyniel's conjecture 21
2.2.3 Planar and bounded genus graphs 21
2.2.4 Computational complexity 22
2.3 Map of classes and bounds 22
2.4 IFA, function, circle and circular arc graphs 25
2.4.1 Filaments and regions 25
2.4.2 Two cops' strategy 26
2.5 Outer string graphs 27
2.5.1 String pairs and Regions 27
2.5.2 Four cop strategy 28
2.6 String graphs 30
2.6.1 Guarding shortest paths 30
2.6.2 Guarding shortest curves 32
2.6.3 Segments, faces and regions 32
2.6.4 Restricted graphs and strategies 34
2.6.5 Proof of Theorem 19 34

2.7 Strings on surfaces . . . 36
2.7.1 Topological preliminaries . . . 36
2.7.2 Guarding non-contractible closed walks . . . 38
2.7.3 The game on higher genus surfaces . . . 39
2.8 Guarding retracts and grids . . . 40
2.8.1 Guarding d-dimensional Grids . . . 40
2.9 Remarks and open problems . . . 41

3 Cops, fast robber and defensive domination on interval graphs 43
3.1 Cops and fast robber game . . . 43
3.2 History and related results . . . 44
3.3 $\mathcal{A}$-defensive domination . . . 45
3.3.1 The problem and motivation . . . 45
3.3.2 Leftmost defense on interval graphs . . . 46
3.3.3 $\mathcal{A}$-defensive domination on interval graphs . . . 48
3.4 Game reductions and strategy . . . 50
3.4.1 Observations and notation . . . 50
3.4.2 Manoeuvres and the restricted game . . . 52
3.4.3 Equivalence of the restricted game . . . 54
3.5 Remarks and open problems . . . 59

4 Ultramonotone games and minors on hypergraphs 61
4.1 History and related research . . . 61
4.2 Tree-depth and minors of graphs . . . 62
4.2.1 Tree-depth of graphs . . . 62
4.2.2 Cops and robber game . . . 63
4.2.3 Graph minors . . . 64
4.3 Tree-depth and minors of hypergraphs . . . 64
4.3.1 Hypertree-depth . . . 64
4.3.2 Alternative characterisations of hypertree-depth . . . 66
4.3.3 Marshals and robber game . . . 66
4.3.4 Hypergraph minors . . . 67
4.4 Tree-depth and minors of hypergraph pairs . . . 68
4.4.1 Hypertree-depth of hypergraph pairs . . . 68
4.4.2 Alternative characterisations of hypertree-depth . . . 70
4.4.3 Marshals and robber game on hypergraph pairs . . . 72
4.4.4 Hypergraph pair minors . . . 74
4.5 Computational complexity and minor ordering . . . 77
4.6 Interaction with other invariants . . . 78
4.7 Remarks and open problems . . . 80

Notation 81

Bibliography 83

Chapter 1

Pursuit games on graphs

We begin with a general, introductory chapter providing history, motivation and context for pursuit game research. While the games are broadly studied since the 80's, the game seems to have appeared first along with a problem of locating a person lost in a cave system, represented as a graph: Inspired by his spelunker friend Richard Breisch, T. D. Parsons [Par76, Par78] first considered such a graph theoretic problem around the year 1976.

A small part of the current motivation for pursuit games research are security applications, but the theoretically most interesting, as well as most successful research seems to be in exploring theoretical applications and connections of these games in graph theory and complexity theory, and in exploring the structure of these games on their own. This is also the direction we approach the games from; exploring the structure of the underlying graphical structures, feasible strategies, and the implied computational complexity and algorithms.

In Section 1.1 we introduce the notation used throughout this thesis and recall some of the classical results required later. Section 1.2 introduces intersection graphs in general as well as defining the classes of our interest. Section 1.3 gives a general overview of pursuit games, presenting a rich framework encompassing many common and studied game variants and modifications. Finally, Section 1.4 presents several notable results, questions and conjectures.

1.1 Notation and preliminaries

In this thesis we mostly follow well-established mathematical notation and basic results, but for the sake of clarity and completeness we briefly recall most of the notation and technical terms. You may find an overview of the notation and common symbols in section Notation on page 81.

As standard references, we recommend *Modern graph theory* by Béla Bollobás [Bol98] as the graph theory resource, and *Lessons in Play: An Introduction to Combinatorial Game Theory* by Michael H. Albert, Richard J. Nowakowski and David Wolfe [ANW07] for combinatorial graph theory.

We recommend the book *Computational complexity: a modern approach* by Sanjeev Arora and Boaz Barak [AB09] or the classical *Computational Complexity* by Christos H. Papadimitriou [Pap94] as the standard computational complexity references. However, basic knowledge of algorithms and complexity classes should be sufficient throughout the thesis.

We use $\mathbb{R}$ to denote the set of real numbers, $\mathbb{R}^+$ the set of positive real numbers, $\mathbb{N}$ the set of natural numbers (including zero). For $a, b \in \mathbb{R}$, (a, b) denotes the open interval $\{x \mid a < x < b\}$, empty when $b \geq a$. For a set $X \subseteq \mathbb{R}^k$, $\mathrm{clos}(X)$ denotes the topological closure (smallest closed superset) of X and $\mathrm{int}(X)$ denotes the topological interior (largest open subset) of X. For a set X, 2^X denotes the powerset of X, that is $\{Y \mid Y \subseteq X\}$.

For two functions $f, g : \mathbb{N} \to \mathbb{R}^+$ we write $f = \mathcal{O}(g)$ if there is $c > 0$ such that $\forall x \in N : f(x) \leq cg(x)$[1]. Similarly, $f = \Omega(g)$ if there is $c > 0$ such that $\forall x \in N : f(x) \geq cg(x)$ and $f = \Theta(g)$ if $f = \mathcal{O}(g)$ and $f = \Omega(g)$.

We generally use minuscule letters $a, b, c, \ldots$ to denote set elements, $f, g, \ldots$ functions, $i, j, k, l, m, n, \ldots$ natural numbers, $u, v, w, \ldots$ graph vertices $x, y, \ldots$ numbers or points, capital letters $A, B, C, X, Y, \ldots$ for sets, G, H for graphs, and then calligraphic letters $\mathcal{A}, \mathcal{G}, \ldots$ to denote set families and graph classes.

In some parts of Section 3, we deal with *multisets*, which is a generalisation of sets allowing multiplicities of the elements. We just sketch basic notions and properties, referring to a standard set-theory resources for details.

One way to define multisets is to consider an *indicator function* $1_A : A \to \mathbb{N}$ for every multiset A, counting the number of occurrences of every element (and defined to be 0 everywhere outside A). Another, equivalent and sometimes more convenient, definition is to distinguish the occurrences of all elements, e.g. by letting the set $\{(x, 1), \ldots (x, k)\}$ represent the multiset with k occurrences of x.

The set-theoretic notions are naturally extended to multisets by considering and comparing the multiplicities as well. For example, for multisets $A \subseteq B \iff \forall x \in A : 1_A(x) \leq 1_B(x)$, a multiset function $f : A \to B$ maps every element $x \in A$ to a multiset $f(x) \subseteq B$ with $|f(x)| = 1_A(x)$ and similarly, for a function $f : A \to B$ to be injective means that for all $y \in B$ $|f^{-1}(y)| = 1_B(y)$.

1.1.1 Graph theory

We recall the basics of graph theory relevant to the thesis. See the books of Bollobás [Bol98] or Diestel [Die10] for more details.

A *graph* is a tuple $G = (V, E)$ where V is a *vertex set* and $E \subseteq \binom{V}{2}$ its *edge set*. We mostly deal with undirected graphs; directed graphs have edges as ordered pairs of vertices. Let $V(G)$ denote the vertex set of G and $E(G)$ the edge set of G. For $u, v \in V(G)$, let $uv = \{u, v\}$ be the edge between u and v. The number of vertices of G is the *order of* G, the number of edges the *size of* G. We use $N_G[v]$ and $N_G[X]$ to denote the *closed neighborhood* of a vertex v or a subset of vertices X (including v, resp. X), and $N_G(v)$ to denote the open neighborhood of v (not containing v). Let the degree be $d_G(v) = |N_G(v)|$. We drop the subscript G when the graph is clear from the context. Vertices u and v are *adjacent* when $uv \in E(G)$.

There are several graphs operations: deletion of an edge, deletion of a vertex (also removing all the edges incident with it), and edge uv contraction: creating a new vertex w adjacent to $N(u) \cup N(v)$ and deleting the two endpoints u, v. A

[1] While this notation is standard and widely used, to rationalise the at first counter-intuitive notation of $f = \mathcal{O}(g)$, we could define $\mathcal{O}(g)$ as the class of such functions f.

subgraph of G is a graph obtainable by deleting some vertices and some edges. An *induced subgraph of* G is a graph obtainable by deleting vertices only.

A *minor of* G is a subgraph obtainable by all three operations: deleting edges and vertices and edge contraction (in any order). An *induced minor* of G is obtained by a sequence of vertex deletions and edge contractions (no direct edge deletion). Graph minors are introduced in more detail in Section 4.2.3, the concept of minor maps and models is recalled in Section 4.4.4.

For any set A of vertices of G, we denote by $G[A]$ the subgraph of G *induced by* A, that is, the subgraph with the vertex set A and with an edge between each pair of vertices of A that are adjacent in G.

Two graphs G, H are *isomorphic* when there is a bijection $f : V(G) \to V(H)$ with

$$\forall u, v \in V(G) : uv \in E(G) \iff f(u)f(v) \in E(H).$$

In some cases when dealing with abstract graphs, the vertex representatives themselves are not as important as the graph structure, so in most cases the graphs are distinguished only up to isomorphism. For a graph H, we say that G *contains* H *as a subgraph (induced subgraph, minor)* when H is isomorphic to some G' that is a subgraph (induced subgraph, minor) of G.

Basic graphs and properties

The *path* on n vertices is $P_n = (\{0, 1, \dots n-1\}, \{\{0,1\}, \{1,2\}, \{2,3\}, \dots \{n-2, n-1\}\})$. The length of a path P_n is $n-1$ (the number of edges). The distance of two vertices $\mathrm{dist}(u, v)$ is the length of a shortest $u - v$ path in G, or ∞ when no such path exists. Graph G is *connected* when there is a path between any two of its vertices. A *component of connectivity of* G is any of the maximal connected subgraphs.

The *cycle* on n vertices C_n is P_n with additional edge $\{n-1, 0\}$. A graph is acyclic when it does not contain any cycle. A *complete graph* on n vertices is $K_n = (\{0, 1, \dots n-1\}, \binom{V}{2})$. An *empty graph* on n vertices is $I_n = (\{0, 1, \dots n-1\}, \emptyset)$. Any connected acyclic graph is called a *tree*.

A *homomorphism* is a map $f : V(G) \to V(H)$ such that for every $uv \in E(G)$, $f(u)f(v) \in E(H)$. When H is not a reflexive graph (that is with loop edges on every vertex) this also forces $f(u) \neq f(v)$ for every $uv \in E(G)$. A *retract* from G to $H \subseteq G$ is a map $f : V(G) \to V(H)$ fixing H (that is $f(v) = v$ for all $v \in V(H)$) and for every $(u, v) \in E(G)$ either $(f(u), f(v)) \in E(H)$ or $f(u) = f(v)$. Note that a retract is also a homomorphism to H with loops added to every vertex.

We say that a set A of vertices *dominates* a vertex set B when every $v \in B$ belongs to A or has a neighbour in A, that is, $B \subseteq N_G[A]$.

Rooted forests and orders

A *rooted tree* is a tree with one distinguished vertex *(the root)*. A *rooted forest* is a graph constituting of a set of rooted trees.

Let F be a rooted forest. We use $\leq_F$ to denote the partial order on $V(F)$ induced by F (where the roots are the smallest elements). Let T be one of the rooted trees composing F. For $s, t \in V(T)$ we denote the *infimum* (with respect to $\leq_F$) of s and t by $s \wedge t$. A *chain* in F is a subset of $V(F)$ that is totally ordered

by $\leq_F$. For $t \in V(F)$, by $\downarrow_F t$ we denote the chain of all $\leq_F$-predecessors of t in F (including t). We omit the subscript F if it is clear from the context. The *height* of a node $t \in V(F)$ is the number of vertices in a path from the root (of the tree containing t) to t. The *height* of F is the maximum height of the vertices in F. For a rooted forest F and a vertex $t \in V(F)$ we let F_t be the subtree of F induced by $\{s \in V(F) \mid t \leq_F s\}$.

Hypergraphs

A *hypergraph* is a pair $H = (V, E)$ with V an arbitrary set of vertices and $E \subseteq 2^V \setminus \{\emptyset\}$ the set of *hyperedges*. In a sense, a hypergraph is a graph with arbitrary size edges, and many of the graph concepts also generalise to hypergraphs. Any graph may be viewed as a hypergraph, but we try to keep the groups of graph and hypergraph notions separated.

A hypergraph is *k-uniform* if $\forall e \in E : |e| = k$. A hyperedge e is *singleton* when $|e| = 1$. A vertex is isolated when it is not contained in any edge (not even a singleton edge). For a hypergraph $H = (V, E)$, define the *underlying graph*

$$\underline{H} = (V, \{(u, v) \mid \exists e \in E : u, v \in e\}).$$

1.1.2 Combinatorial game theory

We sketch the basic concepts and tools of combinatorial game theory we address in our work. For a detailed introduction we recommend the book of Albert et al. [ANW07].

For two players, **A**nna and **B**arbara, a *two-player deterministic full-information memory-less combinatorial game* is described by the following: a state space S consisting of the following *disjoint* sets: the set of **A**-states S_A, the set of **B**-states S_B, the set of states won for Anna W_A, the set of states won for Barbara W_B and the set of draw states W_D. We have $S = S_A \cup S_B \cup W_A \cup W_B \cup W_D$, finite or infinite. Then there is the initial state $s_0 \in S$ and finally a *rule* function $r : (S_A \cup S_B) \to (2^S \setminus \{\emptyset\})$ indicating the valid moves in every state except the final (won or drawn) states.

The game is played as follows: The first game state is s_0. Then on i-th round with the game in the state s_{i-1} one of the following happens:

$s_{i-1} \in S_A$ Anna picks the next state s_i from the set $r(s)$.

$s_{i-1} \in S_B$ Barbara picks the next state s_i from the set $r(s)$.

$s_{i-1} \in W_A$ Anna wins the game.

$s_{i-1} \in W_B$ Barbara wins the game.

$s_{i-1} \in W_D$ The game ends with a draw.

If the game goes on indefinitely avoiding the final states, we also consider it a draw.

Both Anna and Barbara have full information about the set S and the rules r when making their choice. A memory-less strategy for Anna (resp. Barbara) is a function $\mathcal{S} : S_A \to S$ (resp. $\mathcal{S} : S_B \to S$) such that $\mathcal{S}(s) \in r(s)$ (strategy advises

only valid moves). Given two strategies $\mathcal{S}_A$ and $\mathcal{S}_B$, the game is determined as a (potentially infinite) sequence $s_0, s_1, s_2, \ldots$. An Anna's (resp. Barbara's) strategy is winning if it wins against any Barbara's (resp. Anna's) strategy. Similarly, a strategy is non-losing if it wins or draws (including playing an infinite game) against any strategy.

Note that in a memory-less game, it is a well-known fact that memory-less deterministic strategies are as strong as any other strategies (e.g. considering state history or using randomness) and therefore we only need to consider those.

A note on the adjectives of a two-player deterministic full-information memory-less combinatorial game: Two players is an important limitation since with more players the tools for dealing with winning strategies are much weaker (imagine the currently "losing" player helping the currently "second" player etc.). The game is deterministic since the next state is determined only by the rules and the players' choices (without any randomness). In a partial information game, the players would only have some information of the game state (having hidden variables for one or both players). In a memory-less game, the valid moves and won states only depend on the current state (and not on previous states or the number of turns played)[2].

There is an important folklore game-theoretic theorem allowing us to classify the games by the winning player. There are several possible variations on the game conditions, in our case we additionally restrict ourselves to finite state-space games.

Theorem 1. *In a deterministic, full-information, finite-state memory-less two-player combinatorial game, one of the players has a memory-less non-losing strategy. If a draw or endless game is not possible, one player has a winning strategy.*

The *game tree* is a tree rooted in s_0 with every vertex of the tree representing a state from S and the descendants of every vertex representing the state s being the representatives of $r(s)$ (or none if $s \in W_A \cup W_B \cup W_D$). A play (e.g. of two given strategies) is a downward path in this tree. Another view on a game is the *game-state directed graph*, where the vertices are exactly the states of S (labelled with their type S_A, S_B, W_A, W_B or W_D), vertex s_0 is distinguished, and from every state $s \in S_A \in S_B$ there are directed edges to $r(s)$.

These representations are good tools to observe the following two properties of combinatorial games as well as the complexity remarks below. Refer to the book by Albert et al. [ANW07] for further details.

Lemma 2. *In a memory-less game, a winning strategy never has to repeat a game state against any strategy.*

Lemma 3. *In a memory-less finite-state game, a winning strategy can be assumed to have a finite bound on the game length.*

Given a game from a fixed class of games, it is not generally computable to decide the winning (or non-losing) player, or to find the winning strategy, depending

[2]Note that it is possible to encode the game history within the current state, getting a (vastly) extended game state but a memory-less game. However the games we consider are naturally memory-less.

on the computability and size of S and r. However, for finite-state memory-less games we have the following well-known general state-marking algorithm.

Theorem 4. *Given a two-player deterministic full-information memory-less and finite-state combinatorial game (given either by $(S_A, S_B, W_A, W_B, W_D)$, s_0 and r, or by the state-space graph) there is a polynomial time algorithm (in the size of the input) to decide the outcome of the game and find the shortest winning (resp. non-losing) strategy.*

1.2 Intersection graph classes

A big part of the thesis, namely Chapters 2 and 3, deal with intersection graph classes, mostly defined by planar geometrical objects. Here we introduce some of the notable and relevant classes, and point out some of their interesting properties.

An *intersection representation* of graph G is a map $\varphi : V_G \to \mathcal{I}$ for some *ground set* $\mathcal{X}$ and some *image set* $\mathcal{I} \subseteq 2^{\mathcal{X}}$, such that the edges of G are prescribed by intersections of the sets $\varphi(v)$; formally, $uv \in E_G$ if and only if $\varphi(u) \cap \varphi(v) \neq \emptyset$. Given the (multi)set of the images of φ, the corresponding *intersection graph of* φ is uniquely defined up to isomorphism.

Note that we can obtain any graph G as an intersection graph of the set of its edges: Let $\mathcal{X} = E_G$ and $\varphi(v) = \{e \in E_G \mid v \in e\}$. Therefore, for any infinite set $\mathcal{X}$ and $\mathcal{I} = 2^{\mathcal{X}}$ we obtain the class of all (finite) graphs.

Interesting intersection classes are most commonly defined by restricting the ground set $\mathcal{X}$ and the image set $\mathcal{I}$. In some cases the allowed set of images of $\varphi[V_G]$ is also restricted, for example by bounding the number of intersections at a single point of $\mathcal{X}$.

Intersection graphs are widely used in abstract graph theory and when constructing graphs out of combinatorial objects (together with, e.g., incidence graphs). As an example, take $\mathcal{X} = \mathbb{N}$ and $\mathcal{I} = \binom{\mathcal{X}}{2}$, the set of all unordered pairs. It is not difficult to see that this yields the family of (countable) *line graphs* – the intersection graphs of the edges of any given graph.

The intersection graph classes of our interest are mostly geometry or graph-based and we introduce them below. Additional examples of intersection graph classes are the intersection graphs of segments in a plane, balls or unit balls in $\mathbb{R}^k$, convex subsets of a plane or connected subgraphs of a given (fixed) graph. For more information on intersection graphs and their properties, see a book by McKee and McMorris [MM99] or an older classic by Golombic [Gol80].

The classes of planar and bounded genus graphs are closely related to intersection classes (but not natural intersection classes themselves). Planar graphs (denoted PLANAR) can be drawn into plane (with vertices as distinct points and edges as arcs connecting their endpoints, and with the edges crossing only in the endpoints). Graphs of genus at most k (denoted GENUS-k) may be drawn similarly on a surface of genus k.

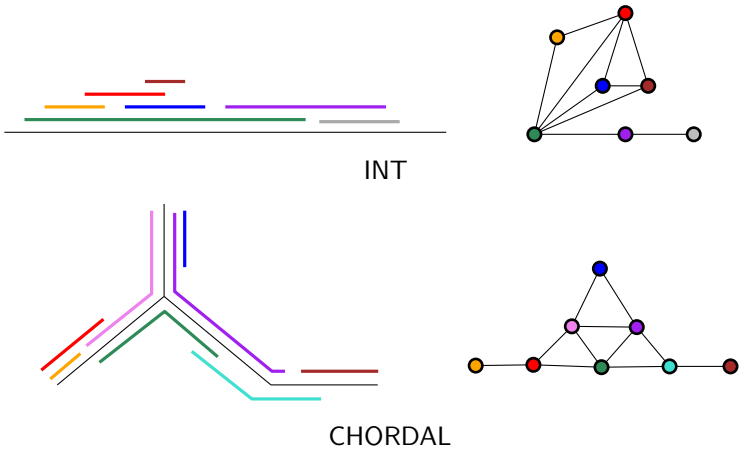


Figure 1.1: Examples of interval and chordal graphs.

1.2.1 Interval and chordal graphs

An *interval graph* (denoted INT) is an intersection graph of intervals of the real line. See Figure 1.1 for an illustration.

There was an extensive research on interval graphs in the 60s and 70s, firs appearing in a paper by Lekkeikerker and Boland in 1962 [LB62]. Today, these graphs are folklore and they appear in many graph theoretic contexts as well as modeling objects outside graph theory.

As for the representation, in case of finite graphs, it does not matter whether the intervals are open or closed. Alternatively, one can define (finite) interval graphs as intersection graphs of subpaths of a path, or of intervals of natural numbers. We may generally assume that all the $2n$ endpoints of the intervals are distinct natural numbers $1 \dots 2n$, where n is the order of the graph, which can be achieved by moving the endpoints without changing their order, and therefore without changing the graph.

A given abstract graph can be recognized as interval and its interval model can be constructed in linear time [BL76, KM89]. Interval graph representation also requires only $\mathcal{O}(n \log n)$ bits, unlike general graphs which require $\Theta(n^2)$ bits. There are also other characterizations of interval graphs by forbidden substructures already by Lekkeikerker and Boland [LB62], but this falls outside of the scope of this thesis.

Formally, an interval representation $\varphi : V \to 2^{\mathbb{R}}$ maps each vertex v to an open interval such that $\varphi(u) \cap \varphi(v) \neq \emptyset$ if and only if $uv \in E$. For any subset X of vertices of an interval graph G let $\varphi[X] = \bigcup_{x \in X} \varphi(x)$ the union of intervals representing the vertices in X. Note that when G is connected, then for any $X \subseteq V_G$ the subgraph $G[X]$ is connected if and only if $\varphi[X]$ is an interval.

For any number i, $\varphi^{-1}(i) = \{v \in V \mid i \in \varphi(v)\}$ is the set of vertices whose intervals contain i[3]. Then, for $X \subseteq \mathbb{R}$, let $\widetilde{\varphi}(X) = \{v \in V \mid \varphi(v) \subseteq X\}$ be the vertices with intervals entirely contained in X. Since here X is usually an interval, we let $\widetilde{\varphi}(i, j) = \widetilde{\varphi}((i, j))$. Note that $\widetilde{\varphi}(i, j) \cap (\varphi^{-1}(i) \cup \varphi^{-1}(j)) = \emptyset$ for every i and j.

[3]This is not exactly the inverse of φ, but we believe this convenient notation won't cause any misunderstanding.

Given an interval I, denote by $\mathrm{L}(I)$ and $\mathrm{R}(I)$ the left and right endpoints of I, respectively, i.e., $\mathrm{L}(I) = \inf(I)$ and $\mathrm{R}(I) = \sup(I)$.

The intervals of a representation are naturally ordered in two ways – by their left and right endpoints. We use these to define two orders on V_G: Let $u <_R v$ if and only if $\mathrm{R}(\varphi u) < \mathrm{R}(\varphi(v))$. Similarly, $u <_L v$ if and only if $\mathrm{L}(\varphi(u)) < \mathrm{L}(\varphi(v))$. Note that these orders are linear thanks to the assumption that all endpoints are different.

In the algorithmic sections, $<_R$ is the commonly and sometimes implicitly used interval order while $<_L$ usually plays an auxiliary role. This is due to our choice of sweeping the graph representation left-to-right. In particular we use the following properties:

Lemma 5. *If we have $a <_R b <_R c$ and $ac \in E$, then also $bc \in E$. Similarly, if $a <_L b <_L c$ and $ac \in E$, then also $ab \in E$.*

Proof. Since $ac \in E$ we have $\mathrm{L}(\varphi(c)) < \mathrm{R}(\varphi(a))$ and therefore $\mathrm{R}(\varphi(b)) \in \varphi(c)$. The proof of the second statement is symmetric. □

The integers $1, \ldots, 2|V|$ are also called *cutpoints*, as every $\varphi^{-1}(i)$ is a vertex cut in the interval graph between the vertex sets $\widetilde{\varphi}(-\infty, i)$ and $\widetilde{\varphi}(i, \infty)$. However, unlike with the usual definition of a cut, here either may be empty.

Chordal graphs (denoted CHORDAL) have several popular equivalent definitions. Within the framework of intersection graphs, a chordal graph is an intersection graph of subtrees of a tree. By definition, chordal graphs are a superclass of interval graphs, and have a "tree-like" structure instead of a "path-like" one of interval graphs.

An older and more common definition is the following: A graph is *chordal* if every cycle of length at least 4 has a *chord* – an edge between two vertices nonadjacent in the cycle. The equivalence was shown by Gavril in 1974 [Gav74]. Other characterizations, as well as linear time recognition algorithms, are based on vertex orderings and perfect elimination schemes. See the book by McKee and McMorris [MM99] for more information.

1.2.2 Circle and circular arc, function and IFA graphs

The following related classes are the topic of Section 2.4. See Figure 1.2 for an illustration.

Circle graphs (denoted CIRCLE) are intersection graphs of chords of a single circle, that is straight line segments between two points on the circle. Bouchet [Bou85] and Naji [Naj85] have shown that circle graphs can be recognized in polynomial time.

Circular arc graphs (denoted CIRCARC) are the intersection graphs of arcs (connected subsets) of a single given circle. These graphs can be also recognized in polynomial, even linear time as shown by McConnell [McC03].

Function graphs (denoted FUN) are the intersection graphs of continuous functions defined on the interval $[0, 1]$[4]. A much better known equivalent class is

[4]You may require the functions to be from $[0, 1]$ to $[0, 1]$, to be piece-wise linear et c.

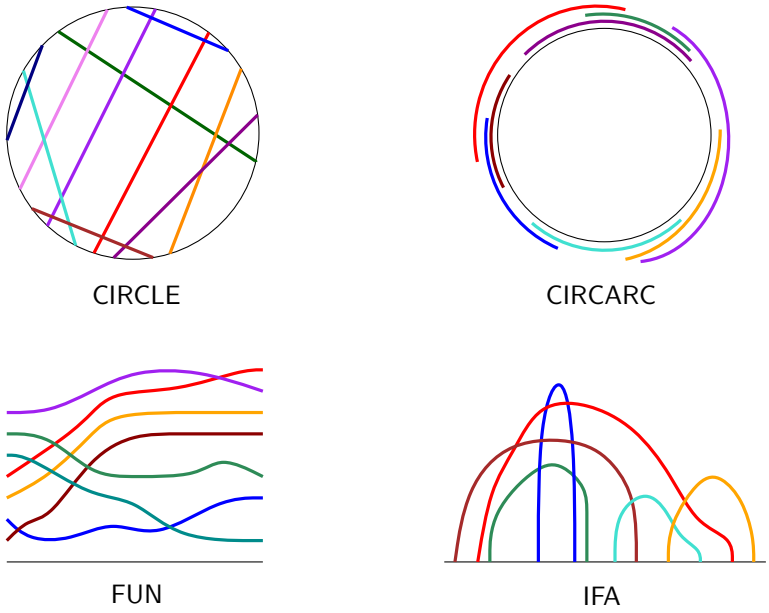


Figure 1.2: Examples of geometrical intersection graphs.

the class of *co-comparability graphs* – that is complements of *comparability graphs*[5]. While this characterization is widely studied in both algorithmic and structural context, we refer the interested reader to the book by Golombic [Gol80], who has also shown the recognition complexity to be polynomial [Gol77].

Interval filalent arc graphs, or *IFA graphs* for short (denoted IFA) are the intersection graphs of filaments over a set of intervals. a *filament over an interval* $[a, b]$ is a continuous function f from $[a, b]$ to $[0, \infty)$ with $f(a) = f(b) = 0$. We call a the *left endpoint* and b the *right endpoint* and assume $a \neq b$.

Notice that, similarly to interval graphs, the filaments over $[1, 3]$ and $[2, 4]$ must intersect, the filaments over nested intervals such as $[1, 4]$ and $[2, 3]$ may or may not intersect. The class was introduced by Gavril in 2000 [Gav00] as a generalization of interval graphs on which his algorithms for weighted maximum clique and weighted maximum independent set can be used. Pergel [Per07] has later shown their recognition to be NP-complete.

While the classes of circle graphs, circular arc graphs and function graphs are generally incomparable, IFA graphs is their common superclass. We sketch the class inclusions for the curious reader: Continuous functions on $[0, 1]$ can be easily completed to filaments without introducing new intersections. The circle of a circle graph may be cut at any point and straightened to a segment while continuously morphing the original chords into filaments. Cutting and straightening the circle of a circular arc graph gives almost an interval graph except for some arcs now split into 2 intervals. The corresponding vertices are all adjacent in the intersection graph and can be connected by filaments going above the interval part almost arbitrarily.

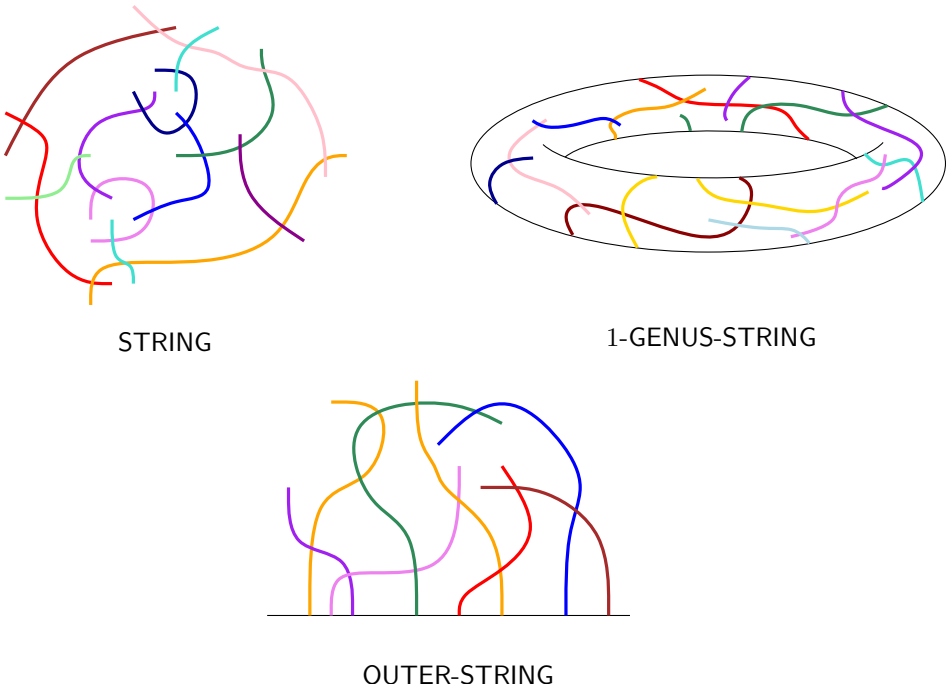


Figure 1.3: Examples of string-based intersection graphs.

1.2.3 String graphs and related classes

String graphs (denoted STRING) are intersection graphs of strings in the plane. A *string* is the image of a continuous function from $[0, 1]$ to $\mathbb{R}^2$, that is a continuous path between two points in the plane. See Figure 1.3 for an illustration. The class was introduced by Sinden [Sin66] and characterized to be equivalent to intersection graphs of arc-connected regions of the plane. See [MM99] for further information.

String graphs on genus g surfaces (denoted GENUS-g-STRING) are defined similarly, but on arbitrary surface of Euler genus g. We are not aware of any work on this class prior to our work on this subject [GGJ+15]. The extension however is quite natural and aims to follow the rich development in the area of graph genus and graphs drawn on surfaces.

A graph G is an *outer-string graph* (denoted OUTER-STRING) if it has a string representation by curves lying in the upper half-plane and intersecting the x-axis in exactly one point, which is an endpoint of the curve. Alternatively, we may require that every string in a plane has at least one endpoint outside of a disc containing all the string intersections (hence the name).

Note that outer-string graphs are related to, but less limited than interval filament arc graphs.

Recognition of string graphs is an interesting story on its own. The problem was first shown to be NP-hard by Kratochvíl in 1991 [Kra91], but even decidability of the recognition problem was open for quite some time. For example, Kratochvíl et al. [KGK86] shows that there are string graphs requiring an number of string intersections exponential in the order of the graph, so any straightforward description of a (piecewise-linear) string model may require (at least) exponential

[5] Also known as *transitively orientable graphs*, a comparability graph for a given partially ordered set connects comparable elements with undirected edges.

space.

In 2001, Schaefer and Štefankovič [SS04] have shown that the number of intersections is always at most exponential, and that string graphs may be recognized in non-deterministic exponential time. Shortly afterwards, Schaefer et al. [SSS03] have finally shown the membership in NP. (The reversed years of publication of those results are due to different publishing delays.)

To our best knowledge, the complexity of recognizing outer-string graphs and g-genus string graphs for $g > 0$ are open problems.

1.2.4 General assumptions and remarks

To simplify our treatment of the matter without a loss of generality, we make the following assumptions on the geometrical graph representations throughout the thesis.

1. No two strings, functions, lines or subtrees share an endpoint (resp. a leaf).
2. In the plane, no three lines or strings intersect in one point.
3. In the plane, whenever two strings touch, they also cross or exactly one of them ends.
4. In the plane, no string self-intersects.
5. The number of intersection points (or connected intersection regions) is finite.

Assumptions 1, 2 and 3 are easily achiever through local perturbation. Assumptions 4 and 5 follow from the fact that strings can be replaced by piece-wise linear curves with finite numbers of linear segments without affecting their intersection graph. For more details see a book by Kratochvíl et al. [KGK86]. Assumption 5 for IFA and function graphs follows from the fact that they are a special type of string graphs. We always assume and maintain these properties.

As noted in Section 1.2.3, string graphs are a superclass of any intersection class of arc-connected objects in the plane (i.e. intersection graphs of connected regions bounded by closed simple Jordan curves). While there are some intersection classes with every object a union of at most k arc-connected objects (of various types, e.g. *2-interval graphs*), weakening of the connectivity condition to topological connectivity would immediately allow all graphs.

Proposition 6. *Every graph can be represented as an intersection graph of topologically connected sets in the plane.*

While this is folklore, one can obtain such representation by first getting an arbitrary string representation of K_n and then replacing unwanted crossings by (no longer string) crossing gadgets preserving topological connectedness but making the sets disjoint. See a book by Prasolov [Pra] for more information.

As a last remark, notice that in the matter below we do not distinguish between placing the player tokens on the vertices of the graph or their geometric representation. This simplifies the language when we focus on the representation of the graph and we believe that it shall not bring any confusion to the reader.

1.3 Game variants

While there are numerous variants and modifications of pursuit games, some of them mentioned in Section 1.4 and three examined closely in this thesis, it is possible to describe a framework fitting most of the games. This presentation of the framework is not meant formally, but rather to give the reader an idea of the scope of variations and choices this family of games allows.

While only several games of the myriad possible ones are directly motivated by other problems from mathematics and computer science, it is of independent interest to understand the full landscape of these games, comparing the significance and strength of individual changes and their interplay in various situations.

The understanding of the "game landscape" may also give new characterizations of old related problems, as well as serve as a criterion of naturalness of proposed definitions. Notable examples are three proposed definitions of directed graph counterpart of tree-width, namely *Directed tree-width* by Johnson et al. [JRST01], *Kelly-width* by Hunter and Kreutzer [HK08], and DAG-width by Obdržálek [Obd06], all of them arguing their case with game characterization of the new parameter. Chapter 4 of this thesis could be also seen as a step in this direction.

1.3.1 Pursuit game framework

Here we sketch a framework of a pursuit game in the sense used in this work. While we give a formal definition, there is no commonly accepted formalism and a slightly informal language of game variants is widely used and sufficient for most uses. See a survey by Alspach [Als04] for a broader overview of known pursuit games.

Below, by "usually" we mean that it either applies to most games known to us, or that other natural definitions are either equivalent or nonsensical for most known games. Examples include victory conditions and game set-up and start.

The game is played by two players: The *pursuer* is usually called *Cops, Cleaners, Rescuers, Marshals, ...*, while the *evader* is called *Robber, Contamination, Virus, Victim, Fugitive, ...*. Each player controls a set of indistinguishable *tokens*, usually of bounded size and quite often with just one evader.

At every moment, every token is either off-board or placed on some *locations* of the game board. Most commonly, the locations are the vertices of a given graph, other variants include edges and hyper-edges. Note that games on infinite, even uncountable, structures are sometimes also considered. The number of tokens in a single location is generally not limited. Some locations may be available to only one player.

Most commonly, the pursuer wins when all (or a defined quantity) of the evader tokens are *captured*, which means that they share location with a pursuer token, or in some cases, that they have been previously removed from the board. The evader wins by avoiding capture indefinitely. In case of full-information games with a finite state-space this is equivalent to repeating a game state, as follows from Lemma 1.

The game rules and the playground structure are known to both players, as well as the position of the pursuer's tokens. This information may even include the

declaration of next moves of the pursuer tokens. The visibility of the evader varies from invisible evader to evader visible from distance at most d to fully visible evader. Invisible robber games were introduced by Tosic already in 1985 [Tos85].

While the games with invisible evader are generally no longer full-information and therefore may not fall into the combinatorial game framework, some of them are equivalent to a different game with full information. A typical example: The hidden information in a game with a single infinitely fast invisible evader can be encoded in a set of evader's possible locations, yielding a contamination clean-up game where the evader has tokens in all contaminated areas.

Every game round consists of a pursuer turn followed by an evader turn. In the beginning of the game, the tokens are generally off-board and the first round has special rules to place the tokens. Most commonly the placement of the tokens is not restricted in any way, but the evader's tokens are placed last.

In most games, including the games in Chapters 2 and 3, every pursuer token may move to distance at most one[6], passing a move is allowed. In the first turn, the cops choose any starting positions instead of moving.

In the *helicopter game* characterizing tree-width, every pursuer may "fly" to a location "above" any vertex, but will land there only after the evader's move, allowing the evader to react. This "announcing" of the moves before they take place is also present in other pursuer move rules (e.g. variations on the first one above).

In another family of games, the pursuer tokens are not placed in the first turn and any already placed tokens may not be moved at all. Every turn, one location is chosen by the pursuer and announced to the evader. After the next evader's turn, a pursuer token is placed in the location and a new location is chosen. This one and similar rules characterize graph parameters related to tree-depth and are the main topic of Chapter 4. Note that the number of pursuer tokens in such games corresponds to the maximum length of the game.

The evader generally moves along a path of length at most s avoiding all pursuer tokens. Here, s is a parameter of the game. Most commonly $s = 1$, which is the case in Chapter 2. A special case $s = \infty$ allows movement arbitrarily far, but it is not possible to skip "over" pursuer tokens, e.g. when they hold an entire cut. We explore these cases in Chapters 3 and 4.

In the games against "contamination" (or virus), new evader tokens are placed in all locations in distance at most s from an old token along a path avoiding all pursuers. Again $1 \leq s \leq \infty$ is a parameter of the game. As noted above, these games are related to games with a single invisible evader, but we refer the curious reader to a paper by Dereniowski et al. [DDTY13] or to the survey by Alspach[Als04].

There are many additional rule variations (e.g. all evaders must be captured at once), restrictions (e.g. pursuers must avoid getting disconnected by the evaders) and playground structures (e.g. directed graphs, hypergraph pairs and groups). An notable condition is the restriction to "monotonic" pursuer strategies (the pursuers may not allow the evader enter a location once occupied by a pursuer). We leave the rest to the imagination of our kind reader and recommend a survey by Alspach [Als04].

[6]Games with faster pursuers are not very common.

1.4 Notable results

There are plenty and various results in the area of pursuit games and it is beyond the scope of this work to attempt to give their overview. A rich family of games with selected results and a brief history can be found in a survey *Searching and sweeping graphs: a brief survey* by B. Alspach [Als04] and in a more recent survey *An annotated bibliography on guaranteed graph searching* by F. V. Fomin and D. M. Thilikos [FT08]. Other specialised surveys include *Meyniel's conjecture on the cop number: a survey* of W. Baird, A. Bonato [BB13] and *Cops, robbers and graphs* by G. Hahn [Hah07] focusing e.g. on games on algebraic and infinite graphs.

The background and results relevant to the individual chapters can be found in the respective sections: Section 2.2 with history of traditional Cops and Robber game, time of capture, Meyniel's conjecture, known bounds for planar graphs and complexity remarks. Section 3.2 on the game with various robber speed. And Section 4.1 with games related to various graph *width* and *depth* parameters.

To give just a few examples of other games, the recent development includes results on games with various visibility [DDTY13, IKK06], speed [CCNV11, FGK$^+$10] or radius of capture [BCP10].

Chapter 2

Cops and robber games on intersection graph classes

The game of Cops and robber, introduced in 1983 by Nowakowski and Winkler [NW83] and independently by Quilliot in his thesis [Qui83], is one of the oldest mathematically studied pursuit game on graphs, and in some sense the most natural turn-based formalization of a real-world coordinated pursuit.

There are even popular board games Scotland Yard [B$^+$83] and its Czech variant Phantom of the old Prague[JAV87] based on the same principle of coordinated graph pursuit[1]. The first time I have encountered the game was in my bachelor thesis on maximum capture time of cop-win graphs [Gav07] and, together with many mathematicians all over the world, I found the game very approachable, yet elegant and interesting to work with.

The main problem – to decide the number of cops required to win on a given graph – has seen a lot of development over the years, and while the number is generally unbounded, many graph classes and parameters have been shown to require only a bounded number of cops. This chapter presents an overview of the most important of these results, focusing on geometrically represented graph classes and our contribution to the area.

It has been asked at several occasions, recently during the Banff Workshop on Graph Searching in October 2012, whether intersection-defined graph classes (other than interval graphs) have bounded maximum cop numbers. The classes in question have included circle graphs, intersection graphs of disks in the plane, graphs of boxicity 2, and others. In this chapter, we solve the question in affirmative in the most general way by bounding the cop number of string graphs on bounded genus surfaces, and therefore also of the intersection graphs of any arc-connected regions on a bounded genus surface.

Section 2.1 formally introduces the game. Section 2.3 presents an overview map of graph classes related to our as well as older results. Section 2.2 presents an overview of the game history and interesting results related to the game, but outside of the scopes of other sections. Sections 2.4, 2.5, 2.6 and 2.7 then present detailed analysis of the game on several geometrical graph classes. Section 2.8 generalises the guarding concept from Section 2.6.1 to retracts and shows an

[1]The board games, however, additionally include some hidden information, resources and pre-coloured edges restricting the movement.

application to grid graphs. Section 2.9 then invites the reader to some open problems and presents some general remarks.

2.1 Game of Cops and robber

We start by a formal definition of the game of *Cops and robber*. Note that it falls very nicely into the framework sketched in Section 1.3.1 as it has been the basis for most developed pursuit games.

This game is played by two players, called "the cops" and "the robber", on a given simple undirected graph G and with a given number of cops k. The game pieces (k cops and one robber) are always placed on the vertices of G and several pieces may share a vertex. The complete game state is always known to both players.

Initially the first player distributes k cops on the vertices arbitrarily, then the robber chooses any starting vertex (depending on the cops' positions). Then the players alternate in turns: First every cop moves to a vertex in distance at most one, then the robber moves to a vertex in distance at most one (passing a move is therefore allowed).

The game ends when the robber is *captured* which happens whenever a cop occupies the same vertex as the robber. The first player wins if he is able to capture the robber, the robber wins if he is able to escape indefinitely.

Note that the game is deterministic, full-information and has a finite state space, so according to Theorem 1, one player has a memory-less non-losing strategy. If the cops have a non-losing strategy, they in fact have a winning strategy and, according to Lemma 2, capture the robber in a number of moves bounded by the state-space size.

Our main interest lies in the minimum number of cops that have a strategy to capture the robber for a given graph and for a given graph class. Note that for finite graphs this number is always finite (but not necessarily for infinite graphs).

Definition 7. *For a graph G, its* cop number $\mathrm{cn}(G)$ *is the least number k such that the cops have a winning strategy on G with k cops. For a class of graphs $\mathcal{C}$, the* maximum cop number $\text{max-cn}(\mathcal{C})$ *is the maximum cop number* $\mathrm{cn}(G)$ *of a* connected *graph $G \in \mathcal{C}$, possibly $+\infty$.*

Note that the restriction to connected graphs is standard due to the following observation. Since most of the graph classes we discuss allow an unbounded number of components, we would need to always specify the necessary connectedness requirement in our bounds.

Lemma 8. *If G has connected components $C_1, \ldots, C_k$, then* $\mathrm{cn}(G) = \sum_{i=1}^{k} \mathrm{cn}(C_i)$.

Proof. This follows from the fact that the cops may never move between the components after the initial placement. Therefore, in every component C_i there has to be at least $\mathrm{cn}(C_i)$, which is sufficient. □

2.2 History and notable results

The game of Cops and robber was introduced by Nowakowski and Winkler in their 1983 article *Vertex to vertex pursuit in a graph* [NW83], and independently by Quilliot in his Master's and PhD theses in 1978 and 1983 [Qui78, Qui83][2]. We point out the following aspects of the game.

2.2.1 One cop game

Graphs with $\mathrm{cn}(G) = 1$ are also called *cop-win*, referring to a simple version of the game only considering a single cop. Both Nowakowski and Winkler, and Quilliot have characterized cop-win graphs to be exactly *dismantlable* graphs, defined in the following way: A vertex u is *dominated* by a different vertex v if $N[u] \subseteq N[v]$. A graph is dismantlable if it can be reduced to a single vertex by repeated removal of dominated vertices.

Bonato et al. [BGHK09] have shown that the maximum number of rounds required to capture the robber in a n vertex cop-win graph is at most $n - 3$; Gavenčiak [Gav07, Gav10] has shown $n - 4$ to be the tight bound.

2.2.2 Meyniel's conjecture

For general connected graphs, Meyniel made the following famous conjecture in a personal communication to Frankl [Fra87] in 1985.

Corollary 9 (Meyniel's conjecture)**.** *For any connected n-vertex graph we have* $\mathrm{cn}(G) = \mathcal{O}(\sqrt{n})$.

In the article, Frankl shows that if the conjecture holds it would be tight, since there are connected graphs that require $\Omega(\sqrt{n})$ cops.

Example 10. *Let G be the incidence graph*[3] *of a finite projective plane of order k. G then has girth*[4] *at least 6, degree $k + 1$ and $n = \Theta(k^2)$ vertices. Such graph requires at least $k + 1$ cops to capture the robber, since in any position of k cops the robber always has a safe adjacent vertex to escape to.*

2.2.3 Planar and bounded genus graphs

Somewhat surprisingly, Aigner and Fromme [AF84] have shown that the cop number of a connected planar graph is at most 3. We build on their result and generalize it to string graphs in Section 2.6.

This result has been generalized to graphs of bounded genus graphs by Quilliot [Qui85] and Schroeder [Sch01], the former showing max-cn(GENUS-g) $\leq 3+2g$, the latter improving this to max-cn(GENUS-g) $\leq 3 + \frac{3}{2}g$. Again, we generalize this result to string graphs on bounded genus surfaces in Section 2.7.

However, while for planar (genus 0) graphs the constant 3 is known to be the best possible, already for toroidal (genus 1) graphs the exact value of the maximum cop number (either 3 or 4) is not known.

[2] However, Quilliot focused on the one-cop version of the game.

[3] Vertices of G are the points and lines of the plane, edges connect a line to all its vertices.

[4] The length of a shortest cycle.

2.2.4 Computational complexity

As mentioned in Section 1.1.2, a game with a fixed number of cops k has state space size $\mathcal{O}(n^{k+1})$ and therefore can be decided in polynomial time. Note that the game is memoryless: the move options do depend only on the current game board state. The algorithms for exploring the full game space (games in *extensive form*), in different contexts usually called *state-marking*, *retrograde analysis* or *backward induction*, can be found in a classic book Fudenberg and Tirole: *Game Theory* [FT91]. These algorithms can also compute the time required to capture the robber and other properties of the game.

The following complexity corollary leverages this observation to some classes and is a further motivation for our study of max-cn.

Corollary 11. *The cop number of graphs from a class* $\mathcal{C}$ *with* max-cn$(\mathcal{C}) \leq k$ *can be computed in time* $n^{\mathcal{O}(k)}$*, where* n *is the order of the graph.*

However, for k part of the input, deciding whether the cop number of a graph is at most k has been shown to be NP-hard by Fomin et al. in 2010 [FGK$^+$10], then PSPACE-hard by Mamino in 2013 [Mam13] and very recently (2015) even EXPTIME-complete by Kinnersley [Kin15], confirming a conjecture of Goldstein and Reingold from 1995 [GR95].

For more details and results in this direction, see the recent book of Bonato and Nowakowski [BN11].

2.3 Map of classes and bounds

Figure 2.1 on the following extended page shows the map of upper and lower bounds on the maximum cop number for several intersection-based and related graph classes. The Hasse diagram depicts known inclusions between the classes. Bounded boxicity of bounded genus graphs has been shown in [EJ13]. The other inclusions are mostly well-known; for details and particular references we recommend the *Information System on Graph Classes and their Inclusions* website[5].

Note that only connected graphs are considered as per Lemma 8. Also, the sources are omitted where the bound follows from the bound for a sub- or super-class.

[5] `http://www.graphclasses.org/`

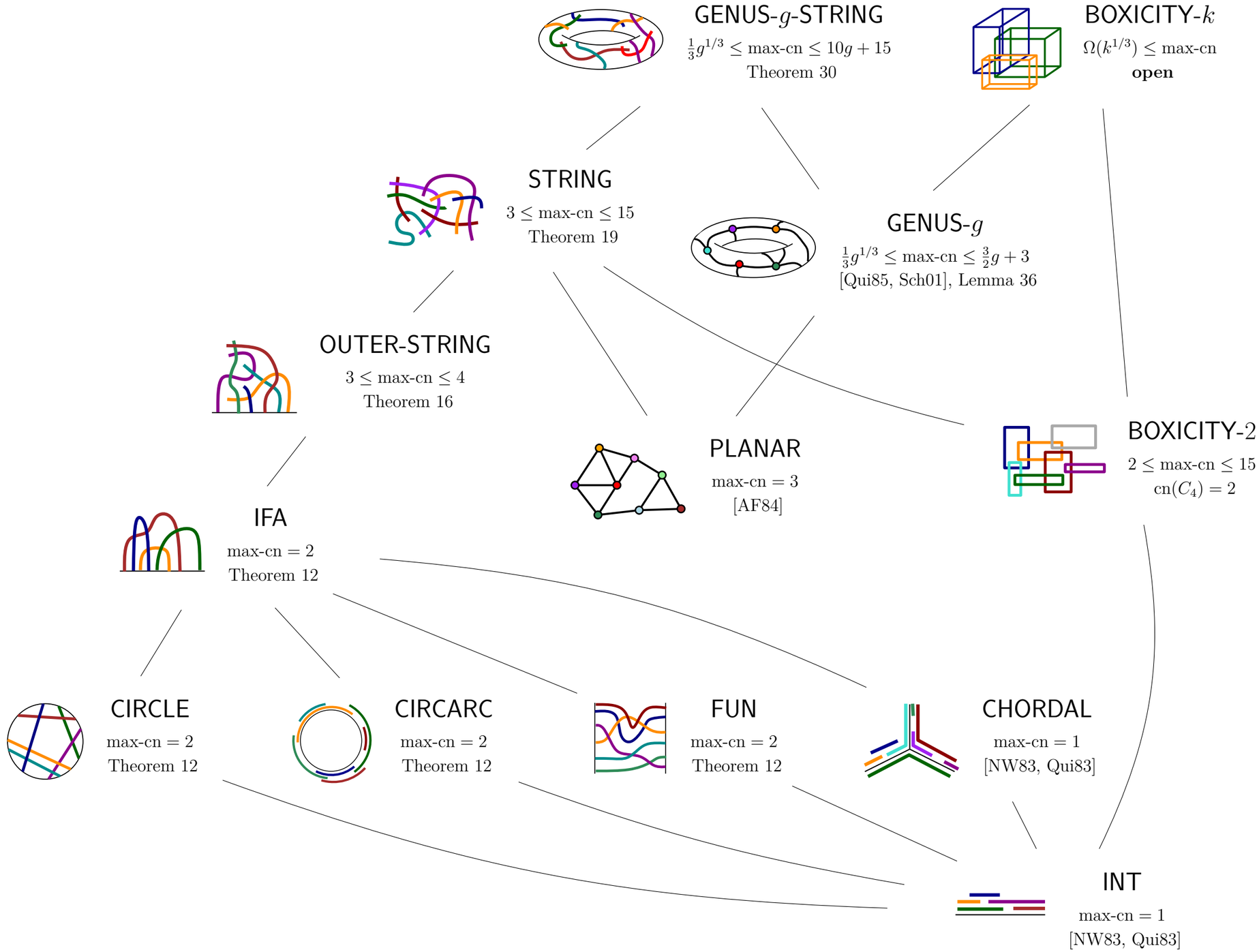


Figure 2.1: The map of bounds on the maximum cop number of several intersection-based and related classes.

2.4 IFA, function, circle and circular arc graphs

In this section, we examine the game on several related classes of intersection graphs: interval filament graphs, function graphs, circle graphs and circular arc graphs, all introduced in Section 1.2 together with their representation assumptions. The following theorem is shown in Section 2.4.2.

Theorem 12.

$$\text{max-cn}(\mathsf{IFA}) = \text{max-cn}(\mathsf{FUN}) = \text{max-cn}(\mathsf{CIRCLE}) = \text{max-cn}(\mathsf{CIRCARC}) = 2.$$

Moreover, $\mathcal{O}(n)$ turns are sufficient to capture the robber, where n is the order of the graph.

For the lower bounds, we have the following easy observation.

Observation 13. *Since the cycle graph C_4 is both function, circle and circular arc graph, and* $\text{cn}(C_4) = 2$*, we have* $\text{max-cn}(\mathsf{FUN}) \geq 2$*,* $\text{max-cn}(\mathsf{CIRCLE}) \geq 2$ *and* $\text{max-cn}(\mathsf{CIRCARC}) \geq 2$*.*

For the upper bounds, we show a strategy for two cops to capture a robber in any connected interval filament graph with a given representation φ, which then directly gives strategies for all the subclasses.

2.4.1 Filaments and regions

It is important that each filament splits the half-plane into two regions: the unbounded *top region* and the *bottom region*. We say that a filament $\varphi(u)$ is *nested in* a filament $\varphi(v)$ if $\varphi(u)$ is contained in the bottom region of $\varphi(v)$. We say that the robber *is/stays in a region* if he is/stays on a filament entirely contained in this region. The robber is *confined by* $\varphi(u)$ if a cop takes $\varphi(u)$ and the robber is in the bottom region of $\varphi(u)$.

Lemma 14. *Suppose that the robber is confined by $\varphi(u)$. Then he stays in the bottom region of $\varphi(u)$ as long as there is a cop on $\varphi(u)$.*

Proof. This is obvious since to move from one region to another, the robber has to use a filament $\varphi(v)$ which crosses $\varphi(u)$. But then v is a neighbor of u, and the cop captures the robber in the next turn. □

A filament $\varphi(u)$ is called *top in x* if it maximizes the value $\varphi(v)(x)$ over all filaments $\varphi(v)$ defined for x. Suppose that ℓ is the left-most and r is the right-most endpoint of the representation. We have a *sequence of top filaments* $\{\varphi(t_i)\}_{i=1}^{k}$ as we traverse from ℓ to r. We note that one filament can appear several times in this sequence. See Figure 2.2 for an example.

Let $\varphi(t_i)$ be top in x. Each filament $\varphi(t_i)$ together with the upward ray starting at $\big(x, \varphi(t_i)(x)\big)$ separates the half-plane into three regions: the *left region*, the *bottom region* and the *right region*. The key property is that there is no filament intersecting the left and right regions and avoiding $\varphi(t_i)$. Also note that the division of filaments into the regions is the same for all x in the same top part of $\varphi(t_i)$.

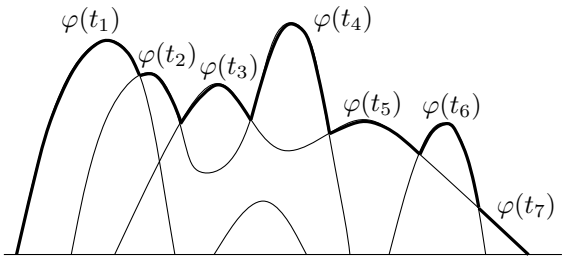


Figure 2.2: An example of a sequence of top filaments. Only the top part of each $\varphi(t_i)$ is depicted in bold. Note that $t_2 = t_4$ and $t_3 = t_5 = t_7$.

Lemma 15. *Suppose that a cop stands on $\varphi(t_i)$ and the robber is in the right region of $\varphi(t_i)$. If the cop moves to the neighboring $\varphi(t_j)$ with the highest index j, the robber cannot move to the left region of $\varphi(t_j)$.*

Proof. Let $\varphi(t_i)$ be on $[a, b]$ and $\varphi(t_j)$ on $[c, d]$ and we have $c < b$. Suppose that the cop moves from $\varphi(t_i)$ to $\varphi(t_j)$ and the robber stands on a filament $\varphi(u)$ defined on $[e, f]$. We know that $c < b < e < f$, so $\varphi(u)$ does not intersect the left region of $\varphi(t_j)$. And since $\varphi(t_j)$ is top, there is no path going to the left region which avoids $\varphi(t_j)$. So the robber cannot move there. □

2.4.2 Two cops' strategy

We show the strategy for two cops on IFA graphs, completing the proof of Theorem 12.

Proof of Theorem 12. We call one cop the *guard*, and the other one the *hunter*. The strategy proceeds in phases. Every phase starts with both cops on a filament $\varphi(u)$ such that the robber is confined by it. The guard stays on $\varphi(u)$ till the robber is either captured or confined by the hunter in some filament $\varphi(v)$ nested in $\varphi(u)$; so according Lemma 14 the robber can only move in the bottom region of $\varphi(u)$. If the confinement happens, the guard moves to the filament $\varphi(v)$ taken by the hunter, ending the phase. In the next phase the hunter proceeds with capture the robber inside the bottom region of $\varphi(v)$.

For the initial phase, we imagine that the guard takes some imaginary filament above all filaments of φ so the robber is confined to its bottom region, i.e., to the entire graph G. We can choose both cops to start the first phase at a filament $\varphi(v)$ with left-most left endpoint and therefore top in G.

Suppose that we are in some phase where the guard is placed on $\varphi(u)$. Let G_u be the subgraph of G induced by the vertices whose filaments are nested in $\varphi(u)$, and let C_u be the connected component of G_u containing the vertex occupied by the robber. Since the guard stays at $\varphi(u)$ till the robber is confined in some nested $\varphi(v)$, the strategy ensures that the robber must remain in C_u throughout the phase, because any vertex in the open neighborhood of C_u is adjacent to the vertex u guarded by the cop.

Let $\{\varphi(t_i)\}_{i=1}^{k}$ be the sequence of top intervals in the restriction of φ to the vertices of C_u. The hunter first goes to $\varphi(t_1)$. When he arrives to $\varphi(t_1)$, the robber cannot be in the left region of $\varphi(t_1)$ since there is no filament of C_u contained there. Now suppose that the hunter is in $\varphi(t_i)$ and assume the induction hypothesis that

the robber is not in the left region of $\varphi(t_i)$. If the robber is confined in $\varphi(t_i)$, the phase ends with the guard moving towards $\varphi(t_i)$. If the robber is in the right region of $\varphi(t_i)$, the hunter moves to the neighbor $\varphi(t_j)$ with highest index j. By Lemma 15 the robber cannot move to the left region of $\varphi(t_j)$ so he is either in the bottom or the right region. The robber cannot stay in the right regions forever since $\varphi(t_k)$ has no filament of C_u contained in the right region, so eventually the robber is confined in $\varphi(t_i)$ or captured directly.

Since there are only finitely many filaments nested in each other, the strategy proceeds in finitely many phases and the robber is eventually captured.

With a small modification we can prove that this strategy captures the robber in $\mathcal{O}(n)$ turns. Suppose that initially both cops are placed in the filament with the left-most endpoint ℓ and there are p phases. Let C_i be the graph the robber is confined to by the guard on u_i in phase i, so $C_i = G$ and let $C_{p+1} = \emptyset$. Let $D_i = C_i \setminus C_{i+1}$ and note that D_i contains all top filaments of C_i.

During the i-th phase the hunter moves to any top filament of C_i in at most 2 moves (note that there must be a filament in G which simultaneously intersects $\varphi(u_i)$ and a top filament of C_i), then to the left-most top filament of C_i in at most $|D_i|$ moves using a shortest path in D_i, then takes at most $|D_i|$ steps over the top filaments of C_i. Finally, when the hunter confines the robber in C_{i+1}, it takes the guard at most $|D_i| + 2$ steps to get to u_{i+1} by a similar argument. Since $\sum |D_i| = n$ and the number of phases is also bounded by n we have used $\mathcal{O}(n)$ turns.

The strategy applies to function, circle and circular arc graphs since they are subclasses of IFA graphs. Similarly, the lower bound from Observation 13 above extends to IFA graphs as a superclass. □

2.5 Outer string graphs

In this section, we show almost tight bound on the maximum cop number of outer-string graphs, introduced in Section 1.2. Namely, we show the following.

Theorem 16.

$$3 \leq \text{max-cn}(\mathsf{OUTER\text{-}STRING}) \leq 4.$$

We first establish some analogues to the method and terminology used in Section 2.4. The intuition behind the analogues is that all IFA graphs are also string graphs in the upper half-plane with both endpoints fixed to the x-axis[6], while outer-string graphs have just one endpoint fixed to the line.

2.5.1 String pairs and Regions

For a given outer-string representation of G, let $v_1, \ldots, v_n$ be the ordering of the vertices of G by the x-coordinates of the (unique) intersection of $\varphi(v_i)$ with the x-axis. We say that v_i is *left* of v_j if $i < j$ and similarly for *right*.

Every pair of intersecting outer-strings (v_i, v_j) divides the half-plane into at least two regions: the unbounded *top region*, the *bottom region* incident with an

[6]Note that filaments have an additional condition on x-monotonicity.

interval of the x-axis, and possibly other *middle regions*. The middle regions do not play any role in our proof, since no string is entirely contained in them and every string intersecting a middle region intersects $\varphi(v_i)$ or $\varphi(v_j)$. The strings entirely contained in the bottom region are *surrounded by* $\varphi(v_i)$ *and* $\varphi(v_j)$, a robber on a vertex surrounded by $\varphi(v_i)$ and $\varphi(v_j)$, each occupied by a cop, is *confined by* $\varphi(v_i)$ *and* $\varphi(v_j)$.

The following is a straightforward analogue of Lemma 14.

Lemma 17. *Suppose that the robber is confined by* $\varphi(v_i)$ *and* $\varphi(v_j)$*. Then he stays in the bottom region of* (v_i, v_j) *as long as there are cops on* v_i *and* v_j*.*

2.5.2 Four cop strategy

We show Theorem 16 by describing a strategy for 2 pairs of cops and an example of a graph requiring 3 cops.

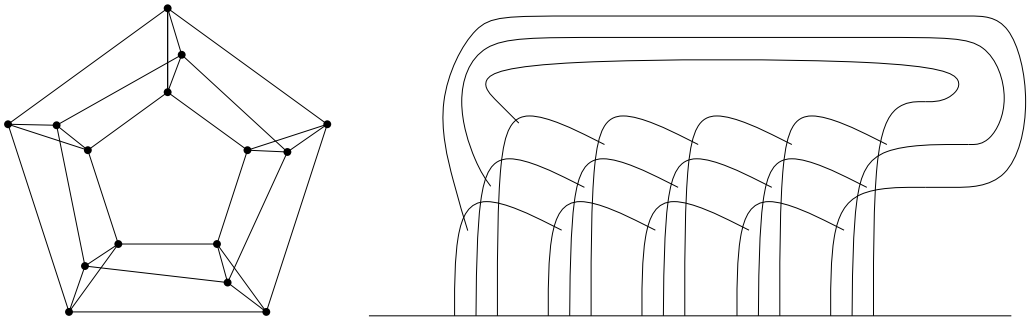


Figure 2.3: The 3-by-5 toroidal grid (left) and its outer-string representation (right).

Theorem 16. Figure 2.3 shows a connected outer string graph requiring three cops, as the robber can safely escape capture in any situation with two cops. Since the graph is vertex transitive, we only need to check the situations with the robber in a fixed vertex and at least one cop adjacent to the robber. However, in all cases there is at least one safe vertex adjacent to the robber.

The cops' strategy has a similar basic structure as the two-cop strategy for interval filament graphs described in Section 2.4. Among the four cops, there are two *guards* and two *hunters*. The strategy is divided into phases. During each phase, the two guards stand on a pair of intersecting strings $\varphi(v_i)$ and $\varphi(v_j)$ confining the robber.

In the beginning of the game, the two guards take an arbitrary pair x and y of adjacent vertices, and the two hunters take an arbitrary vertex. Then the robber takes a vertex r of his choosing, which we may assume is not in the closed neighborhood of x and y. We then fix an outer-string representation φ of G in which the string $\varphi(r)$ is surrounded by $\varphi(x)$ and $\varphi(y)$; it is not hard to see that such a representation exists.

In each phase, let (v_i, v_j) be the pair of adjacent vertices occupied by the guards. Let $G_{i,j}$ be the subgraph induced by the strings entirely contained in the bottom region. When the guards occupy a pair (v_k, v_l), $v_k, v_l \in G_{i,j}$, of adjacent

vertices confining the robber, they switch roles with the guards and the game continues with the next phase with strictly smaller graph $G_{k,l}$.

Let C be the connected component of $G_{i,j}$ containing the vertex taken by the robber at the beginning of the phase. We may assume that the robber only moves on the vertices of C, otherwise he will be caught by one of the guards.

The strings of $\varphi(C)$ partition the upper half-plane into several regions, of which exactly one is unbounded. We say that a vertex w of C is *external*, if the string $\varphi(w)$ has at least one point on the boundary of the unbounded region. Let X be the subgraph of C induced by the external vertices. Note that X is connected.

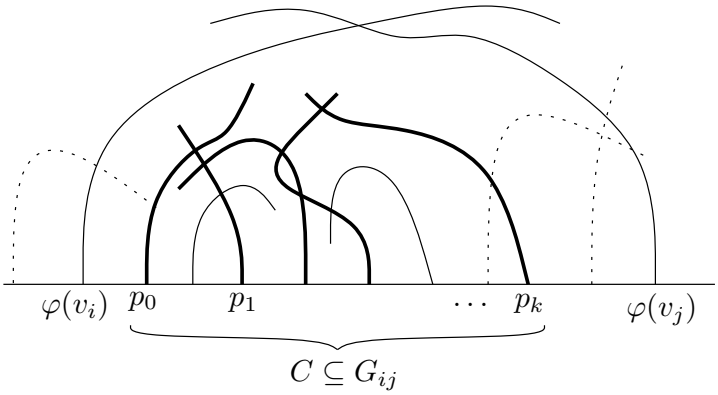


Figure 2.4: Illustration of the situation in one phase. Pair of strings $\varphi(v_i)$ and $\varphi(v_j)$ confines the robber. The bold strings are external in C, the dotted strings are not part of G_{ij} and are not safe for the robber.

Since the vertices of $G_{i,j}$ that are left and right of an external vertex v_a fall into different components of $G_{i,j} - N[v_a]$, we get:

Observation 18. *When the robber at v_r is confined by (v_i, v_j) and a cop occupies an external vertex v_a of $G_{i,j}$ left of v_r, the robber can not safely move to a v_b left of v_a.*

Let $P = p_0, p_1, \ldots, p_k$ be a path in X from the leftmost vertex p_0 of C to the rightmost vertex p_k of C. Note that $p_0, p_k \in X$. See Figure 2.4 for an illustration. Note that P need not be monotone with respect to the left-to-right ordering of strings.

The strategy of the hunters is to first occupy the two vertices p_0 and p_1. Then, throughout this phase the hunters will always occupy a pair of vertices p_i and p_{i+1} for some i, with the robber occupying a vertex that is to the right of p_i.

In the cops' turn with the hunters at p_i, p_{i+1}, either the robber is confined by the hunters and a next phase may begin, or he is right of p_{i+1}. In the second case the cops move with a single step to occupy p_{i+1}, p_{i+2}, let the robber move and repeat this step. Since the path P contains the rightmost vertex of C, the robber must at some point end up at a vertex between the two hunters, beginning a new phase. □

In fact, by an analogous argument as in the proof of Theorem 12, we may show that the strategy requires at most a linear number of moves.

2.6 String graphs

In this section, we show that the maximum cop number of string graphs is at most 15. Our strategy is inspired by the strategy for 3 cops in planar graphs [AF84]. The key difference is that we use Lemma 22 instead of Lemma 20 allowing us to guard curves within the string graph from being crossed, requiring 5 cops for each shortest path of the string-graph.

Theorem 19.

$$3 \leq \text{max-cn}(\mathsf{STRING}) \leq 15.$$

2.6.1 Guarding shortest paths

We say that a group of cops *guards a subgraph* $G' \subseteq G$ when these cops can play indefinitely such that whenever the robber would enter G', he would be captured in the next turn. Note that the cops in general have to react to robber's movement. Also, it may not be trivial for the cops to start guarding G', that is reach the appropriate positions.

We recall a lemma by Aigner and Fromme [AF84] for guarding a shortest path in a graph:

Lemma 20 (Lemma 4 in [AF84])**.** *Let G be a graph, and $P = \{p_0, p_1, \ldots, p_k\}$ be a shortest path. Then a single cop can, after a finite number of initial moves, guard P.*

This result turned out to be particularly useful for planar graphs where one can cut the graph by protecting several shortest paths. For intersection graphs forbidding the robber to visit vertices of P is not sufficient to prevent him from moving from one side of the part to the other. We need a stronger tool to geometrically restrict the robber. We get this by showing that in general graphs we can protect the closed neighbourhood of a given shortest path using five cops, preventing the robber from safely stepping on any string even crossing the protected path.

We first need one additional generalization of Lemma 20 – we protect paths which are not necessarily shortest in G, but are shortest from the point of the robber within a region he is already confined to. Below we combine this generalization with guarding path neighbourhood. We believe that these tools may be of some further interest.

We say that an *$u-v$ path P is shortest relative to $D \subseteq V$* if there is no shorter $u-v$ path in G using at least one vertex of D. Note that P itself may or may not go through D.

When our strategy makes sure that any time in the future of the game, whenever the robber leaves $D \subseteq V$ he is captured immediately, we say that the robber is *confined to D*. Note that this includes the case when the robber cannot even get outside D without being captured. If the robber would be immediately captured by moving to a vertex v, we say that the robber *can not safely move to v*.

Note that in the following we get exactly the original statement for $D = V$. Also, we could equivalently define a relative shortest path by having no shortcuts contained in D with the same results.

Lemma 21. *Let G be a graph, $u, v \in V$ and let P be a shortest $u-v$ path relative to $D \subseteq V$ with the robber confined to D. Then a single cop C can, after a finite number of initial moves, prevent the robber from safely entering P.*

Proof. The proof is analogous to the proof of Aigner and Fromme [AF84]. First, in case there is no $u - v$ path using at least one vertex in D we have $P \cap D = \emptyset$ and there is a 1-cut between P and D due to Menger's theorem. We let the cop guard this cut-vertex in this case, trivially getting our result.

Otherwise, let l be the length of a shortest $u - v$ path using a vertex in D. We prescribe a position for the cop on P for every robber's position $r \in D$. Let $P = (u = p_0, p_1, p_2, \dots p_k = v)$. We let the cop stand on p_a where $a = \mathrm{dist}(r, u)$. In case this would make the cop move off the ends of path P (e.g. to p_{k+1}) he should just stay at the nearest endpoint of P.

To initially get there, the cop first moves to u via any shortest path and then moves along P until his current vertex is the prescribed one. After that moment, $a = \mathrm{dist}(r, u)$ can change by at most 1 in one step and the cop can keep up with the prescribed position.

Now in case the robber would step on $p_i \in P \cap D$, we claim that $\mathrm{dist}(p_i, u) = i$ and the cop captures him immediately. Surely $\mathrm{dist}(p_i, u) \leq i$. Having $\mathrm{dist}(p_i, u) < i$ would mean that there is a $u - v$ path through p_i (and therefore D) that is strictly shorter than P, a contradiction. □

Lemma 22. *Let G be a graph, $u, v \in V$ and let P be a shortest $u-v$ path relative to $D \subseteq V$ with the robber is confined to D. Then five cops $C_{-2}, C_{-1}, C_0, C_1, C_2$ can, after a finite number of initial moves, guard $N[P]$.*

Proof. We assume a one cop's strategy $\mathcal{S}$ preventing the robber from safely entering P as given by Lemma 21 and describe the desired strategy for five cops. In the rest of the proof, we assume any cop happening to be adjacent to the robber will capture him immediately.

The cop C_0 moves according to $\mathcal{S}$. When C_0 moves to p_i, the cops C_j (for $j \in \{-2, -1, 1, 2\}$) move to p_{i+j}. This is always possible since $\mathcal{S}$ moves C_0 to distance at most one and on the path P. In case a cop would have to move beyond an endpoint of P, he just stays on the endpoint. This way the cops always occupy five consecutive vertices of the path or share an endpoint of the path.

Now assume that the robber moves to a vertex r. If r is adjacent to a vertex q, which is in turn adjacent to p_i, strategy $\mathcal{S}$ necessarily moves C_0 to one of $p_{i-2} \dots p_{i+2}$, since otherwise the robber could step on P in two moves without being immediately captured by C_0, which would contradict the properties of $\mathcal{S}$. Therefore, after the cops' move, there is at least one cop on p_i, and so if the robber moves to q, he is captured immediately.

The initial setup procedure is analogous to the one in Lemma 21 for all five cops. □

To guard $N[P]$ with cops only moving on P we can show that five cops are necessary as shown in Fig. 2.5. With the robber on r, there needs to be a cop on each of the vertices $p_{i-2} \dots p_{i+2}$ or the robber could safely move to $N[P]$ in the next turn.

In the following, when we say "start guarding a path", we do not explicitly mention the initial time required to position the five cops onto the path and assume that the strategy waits for enough turns.

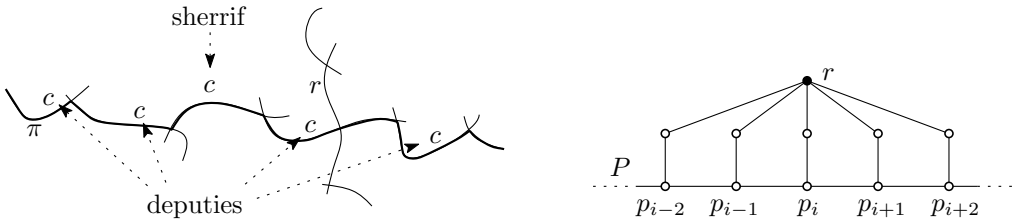


Figure 2.5: **a)** Situation when guarding shortest curve π defined by path P: Five cops guard consecutive strings of π, string r crossing π may not be in P, but is contained in $N[P]$. **b)** The necessity of five cops to guard $N[P]$. With the robber standing on r, there needs to be a cop on each of the vertices $p_{i-2}, \ldots, p_{i+2}$, otherwise the robber could safely move to $N[P]$.

2.6.2 Guarding shortest curves

Since our strategy for string graphs is partially geometric, we introduce the concept of shortest curves as particular curves through the string representation of a shortest path. Note that below we consider any curves sharing only their endpoints to be disjoint.

Let G be a string graph together with a fixed string representation φ, robber confined to $D \subseteq V$ and P a shortest $u - v$ relative to D. Suppose that we choose and fix two points $\pi_u \in \varphi(u)$ and $\pi_v \in \varphi(v)$. Let $\pi_{uv} \subseteq \varphi(P)$ be a curve from π_u to π_v such that $\pi_{uv} \subseteq \bigcup_{p \in P} \varphi(p)$ and for every $p \in P$ π_{uv} has a connected intersection with $\phi(p)$ and these correspond to the points of P in the same order. We call π_{uv} a *shortest curve of P (relative to D)* with endpoints π_u and π_v. A curve π is called a *shortest curve (relative to D)* if it is a shortest curve of some shortest path. We leave out D if $D = V$ or it is clear from the context.

The shortest path in the graph corresponding to a shortest curve π is uniquely defined by the sequence of strings that intersect π on a substring of non-zero length. To *guard a shortest curve* π means to guard its corresponding shortest path. The number of its strings is the *length* of π. Note that the Euclidean length of π plays no role in this paper.

Corollary 23. *Let G be a string graph together with a string representation φ and let π be a shortest curve relative to D such that the robber is confined to D. Then five cops can (after a finite number of initial moves) prevent the robber from entering any string intersecting π.*

Proof. Let P be the shortest path such that π is a shortest curve of P. By guarding $N[P]$, the cops prevent the robber from entering strings intersecting π. See Figure 2.5 for an illustration. □

We also note the following easy observation.

Observation 24. *Any sub-curve (continuous part) of a shortest curve (relative to D) is also a shortest curve (relative to D).*

2.6.3 Segments, faces and regions

For a given string graph G and its string representation φ let the *faces* (of φ) be the open arc-connected regions of $\mathbb{R}^2 \setminus \varphi(G)$, and let a *closed face* be the closure

of a face. As we assume that the number of intersections of φ is finite, the number of faces is also finite. Note that every face is an open set.

A *segment* of string π is a part of the string not containing any intersection with another string between either two intersections, an intersection and an endpoint, or two endpoints. Note that the number of segments is also finite. A *region* is a closed subset of $\mathbb{R}^2$ obtained as a closure of a union of some of the faces.

Denote all vertices internal to a region B by $V_B = \{v \in V \mid \varphi(v) \subseteq \text{int}(B)\}$. We use the following topological result, following from the previous section and Corollary 23.

Proposition 25. *If there is $D \subseteq V$ such that the cops guard disjoint shortest curves π_1 and π_2 (relative to D) between points π_u to π_v such F is the closed face of $\mathbb{R}^2 \setminus (\pi_1 \cup \pi_2)$ containing the robber's string and D is the component of V_F containing the robber, then the robber may not safely leave D.*

Additionally, below we use the following topological lemma.

Lemma 26. *Given two disjoint simple $\pi_u - \pi_v$ curves π_1 and π_2 in $\mathbb{R}^2$, $\pi_u \neq \pi_v$, let F be one of the faces of $\mathbb{R}^2 \setminus (\pi_1 \cup \pi_2)$. For any simple $\pi_u - \pi_v$ curve π_3 contained in* $\text{clos}(F)$ *and going through at least one of its inner points we have that then every face of $F \setminus (\pi_1 \cup \pi_2 \cup \pi_3)$ is bounded by some simple and internally disjoint curves π'_i and π'_3 with $\pi'_i \subseteq \pi_i$, $i \in \{1, 2\}$ and $\pi'_3 \subseteq \pi_3$.*

Proof. For our illustrations we assume that without loss of generality, F is the inner face of $\pi_1 \cup \pi_2$, potentially using circular inversion to attain that. Let R be any (open) face of $F \setminus (\pi_1 \cup \pi_2 \cup \pi_3)$ and $B = \text{clos}(R) \setminus R$ be its boundary. We have that R is arc-connected from definition and so B is a simple closed Jordan curve.

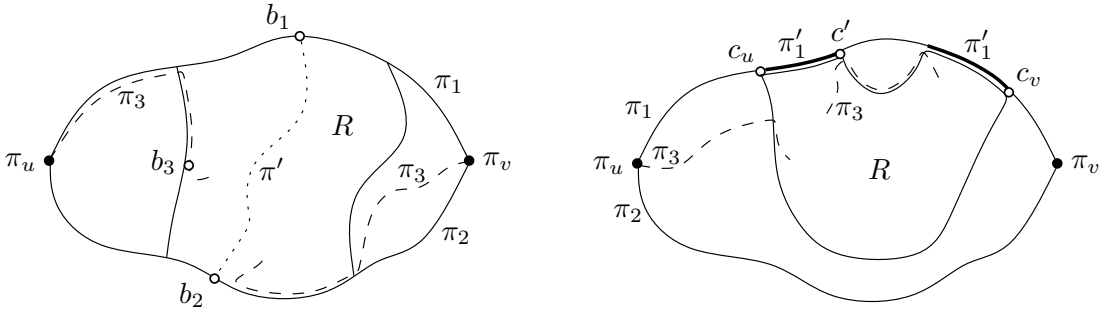


Figure 2.6: *Left:* Illustration of the situation in proof of Lemma 26, showing that there can not be both b_1 and b_2 as in the proof. *Right:* Illustration how a disconnected π'_1 (bold line) would imply π_3 crossing R.

We first establish that $B \subseteq (\pi_i \cup \pi_3)$ for some $i \in \{1, 2\}$. Observe that $B \subseteq \pi_j$ for $j \in \{1, 2, 3\}$ would imply that π_j is not a simple curve. Moreover, there is a point $b_3 \in B \cap \pi_3$ with $b_3 \notin (\pi_1 \cup \pi_2)$ as otherwise we would have R bound by just $\pi_1 \cup \pi_2$ and we would have $R \cup B = F$, a contradiction with $\pi_3 \cap \text{int}(F) \neq \emptyset$.

Now it can not be the case that there are points $b_1 \in B \cap \pi_1$ with $b_1 \notin (\pi_2 \cup \pi_3)$ and $b_2 \in B \cap \pi_2$ with $b_2 \notin (\pi_1 \cup \pi_3)$. If that was the case, there would be a $b_1 - b_2$ curve $\pi' \subseteq (R \cup \{b_1, b_2\})$ separating π_u from π_v in F and not intersecting π_3 (including the endpoints), contradicting π_3 being a $\pi_u - \pi_v$ curve through F. See

Figure 2.6 for an illustration. However, we necessarily have one such b_i, $i \in \{1, 2\}$. Without loss of generality assume that we have such b_1, there is no such b_2 and therefore $B \subseteq \pi_1 \cup \pi_3$.

Let $\pi'_1 = \text{clos}((B \cap \pi_1) \setminus \pi_3)$. Let c_u be the first point of π'_1 going along π_1 from π_u to π_v and c_v last such point. If π'_1 was not connected, let c' be an endpoint of one segment of π'_1 different from c_u, c_v. We would necessarily have $c' \in \pi_3$ since B (as a curve) can be composed from sub-curves of π_1 and π_3. However, is topologically impossible for π_3 to enter c' without going through R, $\mathbb{R}^2 \setminus F$ or π'_1, as illustrated in Figure 2.6. Therefore π'_1 is connected and we can take $\pi'_3 = \text{clos}(B \setminus \pi'_1)$, getting a connected curve $\pi'_3 \subseteq \pi_3$. □

2.6.4 Restricted graphs and strategies

Given a closed region $B \subseteq \mathbb{R}^2$, let G *restricted to* B, denoted $G \restriction_B$, be the intersection graph of the curves of $\varphi \cap B$. This operation may remove vertices (for entire strings outside B), remove edges (crossings outside B) and it also splits each vertex v whose string $\varphi(v)$ leaves and then re-enters B at least once. In the last case, every arc-connected part of $\varphi(v) \cap B$ spans a new vertex v_i. The new vertices are also called the *splits* of v. The new graph is again a string graph with representation denoted $\varphi \restriction_B$ obtained from φ as above. Note that this operation preserves the faces and strings in $\text{int}(B)$ and all representation properties assumed above, namely the vertex set of $G \restriction_B$ is finite. Also, the number of segments does not increase.

Lemma 27. *Let B be a region. If π is a shortest curve (optionally relative to D with the robber confined to D) and $\pi' \subseteq \pi$ is a sub-curve with $\pi' \subseteq B$, then π' is a shortest curve (relative to D) in $G \restriction_B$ and $\varphi \restriction_B$.*

Proof. This follows from Observation 24 and the fact that underlying path of π' is preserved (and if any $p \in P'$ got split into $\{p_i\}$ we use the p_i intersecting π') and no path (e.g. through D) can get shortened by a restriction. □

We now show that a strategy for a restricted graph may be used in the original graph.

Lemma 28. *Let B be a region. If there is a cop's strategy $\mathcal{S}'$ eventually capturing a robber in $G \restriction_B$ confining him to V_B then there is a strategy $\mathcal{S}$ for the same number of cops capturing the robber on G confining him to V_B.*

Proof. The strategy $\mathcal{S}$ plays out as $\mathcal{S}'$ except when $\mathcal{S}'$ would move a cop to a split $v_i \in V_{G\restriction_B}$ of $v \in V_G$, $\mathcal{S}$ moves the cop to v. Note that all such moves are possible. Robber's choices while internal to B are not extended in any way.

Assume the robber moves from internal u to non-internal v, which is split to $v_1, \dots v_k$ in $G \restriction_B$. Note that at least one of $vv_1, \dots v_k$, say v_i, is adjacent to u in $G \restriction_B$, as $\varphi(u)$ has to intersect $\varphi(v)$ in $\text{int}(B)$. Let $\mathcal{S}$ play as $\mathcal{S}'$ would if the robber moved to v_i, capturing him with this move as assumed in the statement. □

2.6.5 Proof of Theorem 19

We are ready to prove that the maximum cop number of string graphs is at most 15.

Theorem 19. Our strategy proceeds in phases, monotonously shrinking the confinement area of the robber. In the beginning of each phase, the robber is confined to $D \subseteq V$ either (A) by a single cop guarding a cut-vertex separating D from the rest of the graph, or (B) by 10 cops guarding two shortest curves forming a simple (non-self-intersecting) cycle surrounding D. In each phase we either decrease the number of the segments of φ, or keep the number of segments and decrease $|D|$.

Let B be the union of the currently guarded paths by Lemma 23. Let D be the component of $G \setminus N[B]$ containing the vertex with the robber and let $Q = N[B] \cap N[D]$. Since our strategy confines the robber to D for the rest of the game we can assume that $V = D \cup Q \cup B$. Let s be the number of segments of φ.

Claim 29. *Let $V = D \cup Q \cup B$, the robber stands on $r \in D$, and one of the following holds:*

(A) 1 cop guards a vertex $c \in B$, $|B| = 1$.

(B) 10 cops guard two shortest curves π_1 and π_2 relative to D between points a to b such that $\pi_1 \cup \pi_2$ forms a simple cycle, $|B| \geq 2$ and additionally $G = G|_F$ where F is the closed face of $\mathbb{R}^2 \setminus (\pi_1 \cup \pi_2)$ containing $\varphi(r)$.

Then 15 cops have a strategy to capture the robber.

Proof of Claim 29. We prove this claim by induction on s and then on $|D|$, the claim obviously holds for either $s \leq 1$ or $|D| = 0$. The strategy proceeds differently according to which of the assumptions (A) and (B) is satisfied.

Case (A). If $Q = \{q\}$, then move the cop guarding c to start guarding q. Let $G' = G \setminus \{c\}$ while also leaving out any irrelevant vertices to have $V' = D' \cup Q' \cup \{q\}$ as above. The rest follows from the induction hypothesis, with the assumption (A), applied to G' with both smaller s' and $D' \subsetneq D$.

If $Q = \{q_1, \ldots, q_k\}$ for $k \geq 2$, let $G' = G \setminus \{c\}$ and let π_{q_i} be any point of $\varphi(c) \cap \varphi(q_i)$. Now let π_1 be a shortest curve between some π_{q_i} and π_{q_j}. We let $\pi_2 \subseteq \varphi(c)$ be the part of $\varphi(c)$ between π_{q_i} and π_{q_j}.

However, $\pi_1 \cup \pi_2$ may not be a simple cycle. Let $\pi_u = \pi_{q_i}$ and let π_v to be the first point of $\pi_1 \cup \pi_2$ along π_2 going from π_u. Note that if there is no other intersection then $\pi_v = \pi_{q_j}$. Now let π_1' and π_2' be the parts of π_1 and π_2 between π_u and π_v, forming a simple cycle.

Let $G'' = G|_F$ where F is the closed face of $\mathbb{R}^2 \setminus (\pi_1 \cup \pi_2)$ containing $\varphi(r)$. Remove any irrelevant vertices from G'' to have $V'' = D'' \cup Q'' \cup B''$ as above and use claim case (B) for smaller $D'' \subsetneq D$ (as P_1 has a neighbour in D) and not increased $|s''|$. Note that B'' uses at least one vertex other than c.

Case (B). If there is no $\pi_u - \pi_v$ path through a vertex of D then, according to Menger theorem, there must be a cut-vertex $c \in B \cup Q$ separating D from B. Let one cop guard c and then stop guarding B. Let $G' = G \setminus (B - c)$ while also leaving out irrelevant vertices to have $V' = D' \cup Q' \cup \{c\}$ as above. We then use claim case (A) for G' with smaller s' and $D' \subseteq D$.

If there is a $\pi_u - \pi_v$ path through a vertex of D, let π_3 be shortest such curve. Note that it is a shortest curve relative to D. Let five cops start guarding π_3 and then let F be the closed face of $\mathbb{R}^2 \setminus (\pi_1 \cup \pi_2 \cup \pi_3)$ containing the robber string. According to Lemma 26 we have that F is delimited by disjoint π_i' and π_j' where $i = 3$ or $j = 3$ and $\pi_i' \cup \pi_j'$ form a simple cycle. We let the cops stop

guarding π_k where $k \notin \{i, j\}$ and restrict the guarding of π_i and π_j to π'_i and π'_j as in Proposition 24. $|B| \geq 2$ as one vertex string can not form a closed loop.

Let $G' = G|_F$ while also removing any irrelevant vertices from G' to have $V' = D' \cup Q' \cup B'$ as above. We then use claim case (B) for G' with non-increased number of segments ($s' \leq s$) and $D' \subsetneq D$. By Lemma 28, the strategy on G' from the induction hypothesis implies a strategy on G. □

The theorem follows by guarding an arbitrary vertex c with one cop, so $B = \{c\}$. We leave out the irrelevant vertices, so $V' = D \cup Q \cup B$. We use Claim 29 with the assumption (A) for $G' = G|_{V'}$. □

2.7 Strings on surfaces

In this section, we generalize the results of the previous section to graphs having a string representation on a bounded genus surface, and we show the following theorem.

Theorem 30. *Let G be a connected graph with a string representation φ on a surface $\mathbf{S}$ of genus g. Then $15 + 10g$ cops have a strategy to capture the robber on G.*

In Lemma 36 below, we also show that there are graphs of genus g requiring $\Omega(g^{\frac{1}{3}})$ cops to capture the robber. See Section 2.7.3 for both proofs.

We assume familiarity with basic topological concepts related to curves on surfaces, such as genus, non-contractible closed curves, the fundamental group of surfaces and graph embedding properties. A suitable treatment of these notions can be found, e.g., in Prasolov's book [Pra] and the book by Mohar and Thomassen [MT01].

2.7.1 Topological preliminaries

We recall several definitions and topological tools.

A *walk* in a graph G is a sequence of vertices $W = w_0, w_1, \ldots w_k$ with w_i and w_{i+1} adjacent; repetitions of vertices and edges are allowed. A *closed walk* is a walk with $w_0 = w_k$. Let $|W| = k$ denote the length of the walk (both open and closed), i.e., the number of steps between vertices along the walk.

Given two walks $W = w_0, w_1, \ldots, w_k$ and $W' = w'_0, w'_1, \ldots, w'_\ell$ with $w_k = w'_0$, we let $W + W'$ denote their concatenation $w_0, w_1, \ldots, w_k, w'_1, w'_2, \ldots, w'_\ell$. Let $-W$ be W with reversed vertex order and let $W_1 - W_2 = W_1 + (-W_2)$.

A curve π is a continuous function from the interval $[0, 1]$ to the surface and carries its direction information. The concatenation of curves $\pi_1 + \pi_2$ is defined naturally whenever $\pi_1(1) = \pi_2(0)$, that is the curves connect. Similarly to walks above, $-\pi$ is the reversed curve and $\pi_1 - \pi_2 = \pi_1 + (-\pi_2)$.

We use the following topological lemma, which directly follows from the properties of the fundamental group. We omit the topological group theory introduction as well as its proof and refer the interested reader to Prasolov's book [Pra] on the subject.

Lemma 31. *Let π_1, π_2 and π_3 be three curves on a surface $\mathbf{S}$, all sharing the same endpoints x and y and oriented from x to y. If the closed curve $\pi_1 - \pi_2$ is non-contractible, then at least one of $\pi_1 - \pi_3$ and $\pi_2 - \pi_3$ is non-contractible as well.*

Let G be a graph with a string representation φ on a surface $\mathbf{S}$. We represent the combinatorial structure of φ by an auxiliary multigraph $A(G)$ embedded on $\mathbf{S}$ and defined as follows: the vertices of $A(G)$ are the endpoints of the strings of φ and the intersection points of pairs of strings of φ, and the edges of $A(G)$ correspond to segments of strings of φ connecting pairs of vertices appearing consecutively on a string of φ. By representing φ by its auxiliary multigraph $A(G)$, we will be able to apply the well-developed theory of graph embeddings on surfaces.

We introduce a relation between a walk in G and a curve on $\mathbf{S}$, allowing us to easily transition between the two, leading to an easy proof of the following lemmas and Theorem 30.

We say that a (closed) walk $W = w_0, w_1, \dots w_k$ in G *imitates* a (closed) curve $\pi \subseteq \varphi[G]$ on the surface $\mathbf{S}$ if π can be partitioned into a sequence of consecutive segments $\pi_0, \pi_1, \dots, \pi_k$ of positive length such that $\pi = \sum_{i=0}^{k} \pi_i$ and $\pi_i \subseteq \varphi(w_i)$ for each $i = 0, \dots, k$. A closed walk W *imitates a non-contractible curve* if there is a non-contractible curve $\pi \subseteq \varphi[G]$ imitated by W.

Lemma 32. *Let φ be a string representation of a connected graph G on a surface $\mathbf{S}$ of genus $g > 0$ and let W be a closed walk in G imitating a non-contractible curve. Then every connected component of the graph $G' = G - N[W]$ has a string representation on a surface of genus at most $g - 1$.*

Proof. Consider the auxiliary multigraph $A(G)$ corresponding to the string representation φ. If $A(G)$ also has an embedding on a surface $\mathbf{S}'$ of genus $g - 1$, then G has a string representation on $\mathbf{S}'$, and we are done. Suppose then, that this is not the case, i.e., $A(G)$ is a graph of genus g, and therefore its embedding on $\mathbf{S}$ is a 2-cell embedding, that is every face of $\mathbf{S} - \varphi$ is homeomorphic to a disk.

Let π be the noncontractible curve imitated by W. The curve π traces a closed walk W' in $A(G)$. Since π is noncontractible, W' contains a noncontractible simple cycle C of $A(G)$. Define the multigraph $A' = A(G) - C$. By standard results on 2-cell embeddings (see [MT01, Chapter 4.2]), the genus of every connected component of A' is strictly smaller than the genus of $A(G)$.

Consider now the string representation $\varphi[G']$ of the graph $G' = G - N[W]$. Its auxiliary multigraph $A(G')$ is a subgraph of A', and hence each of its connected components has an embedding on a surface of genus $g - 1$. This embedding corresponds to a string representation of a connected component of G' on a surface of genus $g - 1$. □

Lemma 33. *If a graph G has no string representation in the plane, then for every string representation φ of G on a surface $\mathbf{S}$ there is a closed walk W in G imitating a non-contractible curve.*

Proof. Let $A(G)$ be the auxiliary multigraph corresponding to the string representation φ. Since $A(G)$ is not planar, the embedding of $A(G)$ contains a noncontractible cycle (see [MT01, Chapter 4.2]), which corresponds to a noncontractible curve on $\mathbf{S}$. This curve is imitated by a closed walk W of G. □

Lemma 34. *Let φ be a string representation of G on a surface $\mathbf{S}$, let $u, v \in V$ be two vertices, and let W_1, W_2, W_3 be three u-v-walks of G. If $W_1 - W_2$ imitates a non-contractible closed curve, then at least one of $W_1 - W_3$ and $W_2 - W_3$ imitates a non-contractible closed curve.*

Proof. Let π_{12} be a non-contractible closed curve imitated by $W_1 - W_2$. Looking at the consecutive segments of π_{12} corresponding to vertices of $W_1 - W_2$ (from the imitation), we have that $\pi_{12} = \pi_1 - \pi_2$ with π_1 imitated by W_1 and π_2 imitated by W_2. Let $x \in \varphi(u)$ be the first point of π_1 and $y \in \varphi(v)$ be the last point of π_1.

Now let π_3 be any x-y curve imitated by W_3 and observe that $\pi_1 - \pi_3$ is imitated by $W_1 - W_3$ and $\pi_2 - \pi_3$ is imitated by $W_2 - W_3$. By Lemma 31, at least one of $\pi_1 - \pi_3$ and $\pi_2 - \pi_3$ is non-contractible and the lemma follows. □

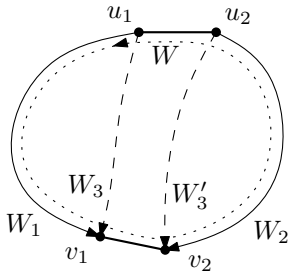


Figure 2.7: An illustration of the situation in the proof of Lemma 35.

2.7.2 Guarding non-contractible closed walks

The following lemma is the main tool to generalise our results to higher-genus graphs.

Lemma 35. *On a graph G with a string representation φ on a surface $\mathbf{S}$ and a shortest closed walk W imitating a non-contractible curve, 10 cops have a strategy to guard $N[W]$ after a finite number of initial moves.*

Proof. First, if $|W| \leq 10$, the cops may occupy every vertex of W for the rest of the game and we are done. Otherwise we divide W into two almost-equally long walks W_1, W_2 and two edges u_1u_2, v_1v_2 with $|W_1| \geq |W_2| \geq |W_1| - 1$, such that $W = W_1 + v_1v_2 - W_2 - u_1u_2$. See Figure 2.7 for an illustration. Note that $|W| = |W_1| + |W_2| + 2$

We claim that both W_1 and W_2 are shortest paths in G. If W_1 is not a shortest u_1-v_1-path, let W_3 be an u_1-v_1-walk with $|W_3| < |W_1|$. Then both closed walks $W_1 - W_3$ and $W_3 + v_1v_2 - W_2 - u_1u_2$ would be shorter than W:

$$|W_1 - W_3| = |W_1| + |W_3| < 2|W_1| \leq |W_1| + |W_2| + 1 < |W|,$$

$$|W_3 + v_1v_2 - W_2 - u_1u_2| = |W_2| + |W_3| + 2 < |W_1| + |W_2| + 2 = |W|$$

and at least one of these two must be non-contractible by Lemma 34, a contradiction with the choice of W. Similarly, a u_2-v_2-walk W_3' with $|W_3'| < |W_2|$ would give us that both $W_1 - W_3'$ and $W_2 - W_3'$ are shorter that W and at least one is non-contractible, a contradiction.

Therefore we may use Lemma 21 (with $D = V$) to guard $N[W_1]$ with 5 cops and guard $N[W_2]$ with the other 5 cops after a finite number of initial moves, guarding both curves imitated by W_1 and W_2 as in Lemma 23. □

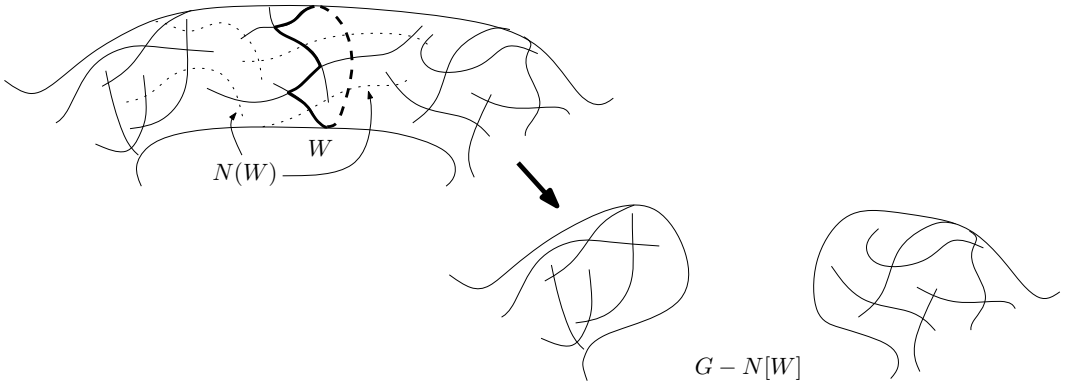


Figure 2.8: An illustration of cutting a surface handle after guarding $N[W]$ (dotted), where W is the closed path imitating a non-contractible curve (bold line) as in Lemma 35. Note that the non-contractible curve going once around a handle is just one of the possibilities, however the proof does not depend on the particular case.

2.7.3 The game on higher genus surfaces

Now we are ready to show that $15 + 10g$ cops win the game on genus g string graph, but first we show an easy lower bound on the number of cops required for both string graphs on genus g surface and graphs of genus g.

Lemma 36. *For every $g > 0$ there is a graph of genus at most g and cop number at least $\frac{1}{2}g^{\frac{1}{3}}$.*

Proof. Consider G_g to be the incidence graph of a projective plane of order $k = \lceil \frac{1}{2}g^{\frac{1}{3}} \rceil$. It has at most than $2k^3$ edges, so the genus is at most $2k^3 \leq g$. Its girth is 6 and its degrees are $k + 1$ which implies that the cop number is at least $k + 1$: Let r be the robber's vertex; every of the k cops is adjacent to or present at most one vertex of $N(r)$, and the robber can move away from the cops in every turn. Therefore, $\mathrm{cn}(G_g) \geq k + 1 \geq \frac{1}{2}g^{\frac{1}{3}}$. □

Theorem 30. We proceed by induction on the genus g, where the case $g = 0$ is proved in Theorem 19. Suppose that $g > 0$, and fix a string representation of G on a surface of genus g. Let W be a shortest closed walk in G imitating a non-contractible curve. See Fig. 2.8 for an illustration.

By Lemma 35, 10 cops may, after a finite amount of moves, prevent the robber from entering $N[W]$. The first part of the cops' strategy is to designate a group of 10 cops that will spend the entire game guarding $N[W]$. Thus, after a finite number of moves the robber will remain confined to a single connected component K of the graph $G' = G - N[W]$.

By Lemma 32, the graph K has a string representation on a surface of genus at most $g - 1$, and by induction, $15 + 10(g - 1)$ cops have a strategy to capture the robber on K. Thus, $15 + 10g$ cops will capture the robber on G. □

2.8 Guarding retracts and grids

In this section we show a generalisation of Lemma 20 and show one of its applications to d-dimensional grid graphs. This material is not directly used in the later sections but gives a broader view and a strong tool. The following lemma is likely to have implicitly appeared earlier in some form, but we were not able to find it in this or similar form.

Lemma 37. *Let H be a retract of G. Then $\mathrm{cn}(H)$ cops can, after a finite number of turns, position one of them such that this cop can guard H from the robber indefinitely. After the positioning, the other $\mathrm{cn}(H) - 1$ cops are free for other tasks.*

Proof. Let $f : G \to H$ be the retraction map. The $\mathrm{cn}(H)$ cops play on H as if a robber on $r \in V(G)$ would be on $f(r)$. Note that $f(r)$ moves with speed at most 1 so the strategy will "capture" the projection $f(r)$ of the robber in finitely many turns. The cop on $f(r)$ in that moment then can intercept the robber – that is be on $f(r')$ where r' is the robber's position. Therefore, if the robber steps on $v \in V(H)$, retract condition ensures $f(v) = v$ and the robber would be captured immediately. Note that the remaining $\mathrm{cn}(H) - 1$ cops are not required for the interception. □

Now Lemma 20 is an easy corollary, since any shortest path is a retract and paths have cop number 1. For a retract map to a shortest path $p_0, p_1, \dots p_k$ we may take $f(v) = p_{\min\{\mathrm{dist}(p_0,v),k\}}$.

Note that unfortunately, it is not straightforward to extend this to neighbourhoods of general retracts as we do in Section 2.6.2 below by keeping "deputy cops" in the second neighbourhood of $f(r')$: even if the retract itself has bounded degree (and therefore the number of deputies required for the retract vertices in distance ≤ 2 of $f(r')$ is bounded) it is not always possible to move the deputy cops together with the cop following $f(r')$ in the required way. See Section 2.6.2 for more details.

2.8.1 Guarding d-dimensional Grids

As an application of the above, and an indication that some d-dimensionally represented graphs might have a bounded cop number even for $d \geq 3$, we bound the cop number of finite d-dimensional grids.

A d-dimensional grid of dimensions $(n_1, n_2, \dots n_d)$ has the integral points of a $n_1 \times \cdots \times n_d$ cube as the vertex set, that is $V = \{(x_1, x_2, \dots x_d) \mid \forall 1 \leq i \leq d : 1 \leq x_i \leq d_i\}$, and edges between every two points in distance 1.

Lemma 38. *A finite d-dimensional grid graph has cop number at most d.*

Proof. We show this by induction on d. A 1-dimensional gird graph is a path with cop number 1. If G is a finite d-dimensional grid graph, let H_i be subgraph on the the vertices with $x_d = i$, that is a $d-1$-dimensional grid graph. Note that every H_i is a retract of G and using Lemma 37, $d - 1$ cops can position one of them to guard H_i.

The cops' strategy starts by guarding H_0 as above, and then in turn starts guarding $H_1, H_2 \dots H_{n_d}$. Whenever some cop successfully starts guarding H_i, the cop guarding H_{i-1} up to that point is free to join the $d-2$ others to start guarding H_{i+1}, therefore only d cops are needed at any point. Since at least one of H_i is guarded at every point, the robber may not safely enter a vertex with $x_d \leq i$, and is eventually captured. □

2.9 Remarks and open problems

There are several open problems concerning the game and maximum cop number of several classes. While all intersection graphs of arc-connected planar (and bounded genus) regions have bounded cop number, the bounds of the individual classes are not very tight. Especially better examples of lower bounds on cop number are scarce. One long-standing well-known problem is the maximum cop number of toroidal graphs, known to be 3 or 4.

It still remains to decide whether other intersection classes have bounded cop number, such as bounded boxicity graphs: Boxicity k graphs (denoted BOXICITY-k) are the intersection graphs of axis-aligned boxes in $\mathbb{R}^k$. There are interesting connections to some well-known classes: Interval graphs are exactly boxicity-1 graphs, every outer-string graph has boxicity at most 2 [Sch84], and every planar graph has boxicity at most 3 [Tho86]. The maximum cop number of boxicity-2 graphs is bounded by the string graphs bound, but the maximum cop number of boxicity-3 is widely open.

Let us remark that one can use our results as a tool or a polynomial time heuristic to prove that a given graph is not a string graph (resp. an outer-string graph, resp. an interval filament graph), by showing that the graph's cop number is more than 15 (resp. 4, resp. 2), which can be done in polynomial time.

For instance, this shows that a graph of girth 5 and minimum degree 16 is not a string graph, since such a graph would have cop number at least 16: in any position of 15 cops with the robber on v, at least one neighbor of v is not adjacent to a cop.

Chapter 3

Cops, fast robber and defensive domination on interval graphs

In this chapter, we examine cop and robber games with different player speeds, focusing in particular on games on interval graphs. As a tool and an independently interesting problem, we introduce and study $\mathcal{A}$-*defensive domination* in Section 3.3 together with the definitions, history and our results.

The results and techniques of the previous chapter and the cop and robber game in general often depend on the fact that the cops and the robber have exactly the same speed. While making the robber slower usually allows one cop to capture him, a faster robber is a different challenge (as well as closer to any security settings). As we shall see, a game with a robber with infinite speed behaves like a game where the robber's position does not matter and the robber merely chooses components of the graph split by the cops.

The various games with fast robber seem to have more connections to established game parameters, such as tree-width and tree-depth, as outlined in Section 1.4. The main problem is still to decide the number of cops required to win on a given graph and, optionally, to find a winning strategy. Frequently, and perhaps due to those connections, the games are generally computationally difficult and mostly tractable only on special graph classes.

While studying the game on interval graphs, one sub-problem pops up repeatedly and we present it as a separate concept of $\mathcal{A}$-defensive domination with its own history, motivation and solution on interval graphs.

Section 3.1 formally introduces the game and our results. Section 3.2 presents an overview of history and some interesting results related to the game. Section 3.3 introduces and solves the problem of $\mathcal{A}$-defensive domination on interval graphs. Section 3.4 introduces and analyses an auxiliary *barrier game* and finishes the proof of the main result of this chapter. Section 3.5 presents some general remarks and open problems.

3.1 Cops and fast robber game

The *cops and s-fast robber game* is closely related to the cops and robber game introduced in Section 2.1 and falls nicely within the games outlined in Section 1.3.1.

The game is played by two players, one controlling k cop tokens (or just cops), the other controlling one robber token (or just the robber), on a given simple undirected graph G. Additionally, the robber speed $s \geq 1$ is given. The tokens move on the vertices of G and several cops may share a vertex.

Initially, the first player positions k cops arbitrarily on the vertices of G, the second player then chooses a starting vertex for the robber according to the distribution of the cops. Then in every *round*, the players alternate in turns: On *cop-turn*, every cop may move to distance at most one. Such move to distance at most one is called a *step*. On *robber-turn*, the robber may do s steps avoiding all the cops, or equivalently move along a path of length at most s in $G - C$, where C is the set of vertices occupied by the cops.

The cops win when a cop steps on a robber (or when the robber has no available starting vertex, or when the graph is empty). Note that a robber is not allowed suicide by moving on a cop. Also, both players have a complete information about G and the game state at all times.

The number of cops required to capture the s-fast robber in a graph G is denoted by $\mathrm{cn}_s(G)$.

In the special case $s = \infty$ the robber may move to an arbitrary vertex of his present component of $G \setminus C$ where C is the set of vertices taken by the cops. Equivalently, the robber may choose any vertex in his component of $G - C$.

The other special case of $s = 1$ is directly equivalent to the cop and robber game introduced in Section 2.1 and therefore $\mathrm{cn}_1(G) = \mathrm{cn}(G)$.

Due to the following natural generalization of Lemma 8, we are mainly interested in connected graphs.

Lemma 39. *If G has connected components $C_1, \ldots, C_k$, then for any $s \geq 1$ we have $\mathrm{cn}_s(G) = \sum_{i=1}^{k} \mathrm{cn}_s(C_i)$.*

Proof. Analogously to the proof of Lemma 8, this follows from the fact that the cops may never move between the components after the initial placement. □

Our main result on this game is summarized in the following theorem.

Theorem 40. *There exists a polynomial-time algorithm that, given an n-vertex interval graph G, computes $\mathrm{cn}_\infty(G)$ and also a winning strategy for the cops that captures the robber in $\mathcal{O}(n^3)$ turns.*

The proof of the theorem may be found in Section 3.4 below.

3.2 History and related results

The cops and s-fast robber game is a generalization of the original Cops and Robber game introduced in Section 2.1, allowing the robber to make up to s steps instead of 1 in one turn.

The motivation for examining the complexity of the ∞-fast robber game does not, slightly surprisingly, come from other games with ∞-fast robber, but from a research on bounded speed games: Fomin et al. [FGK+10] examine several complexity aspects of cops and s-fast robber games and show that, for each $s \geq 2$,

the problem of computing $\mathrm{cn}_s(G)$ is NP-hard (and even W[2]-hard in the version parametrized by cn_s) even if G is a chordal graph, or even a split graph[1]. On the other hand, $\mathrm{cn}_s(G)$ can be computed in polynomial time for an interval graph and every fixed $s < \infty$. While the hardness results of Fomin et al. easily extend to the game of ∞-fast robber for the mentioned classes, the polynomiality proof for interval graphs does not, and our proof takes a different approach.

This game of cops and ∞-fast robber has been further studied by Mehrabian [Meh11a] who gave a 3-approximation polynomial-time algorithm for interval graphs, but the complexity status of the exact problem remained open. In [Meh12], a characterization of graphs with $\mathrm{cn}_\infty(G) = 1$ is given.

For some bounds on $\mathrm{cn}_s(G)$ see [AM11, FKL12, Meh11b]. For other works on the version of the game with different speeds see, e.g., [CCNV11, NS08]

3.3 $\mathcal{A}$-defensive domination

As a main tool in our analysis of cops and ∞-fast robber game, we introduce the problem of $\mathcal{A}$-*defensive domination* – a generalization of a domination problem of Farley and Proskurowski which in turn generalizes domination.

This section is completely independent from the game analysis and we hope it to be of independent interest.

3.3.1 The problem and motivation

Defensive domination has been introduced by Farley and Proskurowski [FP04] in the following way. A simple set $D \subseteq V_G$ is said to be k-*defensive dominating* if for each set $\{a_1, \ldots, a_k\} \subseteq V_G$ of k distinct vertices of G (called a k-*attack*) there exists a set $\{d_1, \ldots, d_k\} \subseteq D$ of k distinct vertices of D (called an *assignment of defenders*) such that for each $i \in \{1, \ldots, k\}$ we have $d_i \in N[a_i]$ (every attacker is assigned a different adjacent or same-vertex defender).

Problem 41 (k-DEFENSIVE DOMINATION)**.** *Given a graph G and an integer k, compute a smallest simple set D that is k-defensive dominating on G.*

The concept of k-defensive domination has been introduced as a generalization of the well-known dominating set problem. They see the dominating set as a "defense unit" placement scheme where every single unit may actively defend an attack on one vertex in distance at most 1. The k-defensive domination asks for defense unit placement that can counter any k-vertex attack. We take a slightly different approach and propose the problem with the possible attack sets as a part of the input.

Farley and Proskurowski [FP04] show that while the problem is generally NP-hard (as it generalizes the problem of smallest dominating set even for $k = 1$), there is a polynomial-time algorithm for any k on trees.

Here, we use an equivalent definition of defense and generalize it to multisets: A vertex (multi)set D (the defender placement) *defends* an attacker (multi)set

[1] A split graph G has $V_G = K \cup I$ with $G[K]$ a clique, $G[I]$ an independent set and arbitrary edges between K and I.

A when there is a map $f : A \to D$ which is injective[2] and when $f(a) = d$ then $d \in N[a]$. Such f is called a *defense* or a *defending mapping*.

We consider multisets to provide a natural but strong generalization of the problem as well as to address a situation with multiple cops on a vertex in our application to the game analysis.

The following generalization allows us to explicitly specify the attacks of interest, both the attacks and the defenses may be multiset, we may limit the vertex defender-capacities by D_{max} and we may specify fixed (pre-placed) defenders D_{min}.

Problem 42 ($\mathcal{A}$-DEFENSIVE DOMINATION)**.** *Given a graph G, a family of vertex multisets $\mathcal{A}$ (the attacks) and vertex multisets $D_{min} \subseteq D_{max}$, compute a smallest multiset of vertices D defending every attack $A \in \mathcal{A}$ such that $D_{min} \subseteq D \subseteq D_{max}$, or answer that no such D exists.*

The natural settings are empty $D_{min} = \emptyset$ and unbounded $D_{max} = t * V$ for some large-enough t. Also, setting $D_{max} = V$ (a simple set) would force the defender set D to be a simple set.

Our statement of the problem differs from the one in [FP04] in two ways. First, we may allow more defenders per vertex. This can be forbidden by setting $D_{max} = V$ as above. However, in this article we only show an algorithm for unbounded capacities D_{max}. Note that considering just unbounded capacities does not interfere with use in the game part of the paper and allows us use simpler the algorithm and technical arguments. See Section 3.5 for a discussion of extensions.

Second, and more importantly, in our case the collection $\mathcal{A}$ is a part of the input and may contain multisets of any size, as opposed containing all the subsets of the vertices of G of given size k. Specifying $\mathcal{A}$ brings more flexibility but having all the sets $\binom{V}{k}$ as part of the input may blow it up substantially even for medium values of k.

Our main result on $\mathcal{A}$-defensive domination is the following theorem.

Theorem 43. *There is an algorithm that solves* $\mathcal{A}$-DEFENSIVE DOMINATION *on interval graphs in time polynomial in the input size for any D_{min} and unbounded capacities D_{max}.*

3.3.2 Leftmost defense on interval graphs

Before we state the algorithm and the proof of the theorem, let us introduce further definitions specific to interval graphs and our algorithm. We assume that the input graph G is given with an interval representation, and we use notation introduced in Section 1.2.1.

With D a multiset of defenders let a *partial defense* (against an attack A) be a function from $A' \subseteq A$ to D mapping every vertex to distance at most 1, similarly to defense defined above. Two partial defenses f_1 and f_2 *agree on a set* $X \subseteq A$ if $f(x) = f'(x)$ for each $x \in X$.

[2]Injectivity of a multiset map means that is every element $d \in D$ is the image of at most c multiset elements where c is the multiplicity of d in D.

Before the algorithm, we introduce a property of a defense we use throughout the algorithm. For a given attack A and defenders D, order the attackers $a_1 <_R \ldots <_R a_k$. A (partial) defense f is called *leftmost (partial) defense* if the defender $f(a_i)$ assigned to attacker a_i is $<_R$-leftmost among the neighbors of a_i from $D \setminus \{f(a_1), \ldots f(a_{i-1})\}$. Such (partial) defense is called *maximal* if f assigns a defender to a_i whenever $N(a_i) \cap (D \setminus \{f(a_1), \ldots f(a_{i-1})\}) \neq \emptyset$.

Note that for given attack and defense, a unique maximal leftmost (partial) defense is computed by the following straightforward algorithm which closely follows the definition of a leftmost (partial) defense:

procedure LeftmostDefense(G, A, D)
  Input: Represented interval (multi)graph G
  Input: Multisets $A \subseteq V_G$, $D \subseteq V_G$
  Output: Maximal leftmost (partial) defense $f : A' \to D$, $A' \subseteq A$
  $f \leftarrow \emptyset$
  Order vertices of A as $a_1 \leq_R a_2 \leq_R \ldots \leq_R a_k$
  for all $i \in \{1, \ldots k\}$ **do**
    $D_i \leftarrow N(a_i) \cap (D \setminus f[A'])$
    if $D_i \neq \emptyset$ **then**
      $A' \leftarrow A' \cup \{a_i\}$
      $f(a_i) \leftarrow \min_{<_R} D_i$
    end if
  end for
  return f
end procedure

Lemma 44. *The algorithm* LeftmostDefense *returns a maximal leftmost (partial) defense.*

Proof. The procedure obviously returns a valid partial defense map, runs in polynomial time and is easily implementable in time $\mathcal{O}(k|V|)$. By induction on i, the mapping f is a leftmost partial defense and also maximal on the set $\{A_1 \ldots A_i\}$. □

Whenever there is no full defense for A and D, the $<_R$-leftmost undefended attacker is called the *leftmost greedily undefended attacker*. Note that f is injective (w.r.t. D) and so the partial function $f^{-1} : D \to A$ is well-defined and injective whenever f is a full map. We can also additionally assume that f^{-1} is monotone$_R$ on every group of assigned defenders on a single vertex.

This computed defense also has the following useful properties.

Lemma 45. *The partial function f returned by* LeftmostDefense *satisfies: Whenever $a_i <_R a_j$ and $f(a_j) \in N[a_i]$, then $f(a_i) \leq_R f(a_j)$.*

Proof. Assume $f(a_i) >_R f(a_j)$. But in that case, a_i would get assigned $f(a_j)$ as a left-most available defender, which is a contradiction. □

Lemma 46. *Whenever there is a defensive map f from A to D on an interval graph G,* LeftmostDefense *returns a valid full defense map.*

Proof. Let f_{ALG} be as returned by LeftmostDefense(G, A, D) and among all defense maps, take $f : A \to D$ such that f and f_{ALG} agree on the longest prefix

$\{a_1 \dots a_{i-1}\}$. Now either $f_{\text{ALG}} = f$ and we are done, or we have either $f_{\text{ALG}}(a_i)$ undefined or $f(a_i) \neq f_{\text{ALG}}(a_i)$.

Note that we have $f(a_i) \in D_i$ (with D_i as in the algorithm) since f and f_{ALG} agree on $\{a_1 \dots a_{i-1}\}$. This immediately rules out the possibility of undefined $f_{\text{ALG}}(a_i)$. Additionally, this shows that $f_{\text{ALG}}(a_i) \leq_R f(a_i)$, since $f_{\text{ALG}}(a_i)$ is minimal from D_i.

The last possibility for us to examine is $f_{\text{ALG}}(a_i) <_R f(a_i)$. In this case we show that there is a full defense map $f' : A \to D$ that agrees with f_{ALG} on a longer prefix, a contradiction with the choice of f. Let $d_i^{\text{ALG}} = f_{\text{ALG}}(a_i)$ and $d_i = f(a_i)$. Moreover let $a_j = f^{-1}(d_i^{\text{ALG}})$ if defined, in which case also note that necessarily $j > i$. See Figure 3.1 for an illustration.

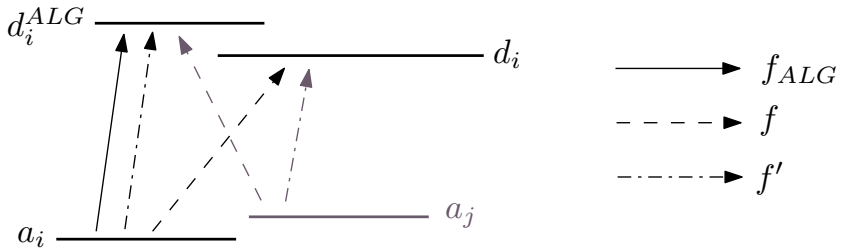


Figure 3.1: Illustration for the proof of Lemma 46

.

Set f' as f except for $f'(a_i) = d_i^{\text{ALG}}$ and $f(a_j) = d_i$ when a_j is defined. In case a_j is undefined, validity of f' is straightforward. In the other case we only need to show $a_j \in N[d_i]$: when $a_i <_R a_j \leq_R d_i$ observe that $a_i \in N[d_i]$, when $d_i^{\text{ALG}} <_R d_i \leq_R a_j$ observe that $a_j \in N[d_i^{\text{ALG}}]$. □

Additionally, when two defenses differ at some point, the algorithm finds defenses that agree up to that point:

Lemma 47. *Let $D = \{d_1 \leq_R \dots\}$ and $D' = \{d'_1 \leq_R \dots\}$ be defenses against an attack $A = \{a_1 \leq_R \dots\}$ and let f and f' be the defense maps computed by* LeftmostDefense. *Assume D and D' agree on $\{d_1 = d'_1, \dots, d_{i-1} = d'_{i-1}\}$, $d_i <_R d'_i$. Then f^{-1} and f'^{-1} agree on $\{d_1 \dots d_{i-1}\}$ (being either equal or both undefined).*

Proof. Assume that f^{-1} and f'^{-1} do not agree on a left-most d_j, $j < i$. Let $a_l = \min_R\{f^{-1}(d_j), f'^{-1}(d_j)\}$ (note that at least one is defined). Now if $f(a_l) = d_i$ then d_i was a left-most candidate defender for a in LeftmostDefense not only for f but also for f' since by assumption we have

$$f[\{a_1 \dots a_{l-1}\}] \cap \{d_1 \dots d_{j-1}\} = f'[\{a_1 \dots a_{l-1}\}] \cap \{d_1 \dots d_{j-1}\}$$

which is a contradiction with existence of such d_j. □

3.3.3 $\mathcal{A}$-defensive domination on interval graphs

Now we state and analyze the main algorithm, and show Theorem 43. The algorithm computing an optimal $\mathcal{A}$-defensive domination set on interval graphs follows:

procedure $\mathcal{A}$-DefensiveDomination(G, $\mathcal{A}$, D_{min})
  Input: Represented interval (multi)graph G
  Input: Multiset family $\mathcal{A} \subseteq 2^{V_G}$ (attack)
  Input: Multiset $D_{min} \subseteq V_G$
  Output: Smallest defender placement $D_{min} \subseteq D \subseteq V_G$
  $D \leftarrow D_{min}$
  loop
    for all $A \in \mathcal{A}$ **do**
      $f_A \leftarrow$LeftmostDefense(G, A, D)
    end for
    if all attackers defended in every f_A **then**
      return D
    end if
    $u \leftarrow$ left-most$_R$ vertex undefended in some f_A.
    $d' \leftarrow$ right-most$_R$ vertex of $N[u]$.
    $D \leftarrow D \cup \{d'\}$ ▷ A and u are referred to as *reason* for this d'.
  end loop
end procedure

Proof of Theorem 43. The algorithm always returns a solution, and any returned solution is a valid defense against all attacks, as certified by the computed maps f_A. Additionally, all the intervals of $D \setminus D_{min}$ are always inclusion-maximal. We only need to show size-optimality.

Let us call the computed defense $D_{\text{ALG}} = \{d_1^{\text{ALG}} \leq_R \dots\}$ and take an optimum-size defense $D_{\text{OPT}} = \{d_1^{\text{OPT}} \leq_R \dots\}$ such that it agrees with D_{ALG} on a longest possible prefix $\{d_1^{\text{ALG}} = d_1^{\text{OPT}}, \dots, d_{i-1}^{\text{ALG}} = d_{i-1}^{\text{OPT}}\}$. We additionally assume that all the intervals of $D_{\text{OPT}} \setminus D_{min}$ are inclusion-maximal (as are those of $D_{\text{ALG}} \setminus D_{min}$). Let $\{f_{\text{ALG},A}\}_{A\in\mathcal{A}}$ and $\{f_{\text{OPT},A}\}_{A\in\mathcal{A}}$ be the defense maps computed by LeftmostDefense for D_{ALG} and D_{OPT} respectively. Note that $f_{\text{OPT},A}$ are well-defined thanks to Lemma 46.

If $D_{\text{ALG}} = D_{\text{OPT}}$ we are done. Otherwise, let $d_i^{\text{ALG}} \neq d_i^{\text{OPT}}$ be the first difference in the solutions. Note that thanks to Lemma 47 the maps $f_{\text{ALG},A}^{-1}$ and $f_{\text{OPT},A}^{-1}$ agree on $\{d_1^{\text{ALG}}, \dots, d_{i-1}^{\text{ALG}}\}$. Consider the two possibilities:

1. $d_i^{\text{ALG}} <_R d_i^{\text{OPT}}$. Let A' and u' be the reason for including d_i^{ALG} in D_{ALG}. Now OPT can not defend u' in A': We have $d_i^{\text{OPT}} \notin N[u']$ since $d_i^{\text{OPT}} >_R d_i^{\text{ALG}}$ and d_i^{ALG} is the right-most neighbor of u'. At the same time, all the vertices of D_{OPT} before d_i^{OPT} are either already used in $f_{\text{OPT},A'}$ (as they are in in $f_{\text{ALG},A'}$) or too far (otherwise u' would not be added to D_{ALG} at that point).

2. $d_i^{\text{ALG}} >_R d_i^{\text{OPT}}$. Take as d_j^{ALG} the *left-endpoint* left-most$_L$ with $j \geq i$ and then set $D'_{\text{OPT}} = D_{\text{OPT}} - d_i^{\text{OPT}} + d_j^{\text{ALG}}$. We show that D'_{OPT} is a valid defense: take any attack $A \in \mathcal{A}$ and let $a = f_{\text{OPT},A}^{-1}(d_i^{\text{OPT}})$. If such a is undefined, let $f'_{\text{OPT},A} = f_{\text{OPT},A}$ and this map trivially defends A using D'_{OPT} as well.

If such a exists, let $d_l^{\text{ALG}} = f_{\text{ALG},A}(a)$ and note that $l \geq i$ thanks to Lemma 47. Now in both cases, $a <_R d_i^{\text{OPT}}$ and $a >_R d_i^{\text{OPT}}$, we straightforwardly obtain $d_j^{\text{ALG}} \in N[a]$. The map $f'_{\text{OPT},A}$ derived from $f_{\text{OPT},A}$ by replacing $f'_{\text{OPT},A}(a_j) = d_j^{\text{ALG}}$ is then a valid defense map. In either case we get a contradiction with the choice of D_{OPT}. □

3.4 Game reductions and strategy

We solve ∞-fast robber game on interval graphs by showing an equivalence with an auxiliary *restricted game* that we introduce below. As you shall find below, this new game has a small enough state space to be solved by a full state space search.

3.4.1 Observations and notation

We start with several observations of the game properties and introduce some notation.

To avoid special treatment of the game-states at the start of the game, we work with a game with modified starting state, which is equivalent to the unmodified game on connected graphs:

Lemma 48. *A connected interval graph G is k-cop-win if and only if k cops win starting all positioned at the leftmost vertex a of G (w.r. to $<_R$) with the robber starting on the rightmost vertex b (w.r. to $<_L$) such that* $\mathrm{dist}(a, b) \geq 2$. *If there is no such b, then the graph is* 1*-cop-win.*

Proof. To show this, assume that k cops have a wining strategy in the unmodified game starting at a configuration C. In the modified game, the cops can first use several moves to get to C ignoring the robber (or even capturing him on the way accidentally) and then play out the winning strategy according to the position of the robber at that point.

If the robber has a wining strategy in the unmodified game for any starting position of the cops, then let C be the cop's positions after their first move from a and let w be the desired robber's starting position for C. The cases $w = a$ or $w \in C$ can happen only if the radius of the graph is ≤ 1 and therefore G can not be robber-win. Also, we can assume $w \notin N[C]$ as such strategy would lose immediately. Otherwise w and b belong to the same component of $G \setminus C$, since every such component is either contained in $N[C]$ (and therefore not containing w) or contains b as the right-most vertex. In that case, the robber can move to w and play out his winning strategy. □

Formally, we denote the game state before cops' move (also *cop-state*) by $\mathcal{C}(C, w)$, where C is a multiset of vertices occupied by the cops and w is the vertex occupied by the robber. The game state before robber's move (also *robber-state*) is $\mathcal{R}(C, A)$ with C as above and A is the set of all vertices the robber may reach in his following move (thus, A is the vertex set of the connected component of $G - C$ containing the robber).

Note that before any robber's move, two states with the robber in the same component of $G - C$ offer the same moves to the robber and this notation already slightly reduces the complexity of the examined states. Also note that A is by definition always connected. By the remarks in Section 1.2.1 we have the following.

Observation 49. *For any game state $\mathcal{R}(C, A)$, $\varphi[A]$ is an interval.*

For a state $\mathcal{R}(C, A)$ we call the interval $\varphi[A]$ the *playground* of the state. The playground (l, r) is an open interval between the cutpoints l and r called specifically the *left and right barrier*.

Note that all vertices containing the barriers, i.e., $\varphi^{-1}(l) \cup \varphi^{-1}(r)$, are occupied by the cops (otherwise the playground would be bigger). The vertices of $\varphi^{-1}(l)$ and in $\varphi^{-1}(r)$ are called the *barriers' support*. Note that the support of either barrier may be empty and in such case, due to the connectedness of G, we may assume that such a barrier is at either 1 or $2|V|$. The vertices in $\widetilde{\varphi}(l, r)$ are called the *playground support*. Note that, by definition, the support of a playground is always disjoint from the supports of the barriers.

Among the cops occupying a barrier's support, we choose and fix one cop per vertex. Let us call these cops *the cops holding the barrier*. Note that a cop may hold both barriers at once, but as we see below, that this may happen only just before capturing the robber.

A playground (l, r) is *feasible*, if $|\varphi^{-1}(l) \cup \varphi^{-1}(r)| \leq k$ where k is the number of cops. That is if the cops are able to hold both barriers at once. A playground (l, r) is *non-trivial* if $\widetilde{\varphi}(l, r)$ is nonempty and it does not contain all vertices of G.

For every feasible and non-trivial playground (l, r), we fix a *canonical* game state

$$\text{canon}(l, r) = \mathcal{R}(\varphi^{-1}(l) \cup \varphi^{-1}(r), \widetilde{\varphi}(l, r))$$

in which the cops occupy all vertices in $\varphi^{-1}(l) \cup \varphi^{-1}(r)$ and the extra cops, if any, are positioned on an arbitrary (say, left-most$_R$) vertex in $\varphi^{-1}(l) \cup \varphi^{-1}(r)$. A game state won by the cops, canonical for every playground with $\widetilde{\varphi}(l, r) = \emptyset$, is denoted by $\mathcal{WIN}$.

Lemma 50. *If $\widetilde{\varphi}(l, r) \neq \emptyset$ then the playground of canon(l, r) is (l, r).*

The cops occupying the vertices in C *threaten* a vertex set T if the cops can occupy all vertices of T after one cop-move. This is equivalent to an existence of a partial surjective mapping from the cops on C to T in such that every cop is assigned to a vertex in distance at most 1. If the cops threaten $\varphi^{-1}(i)$ for some $i \in \{1, \ldots, 2|V|\}$ then we also say that the cops *threaten* the cut i. When considering a set of cops threatening T, we fix a matching between the threatening cops and the vertices in T for the moment. In the rest of this section we introduce some additional notation that allows us to formally define the sets T we are be interested in.

We say that a cop/robber *is over* an interval I if it is located on a vertex v such that $I \subseteq \varphi(v)$. Then the cops $c_1, \ldots, c_p$ *are over* I if c_i is at vertex v_i, $i = 1, \ldots, p$, and $I \subseteq \varphi(v_1) \cup \cdots \cup \varphi(v_p)$. Any maximal interval B such that some cops are over B is called a *base*. Given a base B, $\xi(B) = \bigcup_{x \cap B \neq \emptyset} \varphi(x)$ is called the *cover* of B.

Alternatively, when the cops are positioned on $C \subseteq V$, the bases are the maximal intervals of $\varphi[C]$. The cover of a single base are the individual vertices threatened by the cops of that base.

Note that in game state $\mathcal{R}(C, A)$, if the robber positions himself on a vertex v such that $\varphi(v)$ intersects a base then the cops can catch the robber in the next move. Let $A' \subseteq A$ be the vertices safe for the robber, that is $A' = A \setminus N[C]$. This includes vertices v such that $\varphi(v)$ is entirely contained in $\xi(B) \setminus B$. Any maximal

interval in $\varphi[A']$ is called a *hole in* $\mathcal{R}(C, A)$, or simply a *hole*, if the state is clear from the context. Hence, if $\mathcal{C}(C, r)$ is the state that follows $\mathcal{R}(C, A)$, then either the cops can immediately catch the robber or $\varphi(r)$ is contained in some hole in $\mathcal{R}(C, A)$.

Given a state $\mathcal{R}(C, A)$, a *base collection* $\mathcal{B}$ is a set of bases $\{B_1, \ldots, B_l\}$ such that $\bigcup_{i=1}^{l} \xi(B_i)$ is an interval. Denote the latter interval by $\xi(\mathcal{B})$. A base collection $\mathcal{B}$ is *maximal* if $\mathcal{B} \cup \{B\}$ is not a base collection for any base B, $B \notin \mathcal{B}$.

3.4.2 Manoeuvres and the restricted game

The main tool of our result is to transform an arbitrary cops' winning strategy to a *restricted* strategy in an equivalent but simpler *restricted game* on a smaller state space. Informally, a restricted game state only describes which cuts are held or threatened by the cops and the current playground of the robber or his choices of a new playground.

While a general cops' strategy is mapping from *every* valid state of the game to a move valid in that state, a *restricted cop's strategy* is a mapping from a *restricted game states* $\mathcal{C}(C, r)$ to the maneuvers valid in those states. In the following we fix a constant $Q = 12$, that is the sufficient number of following playgrounds to be considered, as we show later. A *restricted game state* is $\mathcal{WIN}$ (game won by the cops) or one of the following:

Restricted game cop-states denoted $\widetilde{\mathcal{C}}(l, r)$ represent the state $\text{canon}(l, r) = \mathcal{R}(\varphi^{-1}(l) \cup \varphi^{-1}(r), \widetilde{\varphi}(l, r))$, where the cops choose the next maneuver to perform. Note that the corresponding general game state is a robber-state, which is more convenient as the maneuver ignores the position of the robber possibly except for the last turn before capture (see below).

Restricted game robber-states denoted $\widetilde{\mathcal{R}}((l_1, r_1), \ldots (l_q, r_q))$, $q \leq Q$, where the robber chooses the next playground he will be restricted to after the next cop-move. The restricted robber state represents any of the states $\mathcal{R}(C, \widetilde{\varphi}(l, r) \setminus C)$ for some C and l, r such that C is a witness for a split maneuever from $\widetilde{\mathcal{C}}(l, r)$ to $\widetilde{\mathcal{R}}((l_1, r_1), \ldots (l_q, r_q))$, described below. We may assume that this representative is the witness found by the algorithm in Lemma 51.

This state happens in the original game only before the last move in a spit-maneuver (defined below), where the cops threaten multiple cuts and the robber has to decide where to stand before the cops actually choose which two barriers to create and therefore what will be the new playground. Note that the cops actually do not only present robber with the threatened cuts, but additionally inform the robber about the at most Q possibilities of their next turn. The conditions on the maneuvers ensure that the list of options is complete and valid.

Formally, a *maneuver* is a fixed finite sequence of cop-moves not depending on the robber's movement in the meantime, except for automatically capturing the robber if he ends his move adjacent to a cop. A maneuver always starts in a restricted cop-state and ends in a robber-state (where robber chooses the next playground) or $\mathcal{WIN}$. A maneuver is *valid* if k cops are capable to perform the

moves of the maneuver while restricting the robber accordingly, as specified below. There are two types of maneuvers:

Endgame from $\widetilde{\mathcal{C}}(l,r)$ to $\mathcal{WIN}$. Starting from the state canon(l,r), the cops move so that in each turn hold l and r, into position $\mathcal{R}(C,A)$ such that C contains $\varphi^{-1}(l) \cup \varphi^{-1}(r)$ and dominates $\widetilde{\varphi}(l,r)$. In their next move the cops capture the robber. Such a multiset C with $|C| \leq k$ is a *witness* of the maneuver. The maneuver itself is, in this case, the sequence of the above moves that lead from canon(l,r) to $\mathcal{WIN}$.

Split from $\widetilde{\mathcal{C}}(l,r)$ to $\widetilde{\mathcal{R}}((l_1,r_1),\dots(l_q,r_q))$, $q \leq Q$, with $l_i \leq l_{i+1}$ for each $i \in \{1,\dots,q-1\}$.

Starting from the state canon(l,r), the move (while holding l and r) into position $\mathcal{R}(C,A)$ such that

(i) the cops are holding l and r,

(ii) there exists a base collection $\mathcal{B} = \{B_1,\dots,B_{q-1}\}$ such that $r_i, l_{i+1} \in \xi(B_i)$ for each $i = 1,\dots,q-1$,

(iii) the cops are threatening $\varphi^{-1}(l_i) \cup \varphi^{-1}(r_i)$ for each $i \in \{1,\dots,q\}$,

(iv) C dominates $\widetilde{\varphi}(r_i, l_{i+1})$ for each $i \in \{1,\dots,q-1\}$,

(v) C dominates $\widetilde{\varphi}(l,l_1)$ and $\widetilde{\varphi}(r_q,r)$.

Such a multiset C with $|C| \leq k$ is called the *witness* of the maneuver. Note that the cops holding l and r may be used in the threatening mapping. Also note that the condition on dominating $\widetilde{\varphi}(r_i,l_{i+1})$ is trivially satisfied if $r_i > l_{i+1}$.

In the state $\widetilde{\mathcal{R}}((l_1,r_1),\dots(l_q,r_q))$, the robber selects his next position w. Either $w \in N[C]$ and the cops win immediately, or $w \in \widetilde{\varphi}(l_i,r_i)$ for some $i \in \{1,\dots,q\}$. Then, the cops make a move that results in canon(l_i,r_i). Note that in the restricted game the robber is given the choice of i even in the case that for his choice of w both $w \in \widetilde{\varphi}(l_i,r_i)$ and $w \in \widetilde{\varphi}(l_j,r_j)$ and the cops would therefore make the choice of the next playground (out of the two) in the real game, but the choice is up to the robber in the restricted game.

It is not trivial to check for the existence of such witnesses, but we can find a smallest witness using the algorithm for $\mathcal{A}$-defensive domination presented in Section 3.3.

Lemma 51. *There are polynomial-time algorithms deciding the validity of maneuvers* endgame *and* split*.*

Proof. We observe that by the definitions, a multiset D is a smallest witness of a split maneuver from canon(l,r) to one of canon(l_i,r_i), $i \in \{1,\dots p\}$ if and only if D is a smallest $\mathcal{A}$-defensive multiset for $\mathcal{A} = \{\varphi^{-1}(l_i) \cup \varphi^{-1}(r_i) \;\big|\; i \in \{1,\dots p\}\} \cup \binom{Y}{1}$ (possible barriers to be taken) with $D_{min} = \varphi^{-1}(l) \cup \varphi^{-1}(r) \subseteq D$ (pre-placed defenders/cops) and unbounded capacities D_{max}, where $Y = \widetilde{\varphi}(l,l_1) \cup \widetilde{\varphi}(r_p,r) \cup \bigcup_{i=1}^{p-1} \widetilde{\varphi}(r_i,l_{i+1})$ (the vertices between the playgrounds to be dominated individually, preventing robber from safely moving there).

Indeed, suppose first that D is a witness of size k of a split maneuver. Then D is a solution to $\mathcal{A}$-defensive domination: For each $A \in \mathcal{A}$ generated by a playground, the injective mapping $f_X : X \to D$ is given by the possible cop-move from D to occupy A. For $A \in \mathcal{A}$ arising from Y, the mapping (of a single defender) follows from D (simply) dominating Y. On the other hand, it is easy to check that a solution to $\mathcal{A}$-defensive domination satisfies all the witness conditions.

Similarly, a multiset D is a smallest witness of an endgame maneuver from $\text{canon}(l, r)$ if and only if D is a smallest $\binom{\widetilde{\varphi}(l,r)}{1}$-defensive multiset with $D_{min} = \varphi^{-1}(l) \cup \varphi^{-1}(r) \subseteq D$ and unbounded capacities D_{max}.

The algorithm deciding the problem of $\mathcal{A}$-defensive domination is polynomial for interval graphs according to Theorem 43. The input size is polynomial in the size of G and number of playgrounds considered, which is bounded by Q. □

This then allows us to decide the existence of a cops' restricted strategy.

Theorem 52. *There exists a $\mathcal{O}(n^{\mathcal{O}(1)})$-time algorithm that, given an interval graph G with $|G| = n$ and an integer k, decides the existence of a winning restricted strategy using k cops for G.*

Proof. We construct a game-state digraph D representing the restricted game. V_D consists of all restricted game states including $\mathcal{WIN}$, the initial state is the state corresponding to the initial position of the modified game. The only cop-win state is $\mathcal{WIN}$, there are no robber-win states.

For every valid *endgame* maneuver from $\text{canon}(l, r)$ add an arc from $\text{canon}(l, r)$ to $\mathcal{WIN}$. For every valid *split* from $\text{canon}(l, r)$ to one of the $\text{canon}(l_i, r_i)$, $i \in \{1, \ldots, q \leq Q\}$, add an arc from $\text{canon}(l, r)$ to $\mathcal{R}\{(l_1, r_1), \ldots (l_q, r_q)\}$. From every $\mathcal{R}\{(l_1, r_1), \ldots (l_q, r_q)\}$ add arcs to $\text{canon}(l_i, r_i)$, $i \in \{1, \ldots, q\}$.

We decide the game given by D using Theorem 4, giving us either a winning restricted strategy for the cops or a non-losing restricted strategy for the robber.

Since Q is a fixed constant, D has polynomial size, every arc can be decided in polynomial time according to Lemma 51, the game decision algorithm also runs in time polynomial in $|D|$. □

3.4.3 Equivalence of the restricted game

In this section we show that the restricted game is equivalent to the original game in the following sense.

Theorem 53. *For an interval graph G and an integer k, k cops have a winning strategy for the Cops and ∞-fast Robber game if and only if k cops have a winning strategy in the restricted game on G with any interval representation φ.*

Before we prove Theorem 53, we show several lemmas which compose the individual steps of the equivalence and, unfortunately, we need to introduce additional general assumptions and definitions.

A strategy for the Cops and ∞-fast Robber game is called *simple* if at the beginning of any turn in which the robber occupies a vertex p and in which the playground changes, the following holds: for each base B there exist at most two cuts l and r, $l, r \in \xi(B)$, such that if $\text{R}(p) < \text{L}(B)$, then the cops do not take any

cuts in B except possibly for r, and if $\mathrm{L}(p) > \mathrm{R}(B)$, then the cops do not take any cuts in B except possibly for l. We call l and r, respectively, to be the *left* and *right barriers associated with* B. Informally speaking, in a simple strategy if the robber positions himself to the left (right, respectively) of B in a turn in which the playground changes, then the cops either start to hold r (l, respectively) or do not hold any barrier of B.

Lemma 54. *For an interval graph G and an integer k, if k cops have a winning strategy for the Cops and ∞-fast Robber game, then k cops have a simple winning strategy.*

Proof. Let us consider a playground changing move in a winning strategy $\mathcal{S}$ and the two game states $\mathcal{R}(C, A)$ and $\mathcal{C}(C, w)$ preceding the playground change. Let $H_1, \ldots, H_q$ be all holes at the beginning of $\mathcal{R}(C, A)$ ordered so that $\mathrm{R}(H_j) < \mathrm{L}(H_{j+1})$ for each $j \in \{1, \ldots, q-1\}$. Take any base B formed by the cops at the beginning of $\mathcal{R}(C, A)$. If $\mathrm{R}(H_1) \leq \mathrm{L}(B)$, then let $p \in \{1, \ldots, q\}$ be the highest index such that $\mathrm{R}(H_p) \leq \mathrm{L}(B)$, and let $p = 0$ otherwise.

Since $\mathcal{S}$ is winning, for each $j \in \{1, \ldots, p\}$ there exists a barrier $r_j \in \xi(B)$ such that if $w \in \varphi^{-1}(H_j)$, then the cops take in $\mathcal{C}(C, w)$ either no barrier in $\xi(B)$ or they take r_j. Suppose that $r_i \neq r_{i'}$ for some $i, i' \in \{1, \ldots, p\}$. We argue that we can modify the strategy $\mathcal{S}$ so that $r_i = r_{i'}$ and the 'worst-case' length of the new strategy is not greater than the length of $\mathcal{S}$. If r_i is not taken by the cops when $w \in \varphi^{-1}(H_i)$, then one may change $r_i := r_{i'}$. By using a similar argument for $r_{i'}$, we obtain that both r_i and $r_{i'}$ become the endpoints of the playgrounds in the cases when $w \in \varphi^{-1}(H_i)$ and $w \in \varphi^{-1}(H_{i'})$, respectively. Suppose without loss of generality that $i < i'$. Then, however, changing the playground in case of $w \in \varphi^{-1}(H_{i'})$ to be the same as in case of $w \in \varphi^{-1}(H_i)$ results in particular in $r_i = r_{i'}$.

We repeat the same argument for B and $l_{p+1}, \ldots, l_q$, where l_j is the border taken by the cops when $w \in \varphi^{-1}(H_j)$ for each $j = p+1, \ldots, q$. As a result, we obtain a strategy with the same bases as in $\mathcal{S}$, of length not greater than that of $\mathcal{S}$. By repeating the same transformation for each remaining base we obtain the desired simple strategy. □

Note that in a general strategy, in a move that changes the playground many base collections can be formed and each of them can contain many bases (up to one collection per a cop). We prove below that it is enough to consider one base collection at a time. This motivates the following definition:

We say that a strategy for the Cops and ∞-fast Robber game is *semi-restricted* if it is simple and for each cops' move that results in a change of the playground in the preceding robber's move there exists exactly one base collection.

Lemma 55. *Let $\mathcal{S}$ be a winning strategy using k cops. Given any robber state $\mathcal{R}(C, A)$ of $\mathcal{S}$, let $\mathcal{R}(C_i, A_i)$, $i \in \{1, \ldots, q\}$, be the robber states of $\mathcal{S}$ reachable from $\mathcal{R}(C, A)$ in two turns (a robber turn and a cop turn). Let $P = (l, r)$ be the playground corresponding to $\mathcal{R}(C, A)$ and $P_i = (l_i, r_i,)$ be the playground corresponding to $\mathcal{R}(C_i, A_i)$ for each $i \in \{1, \ldots, q\}$. Then, there is a semi-restricted cops' strategy that uses k cops, starts in state canon(l, r) and either wins or results in state canon(l_i, r_i) for some $i \in \{1, \ldots, q\}$.*

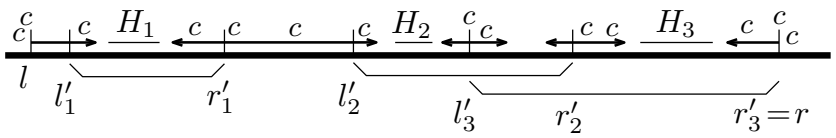


Figure 3.2: An illustration of the playgrounds of $\mathcal{P}$ with $q = 3$. Note that the playgrounds may overlap. The intervals H_j are the holes separating the groups of cops of C.

Proof. In view of Lemma 54, we may assume without loss of generality that $\mathcal{S}$ is simple. If C is a witness for *endgame*, then the considered turn does not violate the definition of semi-restricted strategy. Hence, assume that this is not the case and let $H_1, \ldots, H_q$ be the holes at the beginning of $\mathcal{R}(C, A)$. See Figure 3.2 for an illustration.

We prove the lemma by induction on the number of base collections of a strategy. If the state $\mathcal{R}(C, A)$ consists of one base collection, then the two following moves of $\mathcal{S}$ themselves form a is semi-restricted strategy. Thus, suppose that $\mathcal{S}$ has more than one base collection.

Let $P'_j = (l'_j, r'_j)$ be the playground at the end of $\mathcal{C}(C, w)$ when $w \in \varphi^{-1}(H_j)$ for each $j \in \{1, \ldots, q\}$. Let the corresponding states be $\mathcal{R}(C'_j, B'_j)$, $j \in \{1, \ldots, q\}$. Denote by $P'_{i_1}, \ldots, P'_{i_p}$ all inclusion-maximal playgrounds 'between' the base collections, that is, $P'_{i_j} \nsubseteq \mathcal{B}$ for any base collection $\mathcal{B}$ formed at the beginning of $\mathcal{R}(C, A)$, $j \in \{1, \ldots, p\}$. By assumption, $p > 1$.

The strategy construction is algorithmic and is done by modifying $\mathcal{S}$. First, the modified strategy forms the base collection $\mathcal{B}$ with minimum $\mathrm{L}(\varphi[\mathcal{B}])$. Suppose that the robber occupies a vertex w such that $\mathrm{R}(\varphi(w)) < \mathrm{R}(\varphi[\mathcal{B}])$. Hence, $\varphi(w) \subseteq (l'_j, r'_j)$ for some $j < i_2$. The modified strategy plays the split maneuver to $\widetilde{\mathcal{C}}(l'_j, r'_j)$, as required.

Hence, assume that $\mathrm{R}(\varphi(w)) \geq \mathrm{R}(\varphi[\mathcal{B}])$. The strategy plays a split to $\widetilde{\mathcal{C}}(l'_{i_2}, r)$. Denote by $\mathcal{S}'$ the strategy that performs this maneuver. There exists a (general) strategy that uses $p - 1$ base collections (all base collections, except for $\mathcal{B}$, of the initial strategy $\mathcal{S}$) and in one turn results in a playground $(l'_{j'}, r'_{j'})$ for some $j' \in \{i_2, \ldots, q\}$. By the induction hypothesis, there exists a semi restricted strategy $\mathcal{S}''$ that either wins or results in a state $\mathrm{canon}(l'_{j'}, r'_{j'})$ for some $j' \in \{i_2, \ldots, q\}$. Thus, $\mathcal{S}'$ together with $\mathcal{S}''$ is the strategy that satisfies the conditions of the lemma. Indeed, the number of cops that the latter strategy uses is k because, by definition, no cop is simultaneously used in two base collections. □

The following lemma is used to show that it is sufficient to consider a bounded number of playgrounds resulting from a split maneuver.

Lemma 56. *Let $\mathcal{B} = \{B_1, \ldots, B_q\}$, $q \geq 12$, be a base collection that is formed by k' cops. Let l_i and r_i be the left and right barriers associated with B_i, $i \in \{1, \ldots, q\}$. If $1 \leq j < j' \leq q$ and $j' - j \leq 3$, then endgame maneuver using k' cops is possible from $\mathrm{canon}(l_j, r_{j'})$.*

Proof. We prove the lemma by placing the cops so that they hold l_j and $r_{j'}$ and dominate $\widetilde{\varphi}(l_j, r_{j'})$. The cops in $\mathcal{B}$ used to threaten l_j and $r_{j'}$ are selected occupy the vertices in $\varphi^{-1}(l_j) \cup \varphi^{-1}(r_{j'})$. If $j' = j + 1$, then two additional cops can dominate $\widetilde{\varphi}(l_j, r_{j'})$, which follows directly from the definition of base collection.

The two cops are available, because $q \geq 9$. Hence, assume that $j' > j+1$ and we describe how to dominate $\widetilde{\varphi}(l_j, r_{j'})$ with the remaining cops.

Place the cops initially used to form $B_{j+1}, \ldots, B_{j'-1}$, except for those in Q, in the same way as in the base collection $\mathcal{B}$. The possibly non-dominated vertices are then the ones in $\widetilde{\varphi}(l_{j+1}, r_{j+2})$ if $j' - j = 3$, and $\widetilde{\varphi}(l_j, r_{j+1}) \cup X \cup \widetilde{\varphi}(l_{j'-1}, r_{j'}) \cup Y$, where X (respectively, Y) is the set of vertices initially dominated by the cops in $\mathcal{B}$ present on B_{j+1} (respectively, $B_{j'-1}$). Let $X' \subseteq X$ and $Y' \subseteq Y$ be, respectively, the vertices in X and Y occupied by the cops in Q. The vertex v in X' such that $\mathrm{R}(v) \geq \mathrm{R}(x)$ for each $x \in X'$ dominates X. Another cop can dominate $\widetilde{\varphi}(l_j, r_{j+1})$ by definition of base collection. Similarly, two cops can dominate $\widetilde{\varphi}(l_{j'-1}, r_{j'}) \cup Y$.

If $j' - j < 3$, then the construction is completed. Otherwise, two additional cops can be used to dominate $\widetilde{\varphi}(l_{j+1}, r_{j+2})$. Hence, in the worst case 6 additional cops are used with respect to the ones that initially form $B_j, \ldots, B_{j'}$. No cop from a base B_i, $i < j-1$ (respectively, $i < j'+1$), is used to hold l_j ($r_{j'}$, respectively). Since $q \geq 12$, the total number of cops used to construct the witness of endgame does not exceed k. □

Lemma 57. *Let $\mathcal{S}$ be a winning strategy using k cops. For a robber state $\mathcal{R}(C, A)$ in $\mathcal{S}$ let $\mathcal{R}(C_i, A_i)$, $i \in \{1, \ldots, q\}$, be the robber states of $\mathcal{S}$ reachable from $\mathcal{R}(C, A)$ in two moves (a robber's move and a cops' move). Let $P = (l, r)$ be a playground corresponding to $\mathcal{R}(C, A)$ and $P_i = (l_i, r_i)$ be the playgrounds corresponding to $\mathcal{R}(C_i, A_i)$, $i \in \{1, \ldots, q\}$. Then, there is a restricted cops' strategy that uses k cops, starts in the state $\widetilde{\mathcal{C}}(l, r)$ and either wins or results in one of states $\widetilde{\mathcal{C}}(l_i, r_i)$.*

Proof. By Lemma 55, there exists a semi-restricted search strategy $\mathcal{S}'$ that uses k cops and results in one of the $\mathrm{canon}(l_1, r_1), \ldots, \mathrm{canon}(l_q, r_q)$. The strategy $\mathcal{S}'$ performs several manoeuvres and in the following we analyse two selected moves of $\mathcal{S}'$ after which the playground changes. Namely, let $\mathcal{R}(C', A')$ be a state of $\mathcal{S}'$ such that at the end of the following cops' move the playground changes to one of $(l'_1, r'_1), \ldots, (l'_q, r'_q)$. Let (l', r') be the playground in $\mathcal{R}(C', A')$.

We argue that there exists a restricted strategy $\mathcal{S}''$ that starts from $\mathrm{canon}(l', r')$ and arrives at one of the $\mathrm{canon}(l'_1, r'_1), \ldots, \mathrm{canon}(l'_q, r'_q)$. Let $\mathcal{B} = \{B_1, \ldots, B_p\}$ be the base collection in $\mathcal{R}(C', A')$. If $p < 12$, then the two moves of $\mathcal{S}$ following $\mathcal{R}(C', A')$ form a restricted strategy as required. Hence, let $p \geq 12$ in the following.

In the remainder of this proof we assume that $l' \notin \xi(B_1)$ and $r' \notin \xi(B_p)$ as the other cases are analogous with minor adjustments to border conditions.

This assumption implies that $l'_1 = l'$, $r'_q = r'$ and $p = q - 1$. The strategy $\mathcal{S}''$ performs the first manoeuvre by holding l' and r' and threatening $\varphi^{-1}(l'_1) \cup \varphi^{-1}(r'_1)$ (by using the base B_1). If the robber responds by occupying a vertex in $\widetilde{\varphi}(l', r'_1)$, then the strategy plays to $\widetilde{\mathcal{C}}(l', r'_1) = \widetilde{\mathcal{C}}(l'_1, r'_1)$, and the lemma follows. Otherwise, the restricted strategy plays to $\widetilde{\mathcal{C}}(l'_1, r')$. Then, the strategy continues by holding l'_1 and r' and threatening $\varphi^{-1}(l'_q) \cup \varphi^{-1}(r'_q)$ (by using B_p). If the robber responds by occupying a vertex in $\widetilde{\varphi}(l'_q, r')$, then the strategy plays to $\widetilde{\mathcal{C}}(l'_q, r') = \widetilde{\mathcal{C}}(l'_q, r_q)$, and the lemma follows.

Otherwise, the strategy plays a split to (l'_1, r'_q) and performs iteratively the following maneuver. Let initially $i = 1$. The strategy holds l'_i and r'_q and threatens $\varphi^{-1}(l'_{i+2}) \cup \varphi^{-1}(r'_{i+2})$ by forming the bases B_{i+2} and B_{i+3}. Note that this is possible because no cop that currently holds the barrier l'_i is used in those bases.

If the robber decides to occupy a vertex in $\widetilde{\varphi}(l'_i, r'_{i+2})$, then the strategy plays to $\widetilde{\mathcal{C}}(l'_i, r'_{i+2})$ and, by Lemma 56, an endgame maneuver is valid from this state.

Otherwise, i.e., if the robber occupies a vertex in $\widetilde{\varphi}(l'_{i+2}, r'_q)$, the strategy plays to $\widetilde{\mathcal{C}}(l'_{i+2}, r'_q)$. If $i + 2 \geq q - 3$, then, by Lemma 56, an endgame is possible from $\widetilde{\mathcal{C}}(l'_{i+2}, r'_q)$. If $i + 2 < q - 3$, then we set $i := i + 2$ and repeat the above step. Thus, after a finite number of split maneuver, an endgame maneuver occurs as required. □

We compose those steps into proof of Theorem 53.

Proof of Theorem 53. The "if" part is straightforward, as the cops can play out the maneuvers of a restricted winning strategy. The maneuver properties and witnesses ensure that the moves are possible and that after the maneuver, the robber is inside the respective playground or captured. Note that the capture may occur even playing out a split maneuver.

For the other direction, let $\mathcal{S}$ be a shortest (i.e., with the minimum number of turns) cop's winning strategy using k cops. Note that if the cops play according to $\mathcal{S}$, the game never revisits a game state.

Let S be the subgraph of the game state digraph representing $\mathcal{S}$, in which the vertices are all the cop- and robber-states of the game. From each cop-vertex, there is exactly one cop-move in S, as dictated by $\mathcal{S}$. From each robber-vertex, all the robber-moves are present in S. Note that we can assume S to be acyclic, since it is a shortest winning strategy. Also, fix any total ordering o of the states of S extending the partial order given by the arcs (moves).

For any robber-state in $\mathcal{S}$ find the maximum (with respect to o) robber-state $\mathcal{R}(C, B)$ in S with the same playground. We construct a winning restricted strategy $\mathcal{T}$ using k cops as follows. Let $\mathcal{R}(C_i, A_i)$ and (l_i, r_i), $i \in \{1, \ldots, t\}$, be the states in S and the corresponding playgrounds reachable from $\mathcal{R}(C, A)$ in two moves. By Lemma 57, there exists a restricted strategy that uses k cops, starts in state $\widetilde{\mathcal{R}}(l, r) = \mathcal{R}(C, A)$ and ends in $\widetilde{\mathcal{R}}(l_i, r_i) = \mathcal{R}(C_i, A_i)$ for some $i \in \{1, \ldots, t\}$. Note that all $\widetilde{\mathcal{R}}(l_i, r_i)$ are different from $\widetilde{\mathcal{R}}(l, r)$, because the state $\mathcal{R}(C, A)$ is latest such state in $\mathcal{S}$.

Now we have that the latest occurrence of a robber-state with playground (l_i, r_i) is (with respect to o) later than that of (l, r). Therefore, by playing $\mathcal{T}$, the latest state (with respect to o) with the same playground as the current one only increases. This proves that $\mathcal{T}$ is acyclic and therefore finite (as $\mathcal{S}$ is), and winning for the cops, as there is no draw or robber-win position in the game. □

And with this, we may finally prove Theorem 40.

Proof of Theorem 40. According to Theorem 52, we can decide the existence of a winning restricted cops' strategy in polynomial time. By Theorem 53, such a strategy exists if and only if a general winning cops' strategy exists.

Note that any play of a restricted cops' winning strategy visits every restricted cop-state (playground) at most once. There are $\mathcal{O}(n^2)$ playgrounds and every robber-state is followed by a maneuver to a cop-state. It is easy to see that playing out any single maneuver takes $\mathcal{O}(n)$ cop-moves when the moving cops follow shortest paths to their destinations. Therefore the game takes $\mathcal{O}(n^3)$ turns. □

3.5 Remarks and open problems

Since the game is already NP-hard for general chordal graphs and even split graphs, it might be interesting to consider the complexity of the game on chordal graphs with bounded asteroidal number (or, asymptotically equivalently, the number of leaves of the underlying tree for the standard intersection representation of chordal graphs) and the class of circular-arc graphs. It seems that the notion of playgrounds of the reduced game can be extended to such graphs and they might have some common properties, but the analysis does not extend in a straightforward way.

The definition of $\mathcal{A}$-defensive domination generalizes k-defensive domination, but explicitly specifying $\mathcal{A} = \binom{V}{k}$ is not practical for even small values of k, making the complexity of the problems incomparable. One hope would be to find an algorithm for a left-most undefended attacker in some of the k-attacks for a given defense, and plug it into the current algorithm.

In this paper we only show an algorithm for $\mathcal{A}$-defensive domination assuming an unbounded number of defenders allowed on individual vertices. We strongly believe that specifying the number of allowed defenders for every vertex (in D_{max}) still allows for a polynomial algorithm. Together with the previous question, this would give a truly general answer for interval graphs.

It would of course be of interest to see the complexity of $\mathcal{A}$-defensive domination algorithms for other classes, both with specifying D_{min} and D_{max} as part of the input and fixed for the problems. Trees and, more generally, bounded tree-width and path-width graphs seem like natural candidates.

Chapter 4

Ultramonotone games and minors on hypergraphs

In the following chapter we look at *tree-depth* as a graph parameter together with its game characterisation, and we introduce and examine extensions of these concepts to hypergraphs and hypergraph pairs. We also recall the famous graph minors together with their relation to tree-depth, tree-width and ultramonotone games.

In Section 4.1 we show some of the related results and motivation for our research. Section 4.2 recalls tree-depth, its many characterisations and hypergraph minors. Section 4.3 introduces the notion of hypertree-depth, its relation to tree-depth, defines the ultramonotone marshals and robber game and hypergraph minors. Section 4.4 generalises all these concepts to hypergraph pairs, also introducing those. Also, some of the results of Section 4.3 are shown here in greater generality. Section 4.7 concludes with several remarks and open questions.

4.1 History and related research

The tree-depth of a graph is a parameter introduced by Nešetřil and Ossona de Mendez in 2006 [NdM06]. It has received much attention due to its connections to graph coloring and homomorphism dualities, and it plays a central role in the theory of graph classes of bounded expansion [NdM08a, NdM08b, NdM08c].

The main motivation of this research direction is to extend the notion of tree-depth to the hypergraph setting while preserving as many nice and useful properties as possible, with a big part of the intuition coming from monotonicity of the defining games. The same motivation applies for hypergraph minors also introduced and discussed below and in Sections 4.2.3, 4.3.4 and 4.4.4.

Note that extending well-known graph notions to hypergraphs or digraphs is far from straightforward and usually several competing variants emerge. Then the fitness measures are the usefulness, interactions, elegance and simplicity of those concepts. And frequently, alternative notions have their strengths in different settings. In our opinion, the games and basic subgraph and minor monotonicity are great basic criteria.

To illustrate, consider the extensions of tree-width to directed graphs: *DAG-width* by Obdržálek [Obd06], *directed tree-width* by Johnson et al. [JRST01], and *Kelly-width* by Hunter and Kreutzer [HK08]. All these measures are also defined

by games, some of them related to the *helicopter game* characterising tree-width. See the *Graph minors* series for details on tree-width and the game, namely part III. [RS86].

For *tree-width* there are the hypergraph notions of hypertree-width[1] by Gottlob et al. [GLS99], generalised hypertree-width by Gottlob et al. [GLS03] and marshal-width studied by Adler [Adl04]. These notions are all characterised (and sometimes motivated) by various marshals and robber games. The ultramonotone robber and marshals game was already used by Adler as a tool for constructing hypergraph counterexamples [Adl06]. Generalized hyperpath-width by Miklós [Mik88] is another example in this direction.

Graph minor is a central notion in a big part of modern structural graph theory. Since their introduction in 1983 by Robertson and Seymour in their famous Graph Minors series [RS83], there is a strong steady stream of research and applications. Today, in December 2015, there are over 400 published papers indexed by DBLP[2] with "graph minor" in the title. The minor relation (as well as induced minors, immersions etc.) extends the subgraph relation to more general "structure containment" (see definition in Section 4.2.3). We extend the minor relation to hypergraphs motivated by the symbiotic relations of minors and the games characterising tree-width and tree-depth.

Hypergraph pairs of Section 4.4 are introduced in order to capture a Marshals and robber game with asymmetric game boards (and possible moves) for the marshals and the robber. This framework was introduced in [Adl08] as a common generalization of tree-width of graphs and hypertree-width of hypergraphs. Here it allows us to obtain hypergraph minors, graph minors and induced graph minors as special cases. We also propose this as a confirmation that our notion of tree-depth and minors in hypergraphs are indeed a robust generalization, and in our case we use the general setting to show some of the results on hypergraphs.

4.2 Tree-depth and minors of graphs

In this section we briefly introduce the traditional graph concepts of tree-depth (with its various characterisations) and graph minors.

4.2.1 Tree-depth of graphs

Tree-depth allows several equivalent definitions, namely via rooted forest closure, recursive definition, vertex rankings, centred colourings, and finally a ultramonotone cops and robber game. For all these definitions and their equivalences we recommend the great book *Sparsity* by Nešetřil and de Mendez [NdM12].

[1]Note on the name: The exact hyphenation follows from "tree-depth" being hyphenated while the "hyper-" prefix is usually used in the closed form. However, there is no direct relation to hypertrees, and "hyper-tree-depth" and "hypertreedepth" are other possible variants.

[2]DBLP (`http://dblp.uni-trier.de/`) is a computer science bibliography database by University of Trier.

Rooted forest closure and decomposition

Given a rooted forest F, $\mathrm{clos}(F)$ is the graph obtained from F by adding an edge between every two vertices u, v with u below v (in the rooted forest partial order).

For a graph G, $\mathrm{td}(G)$ is the smallest possible height of a rooted forest F such that $G \subseteq \mathrm{clos}(F)$. Note that while F and G may be assumed to have the same vertex set, $F \subseteq G$ generally does not hold.

This can be also treated as a tree decomposition of G with F the decomposition forest (with an identity mapping between $V(F)$ and $V(G)$) and the condition that every edge $\{u, v\} \in E(G)$ is a subset of some chain of comparable vertices of F. $\mathrm{td}(G)$ is then again the smallest possible height of such F.

Recursive definition

For an empty graph G, $\mathrm{td}(G) = 0$. For a disconnected graph G with components $G_1, \dots G_k$ we have

$$\mathrm{td}(G) = \max\{\mathrm{td}(G_i) \;\Big|\; 1 \leq i \leq k\}.$$

For a connected graph G we have

$$\mathrm{td}(G) = 1 + \min\{\mathrm{td}(G - v) \;\Big|\; v \in V(G)\}.$$

Vertex ranking and centred colouring

A *vertex ranking* of G is a function $r : V(G) \to \{1..k\}$ such that for every two vertices u and v with $r(u) = r(v)$ and every u-v path P, there is some vertex $w \in P$ with $r(w) > r(u)$. The smallest h for which such r exists is $\mathrm{td}(G)$.

A *centred colouring* of G is a mapping $c : V(G) \to C$ such that for every connected $H \subseteq G$ some colour appears exactly once in H. $\mathrm{td}(G)$ is the smallest number of colours where such c exists.

4.2.2 Cops and robber game

In [GHK+09], Ganian et al. give a cop and robber game characterization of tree-depth, calling the game "lift-free" k-cops and robber game (as a modification of the Helicopter game mentioned in Section 1.3.1). In [NdM12], Nešetřil and de Mendez introduce an equivalent game called *selection-deletion game*. We prefer to call the game the *ultramonotone cops and robber game*.

At the beginning of the game, the robber picks his starting vertex and all the k cops start in a reserve off-board. Then in the i-th round the cops announce a vertex v_i to place a cop from the reserve. The robber may then move from his current position along a cop-free path of any length (moving in a component of $G - \{v_1, \dots v_{i-1}\}$). Then one of the cops off-board is placed at v_i. The cops are never removed from the board. The cops win the game if they capture the robber before running out of cops in the reserve (that is in k rounds), the robber wins otherwise. For a given G, $\mathrm{td}(G) \leq k$ if and only if k cops have a winning strategy.

The "ultramonotonicity" in the name of the game refers to the fact that cops are never removed. This is related to a weaker *monotonicity condition* which demands that any area once made inaccessible for the robber may never get

accessible to him again. The condition on never moving the cops may be seen as a strengthening of this requirement. Below we modify the game to marshals and hypergraphs.

4.2.3 Graph minors

As mentioned in Section 4.1, *graph minors* have been introduced in 1983 by Robertson and Seymour [RS83] and are already a standard part of graph theory tools. We give a short definition and remember several useful properties.

A graph H is a *minor* of a graph G, denoted $G \preccurlyeq H$ if H can be obtained from a subgraph of G by a sequence of edge contractions. Alternatively, a minor of G is any graph obtainable from G by a series of vertex deletions, edge deletions and contractions in any order.

It is straightforward that the minor relation is a partial order, and it generalises the subgraph relation. One of the most important results on graph minors is the following by Robertson and Seymour.

Theorem 58 (Graph Minor Theorem, [RS04])**.** *Any infinite set of finite simple graphs contains two minor-comparable elements.*

The main consequence of this theorem is the *forbidden minor characterisation*: For any non-trivial minor-closed graph class $\mathcal{G}$, there is a *finite* set of graphs $\{F_1, \dots F_k\}$ such that:

$$\forall G : G \in \mathcal{G} \iff (\forall i : F_i \not\preccurlyeq G)\,.$$

An alternative definition of the minor relation is via *minor models*: A minor model of H in G is a function m mapping every $V(H)$ to a *connected* subgraph of G such that $m(u) \cap m(v) = \emptyset$ for all distinct $u, v \in V(H)$ and for every $uv \in E(H)$ the subgraphs $m(u)$ and $m(v)$ of G are connected by a direct edge of G. We then have $H \preccurlyeq G$ if and only if there is a minor model of H in G.

The equivalence can be easily demonstrated by taking the target subgraphs of m as the connected subgraphs to be contracted to a single vertex each, yielding a supergraph of H. On the other hand, the set of vertices of G eventually contracted to form some vertex $v \in V(H)$ can be taken as $m(v)$. We omit the details.

4.3 Tree-depth and minors of hypergraphs

In this section we generalise the above concepts to hypergraphs, showing their most interesting properties. Section 4.4 introduces these concepts in an even more general setting, sometimes yielding definitions below as specialisations, but we present the matter in the (logical) order of increased generality. However, some of the proofs are postponed to the more general setting.

4.3.1 Hypertree-depth

Hypertree-depth is defined as follows. Let H be a hypergraph. A *decomposition forest* of H is a tuple (F, C), where F is a rooted forest, and $C\colon V(F) \to E(H)$ is a mapping, satisfying the following conditions.

(dec1) For every vertex $v \in V(H)$ there is a node $t \in V(F)$ such that $v \in C(t)$.

(dec2) For every edge $e \in E(\underline{H})$ there are $\leq_F$-comparable nodes $s, t \in V(F)$ such that $e \subseteq C(s) \cup C(t)$.

(dec3) All $s, t \in V(F)$ satisfy: If $C(s) \cap C(t) \neq \emptyset$ then the node $s \wedge t$ exists and $C(s) \cap C(t) \subseteq \bigcup_{r \in \downarrow(s \wedge t)} C(r)$.

The *hypertree-depth* of a hypergraph H, $\mathrm{hd}(H)$, is the minimum height of a rooted forest F, taken over all decomposition forests (F, C) of H. Note that if H has isolated vertices (not present in any edge), then H does not have a decomposition forest. In this case $\mathrm{hd}(H) = \infty$.

It is easy to see that we can equivalently replace condition **(dec2)** by the following condition **(dec2′)**.

(dec2′) every hyperedge $e \in E(H)$ satisfies $e \subseteq \bigcup_{t \in Z} C(t)$ for some chain Z in F.

The trivial forest consisting of isolated vertices, each of them corresponding to one hyperedge of a given hypergraph satisfies **(dec1)** and **(dec2)**. Thus, condition **(dec3)** might be viewed as the condition enforcing non-triviality of a decomposition forest and cannot be omitted. In particular, for a decomposition forest of a hypergraph H, condition **(dec3)** ensures that every tree of the decomposition forest corresponds to a union of connected components of $\underline{H}$, i.e. two hyperedges can be mapped to different trees of a decomposition forest only if their vertices are contained in two distinct connected components of $\underline{H}$.

When we view a graph G as a 2-uniform hypergraph, generally $\mathrm{td}(G) \neq \mathrm{hd}(G)$. However, $\mathrm{td}(G)$ and $\mathrm{hd}(G)$ are linearly bounded.

Lemma 59. *Any hypergraph H with edges of size at most k satisfies*

$$k \,\mathrm{hd}(H) \geq \mathrm{td}(\underline{H}) \geq \mathrm{hd}(H).$$

In particular, any graph G satisfies $2\,\mathrm{hd}(G) \geq \mathrm{td}(G) \geq \mathrm{hd}(G)$.

Proof. Let $G = \underline{H}$. For the first inequality, assume we have an optimal decomposition tree (F, C) of H. Define $f : V(G) \to V(F)$ by letting $f(v) = u$ with u the lowest node of F with $v \in C(u)$. This node is unique by **(dec3)**.

Now f may not be injective – up to k vertices may map to the same node (since edges of H have size at most k). We can resolve that by splitting every such node u into a path $u_1, \ldots u_k$, making u_1 adjacent to the parent of u and u_k

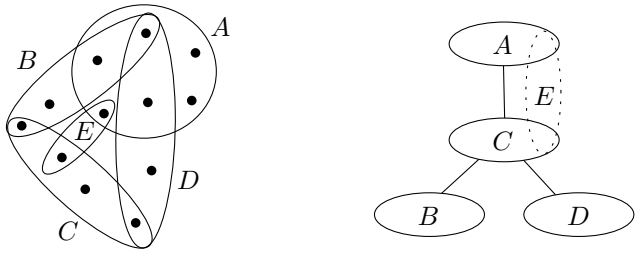


Figure 4.1: Example of a decomposition tree of a hypergraph.

to all children of u. We also change f to map the vertices to $u_1, \dots u_k$ arbitrarily injectively. Let F' be the resulting forest and f' the resulting injective mapping.

We now have $G \subseteq \mathrm{clos}(F')$ as shown by f': let $e = xy \in E(G)$ and s, t be as in **(dec2)**. Then $f'(x) \leq s$ and $f'(y) \leq t$ and therefore $f'(x)$ and $f'(y)$ are $\leq_{F'}$-comparable. Obviously, the depth of F' is at most k times the depth of F.

For the second inequality, let F be an optimal forest with $G \subseteq \mathrm{clos}(F)$. Define $C : V(F) = V(G) \to E(G)$ with $C(u)$ an arbitrary edge of containing u. Then (F, C) trivially satisfies **(dec1)** and **(dec2)**. For **(dec3)**, have $s, t \in V(F)$ and $U = C(s) \cap C(T)$. Note that s and t are also vertices of G and $s \in C(s)$ and $t \in C(t)$. Now either s and t are comparable, in which case the required node is the lower of s, t. Or s and t are incomparable, in which case every $u \in U$ is adjacent to both s and t in $G = \underline{H}$, and therefore is below both s and t and $u \in C(u)$. This gives us that $U \subseteq \bigcup_{r \in \downarrow(s \wedge t)} C(r)$. □

Note that this is tight, as we have $\mathrm{td}(K_{kn}) = 2\,\mathrm{hd}(K^k_{kn}) = kn$. (with K^k_n the complete k-uniform hypergraph on n vertices). Furthermore, for graphs we have $\mathrm{td}(P_{2^n-2}) = \mathrm{hd}(P_{2^n-2}) = n$, where P_n is the path with n edges (cf. Example 77). For $k \geq 2$, we can replace each edge of P_n with a size k hyperedge, introducing new vertices for each edge. This gives an almost tight bound (slack by an additive factor of $k - 2$).

It might seem desirable for a hypergraph extension of tree-depth definition that when we view a graph G as a hypergraph, we would have $\mathrm{td}(G) = \mathrm{hd}(G)$. Our definition does not satisfy this (and neither do tree-width and generalized hypertree-width, nor path-width and hyperpath-width). The reason for this is that tree-depth counts the vertices in the decomposition, while the hypergraph measures hypertree-width, hyperpath-width and hypertree-depth count the number of edges covering certain vertex sets. This stems from the applications in database theory and constraint satisfaction problems, where tuples are counted rather that elements.

4.3.2 Alternative characterisations of hypertree-depth

Hypertree-depth allows a natural inductive definition as well as characterisations by *vertex-hyperrankings* and *centred hypercolourings*, as done for tree-depth by Nešetřil et al. [NdM06]. We give these in a slightly more general setting for hypergraph pairs in Section 4.4.2 and do not repeat the hypergraph version here. The definitions and properties follow naturally for any H by considering the hypergraph pair $(H, \underline{H})$.

4.3.3 Marshals and robber game

We generalise the game defined in Section 4.2.2. The *ultramonotone*[3] *robber and marshals game* on a hypergraph H is played by two players: first controlling k the marshals and second the robber. The marshals are initially off-board and are

[3]The word 'ultramonotone' refers to the fact that the ultramonotone robber and marshals game is 'monotone' in the sense of [GLS03] (robber's area monotonically shrinks). Not removing the marshals is much stronger – and hence the term.

placed on hyperedges of H. Once they occupy a hyperedge, they stay there for the rest of the game. The robber moves on vertices of H.

In a play of the ultramonotone robber and marshals game, the robber begins by choosing a vertex $r_0 \in V(H)$ and placing the robber on r_0. Then, in every round of the game with robber at r and marshals on $M \subseteq E$, the marshals select a hyperedge $e' \in E(H)$ and reveals it to the robber. The robber then selects a vertex $r' \in V(H)$ that is connected to r by a path in $\underline{H} \setminus \bigcup M$. He moves the robber to r' and the marshals place a new marshal on e'.

The marshals win if they reach a position $r \in \bigcup M$ before they run out of marshals. The robber wins if the marshals run out of marshal tokens. A *winning strategy* for the marshals is then a winning strategy for the marshals such that every play played according to the strategy has length at most k rounds (the initial robber placement is not considered a round).

This game characterizes hypertree-depth. The following theorem follows directly from Lemma 64, Theorem 69 and Observation 68 below.

Theorem 60. *A hypergraph H has $\mathrm{hd}(H) \leq k$ if and only if k marshals have a winning strategy for the ultramonotone robber and marshals' game on H.*

4.3.4 Hypergraph minors

In this section we propose a concept of minors for hypergraphs. The concept of minor maps and models are omitted, as they follow from applying the definitions for hypergraph pairs (introduced in Section 4.4.4) to $(\underline{H}, H)$.

For a hypergraph H, the hypergraph obtained from H by *contracting* an edge $e = \{x, y\}$ such that $e \subseteq e' \in E(H)$ is the hypergraph H/e with $V(H/e) = V(H) \setminus e \cup \{v_e\}$ and

$$E(H/e) = \Big\{h \in E(H) \;\Big|\; h \cap e = \emptyset\Big\} \cup \Big\{(h \setminus e) \cup \{v_e\} \;\Big|\; h \in E(H),\, h \cap e \neq \emptyset\Big\}.$$

In other words, v_e is the new contracted vertex and every hyperedge containing some $v \in e$ is set to contain v_e. Note that we usually consider $|e| = 2$ which corresponds to contracting $e \in E(\underline{H})$.

Let H and H' be hypergraphs. Then H is a *minor* of H', denoted $H \preccurlyeq H'$ if H can be obtained from H' by a sequence of the following operations:

- vertex deletion,
- contraction of an edge $e \in E(\underline{H})$,
- addition of a hyperedge e such that $\underline{H}[e]$ is a clique,
- deletion of a proper sub-hyperedge $e \in E(H)$, $e \subsetneq e' \in E(H)$.

Note that we do not allow arbitrary hyperedge removal, so a sub-hypergraph is not necessarily a minor. From our game and decomposition point of view, a hypergraph with a hyperedge containing all the vertices would be trivial, while removing this edge (even if preserving the underlying graph) might increase the complexity arbitrarily. While this argument is not rigorous, it is not bound to a single width parameter or game, but captures most scenarios where the hyperedges represent a unit to be taken, counted or paid for.

The first two operations are analogous to traditional graph minors. In the spirit of the previous paragraph, the third operation only decreases the 'cost' without changing the underlying graph, and the last removes hyperedges that are 'made redundant' by other edges.

Hypergraph minors behave as graph minors of the underlying graphs.

Lemma 61. *Let H and H' be hypergraphs. If $H \preccurlyeq H'$, then $\underline{H} \preccurlyeq \underline{H'}$.*

Proof. It is straightforward to see that every hypergraph minor operation on H has an effect on $\underline{H}$ corresponding to some minor operation (or no effect at all). □

Note that the converse does not hold: If H is the path of length two and H' is the triangle (both viewed as hypergraphs), then $\underline{H} \preccurlyeq \underline{H'}$, but $H \not\preccurlyeq H'$ because we do not allow deleting arbitrary hyperedges.

It seems desirable to have graph minors as a special case of hypergraph minors. For this, we move one step further and generalise minors to hypergraph pairs in Section 4.4.4, getting a if-and-only-if relation with graph minors in Observation 72. There we also introduce hypergraph minor models and maps.

4.4 Tree-depth and minors of hypergraph pairs

In the ultramonotone robber and marshals game on a hypergraph H we restricted the robber's movement to the underlying graph $\underline{H}$. However, we can lift this restriction and instead choose any graph G with $V(G) = V(H)$ as the 'game board' for the robber. With this, we generalize our setting from hypergraphs to hypergraph pairs; a setting used previously by Isolde Adler [Adl08] to define a common generalization of tree-width and hypertree-width.

This allows us to handle both hypertree-depth of hypergraphs and tree-depth of graphs at the same time. Moreover, we will use this slightly more flexible framework in Section 4.4.4 to capture both minors in hypergraphs and (traditional) minors in graphs.

A *hypergraph pair* is defined as a pair (G, H) consisting of a graph G and a hypergraph H with $V(G) = V(H)$. There is no other restriction on the edges or the hyperedges. A hypergraph pair (G', H') is an *(induced) subhypergraph pair* of (G, H) if G' is an (induced) subgraph of G and H' is an (induced) subhypergraph of H.

4.4.1 Hypertree-depth of hypergraph pairs

We also extend the definition of hypertree-depth to hypergraph pairs (G, H) and give some intuition below the definition.

A *decomposition forest* of (G, H) is a triple (F, B, C), where F is a rooted forest, $B\colon V(F) \to 2^{V(G)}$, and $C\colon V(F) \to E(H)$ are mappings, satisfying the following conditions.

(pdec0) $B(t) \subseteq C(t)$ for every $t \in V(F)$,

(pdec1) For every vertex $v \in V(G)$ there is a node $t \in V(F)$ such that $v \in B(t)$.

(pdec2) For every edge $e \in E(G)$ there are $\leq_F$-comparable nodes $s, t \in V(F)$ such that $e \subseteq B(s) \cup B(t)$,

(pdec3) All $s, t \in V(F)$ with $s \neq t$ satisfy $B(s) \cap B(t) = \emptyset$.

The *hypertree-depth* $\mathrm{hd}(G, H)$ of a hypergraph pair (G, H) is the minimum height of a rooted forest F, taken over all decomposition forests (F, B, C) of (G, H).

Note that $\mathrm{hd}(G, H) = 1$ if and only if for every connected component $C \subseteq V(G)$ of G, there exists a hyperedge $e \in E(H)$ such that $C \subseteq e$. $\mathrm{hd}(G, H)$ is bounded from above by the minimum number of hyperedges of H necessary to cover $V(G)$.

The forest F and the mapping C play the same roles as on hypergraphs, and intuitively, $B(s)$ indicates which of the vertices of $C(s)$ are 'covered' in the branch of F containing s. If $G \neq \underline{H}$, we may have a hyperedge e with $G[e]$ disconnected and e present as C of multiple nodes s_i. In this case, B distinguishes which connected components of $G[V \setminus B(\downarrow t_i)]$ (with t_i the parent of s_i) are covered by the nodes above s_i.

This 'entire connected components' property of B is illustrated in the following lemma.

Lemma 62. *Let (G, H) be a hypergraph pair with a decomposition forest (F, B, C), and let $X \subseteq V(G)$ be connected in G.*

1. *There exists precisely one connected component T of F such that $X \cap B(T) \neq \emptyset$ and we have $X \subseteq B(T)$.*

2. *If T is a connected component of F and $t \in V(T)$ with $X \subseteq B(T_t) \setminus B(\downarrow t)$, then t has precisely one child s such that $X \subseteq B(T_s)$.*

Proof. **1.** If there were two trees T_1, T_2 of F with $B(T_1)$ and $B(T_2)$ intersecting X, then by **(pdec3)** we have $|X| > 1$, and there is an edge $\{u, v\} \in E(G[X])$ with $u \in B(T_1)$ and $v \in B(T_2)$, contradicting **(pdec2)**.

2. Similarly, if there are two child nodes s_1 and s_2 with $B(T_{s_1})$ and $B(T_{s_2})$ intersecting X, there is an edge $\{u, v\} \in E(G[X])$ with $u \in B(T_{s_1})$ and $v \in B(T_{s_2})$, which is impossible by **(pdec2)** and **(pdec3)**. □

We say that a decomposition tree (F, B, C) of (G, H) is *small*, if $B(t) \neq \emptyset$ for every $t \in V(F)$. Leaving out all the nodes with $B(t) = \emptyset$ (possibly contracting the edge to the parent of t) yields a small decomposition tree with smaller or equal depth.

Replacing condition **(pdec3)** with the following condition **(pdec3′)** we obtain another definition of decomposition forest which more closely resembles that of a decomposition forest of a (single) hypergraph. Trivially, any decomposition forest satisfying **(pdec3)** satisfies **(pdec3′)**. On the other hand we get the following lemma.

(pdec3′) All $s, t \in V(F)$ satisfy: If $B(s) \cap B(t) \neq \emptyset$ then the node $s \wedge t$ exists and $B(s) \cap B(t) \subseteq B(\downarrow (s \wedge t))$,

Lemma 63. *For every decomposition forest (F, B, C) that satisfies **(pdec0)**, **(pdec1)**, **(pdec2)** and **(pdec3′)**, but not necessarily **(pdec3)**, there is a mapping B' such that for all $t \in V(F)$, $B'(t) \subseteq B(t)$ and (F, B', C) is a decomposition forest.*

Proof. We set $B'(t) = B(t) \setminus \bigcup_{s<_F t} B(s)$. Now **(pdec3)** is satisfied: for any $s, t \in F$, $B'(s) \cap B'(t) \subseteq B(s) \cap B(t) \subseteq B(\downarrow (s \wedge t))$ and these vertices have been removed from at least one of $B'(s)$ or $B'(t)$. It is easy to see that **(pdec0)**, **(pdec1)** and **(pdec2)** are preserved by going from B to B'. □

Now we finally connect the hypertree-depth of hypergraphs and hypergraph pairs.

Lemma 64. *Let H be a hypergraph. Then* $\mathrm{hd}(H) = \mathrm{hd}(\underline{H}, H)$.

Proof. We may assume that $\underline{H}$ is connected, or proceed with individual components.

Towards a proof of $\mathrm{hd}(H) \leq \mathrm{hd}(\underline{H}, H)$, let (F, B, C) be a small decomposition forest of $(\underline{H}, H)$ such that the height of F is $\mathrm{hd}(\underline{H}, H)$. We claim that (F, C) is a decomposition forest of H witnessing $\mathrm{hd}(H) \leq \mathrm{hd}(\underline{H}, H)$. By **(pdec2)**, (F, C) satisfies **(dec2)**.

Towards showing that condition **(dec3)** is satisfied, let $s, t \in V(F)$ be vertices in the same component of F and let $v \in C(s) \cap C(t)$. Since (F, B, C) is small, there exists a vertex $w \in B(s) \setminus (\bigcup_{r<s} B(r))$. By **(pdec0)**, $\{v, w\} \in E(\underline{H})$. Hence by condition **(pdec2)** we must have $v \in B(T_s) \cup (\bigcup_{r \in \downarrow s} B(r))$. Applying the analogous argument to t, we must have $v \in B(T_t) \cup (\bigcup_{r \in \downarrow t} B(r))$.

Since B is a partition of $V(G)$, we have a unique u with $v \in B(u)$, $u \leq s \wedge t$, and we are done.

Conversely, if (F, C) is a decomposition forest of H of height $\mathrm{hd}(H)$, let $B(t) := C(t)$ for all $t \in V(F)$. Then (F, B, C) is a decomposition forest satisfying **(pdec3′)** instead of **(pdec3)**. By Remark 63, there is a decomposition forest (F, B', C) of $(\underline{H}, H)$ witnessing $\mathrm{hd}(\underline{H}, H) \leq \mathrm{hd}(H)$. □

Note that in the hypergraph pair framework, we can actually have $\mathrm{td}(G) = \mathrm{hd}(G, H)$ for H a hypergraph consisting of all singletons over $V(G)$:

Lemma 65. *Let (G, H) be a (1-uniform) hypergraph pair where $E(H) = \{\{v\} \mid v \in V(G)\}$. Then* $\mathrm{hd}(G, H) = \mathrm{td}(G)$.

Proof. A small decomposition forest (F, B, C) of such (G, H) has $B = C$, both maps to singletons. Let F' be a copy of F with the set $V(G)$ obtained by replacing $v \in V(F)$ with the only element of $B(v)$. Then we directly have $G = \mathrm{clos}(F')$.

On the other hand, if F is optimal such that $G = \mathrm{clos}(F')$, then (F, s, s) is a decomposition forest of (G, H), where $s(v) = \{v\}$. □

4.4.2 Alternative characterisations of hypertree-depth

In spirit of Nešetřil et al. [NdM06], we present alternative characterisations of hypertree-depth. We give an inductive definition together with definitions based on *vertex-hyperrankings* and *centred hypercolourings*. For brevity, we give

the definitions only for hypergraph pairs, but the definitions and properties for hypergraphs follow easily by using $(\underline{H}, H)$ as the hypergraph pair.

These characterisations and their similarity to those of Nešetřil et al. are indication that our notion of hypertree-depth is the correct hypergraph generalisation of tree-depth. Note that the corresponding definitions for graphs G are obtained by using the hypergraph pair $(G, (V(G), \{\{v\}, v \in V(G)\}))$ (1-uniform hypergraphs as in Lemma 65).

Recursive definition

Similarly to tree-depth of graphs [NdM06], hypertree-depth has an inductive definition.

Lemma 66. *Let (G, H) be a hypergraph pair satisfying $V(G) = \bigcup E(H)$ and $|V(H)| > 0$. Let $G_1, \dots G_p$ be the connected components of G. Then*

$$\mathrm{hd}(G,H) = \begin{cases} 1, & \text{if there exists } e \in E(H) \text{ with } V(G) \subseteq e, \\ 1 + \min\limits_{e \in E(H)} \mathrm{hd}\left(G \setminus e, H[V(G) \setminus e]\right), & \text{if } p = 1 \text{ and } V(G) \setminus e \neq \emptyset \text{ for all } e \in E(H), \\ \max\limits_{i \in \{1,\dots,p\}} \mathrm{hd}\left(G_i, H[V(G_i)]\right), & \text{otherwise.} \end{cases}$$

Proof. We show this by induction on $\mathrm{hd}(G, H)$. If $\mathrm{hd}(G, H) = 1$ then we are in the first case. Now let $\mathrm{hd}(G, H) = k > 1$ and assume the lemma has been proven for all hypergraph pairs with hypertree-depth at most $k - 1$. If G is connected, by Lemma 62 there exists a decomposition *tree* (T, B, C) of height k for $\mathrm{hd}(G, H)$. Let r be the root of T and let $F := T \setminus r$ be the rooted forest obtained from T be deleting r and choosing the children of r as the new roots of F. Then the restriction $(F, B|_{V(F)}, C|_{V(F)})$ shows that $\mathrm{hd}\left(G \setminus C(r), H[V(G) \setminus C(r)]\right) \leq k - 1$ and $\mathrm{hd}(G, H) \geq \mathrm{hd}\left(G \setminus C(r), H[V(G) \setminus C(r)]\right) + 1$.

Conversely, let $e \in E(H)$ and let (F, B, C) be a decomposition forest of height $\mathrm{hd}(G \setminus e, H[V(G) \setminus e])$ for $(G \setminus e, H[V(G) \setminus e])$. Let T be the rooted tree obtained from F by introducing a new node r as the new root and connecting r to all old roots of F by edges. Letting $B(r) = C(r) = e$ we see that $\mathrm{hd}(G, H) \leq \mathrm{hd}\left(G \setminus e, H[V(G) \setminus e]\right) + 1$.

If G is not connected, it is easy to see that the third case applies. □

Vertex-hyperrankings and centred hypercolourings

Let (G, H) be a hypergraph pair. A *vertex-hyperranking* of (G, H) is a (not necessarily proper) colouring $c\colon V(G) \to \{1, \dots k\}$ such that for every $i \leq k$ and for every connected component X of $G_i := G[\{v \in V(G) \mid c(v) \geq i\}]$ there is a hyperedge $e \in E(H)$ such that $\{v \in X \mid c(v) = i\} \subseteq e$.

Let (G, H) be a hypergraph pair. A *centred hypercolouring* of (G, H) is a (not necessarily proper) colouring of $V(G)$ such that for every induced subhypergraph pair (G', H') where G' is connected, there exists a colour $a = a(G', H')$ colouring at least one vertex of G' and a hyperedge $e \in E(H')$, such that $\{v \in V(G') \mid v \text{ has colour } a\} \subseteq e$.

Theorem 67. *For every hypergraph pair (G, H), the following are equivalent:*

- $\mathrm{hd}(G, H)$ *has a vertex-hyperranking with labels* $\{1, \dots k\}$,
- (G, H) *has a centred hypercolouring with at most* k *colours, and*
- $\mathrm{hd}(G, H) \leq k$.

Proof. We may assume that G is connected. Assume that (G, H) has a centred hypercolouring with k colours. Then there exists a colour a and a hyperedge $e \in E(H)$ such that $\emptyset \neq X := \{v \in V(G) \mid v \text{ has colour } a\} \subseteq e$. If $k = 1$, then $V(G) \subseteq e$ and hence $\mathrm{hd}(G, H) \leq 1$.

Let $k > 1$ and assume that we have proved the theorem for all hypergraph pairs having centred hypercolourings with $k - 1$ colours. Since the induced subhypergraph pair $(G \setminus X, H \setminus X)$ has a centred hypercolouring with $k - 1$ colours, we know that $\mathrm{hd}(G \setminus X, H \setminus X) \leq k - 1$, witnessed by a decomposition forest (F, B, C). We create a new root r with $B(r) = X$ and $C(r) = e$, and we let the old roots of F be the children of r. In this way we obtain a decomposition tree witnessing $\mathrm{hd}(G, H) \leq k$.

Now assume that $\mathrm{hd}(G, H) = k$, witnessed by a decomposition forest (F, B, C). Rank a $v \in V(G)$ with the minimum depth of a node $t \in V(F)$ with $v \in B(t)$. We claim that this is a vertex-hyperranking: Let (G', H') be an induced subhypergraph pair of (G, H) such that G' is connected. By Lemma 62 there is a connected component $T \subseteq F$ with $V(G) \subseteq B(T)$ and T is the unique component with $V(G) \cap B(T) \neq \emptyset$. Let t in $V(T)$ be the node of minimum depth such that $B(t) \cap V(G) \neq \emptyset$. From **(pdec2)** we get that all $V(G')$ are above t and we have a proper vertex-hyperranking.

The last of the three implications follows from the fact that every vertex-hyperranking of a hypergraph pair (G, H) is a centred hypercolouring. □

4.4.3 Marshals and robber game on hypergraph pairs

The *ultramonotone* robber and marshals game on a hypergraph pair (G, H) is played like the ultramonotone robber and marshals game on a hypergraph introduced in Section 4.3.3, with the difference that the marshals 'play on H' while the robber 'plays on G': The marshals occupy the hyperedges of H and the robber moves via paths in G, but only on the vertices not blocked by the already placed marshals.

Formally, in a play of the ultramonotone robber and k-marshals game on a hypergraph pair (G, H), the robber begins by choosing a vertex $r_0 \in V(H)$ and placing the robber on r_0. Then, in every round of the game with the robber on r and the marshals on M, the marshals select a hyperedge $e' \in E(H)$ and reveal it to the robber. The robber then selects a vertex $r' \in V(H)$ that is connected to r by a path in $G \setminus \bigcup M$. He moves the robber to r' and the marshals place a marshal token on e'.

Again, the marshals win if they reaches a position with $r \in \bigcup M$, the robber wins if the marshals run out of their k tokens before that. The winning strategies are defined as in Section 4.3.3 When the the robber is on r and the marshals are on M, the component of $G \setminus \bigcup M$ containing r is called the *robber space (with respect to M).*

This is a generalization of the game for hypergraphs, as stated by the following straightforward observation:

Observation 68. *For a hypergraph H, k marshals have a winning strategy on H if and only if k marshals have a winning strategy on $(\underline{H}, H)$.*

We now present and prove the equivalence of the ultramonotone robber and marshals game and the hypertree-depth on hypergraph pairs. This also yields the proof of the equivalence on hypergraphs (Theorem 60).

Theorem 69. *A hypergraph pair (G, H) has $\mathrm{hd}(G, H) \leq k$ if and only if k marshals have a winning strategy for the ultramonotone robber and marshals game on (G, H).*

Proof. If a vertex of $V(G)$ is not contained in any hyperedge of H, then $\mathrm{hd}(G, H) = \infty$ and there is no marshals' winning strategy. From now on we assume that $V(G) \subseteq \bigcup E(H)$.

First, let (F, B, C) be a decomposition tree of (G, H) of height $\mathrm{hd}(G, H)$. We now describe a strategy for the marshals. Intuitively, the marshals play along the C-labels of a branch of F, starting at a root and always following the robber.

First, the robber selects a vertex $r_0 \in V(G)$ and the play is in position $(r_0, \emptyset)$. Since the robber is only allowed to move along paths in G, he will now stay in the connected component $X_0 \subseteq V(G)$ with $r_0 \in X_0$.

By Lemma 62, there exists precisely one component T of F such that $X \subseteq B(T)$. The marshals select t to be the root of T and announce their move to $C(t)$. The robber moves to a vertex r that is connected to r_0 in G and the marshals complete their move.

If $r \in C(t)$, the marshals win. Otherwise, by Lemma 62, there is a unique child s of t such that $r \in B(T_s)$, and the marshals announce their move to $C(s)$. Continuing in this way along a chain in T, the marshals capture the robber in at most k rounds.

Towards a proof of the other implication, suppose that k marshals have a winning strategy $\mathcal{S}$ in the ultramonotone robber and marshals game on (G, H).

First, observe that we may assume that the marshals' moves according to $\mathcal{S}$ depend only on the robber space, that is the component of $V(G) \setminus \bigcup M$ containing r. Indeed, a robber on r_1 and r_2 from the same robber space has exactly the same possibilities to move before the marshal token is placed.

Second, we may assume that in each move played according to $\mathcal{S}$ the robber space size decreases by at least one vertex as any marshal moves not shrinking the robber space can be omitted.

Now we construct the nodes of a tree T while keeping track of a game position for every node constructed. We construct a separate rooted decomposition for every connected component A of G with a root vertex t_0 as follows. Suppose in the first move, the robber chooses a vertex $r_0 \in V(A)$. The node t_0 corresponds to the game position $(r = r_0, M = \emptyset)$.

Now for every node t of T corresponding to a position (r, M) we recursively define $C(t)$, $B(t)$, the set of t's children and their corresponding game positions. Let X be the robber space. Let $e = \mathcal{S}(r, M)$ be the next marshals' move according to $\mathcal{S}$. Then set $B(t) = e \cap X$ and $C(t_0) = e$. Also, let X_i be the components of $X \setminus e$. For every X_i, choose arbitrary $r_i \in X_i$ and add a node t_i to T as a child of t.

The position corresponding to t_i is $(r_i, M \cup \{e\})$. Note that a node corresponding to a position winning for marshals (when $r \in \bigcup M$) has no children.

Let F be the rooted forest obtained by doing this for all connected components of G. We claim that (F, B, C) is a decomposition forest of height at most k for (G, H). Condition **(pdec0)** is satisfied by construction.

Condition **(pdec1)** is satisfied, because $\mathcal{S}$ is a winning strategy and hence the robber is captured in one leaf node of F, regardless on which vertex he chooses to reside.

Condition **(pdec3)** is satisfied: $B(s)$ and $B(t)$ for s, t comparable are disjoint by the construction, for s, t incomparable we have that the robber spaces X_s at s and X_t at t are disjoint.

For **(pdec2)**, let $\{u, v\} \in E(G)$. Assume the robber has the strategy to play staying inside the set $\{u, v\}$ through the entire game. Since $\mathcal{S}$ is a winning strategy, there are hyperedges e and f taken by the marshals in a single play with $u \in e$ and $v \in f$. By construction of (F, B, C), there are nodes s, t with $e \in C(s)$ and $f \in C(t)$ in a chain corresponding to the play above.

Since any play of $\mathcal{S}$ has at most k rounds, F has height at most k. □

4.4.4 Hypergraph pair minors

We now generalise the notion of graph minors to hypergraph pairs along the lines of Section 4.3.4, and provide additional general properties and examples.

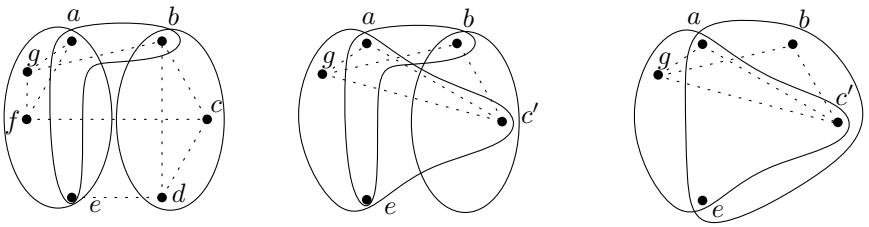
We start with the operations: Let (G, H) be a hypergraph pair and let $v \in V(G)$. The hypergraph pair $(G \setminus v, H[V(G) \setminus v])$ is called the hypergraph pair obtained from (G, H) by *deleting vertex* v, denoted $(G, H) \setminus v$. For $e \in E(G)$, the hypergraph pair $(G \setminus e, H)$ is called the hypergraph pair obtained from (G, H) by *deleting edge* e. Let $e \subseteq V(H)$, then $(G, H+e)$, where $H+e = (V(H), E(H) \cup \{e\})$, is the hypergraph pair obtained from (G, H) by *adding hyperedge* e. If $e, e' \in E(H)$ and $e \subsetneq e'$, then $(G, H \setminus e)$ is the hypergraph pair obtained from (G, H) by *deleting hyperedge* e. Let $e = \{x, y\} \in E(G)$. Then the hypergraph pair obtained from (G, H) by *contracting edge* e, denoted by $(G, H)/e$, is the hypergraph pair obtained by contracting e in both parts, i.e. $(G, H)/e = (G/e, H/e)$.

A hypergraph pair (G', H') is a *minor* of (G, H), denoted by $(G', H') \preccurlyeq (G, H)$, if (G', H') can be obtained from (G, H) by a sequence of the following operations:

- vertex deletion,
- graph edge contraction,
- graph edge deletion,
- hyperedge addition, and
- deletions of a proper sub-hyperedge.

Note that if H is a hypergraph and (G, H') is a hypergraph pair with $(G, H') \preccurlyeq (\underline{H}, H)$, then G is not necessarily the underlying graph of H' (for example, G could be obtained from $\underline{H}$ by deleting an edge).

The intuitive motivation for these operations is similar to hypergraph minor operations, viewing the graph as defining the complexity and the hypergraph as the means of decomposition or covering it. Namely, the first two and the last operations are the same as for hypergraphs; the third decreases the complexity



a) a hypergraph pair b) contraction of edge cf and deletion of vertex d c) addition of hyperedge $abc'e$ and deletion of its proper subhyperedges abe and bc'

Figure 4.2: Example of minor operations on a hypergraph pair (edges of the graph are dotted).

of the underlying structure without decreasing the covering potential; and the fourth is similar to 'addition of hyperedges on cliques' in hypergraph minors, but here we need not restrict it by the underlying graph.

The following lemma shows a direct connection between hypergraph minors and hypergraph pair minors.

Lemma 70. *Let H and H' be hypergraphs. Then $H \preccurlyeq H'$ if and only if $(\underline{H}, H) \preccurlyeq (\underline{H'}, H')$.*

Proof. The "only if" follows from the fact that the hypergraph minor operations are special cases of minor operations in hypergraph pairs of the form $(\underline{H}, H)$.

For the "if" part, let $o_1, \dots o_k$ be a sequence of hypergraph pair minor operations taking $(G_0, H_0) = (\underline{H'}, H')$ to $(G_k, H_k) = (\underline{H}, H)$. We modify the sequence to only contain the operations for hypergraph minors, showing $H \preccurlyeq H'$. Notice that in any intermediate state (G_i, H_i), we have $G_i \subseteq \underline{H_i}$.

First, we make sure that all the vertex deletions come first in the sequence by "bubbling" every vertex deletion to front, adjusting the sequence locally to get the same result in the following way.

When swapping v'-deletion with uv-edge contraction resulting in v', drop both operations and replace them with u-deletion and v-deletion. When swapping v-deletion with uv-edge-deletion, drop the edge deletion. When swapping v-deletion with e-hyperedge-addition such that $v \in e$, replace e-addition with $e \setminus \{v\}$-addition. When swapping v-deletion with e-hyperedge-deletion such that e is a proper subhyperedge, we either replace it with $e \setminus \{v\}$-deletion (if $v \in e$), remove (if $e \cup \{v\}$ is a hyperedge), or leave it as-is (otherwise). Swapping two operations on disjoint vertex sets requires no changes.

Second, in a similar manner, we make sure all the edge contractions come just after the vertex deletions. Assume we "bubble" a uv-contraction resulting in vertex v'. When swapping with vw-edge-removal, replace the removal with $v'w$-removal (if uw is not an edge) or drop the edge removal (if uv is an edge). Swapping with hyperedge operations and independent edge contractions requires no changes.

Now we have a modified sequence still resulting in $(G_k, H_k) = (\underline{H}, H)$. Notice that all vertex deletions and edge contractions now preserve $G_i = \underline{H_i}$. If this is true for all the operations, it is easy to see that there are no edge deletions and the hyperedge operations satisfy the required conditions and we are done.

Otherwise, let i be first such that $G_i \neq \underline{H_i}$ and take $uv \in E(\underline{H_i}) - E(G_i)$. All subsequent operations (hyperedge addition, removal of a proper subhyperedge, edge deletion) preserve the fact that $uv \notin E(G_j)$ and $uv \in E(\underline{H_j})$ for all $j > i$, a contradiction with $(G_k, H_k) = (\underline{H}, H)$. □

The following example shows some basic properties of minors.

Example 71. *1. If H is an induced subhypergraph of H' then $H \preccurlyeq H'$.*

2. Let $n \in \mathbb{N}$. Any hypergraph pair (G, H) with $|V(G)| \leq n$ is a minor of (K_n, I_n).

3. We have $P_3 \preccurlyeq K_3$, but $(P_3, P_3) \not\preccurlyeq (K_3, K_3)$.

However, minors in hypergraph pairs do generalise minors in graphs in the following sense.

Observation 72. *Let G and F be graphs. Then*

$$\begin{aligned} G \preccurlyeq F &\iff (G, K^0_{V(G)}) \preccurlyeq (F, K^0_{V(F)}) \\ &\iff (G, K^1_{V(G)}) \preccurlyeq (F, K^1_{V(F)}) \end{aligned}$$

where $K^i_V = (V, \binom{V}{i})$ is the complete i-uniform hypergraph on V.

Minor maps and models

Minor maps and minor models are well-known concepts for graph minors. We generalise them to the hypergraph pair setting. The hypergraph variants are not specified explicitly and can be obtained by specialising the pair concepts and properties to $(\underline{H}, H)$.

We first recall the definitions graphs.

Let G and G' be graphs. A *minor map* (from G to G') is a function μ defined on $V(G) \cup E(G)$, that takes a vertex $v \in V(G)$ to a non-empty connected subset $\mu(v) \subseteq V(G')$ and an edge $e \in E(G)$ to an edge $\mu(e) \in E(G')$ such that

- any vertices $u, v \in V(G)$ with $u \neq v$ satisfy $\mu(u) \cap \mu(v) = \emptyset$, and
- for every edge $\{u, v\} \in E(G)$ the image $\mu(\{u, v\})$ connects the vertex images $\mu(u)$ and $\mu(v)$.

It is straightforward to verify that there is a minor map from G to G' if and only if $G \preccurlyeq G'$.

The subgraph $\bigcup_{v \in V(G)} G'[\mu(v)] \cup \bigcup_{e \in E(G)} \mu(e)$ of G' is called a *minor model* of G in G'.

Now for hypergraph pairs: Let (G, H) and (G', H') be hypergraph pairs, and let μ be a minor map from G to G'. We say that μ is a *minor map* from (G, H) to (G', H'), if for every hyperedge $e \in E(H')$, the set $\{v \in V(G) \mid \mu(v) \cap e \neq \emptyset\}$ is a subset of some hyperedge of H or empty.

We call a hypergraph pair (M, F) a *minor model* of (G, H) in (G', H'), if M is a model of G in G' and $F = H'[V(M)]$.

The following lemma connects minor maps and minor models to minors. We only sketch the (otherwise straightforward) proof.

Lemma 73. *Any two hypergraph pairs (G, H) and (G', H') satisfy*

$$(G, H) \preccurlyeq (G', H') \iff \text{there is a minor map from } (G, H) \text{ to } (G', H').$$

Proof. First the 'only if' part. Assume a minor operation sequence. We compute a minor map $(G, H) \to (G', H')$ for every prefix of the operation sequence, starting with identity and adapting it locally with every minor operation. These modifications straightforwardly follow for every operation and we omit the discussion.

For the 'if' part, assume a minor map μ. To get the minor, first delete all vertices not in the image of μ, contract the individual images of μ (arbitrarily) and then remove edges and hyperedges as necessary. Again, we omit the details. □

4.5 Computational complexity and minor ordering

For graphs, a famous result by Robertson and Seymour shows that testing for a fixed minor is solvable in cubic time, i.e. for a fixed graph G, given a graph G', there is a cubic time algorithms that decides whether $G \preccurlyeq G'$ holds [RS95].

In contrast, testing for a fixed hypergraph minor can be NP-hard. Fellows et al. [FKMP95] show that there exists a graph G_0 such that testing for G_0 as an induced minor is NP-complete. We extend this to the following.

Lemma 74. *There exists a graph G_0 such that deciding whether a given hypergraph pair (G, H) has (G_0, G_0) as a minor is NP-complete.*

Proof. For NP-hardness, observe that for any pair of graphs G_0 and G the graph G_0 is an induced minor of G' if and only if $(G_0, G_0) \preccurlyeq (G', G')$, and use the result of Fellows et al. [FKMP95]. The hypergraph pair minor problem is obviously in NP. □

The induced minor ordering is known to have infinite antichains, as shown by Matoušek et al. [MNT88]. This implies that the minor orderings of hypergraphs and hypergraph pairs have infinite antichains as well. Here we give an explicit example of an infinite $\preccurlyeq$-antichain of hypergraph pairs.

Remark 75. *For every integer $n \geq 0$ let G_n be an arbitrary graph with n vertices and let $K_V^i = (V, \binom{V}{i})$ as in Observation 72.*

Then the sequence $\left((G_n, K^{n-1}_{V(G_n)})\right)_{n\in\mathbb{N}}$ of hypergraph pairs is an infinite $\preccurlyeq$-antichain. In particular, the sequence $\left(K_n^{n-1}\right)_{n\in\mathbb{N}}$ is an infinite hypergraph $\preccurlyeq$-antichain.

Proof. Let $i, j \in \mathbb{N}$ with $i < j$. To obtain G_i as a minor of G_j, we have to use at least one vertex deletion or edge contraction. The first operation of either kind leaves a hyperedge covering all the vertices of G_j (or such hyperedge was created before the operation). We cannot remove this hyperedge so we cannot produce $(G_i, K^{i-1}_{V(G_i)})$ as a minor. □

4.6 Interaction with other invariants

In this section we compare hypertree-depth to generalised hyperpath-width [Mik88] and to generalised hypertree-width [GLS03], and show that these invariants do not increase under taking minors. These properties are shown in the more general setting of hypergraph pairs, but trivially follow for hypergraphs as well.

We first recall the definition of tree decomposition, hyperpath-width and hypertree-width.

Given a graph G, (T, χ) is a *tree decomposition* of G when T is a tree and $\chi : V(T) \to 2^{V(G)}$ such that

1. $\bigcup_{v \in V(G)} \chi(v) = V(G)$
2. $\forall e \in E(G) \exists t \in T(V) : e \subseteq \chi(t)$
3. $\forall v \in V(G)$: the set $T_v = \{t \in V(T) \mid v \in \chi(t)\}$ induces a connected subtree of T.

For more information on tree-width and tree decomposition, refer to Graph Theory [Die10] or the original publication of Robertson and Seymour [RS86].

Let (G, H) be a hypergraph pair. A *generalised hypertree decomposition* [Adl08] of (G, H) is a triple (T, χ, λ) where (T, χ) is a tree decomposition of G and $\lambda \colon V(T) \to 2^{E(H)}$ is a mapping such that

(ghd) Every $t \in V(T)$ satisfies $\chi(t) \subseteq \bigcup \lambda(t)$.

The *width* of (T, χ, λ) is defined as

$$\mathrm{w}(T, \chi, \lambda) = \max \Big\{ |\lambda(t)| \;\Big|\; t \in V(T) \Big\}.$$

The *generalised hypertree-width* $\mathrm{ghw}(G, H)$ *of* (G, H) is then the smallest width of a generalised hypertree decomposition of (G, H).

Note that any hypergraph pair (G, H) with an edge $e \in E(H)$ such that $V(G) \subseteq e$ satisfies $\mathrm{ghw}(G, H) = 1$. Moreover, any graph G satisfies $\mathrm{tw}(G) + 1 = \mathrm{ghw}(G, K^1_{V(G)})$ where $\mathrm{tw}(G)$ is the tree-width of G. For a hypergraph H we let $\mathrm{ghw}(H) := \mathrm{ghw}(\underline{H}, H)$, and we obtain the original notion of generalised hypertree-width of a hypergraph, as defined by Gottlob, Leone and Scarcello [GLS03].

If we alter the definition of generalised hypertree decompositions by requiring that T be a path, we obtain the notions of *generalised hyperpath decomposition* and the *generalised hyperpath-width* of (G, H), denoted by $\mathrm{hpw}(G, H)$. This concept is also introduced by Adler [Adl08]. Again, for a hypergraph H we let $\mathrm{hpw}(H) := \mathrm{hpw}(\underline{H}, H)$.

Theorem 76. *Any hypergraph pair* (G, H) *satisfies*

$$\mathrm{ghw}(G, H) \leq \mathrm{hpw}(G, H) \leq \mathrm{hd}(G, H).$$

Proof. The first inequality follows immediately from the fact that every generalised hyperpath decomposition is a generalised hypertree decomposition.

Towards the second inequality, let (F, B, C) be a decomposition forest of (G, H) of height k. Let $\ell_1, \ldots \ell_r$ be an enumeration of the leaves of the trees in F, i.e.

of the nodes of F with degree 1 that are not roots, together with all nodes of degree 0. We define a path P with vertices $V(P) = \{\ell_1, \dots \ell_r\}$, connecting ℓ_i to ℓ_{i+1} by an edge, for $1 \leq i < r$. For every $1 \leq i \leq r$ we let $\chi(\ell_i) := B(\downarrow \ell_i)$ and $\lambda(\ell_i) := \{C(t) \mid t \in \downarrow \ell_i\}$. It is easy to see that (P, χ, λ) is a generalised hyperpath decomposition of width $\leq k$ of (G, H). □

In general, these inequalities are sharp:

Example 77. *Let $\mathcal{T}$ be the class of all trees. It is easy to see that the generalised hypertree-width of $\mathcal{T}$ is bounded by* 1*, but $\mathcal{T}$ has unbounded generalised hyperpath-width. Similarly, the generalised hyperpath-width of the class $\mathcal{P}$ of all paths is bounded by* 1*, but $\mathcal{P}$ has unbounded hypertree-depth, since the hypertree-depth of a path of length n is $\lfloor \log_2(n+2) \rfloor$. This follows from the definition and properties of vertex hyperranking by induction on the length of the path.*

None of our hypergraph pair and hypergraph invariants can increase when taking minors, as shown by the following two lemmas.

Lemma 78. *Let $(G', H') \preccurlyeq (G, H)$. Then* $\mathrm{hd}(G', H') \leq \mathrm{hd}(G, H)$*,* $\mathrm{hpw}(G', H') \leq \mathrm{hpw}(G, H)$*, and* $\mathrm{ghw}(G', H') \leq \mathrm{ghw}(G, H)$*.*

Proof. Let μ be a minor map from (G', H') to (G, H) and for $X \subseteq V(G)$, let $\eta(X) = \{v \in V(G') \mid \mu(v) \cap X \neq \emptyset\}$. Observe that if $\eta(X) \neq \emptyset$ and there is a hyperedge $e \in E(H)$ with $X \subseteq e$, then there exists $e' \in E(H')$ with $\eta(X) \subseteq e'$. Generally, if $\eta(X) \neq \emptyset$ and some k hyperedges of H cover X, then there are at most k hyperedges of H' covering $\eta(X)$.

We first show the inequalities for ghw and hpw. We assume that $\mathrm{ghw}(G, H) < \infty$ ($\mathrm{hpw}(G, H) < \infty$, respectively). Let (T, χ, λ) be a generalised hypertree decomposition of (G, H) with width $\mathrm{ghw}(G, H)$. Let $T' = T$ and for $t \in T$, let $\chi'(t) = \eta(\chi(t))$. Now (T', χ') is a tree decomposition of G': For $v' \in V(G')$, let S be all the nodes t' of T' with $\chi'(t') \cap \mu(v') \neq \emptyset$. Since $\mu(v')$ is connected in G, it follows from the properties of a tree decomposition that S is connected in T.

Let S' be the nodes of T' containing v'. From definition, we have $S' = S$ and the occurrences of v' in T form a subtree. The other conditions for tree decompositions are satisfied trivially. We let $\lambda'(t)$ be the hyperedges covering $\eta(\lambda(t))$ from the observation above, satisfying **(ghd)** for (T', χ', λ').

Now we have that (T', χ', λ') is a hypertree decomposition of (G', H') showing $\mathrm{ghw}(G', H') \leq \mathrm{ghw}(G, H)$. This also proves the inequality for hpw, since if T is a path, T' is a path as well.

Finally, we show that $\mathrm{hd}(G', H') \leq \mathrm{hd}(G, H)$. Assume that $\mathrm{hd}(G, H) < \infty$. Let (F, B, C) be a decomposition forest of (G, H) with depth $\mathrm{hd}(G, H)$. Let $F' = F$ and for $t \in V(F')$, define $C'(t)$ to be the hyperedge covering $\eta(C(t))$. Define B' recursively on F' (from the roots up) as $B'(t) = \eta(B(t)) \setminus \bigcup_{s<t} B'(s)$.

Now (F', B', C') trivially satisfies **(pdec0)** and **(pdec1)**. For **(pdec2)**, let $e' = (x', y')$ be an edge of G', $e = \mu(e') \in E(G)$ and take $s, t \in V(F)$ as in **(pdec2)** for (G, H) and e. From definition of B', we have $x' \in \bigcup B'(\downarrow s)$ and $y' \in \bigcup B'(\downarrow t)$ or vice-versa.

For **(pdec3)**, suppose that there are $s, t \in F$ with $v' \in B'(s) \cap B'(t) \neq \emptyset$. From definition of B', s and t are incomparable in F. In (G, H), $B(s)$ and $B(t)$

are separated by $S = \bigcup B(\downarrow (s \wedge t))$. But $\mu(v)$ is connected, intersects both $B(s)$ and $B(t)$ and avoids S – a contradiction with Lemma 62.

Therefore, (F', B', C') is is a decomposition forest of (G', H') showing that $\mathrm{hd}(G', H') \leq \mathrm{hd}(G, H)$. □

Lemma 79. *Let $H' \preccurlyeq H$. Then $\mathrm{hd}(H') \leq \mathrm{hd}(H)$, $\mathrm{hpw}(H') \leq \mathrm{hpw}(H)$, and $\mathrm{ghw}(H') \leq \mathrm{ghw}(H)$.*

Proof. This follows directly from Lemma 78 on $(\underline{H}, H)$. □

4.7 Remarks and open problems

We introduced the notion of hypertree-depth of hypergraphs and hypergraph pairs and we characterised it using the ultramonotone robber and marshals game, and using vertex-hyperrankings and centred hypercolourings. We believe that hypertree-depth is a very natural notion with potential applications in other areas, such as databases or constraint satisfaction.

Moreover, we introduced minors in hypergraphs and hypergraph pairs and studied their properties. In particular, neither generalised hypertree-width, nor generalised hyperpath-width, nor hypertree-depth increase under taking hypegraph minors.

Within the new framework of minors in hypergraph pairs, many open questions arise. Since testing for minors in hypergraph pairs is NP-hard, a natural question to ask is, for which classes of hypergraph pairs the problem is in PTIME. (For example, this is obviously the case for pairs as in Observation 72.)

A similar question arises for the minor ordering $\preccurlyeq$ in hypergraph pairs (cf. [FHR09]): Which classes of hypergraph pairs have no infinite $\preccurlyeq$-antichains?

Let $\mathcal{C}$ be a class of graphs closed under taking minors, and let O be the obstruction set for $\mathcal{C}$ (i.e. O is the set of $\preccurlyeq$-minimal graphs that are not in $\mathcal{C}$). Robertson and Seymour show [RS04] that O is always *finite*. Similarly, for a class $\mathcal{C}'$ of hypergraph pairs that is closed under taking minors, we let the *obstruction set* of $\mathcal{C}'$ be the set of all $\preccurlyeq$-minimal hypergraph pairs that are not in $\mathcal{C}'$.

Example 80. *Let $\mathcal{C}$ be a class of graphs closed under taking minors with obstruction set O. Then the class of all pairs (G, H) with $G \in \mathcal{C}$ and H any hypergraph satisfying $V(G) = V(H)$ is closed under taking minors, and it has a* finite *obstruction set $\{(G, H) \mid G \in O, H$ hypergraph with $V(H) = V(G)\}$.*

Which classes $\mathcal{C}'$ of hypergraph pairs that are closed under taking minors have a *finite* obstruction set? And in particular, what is the obstruction set for the hypergraph pairs of generalised hypertree-width (hyperpath-width, hypertree-depth) at most 1? Would a weaker notion of a minor relation (e.g. by allowing more minor operations) have some finiteness property while preserving game monotonicity?

Notation

General notation

$\mathbb{R}, \mathbb{R}^+, \mathbb{N}$	real, positive real and natural numbers	*6*
$\mathcal{O}(f)$	functions asymptotically at most f	*6*
$\text{dist}(u, v)$	shortest-path distance (number of edges)	*7*
$\text{clos}(X)$	topological set closure	*6*
$\text{int}(X)$	topological set interior	*6*
$G\restriction_S$	graph restriction to a subset or a region	*34*
$\mathcal{X}$	ground set for an intersection graph	*10*
$\mathcal{I}$	set of possible representatives, $\mathcal{I} \subseteq 2^{\mathcal{X}}$	*10*
$\varphi(v)$	geometric object representing v	*10*
$\varphi[S]$	$\bigcup_{v \in S} \varphi(v)$	*11*
$\varphi^{-1}(X)$	vertices v with $\varphi(v) \cap X \neq \emptyset$	*11*
P	class of polinomially solvable problems	
NP	class of non-deterministically polynomial problems	
PSPACE	problems solvable in polynomial space	
EXPTIME	problems solvable in exponential time	
XP	problems solvable in time $O(n^k)$ with (fixed) parameter k	
FPT	problems solvable in time $\mathcal{O}(f(k)n^c)$ with (fixed) parameter k	
W[k]	parametrized complexity hardness classes related to FPT	
INT	interval graphs	*11*
CHORDAL	chordal graphs	*12*
$\text{cn}(G)$	cop-number of graph G	*20*
$\text{max-cn}(\mathcal{G})$	maximal cop-number among connected graphs of $\mathcal{G}$	*20*

Chapter 2 specific notation

IFA	interval filament graphs	*13*
CIRCLE	circle graphs	*12*
FUN	function graphs	*12*
CIRCARC	circular arc graphs	*12*
STRING	string graphs	*14*
OUTER-STRING	outer-string graphs	*14*
GENUS-k-STRING	string graphs on genus k surface	*14*
PLANAR	planar graphs	*10*
GENUS-k	graphs on genus k surface	*10*
BOXICITY-k	graphs of boxicity k	*41*

Chapter 3 specific notation

$\mathcal{A}$	attack family in $\mathcal{A}$-defensive domination	*46*
$\mathcal{B}$	set of bases in the ∞-fast robber game	*52*
$\mathcal{C}(C, r)$	game state on cops' turn	*52*
$\mathcal{R}(C, A)$	game state on robber's turn	*50*
$\mathcal{S}$	cops' strategy	*8*
$\mathcal{WIN}$	game state won for the cops	*51*
$\widetilde{\varphi}(i, j), \widetilde{\varphi}(X)$	intervals contained in (i, j) or in a set X	*11*
$\mathrm{L}(u)$	left interval endpoint	*12*
$\mathrm{R}(u)$	right interval endpoint	*12*
$(i, j,)$	open interval l to r	*51*
$\xi(B)$	cover of a base	*51*
$\widetilde{\mathcal{R}}((l_1, r_1), \dots)$	restricted game state on robber's turn	*52*
$\widetilde{\mathcal{C}}(l, r)$	restricted game state on cops' turn	*52*
$\mathrm{canon}(l, r)$	canonical state $\mathcal{C}(\varphi^{-1}(l) \cup \varphi^{-1}(r), \widetilde{\varphi}(l, r))$	*51*
$<_R, \leq_R, \dots$	intervals ordered by right endpoints	*12*
$<_L, \leq_L, \dots$	intervals ordered by left endpoints	*12*
OPT	an optimal solution to a problem	*49*
ALG	the solution found by the examined algorithm	*47*

Chapter 4 specific notation

clos	rooted forest closure	*63*
$H_1 \preccurlyeq H_2$	minor relation	*64*
td	tree-depth	*63*
hd	hypertree-depth	*65*
hpw	hyperpath-width	*78*
ghw	generalised hypertree-width	*78*

Bibliography

[AB09] Sanjeev Arora and Boaz Barak. *Computational complexity: a modern approach*. Cambridge University Press, 2009.

[Adl04] Isolde Adler. Marshals, monotone marshals, and hypertree-width. *Journal of Graph Theory*, 47(4):275–296, 2004.

[Adl06] Isolde Adler. *Width functions for hypertree decompositions*. PhD thesis, 2006. PhD Thesis, , 2006.

[Adl08] Isolde Adler. Tree-related widths of graphs and hypergraphs. *SIAM J. Discrete Math.*, 22(1):102–123, 2008.

[AF84] M. Aigner and M. Fromme. Game of cops and robbers. *Discrete Appl. Math.*, 8(1):1–12, 1984.

[AGK12] Isolde Adler, Tomáš Gavenčiak, and Tereza Klimošová. Hypertree-depth and minors in hypergraphs. *Theor. Comput. Sci.*, 463:84–95, 2012.

[Als04] B. Alspach. Searching and sweeping graphs: A brief survey. *Le Matematiche*, 59:5–37, 2004.

[AM11] Noga Alon and Abbas Mehrabian. On a generalization of meyniel's conjecture on the cops and robbers game. *Electr. J. Comb.*, 18(1), 2011.

[ANW07] Michael H. Albert, Richard J. Nowakowski, and David Wolfe. *Lessons in Play: An Introduction to Combinatorial Game Theory*. AK Peters, USA, 2007.

[B+83] Manfred Burggraf et al. Scotland yard. Board game, 1983.

[BB13] William Baird and Anthony Bonato. Meyniel's conjecture on the cop number: a survey. *arXiv preprint arXiv:1308.3385*, 2013.

[BCP10] Anthony Bonato, Ehsan Chiniforooshan, and Pawel Pralat. Cops and robbers from a distance. *Theor. Comput. Sci.*, 411(43):3834–3844, 2010.

[BGHK09] A. Bonato, P. Golovach, G. Hahn, and J. Kratochvíl. The capture time of a graph. *Discrete Mathematics*, 309(18):5588–5595, 2009.

[BL76] Kellogg S. Booth and George S. Lueker. Testing for the consecutive ones property, interval graphs, and graph planarity using PQ-tree algorithms. *J. Comput. Syst. Sci.*, 13:335–379, 1976.

[BN11] A. Bonato and R. J. Nowakowski. *The Game of Cops and Robbers on Graphs*. American Mathematical Society, 2011.

[Bol98] Béla Bollobás. *Modern graph theory*. Graduate Texts in Mathematics 184. Springer New York., 1998.

[Bou85] A. Bouchet. A polynomial algorithm for recognizing circle graphs. pages 569–572, 1985.

[CCNV11] Jérémie Chalopin, Victor Chepoi, Nicolas Nisse, and Yann Vaxès. Cop and robber games when the robber can hide and ride. *SIAM J. Discrete Math.*, 25(1):333–359, 2011.

[DDTY13] Dariusz Dereniowski, Danny Dyer, Ryan M. Tifenbach, and Boting Yang. Zero-visibility cops and robber game on a graph. In *Frontiers in Algorithmics and Algorithmic Aspects in Information and Management*, volume 7924 of *LNCS*, pages 175–186. Springer Berlin Heidelberg, 2013.

[DGK15] Dariusz Dereniowski, Tomáš Gavenčiak, and Jan Kratochvíl. Cops, a fast robber and defensive domination on interval graphs. Submitted to JGT, 2015.

[Die10] Reinhard Diestel. *Graph Theory*. Springer Berlin., fourth edition, 2010.

[EJ13] Louis Esperet and Gwenaël Joret. Boxicity of graphs on surfaces. *Graphs and Combinatorics*, 29(3):417–427, 2013.

[FGK+10] F. V. Fomin, P. A. Golovach, J. Kratochvíl, N. Nisse, and K. Suchan. Pursuing a fast robber on a graph. *Theor. Comput. Sci.*, 411(7–9):1167–1181, 2010.

[FHR09] Michael R. Fellows, Danny Hermelin, and Frances A. Rosamond. Well-quasi-orders in subclasses of bounded treewidth graphs. In *IWPEC*, pages 149–160, 2009.

[FKL12] Alan M. Frieze, Michael Krivelevich, and Po-Shen Loh. Variations on cops and robbers. *Journal of Graph Theory*, 69(4):383–402, 2012.

[FKMP95] Michael R. Fellows, Jan Kratochvíl, Martin Middendorf, and Frank Pfeiffer. The complexity of induced minors and related problems. *Algorithmica*, 13(3):266–282, 1995.

[FP04] A.M. Farley and A. Proskurowski. Defensive domination. *Congressus Numerantium*, 168:97–107, 2004.

[Fra87] Peter Frankl. Cops and robbers in graphs with large girth and cayley graphs. *Discrete Applied Mathematics*, 17(3):301–305, 1987.

[FT91] D. Fudenberg and J. Tirole. *Game Theory*. MIT Press, 1991.

[FT08] Fedor V. Fomin and Dimitrios M. Thilikos. An annotated bibliography on guaranteed graph searching. *Theor. Comput. Sci.*, 399(3):236–245, 2008.

[Gav74] Fanica Gavril. The intersection graphs of subtrees in trees are exactly the chordal graphs. *J. Comb. Theory, Ser. B*, 16:47–56, 1974.

[Gav00] Fanica Gavril. Maximum weight independent sets and cliques in intersection graphs of filaments. *Information Processing Letters*, 73(5–6):181–188, 2000.

[Gav07] T. Gavenčiak. Cop-win graphs with maximum capture-time. Master's thesis, Charles University, Prague, Czech Republic, 2007.

[Gav10] Tomáš Gavenčiak. Cop-win graphs with maximum capture-time. *Discrete Mathematics*, 310(10–11):1557–1563, 2010.

[Gav11] Tomáš Gavenčiak. Catching a fast robber on interval graphs. In *Proceedings of the 8th annual conference on Theory and applications of models of computation*, TAMC'11, pages 353–364, Berlin, Heidelberg, 2011. Springer-Verlag.

[GGJ+15] T. Gavenčiak, P. Gordinowicz, V. Jelínek, P. Klavík, and J. Kratochvíl. Cops and robbers on string graphs. In *ISAAC 2015 (accepted)*, 2015.

[GHK+09] Robert Ganian, Petr Hlinený, Joachim Kneis, Alexander Langer, Jan Obdrzálek, and Peter Rossmanith. On digraph width measures in parameterized algorithmics. In *IWPEC*, pages 185–197, 2009.

[GJKK13] Tomás Gavenciak, Vít Jelínek, Pavel Klavík, and Jan Kratochvíl. Cops and robbers on intersection graphs. In *Algorithms and Computation - 24th International Symposium, ISAAC 2013, Hong Kong, China, December 16-18, 2013, Proceedings*, pages 174–184, 2013.

[GLS99] Georg Gottlob, Nicola Leone, and Francesco Scarcello. Hypertree decompositions and tractable queries. In *Proceedings of the eighteenth ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems*, pages 21–32. ACM, 1999.

[GLS03] Georg Gottlob, Nicola Leone, and Francesco Scarcello. Robbers, marshals, and guards: game theoretic and logical characterizations of hypertree width. *Journal of computer and system sciences*, 66(4):775–808, 2003.

[Gol77] M.C. Golumbic. The complexity of comparability graph recognition and coloring. *Computing*, 18:199–208, 1977.

[Gol80] M. C. Golombic. *Algorithmic Graph Theory and Perfect Graphs*. Academic Press, New York, 1980.

[GR95] A.S. Goldstein and E.M. Reingold. The complexity of pursuit on a graph. *Theoretical computer science*, 143(1):93–112, 1995.

[Hah07] Gena Hahn. Cops, robbers and graphs. *Tatra Mt. Math. Publ*, 36(163):163–176, 2007.

[HK08] Paul Hunter and Stephan Kreutzer. Digraph measures: Kelly decompositions, games, and orderings. *Theor. Comput. Sci.*, 399(3):206–219, 2008.

[IKK06] Volkan Isler, Sampath Kannan, and Sanjeev Khanna. Randomized pursuit-evasion with local visibility. *SIAM J. Discrete Math.*, 20(1):26–41, 2006.

[JAV87] JAVOZ Jablonec nad Nisou. Fantom staré prahy. Board game, 1987.

[JRST01] Thor Johnson, Neil Robertson, Paul D Seymour, and Robin Thomas. Directed tree-width. *Journal of Combinatorial Theory, Series B*, 82(1):138–154, 2001.

[KGK86] J. Kratochvíl, M. Goljan, and P. Kučera. *String graphs*. Rozpravy ČSAV.: Řada matem. a přírodních věd. Academia, 1986.

[Kin15] William B. Kinnersley. Cops and robbers is exptime-complete. *J. Comb. Theory, Ser. B*, 111:201–220, 2015.

[KM89] Norbert Korte and Rolf H. Möhring. An incremental linear-time algorithm for recognizing interval graphs. *SIAM J. Comput.*, 18(1):68–81, 1989.

[Kra91] Jan Kratochvíl. String graphs. II. recognizing string graphs is NP-hard. *Journal of Combinatorial Theory, Series B*, 52(1):67–78, 1991.

[LB62] C. Lekkeikerker and J. Boland. Representation of a finite graph by a set of intervals on the real line. *Fundamenta Mathematicae*, 51(1):45–64, 1962.

[Mam13] Marcello Mamino. On the computational complexity of a game of cops and robbers. *Theor. Comput. Sci.*, 477:48–56, 2013.

[McC03] Ross M. McConnell. Linear-time recognition of circular-arc graphs. *Algorithmica*, 37(2):93–147, 2003.

[Meh11a] Abbas Mehrabian. Cops and robber game with a fast robber on interval, chordal, and planar graphs. *CoRR*, abs/1008.4210, 2011.

[Meh11b] Abbas Mehrabian. Lower bounds for the cop number when the robber is fast. *Combinatorics, Probability & Computing*, 20(4):617–621, 2011.

[Meh12] Abbas Mehrabian. Cops and robber game with a fast robber on expander graphs and random graphs. *Annals of Combinatorics*, 16(4):829–846, 2012.

[Mik88] Zoltán Miklós. Understanding tractable decompositions for constraint satisfaction, 1988. PhD Thesis, University of Oxford, 2008.

[MM99] T. A. McKee and F. R. McMorris. *Topics in Intersection Graph Theory.* Society for Industrial Mathematics, 1999.

[MNT88] Jiří Matoušek, Jaroslav Nešetřil, and Robin Thomas. On polynomial time decidability of induced-minor-closed classes. *Comment. Math. Univ. Carolin.*, 29(4):703–710, 1988.

[MT01] B. Mohar and C. Thomassen. *Graphs on Surfaces.* The John Hopkins University Press, 2001.

[Naj85] Walid Naji. Reconnaissance des graphes de cordes. *Discrete Math.*, 54:329–337, 1985.

[NdM06] Jaroslav Nešetřil and Patrice Ossona de Mendez. Tree-depth, subgraph coloring and homomorphism bounds. *Eur. J. Comb.*, 27(6):1022–1041, 2006.

[NdM08a] Jaroslav Nešetřil and Patrice Ossona de Mendez. Grad and classes with bounded expansion I. Decompositions. *Eur. J. Comb.*, 29(3):760–776, 2008.

[NdM08b] Jaroslav Nešetřil and Patrice Ossona de Mendez. Grad and classes with bounded expansion II. Algorithmic aspects. *Eur. J. Comb.*, 29(3):777–791, 2008.

[NdM08c] Jaroslav Nešetřil and Patrice Ossona de Mendez. Grad and classes with bounded expansion III. Restricted graph homomorphism dualities. *Eur. J. Comb.*, 29(4):1012–1024, 2008.

[NdM12] Jaroslav Nesetril and Patrice Ossona de Mendez. *Sparsity – Graphs, Structures, and Algorithms*, volume 28 of *Algorithms and combinatorics.* Springer, 2012.

[NS08] N. Nisse and K. Suchan. Fast robber in planar graphs. In *Proceedings of the 34th International Workshop on Graph-Theoretic Concepts in Computer Science*, volume 5344 of *Lecture Notes in Computer Science*, pages 33–44, 2008.

[NW83] R. Nowakowski and P. Winkler. Vertex to vertex pursuit in a graph. *Discrete Math.*, 43(2):235–239, 1983.

[Obd06] Jan Obdržálek. Dag-width: connectivity measure for directed graphs. In *SODA '06: Proceedings of the seventeenth annual ACM-SIAM symposium on Discrete algorithm*, pages 814–821, New York, NY, USA, 2006. ACM.

[Pap94] C.H. Papadimitriou. *Computational Complexity.* Theoretical computer science. Addison-Wesley, 1994.

[Par76] T. D. Parsons. Pursuit-evasion in a graph. pages 426–441, 1976.

[Par78] T. D. Parsons. The search number of a connected graph. In *Proc. 9th Southeastern Conf. Combinatorics, Graph Theory, and Computing*, pages 549–554, 1978.

[Per07] Martin Pergel. Recognition of polygon-circle graphs and graphs of interval filaments is NP-complete. In *Graph-Theoretic Concepts in Computer Science*, volume 4769 of *Lecture Notes in Computer Science*, pages 238–247. Springer Berlin Heidelberg, 2007.

[Pra] V.V. Prasolov. *Elements of Combinatorial and Differential Topology*. Graduate studies in mathematics. American Mathematical Soc.

[Qui78] A. Quilliot. Jeux et points fixes sur les graphes. Master's thesis, 1978.

[Qui83] A. Quilliot. *Problèmes de jeux, de point fixe, de connectivité et de représentation sur des graphes, des ensembles ordonnés et des hypergraphes*. PhD thesis, 1983.

[Qui85] A. Quilliot. A short note about pursuit games played on a graph with a given genus. *Journal of combinatorial theory. Series B*, 38(1):89–92, 1985.

[RS83] Neil Robertson and Paul D. Seymour. Graph minors. i. excluding a forest. *J. Comb. Theory, Ser. B*, 35(1):39–61, 1983.

[RS86] N. Robertson and P. D. Seymour. Graph minors II: algorithmic aspects of tree-width. *Journal Algorithms*, 7:309–322, 1986.

[RS95] Neil Robertson and Paul D. Seymour. Graph minors. XIII. The disjoint paths problem. *J. Comb. Theory, Ser. B*, 63(1):65–110, 1995.

[RS04] Neil Robertson and Paul D. Seymour. Graph minors. XX. Wagner's conjecture. *J. Comb. Theory, Ser. B*, 92(2):325–357, 2004.

[Sch84] E. R. Scheinerman. *Intersection classes and multiple intersection parameters of graphs*. Doctoral dissertation, 1984.

[Sch01] B. S. W. Schroeder. The copnumber of a graph is bounded by 3/2 genus(g) + 3. Trends Math., Birkhäuser Boston, pages 243–263, 2001.

[Sin66] F. W. Sinden. Topology of thin film RC-circuits. *Bell System Technical Journal*, 45:1639—1662, 1966.

[SS04] Marcus Schaefer and Daniel Stefankovic. Decidability of string graphs. *Journal of Computer and System Sciences*, 68(2):319–334, 2004. Special Issue on STOC 2001.

[SSS03] Marcus Schaefer, Eric Sedgwick, and Daniel Stefankovic. Recognizing string graphs in NP. *Journal of Computer and System Sciences*, 67(2):365–380, 2003. Special Issue on STOC 2002.

[Tho86] Carsten Thomassen. Interval representations of planar graphs. *Journal of Combinatorial Theory, Series B*, 40(1):9–20, 1986.

[Tos85] R. Tosic. Inductive classes of graphs. In *Proceedings of the Sixth Yugoslav Seminar on Graph Theory*, pages 233–237. University of Novi Sad, 1985.