



Monotonicity of the Cops and Robber Game for Bounded Depth Treewidth

Isolde Adler  

University of Bamberg, Germany

Eva Fluck  

RWTH Aachen University, Germany

Abstract

We study a variation of the cops and robber game characterising treewidth, where in each round at most one cop may be placed and in each play at most q rounds are played, where q is a parameter of the game. We prove that if k cops have a winning strategy in this game, then k cops have a monotone winning strategy. As a corollary we obtain a new characterisation of bounded depth treewidth, and we give a positive answer to an open question by Fluck, Seppelt and Spitzer (2024), thus showing that graph classes of bounded depth treewidth are homomorphism distinguishing closed.

Our proof of monotonicity substantially reorganises a winning strategy by first transforming it into a pre-tree decomposition, which is inspired by decompositions of matroids, and then applying an intricate breadth-first “cleaning up” procedure along the pre-tree decomposition (which may temporarily lose the property of representing a strategy), in order to achieve monotonicity while controlling the number of rounds simultaneously across all branches of the decomposition via a vertex exchange argument. We believe this can be useful in future research.

2012 ACM Subject Classification Mathematics of computing → Graph theory

Keywords and phrases tree decompositions, treewidth, treedepth, cops-and-robber game, monotonicity, homomorphism distinguishing closure

Digital Object Identifier 10.4230/LIPIcs.MFCS.2024.6

Related Version *Full Version:* <https://doi.org/10.48550/arXiv.2402.09139> [4]

1 Introduction

Search games were introduced by Parsons and Petrov in [36, 37, 38] and since then gained much interest in many (applied and theoretical) areas of computer science and in discrete mathematics [6, 10, 9, 28, 17, 35, 25, 15, 24, 21]. In search games on graphs, a fugitive and a set of searchers move on a graph, according to given rules. The searchers’ goal is to capture the fugitive, and the fugitive tries to escape. Here the interest lies in minimising the resources needed to guarantee capture. Typically this means minimising the number of searchers, but we also seek to bound the number of rounds of the game, if the searchers can only move one by one. Search games have proven very useful for providing a deep understanding of structural and algorithmic properties of width parameters of graphs, such as treewidth [8, 43], pathwidth [9], cutwidth [30], directed treewidth [26], treedepth [33], and b -branched treewidth [14, 32].

The crux in relating a given variant of a search game to a width parameter often lies in the question of whether the game is *monotone*, i. e. whether the searchers always have a winning strategy in which a previously cleared area never needs to be searched again – without needing additional resources. Furthermore, monotonicity of a search game provides a polynomial space certificate for proving that determining the winner is in NP.

In their classic paper [43], Seymour and Thomas proved monotonicity of the cops and robber game characterising treewidth. They use a very elegant inductive argument via the dual concept of *brambles*. In this paper we study a variation of this game, where k cops try



© Isolde Adler and Eva Fluck;

licensed under Creative Commons License CC-BY 4.0

49th International Symposium on Mathematical Foundations of Computer Science (MFCS 2024).

Editors: Rastislav Kráľovič and Antonín Kučera; Article No. 6; pp. 6:1–6:18

Leibniz International Proceedings in Informatics



LIPIC Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

to capture a robber, but they are limited to placing at most one cop per round and playing at most q rounds, for a fixed number $q \in \mathbb{N}$. It is an open question from [13], whether this game is monotone. We give a positive answer to this question.

The notion of *treedepth* was first introduced by Nešetřil and Ossona de Mendes [33]. They exhibit a number of equivalent parameters, and a characterisation by a monotone game is implicitly given. This was subsequently made more explicit in [19]. In [18], a characterisation by a different game called *lifo game* is given for which monotonicity is proven. The game we study can be seen as generalising the monotone game implicit in [33]. However, it also captures treewidth and it is not monotone by definition.

Recently, width parameters received a renewed interest in the context of counting homomorphisms and the expressive power of logics [12, 20, 11, 41, 13]. In this context a non-monotone search game characterisation of the width parameter is useful to ensure that there are no graphs of higher width that can be added to the graph class without changing the expressive power of the logic [34, 13]. The main obstacle then is to find such a non-monotone characterisation, as the natural characterisation as a search-game of many graph parameters is inherently monotone. *Bounded depth treewidth* and the game studied in this paper were first defined in [13]. An equivalent characterisation of these graph classes by so-called k -pebble forest covers of depth q , which is bounded width treedepth, was already given in [1].

Homomorphism Counts. Homomorphism counts are an emerging tool to study equivalence relations between graphs. Many equivalence relations between graphs can be characterized as homomorphism indistinguishability relations, these include graph isomorphism [29], graph isomorphism relaxations [31, 22, 40], cospectrality (folklore) and equivalence with respect to first-order logic with counting quantifiers [12, 20, 11, 13]. In order to study the expressiveness of such equivalence relations, it is crucial to know under which circumstances distinct graph classes yield distinct equivalence relations. Towards this question one considers the closure of a graph class under homomorphism indistinguishability. Let \mathcal{F} be a graph class. Two graphs G, H are *homomorphism indistinguishable over \mathcal{F}* , if for all $F \in \mathcal{F}$ the number of homomorphisms from F to H equals the number of homomorphisms from F to G . The graph class \mathcal{F} is *homomorphism distinguishing closed*, if for every graph $F \notin \mathcal{F}$ there exists two graphs G, H , that are homomorphism indistinguishable over \mathcal{F} but that do not have the same number of homomorphisms from F . It has been conjectured by Roberson [39], that all graph classes that are closed under taking minors and disjoint unions are homomorphism distinguishing closed. So far the list of graph classes for which the conjecture is confirmed is short: the class of all planar graphs [39], graph classes that are essentially finite [42], the classes of all graphs of treewidth at most $k - 1$ [34] and the classes of all graphs of treedepth at most q [13]. The latter two results rely on characterisations of the graph classes in terms of non-monotone cops-and-robber games. We study *bounded depth treewidth*, which bounds both the width and the depth simultaneously. We give a game characterisation that does not rely on monotonicity, and as a consequence we obtain that the classes of all graphs of bounded depth treewidth are also homomorphism distinguishing closed.

Our Contribution. We show the following (cf. Theorem 27).

Fix integers $k, q \geq 1$. For every graph G the following are equivalent.

- *G has a tree decomposition of width at most $k - 1$ and depth at most q .*
- *k cops have a monotone winning strategy in the cops and robber game on G with at most q placements.*
- *k cops have a winning strategy in the cops and robber game on G with at most q placements.*

The equivalence between the last two statements gives a positive answer to an open question from [13]. Our proof of monotonicity gives both a proof of monotonicity for the classical cops and robber game characterising treewidth as well as for the game characterising treedepth as special cases, by removing the bound on the number of rounds or respectively the number of cops. As a corollary, we obtain the following (cf. Theorem 18).

Let $k, q \geq 0$ be integers. The class of all graphs having a tree decomposition of width at most $k - 1$ and depth at most q is homomorphism distinguishing closed.

Proof Techniques. In contrast to the proof of monotonicity of the classic cops and robber game [43], our proof does not use a dual concept such as brambles. Instead, we modify a (possibly non-monotone) winning strategy, turning it first into what we call a *pre-tree decomposition*, and then cleaning it up while keeping track of width and depth, thus finally transforming the pre-tree decomposition into a monotone winning strategy. Our concept of pre-tree decomposition is inspired by decompositions of matroids and it is based on ideas from [3, 7]. Our cleaning-up technique is similar to the proof of monotonicity of the game for b -branching treewidth [32]. However, the cleaning-up technique in [32] loses track of the number of cop movements, as local modifications may have non-local effects that are not controlled. We need to keep track in order to control the depth.

This poses a major challenge which we resolve in our proof by a fine grained cleaning-up technique in our pre-tree decomposition based on a careful decision of which vertices to “push up and through the tree” and which to “push down”. The vertices “pushed up” may have an effect on the part of the pre-tree decomposition that was processed in previous steps, which we manage to control by a vertex exchange argument. Additionally we keep track of how the first modification at some node in the pre-tree decomposition relates back to the original strategy. We believe that our techniques will also help in future research.

Our proof provides an independent proof of monotonicity of the classic game characterising treewidth as a special case, namely when q is greater than or equal to the number of vertices of the graph. Our proof strategy is entirely different of the original proof of [43], as it does not use an equivalence via a dual object such as brambles. Instead, we provide a more direct transformation of a (possibly non-monotone) winning strategy.

Further Related Research. Search games are used to model a variety of real-world problems such as searching a lost person in a system of caves [36], clearing contaminated tunnels [28], searching environments in robotics [24], and modelling bugs in distributed environments [17], cf. [16] for a survey.

There is a fine line between games that are monotone and those that are not. For example, the marshalls and robber game played on a hypergraph is a natural generalisation of the cops and robber game, it is related to hypertreewidth, but it is not monotone [2]. However, the monotone and the non-monotone variants are strongly related [5] to each other. In a directed graph setting the games are also not monotone [27].

Structure of the Paper. In Section 2 we fix our notation and we define tree decompositions of bounded depth and width. Section 3 introduces pre-tree decomposition, relevant properties, and establishes a relation to tree decompositions. The game is introduced in Section 4, and in Section 5 we give the main construction, showing how to make a strategy tree exact while maintaining the bounds on width and depth. The insights given by our answer to the open question in the area of homomorphism counts are briefly discussed in Section 6. Due to space restrictions all proofs are deferred to the appendix.

2 Preliminaries

Sets and Partitions. Let A be a finite set. We write 2^A to denote the power-set of A and, for $k \in \mathbb{N}$, $\binom{A}{\leq k}$ to denote all subsets of A of size $\leq k$. $\text{Part}(A)$ is the set of all *ordered partitions* of A , where we allow partitions to contain multiple (but finitely many) copies of the empty set. Let $\pi = \{X_1, \dots, X_d\} \in \text{Part}(A)$ and $F \subseteq A$. For $i \in [d]$, the partition

$$i_{X_i \rightarrow F} := \{X_1 \setminus F, \dots, X_{i-1} \setminus F, X_i \cup F, X_{i+1} \setminus F, \dots, X_d \setminus F\},$$

is called the F -*extension in X_i of π* . A function $w: \text{Part}(A) \rightarrow \mathbb{N}$ is *submodular* if, for all $\pi, \pi' \in \text{Part}(A)$, for all sets $X \in \pi$ and $Y \in \pi'$ with $X \cup Y \neq A$, it holds that

$$w(\pi) + w(\pi') \geq w(\pi_{X \rightarrow \bar{Y}}) + w(\pi'_{Y \rightarrow \bar{X}}).$$

Let $f: A \rightarrow B$ be a function and $C \subseteq A$. By $f|_C$ we denote the restriction of f to C , i.e. $f|_C: C \rightarrow B$ and $f|_C(c) = f(c)$, for all $c \in C$.

Graphs. A graph G is a tuple $(V(G), E(G))$, where $V(G)$ is a finite set of vertices and $E(G) \subseteq \binom{V(G)}{\leq 2}$ is the set of edges. We usually write uv or vu to denote the edge $\{u, v\} \in E(G)$. We write $\overrightarrow{E(G)}$, when we orient the edges of G , that is $\overrightarrow{E(G)} := \{(u, v), (v, u) \mid uv \in E(G)\}$, and call $(u, v) \in \overrightarrow{E(G)}$ an arc. If G is clear from the context we write V, E instead of $V(G), E(G)$. By G° we denote the graph obtained from G by adding all self-loops that are not present in G , that is $V(G^\circ) := V(G)$ and $E(G^\circ) := E(G) \cup \{vv \mid v \in V(G)\}$. For $U \subseteq V$ we write $G[U]$ to denote the subgraph of G induced by U . For $v \in V$ we write $E_G(v) := \{uv \mid uv \in E(G)\}$ for the edges incident to v and $N_G(v) := \{u \mid uv \in E(G)\}$ for its neighbours.

A *tree* is a graph where any two vertices are connected by exactly one path. A *rooted tree* (T, r) is a tree T together with some designated vertex $r \in V(T)$, the *root* of T . At times, the following alternative definition is more convenient. We can view a rooted tree (T, r) as a pair $(V(T), \preceq)$, where \preceq is a partial order on $V(T)$ and for every $v \in V(T)$ the elements of the set $\{u \in V(T) \mid u \preceq v\}$ are pairwise comparable: The minimal element of \preceq is precisely the root of T , and we let $v \preceq w$ if v is on the unique path from r to w . Let $t, t' \in V(T)$, we call $t^* \in V(T)$ the *greatest common ancestor* if $t^* \preceq t, t'$ but for all $t'' \in V(T)$ with $t^* \prec t''$ either $t'' \not\preceq t$ or $t'' \not\preceq t'$. By $L(T)$ we denote the set of all *leaves* of T , that is $L(T) := \{\ell \in V(T) \mid \text{there is no } s \in V(T) \text{ such that } \ell \prec s\}$ is the set of all maximal elements of \preceq . All vertices that are not leaves are called *inner vertices*.

► **Definition 1.** Let G be a graph, let (T, r) be a rooted tree and let $\beta: V(T) \rightarrow 2^{V(G)}$ be a function from the nodes of T to sets of vertices of G . We call (T, r, β) a *tree decomposition* of G , if

(T1) $\bigcup_{t \in V(T)} G[\beta(t)] = G$, and

(T2) for every vertex $v \in G$, the graph $T_v := T[\{t \in V(T) \mid v \in \beta(t)\}]$ is connected.

The sets $\beta(t)$ are called the *bags* of this tree decomposition.

The *width* of a tree decomposition (T, r, β) is $\text{wd}(T, r, \beta) := \max_{t \in V(T)} |\beta(t)| - 1$, the *depth* is $\text{dp}(T, r, \beta) := \max_{\ell \in L(T)} |\bigcup_{t \preceq \ell} \beta(t)|$. The *treewidth* of a graph G is the minimum width of any tree decomposition of G , the *treedepth* of a graph G is the minimum depth of any tree decomposition (see [13]). For $k, q \geq 1$ we define the class \mathcal{T}_q^k to be all graphs that have a tree decomposition (T, r, β) with $\text{wd}(T, r, \beta) \leq k - 1$ and $\text{dp}(T, r, \beta) \leq q$. The following lemma is a well known consequence from (T2).

► **Lemma 2.** Let G be a graph and $U \subseteq V(G)$ connected in G . Let (T, r, β) be a tree decomposition of G , then $T_U := T[\{t \in V(T) \mid U \cap \beta(t) \neq \emptyset\}]$ is connected.

3 Pre-Tree Decomposition, Exactness and Submodularity

Here we consider a definition of tree decompositions that is inspired by matroid tree decompositions [23]. We relax this definition into what we call a pre-tree decomposition.

► **Definition 3.** Let $G = (V(G), E(G))$ be a graph. Let $X \subseteq E(G)$. We define the boundary of a set of edges $\delta(X) := \{v \in V(G) \mid \exists e \in X, e' \in E(G) \setminus X, v \in e \cap e'\}$. Let π be a partition of $E(G)$. We define the boundary of a partition

$$\delta(\pi) := \bigcup_{X \in \pi} \delta(X).$$

A tuple (T, r, β, γ) , where (T, r) is a (rooted) tree, $\beta: V(T) \rightarrow 2^{V(G)}$ and $\gamma: \overrightarrow{E(T)} \rightarrow 2^{E(G)}$, is a (rooted) pre-tree decomposition if:

(PT1) $\beta(r) = \emptyset$ and for every connected component C of G , there is a child c of the root with $\gamma(r, c) = E(C)$.

(PT2) For every leaf $\ell \in L(T)$ with neighbour t , it holds that $|\gamma(t, \ell)| \leq 1$.

(PT3) For every $t \in V(T)$, the tuple π_t is a partition of $E(G)$ and $\delta(\pi_t) \subseteq \beta(t)$.

For every internal node $t \in V(T) \setminus L(T)$, we define $\pi_t := (\gamma(t, t_1), \dots, \gamma(t, t_d))$, where $N(t) = \{t_1, \dots, t_d\}$ is an arbitrary enumeration of the neighbours of t . For a leaf $\ell \in L(T)$ with parent p we define $\pi_\ell := (\gamma(\ell, p), \overline{\gamma(\ell, p)})$.

(PT4) For every edge $st \in E(T)$, it holds that $\gamma(s, t) \cap \gamma(t, s) = \emptyset$.

We call an edge $st \in E(T)$ exact if $\gamma(s, t) \cup \gamma(t, s) = E(G)$, we call (T, r, γ, β) exact, if every edge is exact and $\beta(t) = \delta(\pi_t)$, for all $t \in V(T)$. We call $\beta(t)$ the bag at node t and $\gamma(s, t)$ the cone at arc (s, t) .

The reader may note that the boundary of a set of edges is symmetric, that is for all $X \subseteq E(G)$ it holds that $\delta(X) = \delta(E(G) \setminus X)$. Furthermore it holds that $v \in \delta(X)$ if and only if $\emptyset \neq E_G(v) \cap X \neq E_G(v)$, for all $v \in V(G)$. The function γ describes a partition of the edges of the graph at every inner node, whereas the function β gives a vertex separator for this partition. This separator may contain more vertices than necessary at a certain node, which is needed to define the depth of a pre-tree decomposition, as seen below. For some edge $st \in E(T)$ with $s \prec t$, we can view $\gamma(s, t)$ as the set of edges that need to be decomposed in the subtree below and $\gamma(t, s)$ as the set of edges that is for sure decomposed somewhere else within the tree. With this point of view the axioms correspond to the following ideas:

(PT1) We start by separating the different connected components of the graph and assign one distinct subtree to decompose each component. This way we also ensure that all of the graph is decomposed in some subtree.

(PT2) We want to decompose the graph into single edges. We do allow empty leafs and even empty subtrees, to ease our cleaning up procedure in the following sections. The reader may recall that the root of a rooted tree is never a leaf, by definition.

(PT3) At every node of the tree we make sure that all edges of the original graph are accounted for and that β is indeed a separator for the given partition.

(PT4) If a parent node assigns an edge to the set that still has to be decomposed, the child node can not assign this edge to the set that is already decomposed somewhere else. But the other direction is possible, if the parent node assigns an edge to the set that is decomposed somewhere else, the child can still assign it to one of its subtrees. If the latter is also not the case the edge is exact.

Similar to the definition of width and depth for tree decompositions we define the width and depth of a pre-tree decomposition. We slightly adapt the definition of depth as (T2) does not hold in pre-tree decompositions.

► **Definition 4.** The width of a partition π of the edges of a graph is

$$\text{wd}(\pi) := |\delta(\pi)|.$$

The width of a pre-tree decomposition is

$$\text{wd}(T, r, \beta, \gamma) := \max_{t \in V(T)} |\beta(t)| - 1.$$

The depth of a rooted pre-tree decomposition is

$$\text{dp}(T, r, \beta, \gamma) := \max_{t \in V(T)} \sum_{r \prec s \preceq t} |\beta(s) \setminus \beta(p_s)|,$$

where p_s is the parent of s .

The reader may note that the width of a pre-tree decomposition only gets smaller if one sets $\beta(t) := \delta(\pi_t)$, for all nodes $t \in V(T)$, but the depth can get larger. We show that the width of a partition of the edges as defined above is submodular. We need this property to show that our main construction does not enlarge the width of the pre-tree decomposition.

► **Lemma 5** ([7]). For every graph G , wd is submodular.

We continue this section with some lemmas, that help us to get comfortable with the definition of a pre-tree decomposition and are useful to prove that our cleaning up procedure in the following sections is correct. We start with a lemma about the cones along a path of exact edges. It is a direct consequence of exactness and the fact that the cones incident to a node form a partition of the edges.

► **Lemma 6.** Let (T, r, β, γ) be a pre-tree decomposition of a graph G . Let $P = t_1, \dots, t_\ell$ be a path in T , such that every edge $t_i t_{i+1}$, for $i \in [\ell - 1]$, is exact. Then it holds that $\gamma(t_1, t_2) \supseteq \gamma(t_2, t_3) \supseteq \dots \supseteq \gamma(t_{\ell-1}, t_\ell)$.

The following lemma shows, that (PT3) spreads over exact edges, that is any subtree of T that only contains exact edges induces a partition of the edges of the original graph. It is again a direct consequence of exactness and the partitions at the nodes together with the previous lemma.

► **Lemma 7.** Let (T, r, β, γ) be a pre-tree decomposition of a graph G and let (T', r') be a subtree of (T, r) , where r' is the minimal node of T' with respect to \preceq , such that all edges of T' are exact. We pick arbitrary enumerations of $N_T(V(T')) := \{t_1, \dots, t_a\}$ and of $L(T) \cap L(T') := \{\ell_1, \dots, \ell_b\}$. We define $U := \{t_1, \dots, t_a, \ell_1, \dots, \ell_b\}$ and define $s: U \rightarrow V(T')$ to be the natural mapping to the corresponding neighbour in $V(T')$. Then $(\gamma(s(t_1), t_1), \dots, \gamma(s(t_a), t_a), \gamma(s(\ell_1), \ell_1), \dots, \gamma(s(\ell_b), \ell_b)))$ is an ordered partition of $E(G)$.

The next lemma is needed to prove how one can translate exact pre-tree decompositions into tree-decompositions. Furthermore it will help us bound the depth within our cleaning up procedure in the following sections. The Lemma follows from the combination of Lemma 6 with the fact that the boundary of the partition is the union of the boundaries of the cones.

► **Lemma 8.** Let (T, r, β, γ) be a pre-tree decomposition of a graph G and let (T', r') be a subtree of (T, r) , where r' is the minimal node of T' with respect to \preceq , such that all edges of T' are exact. It holds that the induced subgraph $T'_v := T[t \in V(T') \mid v \in \delta(\pi_t)]$, for every vertex $v \in V(G)$, is connected. In particular, if $r = r'$, for every $t \in V(T')$ it holds that

$$\sum_{\substack{s \preceq t \\ s \neq r}} |\delta(\pi_s) \setminus \delta(\pi_{p_s})| = \left| \bigcup_{s \preceq t} \delta(\pi_s) \right|,$$

where p_s is the parent of s .

We conclude this section with a lemma that shows that a pre-tree decomposition of a graph G is indeed a relaxation of a tree-decomposition of G . If every edge is exact and all bags are exactly the boundary of the partition then we can construct a tree decomposition. We need to start with a pre-tree decomposition of the graph G° with all self-loops added to ensure that every non-isolated vertex does appear in some bag and that the components corresponding to isolated vertices are covered by the pre-tree decomposition. If we drop the cones from the tuple we get a tree decomposition by Lemmas 7 and 8. On the other hand we can transform a tree-decomposition into a pre-tree decomposition, by copying the tree-decomposition of each connected component of G and adding leaves that correspond to the edges of G° .

► **Lemma 9.** *Let $k, q \geq 1$. Let $G = (V, E)$ be a graph. Any tree-decomposition of G of width $\leq k - 1$ and depth $\leq q$ gives rise to an exact pre-tree decomposition of G° of width $\leq k - 1$ and depth $\leq q$ and vice versa.*

Proof. Let (T, r, β, γ) be an exact pre-tree decomposition of G° of width $\leq k - 1$ and depth $\leq q$. We define $\beta': V(T) \rightarrow 2^{V(G)}$ as follows

$$\beta'(t) := \begin{cases} \{v\} & \text{if } t \in L(T) \text{ and } r \text{ is parent of } t \text{ and } \gamma(r, t) = \{vv\}, \\ \beta(t) & \text{otherwise.} \end{cases}$$

▷ **Claim 10.** (T, β') is a tree-decomposition of width $\leq k - 1$ and depth $\leq q$.

Proof. From (PT1), (PT2) and Lemma 7 applied to the complete tree (T, r) we get that for every edge $uv \in E(G^\circ)$ there is some leaf ℓ with parent p and $\gamma(p, \ell) = \{uv\}$. Thus if $u = v$, then $\beta'(\ell) = \{v\}$ and thus $vv \in E(G[\beta'(\ell)])$. Otherwise it holds that $uu, vv \in E(G^\circ) \setminus \{uv\}$ and thus $u, v \in \beta'(\ell)$ and $uv \in E(G[\beta'(\ell)])$. All in all we get that (T1) holds.

By Lemma 8 applied to the complete tree (T, r) we know that all T_v are connected. Therefore (T2) also holds and (T, β') is a tree-decomposition.

The width and depth are obvious as $k, q \geq 1$. ◁

Now let (T, r, β) be a tree-decomposition of G of width $\leq k - 1$ and depth $\leq q$. W.l.o.g. β is *tight*, that is for all $t \in V(T)$ and $v \in \beta(t)$, that (T, r, β') , where $\beta'(t) := \beta(t) \setminus \{v\}$ and $\beta'(s) = \beta(s)$, for all $s \in V(T) \setminus \{t\}$, is not a tree-decomposition of G . We construct a new tree T' with root r' and functions $\beta': V(T') \rightarrow 2^{V(G)}$, $\gamma: \overline{E(T')} \rightarrow 2^{E(G^\circ)}$ and $f: V(T') \setminus (L(T') \cup \{r'\}) \rightarrow V(T)$ as follows. Let C be a connected component of G and let $V_C := \{t \in V(T) \mid V(C) \cap \beta(t) \neq \emptyset\}$. By Lemma 2 V_C is connected. If C contains only an isolated vertex v , then $V_C = \{t\}$, for some $t \in V(T)$. We add a new node t_v to T' and connect it to the root. We set $\beta'(t_v) = \emptyset$, $\gamma(r', t_v) = \{vv\}$ and $\gamma(t_v, r') = E(G^\circ) \setminus \{vv\}$. Otherwise let T_C be a copy of the subtree induced by V_C with root r_C and vertices V_C^* and $f|_{V_C^*}: V_C^* \rightarrow V_C$ the natural bijection between the copies and their originals. We attach r_C to the root r' . For every $v \in V(C)$, there is some $t_v \in V_C$ such that $v \in \beta(t_v)$, as C is not an isolated vertex. We add a new leaf t'_v that we attach to $f|_{V(T_C)}^{-1}(t_v)$ and set $\beta'(t'_v) = \{v\}$, $\gamma(f|_{V_C^*}^{-1}(t_v), t'_v) = \{vv\}$ and $\gamma(t'_v, f|_{V_C^*}^{-1}(t_v)) = E(G^\circ) \setminus \{vv\}$. For every $e \in E_G(C)$ there is some $t_e \in V_C$ such that $e \subseteq \beta(t_e)$. We add a new leaf t'_e that we attach to $f|_{V_C^*}^{-1}(t_e)$ and set $\beta'(t'_e) = e$, $\gamma(f|_{V_C^*}^{-1}(t_e), t'_e) = \{e\}$ and $\gamma(t'_e, f|_{V_C^*}^{-1}(t_e)) = E(G^\circ) \setminus \{e\}$. For every node $t \in V_C^*$ with parent p we add all edges $e \in E_G(C)$, where t'_e is a descendant of t , and all self-loops $vv \in E_{G^\circ}(C)$, where t'_v is a descendant of t , to $\gamma(p, t)$. Furthermore we set $\gamma(t, p) := E(G^\circ) \setminus \gamma(p, t)$ and $\beta'(t) := \delta(\pi_t) \subseteq \beta(f(t)) \cap V(C)$. By tightness of β there is some

$v \in \beta(f(\ell))$ such that $T_v = \{f(\ell)\}$, for every $\ell \in L(T_C)$, thus no leaf of T_C is a leaf in T' , thus (T', r', β', γ) satisfies (PT2). (PT1), (PT3) and (PT4) hold by construction. Furthermore every edge is exact by construction. Thus we get that (T', r', β', γ) is an exact pre-tree decomposition of G° .

The width is obvious as every bag in β' is a subset of some bag in β . To see that the depth bound also holds we observe two things. For every leaf $\ell \in L(T')$ with parent p we get that $\beta'(\ell) \setminus \beta'(p) = \emptyset$. For every inner node $t \in V(T') \setminus L(T')$ with parent p we get that $\beta'(t) \setminus \beta'(p) \subseteq \beta(f(t))$ and, if $p \neq r'$, $\beta'(t) \setminus \beta'(p) \subseteq \beta(f(t)) \setminus \beta(f(p))$, by the tightness of β . ◀

4 The Game

In the cops and robber game on a graph G , the cops occupy sets X of at most k vertices of G , and the robber moves on edges of G . In order to make the rules precise, we need *edge components* of G that arise when the cops are blocking a set X .

► **Definition 11.** Let $G = (V, E)$ be a graph and $X \subseteq V$. We let the edge component graph of G with respect to X be the graph G^X obtained as the disjoint union of the following graphs. (In order to make all graphs disjoint we introduce copies of vertices where needed.)

- For every $uv \in E(G[X])$, the graph $G_{uv} := (\{u, v\}, \{uv\})$, and
- for every connected component C of $G \setminus X$, the graph G_C , with $V(G_C) := V(C) \cup N_G(V(C))$ and $E(G_C) := E(C) \cup E(V(C), X)$, where $E(V(C), X)$ is the set of edges of G incident to both a vertex of C and a vertex in X .

The reader may note that G^X may contain multiple copies of the vertices in X , but exactly one copy of each edge in G .

► **Observation 12.** There is a natural bijection $\Psi: E(G^X) \rightarrow E(G)$ between the edges of G^X and the edges of G .

► **Definition 13** (q -rounds k -cops-and-robber game). Let G be a graph and let $k, q \geq 1$. The q -rounds k -cops-and-robber game $\text{CR}_q^k(G)$ is defined as follows:

If G does not contain any edges the cop player wins immediately.

- The cop positions are sets $X \in V(G)^{\leq k}$.
- The robber position is an edge $uv \in E(G)$.
- The initial position (X_0, u_0v_0) of the game is $X_0 = \emptyset$ and $u_0v_0 \in E(G)$, thus the game starts with no cops positioned on G and the robber on an arbitrary edge in a connected component of G of his choice.
- For $X \subseteq V(G)$ and $uv \in E(G)$, we write $\text{esc}(X, uv) := \Psi(E(C))$ for the component C of the graph G^X , such that $uv \in E(C)$. We call this component the robber escape space. Thus if the cops are at positions X and robber at an edge uv we write $(X, \text{esc}(X, uv))$ for the position of the game.
- In round i the cop player can move from the set X_{i-1} to a set X_i , if $|X_i \setminus X_{i-1}| \leq 1$, that is the cop player can add at most one new vertex to his position.
- In round i the robber player can move along a path with no internal vertex in $X_{i-1} \cap X_i$. Thus the robber player can move to some edge $u_i v_i$, such that the edge $\Psi^{-1}(u_i v_i)$ is in a connected component of G^{X_i} that is contained in $\Psi^{-1}(\text{esc}(u_{i-1} v_{i-1}, X_{i-1} \cap X_i))$ via a path $p = w_1, \dots, w_\ell$ where $\{w_1, w_2\} = \{u_{i-1}, v_{i-1}\}$ and $\{w_{\ell-1}, w_\ell\} = \{u_i, v_i\}$ and $\{w_2, \dots, w_{\ell-1}\} \cap X_i \cap X_{i-1} = \emptyset$.
- The cop-player wins in round i , if $\{u_i, v_i\} \subseteq X_i$, and we say the cop player captures the robber in round i . The robber-player wins if the cop player has not won in round q .

We call the game monotone q -round k -cops-and-robber game, if we further restrict the movement of the cop player such that always $\text{esc}(X_{i-1}, u_{i-1}v_{i-1}) \supseteq \text{esc}(X_{i-1} \cap X_i, u_{i-1}v_{i-1})$ and write $\text{mon-CR}_q^k(G)$.

This notion of monotone is also known as robber-monotone. One can also define the notion of cop-monotone, that is during a single play, the cop player may never revisit a vertex they have previously left. This is the stronger notion of monotone as a cop-monotone play is also robber-monotone. Our construction goes through unchanged with the notion of cop-monotone as the strategy that is derived from the final decomposition is cop-monotone. The game played on the graph G° corresponds to a game on G , where the robber can hide both inside a vertex or an edge. It is easy to see that this does not benefit the robber player, that is he wins the game $\text{CR}_q^k(G)$ if and only if he wins the game $\text{CR}_q^k(G^\circ)$, as the components that are reachable by the robber player are essentially the same. In [13], the authors introduce a cops-and-robber game, where the robber player can only hide in the vertices. Again this does not pose a restriction for the robber player with the same argument as above. There is a tight connection between the cops and robber game defined above and tree decompositions of graphs.

► **Lemma 14** ([13]). *Let G be a graph and $k, q \in \mathbb{N}$. The cop player wins $\text{mon-CR}_q^k(G)$ if and only if $G \in \mathcal{T}_q^k$.*

Towards strengthening the above connection to also include the non-monotone game we first introduce how to construct a pre-tree decomposition from a winning strategy of the cop player.

► **Definition 15** (strategy tree). *Let G be a graph without isolated vertices and let $k, q \in \mathbb{N}$. Let $\sigma: V(G)^{\leq k} \times E(G) \rightarrow V(G)^{\leq k}$ a cop strategy such that for all $X \in V(G)^{\leq k}$, for all $uv \in E(G)$ and for all $u'v' \in \text{esc}(X, uv)$ we have that $\sigma(X, uv) = \sigma(X, u'v')$. We write $\sigma(X, \text{esc}(X, uv))$ instead of $\sigma(X, uv)$.*

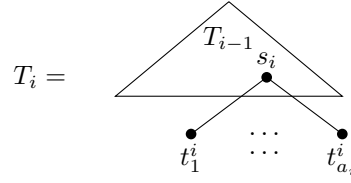
The strategy tree of σ is a pre-tree decomposition (T, r, β, γ) , inductively defined as follows:

- $\beta(r) = \emptyset$,
- for every connected component C of G , there is a child c of the root r and $\gamma(r, c) = E(C)$,
- for every node $t \in V(T) \setminus \{r\}$ with parent $s \in V(T)$,
 - if the robber player is caught, we set $\beta(t) = e$, where $\gamma(s, t) = \{e\}$,
 - else $\beta(t) = \sigma(\beta(s), \gamma(s, t))$ and
 - for every connected component C of $G^{\beta(t)}$, that has a non-empty intersection with $\Psi^{-1}(\gamma(s, t))$, there is a child c of t and $\gamma(t, c) = \Psi(E(C))$,
 - $\gamma(t, s) := E(G) \setminus \bigcup_{c \text{ child of } t} \gamma(t, c)$, if $t \notin L(T)$, and
 - $\gamma(t, s) := E(G) \setminus \gamma(s, t)$, if $t \in L(T)$.

We call $t \in V(T)$ a branching node if the cop player placed a new cop incident to the robber escape space.

Observe that if $t \in V(T)$ is a leaf, then the robber is captured and the depth of (T, r, β, γ) is $\leq q$ if and only if σ is a winning strategy in $\text{CR}_q^k(G)$.

Note that w.l.o.g. every child of the root is a branching node, as the cop player w.l.o.g. only plays positions that are inside the component the robber player chose in the first round. If the game is played on G° , then every branching node that does not correspond to the placement of a cop onto an isolated vertex has more than one child. We observe that the monotone moves of the cop player correspond to the exact edges in the strategy tree by construction.



■ **Figure 1** The subtree T_i appearing in the construction.

► **Lemma 16.** For edge $st \in E(T)$, where $s \prec t$ it holds that the move $\sigma(\beta(s), \gamma(s, t))$ is monotone if and only if st is exact.

The following lemma about the self-loops of the graph G° is key to prove the construction in the next section does not enlarge the depth of the pre-tree decomposition. To prove this one finds the node where the vertex incident to the self-loop was introduced into the bag. At this node the self-loop either was not in the robber escape space in the first place or gets removed from the robber escape space. Then as long as the cops occupy the vertex incident to the self-loop, the self-loop is not reachable by the robber.

► **Lemma 17.** Considering the game on G° and some $s \in V(T) \setminus L(T)$. For all self-loops vv incident to $\beta(s)$ it holds that either $vv \in \gamma(s, p_s)$ or there is a child c of s such that $\gamma(s, c) = \{vv\}$. Furthermore s has a child c with $\gamma(s, c) = \{vv\}$, for some non-isolated vertex v if and only if s is a branching node and $v \in \beta(s) \setminus \beta(p_s)$.

5 Making a Strategy Tree Exact

Our goal is to prove the following theorem.

► **Theorem 18.** Let $G = (V(G), E(G))$ be a graph, let $k, q \geq 1$ and let (T, r, β, γ) be a strategy tree for some cop strategy $\sigma: V(G^\circ)^k \times E(G^\circ) \rightarrow V(G^\circ)^k$. If σ is a winning strategy in $\text{CR}_q^k(G^\circ)$, then there is a tree decomposition of G with width $\leq k - 1$ and depth $\leq q$.

To prove this we construct an exact pre-tree decomposition of G° from the strategy tree, starting at the root r and traversing the tree nodes in a breadth-first-search. We then use Lemma 9 to get the desired tree decomposition. When we consider a node we change the pre-tree decomposition so that all incident edges are exact afterwards. Note that by the choice of the traversal we only need to consider outgoing edges.

The Construction. Let (T, r, β, γ) be the pre-tree decomposition of G° from a winning strategy as in Theorem 18. Let s_1, \dots, s_{n_T} be an order of the nodes of T in bfs where $s_1 = r$. Let $\beta_0 := \beta$ and $\gamma_0 := \gamma$. We construct a sequence $(T, r, \beta_0, \gamma_0), \dots, (T, r, \beta_{n_T}, \gamma_{n_T})$ of pre-tree decompositions, such that $(T, r, \beta_{n_T}, \gamma_{n_T})$ is exact. We say s_i is considered in step i . Let

$$T_i := T[\{s_1, \dots, s_i\} \cup N_T(\{s_1, \dots, s_i\})] = T[V(T_{i-1}) \cup N_T(s_i)].$$

See Figure 1 for an illustration of T_i . (It will become clear that this is the subtree of all nodes where the pre-tree decomposition is modified in or before step i . We also point out that edges from T_i to $T \setminus T_i$ may become non-exact during our modification process.)

If s_i is a leaf, there are no outgoing edges that are not exact, and we set $\beta_i := \beta_{i-1}$ and $\gamma_i := \gamma_{i-1}$. Otherwise let $t_1^i, \dots, t_{a_i}^i \in N_T(s_i)$ be all children of s_i .

- We pick pairwise disjoint $F_1^i, \dots, F_{a_i}^i \subseteq E(G^\circ)$, with

$$F_j^i \subseteq \overline{\gamma_{i-1}(t_j^i, s_i)} \cap \overline{\gamma_{i-1}(s_i, t_j^i)},$$

such that the partition π^* that results from taking the F_j^i -extensions in $\gamma_{i-1}(s_i, t_j^i)$ (in arbitrary order) has the minimum size boundary. If there are multiple optimal choices for $F_1^i, \dots, F_{a_i}^i$ we select the one that minimizes the size of $\bigcup_{j \in [a_i]} F_j^i$, if there are still several options we break ties arbitrarily.

- Let $F^i := \bigcup_{j \in [a_i]} F_j^i$ and $R_j^i := \left(\left(\overline{\gamma_{i-1}(t_j^i, s_i)} \cap \overline{\gamma_{i-1}(s_i, t_j^i)} \right) \cup F^i \right) \setminus F_j^i$. For every vertex $p \in V(T_i)$ with child c we set

$$\gamma_i(p, c) := \begin{cases} (\gamma_{i-1}(s_i, t_j^i) \setminus F^i) \cup F_j^i & \text{if } (p, c) = (s_i, t_j^i), \text{ for some } j \in [a_i], \\ \gamma_{i-1}(p, c) \setminus R_j^i & \text{if } p = t_j^i, \text{ for some } j \in [a_i], \\ \gamma_{i-1}(p, c) \cup F^i & \text{if } p \prec s_i, \\ \gamma_{i-1}(p, c) \setminus F^i & \text{otherwise,} \end{cases}$$

and

$$\gamma_i(c, p) := \begin{cases} \gamma_{i-1}(c, p) \cup R_j^i & \text{if } (p, c) = (s_i, t_j^i), \text{ for some } j \in [a_i], \\ \gamma_{i-1}(c, p) & \text{if } p = t_j^i, \text{ for some } j \in [a_i], \\ \gamma_{i-1}(c, p) \setminus F^i & \text{if } p \prec s_i, \\ \gamma_{i-1}(c, p) \cup F^i & \text{otherwise,} \end{cases}$$

and all other $uv \in \overrightarrow{E(T)}$ we set $\gamma_i(u, v) := \gamma_{i-1}(u, v)$. Furthermore we set

$$\beta_i(t) := \begin{cases} \delta(\pi_t^i) & \text{if } t \in V(T_i), \\ \beta_{i-1}(t) & \text{otherwise.} \end{cases}$$

Intuitively in the construction above we pick F_j^i , that is the set of edges that we add to the cone at the arc pointing towards t_j^i , from the set of edges that are not covered by the cones along the arcs in such a way that the boundary at s_i is minimized. R_j^i then corresponds to the edges that we need to remove from all cones at arcs that point away from s_i at the child t_j^i and in turn add to the cone at the arc pointing towards s_i to make the edge exact. Then we push the change at s_i through T_{i-1} , that is for all edges in T_{i-1} we add F^i to the arc that points towards s_i and remove F^i from the arcs in the other direction. This push can also be interpreted in terms of extensions.

- **Lemma 19.** *Let $t \in V(T_i) \setminus \{s_i\}$ and t' is next node on the path from t to s_i in T . Then*

$$\pi_t^i = \left(\pi_t^{i-1} \right)_{\gamma_{i-1}(t, t') \rightarrow \overline{\gamma_i(t', t)}}$$

and if additionally $t \in V(T_{i-1})$ also

$$\pi_t^i = \left(\pi_t^{i-1} \right)_{\gamma_{i-1}(t, t') \rightarrow F^i}.$$

Observe furthermore that if $\beta_i(s_i) = \beta_{i-1}(s_i)$, then there are no changes to the bags at other nodes than the t_j^i by minimality of $|F^i|$, and if $\beta_i(s_i) \neq \beta_{i-1}(s_i)$, we have $|\beta_i(s_i)| < |\beta_{i-1}(s_i)|$ again by the minimality of the choice. For every $i \in [n_T]$, before step i we only remove edges from the cones pointing downwards from s_i , thus we obtain the following observation.

- **Lemma 20.** *Let $i, j \in [n_T]$ such that s_i is the parent of s_j . Then $\gamma_\alpha(s_i, s_j) \subseteq \gamma(s_i, s_j)$, for all $\alpha < i$.*

The Proof Idea. We prove Theorem 18 in three steps. First we prove that the construction indeed yields an exact pre-tree decomposition. Next we show that the width can be bounded as desired and lastly we prove that the construction yields the desired depth.

In step i , every edge incident to s_i is made exact and for every other edge in T_i we remove from one arc exactly what we add to the other arc. We get that our construction indeed yields an exact pre-tree decomposition.

► **Lemma 21.** *For all $i \in [n_T]$, $(T, r, \beta_i, \gamma_i)$ is a pre-tree decomposition. Furthermore all edges in $E(T_i)$ are exact.*

Hence, for $i = n_T$, we get that $(T, r, \beta_{n_T}, \gamma_{n_T})$ is an exact pre-tree decomposition. Note that it is possible that $\gamma_{n_T}(s, t)$ is empty for an arc $(s, t) \in \overrightarrow{E(T)}$. By Lemma 9 we obtain a tree decomposition, from this pre-tree decomposition. We show below that the width and depth are as stated in the theorem.

Our construction does not change the width of the decomposition. To prove this we observe that in step i the bound in s_i is minimal. We then push the change through the subtree T_i and find that if a change would increase the width, we could push this change back to the node s_i and find an even smaller bound there, which contradicts the minimality of our choice. This argument yields the following lemma.

► **Lemma 22.** $\text{wd}(T, r, \beta_i, \gamma_i) \leq \text{wd}(T, r, \beta, \gamma)$, for all $i \in [n_T]$.

Proof. $\overline{\gamma_i(t', t)}$ We prove the statement for all $0 \leq i \leq n_T$ by induction. As $(T, r, \beta_0, \gamma_0) = (T, r, \beta, \gamma)$, the statement clearly holds for $i = 0$. Next we show that $\text{wd}(T, r, \beta_i, \gamma_i) \leq \text{wd}(T, r, \beta_{i-1}, \gamma_{i-1})$, for all $i \in [n_T]$. Obviously $|\beta_i(t)| = |\beta_{i-1}(t)|$, for all $t \notin T_i$. Furthermore by construction $|\beta_i(s_i)| \leq |\beta_{i-1}(s_i)|$. Let $j \in [a_i]$, let $X := \gamma_i(s_i, t_j^i)$ and let $Y := \gamma_{i-1}(t_j^i, s_i)$. By Lemma 19 it holds that

$$|\beta_i(t_j^i)| = \text{wd} \left(\left(\pi_{t_j^i}^{i-1} \right)_{Y \rightarrow \overline{X}} \right) \leq \text{wd}(\pi_{t_j^i}^{i-1}) \leq |\beta_{i-1}(t_j^i)|$$

as otherwise by submodularity for the partitions $\pi_{t_j^i}^{i-1}$ and $\pi_{s_i}^i$, we get that

$$\text{wd} \left(\left(\pi_{s_i}^i \right)_{X \rightarrow \overline{Y}} \right) < \text{wd}(\pi_{s_i}^i),$$

which contradicts the minimality of the bound for $F_1^i, \dots, F_{a_i}^i$.

Lastly assume there is a node t in $V(T_i) \setminus \{s_i, t_1^i, \dots, t_{a_i}^i\}$ such that $|\beta_i(t)| > |\beta_{i-1}(t)|$. We assume t is of minimal distance to s_i with this property. Let $x_0 = t, x_1, \dots, x_b = s_i$ be the path from t to s_i . By minimality of the distance we know that $|\beta_i(x_1)| \leq |\beta_{i-1}(x_1)|$. Additionally we know that all edges on the path from s_i to x_1 are exact in γ_i , as well as the edge $x_1 t$ in γ_{i-1} . Now let $Y := \gamma_{i-1}(t, x_1)$, $X_\alpha := \gamma_i(x_{\alpha+1}, x_\alpha)$ and $Z_\alpha := \gamma_i(x_\alpha, x_{\alpha+1})$, for all $0 \leq \alpha < b$. From Lemma 19 we get $(\pi_t^{i-1})_{Y \rightarrow F^i} = (\pi_t^{i-1})_{Y \rightarrow \overline{X_0}}$. Thus assuming that $\text{wd} \left((\pi_t^{i-1})_{Y \rightarrow F^i} \right) = |\beta_i(t)| > |\beta_{i-1}(t)| = \text{wd}(\pi_t^{i-1})$ using submodularity we get that $\text{wd}(\pi_{x_1}^i) > \text{wd} \left((\pi_{x_1}^i)_{X_0 \rightarrow \overline{Y}} \right)$. As the edge $x_1 t$ was exact at step $i - 1$, we know that

$$F' := \overline{Y} \setminus X_0 = F^i \setminus Y \subseteq F^i.$$

We now push this change back to s_i along the path x_1, \dots, x_b and we again find a contradiction to the minimality of the bound of $F_1^i, \dots, F_{a_i}^i$. For this, let us assume we have pushed the change to x_α , that is we changed $\pi_{x_\alpha}^i$ to $\pi_{x_\alpha}^*$ and we know that $\text{wd}(\pi_{x_\alpha}^*) < \text{wd}(\pi_{x_\alpha}^i)$. As $x_\alpha x_{\alpha+1}$ is exact in γ_i , we get that $(\pi_{x_\alpha}^*)_{(Z_\alpha \setminus F') \rightarrow \overline{X_\alpha}} = \pi_{x_\alpha}^i$. Let

$$\pi_{x_{\alpha+1}}^* := \left(\pi_{x_{\alpha+1}}^i \right)_{X_\alpha \rightarrow \overline{(Z_\alpha \setminus F')}} = \left(\pi_{x_{\alpha+1}}^i \right)_{X_\alpha \rightarrow F'},$$

then by submodularity $\text{wd}(\pi_{x_{\alpha+1}}^*) < \text{wd}(\pi_{x_{\alpha+1}}^i)$. When we have pushed the change to $\alpha = b$, we find the desired contradiction. ◀

To prove that our construction does not increase the depth we show that in every step i the depth up to the nodes in T_i is bounded by the depth up to these nodes in the original tree. We prove this by induction on the number of steps. We recall that $V(T_i) = V(T_{i-1}) \cup \{t_1^i, \dots, t_{a_i}^i\}$ and that $s_i \in L(T_{i-1})$. For the nodes $t \in V(T_{i-1})$ we can directly build upon the induction hypothesis. But the nodes t_j^i , with $j \in [a_i]$, are added into the subtree. Here we need to compare directly to the original bags, as we can no longer use that in step $i-1$ the depth at these nodes is bounded by the depth in the original strategy tree. We can prove for these nodes that every vertex newly placed at one of these nodes in step i is also newly placed in the original strategy. Then we can show that the difference between depth at these nodes and their parent in step i can be bounded by the difference in the original strategy tree.

► **Lemma 23.** *Every $j \in [a_i]$ satisfies $\beta_i(t_j^i) \setminus \beta_i(s_i) \subseteq \beta(t_j^i) \setminus \beta(s_i)$.*

Proof. Let $v \in \beta_i(t_j^i) \setminus \beta_i(s_i)$. Since $v \in \beta_i(t_j^i)$ it holds that $E_{G^\circ}(v) \not\subseteq \gamma_i(t_j^i, s_i)$. As $v \notin \beta_i(s_i)$ we get that $v \notin \delta(\gamma_i(t_j^i, s_i))$ and thus $E_{G^\circ}(v) \cap \gamma_i(t_j^i, s_i) = \emptyset$. By construction we have that $\gamma_i(t_j^i, s_i) \supseteq \gamma_{i-1}(t_j^i, s_i) = \gamma(t_j^i, s_i)$, and thus $vv \notin \gamma(t_j^i, s_i)$. As $v \in \beta_i(t_j^i) = \delta(\pi_{t_j^i}^i)$, there are two distinct children c_1, c_2 of t_j^i such that $v \in \delta(\gamma_i(t_j^i, c_\ell))$ and thus $E_{G^\circ}(v) \cap \gamma_i(t_j^i, c_\ell) \neq \emptyset$, for $\ell = 1, 2$. By construction we have $\gamma_i(t_j^i, c_\ell) \subseteq \gamma_{i-1}(t_j^i, c_\ell) = \gamma(t_j^i, c_\ell)$, for $\ell = 1, 2$. And thus $v \in \delta(\pi_{t_j^i}^i) \subseteq \beta(t_j^i)$. By Lemma 17 there thus is a child c of t_j^i such that $\gamma(t_j^i, c) = \{vv\}$ and, by Lemma 17, $v \in \beta(t_j^i) \setminus \beta(s_i)$. ◀

For the nodes $t \in V(T_{i-1})$ we show that the depth is not only bounded by the depth within the original pre-tree decomposition, but within the previous step. We do this by a vertex exchange argument, that is we track all vertices added or removed from any bag within T_{i-1} . For the vertices that are added to any bag in $V(T_{i-1})$ we get the following lemma.

► **Lemma 24.** *Let $i \in [n_T]$ and let $t \in V(T_{i-1})$. If $v \in \beta_i(t) \setminus \beta_{i-1}(t)$, then $v \in \beta_i(t^*)$, for all t^* on the path from t to s_i .*

Proof. Let $t^* \neq t$. Let t' be the next node on the path from t to s_i . Per definition it holds that $\gamma_i(t, t') = \gamma_{i-1}(t, t') \cup F^i$. As $\gamma_i(t, t')$ is the only set incident to t where edges are added in step i , we get that $v \in \delta(\gamma_i(t, t'))$. Combined with $v \notin \delta(\gamma_{i-1}(t, t'))$ we get that $E_{G^\circ}(v) \cap \gamma_{i-1}(t, t') = \emptyset$ and thus $\emptyset \neq E_{G^\circ}(v) \cap F^i \neq E_{G^\circ}(v)$. Now suppose that $v \notin \beta_i(t^*)$, and thus also $v \notin \delta(\gamma_i(t^*, p))$, where p is the next node on the path from t^* to t . As $\emptyset \neq E_{G^\circ}(v) \cap F^i \neq E_{G^\circ}(v)$ this implies that $E_{G^\circ}(v) \cap \gamma_i(t^*, p) = E_{G^\circ}(v) \cap (\gamma_{i-1}(t^*, p) \setminus F_i) = \emptyset$. We know from Lemma 21 that all edges in T_i are exact and thus that $\gamma_i(t', t) \subseteq \gamma_i(t^*, p)$ by Lemma 6. Thus we get that $E_{G^\circ}(v) \cap \gamma_i(t', t) = \emptyset$. This is a contradiction to the assumption that $v \in \delta(\gamma_i(t, t')) = \delta(\gamma_i(t', t))$ and thus $v \in \beta_i(t^*)$. ◀

The next lemma is used to show that a vertex that disappears from a bag is also removed from all bags that determine the depth at that bag, especially if a vertex disappears from the bag at s_i , then it disappears from every bag in $V(T_i)$.

► **Lemma 25.** *Let $i \in [n_T]$ and let $t \in V(T_{i-1})$. If $v \in \beta_{i-1}(t) \setminus \beta_i(t)$, then $v \notin \beta_i(t^*)$, for all $t^* \in V(T_{i-1})$ such that t is contained in the path from t^* to s_i .*

6:14 Monotonicity of the Cops and Robber Game for Bounded Depth Treewidth

Proof. We have $E_{G^\circ}(v) \cap F^i \neq \emptyset$.

Let $t = s_i$. Since $v \in \beta_{i-1}(s_i) = \delta(\pi_{s_i}^{i-1})$ it holds that $E_{G^\circ}(v) \not\subseteq \gamma_{i-1}(s_i, p_{s_i})$. As $v \notin \beta_i(s_i)$ we get that $v \notin \delta(\gamma_i(s_i, p_{s_i}))$ and thus $E_{G^\circ}(v) \cap \gamma_i(s_i, p_{s_i}) = E_{G^\circ}(v) \cap \gamma_{i-1}(s_i, p_{s_i}) \cap F^i = \emptyset$. Now let $t^* \in V(T_{i-1})$ and t' be the next node on the path from t^* to s_i . Then by Lemma 21 we get that $\gamma_i(t^*, t') \supseteq \gamma_i(p_{s_i}, s_i) \supseteq E_{G^\circ}(v)$ and thus $v \notin \beta_i(t^*)$.

Otherwise let $t \neq s_i$. Let t' be the next node on the path from t to s_i . Since $v \in \delta(\pi_t^{i-1})$, we get that $E_{G^\circ}(v) \not\subseteq \gamma_{i-1}(t, t')$. As $v \notin \delta(\gamma_i(t, t'))$ and $E_{G^\circ}(v) \cap F^i \neq \emptyset$ it follows that $E_{G^\circ}(v) \subseteq \gamma_i(t, t') = \gamma_{i-1}(t, t') \cup F^i$ and that $E_{G^\circ}(v) \cap \gamma_{i-1}(t', t) \subseteq E_{G^\circ}(v) \cap F^i$. Assume there is some $t^* \in V(T_{i-1})$ such that $v \in \beta_i(t^*)$. We observe that due to Lemma 21 and because all edges incident to v are contained in $\gamma_i(t, t')$, we get that t is not contained in the path from t^* to s_i . \blacktriangleleft

This induction then yields the following lemma.

► **Lemma 26.** *For all $i \in [n_T]$ and all $t \in V(T_i)$, it holds that*

$$\sum_{r \prec s \preceq t} |\beta_i(s) \setminus \beta_i(p_s)| \leq \sum_{r \prec s \preceq t} |\beta(s) \setminus \beta(p_s)|.$$

Proof. Let $\ell \in [n_T]$. As by construction $\beta_\ell(t) = \delta(\pi_t^\ell)$, for all $t \in V(T_\ell)$, we get from Lemma 8 and Lemma 21 that $|\bigcup_{s \preceq t} \beta_\ell(s)| = \sum_{r \prec s \preceq t} |\beta_\ell(s) \setminus \beta_\ell(p_s)|$. Thus it suffices to show that $|\bigcup_{s \preceq t} \beta_i(s)| \leq \sum_{r \prec s \preceq t} |\beta(s) \setminus \beta(p_s)|$.

We prove the statement by induction on the steps i . Recall that $(T, r, \beta_0, \gamma_0) = (T, r, \beta, \gamma)$, thus the statement holds for $i = 0$. Now assume the statement holds for $i - 1$, thus for all $t \in V(T_{i-1})$ it holds that $|\bigcup_{s \preceq t} \beta_{i-1}(s)| \leq \sum_{r \prec s \preceq t} |\beta(s) \setminus \beta(p_s)|$. We recall that $V(T_i) = V(T_{i-1}) \cup \{t_1^i, \dots, t_{a_i}^i\}$ and that $s_i \in L(T_{i-1})$. We consider all vertices that appear at a bag at any node in T_i due to the changes in step i .

Let $t \in V(T_{i-1})$. Let $U := \left(\bigcup_{s \preceq t} \beta_i(s)\right) \setminus \left(\bigcup_{s \preceq t} \beta_{i-1}(s)\right)$. Let t^* be the greatest common ancestor of t and s_i . As t^* is on every path from some node $s \preceq t$ to s_i , from Lemma 24 we know that $u \in \beta_i(t^*) \setminus \beta_{i-1}(t^*)$, for all $u \in U$. Let $W := \beta_{i-1}(t^*) \setminus \beta_i(t^*)$. As by Lemma 22 $|\beta_i(t^*)| \leq |\beta_{i-1}(t^*)|$, we know that $|U| \leq |W|$. Applying Lemma 25 we get that $W \subseteq \left(\bigcup_{s \preceq t} \beta_{i-1}(s)\right) \setminus \left(\bigcup_{s \preceq t} \beta_i(s)\right)$ and using this *vertex exchange* we conclude that $\left|\bigcup_{s \preceq t} \beta_i(s)\right| \leq \left|\bigcup_{s \preceq t} \beta_{i-1}(s)\right|$.

Otherwise it holds that $t = t_j^i$ for some $j \in [a_i]$. By construction we can conclude that $\bigcup_{s \preceq t} \beta_i(s) = \bigcup_{s \preceq s_i} \beta_i(s) \cup \beta_i(t) \setminus \beta_i(s_i)$. We know that $|\bigcup_{s \preceq s_i} \beta_i(s)| \leq \sum_{r \prec s \preceq s_i} |\beta(s) \setminus \beta(p_s)|$ and by Lemma 23 we have $\beta_i(t) \setminus \beta_i(s_i) \subseteq \beta(t) \setminus \beta(s_i)$. Thus we can bound the union $|\bigcup_{s \preceq t} \beta_i(s)| \leq \sum_{r \prec s \preceq t} |\beta(s) \setminus \beta(p_s)|$. \blacktriangleleft

Summarising all results we get the following equivalences.

► **Theorem 27.** *Let $k, q \geq 1$ and G be a graph. The following are equivalent:*

- (1) G admits a tree decomposition of width at most $k - 1$ and depth at most q .
- (2) G° admits a tree decomposition of width at most $k - 1$ and depth at most q .
- (3) G° admits an exact pre-tree decomposition of width at most $k - 1$ and depth at most q .
- (4) The cop player wins $\text{mon-CR}_q^k(G^\circ)$.
- (5) The cop player wins $\text{CR}_q^k(G^\circ)$.
- (6) The cop player wins $\text{mon-CR}_q^k(G)$.
- (7) The cop player wins $\text{CR}_q^k(G)$.

6 Excursion on Counting Homomorphisms

In this section we give an overview over the field of counting homomorphisms and the equivalence relations on graphs, that can be derived from these counts. We focus ourselves to the results and open questions regarding the homomorphism counts from graphs in the class \mathcal{T}_q^k , for fixed $k, q \geq 0$.

We recall the definition of homomorphism distinguishing closed from the introduction. In [13] the authors have reduced the question whether the class \mathcal{T}_q^k is homomorphism distinguishing closed down to the question if monotonicity is a restriction for the cop player. The following lemma is thus implied in [13].

► **Lemma 28** ([13]). *Let $k, q \geq 1$. The graph class $\mathcal{C} := \{G \mid \text{cop player wins } \text{CR}_q^k(G)\}$ is homomorphism distinguishing closed.*

In this paper we show that the cop player wins $\text{CR}_q^k(G)$ if and only if $G \in \mathcal{T}_q^k$, thus the we get the following.

► **Theorem 29**. *Let $k, q \geq 0$. The class \mathcal{T}_q^k is homomorphism distinguishing closed.*

7 Conclusion

We gave a new characterisation of bounded depth treewidth by the cops and robber game with both a bound on the number of cops and on the number of rounds, where the cop player is allowed to make non-monotone moves. As a corollary we gave a positive answer to an open question on homomorphism counts. The core of our contribution is a proof of monotonicity of this game. For this proof we substantially reorganise a winning strategy. First we transform it into a pre-tree decomposition. Then we apply a breadth-first “cleaning up” procedure along the pre-tree decomposition (which may temporarily lose the property of representing a strategy), in order to achieve monotonicity while controlling the number of cop rounds simultaneously across all branches of the decomposition via a vertex exchange argument. As an interesting observation we obtain that cop moves onto some vertex not incident to the robber escape space, i. e. to positions that are not part of the boundary, can be ignored and the depth of the exact pre-tree decomposition is the number of cops placed into the robber escape space. To see that consider the proof of Lemma 23 where we compute how much larger the depth at some node t_j^i at step i is than at the considered node s_i . The depth increases only if the node t_j^i is branching by Lemma 17 as t_j^i has a child where the cone contains only a self-loop and hence this is a move into the robber escape space.

► **Corollary 30**. $\text{dp}(T, r, \beta_{n_T}, \gamma_{n_T}) \leq \max_{\ell \in L(T)} |\{t \in V(T) \mid t \preceq \ell \text{ and } t \text{ is branching}\}|$.

We note that we use a slightly different notion of branching node as [32], as we use a different game characterisation. On a graph without isolated vertices our branching nodes are branching nodes in the sense of [32], but not the other way around, as in the non-deterministic cops and robber game, the cop player can chose if he wants to branch in the strategy tree, that is if he wants to know the position of the robber or not.

In the future, it would be interesting to know if it is possible to give a proof that entirely argues with game strategies (not requiring pre-tree decompositions), and we leave this open. We also leave open whether a dual object similar to brambles can be defined for bounded depth treewidth. Finally, given a winning strategy for k cops with q rounds, it would be interesting to know if it is possible to bound the number of cops necessary for winning with only $q - 1$ rounds in terms of k and q , given that the cop player still can win.

References

- 1 Samson Abramsky, Anuj Dawar, and Pengming Wang. The pebbling comonad in finite model theory. In *32nd Annual ACM/IEEE Symposium on Logic in Computer Science, LICS 2017, Reykjavik, Iceland, June 20-23, 2017*, pages 1–12. IEEE Computer Society, 2017. doi:10.1109/LICS.2017.8005129.
- 2 Isolde Adler. Marshals, monotone marshals, and hypertree-width. *J. Graph Theory*, 47(4):275–296, 2004. doi:10.1002/JGT.20025.
- 3 Isolde Adler. Games for width parameters and monotonicity. *CoRR*, abs/0906.3857, 2009. arXiv:0906.3857.
- 4 Isolde Adler and Eva Fluck. Monotonicity of the cops and robber game for bounded depth treewidth. *CoRR*, abs/2402.09139, 2024. doi:10.48550/arXiv.2402.09139.
- 5 Isolde Adler, Georg Gottlob, and Martin Grohe. Hypertree width and related hypergraph invariants. *Eur. J. Comb.*, 28(8):2167–2181, 2007. doi:10.1016/J.EJC.2007.04.013.
- 6 Martin Aigner and M. Fromme. A game of cops and robbers. *Discret. Appl. Math.*, 8(1):1–12, 1984. doi:10.1016/0166-218X(84)90073-8.
- 7 Omid Amini, Frédéric Mazoit, Nicolas Nisse, and Stéphan Thomassé. Submodular partition functions. *Discret. Math.*, 309(20):6000–6008, 2009. doi:10.1016/J.DISC.2009.04.033.
- 8 Daniel Bienstock. Graph searching, path-width, tree-width and related problems (A survey). In Fred Roberts, Frank Hwang, and Clyde L. Monma, editors, *Reliability Of Computer And Communication Networks, Proceedings of a DIMACS Workshop, New Brunswick, New Jersey, USA, December 2-4, 1989*, volume 5 of *DIMACS Series in Discrete Mathematics and Theoretical Computer Science*, pages 33–50. DIMACS/AMS, 1989. doi:10.1090/DIMACS/005/02.
- 9 Daniel Bienstock and Paul D. Seymour. Monotonicity in graph searching. *J. Algorithms*, 12(2):239–245, 1991. doi:10.1016/0196-6774(91)90003-H.
- 10 Hans L. Bodlaender and Dimitrios M. Thilikos. Computing small search numbers in linear time. In Rodney G. Downey, Michael R. Fellows, and Frank K. H. A. Dehne, editors, *Parameterized and Exact Computation, First International Workshop, IWPEC 2004, Bergen, Norway, September 14-17, 2004, Proceedings*, volume 3162 of *Lecture Notes in Computer Science*, pages 37–48. Springer, 2004. doi:10.1007/978-3-540-28639-4_4.
- 11 Anuj Dawar, Tomáš Jakl, and Luca Reggio. Lovász-Type Theorems and Game Comonads. In *2021 36th Annual ACM/IEEE Symposium on Logic in Computer Science (LICS)*, pages 1–13, June 2021. doi:10.1109/LICS52264.2021.9470609.
- 12 Zdeněk Dvořák. On recognizing graphs by numbers of homomorphisms. *Journal of Graph Theory*, 64(4):330–342, August 2010. doi:10.1002/jgt.20461.
- 13 Eva Fluck, Tim Seppelt, and Gian Luca Spitzer. Going Deep and Going Wide: Counting Logic and Homomorphism Indistinguishability over Graphs of Bounded Treedepth and Treewidth. In Aniello Murano and Alexandra Silva, editors, *32nd EACSL Annual Conference on Computer Science Logic (CSL 2024)*, volume 288 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 27:1–27:17, Dagstuhl, Germany, 2024. Schloss Dagstuhl – Leibniz-Zentrum für Informatik. doi:10.4230/LIPIcs.CSL.2024.27.
- 14 Fedor V. Fomin, Pierre Fraignaud, and Nicolas Nisse. Nondeterministic graph searching: From pathwidth to treewidth. *Algorithmica*, 53(3):358–373, 2009. doi:10.1007/S00453-007-9041-6.
- 15 Fedor V. Fomin, Petr A. Golovach, and Jan Kratochvíl. On tractability of cops and robbers game. In Giorgio Ausiello, Juhani Karhumäki, Giancarlo Mauri, and C.-H. Luke Ong, editors, *Fifth IFIP International Conference On Theoretical Computer Science - TCS 2008, IFIP 20th World Computer Congress, TC 1, Foundations of Computer Science, September 7-10, 2008, Milano, Italy*, volume 273 of *IFIP*, pages 171–185. Springer, 2008. doi:10.1007/978-0-387-09680-3_12.
- 16 Fedor V. Fomin and Dimitrios M. Thilikos. An annotated bibliography on guaranteed graph searching. *Theor. Comput. Sci.*, 399(3):236–245, 2008. doi:10.1016/J.TCS.2008.02.040.

- 17 Matthew K. Franklin, Zvi Galil, and Moti Yung. Eavesdropping games: a graph-theoretic approach to privacy in distributed systems. *J. ACM*, 47(2):225–243, 2000. doi:10.1145/333979.333980.
- 18 Archontia C. Giannopoulou, Paul Hunter, and Dimitrios M. Thilikos. Lifo-search: A min-max theorem and a searching game for cycle-rank and tree-depth. *Discret. Appl. Math.*, 160(15):2089–2097, 2012. doi:10.1016/J.DAM.2012.03.015.
- 19 Archontia C. Giannopoulou and Dimitrios M. Thilikos. A min-max theorem for lifo-search. *Electron. Notes Discret. Math.*, 38:395–400, 2011. doi:10.1016/J.ENDM.2011.09.064.
- 20 Martin Grohe. Counting Bounded Tree Depth Homomorphisms. In *Proceedings of the 35th Annual ACM/IEEE Symposium on Logic in Computer Science, LICS '20*, pages 507–520, New York, NY, USA, 2020. Association for Computing Machinery. event-place: Saarbrücken, Germany. doi:10.1145/3373718.3394739.
- 21 Martin Grohe and Dániel Marx. Constraint solving via fractional edge covers. *ACM Trans. Algorithms*, 11(1):4:1–4:20, 2014. doi:10.1145/2636918.
- 22 Martin Grohe, Gaurav Rattan, and Tim Seppelt. Homomorphism Tensors and Linear Equations. In Mikołaj Bojańczyk, Emanuela Merelli, and David P. Woodruff, editors, *49th International Colloquium on Automata, Languages, and Programming (ICALP 2022)*, volume 229 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 70:1–70:20, Dagstuhl, Germany, 2022. Schloss Dagstuhl – Leibniz-Zentrum für Informatik. doi:10.4230/LIPIcs.ICALP.2022.70.
- 23 Petr Hliněný and Geoff Whittle. Matroid tree-width. *Eur. J. Comb.*, 27(7):1117–1128, 2006. doi:10.1016/J.EJC.2006.06.005.
- 24 Geoffrey A. Hollinger, Athanasios Kehagias, and Sanjiv Singh. GSST: anytime guaranteed search. *Auton. Robots*, 29(1):99–118, 2010. doi:10.1007/S10514-010-9189-9.
- 25 Paul Hunter and Stephan Kreutzer. Digraph measures: Kelly decompositions, games, and orderings. *Theor. Comput. Sci.*, 399(3):206–219, 2008. doi:10.1016/J.TCS.2008.02.038.
- 26 Thor Johnson, Neil Robertson, Paul D. Seymour, and Robin Thomas. Directed tree-width. *J. Comb. Theory, Ser. B*, 82(1):138–154, 2001. doi:10.1006/JCTB.2000.2031.
- 27 Stephan Kreutzer and Sebastian Ordyniak. Digraph decompositions and monotonicity in digraph searching. *Theor. Comput. Sci.*, 412(35):4688–4703, 2011. doi:10.1016/j.tcs.2011.05.003.
- 28 Andrea S. LaPaugh. Recontamination does not help to search a graph. *J. ACM*, 40(2):224–245, 1993. doi:10.1145/151261.151263.
- 29 László Lovász. Operations with structures. *Acta Mathematica Academiae Scientiarum Hungarica*, 18(3):321–328, September 1967. doi:10.1007/BF02280291.
- 30 Fillia Makedon and Ivan Hal Sudborough. On minimizing width in linear layouts. *Discret. Appl. Math.*, 23(3):243–265, 1989. doi:10.1016/0166-218X(89)90016-4.
- 31 Laura Mančínska and David E. Roberson. Quantum isomorphism is equivalent to equality of homomorphism counts from planar graphs. In *2020 IEEE 61st Annual Symposium on Foundations of Computer Science (FOCS)*, pages 661–672, 2020. doi:10.1109/FOCS46700.2020.00067.
- 32 Frédéric Mazoit and Nicolas Nisse. Monotonicity of non-deterministic graph searching. *Theor. Comput. Sci.*, 399(3):169–178, 2008. doi:10.1016/J.TCS.2008.02.036.
- 33 Jaroslav Nesetril and Patrice Ossona de Mendez. Tree-depth, subgraph coloring and homomorphism bounds. *Eur. J. Comb.*, 27(6):1022–1041, 2006. doi:10.1016/J.EJC.2005.01.010.
- 34 Daniel Neuen. Homomorphism-Distinguishing Closedness for Graphs of Bounded Tree-Width, April 2023. doi:10.48550/arXiv.2304.07011.
- 35 Jan Obdržálek. Dag-width: connectivity measure for directed graphs. In *Proceedings of the Seventeenth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2006, Miami, Florida, USA, January 22-26, 2006*, pages 814–821. ACM Press, 2006. URL: <http://dl.acm.org/citation.cfm?id=1109557.1109647>.
- 36 T. D. Parsons. Pursuit-evasion in a graph. In Yousef Alavi and Don R. Lick, editors, *Theory and Applications of Graphs*, pages 426–441, Berlin, Heidelberg, 1978. Springer Berlin Heidelberg.

- 37 Torrence D Parsons. The search number of a connected graph. In *Proc. 9th South-Eastern Conf. on Combinatorics, Graph Theory, and Computing*, pages 549–554, 1978.
- 38 Nikolai N. Petrov. A problem of pursuit in the absence of information on the pursued. *Differentsial'nye Uravneniya*, 18(1):345–1352, 1982.
- 39 David E. Roberson. Oddomorphisms and homomorphism indistinguishability over graphs of bounded degree, June 2022. doi:10.48550/arXiv.2206.10321.
- 40 David E. Roberson and Tim Seppelt. Lasserre Hierarchy for Graph Isomorphism and Homomorphism Indistinguishability. In Kousha Etessami, Uriel Feige, and Gabriele Puppis, editors, *50th International Colloquium on Automata, Languages, and Programming (ICALP 2023)*, volume 261 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 101:1–101:18, Dagstuhl, Germany, 2023. Schloss Dagstuhl – Leibniz-Zentrum für Informatik. doi:10.4230/LIPIcs.ICALP.2023.101.
- 41 Benjamin Scheidt and Nicole Schweikardt. Counting homomorphisms from hypergraphs of bounded generalised hypertree width: A logical characterisation. In Jérôme Leroux, Sylvain Lombardy, and David Peleg, editors, *48th International Symposium on Mathematical Foundations of Computer Science, MFCS 2023, August 28 to September 1, 2023, Bordeaux, France*, volume 272 of *LIPIcs*, pages 79:1–79:15. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2023. doi:10.4230/LIPICS.MFCS.2023.79.
- 42 Tim Seppelt. Logical Equivalences, Homomorphism Indistinguishability, and Forbidden Minors. In Jérôme Leroux, Sylvain Lombardy, and David Peleg, editors, *48th International Symposium on Mathematical Foundations of Computer Science (MFCS 2023)*, volume 272 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 82:1–82:15, Dagstuhl, Germany, 2023. Schloss Dagstuhl – Leibniz-Zentrum für Informatik. doi:10.4230/LIPIcs.MFCS.2023.82.
- 43 Paul D. Seymour and Robin Thomas. Graph searching and a min-max theorem for tree-width. *J. Comb. Theory, Ser. B*, 58(1):22–33, 1993. doi:10.1006/JCTB.1993.1027.