Equivalent Definitions for Block Elimination Distance and a Polynomial Kernel

Filippos Mavropoulos
R.N: AL1190015

Examination committee:
Archontia C. Giannopoulou, Department of Informatics and Telecommunications, National and Kapodistrian University of Athens.
Stavros G. Kolliopoulos, Department of Informatics and Telecommunications, National and Kapodistrian University of Athens.
Dimitrios M. Thilikos, Department of Mathematics, National and Kapodistrian University of Athens.

Supervisor:
Archontia C. Giannopoulou, Assistant Professor, Department of Informatics and Telecommunications
National and Kapodistrian University of Athens.

ABSTRACT

Block elimination distance is a parameter on graphs that measures the distance of the biconnected components of a graph from a given class of graphs. When the indicated graph class is the class of edgeless graphs, we call the parameter *block treedepth*. In this thesis, we prove that block treedepth is equivalent to the minimum number of colors needed to color a graph such that every biconnected subgraph has a vertex of unique color. Additionally, we introduce a special kind of non-proper edge coloring that can serve as an alternative for block treedepth, called *cycle edge ranking*. Moreover, we make a connection between block treedepth and graph searching games by introducing two versions of the cops and robbers game that can be used to calculate the block treedepth of a graph. Finally, we prove that block treedepth has a polynomial kernel when parameterized by the vertex cover number.

ΣΥΝΟΨΗ

Η *δισυνεκτική απόσταση εξάλειψης* είναι μία παράμετρος σε γραφήματα που μετράει την "απόσταση" των δισυνεκτικών συνιστωσών ενός γραφήματος από μία δεδομένη κλάση γραφημάτων. Όταν η υποδεικνυόμενη κλάση γραφημάτων είναι η κλάση των γραφημάτων χωρίς ακμές, χρησιμοποιούμε τον όρο *δισυνεκτικό δεντροβάθος* για να περιγράψουμε την παράμετρο. Στην παρούσα διπλωματική, αποδεικνύουμε ότι το δισυνεκτικό δεντροβάθος είναι ισοδύναμο με το ελάχιστο πλήθος χρωμάτων ενός χρωματισμού που αποδίδει κορυφή μοναδικού χρώματος σε κάθε δισυνεκτικό υπογράφημα του γραφήματος. Επιπροσθέτως, εισάγουμε μια ειδική περίπτωση ενός μη έγκυρου ακμοχρωματισμού που μπορεί να λειτουργήσει ως ένας εναλλακτικός ορισμός του δισυνεκτικού δεντροβάθους, ο οποίος καλείται *κατάταξη ακμών κύκλου*. Έπειτα, συνδέουμε το δισυνεκτικό δεντροβάθος με παίγνια αναζήτησης σε γραφήματα, εισάγοντας δύο παραλλαγές του παίγνιου κλέφτες και αστυνόμοι οι οποίες μπορούν να χρησιμοποιηθούν για τον υπολογισμό του δισυνεκτικού δεντροβάθους ενός γραφήματος. Τέλος, αποδεικνύουμε ότι το δισυνεκτικό δεντροβάθος επιδέχεται πολυωνυμικό πυρήνα παραμετρικοποιημένο ως προς το μέγεθος του καλύμματος κορυφών.

ACKNOWLEDGEMENTS

First, I wish to express my gratitude to assistant professor Archontia C. Giannopoulou for her precious guidance and for her support as the supervisor of this thesis. I would also like to sincerely thank professor Dimitrios M. Thilikos for his crucial contribution to the thesis and professor Stavros G. Kolliopoulos for offering his time as a member of the examination committee.

There are many friends that I would like to thank from the academia. Christoforos, for knowing the right moment to offer a warm smile. Danai, for sharing common paths for the past eight years. John, for being a bright light in the darkness. Yung, for our past and future meaningful conversations.

Lastly, there are many people from outside the academia to whom I owe my gratitude. I would like to thank all my friends for their emotional support, my parents, Grigoris and Vivian, for their unconditional love, and Glykeria for being so kind and thoughtful.

CONTENTS

CHAPTER 1

INTRODUCTION

1.1 In General

A popular concept in graph theory is asking for the minimum amount of modifications needed for a given graph to reach a certain state. The number of modifications are usually described as *distance parameters*. Distance parameters on graphs serve as a measure of how far a graph is from a given graph class $\mathcal{C}$. This roughly means that a distance parameter checks how many removals (usually vertex removals) one would have to apply on a graph in order for the resulting graph to belong to a certain class. One of the most basic and simple distance parameters is the *vertex deletion distance*, whose importance originates from the fact that for a variety of graph classes, vertex deletion distance describes well-studied NP-complete problem, such as, the VERTEX COVER problem. A more general distance parameter called *elimination distance* was proposed by Bulian and Dawar in 2016 ([2]) while studying the GRAPH ISOMORPHISM problem. The two parameters are quite similar since they both only involve vertex deletions. However, elimination distance is a parameter that is applied simultaneously on every connected component of a graph. A special case of elimination distance called *treedepth* has been around for a much longer time. Treedepth in a sense measures how close a graph is to the empty graph if we are allowed to delete vertices from multiple connected components in every step and it goes with many different names (e.g. *vertex rankings* [1], *centered colorings* [12]). Regarding the computational complexity of the corresponding decision problem of treedepth, it has been proven that, given an integer k, it is NP-complete to decide whether the treedepth of a graph is equal to k ([13]) and is also fixed parameter tractable when parameterized by k [10].

1.2 In this thesis

In 2021, Diner, Giannopoulou, Stamoulis and Thilikos [5] proposed an even more flexible distance parameter called *block elimination distance*. As the name suggests, here the vertex deletions are applied on the two-connected components of the graph. In the third chapter, we introduce *block treedepth* as the analogous of treedepth for block elimination distance. Moreover, we prove that block treedepth is a parameter which is equivalent to 2-connected centered colorings. Moving on, we introduce an edge coloring for graphs called *cycle edge ranking* and, again, we prove its equivalence to block treedepth. Furthermore, we connect block treedepth with two versions of the cops and robbers game, the Searcher-Stationary and the LIFO-Search. Finally, the fourth chapter of this thesis is dedicated to reduction rules that will lead to a proof that the corresponding decision problem of block treedepth admits a cubic kernel when the vertex cover number of the graph is the parameter, thus coming to the conclusion that BLOCK TREEDEPTH is in FPT when parameterized by VERTEX COVER.

CHAPTER 2

PRELIMINARIES

In this chapter we introduce all the notions, definitions and concepts that will be needed throughout the thesis.

2.1 Set Theory

We start with some basic concepts. We use $\mathbb{N}$ to represent the set of natural numbers and $\mathbb{Z}$ respectively for the set of integers. By $[k]$ we denote the set of all integers that are greater or equal than 1 and less or equal than k. For a family of sets $\mathcal{A}$, $\bigcup_{A\in\mathcal{A}} A$ denotes the union of all elements that are part of at least one set in $\mathcal{A}$. For a given alphabet Σ, we define as Σ^* the set of all finite length strings that can be generated by arbitrarily concatenating elements of L. If S is a set, then $\binom{S}{2}$ denotes the set of all subsets of S that contain exactly two elements. A binary relation R on a set S is a subset of $S \times S$. If $(a, b) \in R$ we write aRb. A relation R is called transitive if for every $a, b, c \in S$ when aRb and bRc are true, aRc also holds. A relation R is called antisymmetric if for every $a, b \in R$, if both aRb and bRa are true, then $a = b$ holds. A partially ordered set (S, R) is a set S along with an antisymmetric and transitive relation R on the elements of S. A chain C of S is a subset of S such that for every two elements $a, b \in C$, either aRb or bRa holds. An antichain C of S is a subset of S such that for every elements $a, b \in C$ neither aRb nor bRa holds. A non-trivial antichain is an antichain that is not an empty set or a singleton.

2.2 Graph Theory

Here, we state basic concepts on graphs. We borrow some terminology from [3]. All graphs in this thesis are considered to be finite, undirected and simple (no loops or multiple edges) unless stated otherwise. The set of all such graphs is denoted by $\mathcal{G}$. For a graph G, we use $V(G)$ to denote the set of vertices of G and $E(G)$ to denote the set of edges of G. If u and v are the endpoints of an edge e, then we say that $e = \{u, v\}$ or $e = uv$. We say that H is a *subgraph* of G if $V(H) \subseteq V(G)$ and $E(H) \subseteq E(G)$. Also, for a set $S \subseteq V(G)$, the *induced subgraph* $G[S]$ is the graph that has S as the vertex set and its edge set contains exactly all edges of $E(G)$ with both endpoints in S. A *path* of

G is a sequence of vertices $(v_1, \dots, v_k)$ such that $v_i v_{i+1} \in E(G)$ for every $i \in [k-1]$. A cycle of a graph G is a sequence of vertices $(v_1, \dots, v_k)$ such that $v_i v_{i+1} \in E(G)$ for every $i \in [k-1]$ and also $v_k v_1 \in E(G)$. A graph will be called *connected* if for any two vertices in $V(G)$ there exists a path connecting them. A graph G is a *forest* if it contains no cycle, and G is a *tree* if it is a connected forest. The distance between two vertices u, v is a function $\mathsf{dist} : V(G) \times V(G) \to \mathbb{N}$ which is equal to one less than the number of vertices on the shortest path connecting u with v and is denoted $\mathsf{dist}(u, v)$. The *neighborhood* of a vertex $u \in V(G)$ is denoted as $N_G(u) = \{v \in V(G) \mid \{u, v\} \in E(G)\}$. For a subset $S \subseteq V(G)$, we set $N_G(S) = \bigcup\{N_G(u) \mid u \in V(G)\}$. With $G \setminus u$ we denote the induced subgraph of $V(G) \setminus \{u\}$, while for an edge $e \in E(G)$ we denote $G \setminus e = (V(G), E(G) \setminus \{e\})$. A *cut-vertex* is a vertex $u \in V(G)$ such that $G \setminus u$ has more connected components than G. A graph G is called *biconnected* if it has more than two vertices and it has no cut-vertex. A *block graph* is a graph that is either biconnected, a vertex or an edge. A *block* of a graph G is a maximal biconnected component of G. The *degree* of a vertex is equal to $\mathsf{deg}(u) = |N_G(u)|$. A vertex is called *isolated* if $\mathsf{deg}(u) = 0$ and *pendant* if $\mathsf{deg}(u) = 1$. A *leaf* is a vertex u of a tree that has $\mathsf{deg}(u) = 1$. An acyclic graph $T = (V(T), E(T), r)$ is called *rooted tree*, where $r \in V(T)$ is the *root*. For vertices $t_1, t_2 \in V(T)$, we say that t_1 is an *ancestor* of t_2 if t_1 belongs to the unique path from r to t_2. The vertex t_1 is a *descendant* of t_2 if t_1 belongs to a path from t_2 to any leaf of T other than the root. If t_1 is an ancestor of t_2 we write $t_1 \leq_{T,r} t_2$ The *depth* of a rooted tree is equal to the maximum size of a path from the root to a leaf. A *rooted forest* is a graph whose connected components are rooted trees. A *bridge* is an edge such that $G \setminus e$ has more connected components than G. A *graph property* of a graph is a binary function $\mathsf{p} : \mathcal{G} \to \{0, 1\}$.We say that a graph G has a property p if $\mathsf{p}(G) = 1$ and we say that G does not have this property otherwise. A class of graphs $\mathcal{G}_{\mathsf{p}}$ is the set of all graphs that have a property p.

Definition 2.1. A distance parameter of a graph G is a function $\mathsf{f} : \mathcal{G} \to [k]$, for some $k \in \mathbb{N}$.

Definition 2.2. A *(proper) coloring* of a graph G is a function $\mathsf{f} : V(G) \to [k]$ for some $k \in \mathbb{N}$ such that, if $\mathsf{f}(u) = \mathsf{f}(v)$ then $uv \notin E(G)$. For a subgraph $H \subseteq_{sb} G$, we say that $\mathsf{f}|_H : V(H) \to [k]$ is the *restriction* of f with respect to H. The *size* of a coloring is defined to be equal to $|\mathsf{Im}(\mathsf{f})|$.

Definition 2.3. An *edge k-coloring* of a graph G is a function $\mathsf{f} : E(G) \to [k]$ for some $k \in \mathbb{N}$.

2.3 Parameterized Complexity and Algorithms

Parameterized complexity is a recent field of complexity theory that focuses on tracing parameters that would make a problem easier to cope with when these parameters are fixed. There are many well studied algorithmic problems that are NP-hard. This makes them require too much time in order to find solutions. Towards an attempt to further classification and better understanding of NP-hard problems, we focus on *fixed parameter tractability* (FPT). This basically is the class that contains all problems that can be solved in polynomial time when we consider suitable parameters as part of the input and are exponential only towards the fixed parameters. We borrow the following terminology and notations from [4].

Definition 2.4. A *parameterized problem* is a language $L \subseteq \Sigma^* \times \mathbb{N}$, where Σ is a finite alphabet. For an instance $(x, k) \in \Sigma^* \times \mathbb{N}$, k is called the *parameter*. The *size* of an instance (x, k) is equal to $|x| + k$.

In this thesis, the input x will be a graph G.

Definition 2.5. A parameterized problem L is called *fixed parameter tractable* (FPT) if there exists an algorithm $\mathcal{A}$, a computable function $\mathsf{f} : \mathbb{N} \to \mathbb{N}$ and a constant c such that, given $(x.k) \in \Sigma^* \times \mathbb{N}$, the algorithm $\mathcal{A}$ correctly decides whether $(x, k) \in L$ in time bounded by $\mathsf{f}(k) \cdot |(x, k)|^c$. The complexity class containing all fixed parameter tractable problems is denoted as FPT.

Definition 2.6. A *reduction rule* is a function $\phi : \Sigma^* \times \mathbb{N} \to \Sigma^* \times \mathbb{N}$ that maps an instance (G, k) of a problem L to an equivalent instance (G', k') of L such that ϕ is computable in polynomial time with respect to $|I|$ and k. We say that two instances are *equivalent* when $(G, k) \in L$ if and only if $(G', k') \in L$. A reduction rule is called *safe* if it produces equivalent instances of a problem.

Definition 2.7. A *kernelization algorithm* or a *kernel* for a parameterized problem L is an algorithm $\mathcal{A}$ that, given an instance (G, k) of L, works in polynomial time and returns an equivalent instance (G', k') of L. Moreover, it is required that there exists a computable function g such that whenever (G', k') is the output of an instance (G, k), we have that $|G'| + k' \leq \mathsf{g}(k)$.

Theorem 2.8. [4] A decidable parameterized problem L is in FPT if and only if L admits a kernelization algorithm.

CHAPTER 3

DISTANCE PARAMETERS

In this chapter we describe *distance parameters* for graphs. We begin with *vertex deletion distance*, then move on to *elimination distance* and finally we introduce *block elimination distance*. The order in which they are introduced is of importance, since each measure can and will be described as a relaxed alternative of the previous one.

3.1 Vertex Deletion Distance

A natural question one could ask for graphs is what is the least number of vertices that should be removed in order for a graph to obtain a certain property. This exactly is the concept of vertex deletion distance. A formal definition is the following:

Definition 3.1. The vertex deletion distance of a graph G from a target class $\mathcal{C}$ is defined as

$$\mathsf{vd}_{\mathcal{C}}(G) = \begin{cases} 0 & \text{if } G \in \mathcal{C} \\ 1 + \min\{\mathsf{vd}_{\mathcal{C}}(G \setminus v) \mid v \in V(G)\} & \text{if } G \notin \mathcal{C} \end{cases}.$$

It is in our interests to explore the nature of vertex deletion distance and what it represents when we indicate certain graph classes as the target classes in vertex deletion distance. First, we focus on the simplest graph class which is the class of edgeless graphs, denoted by $\mathcal{E}$. By focusing on $\mathsf{vd}_{\mathcal{E}}$, we notice that the parameter is equal to the number of vertices one needs to delete from a graph in order for the resulting graph to be edgeless. For someone who is familiar with the basic concepts of graph theory, it is easy to observe that this question is equivalent to the following famous NP-hard problem:

VERTEX COVER
Input: A graph G and an integer k.
Question: Is there a set $S \subseteq V(G)$ such that every edge has at least one end-point in S and $\|S\| \leq k$?

We define the vertex cover number of a graph G, $\mathsf{vc}(G)$, to be equal to the minimum integer k for which the VERTEX COVER problem has a positive answer and we claim that it is equal to $\mathsf{vd}_{\mathcal{E}}(G)$.

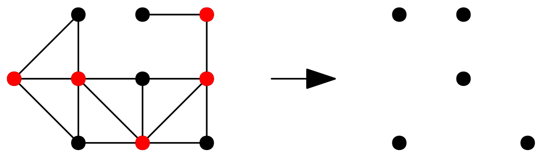

Figure 3.1: An example of an optimal vertex cover.

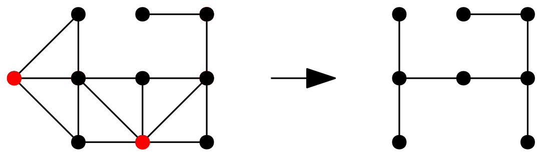

Figure 3.2: An example of an optimal feedback vertex set.

Claim 3.2. The vertex cover number of a graph G is equal to $\mathsf{vd}_{\mathcal{E}}(G)$.

Proof. Let S be a vertex cover of minimum size for G. If we remove every vertex of S from G in an arbitrary order, the resulting graph $G \setminus S$ is an induced subgraph of G and since every edge of G has at least one end-point in S, $G \setminus S$ is edgeless. Thus, $\mathsf{vc}(G) \geq \mathsf{vd}_{\mathcal{E}}(G)$. On the other hand, if one can delete $\mathsf{vd}_{\mathcal{E}}(G)$ vertices from G and eliminate all edges, then there certainly exists a subset $S \subseteq V(G)$ of size $|S| \leq \mathsf{vd}_{\mathcal{E}}(G)$ that is a vertex cover for G. Thus, $\mathsf{vc}(G) \leq |S|$. All-together, we get that $\mathsf{vc}(G) = \mathsf{vd}_{\mathcal{E}}(G)$ □

A similar situation arises when the indicated class is defined as the class of forest graphs, denoted by $\mathcal{F}$. Here, $\mathsf{vd}_{\mathcal{F}}$ represents the minimum number of vertices needed to be removed in order for the resulting graph to be acyclic. The equivalent problem this time is called FEEDBACK VERTEX SET (see Figure 3.2), another well studied NP-hard problem.

FEEDBACK VERTEX SET
Input: A graph G and an integer k.
Question: Is there a set $S \subseteq V(G)$ such that every cycle of G has at least one vertex in S and $|S| \leq k$?

Claim 3.3. The size of an optimal feedback vertex set for a graph G is equal to $\mathsf{vd}_{\mathcal{F}}$.

Proof. Let $\mathsf{fvs}(G)$ be the size of a minimum sized feedback vertex set for G and let S be a feedback vertex set of minimum size for the graph. Obviously, the removal of every vertex in S results in $G \setminus S$. Every cycle in G had a vertex removed. Thus, $G \setminus S \in \mathcal{F}$ or equivalently, $\mathsf{fvs}(G) \geq \mathsf{vd}_{\mathcal{F}}(G)$. Now, there exists a subset of $V(G)$ of size $\mathsf{vd}_{\mathcal{F}}(G)$ from G whose removal breaks every cycle in G. This subset is not necessarily of minimum size. Hence, $\mathsf{fvs}(G) \leq \mathsf{vd}_{\mathcal{F}}(G)$ which concludes the proof. □

Of course, the list of problems that can be represented as a special case of vertex deletion distance does not end here. In fact, for every class $\mathcal{C}$ the vertex deletion distance to this class gives birth to a different graph-algorithmic problem, many of which problems are NP-hard.

3.2 Elimination Distance

Sometimes, there exist algorithmic problems for graphs whose questions would be more meaningful if we could focus on a certain connected component of the graph and not the whole graph. This gives rise to the following question: is there a distance parameter for graphs which would recursively focus on the connected components of the graph? The concept of elimination distance describes exactly this idea and is defined as the following recursive function:

Definition 3.4. [2] Let G be a graph and let $\mathcal{C}$ be a class of graphs. The elimination distance of G from $\mathcal{C}$ is equal to

$$\mathsf{ed}_{\mathcal{C}}(G) = \begin{cases} 0 & \text{if } G \in \mathcal{C} \\ \max\{\mathsf{ed}_{\mathcal{C}}(C) \mid C \in \mathsf{cc}(G)\} & \text{if } G \notin \mathcal{C} \text{ and } G \text{ not connected} \\ 1 + \min\{\mathsf{ed}_{\mathcal{C}}(G \setminus v) \mid v \in V(G)\} & \text{if } G \notin \mathcal{C} \text{ and } G \text{ is connected} \end{cases}$$

where $\mathsf{cc}(G)$ is the set of all maximal connected components of G.

3.2.1 Treedepth

Elimination distance was first introduced by Bulian and Dawar in 2016 as a tool to show that the GRAPH ISOMORPHISM problem is fixed parameter tractable. However, this concept was not entirely unknown until then. Consider the case in which the target class is defined as the class of edgeless graphs, $\mathcal{E}$. In this certain case, we use the term *treedepth* in order to describe $\mathsf{ed}_{\mathcal{E}}(G)$.

Definition 3.5. [11] Let G be a graph. The treedepth of G is denoted as $\mathsf{td}(G)$ and is defined as

$$\mathsf{td}(G) = \begin{cases} 0 & \text{if } E(G) = \emptyset \\ \max\{\mathsf{td}(C) \mid C \in \mathsf{cc}(G)\} & \text{if } E(G) \neq \emptyset \text{ and } G \text{ not connected} \\ 1 + \min\{\mathsf{td}(G \setminus v) \mid v \in V(G)\} & \text{if } E(G) \neq \emptyset \text{ and } G \text{ is connected} \end{cases}$$

Treedepth is a notion that has been rediscovered many times and appears with different names and definitions throughout the literature. It is worth mentioning that in the original definition the treedepth of an edgeless graph is defined to be equal to 1 instead of 0, but this is just a technical discrepancy that does not create any inconsistency. We move on to a definition that appears in [11] for the first time and will help us define treedepth in a alternate way.

Definition 3.6. [11] The *closure* of a rooted forest F, $\mathsf{clos}(F)$, is a graph F' such that $V(F') = V(F)$ and $E(F') = \{xy \mid x \text{ is an ancestor of } \}y$ and $x \neq y$. The treedepth of a graph G is the minimum height of a rooted forest F such that $G \subseteq_{sb} \mathsf{clos}(F)$.

Definition 3.6 is useful, because it will be used to define a tree-like decomposition of graphs based on the order that the treedepth recursion picks the deleted vertices in every step.

Definition 3.7. A *treedepth decomposition* of a graph G is a rooted forest F whose vertices is exactly $V(G)$ and if $uv \in E(G)$, then u is either an ancestor or a descendant of v in F. The treedepth of G is equal to the minimum height of a treedepth decomposition reduced by one.

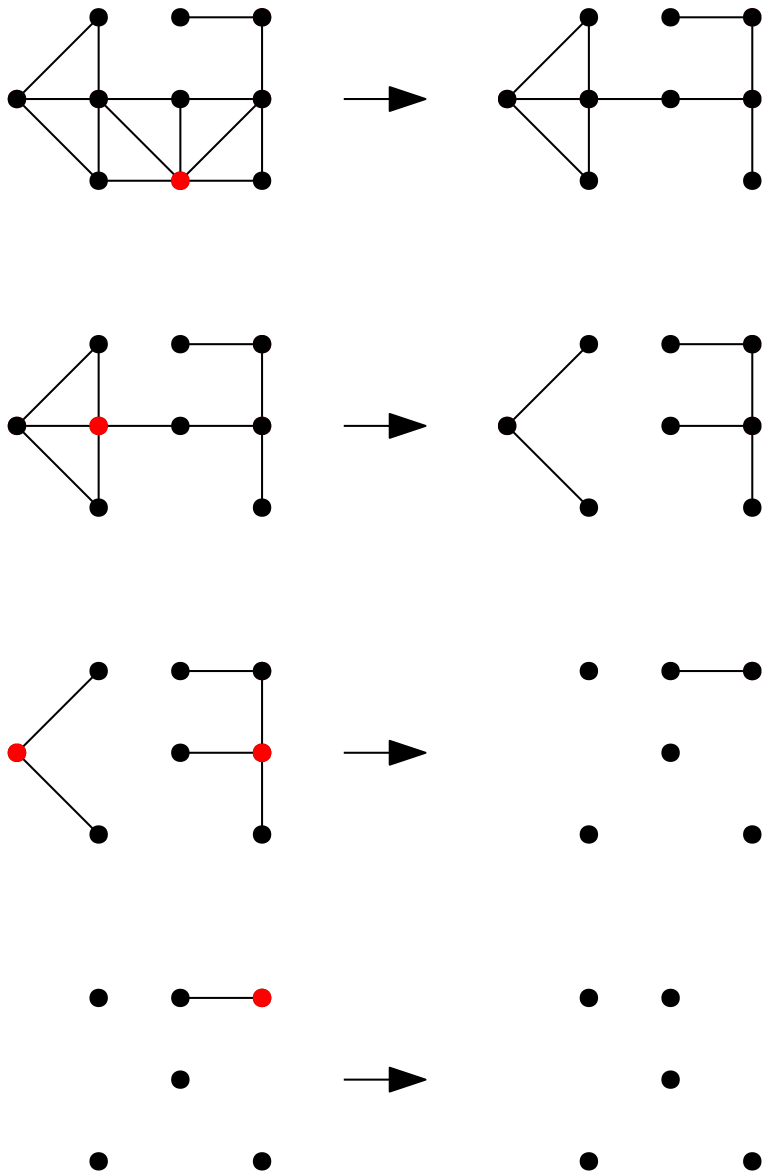

Figure 3.3: A graph whose treedepth is equal to 4. Here, we see an optimal recursive choice of the deleted vertices.

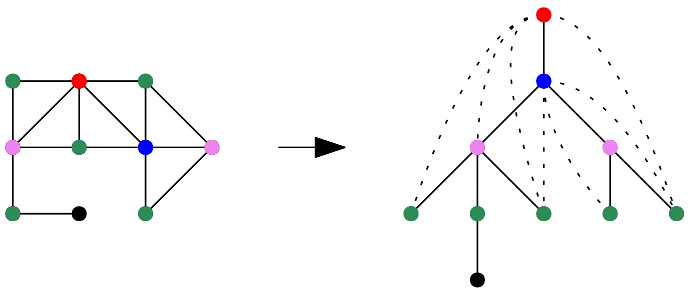

Figure 3.4: An example of a treedepth decomposition of minimum depth. If we match every color of the decomposition to a number in a rising order from top to bottom (red= 1, blue= 2 etc.), we see that the coloring is also a vertex ranking.

The basic idea in order to obtain a treedepth decomposition of a graph G is exactly the following recursive process. If G is connected and has exactly one vertex, then $F = G$ and the root is exactly the unique vertex of G. If G is connected and has more than one vertices, then pick a vertex $u \in V(G)$ as the root. Then, repeat the process for every connected components of $G \setminus u$ and connect u with the root of every decomposition of the corresponding connected components of $G \setminus u$. Finally, if G is not connected then the treedepth decomposition is the graph that is acquired by the union of the treedepth decompositions of every connected component separately. An example can be found in Figure 3.4.

An interesting observation about treedepth is that it is always upper bounded by vertex cover number. This is due to the fact that treedepth is a special case of elimination distance while vertex cover number originates from vertex deletion distance. In the first case, vertices are picked to be deleted from the connected components of the graph while in the second case one cannot delete more than one vertex in one step. What is more, for every non-negative integers c, k, there exists a graph G such that $\mathsf{td}(G) \leq k$ and $\mathsf{vc}(G) > c \cdot k$. For example, consider the clique graph K_{k+1} and let G be a graph that has c copies of K_{k+1}. Obviously, the treedepth of this graph is equal to k while its vertex cover number is equal to $c \cdot k$. Moreover, this can be generalized for elimination distance and vertex deletion distance. When the indicated classes are the same, elimination distance is always upper bounded by vertex deletion distance and there is no function of elimination distance which upper bounds vertex deletion distance.

3.2.2 Centered Colorings

Vertex colorings can usually be interpreted as a vertex labelling serving a certain purpose. In terms of treedepth, this purpose can be described through a special kind of colorings called centered colorings.

Definition 3.8. A *centered coloring* $\phi : V(G) \to \mathbb{N}$ of a graph G is a coloring such that for every connected subgraph H, there exists a color $c \in \mathbb{N}$ such that $\phi|_H^{-1}(c)$ is a singleton. A vertex of unique color in H is called *center* of H. The minimum size of a centered coloring for G is denoted $\chi_c(G)$.

The importance of centered colorings is that, in a way, they describe the order in which vertices are picked to be deleted when calculating treedepth. The relation between the treedepth of a graph G and the minimum size of a centered coloring for G can be better comprehended through the proof of the following theorem.

Theorem 3.9. [11] For every graph G, it holds that $\mathsf{td}(G) = \chi_c(G) - 1$.

Proof. Definition 3.7 states that for every graph G, the minimum height of a treedepth decomposition T among every treedepth decompositions for G is equal to $\mathsf{td}(G) + 1$. Thus, it suffices to prove that the height of a treedepth decomposition of minimum height is equal to the size of an optimal centered coloring for G. Also, notice that neither the depth of a treedepth decomposition nor χ_c is affected by the amount of connected components of G. Thus, in what follows we make the assumption that G is connected.

First, we prove that $\chi_c(G) \geq \mathsf{depth}(T)$, where $\mathsf{depth}(T)$ is the depth of T. We will create a new treedepth decomposition T' using the following process. Since G is connected, it has at least one center. Arbitrarily pick a center u of G as the root of T' and remove the center from G. Then, arbitrarily pick a center from every connected component of $G \setminus u$, add all centers in T' and connect them to the root. Repeat the process for every connected component that pops up after the removal of the centers that were picked previously and . Obviously, the process will terminate in $\chi_c(G)$ steps. We claim that T' is indeed a treedepth decomposition for G. Let $uv \in E(G)$ be an edge of G. It is clear that u and v cannot be picked at the same stage of the process described above as they can not have the same color. This holds because because they are connected with an edge and thus they cannot be separated unless one vertex of the edge is removed. Suppose that u, without loss of generality, is picked first. Then v will continue to be a vertex in a connected component acquired after the removal of u. Thus v will be a descendant of u in T'. Hence, T' is a treedepth decomposition for G of height $\chi_c(G)$. Since T' is not necessarily of optimal height, we get that $\chi_c(G) \geq \mathsf{depth}(T)$.

Now, we need to prove that $\chi_c(G) \leq \mathsf{depth}(T)$. Given the optimal treedepth decomposition T for G, we will construct a centered coloring using at most $\mathsf{depth}(T)$ colors. Let u be the root of T. We claim that the coloring $\phi(v) = \mathsf{dist}(u, v) + 1$ is a centered coloring for G. We need to prove that every connected subgraph $H \subseteq_{sb} G$ has a center via ϕ. Let, towards a contradiction, H be a connected subgraph that has no center when we restrict ϕ on its vertices. This means that for every color that appears in H there exist at least two vertices that have this color. Let v_1 and v_2 be two vertices of minimum color. Since H is connected, there exists a path from v_1 to v_2. This path, p, can be written as $v_1, w_1, \ldots, w_j, v_2$ for some $j \in \mathbb{N}$. For every vertex w on this path, it holds that $\phi(w) > \phi(v_2) = \phi(v_1)$. From the definition of a treedepth decomposition, v_1 is either an ancestor or a descendant of w_1 and since v_1 is of minimum color in H, we get that w_1 is a descendant of v_1. Now, for every integer $k \in [j]$ we have that w_j is either an ancestor or a descendant of w_{j+1}. Thus, every vertex on p has a descendant or ancestor relation with v_1 and since v_1 is of minimum color, we get that v_2 is also a descendant of v_1. However, this is a contradiction because $\phi(v_2) = \phi(v_1)$ which implies that $\mathsf{dist}_T(v_1, u) = \mathsf{dist}_T(v_2, u)$. Thus, H has a center. Because of the fact that ϕ is not necessarily of minimum size, we have that $\chi_c(G) \leq \mathsf{depth}(T)$. Altogether, $\chi_c(G) = \mathsf{depth}(T) = \mathsf{td}(G) + 1$. □

Besides the obvious, the proof of Theorem 3.9 states something stronger than just the equivalence of treedepth and centered colorings. Since graph colorings can be interpreted as labelling of vertices, when we mention colors we could also refer to numbers representing the colors. During the transition from a treedepth decomposition of G to a centered coloring, notice that the center of every connected subgraph is in fact the vertex whose color is the smallest among every color in the subgraph. This means that treedepth actually is equivalent to a certain kind of colorings that ensure that in every connected subgraph, the smallest color is unique. This concept is mentioned as *vertex rankings* in the literature ([1]) and is in fact a concept that is prior to treedepth. Figure 3.4 offers an example of a centered coloring and a vertex ranking, as well as an example of the transition from a centered coloring to a treedepth decomposition and vise versa.

3.2.3 A Searching Game for Treedepth

Searching games ([6]) are often used to describe a situation during which a fugitive is trying to avoid a set of searchers. In graph theory, the pursuit takes place on a graph G where $V(G)$ represents a set of rooms while $E(G)$ represents a set of aisles that connect the rooms. Graph searching games are useful because they usually serve as characterisations for graph parameters while offering an intuitive perspective that can help better understand the parameter.

A famous variation of searching game is the *cops and robbers* game. In this variation, the fugitive is impersonated by a robber and the searchers can be interpreted as the cops. Even though the two opposing sides always remain the same, different rules give rise to alternate versions of the game. For example, there are variations in which the robber is visible and others where the robber is invisible. Also, in some versions the cops cannot be moved once they are placed on a vertex and in other they have the ability to move. Here, we will be focusing on two versions of the game.

The first version of the game is called Searcher-Stationary. In this version, the robber is visible and the cops cannot move once they are placed in the graph. The game starts with the robber picking a vertex as its initial position. Then, the cops are informed of the robbers position and proceed with picking the vertex that they are going to occupy. Once they made their decision, they announce the position that they picked and then the robber picks his next destination right before the cops occupy the selected vertex. The robber can move through vertices that are connected with edges and are not occupied by the cops, and can cover an infinite distance in a single round. In other words, the robber can move from one vertex to another in one round as long as there is a path connecting the two vertices that is not blocked by a vertex occupied by the cops. The process is repeated until the robber cannot move and the cops occupy the vertex on which the robber is standing on. When this happens, the robber is considered captured and the game is over. Consider that the robber is a genius and always acts in his best interest.

In the second version the robber is invisible and the cops are allowed to move in the graph in a Last-In-First-Out way. The name of this version is LIFO-Search. This means that a cop can be moved after every cop that entered the graph later, is pulled out from the graph. Here, the robber selects a vertex at the beginning of the game and can move on paths that are not intercepted by cops. The cops consider the robber to be omnipresent and thus have to search every vertex of the graph, while every other rule remains the same. The minimum amount of cops needed to capture a robber on a graph G in the Searcher-Stationary version is called *stationary cop number* of G, denoted by $\mathsf{vco}(G)$, while the minimum amount of cops needed to capture a robber on a graph G in the LIFO-Search version is called *lifo cop number* of G, and is denoted by $\mathsf{ico}(G)$ Our purpose is to relate the cop number of a graph with its treedepth. However, before we state a theorem from [7] that points the relation, we will first introduce a notion that is also equivalent to treedepth and has been used as a tool to offer a better insight to the cop and robber game that we described above.

Definition 3.10. A *shelter* of a graph G is a collection $\mathcal{S}$ of non-empty connected sets in G such that for every non-minimal set $S \in \mathcal{S}$ no vertex belongs to all its children, in other words,

$$\bigcap\{S' \mid S' \in M_{\mathcal{S}}(S)\} = \emptyset,$$

where $M_{\mathcal{S}}(S)$ is the $\subseteq$-maximal elements of $\{S' \in \mathcal{S} : S' \subset S\}$. The *thickness* of a shelter $\mathcal{S}$ is the minimal length of a maximal $\subseteq$-chain and is denoted by $\mathsf{th}(S)$.
We are now ready to state a theorem that relates treedepth with cops and robbers and shelters.

Theorem 3.11. [7] Let G be a graph and k be a non-negative integer. Then the following are equivalent.

- (i) $\mathsf{td}(G) \leq k$,
- (ii) for every shelter $\mathcal{S}$ of G, $\mathsf{th}(\mathcal{S}) \leq k+1$,
- (iii) $\mathsf{vco}(G) \leq k+1$,
- (iv) $\mathsf{ico}(G) \leq k+1$.

The two versions that we described can offer a good intuition when it comes to calculating the treedepth of whole graph classes in the general case. In order to give an example, we first define a graph G to be a *star* if there exists a vertex $u \in V(G)$ such that $uv \in E(G)$ for every other vertex in $V(G)$, and for any vertices $v_1, v_2 \neq u$ we have that $v_1v_2 \notin E(G)$. Here, we call u the center of the star. The LIFO-Search game played on a star graph S would require exactly 2 cops to cover the whole graph. The first one will be standing on the center of S the whole time, thus blocking any movement of the robber. Then the second cop will search every other vertex one by one until the robber is caught (see Figure 3.5). This means that for every star graph S, $\mathsf{ico}(S) \leq 2$ or equivalently $\mathsf{td}(S) \leq 1$.

A different example is the application of the Searcher-Stationary version applied on the path with k vertices, denoted by P_k. We will construct a strategy for the cops in the Searcher-Stationary version applied on path graphs in order to calculate the stationary cop number of P_k, which will help us prove that the treedepth of P_k is equal to $\lfloor \log_2(k) \rfloor$. The following claim is clearly equivalent to the previous statement.

Claim 3.12. For every $k \geq 0$, $\mathsf{vco}(P_{2^k}) = k+1$.

Proof. We will prove the claim by induction on k. For the base case, $P_{2^0} = P_1$ is a path graph on one vertex. Thus, the robber only has one option and cannot move and indeed $\mathsf{vco}(P_1) = 1$. Let the claim be true for every $k' < k$. We will prove that the claim is also true for k. Consider that we place a cop on the middle vertex of P_{2^k}. Thus, now the robber is restricted to move on a path of length 2^{k-1} and by induction hypothesis, from then on there are k cops needed to capture it. This proves that $\mathsf{vco}(P_{2^k}) \leq k+1$. Now, We will prove that $\mathsf{vco}(P_{2^k}) \geq k+1$ by constructing a shelter $\mathcal{S}$ such that $\mathsf{th}(\mathcal{S}) \geq k+1$. We define the root of $\mathcal{S}$ to be the whole path. Then we split P_{2^k} in half, creating two $P_{2^{k-1}}$ subgraphs and making these subgraphs the children of the original path. We continue this process with every leaf of $\mathcal{S}$ until all the resulting paths consist of a single vertex. The process will be repeated $\log(2^k) = k$. Every step extends the length of every maximal chain by exactly 1. Thus, $\mathsf{vco}(P_{2^k}) \geq k+1$, which proves that $\mathsf{vco}(P_{2^k}) \geq k+1$. Altogether, $\mathsf{vco}(P_{2^k}) = k+1$. $\square$

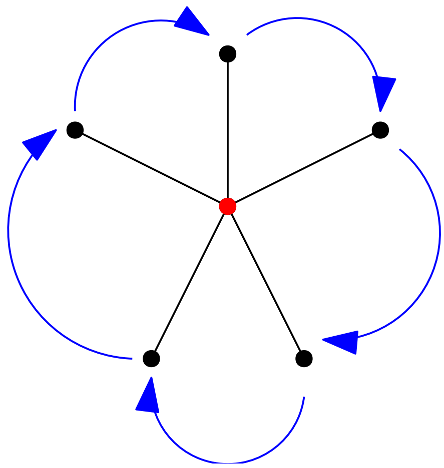

Figure 3.5: A star graph and a demonstration of how the cops would capture an invisible robber in a LIFO-Search game. The red vertex represents the first cop that is placed in the graph while the blue arrows represent the movements of the second cop.

3.3 Block Elimination Distance

In this section, we study a more flexible graph distance parameter called *block elimination distance*. Block elimination distance is quite similar to elimination distance. The idea is that instead of applying the recursion on connected components, we apply it on the biconnected components of the graph instead. Formally, block elimination distance is described as follows:

Definition 3.13. The block elimination distance of a graph G from a class $\mathcal{C}$ is defined as

$$\mathsf{bed}_{\mathcal{C}}(G) = \begin{cases} 0 & \text{if } G \in \mathcal{C} \\ \max\{\mathsf{bed}_{\mathcal{C}}(B) \mid B \in \mathsf{bc}(G)\} & \text{if } G \notin \mathcal{C} \text{ and } G \text{ is not biconnected} \\ 1 + \min\{\mathsf{bed}_{\mathcal{C}}(G \setminus v \mid v \in V(G))\} & \text{if } G \notin \mathcal{C} \text{ and } G \text{ is biconnected} \end{cases}$$

where $\mathsf{bc}(G)$ denotes the biconnected components of G.

A forest-like representation of the order in which the vertices are deleted during the calculation of the block elimination distance of graph can be quite a useful tool for proofs. Again, we borrow the following definition from [5].

Definition 3.14. Let $\mathcal{C}$ be a non-trivial hereditary class of graphs and let G be a graph. Let (F, R, τ) be a triple consisting of a rooted forest F, its set of roots R and a function $\tau : V(G) \to 2^{V(F)}$. Also, given a vertex $t \in V(F)$, we define $\mathsf{d}_{F,R}(t) = \{t' \in V(F) \mid t \leq_{F,R} t'\}$ as the set of descendants of t in F. Moreover, for every $S \subseteq V(F)$, we define $\tau^{-1}(S) = \{v \in V(G) \mid \tau(v) \cap S \neq \emptyset\}$ and for every $t \in V(F)$ we define $G_t = G[\tau^{-1}\mathsf{d}_{F,R}(t)]$. We say that (F, R, τ) is a *$\mathcal{C}$-block tree layout* of G if the following hold:

1. for every $v \in V(G)$, $\tau(v)$ is an (F, R)-antichain,
2. for every $t \in V(F)$, G_t is a block-graph,

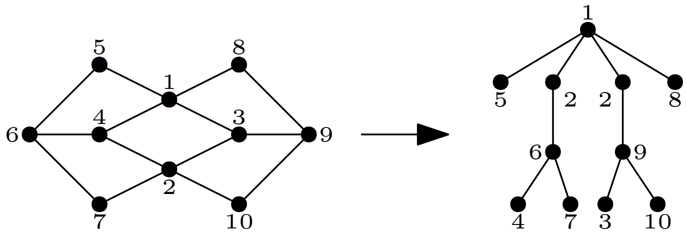

Figure 3.6: A $\mathcal{C}$-block tree layout where $\mathcal{C} = \mathcal{E}$.

3. for every $t \notin L(F, R)$, $|\tau^{-1}(\{t\})| = 1$ and $G_t \notin \mathcal{C}$,
4. for every $t \in L(F, R)$, $G_t \in \mathcal{C}$
5. for every non-trivial (F, R)-antichain C, the graph $\bigcup\{G_t \mid t \in C\}$ is not biconnected.

The depth of a $\mathcal{C}$-block tree layout (F, R, τ) is equal to the depth of the rooted forest (F, R).

In a way, $\mathcal{C}$-block tree layouts are the analogous of treedepth decompositions for block elimination distance. There is a relation between the order in which vertices are deleted from a graph during calculating its block elimination distance from a given class and the depth of the image of each vertex via τ in the tree layout. However, the relation is not as obvious as in treedepth decompositions. For example, in Figure 3.6 we see that a vertex of a graph G can be mapped to more than one vertex in its block tree layout. What is more, the vertices of a block tree layout that are the image of a vertex in G do not have to be in the same depth. The only restriction is that they have to form an anti-chain, meaning that there are no two vertices in a block tree layout that are the image of the same vertex from G and have a descendant-ancestor relation. The following theorem summarizes the usefulness of block tree layouts.

Theorem 3.15. [5] Let $\mathcal{C}$ be a non-trivial hereditary class and let G be a graph. Then the minimum depth of a $\mathcal{C}$-block tree layout of G is equal to $\mathsf{bed}_{\mathcal{C}}(G) - 1$.

3.3.1 Block Treedepth and Block-Centered Colorings

Here we focus on the special case that $\mathcal{C}$ is the class of edgeless graphs. In this case, we get a graph measure which is similar to treedepth but instead applied on the biconnected components of the graph, which we call *block treedepth*. We use the notation $\mathsf{td}_2(G)$ to describe $\mathsf{bed}_{\mathcal{E}}(G)$. Our purpose is to find equivalent coloring definitions for block treedepth that are analogous to those of treedepth. The following definition traces back to [8], although it is mentioned as 2-connected centered coloring. What is interesting about it, is that block-centered colorings can be traced in the literacy before block elimination distance was firstly mentioned although the first is a special case of the second.

Definition 3.16. A *block-centered coloring* $\phi : V(G) \to \mathbb{N}$ of a graph G is a proper coloring such that for every biconnected subgraph $B \in \mathsf{bc}(G)$ of G there exists a color $c \in [k]$ satisfying $|\phi^{-1}|_{V(B)}(c)| = 1$. The minimum amount of colors needed to color a graph G block-centrally is called *block-centered coloring number* of G and is denoted as $\chi_{bc}(G)$.

We now proceed to prove the equivalence of the block-centered coloring number of a graph G and its block treedepth.

Proposition 3.17. For every non-negative integer k, $\chi_{bc}(G) - 1 \leq k$ if and only if $\mathsf{td}_2(G) \leq k$.

Proof. We will use induction on k. It is trivial to check that when $k = 0$ then $\chi_{bc}(G) - 1 = \mathsf{bed}(G) = 0$ (here, G is an edgeless graph).

We will prove that $\chi_{bc}(G) - 1 \leq k$ implies that $\mathsf{td}_2(G) \leq k$. Assume that $k \geq 1$, $\chi_{bc}(G) - 1 \leq k$ and let ϕ be a block-centered coloring for G using at most $k+1$ colors. We first consider the case in which G is biconnected. This means that there exists a vertex $v \in V(G)$ and a color $c \in [k+1]$ such that v is the only vertex in $V(G)$ satisfying $\phi(v) = c$. Then $H = G \setminus v$ has a block-centered coloring $\mathsf{h} = \phi|_{V(H)}$ that uses at most k colors. Thus $\chi_{bc}(H) \leq k$ and by induction hypothesis we get $\mathsf{td}_2(H) \leq k - 1$. By the definition of block elimination distance and since G is biconnected, it holds that $\mathsf{td}_2(G) \leq 1 + \mathsf{td}_2(H) \leq k$. Now assume that G is not biconnected. In this case, there exist vertices $v_1, \ldots, v_l \in V(G)$ witnessing the unique colors by restricting ϕ in every maximal biconnected component $\{B_1, \ldots, B_l\}$ of G respectively. By induction hypothesis applied on each $B_i \setminus v_i$, we get that $\mathsf{td}_2(B_i \setminus v_i) \leq k - 1$. Thus, $\mathsf{td}_2(G) = \mathsf{max}\{\mathsf{td}_2(B) \mid B \in \{B_1, \ldots, B_l\} \leq 1{+}\mathsf{max}\{\mathsf{td}_2(B) \mid B \in \{B_1 \backslash v_1, \ldots, B_l \backslash v_l\} \leq k$.

Finally, we will prove that $\mathsf{td}_2(G) \leq k$ implies that $\chi_{bc}(G) \leq k + 1$. In fact, we will use induction to prove an even stronger statement, that $\mathsf{td}_2(G) \leq k$ implies that $\chi_{bc}(G) \leq k + 1$ and given any vertex $u \in V(G)$ and any color $c \in [k + 1]$ there exists a coloring ϕ of size $k + 1$ such that $\phi(u) = c$. For $k = 0$, the case is trivial since G is edgeless and the only color in use is 1. Suppose that the statement is true for every integer $i \leq k$. We will prove that the statement is also true for $k + 1$ by using induction on the number of the blocks, l, of G. For $l = 1$, notice that G is a block graph, meaning that it either is a vertex, an edge, or it is biconnected. This means that there exists a vertex $v \in V(G)$ such that $\mathsf{td}_2(G \setminus v) \leq k$. By induction hypothesis, $G \setminus v$ has a block-centered coloring that uses at most k colors. We extend the coloring to G by assigning the color $k + 1$ on v. The new coloring is indeed an extension to a block-centered coloring for G. since every biconnected subgraph that does not contain v has a center from the original coloring and for every subgraph that contains v, it also serves as its center. Now, given a block-centered coloring ϕ of size $k + 1$, a color c and a vertex u, if $\phi(u) \neq c$ then we assign $\phi(u)$ to every vertex that is colored with c and assign c to every vertex that is colored with $\phi(u)$. The coloring obviously remains block-centered and has the original size. This proves the base case. Suppose that the statement holds for every integer $j \leq l$. We will prove that it also holds for $l + 1$. Given a graph G with $l + 1$ blocks, we calculate the block cut tree of G. The block cut tree of a graph is a graph T_G whose vertices are the blocks of G and whose edge set contains a pair of vertices if and only if the blocks that they represent intersect in G. Notice that T_G is a forest, since a cycle in T_G would imply the existence of a "cycle" of blocks in G which contradicts to the maximality of blocks. We arbitrarily pick any vertex $v \in V(T_G)$ to be the root of T_G and perform a breadth first search on the block cut tree. Let m be the last vertex that was visited during the search and let B_m be its corresponding block in G. Notice that m is a leaf. This means that B_m intersects with at most one other block in G. Let B'_m be the union of every other block of G other than B_m. Now, notice that B_m and B'_m intersect with at most one vertex. Both B_m and B'_m have less than $l + 1$

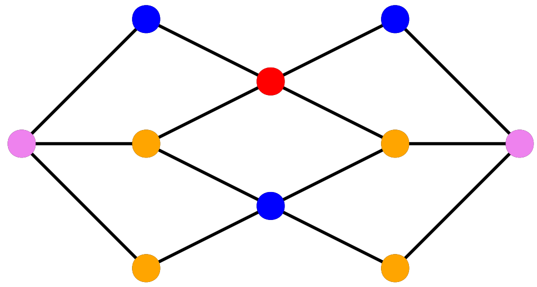

Figure 3.7: An example of a biconnected graph whose block-centered chromatic number is equal to 4.

blocks. By induction hypothesis, they also have block-centered colorings χ and χ' of size $k+1$ using the colors in $[k+1]$. If they do not intersect, then the union of the two colorings is obviously a block-centered coloring for G. Suppose that they intersect and that $V(B'_m) \cap V(B_m) = \{z\}$. Notice that, by induction hypothesis, we can rearrange χ in order to achieve $\chi(z)$ to be equal to $\chi'(z)$. We create a new coloring $\phi = \chi \cup \chi'$. This is possible since the two colorings agree on the colors of vertices that are in both graphs. Also, there is no biconnected subgraph of G that is both a subgraph of B_m and B'_m. Thus, ϕ is a block-centered coloring of G using $k+1$ colors. To conclude the proof, for any vertex $u \in V(G)$ and any color $i \in [k+1]$, if $\phi(u) \neq i$ then arbitrarily pick a color $j \in [k+1]$ and switch the color of every vertex that is colored with i to j and vise versa. The new coloring remains a block-centered coloring for G. Altogether, $\chi_{bc}(G) \leq k+1$. This concludes the proof. □

Combining Theorem 3.15 and Proposition 3.17 we get that the minimum depth of a block tree layout for a graph G is equal to $\chi_{bc}(G) - 1$. This implies a relation between the block-centered chromatic number of a graph and the depth of an optimal block tree layout. Nevertheless, unlike centered colorings, this connection does not occur by just assigning a different color to every level of a block tree layout, since the same vertex can be mapped in several levels. What is more, we saw that centered colorings can also be comprehended as a special case of vertex rankings through which the vertex of unique color (via a centered coloring) of a connected subgraph of a graph is also the vertex whose assigned color is the one with the smallest index. Unfortunately, this is not the case with block-centered colorings. This is due to the fact that, as we saw in the proof of Proposition 3.17, during the first stage of the construction of the coloring there might be adjacent vertices that share the same color. These vertices ought to be in different maximal biconnected components and in order to prevent this from happening we perform a permutation of the colors in one of the components. An example of such a permutation can be found in Figure 3.8 This can lead to the colors of smallest indices to not be unique in certain biconnected components of a graph. Finally, notice that every biconnected subgraph of a graph is also a connected subgraph. This implies that a centered coloring for a graph G would also be a block-centered coloring for G, meaning that $\chi_{bc}(G) \leq \chi_c(G)$ or equivalently $\mathsf{td}_2(G) \leq \mathsf{td}(G)$ for every graph G.

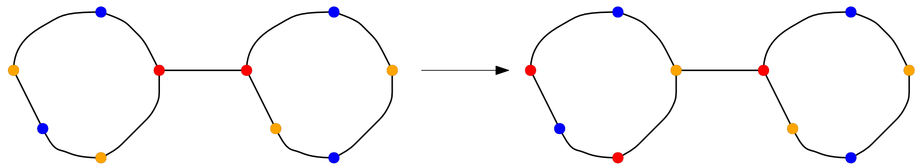

Figure 3.8: An example of how a permutation of colors works in a biconnected component in order to fix a block-centered coloring into a proper coloring. Here, every red vertex in the left biconnected component becomes an orange vertex and vice versa.

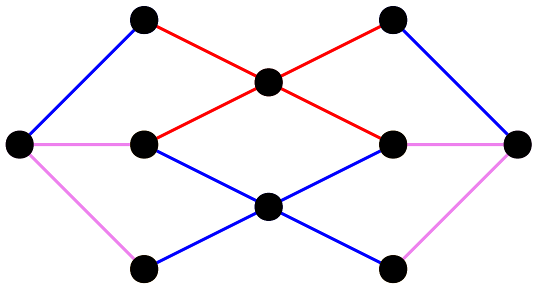

Figure 3.9: An example of a cycle edge ranking. Here, red represents the color of the minimum index while pink represent the color of the highest index. The fact that there are only 3 colors needed implies that the block treedepth of this graph is less than or equal to 3.

3.3.2 Cycle Edge Rankings

In an attempt to define an analogous of vertex rankings for block treedepth, we introduce a coloring that resembles rankings and is equivalent to td_2. In this direction, we introduce *cycle edge rankings*.

Definition 3.18. An edge coloring of a graph G, $\mathsf{r} : E(G) \to \mathbb{N}$, is called a *cycle edge ranking* if for every cycle $C \leq_{\text{sb}} G$ either of the following holds:

- $|\mathsf{argmin}_{e \in E(C)}(\mathsf{r}(E(C))| = 1$, or
- $\bigcap \mathsf{argmin}_{e \in E(C)}(\mathsf{r}(E(C)) = \{v\}$ for some $v \in V(C)$.

The minimum amount of colors a cycle edge ranking uses for a graph G is denoted as $\mathsf{cer}(G)$.

In plain words, cycle edge rankings are a kind of edge colorings such that for every cycle of a graph G either the color of smallest index is unique, or the edges with the color of the smallest index share a common vertex. Notice that cycle edge rankings are not necessarily proper edge colorings, meaning that two edges sharing a vertex can have the same color. As a matter of fact, it is quite common for cycle edge rankings to assign same colors to adjacent edges as we can see in Figure 3.9. We now proceed to prove the equivalence of cycle edge ranking and block treedepth.

Proposition 3.19. For every non-negative integer k, $\mathsf{cer}(G) \leq k$ if and only if $\mathsf{td}_2(G) \leq k$.

Proof. We will use induction on k. For $k = 0$, it is obvious that $\mathsf{td}_2(G) = 0$ if and only if G is edgeless and that G is edgeless if and only if $\mathsf{cer}(G) = 0$, which concludes the base case for both directions.

First we will prove that when $\mathsf{td}_2(G) \leq k$ is true then $\mathsf{cer}(G) \leq k$ is also true. Assume that $k \geq 1$ and $\mathsf{td}_2(G) \leq k$ and that G is biconnected. This means that there exists a vertex $u \in V(G)$ such that $\mathsf{td}_2(H) \leq k-1$, where $H = G \setminus u$. By induction hypothesis, H has a cycle edge ranking that uses at most $k-1$ colors. We produce a new cycle edge ranking r' by setting $\mathsf{r}'(e) = 1$ for every edge e whose one endpoint is u and $\mathsf{r}'(e) = \mathsf{r}(e) + 1$ for every remaining edge. We claim that r' is a cycle edge ranking for G. Let C be a cycle of G. If C does not contain u, then $\mathsf{r}'|_H$ confirms that the extension of r in G is a cycle edge ranking. Suppose that $u \in V(C)$. Notice that if $e \in \mathsf{argmin}_{e \in E(C)}(\mathsf{r}(E(C))$ then $\mathsf{r}'(e) = 1$. By construction, the only edges in G satisfying $\mathsf{r}'(e) = 1$ all share a common endpoint. Thus, r' is a cycle edge ranking for G using at most k colors and $\mathsf{cer(G)} \leq \mathsf{k}$. Now, assume that G is not biconnected and that $\mathsf{td}_2(G) \leq k$. Let $B = \{B_1, \ldots, B_l\}$ be the set of all blocks of G. For every block $B_i \in B$, it is true that $\mathsf{td}_2(B_i) \leq k$. This means that every block has a cycle edge ranking ϕ_{i} using at most k colors, all from $[k]$. We claim that $\phi = \bigcup_{i \in [l]} \phi_{\mathsf{i}}$ is a cycle edge ranking of G. Indeed, since the coloring is not proper, there is no problem with adjacent edges having the same colors in ϕ. On the other hand, no cycle can be contained in more that one block. Thus, for every cycle $C \subseteq_{\mathsf{sb}} (G)$, there exists a block B_j containing it and hence $\phi\mid_{B_j}$ confirms that either all edges of minimum color in C share a vertex or the edge of minimum color is unique. Altogether, $\mathsf{cer}(G) \leq k$.

Now, we will prove that when $\mathsf{cer}(G) \leq k$ is true then $\mathsf{td}_2(G) \leq k$ is also true. Thus, assume that $k \geq 1$ and that $\mathsf{cer}(G) \leq k$. Let $B \in \mathcal{B}$ be a maximal biconnected component of G. Suppose, towards a contradiction, that $e_1, e_2 \in E(B)$ are two non-adjacent edges satisfying $\mathsf{r}(e_1) = \mathsf{r}(e_2) = 1$. For every two edges that belong to the same biconnected component, there exists a cycle containing both edges. Thus, either $|\{e \in B \mid \mathsf{r}(e) = 1\}| = 1$ or $\bigcap\{e \in B \mid \mathsf{r}(e) = 1\} \in V(B)$ holds. We pick a vertex v from B as follows: if $\{e \in E(B) \mid \mathsf{r}(e) = 1\} = \{u_1u_2\}$ then set $v = u_1$. Otherwise, $v = \bigcap\{e \in B \mid \mathsf{r}(e) = 1\}$. Let $H = B \setminus v$. The restriction $\mathsf{r}|_H$ of r on H uses at most $k-1$ colors and is a cycle edge ranking for H. Therefore, by induction hypothesis $\mathsf{td}_2(H) \leq k-1$. H is acquired from G by removing exactly one vertex from every biconnected component of G and thus we get that $\mathsf{td}_2(B) \leq \mathsf{td}_2(H)+1 \leq k$. Since this is true for any block of G, by definition of block treedepth $\mathsf{td}_2(G) \leq k$. This concludes the proof. □

The relation between cycle edge rankings and block-centered colorings is actually not that hard to spot. Previously, we saw that if we know the order in which vertices where eliminated from a graph G while calculating its block treedepth, we can produce a block-centered coloring for G. In order to make this coloring a proper coloring we need to perform some permutations. Suppose that we do not perform the aforementioned permutations presented in the proof of Lemma 3.17. Then we have a coloring that does assign a vertex of unique color to every biconnected subgraph of G but might not be proper. Now, if we transfer the color of every vertex to its adjacent edges in a way such that every edge gets the color of smallest index among the colors that its endpoints are assigned by the (non- proper) block-centered coloring, we get a cycle edge ranking for the graph.

3.3.3 A Searching Game for Block Treedepth

In an attempt to better understand block treedepth, we define cops and robbers games for block treedepth. The games are quite similar to those that were described for treedepth, but of course we adapt the rules to fit the nature of block treedepth. The general rules for the game are that we have a unique robber that is always acting in its best interest and an infinite amount of cops that are placed on vertices. The robber however now stands on edges. At the beginning, the robber selects an edge. Then the cops announce the vertex that they are going to occupy, and right before they occupy it the robber can move again. However, unlike the first variation that we described, the robber can only move between edges that belong in the same block. For example, the robber cannot use cut-vertices to move from a block of a graph to another one. A humorous way to physicalize the extra restriction in order to make it easier to visualize this version of the game, is to think of the robber as acrophobic, thus cannot use bridges. The game ends once the cops capture both the endpoints of the edge that the robber is standing on. Again, we have a Searcher-Stationary version of the game in which the robber is visible and the cops cannot move once placed on a vertex and a LIFO-search version, where the robber is invisible and the cops can move in a LIFO order. The minimum amount of cops needed to capture a robber on a graph G in this variation is denoted by $\mathsf{vcb}(G)$ in the Searcher-Stationary version and $\mathsf{icb}(G)$ in the LIFO-search version. Before we relate block treedepth with $\mathsf{cb}(G)$, we give the definition of shacks as an analogous of shelters but for block treedepth.

Definition 3.20. A *shack* of a graph G is a collection $\mathcal{S}$ of non-empty block-subgraphs of G such that for every non-minimal set $S \in \mathcal{S}$ νo vertex belongs to all its children. In other words,

$$\bigcap\{S' \mid S' \in M_{\mathcal{S}}(S)\} = \emptyset.$$

where $M_{\mathcal{S}}(S)$ is the $\subseteq$-maximal elements of $\{S' \in \mathcal{S} : S' \subset S\}$. The *thickness* of a shelter $\mathcal{S}$ is the minimal length of a maximal $\subseteq$-chain and is denoted by $\mathsf{th}(\mathcal{S})$.

Now, we have the tools to introduce the analogous of Theorem 3.11 for this new version of the game.

Theorem 3.21. Let G be a graph with at least one edge and k be a non-negative integer. Then the following are equivalent.

(i) for every shack $\mathcal{S}$ of G, $\mathsf{th}(\mathcal{S}) \leq k+1$.

(ii) $\mathsf{td}_2(G) \leq k$.

(iii) $\mathsf{vcb}(G) \leq k+1$.

(iv) $\mathsf{icb}(G) \leq k+1$.

Proof. (i) $\Rightarrow$ (ii). We will prove the contrapositive by induction. For $k = 0$, let G be a graph such that $\mathsf{td}_2(G) > 0$. Of course this graph has a shack of thickness at least 2, since any graph that has an edge has a corresponding shack that consists of the graph itself and all its vertices as singletons. Suppose that if G is a graph such that $\mathsf{td}_2(G) > k$ then some shack S of G satisfies $\mathsf{th}(\mathcal{S}) > k+1$. We will prove that if $\mathsf{td}_2(G) > k+1$ for a graph G, then G has a shack of thickness at least $k+2$. From the definition of block treedepth, it is true that there exists a block B of G such that $\mathsf{td}_2(B) = \mathsf{td}_2(G)$. For every vertex $v \in V(B)$ let B_v be equal to the graph $B \setminus v$. It is true that $\mathsf{td}_2(B_v) \geq$

$\mathsf{td}_2(B) - 1 > k$ and thus, by induction hypothesis, for every B_v there exists a shack $\mathcal{S}_v$ satisfying $\mathsf{th}(\mathcal{S}_v) > k + 1$. We claim that $\mathcal{S}' = \{B\} \cup \bigcup_{v \in V(B)} \mathcal{S}_v$ is a shack of B. Indeed, $\bigcap\{S' \mid S' \in M_{\mathcal{S}'}(B)\} = \emptyset$, since every vertex of $V(B)$ is absent from at least one set in $M_{\mathcal{S}'}(B)$. Also, $\mathsf{th}(\mathcal{S}') = \mathsf{th}(\mathcal{S}_v) + 1 > k + 2$. Finally, it is easy to check that since B is a subgraph of G and has a shack of thickness greater than $k + 2$ then G also has a shack with the same property, because the shack of the subgraph is a shack for G too. This concludes the first proof.

(ii) ⇒ (iii). We will prove the statement by induction on k. For $k = 1$, let G be a graph with $\mathsf{td}_2(G) \leq 1$. Since G is not edgeless, G is a forest graph. In the current version of cops and robbers, the robber is not allowed to move in a forest and thus all the cops have to do is to cover the endpoints of the edge that the robber is sitting on, meaning that $\mathsf{vcb}(G) \leq 2$. Suppose that if $\mathsf{td}_2(G) \leq k$ then $\mathsf{vcb}(G) \leq k + 1$ holds. We will prove that if $\mathsf{td}_2(G) \leq k+1$ then $\mathsf{vcb}(G) \leq k+1$ is also true. Let B be any biconnected component of G. By the definition of block treedepth, we know that $\mathsf{td}_2(B) \leq \mathsf{td}_2(G)$. Then, there exists a vertex $v \in V(B)$ such that $\mathsf{td}_2(B \setminus v) = \mathsf{td}_2(G) - 1 \leq k$. By induction hypothesis, it holds that $\mathsf{vcb}(B \setminus v) \leq k+1$. Consider that we place a cop on v in B. Then, we need at most $k + 1$ cops to capture the robber in the remaining graph. Thus, $\mathsf{vcb}(B) \leq k + 2$. Notice that according to the rules the robber cannot transfer from a biconnected component to another since that would require moving between edges that do not belong in the same biconnected component. Hence, since $k + 2$ cops are enough in order to secure any biconnected component, we have that $\mathsf{vcb}(G) \leq k+2$, which concludes the proof.

(iii) ⇒ (i). Again, we use induction to prove the contrapositive. For $k = 1$, let G be a graph and $\mathcal{S}$ be a shack such that $\mathsf{th}(\mathcal{S}) > 2$. Then G has at least one cycle and thus there are at least 3 cops needed in order to capture a visible fugitive, meaning that $\mathsf{vcb}(G) > 2$. Now, suppose that if G has a shack $\mathcal{S}$ such that $\mathsf{th}(\mathcal{S}) > k+$ then $\mathsf{vcb}(G) > k+1$ is true. We will prove that G having a shack $\mathcal{S}$ such that $\mathsf{th}(\mathcal{S}) > k+2$ implies that $\mathsf{vcb}(G) > k+2$. Let B be any maximal element of S and let B' be a child of B in S. This means that B' has a shack S' whose thickness is greater than $k+1$. By induction hypothesis, $\mathsf{vcb}(B') > k + 1$. Now, let v be a vertex in $V(B)$. Suppose that the robber is sitting on an edge in B whose one endpoint is v and let u be the first vertex that the cops cover in B. By definition of a shack, there exists a child B'' of B such that $u \notin V(B'')$. If the robber picks an edge in $E(B'')$ to stand on, there are at least k cops needed in order to capture it from then on. Hence, $\mathsf{vcb}(G) \geq \mathsf{vcb}(B) > k + 2$, which concludes the proof.

(ii) ⇒ (iv). We will prove the statement by induction on k, following a quite similar method as in (ii) ⇒ (iii). For $k = 1$, let G be a graph with $\mathsf{td}_2(G) \leq 1$. Again, G is not edgeless and thus it is a forest. Regardless that the robber is invisible, it cannot move away from the edge that it originally selected since the graph is a forest and thus a strategy for the cops is to cover the endpoints of every edge one by one, meaning that $\mathsf{icb}(G) \leq 2$. Now, we assume that whenever $\mathsf{td}_2(G) \leq k$ then $\mathsf{icb}(G) \leq k + 1$ is true. We will prove that if $\mathsf{td}_2(G) \leq k + 1$ then $\mathsf{icb}(G) \leq k + 1$ is also true. Let B_i be any biconnected component of G and v_i be a vertex in $V(B_i)$ such that such that $\mathsf{td}_2(B_i \setminus v_i) = \mathsf{td}_2(G) - 1 \leq k$. By induction hypothesis, it holds that $\mathsf{icb}(B_i \setminus v_i) \leq k + 1$. Consider that we place a cop on v_i in B_i. Then, we need at most $k + 1$ cops to capture the robber in the remaining biconnected component.Thus, $\mathsf{icb}(B_i) \leq k + 2$. Once again, according to the rules the robber cannot transfer from a biconnected component to another since that would require moving between edges that do not belong in the same biconnected component. Hence, since $k + 2$ cops are enough in order to secure any biconnected component, a valid strategy for the cops would be to exhaustively check every biconnected component one by one using $k+2$ cops. Altogether, we have that $\mathsf{icb}(G) \leq k+2$.

(iv) $\Rightarrow$ (i). We will prove the contrapositive by induction. Indeed, for $k = 1$ the contrapositive of the statement is true since every graph that has a shack of thickness $\mathtt{th}(\mathcal{S}) > 2$ also has a cycle and thus there are more than 2 non-stationary cops needed to search every edge and to capture an invisible robber. Suppose that the statement is also true for every $k' < k$, that is for every $k' \leq k$ G having a shack of thickness $\mathtt{th}(\mathcal{S}) > k'+1$ implies that $\mathsf{icb}(G) > k'+1$. We will prove that the statement is also true for $k+1$. Let G be a graph that has at least one shack $\mathcal{S}$ that satisfies $\mathtt{th}(\mathcal{S}) > k+2$ and let M be any maximal element of $\mathcal{S}$ that is also a maximal biconnected component of G. Then, any child M' of M has a shack of thickness $\mathtt{th}(\mathcal{S}) > k+1$. Suppose that the cops want to search for the robber in M. This means that they pick a vertex $v' \in V(M)$ to place a cop on. By definition of shack, there exists a child M' of M in $\mathcal{S}$ such that $v' \notin V(M')$. By induction hypothesis, there are at least $k+1$ non-stationary cops needed to exhaustively search M'. The first cop cannot move unless every next cop placed in M' is removed from the graph and on the other hand, if all the other $k+1$ cops are removed and the $k+2$-th cop decides to move to a vertex v'', then there exists a different biconnected subgraph M'' of M such that $v' \notin M''$ for which there are at least $k+1$ cops needed to completely search. Thus, $\mathsf{vcb}(G) \geq \mathsf{vcb}(M) > k+2$. $\square$

CHAPTER 4

A POLYNOMIAL KERNEL

Consider the following problem:

BLOCK TREEDEPTH
Input: A graph G, an integer k and a vertex cover C of G.
Parameter: $\lvert C \rvert$.
Question: Is the block treedepth of G at most k?

Our aim here is to prove that there exists a polynomial kernel for BLOCK TREEDEPTH when parameterized by vertex cover number. The idea behind the rules that will eventually lead to the kernel is similar to the rules for treedepth parameterized by vertex cover [9], but of course adjusted to the nature of block treedepth.

In what follows, $G = (V, E)$ is a graph that has at least one cycle (otherwise the problem would be trivial), C is a non-empty cover of G, k a positive integer and finally I represents the independent set $V(G) \setminus C$. We will describe some rules that will not change the block-treedepth of G.

Lemma 4.1. Let $u \in V(G)$ be an isolated vertex of G and let $G' = G[V(G) \setminus \{u\}]$. Then $\mathtt{td}_2(G) = \mathtt{td}_2(G')$.

Proof. Clearly, $\mathtt{td}_2(G) \geq \mathtt{td}_2(G')$. For the converse inequality, let ϕ be a block-centered coloring of G'. Since u is not part of any biconnected component of G and since u has no neighbors in G, we arbitrarily assign a color c to u that is already being used by ϕ on G' and extend ϕ to a block-centered coloring ϕ' of G using at most as many colors as ϕ uses. Thus, $\mathtt{td}_2(G) \leq \mathtt{td}_2(G')$, which concludes the proof. □

This lemma confirms that the following rule is safe.

Rule 1. Let $u \in V(G)$ be an isolated vertex. Then, delete u from G.

Lemma 4.2. Let $e \in E(G)$ be a bridge and $G' = (V, E \setminus \{e\})$. Then $\mathtt{td}_2(G) = \mathtt{td}_2(G')$.

Proof. It is trivial that $\mathtt{td}_2(G) \geq \mathtt{td}_2(G')$. Now, let ϕ be a block-centered coloring for G'. We will prove that every biconnected subgraph of G has a vertex of unique color when colored by ϕ. Let H be a biconnected subgraph of G. Since H is a biconnected subgraph of G' and e does not participate in any biconnected component, then H is also a biconnected subgraph of G. Thus, ϕ is a block-centered coloring for G. This shows that $\mathsf{td}_2(G) \leq \mathsf{td}_2(G')$, resulting to $\mathsf{td}_2(G) = \mathsf{td}_2(G')$. $\square$

Rule 2. Let $e \in E(G)$ be a bridge. Then, delete e from G.

Lemma 4.3. Suppose that $\mathsf{td}_2(G) \leq k$ and let $u_1, u_2 \in V(G)$ be non-adjacent vertices such that $|N_G(u_1) \cap N_G(u_2)| > k$. If $G' = (V, E \cup \{u_1, u_2\})$, then $\mathsf{td}_2(G) = \mathsf{td}_2(G')$.

Proof. It is easy to check that $\mathsf{td}_2(G) \leq \mathsf{td}_2(G')$. In order to prove the converse, let ϕ be a block-centered coloring for G using at most k colors. First, we claim that after adding $\{u_1, u_2\}$ in $E(G)$, ϕ remains a proper coloring. In order to prove this, notice that for every pair $v_1, v_2 \in N_G(u_1) \cap N_G(u_2)$ the induced subgraph $H = G[\{u_1, u_2, v_1, v_2\}]$ is biconnected. Thus $\phi(u_1) \neq \phi(u_2)$, since otherwise every vertex $v \in N_G(u_1) \cap N_G(u_2)$ would need to be assigned a different color via ϕ and this would contradict the fact that ϕ uses at most k colors. Now, let B be a biconnected subgraph of G' that is not a biconnected subgraph of G. Clearly, $u_1, u_2 \in V(B)$. Notice that since B is biconnected in G' but not in G, then B either contains no vertex from $N_G(u_1) \cap N_G(u_2)$ or $V(B) = \{u_1, u_2, w\}$, where $w \in N_G(u_1) \cap N_G(u_2)$. In any case notice that for every pair $\{y_1, y_2\} \in N_G(u_1) \cap N_G(u_2)$ the subgraph $B' = G[V(B) \cup \{y_1, y_2\}]$ is biconnected. This means that either y_1 or y_2 have a unique color in B' or that some vertex of B has a unique color. The first case cannot be true, since otherwise every vertex in $N_G(u_1) \cap N_G(u_2)$ would need a different color and, as mentioned above, this is not possible. Thus, B has a vertex of unique color in G and hence it also has a vertex of unique color in G'. This implies that $\mathsf{td}_2(G) \geq \mathsf{td}_2(G')$, which concludes the proof. $\square$

We can safely introduce the third rule, which adds some edges to the graph.

Rule 3. Let u, v be non-adjacent vertices of G. If $|N_G(u_1) \cap N_G(u_2)| > k$ and at least one of $u \in C$ or $v \in C$ hold. Then, add the edge $\{u, v\}$ to G.

Lemma 4.4. Let $\mathsf{td}_2(G) \leq k$ and let $u \in V(G)$ be a simplicial vertex such that for every pair of vertices $v_1, v_2 \in N_G(u)$ it holds that $|(N_G(v_1) \cap N_G(v_2)) \setminus N_G(u)| > k$. If $G' = (V \setminus \{u\}, E)$, then $\mathsf{td}_2(G) = \mathsf{td}_2(G')$.

Proof. First, it is trivial to show that $\mathsf{td}_2(G) \geq \mathsf{td}_2(G')$. It remains to prove the opposite inequality. In what follows, we denote the clique neighborhood of u as $Q = N_G(u)$ and $R = \{v \in V(G) \mid \exists u_1, u_2 \in Q : v \in (N_G(u_1) \cap N_G(u_2)) \setminus N_G(u)\} \setminus \{u\}$. Suppose that ϕ' is a block-centered coloring of G'. We will extend ϕ' to a block-centered coloring of G, using $k+1$ colors by describing a construction. During this construction, consider u to be colorless. Initially, let B_1 be the maximal block of G that contains u. Notice that u belongs to exactly one maximal block, since every block containing u also contains at least two vertices from Q (actually contains Q) and two blocks of a graph cannot share an edge. Since B_1 is a biconnected subgraph of G, then $B_1 \cap G'$ is either a biconnected subgraph of G' or an edge of G'. In both cases, B_1 contains a vertex (say w_1) of unique color when we restrict ϕ' on its vertices. Now, let B_2 be the maximal block of $G \setminus \{w_1\}$ containing u. B_2 also contains a vertex (say w_2) of unique color via ϕ'. By repeating this process we claim that there exists a minimum integer $l > 0$ such that $B_l = G[u]$, meaning that B_l consists of exactly one vertex, which is u. It is trivial to check that B_i is a monotonically decreasing sequence of graphs (the relation here is the subgraph

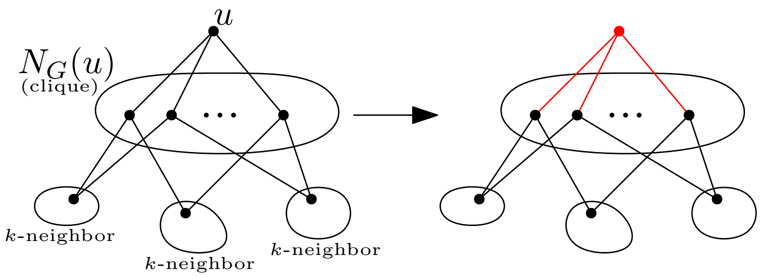

Figure 4.1: An example of the application of Rule 4 on a graph. Here we delete u because its neighborhood is a clique and every pair of vertices from the clique have a common neighborhood of size at least k, without counting u.

relation). Now, since $\mathsf{td}_2(G) \leq k$ notice that B_1 has at most $k+1$ colors via ϕ'. Thus B_{k+1} is exactly the empty graph. This means that there exists $l' \leq k+1$ such that $B_{l'} = G[\emptyset]$. Due to the fact that we consider u to be colorless, it is true that $u \in V(B_i)$ for all $i < l'$. These facts lead to the proof of the aforementioned claim. Moreover, notice that $l = l' - 1 \leq k$. Now, it is time to color u. We claim that any color that can be found in $B_{l-1} \setminus w_{l-1}$ and cannot be found in Q, would extend ϕ' to a block-centered coloring ϕ for G. Apparently, we first have to prove that $V(B_{l-1} \setminus w_{l-1}) \setminus Q \neq \emptyset$. For starters, there exist at least two vertices $q_1, q_2 \in Q$ such that $q_1, q_2 \in V(B_{l-1})$, since otherwise u would have at most one neighbor in B_{l-1} and thus u would not be a part of a larger biconnected component. Moreover, we claim that $V(B_{l-1})$ contains at least one common neighbor of q_1 and q_2. Notice that as long as q_1 and q_2 belong in B_i for some integer $i < k$, then every vertex that belongs to their common neighborhood and is not equal to any vertex w_j for any $j < i$ also belongs to B_i. This implies that it would take more than k iterations in order to eliminate every common neighbor of q_1 and q_2 from B_i. However, it is already proven that $l < k$. Thus, B_{l-1} indeed contains at least one common neighbor of q_1, q_2 and hence it is indeed true that $V(B_{l-1} \setminus w_{l-1}) \setminus Q \neq \emptyset$.

Finally, we have to prove that by arbitrarily assigning any color from $V(B_{l-1} \setminus w_{l-1}) \setminus Q$ to u, ϕ' is in fact extended to a block-centered coloring ϕ of G. Let B be a biconnected subgraph of G that contains u. It is easy to check that for any biconnected subgraph B of G containing u, it holds that $B_{l-1} \subseteq_{sb} B \subseteq_{sb} B_1$. This implies that there exists a positive integer $i < l-1$ such that $B \subseteq_{sb} B_i$ but $B \not\subseteq_{sb} B_{i+1}$. Observe that $w_i \in V(B)$ since otherwise B would either not be biconnected or it would be a subgraph of B_{i+1}. However, the color of w_i is unique in B_i and that means that its color is also unique in B. In order to conclude the proof, notice that for every vertex w_j, $j \in [l-1]$ $\phi(u) \neq \phi(w_j)$. This is true because there exists a vertex $q \in V(B_{l-1} \setminus w_{l-1}) \setminus Q$ such that $\phi(q) = \phi(u)$ and $q \in B_j$ for every integer $j \in [l-1]$. Hence, there exists a block-centered coloring for G using at most $\chi_{bc}(G')$ colors and thus $\mathsf{td}_2(G) = \mathsf{td}_2(G')$. □

The fourth and last rule follows.

Rule 4. Let $u \in V(G)$ be a simplicial vertex such that for every pair of vertices $v_1, v_2 \in N_G(u)$ it holds that $|(N_G(v_1) \cap N_G(v_2)) \setminus N_G(u)| > k$. Then, delete u from G.

Now we are ready to prove the existence of a cubic kernel for our problem.

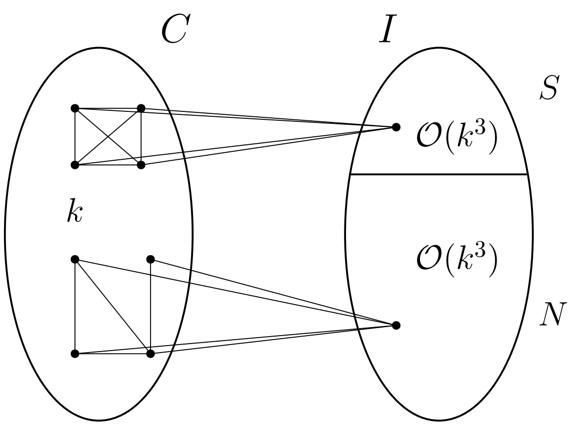

Figure 4.2: A graph partitioned into its vertex cover and independent set that is a sketch for the proof of Lemma 4.5.

Lemma 4.5. Let G be a graph such that $\mathtt{td}_2(G) \leq k$ and $|C| > k$. If none of the aforementioned rules are applicable to G, then the number of vertices of G is $\mathcal{O}(|C|^3)$.

Proof. Let S be the set of simplicial vertices of G that are not in C, N be the set of non-simplicial vertices not in C and $I = V(G) \setminus C$. Since Rule 1 and Rule 2 have been exhaustively applied, every vertex in G has at least 2 neighbors. Thus, every vertex in I has at least two neighbors in C. Now, from Rule 4, for every vertex $u \in S$ there exists an edge $\{v_1, v_2\} \in E(G[C])$ such that the common neighbors in S are at most k. We associate every vertex $u \in S$ with an edge in $E(G[C])$. Notice also that every edge in $E(G[C])$ is associated with at most k vertices in S. Thus, $|S| \leq k\frac{|C|(|C|-1)}{2} \leq |C|^3$. Now, let u be a vertex in N. This vertex is not simplicial and its whole neighborhood is in C. Therefore, there exists vertices $u_1, u_2 \in C$ such that $\{u_1, u_2\} \notin E(G)$. Notice that according to Rule 3 u_1 and u_2 cannot have more than k common neighbors. Every vertex in N has a corresponding pair of non-adjacent vertices in C. This implies that $|N| \leq k\frac{|C|(|C|-1)}{2} \leq |C|^3$. Finally, we have that $V(G) = |C|+|S|+|N| \leq 3|C|^3$. $\square$

We proceed to prove the main theorem of the chapter.

Theorem 4.6. BLOCK TREEDEPTH admits a polynomial kernel when parameterized by vertex cover number.

Proof. Consider (G, C, k) to be an instance of BLOCK TREEDEPTH parameterized by VERTEX COVER. In the case that $|C| < k$, the problem is trivial and (G, C, k) is obviously a YES-instance. Now, suppose that $|C| \geq k$. After exhaustively applying the aforementioned rules we get a graph G' such that $\mathtt{td}_2(G) \leq k$ if and only if $\mathtt{td}_2(G') \leq k$. We claim that $C' = V(G') \cap C$ is a vertex cover of G'. This certainly holds for every rule involving vertex or edge removals. However, this also applies for Rule 3 regardless of the fact that it adds an edge, as at least one of the vertices of this edge already belongs to the original vertex cover. This leads us to a new instance (G', C', k). Now, Lemma 4.5 guarantees that if $|V(G')| > 3|C'|^3$ then $\mathtt{td}_2(G') > k$ and thus (G', C', k) would be a NO-instance. In conclusion, BLOCK-CENTERED COLORING admits a cubic kernel when parameterized by vertex cover number. $\square$

Corollary 4.7. BLOCK TREEDEPTH is in FPT when parameterized by VERTEX COVER.

Proof. The proof follows by combining the results of Theorem 4.6 and Theorem 2.8, since the exhaustive application of the aforementioned reduction rules describes a kernelization algorithm. □

CHAPTER 5

CONCLUSION

Throughout this thesis, we introduced block treedepth from the view of block elimination distance as an analogous of what treedepth is for elimination distance. We provided two equivalent coloring definitions of block treedepth and then we saw how they can serve as an alternate tool to block tree layouts for technical proofs and we proved that there exist searching games that can be used to describe block treedepth. Furthermore, using techniques similar to the ones used for treedepth, we managed to prove that the problem of deciding the block treedepth of a graph admits a cubic kernel and is fixed parameter tractable when parameterized by vertex cover number. For future work, we conjecture that a parameterization of the problem by the size of a minimum feedback vertex set would possibly offer an even smaller kernel, since feedback vertex is more suitable for the nature of block treedepth. Finally, a fixed parameter algorithm for both block treedepth and block elimination distance would be interesting since these parameters seem promising when we consider parameterizing other problems by them.

BIBLIOGRAPHY

[1] Hans L. Bodlaender, Jitender S. Deogun, Klaus Jansen, and Ton Kloks, *Rankings of Graphs*, SIAM Journal on Discrete Mathematics, 11(1), 168–181.

[2] Jannis Bulian, and Anuj Dawar, *Graph Isomorphism Parameterized by Elimination Distance to Bounded Degree*, Algorithmica. 2015,75(2):363-382.

[3] Reinhard Diestel, *Graph Theory*, Electronic Edition 2005.

[4] Marek Cygan, Fedor V. Fomin, Lukasz Kowalik, Daniel Lokshtanov, Dániel Marx, Marcin Pilipczuk, Micha Pilipczuk, and Saket Saurabh, *Parameterized Algorithms*, Springer, 2015.

[5] Öznur Yaşar Diner, Archontia C. Giannopoulou, Giannos Stamoulis, and Dimitrios M. Thilikos, *Block Elimination Distance*, Graphs and Combinatorics, 38(5).

[6] Fedor V. Fomin, and Dimitrios M. Thilikos, *An annotated bibliography on guaranteed graph searching*, Theoretical Computer Science, 399(3), 236–245.

[7] 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*, Discrete Applied Mathematics, 160(15), 2089–2097.

[8] Tony Huyng, Gwenaël Joret, Piotr Micek, Michał T. Seweryn, and Paul Wollan, *Excluding a Ladder*, Combinatorica, November 2021.

[9] Yasuaki Kobayashi, and Hisao Tamaki, *Treedepth Parameterized by Vertex Cover Number*, 11th International Symposium on Parameterized and Exact Computation (IPEC 2016)

[10] Wojciech Nadara, Michał Pilipczuk, and Marcin Smulewicz, *Computing treedepth in polynomial space and linear fpt time*, May 2022.

[11] Jaroslav Nešetřil, and Patrice Ossona de Mendez, *Tree-depth, subgraph coloring and homomorphism bounds*, European Journal of Combinatorics, Elsevier, 2006, 27(6), pp.1022-1041.

[12] Jaroslav Nešetřil, and Patrice Ossona de Mendez, *Grad and classes with bounded expansion I. Decompositions*, European Journal of Combinatorics, 2008, 29(3), 760–776

[13] Alex Pothen, *The complexity of optimal elimination trees*,Technical Report CS 88-16, Department of Computer Science, Penn State 1988.