TAMPEREEN TEKNILLINEN YLIOPISTO
TAMPERE UNIVERSITY OF TECHNOLOGY

Juho Lauri

# Chasing the Rainbow Connection: Hardness, Algorithms, and Bounds

Julkaisu 1428 • Publication 1428

Tampere 2016

Tampereen teknillinen yliopisto. Julkaisu 1428
Tampere University of Technology. Publication 1428

Juho Lauri

# Chasing the Rainbow Connection: Hardness, Algorithms, and Bounds

Thesis for the degree of Doctor of Science in Technology to be presented with due permission for public examination and criticism in Tietotalo Building, Auditorium TB224, at Tampere University of Technology, on the 3rd of November 2016, at 12 noon.

Tampereen teknillinen yliopisto - Tampere University of Technology
Tampere 2016

ISBN 978-952-15-3836-0 (printed)
ISBN 978-952-15-3842-1 (PDF)
ISSN 1459-2045

# Abstract

We study rainbow connectivity of graphs from the algorithmic and graph-theoretic points of view. The study is divided into three parts. First, we study the complexity of deciding whether a given edge-colored graph is rainbow-connected. That is, we seek to verify whether the graph has a path on which no color repeats between each pair of its vertices. We obtain a comprehensive map of the hardness landscape of the problem. While the problem is NP-complete in general, we identify several structural properties that render the problem tractable. At the same time, we strengthen the known NP-completeness results for the problem. We pinpoint various parameters for which the problem is fixed-parameter tractable, including dichotomy results for popular width parameters, such as treewidth and pathwidth. The study extends to variants of the problem that consider vertex-colored graphs and/or rainbow shortest paths. We also consider upper and lower bounds for exact parameterized algorithms. In particular, we show that when parameterized by the number of colors $k$, the existence of a rainbow $s$-$t$ path can be decided in $O^*(2^k)$ time and polynomial space. For the highly related problem of finding a path on which all the $k$ colors appear, i.e., a colorful path, we explain the modest progress over the last twenty years. Namely, we prove that the existence of an algorithm for finding a colorful path in $(2-\varepsilon)^k n^{O(1)}$ time for some $\varepsilon > 0$ disproves the so-called Set Cover Conjecture.

Second, we focus on the problem of finding a rainbow coloring. The minimum number of colors for which a graph $G$ is rainbow-connected is known as its rainbow connection number, denoted by $\mathrm{rc}(G)$. Likewise, the minimum number of colors required to establish a rainbow shortest path between each pair of vertices in $G$ is known as its strong rainbow connection number, denoted by $\mathrm{src}(G)$. We give new hardness results for computing $\mathrm{rc}(G)$ and $\mathrm{src}(G)$, including their vertex variants. The hardness results exclude polynomial-time algorithms for restricted graph classes and also fast exact exponential-time algorithms (under reasonable complexity assumptions). For positive results, we show that rainbow coloring is tractable for e.g., graphs of bounded treewidth. In addition, we give positive parameterized results for certain variants and relaxations of the problems in which the goal is to save $k$ colors from a trivial upper bound, or to rainbow connect only a certain number of vertex pairs.

Third, we take a more graph-theoretic view on rainbow coloring. We observe upper bounds on the rainbow connection numbers in terms of other well-known graph parameters. Furthermore, despite the interest, there have been few results on the strong rainbow connection number of a graph. We give improved bounds and determine exactly the rainbow and strong rainbow connection numbers for some subclasses of chordal graphs. Finally, we pose open problems and conjectures arising from our work.

# Preface

Despite the single name on the front cover, this thesis would not have seen the light of day without the help and support of several people. I am sincerely grateful to both my supervisor, professor Tapio Elomaa, and my advisor, D.Sc. Henri Hansen. Tapio routinely handled all the administrative tasks allowing me to concentrate on research. Henri's lecturing in as early as 2008 inspired me to get curious about theory. We had many fruitful discussions already during the time of my Master's thesis, and our meetings were no different later on. All the guidance on navigating the academia I received from both Tapio and Henri was invaluable.

Unknowingly, the work on this thesis began in January 2013 while I was at Michigan Technological University. At the time, the topic was something that I explored out of curiosity alongside with my studies. Little did I know that the work would evolve to be a foundation for my Master's thesis, and eventually a part of this thesis as well. I wish to warmly thank professor Melissa Keranen for being an excellent teacher, and for the many discussions and collaboration.

I quickly found it stimulating, productive, and fun to work on problems not strictly related to the thesis. The most valuable of such undertakings was a 4-month visit to Aalto University in the Summer of 2014, prior to the beginning of my doctoral studies. I want to thank professor Petteri Kaski for the opportunity to work under his guidance. Our many discussions contributed much to my professional growth.

A crucial part of the process was (and is) networking. This might require pushing the boundaries of what you feel comfortable with, but I'm glad that I did. First, I want to thank professor Stefan Szeider for an interesting talk at the 2013 SAT/SMT Summer School, and for a visit to TU Wien in December 2014. Second, I am happy to have met professor Mikko Koivisto in June 2015 at the annual meeting for The Finnish Society for Computer Science in Jyväskylä. I very much enjoyed the later visits, problem solving sessions, and discussions at the University of Helsinki. Third, I thank professor Łukasz Kowalik for his openness and positive attitude. I learned a lot in the whiteboard sessions during my visit to the University of Warsaw in September 2015.

I thank professor Yongtang Shi from Nankai University and professor Jukka Suomela from Aalto University for kindly agreeing to pre-examine my thesis. Their comments and careful reading improved this work. Similarly, I thank professor Pinar Heggernes from the University of Bergen for acting as my opponent. Furthermore, I thank all the anonymous reviewers of various journals and conferences for helpful comments and suggestions related to the manuscripts included in this thesis.

I thank all my co-authors involved with the projects outside of this thesis as well that I got to meet. In no particular order, many thanks Missy, Petteri, Łukasz, Robert, Eduard,

Henri, and Marzio for sharing your expertise. I also thank Pierre Hauweele for plugging in my code and all the practical help with GraPHedron.

The financial support from the Emil Aaltonen Foundation for funding my research between 2015 – 2017 is gratefully acknowledged. I also thank the Finnish Foundation for Technology Promotion (TES) for their support.

The much needed balance for work came in the form of friends and family. I have had the pleasure of sharing (two separate) offices with Elina, Henri, and Kalle, all of whom have become more important to me than just co-workers. I also thank other people at the Department of Mathematics for making me feel at home. In particular, I wish to thank Tiina Sävilahti and Kari Suomela for helping me (with a smile) with whatever I happened to need help with. For my many other friends here and around the world: thanks!

Most importantly, I am thankful for my parents for all the love and support throughout the years. Mikko, thank you for the countless discussions, and for being a brother I could always look up to. Soffi, thank you for everything.

Tampere, October 2016,

Juho Lauri

*To the loving memory of Mari.*

# Contents

# List of publications

This thesis consists of the following five publications. They are referred to as [P1] – [P5] in the text. The publications are presented in an order natural for the discussion to follow. For each publication, the author ordering is alphabetical.

[P1] Juho Lauri. Further hardness results on rainbow and strong rainbow connectivity. *Discrete Applied Mathematics* 201 (2016), pp. 191 – 200.

[P2] Juho Lauri. Complexity of rainbow vertex connectivity problems for restricted graph classes. Submitted to *Discrete Applied Mathematics.*

[P3] Łukasz Kowalik and Juho Lauri. On finding rainbow and colorful paths. *Theoretical Computer Science* 628 (2016), pp. 110 – 114.

[P4] Eduard Eiben, Robert Ganian, and Juho Lauri. On the complexity of rainbow coloring problems. Accepted to *Discrete Applied Mathematics.*

[P5] Łukasz Kowalik, Juho Lauri, and Arkadiusz Socała. On the fine-grained complexity of rainbow coloring. In: *Proceedings of the 24th Annual European Symposium on Algorithms, ESA 2016, Aarhus, Denmark, August 22-24*. 2016, 58:1 – 58:16.

The preliminary version of [P4] appeared as: Eduard Eiben, Robert Ganian, and Juho Lauri. On the complexity of rainbow coloring problems, In: *Proceedings of the 26th International Workshop on Combinatorial Algorithms, IWOCA 2015, Verona, Italy, October 5 – 7.* 2015, pp. 209 – 220.

# 1 Introduction

This introductory chapter gives a gentle introduction to the topic of rainbow coloring and connectivity problems. Some graph-theoretic and complexity-theoretic background is useful when moving on to defining the research goals. We refer the reader unfamiliar with these concepts to Chapter 2. Throughout the rest of the thesis, we may sometimes refer to computational problems assumed to be common knowledge in the field. For completeness, we provide the reader with a list such problems in Appendix A.

## 1.1 Background

Connectivity is a fundamental graph concept. A basic graph-theoretical question is the following: given a graph, is there a path between two vertices? Often in applications, it is not sufficient merely to determine the existence of any such a path. Instead, we wish to determine the existence of a path with particular properties. A classical example in e.g., pathfinding and routing is finding a shortest path between two vertices. Another example is finding a longest path to maximize the coverage of a forward-moving agent maneuvering through the graph. Solving such pathfinding problems is of significant practical and theoretical interest.

Applicatons rarely conform to simple theoretical models. For instance, it could be useful to associate vertices and/or edges with numbers or colors. Such quantities can represent length, time, or type of a resource. For example, the well-known textbook of Kleinberg and Tardos [51] describes the following application in monitoring and marketing. A company has a website that services both subscribers and nonsubscribers. All content of the website is shown to subscribers, but access for nonsubscribers is limited. More specifically, nonsubscribers can view any page, but the maximum number of pages viewed in a single session is limited. The website is modeled by a (directed) graph $G = (V, E)$, in which the vertices correspond to pages, and edges to hyperlinks between pages. The website has a front page, which a particular vertex $s \in V$ corresponds to. The pages are divided into zones. For instance, there is a distinct zone for music, politics, sports, and so on. More formally, the user session is tracked by dividing the vertex set $V$ into color classes $Z_1, Z_2, \ldots, Z_k \subseteq V$, where each color class $Z_i$ corresponds to such a zone. A navigation path of a nonsubscriber starting from $s$ is restricted to include at most one page from each zone $Z_i$. Otherwise, the user's session is terminated, and an ad is shown suggesting the user becomes a subscriber. A question the company asks is whether it is possible for a nonsubscriber to navigate from the front page $s$ to some other page $t$ in a single session, i.e., whether there is an $s$-$t$ path that passes through each zone at most once.

Formally, the colored paths the above problem considers are *rainbow*. That is, a rainbow path is a colored path on which no color repeats. Such paths naturally capture the constraint of avoiding the use of the same type of resource twice. A path in a vertex-colored graph is *vertex rainbow* if its internal vertices have distinct colors. Rainbow paths, in both edge-colored and vertex-colored graphs, are the central to this thesis.

**Motivation**. The $k$-RC problem asks whether the edges of a given graph can be colored in $k$ colors such that each vertex pair is connected by a rainbow path. The smallest such $k$ is known as the *rainbow connection number* of the graph, and can be viewed as yet another measure of graph connectivity. Similarly, the smallest number of colors needed to establish a rainbow *shortest* path between each pair of vertices is known as the *strong rainbow connection number* of a graph. Both concepts were introduced by Chartrand, Johns, McKeon, and Zhang [11] in 2008, while also featured in an earlier book of Chartrand and Zhang [13]. Rainbow coloring has attracted considerable attention with over 200 papers published on the topic by now (for an overview, see the survey of Li, Shi, and Sun [58]).

The concept of rainbow connectivity can be seen as a natural, interesting way of strengthening the connectivity property. Possible applications have been suggested that fall under the umbrella term of telecommunications and data transfer. For example, Chakraborty, Fischer, Matsliah, and Yuster [7] describe the following example in message routing. Given a network $G$, we wish to route message between every pair of vertices in a pipeline while requiring each edge (corresponding to a link) to use a distinct channel. The objective is to minimize the number of distinct channels used in the network. This number is precisely the rainbow connection number of $G$. In addition, Dorbec, Schiermeyer, Sidorowicz, and Sopena [26] note that rainbow paths generally appear in the context of onion routing, using layered encryption.

The concept of rainbow paths also appears in the context of broadcast scheduling. In such problems, we have a network of broadcast transceivers that operate on a shared medium. The goal is to schedule the use of the medium to ensure communication over the entire network. Moreover, the objective is to avoid transmission conflicts: simultaneous transmissions on the same medium conflict, and are thus expected to fail. Such problems are often modeled as vertex or edge coloring problems. In particular, these coloring problems often have a distance constraint. For instance, in the *distance-2 coloring problem*, the goal is to color the vertices of a graph with a smallest number of colors such that two vertices at a distance at most 2 receive distinct colors. A broadcast scheduling problem solved as an instance of distance-2 coloring guarantees every path of length 2 is rainbow. A relaxation suggested by Joseph and DiPippo [45] only requires one such path between every pair of vertices.

On the theoretical side, the problems pose several interesting questions. We explore these questions more in-depth in the following.

**Research goals**. It seems fair to argue most of the research has focused on the combinatorial aspects of the problem. In this thesis, we focus on the computational complexity of the problem along with some of its variants. We will also obtain results that can be seen as purely combinatorial results in the problem domain. In what is to follow, we highlight particular research questions and problems we study.

**Problem 1: What is the complexity of rainbow connectivity?**

Recall that in the $k$-RC problem, we are asked if the edges of an $n$-vertex graph can be colored such that there is a rainbow path between each pair of vertices. To prove that $k$-RC is NP-complete, we must show that the problem is in NP. Clearly, as there is a budget of $k$ colors, each rainbow path can be of length at most $k$. Thus, the certificate for proving membership in NP is roughly a set of size $n^2$, containing a colored path of length at most $k$ for each vertex pair. This observation motivates the study of the Rainbow Connectivity problem, in which we are given an edge-colored graph $G$, and have to decide whether $G$ is rainbow-connected. A similar question can be formulated for the vertex variant, namely Rainbow Vertex Connectivity. In this problem, the input graph $G$ is vertex-colored, and the question is whether $G$ is rainbow vertex-connected, i.e., has a path on which no color repeats on its internal vertices between each pair of vertices. For both problems, we also study their strong variants in which we ask for the existence of a *shortest* path — either rainbow or rainbow vertex, depending on the problem — between each pair of vertices. In general, we refer to these problems as *rainbow connectivity problems.*

We further study the problems on restricted graph classes. In particular, we aim to find the strongest possible restriction of the input under which the problems still remain hard. Moreover, we seek to determine graph classes for which the complexity of the problems differ. At the same time, we aim to identify structural parameters whose boundedness render the problems tractable.

**Problem 2: How fast of an algorithm can one have for finding a rainbow path?**

Clearly, an algorithm for determining whether a rainbow path between two vertices exists can be used for deciding rainbow connectivity problems. Not surprisingly, the problem is NP-complete, making the existence of a polynomial-time algorithm highly unlikely. But how fast of an exponential-time algorithm can one hope for?

We approach the question from the viewpoint of parameterized complexity, and take as the natural parameter the number of colors $k$. In other words, we push the apparently unavoidable exponential runtime dependence to the parameter $k$, and seek to find an algorithm running in time $f(k)n^{O(1)}$, where $f$ is a computable function depending solely on $k$. Questions precisely like this are at the core of the so-called optimality program in parameterized complexity (see e.g., [64]). For concreteness, we ask: what is the best possible $f(k)$ in the above runtime? In other words, we seek to systematically understand the underlying nature of hard computational problems. In particular, we strive to find lower bounds (under plausible complexity assumptions) together with matching upper bounds.

**Problem 3: Study the computational aspects of finding rainbow colorings**

We separate the problem of *finding* a rainbow coloring from the problem of *verifying* a given rainbow coloring. Indeed, *rainbow coloring problems* such as $k$-RC stand in contrast to rainbow connectivity problems. In such problems the coloring is given, whereas in rainbow coloring problems we have a budget of $k$ colors to construct a desired coloring.

We study rainbow coloring problems on restricted graph classes. First, we seek to pinpoint additional graph classes for which the problem remains hard. Second, we consider approximability of the problems. That is, even if a problem is hard to compute exactly, it could be possible we can find an almost optimal solution efficiently. What is the case for rainbow coloring?

In the spirit of the optimality program, we also study lower bounds (under reasonable complexity assumptions) for algorithms that find rainbow colorings. In particular, how much of an improvement could one hope for over the naive brute-force algorithm that goes through all possible colorings? In addition, we consider natural relaxations of the problem, in which we only wish to rainbow-connect specific vertex pairs by rainbow paths or wish to maximize the number of rainbow-connected vertex pairs with a budget of $k$ colors.

### Problem 4: Study the rainbow connection number of subclasses of chordal graphs

A fair amount of research on exact polynomial-time algorithms and upper bounds on the rainbow connection number has been concentrated on chordal graphs and their subclasses. A possible motivation is the following simple observation. It is known that there is an efficient algorithm to find a clique cover for a chordal graph, i.e., a (smallest) collection $\mathcal{C}$ of cliques that cover all edges of the graph. Color the graph by assigning a distinct color to each $C \in \mathcal{C}$, that is, color every edge of $C$ with the same color. It is easy to prove the graph is rainbow-connected under the obtained coloring. In fact, this coloring is optimal for some graphs. When does such a “simple” coloring cease to be optimal?

We continue the investigation of the rainbow connection number of subclasses of chordal graphs, including block graphs, split graphs, and chordal diametral path graphs. Such a study brings us closer to the combinatorial nature of the problem. In particular, the goal is to understand what features or structural properties of graphs make the problem hard, or conversely, render it tractable. Instead of tools from parameterized complexity, we approach the problems from the viewpoint of exact polynomial-time algorithms and combinatorial methods. This is in contrast to e.g., algorithmic metatheorems, that often abstract the problem quite heavily, and thus might obscure features that enable purely combinatorial algorithms.

### Problem 5: Find improved upper bounds on the rainbow connection numbers

Finding bounds as tight as possible for a graph parameter is a typical line of research within graph theory. Indeed, can we bound the rainbow connection number in terms of some other well-known parameter? In particular, we will focus on the strong rainbow connection number for which results have been considerable more scarce. A possible reason for the lack of results lies in the difficulty of studying it. For instance, in [58], the authors write: “The investigation of strong rainbow connection number is much harder than that of rainbow connection number.” The reason given is that the strong rainbow connection number is not a *monotone graph property*. That is, a property is monotone if it does not increase under edge addition. Thus, despite the interest, there have been less results concerning the strong rainbow connection number.

## 1.2 Summary of the main contributions

We summarize below the main contribution of this thesis, and how they answer the outlined research problems.

1. In [P1] and [P2], we considerably extend the known NP-completeness results for rainbow connectivity problems. For example, we show the problems remain intractable for bipartite planar graphs, interval graphs, $k$-regular graphs for every $k \geq 3$, graphs of bounded pathwidth, and graphs of bounded bandwidth. On the other hand, the problems are fixed-parameter tractable for e.g., tree-depth. Previously, no graph class was known where the complexity of the two problems Rainbow Connectivity and Strong Rainbow Connectivity would differ. We show that for block graphs, which form a subclass of chordal graphs, Rainbow Connectivity is NP-complete while Strong Rainbow Connectivity is in P.

2. In [P3], we show the existence of a rainbow path between two vertices can be decided in $2^k n^{O(1)}$ time and polynomial space, where $k$ is the number of colors in the given coloring. Moreover, we show that for any $\varepsilon > 0$, the existence of a $(2-\varepsilon)^k n^{O(1)}$-time algorithm for finding a path on which all $k$ colors appear, i.e., colorful path, violates the so-called Set Cover Conjecture.

3. In [P4], we prove that for every $k \geq 2$, it is NP-complete to decide if the vertices of a graph can be colored in $k$ colors such that there is a vertex rainbow shortest path between each pair of vertices. Moreover, the problem cannot be approximated in polynomial time within a factor of $n^{1/2-\varepsilon}$ for any $\varepsilon > 0$, unless $\mathsf{P} = \mathsf{NP}$. In fact, the same is true when restricted to graphs of diameter 3. We give positive results for rainbow coloring graphs of bounded vertex cover number and bounded treewidth. Moreover, we give a linear-time algorithm which decides whether it is possible to obtain a rainbow coloring by saving a fixed number of colors from a trivial upper bound.

4. The main result of [P5] states that there is no $2^{o(n^{3/2})}$-time algorithm for $k$-RC, for any $k \geq 2$ unless the Exponential Time Hypothesis (ETH) fails. In Section 4.1, we show that it is NP-complete to decide whether a split graph with a dominating vertex can be strongly rainbow-connected in $k$ colors. Furthermore, the strong rainbow connection number of a an $n$-vertex split graph cannot be approximated in polynomial time within a factor of $n^{1/2-\varepsilon}$ for any $\varepsilon > 0$, unless $\mathsf{P} = \mathsf{NP}$. We give further positive parameterized results in [P5] for relaxations of rainbow coloring, in which the goal is to rainbow-connect a maximum number of vertex pairs.

5. In Section 5.2, we determine exactly the strong rainbow connection number of block graphs. Moreover, we show that the quantity can be computed in linear time. Furthermore, we provide a polynomial-time characterization of bridgeless block graphs that have rainbow connection number 1, 2, 3, or 4. Finally, in Section 5.3, we give a tight upper bound for the rainbow connection number of chordal diametral path graphs, which form a subclass of chordal graphs.

## 1.3 Author's contribution

The main contributions of this work, as detailed in the previous subsection, involve concepts and results in computational complexity and mathematics. Joint research in these areas is a sharing of skills and ideas that is often impossible to attribute to individuals separately. It is characteristic of such work that ideas grow from discussions among all partners, and there are no specified roles for the involved researchers (this can be in contrast to e.g., laboratory sciences).

Nevertheless, below we give as exact breakdown of the author's contribution as possible. We stress that in each publication, the author ordering is alphabetical. As such, the author ordering does not imply any level of contribution.

**[P1] and [P2]** The present author is the sole author of both manuscripts.

**[P3]** The results and the writing of the manuscript are joint work with Łukasz Kowalik.

**[P4]** The results of Section 3 are due to the present author. The results of Section 4 are joint work with Eduard Eiben and Robert Ganian, as is the writing of the manuscript. The results of Section 6 are joint work of Eduard Eiben and Robert Ganian.

**[P5]** The results of Section 2 are joint work of Łukasz Kowalik and Arkadiusz Socała. The remaining results are joint work with Łukasz Kowalik.

The results presented in Section 5.1 and Section 5.2 are joint work with Melissa Keranen, and are based on the manuscripts [48, 49]. In addition, the results given in Section 5.3 are joint work with Henri Riihimäki, and are based on the manuscript [54].

## 1.4 The structure of the thesis

The thesis is divided into six chapters. In Chapter 2 we review basics of structural graph theory, and give an introduction to the combinatorial nature of rainbow coloring. Furthermore, we consider fixed-parameter tractability and techniques for proving lower bounds on exact and parameterized algorithms. A reader familiar with the concepts may freely proceed to Chapter 3.

In Chapter 3, we focus on the problem of verifying if a given graph is rainbow-connected under a given coloring. We obtain a thorough classification of the considered problems with respect to NP-completeness. These hardness results have negative consequences for several parameterized algorithms. We show the problems can be solved in $O^*(2^k)$ time and polynomial space parameterized by the number of colors $k$. We conclude the chapter by deriving lower bounds on parameterized algorithms solving related problems.

In Chapter 4 we proceed to the problem of finding a rainbow coloring for a given graph. In particular, we derive additional hardness results for restricted graph classes, implying results on the hardness of approximating the rainbow connection numbers. In addition, we obtain lower bounds under the Exponential Time Hypothesis (ETH). Such a result implies it is very unlikely to obtain an algorithm significantly faster than brute-force for many rainbow coloring problems. These results are complemented by positive results through a structural parameterization. In particular, we show several rainbow coloring problems are tractable parameterized by e.g., treewidth and the vertex cover number.

Finally, we turn our attention to purely combinatorial methods in Chapter 5. In particular, we will consider polynomial-time algorithms for exact or approximate rainbow coloring subclasses of chordal graphs. A particular focus will be on the strong rainbow connection number. Chapter 6 concludes the thesis along with some open problems.

2

# Preliminaries

In this chapter, we review some basics of structural graph theory and algorithms.

## 2.1 Notation

For a positive integer $n$, we write $[n] = \{1, 2, \ldots, n\}$.

We use standard asymptotic notation. For a function $h : \mathbb{N} \to \mathbb{N}$, we write $h(n) = n^{O(1)}$ to denote that $h$ is bounded from above by some polynomial. Sometimes it will be convenient to use a modified big O notation that suppresses all polynomially bounded factors. For two functions $f$ and $g$, we write $f(n) = O^*(g(n))$ if $f(n) = O(g(n)\mathsf{poly}(n))$, where $\mathsf{poly}(n)$ represents some polynomial in $n$.

## 2.2 Structural graph theory

In this section, we review the basics of structural graph theory, with an emphasis on some structured graph classes.

**Graphs**. A *graph* is an ordered pair $G = (V, E)$ such that $V$ is a finite set, known as the vertex set, and $E$ is a finite set of 2-element subsets of $V$, called the edge set. If $V = E = \emptyset$, the graph $G$ is said to be *empty*. Typically, one refers to the elements of $V$ as *vertices* of the graph $G$, and the elements of $E$ as the *edges* of the graph $G$. Unless mentioned otherwise, all graphs in this thesis are *undirected*, that is, the edges of a graph are unordered pairs. In addition, we will always assume there are no self-loops, that is, $\{v, v\} \notin E$ for all $v \in V$. For a graph $G$, we denote by $V(G)$ and $E(G)$ its vertex set and edge set, respectively. To reduce clutter, an edge $\{u, v\}$ is often denoted as $uv$.

We say two graphs $G$ and $H$ are *isomorphic* if there are bijections $\Phi_V : V(G) \to V(H)$ and $\Phi_E : E(G) \to E(H)$ such that $\Phi_E(vw) = (\Phi_V(v), \Phi_V(w))$ for all $vw \in E(G)$. If $G$ and $H$ are isomorphic, we write $G \simeq H$.

A vertex $v$ is *incident* with an edge $e$ if $v \in e$. Two vertices $x$ and $y$ are *adjacent* (or *neighbors*) if $xy$ is an edge of $G$. Two edges $e \neq f$ are *adjacent* if they have an end in common. The *line graph* of a graph $G$, denoted by $L(G)$, has a vertex for each edge in $G$ with an edge between two vertices if the corresponding edges are adjacent in $G$. The *degree* of a vertex $v$ is the number of edges incident to $v$. The *minimum degree* of a graph $G$ is denoted by $\delta(G)$; sometimes we shorten this to $\delta$ if it is clear from the context what $G$ is. Similarly, we may denote the *maximum degree* of a graph $G$ by $\Delta(G)$, or just $\Delta$.

Let $G = (V, E)$ and $G' = (V', E')$ be two graphs. We say $G'$ is a *subgraph* of $G$ if $V' \subseteq V$ and $E' \subseteq E$, and denote this by $G' \subseteq G$. If $G' \subseteq G$ and $G'$ contains all the edges $xy \in E$ with $x, y \in V'$, then $G'$ is an *induced subgraph* of $G$; we say that $V'$ *induces* $G'$ in $G$, and write $G' = G[V']$. We say a graph $G$ is *$H$-free* if $G$ does not contain graph $H$ as an induced subgraph. When we say $G$ is $(H_1, H_2, \ldots, H_k)$-free, we mean $G$ is $H$-free for each $H_i \in \{H_1, H_2, \ldots, H_k\}$. Let $e = uv$ be an edge of a graph $G$. By $G/e$ we denote the graph obtained from $G$ by *contracting* the edge $e$ into a new vertex $v_e$, which becomes adjacent to all the former neighbors of $u$ and $v$. Finally, we say $H$ is a *minor* of $G$ if $H$ is obtainable from $G$ by deleting edges and vertices and by contracting edges. In particular, we say that a graph $G$ is *$H$-minor-free* if $H$ is not a minor of $G$. When we say $G$ is $(H_1, H_2, \ldots, H_k)$-minor-free, we mean $G$ is $H_i$-minor-free for each $H_i \in \{H_1, H_2, \ldots, H_k\}$.

A *path* is a non-empty graph $P = (V, E)$ of the form $V = \{x_0, x_1, \ldots, x_k\}$, $E = \{x_0x_1, x_1x_2, \ldots, x_{k-1}x_k\}$, where the $x_i$'s are all distinct. The number of edges of a path is its *length.* A path on $n$ vertices is known as a *path graph*, and denoted by $P_n$. A graph is *connected* if there is a path between each pair of its vertices. Otherwise, the graph is *disconnected.* A maximal connected subgraph of $G$ is called a *connected component* (or just *component*) of $G$. A *cut vertex* of $G$ is a vertex whose removal increases the number of number of components of $G$. A graph $G$ is said to be *$k$-vertex-connected* if $G$ remains connected after the removal of any vertex set of size at most $k - 1$. In particular, we may call a 2-vertex-connected graph *biconnected.* Similarly, a *cut edge* (or *bridge*) is an edge whose removal increases the number of connected components of $G$. A graph $G$ is *$k$-edge-connected* if $G$ remains connected after the removal of any edge set of size at most $k - 1$.

For a more thorough treatment, we refer the reader to [24].

**Graph invariants**. Let $G = (V, E)$ be an undirected simple graph. A *vertex coloring* (or just *coloring*) is a function $c : V \to [k]$ assigning a color from $[k]$ to each vertex. The coloring is said to be *proper* if $c(u) \neq c(v)$ for every $uv \in E$. A graph $G$ is said to be *$k$-colorable* if there exists a proper coloring using $k$ colors for it. The minimum $k$ for which a graph $G$ is $k$-colorable is known as its *chromatic number*, denoted by $\chi(G)$. A complete subgraph of $G$ is a *clique.* The *clique number* of a graph $G$, denoted by $\omega(G)$, is the size of a largest clique in $G$. A *vertex cover* of $G$ is a set of vertices such that each edge is incident to at least one vertex in the set. For example, a set of vertices is a vertex cover precisely when its complement forms an *independent set*, that is, a set of pairwise non-adjacent vertices. The *vertex cover number* of a graph $G$, denoted by $\tau(G)$, is the size of a smallest vertex cover in $G$. A subset $S$ of $V$ is called a *dominating set* if every vertex in $V \setminus S$ is adjacent to some vertex in $S$. The *domination number* of a graph $G$, denoted by $\gamma(G)$, is the size of a smallest dominating set in $G$. A dominating set $S$ is called a *connected dominating set* if the graph induced by $S$ is connected. The *connected domination number* of a graph $G$, denoted by $\gamma_c(G)$, is the size of a smallest connected dominating set of the graph $G$. In particular, if $G$ has a dominating set $S = \{v\}$, we might say $v$ is a *dominating vertex.*

**Graph classes**. A *complete graph* on $n$ vertices, denoted by $K_n$, has all the possible

$\binom{n}{2}$ edges. In particular, we will call $K_3$ a *triangle.* A 2-colorable graph is *bipartite.* A *complete bipartite graph* consists of two non-empty independent sets $X$ and $Y$ with $xy$ being an edge whenever $x \in X$ and $y \in Y$. A complete bipartite graph is denoted by $K_{n,m}$, and it has $n + m = |X| + |Y|$ vertices. In particular, we will call $K_{1,3}$ a *claw.* We also denote the graph $K_{1,n}$ as $S_n$, and call it a *star.*

A graph is said to be *planar* if it can be embedded in the plane with no crossing edges. Equivalently, a graph is planar if it is $(K_{3,3}, K_5)$-minor-free. A graph is *outerplanar* if it has a crossing-free embedding in the plane such that all vertices are on the same face. Clearly, each outerplanar graph is planar. Another superclass of outerplanar graphs is formed by *series-parallel graphs.* Series-parallel graphs are exactly the $K_4$-minor-free graphs [28]. In a *cactus graph*, every edge is in at most one cycle. Cactus graphs form a subclass of outerplanar graphs.

A graph is *k-regular* if every vertex has degree exactly $k$. In particular, we will call a 3-regular graph *cubic.* A connected 2-regular graph is a *cycle graph.* A cycle graph on $n$ vertices is denoted by $C_n$. For convenience, if $H \simeq C_n$ is an induced subgraph of a graph $G$ for any $n \geq 1$, we can say $H$ is an *induced cycle* (of $G$).

A *chord* is an edge joining two non-consecutive vertices in a cycle. A graph is *chordal* if every cycle of length 4 or more has a chord. Equivalently, a graph is chordal if it contains no induced cycle of length 4 or more. Chordal graphs are precisely the class of graphs admitting a *clique tree* [37]. A clique tree of a connected chordal graph $G$ is any tree $T$ whose vertices are the maximal cliques of $G$ such that for every two maximal cliques $C_i, C_j$, each clique on the path from $C_i$ to $C_j$ in $T$ contains $C_i \cap C_j$. A subclass of chordal graphs is formed by *interval graphs.* A graph is an interval graph if and only if it admits a clique tree that is a path [38]. In a *block graph*, every maximal biconnected component, known as a *block*, is a clique. In other words, every edge of a block graph $G$ lies in a unique block, and $G$ is the union of its blocks. It is easy to see that block graphs are also chordal. A graph whose vertex set can be partitioned into a clique and an independent set is known as a *split graph.* It is known that each split graph is chordal. Finally, a $(2K_2, C_4, P_4)$-free graph[1] is a *threshold graph*, and they form a subclass of split graphs.

Three vertices of a graph form an *asteroidal triple* if every two of them are connected by a path avoiding the neighborhood of the third. A graph is *AT-free* if it does not contain an asteroidal triple. In general, AT-free graphs are not chordal.

**Width parameters**. Roughly speaking, width parameters measure the closeness of a graph to a particular kind of graph. Such parameters are typically defined through various decompositions that can be exploited algorithmically. However, we deviate from this perhaps standard practice as we will not give explicit algorithms leveraging such decompositions. Instead, we use slightly different but equivalent definitions. A *proper interval graph* (also known as a *unit interval graph*) is a graph that is both interval and claw-free (see [70]). The *bandwidth* of a graph $G$, denoted by $\mathrm{bw}(G)$, is one less than the minimum clique number of any proper interval graph having $G$ as a subgraph [47]. The *pathwidth* of a graph $G$, denoted by $\mathrm{pw}(G)$, is one less than the minimum clique number of any interval graph having $G$ as a subgraph. The *treewidth* of a graph $G$, denoted by $\mathrm{tw}(G)$, is one less than the minimum clique number of any chordal graph having $G$ as a subgraph. Indeed, for a graph $G$, we have that $\mathrm{tw}(G) \leq \mathrm{pw}(G) \leq \mathrm{bw}(G)$ (for a proof, see [3]). Finally, a $(C_4, P_4)$-free graph is *trivially perfect.* The *tree-depth* of a graph $G$,

[1]By $2K_2$ we mean the 4-vertex graph whose both components are a $K_2$.

denoted by $\mathrm{td}(G)$, is the minimum clique number of any trivially perfect graph having $G$ as a subgraph. Here, we have that $\mathrm{pw}(G) \leq \mathrm{td}(G) - 1$ (for a proof, see [4]).

## 2.3 Rainbow connections in graphs

In this section, we briefly survey some combinatorial properties of rainbow colorings. These will be useful already for Chapter 3, but especially for Chapters 4 and 5.

The concept of rainbow coloring was introduced by Chartrand, Zhang, Johns, and McKeon in 2008 [11], while also briefly featured in an earlier book of Chartrand and Zhang [13]. We say a path in an edge-colored graph is *rainbow* if no color repeats on it. A graph is *rainbow-connected* if there is a rainbow path between every pair of its vertices. The minimum number of colors for which a graph $G$ is rainbow-connected is known as its *rainbow connection number*, denoted by $\mathrm{rc}(G)$. Similarly, a graph is said to be *strongly rainbow-connected* if there is a rainbow shortest path between every pair of its vertices. Likewise, the minimum number of colors for which a graph $G$ is strongly rainbow-connected is known as its *strong rainbow connection number*, denoted by $\mathrm{src}(G)$. If a graph $G$ is (strongly) rainbow-connected under some edge-coloring $c : E \to \mathbb{N}$, we might also say $G$ is *(strongly) rainbow colored.* Moreover, such a $c$ can be called a *(strong) rainbow coloring* (of $G$). To avoid confusion, we stress that the rainbow connection numbers are properties of uncolored graphs.

The *diameter* of a graph $G$, denoted by $\mathrm{diam}(G)$, is the length of a longest shortest path in $G$. It is easy to see that to rainbow-connect any connected graph $G$, at least $\mathrm{diam}(G)$ colors are needed. On the other hand, one can use at most $m$ colors, where $m$ stands for the number of edges. Finally, as every strongly rainbow-connected graph is also rainbow-connected, we have that $\mathrm{diam}(G) \leq \mathrm{rc}(G) \leq \mathrm{src}(G) \leq m$, for any connected graph $G$. It is also straightforward to verify that in any (strong) rainbow coloring of $G$, every bridge of $G$ must receive a distinct color (for a proof, see [12]).

Chartrand *et al.* [11] established basic combinatorial properties and determined the exact rainbow connection number for some structured graph classes. For instance, they showed the following.

**Theorem 2.1** (Chartrand *et al.* [11])**.** *Let $G$ be a connected graph with $n$ vertices and $m$ edges.*

- $\mathrm{rc}(G) = \mathrm{src}(G) = 1$ *if and only if $G$ is complete.*
- $\mathrm{rc}(G) = \mathrm{src}(G) = m$ *if and only if $G$ is a tree.*
- $\mathrm{rc}(C_n) = \mathrm{src}(C_n) = \lceil n/2 \rceil$*, for $n \geq 4$.*
- $\mathrm{rc}(G) = 2$ *if and only if* $\mathrm{src}(G) = 2$.

By considering star graphs (that is, $K_{1,n}$ for some $n \geq 1$), one can see the difference between $\mathrm{diam}(G)$ and $\mathrm{rc}(G)$ can be made arbitrarily large. Here, $\mathrm{rc}(G)$ can be replaced by $\mathrm{src}(G)$, as both equal $m$ for a tree.

For complete bipartite graphs, the following results were obtained by Chartrand *et al.* [11].

**Theorem 2.2** (Chartrand *et al.* [11])**.** *Let $K_{s,t}$ be a complete bipartite graph for two integers $s, t \geq 1$.*

- $\mathrm{rc}(K_{s,t}) = \min\{\lceil \sqrt[s]{t} \rceil, 4\}$, *for* $2 \leq s \leq t$.
- $\mathrm{src}(K_{s,t}) = \lceil \sqrt[s]{t} \rceil$, *for* $1 \leq s \leq t$.

Since the work of Chartrand *et al.* [11], there has been significant interest in the concept of rainbow coloring, with over 200 papers published by now on the topic.

For the sake of providing more context for the reader, we highlight some results on the rainbow connection number in the following. It should be noted there are far less results for the strong rainbow connection number. A possible reason, along with some results, are presented in Chapter 5.

**Theorem 2.3.** *Let $G$ be a connected graph with $n$ vertices and $m$ edges. Then,*

- $\mathrm{rc}(G) \leq \lceil n/2 \rceil$, *where $G$ is 2-connected, and this is tight (Ekstein et al. [29]);*
- $\mathrm{rc}(G) \leq \frac{20n}{\delta}$ *(Krivelevich and Yuster [53]); and*
- $\mathrm{rc}(G) \leq \frac{3n}{\delta+1} + 3$ *(Chandran et al. [8]).*

For several combinatorial results not mentioned here, we refer the reader to the survey of Li, Shi, and Sun [58], or the book of Li and Sun [60].

In addition to edge-colored graph, it is natural to consider rainbow connection in vertex-colored graphs. Krivelevich and Yuster [53] introduced the vertex variant of the rainbow connection number in the following way. A path in a vertex-colored graph is *vertex rainbow* (or just *rainbow*, if there is no danger for confusion), if no color repeats on its internal vertices. That is, a path of length at most two is always vertex rainbow regardless of the underlying vertex-coloring. A graph is *rainbow vertex-connected* if there is a vertex rainbow path between each pair of its vertices. The minimum number of colors for which a graph $G$ is rainbow vertex-connected is known as its *vertex rainbow connection number*, denoted by $\mathrm{rvc}(G)$. In a way analogous to the strong rainbow connection number, Li, Mao, and Shi [57] introduced the *strong vertex rainbow connection number* of a graph $G$, denoted by $\mathrm{srvc}(G)$. It can be verified that $\mathrm{diam}(G) - 1 \leq \mathrm{rvc}(G) \leq \mathrm{srvc}(G) \leq n - 2$ (for a proof, see [57]).

The reader might wonder about the motivation behind the definition of the rainbow vertex connection numbers. In particular, why does one restrict the rainbow property to hold only on the internal vertices of the graph? Suppose we did instead require each vertex on a path to hold a distinct color. Clearly, both $\mathrm{rvc}(G)$ and $\mathrm{srvc}(G)$ would then equal $n$, as we need a distinct color for each vertex. Thus, the parameters become uninterestingly high and it appears more sensible to use the definition given in [53]. For the problem of *verifying* whether a vertex rainbow path exists between two vertices $s$ and $t$, the definitions can be shown to be computationally equivalent.

It is natural to ask for connections between the edge and vertex variants of the rainbow connection numbers. For instance, is one an upper bound for the other? Are the two parameters connected through say the line graph of the graph? For the first question, the answer is no. For the $n$-vertex star graph $G \simeq S_n$, we have $\mathrm{rc}(G) = \mathrm{src}(G) = m$. However, as $\mathrm{diam}(G) = 2$, we have $\mathrm{rvc}(G) = \mathrm{srvc}(G) = 1$. To see that $\mathrm{rc}(H) < \mathrm{rvc}(H)$ is possible, consider the graph $H$ obtained by attaching a triangle to each vertex of $K_n$ (see Figure 2.1). Here, $H$ has $n$ cut vertices, and thus $\mathrm{rvc}(H) = n$ is easy to prove. On the other hand, it is not difficult to see that $\mathrm{rc}(H) \leq 4$. For the vertex variants, it is true

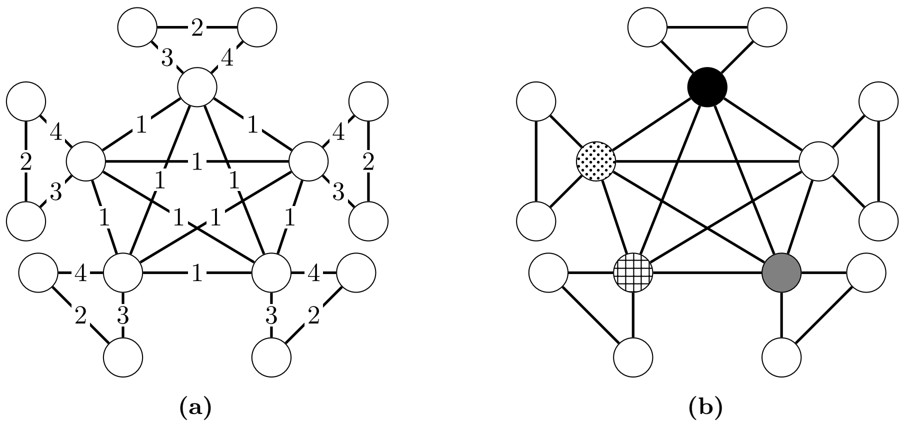


**Figure 2.1:** The graph $H$ formed by attaching a triangle to each vertex of $K_n$ for $n = 5$. It holds that **(a)** $\mathrm{rc}(H) \leq 4$, but **(b)** $\mathrm{rvc}(H) = n$.

that $\mathrm{srvc}(G) \leq \mathrm{src}(G)$ when $\mathrm{diam}(G) \leq 2$. In general, it seems unknown whether the statement is true for graphs of diameter at least 3.

In general, it is unknown what the relationship between say $\mathrm{rc}(G)$ and $\mathrm{rc}(L(G))$ is, where $L(G)$ is the line graph of $G$. In fact, this is explicitly mentioned as an open problem in [58]. For some results concerning $\mathrm{rc}(L(G))$, see [59, 61].

## 2.4 Fixed-parameter tractability

Traditionally, when analyzing algorithms, one assumes a one-dimensional view: the time taken by an algorithm is measured as a function of the input size. However, it might be beneficial to analyze an algorithm in terms of other parameters besides input size. Indeed, virtually every problem is filled with parameters that reflect its structure. Especially for a graph problem, there are several parameters reflecting the topology or shape of a graph. Such parameters include, among others, treewidth, clique number, chromatic number, and several distance parameters.

In parameterized complexity we take a two-dimensional view: how does an algorithm perform when measured by both input size and some structural parameter that is independent of the input size? In other words, we seek to understand the contribution of such parameters to the overall complexity of a problem. In particular, for computationally difficult problems,[2] the aim is to investigate whether the exponential worst-case complexity of a problem can be isolated into the additional parameter. We will briefly introduce the central concepts of parameterized complexity. For a comprehensive treatment, we refer the reader to [22, 27, 68].

In order to perform such two-dimensional analysis, we need to make precise the idea of measuring not only by input size, but with an additional independent parameter.

**Definition 2.4.** A *parameterized problem* is a language $L \subseteq \Sigma^* \times \mathbb{N}$, where $\Sigma$ is a fixed, finite alphabet. For an instance $(x, k) \in \Sigma^* \times \mathbb{N}$, we call $k$ the *parameter*.

[2]By no means is the analysis limited to hard problems —such an analysis for a problem in P can also be fruitful.

For a graph-theoretic example, consider the problem Chromatic Number parameterized by solution size $k$. That is, the goal is to decide whether a given graph $G$ is $k$-colorable. An instance $(G, k)$ belongs to the Chromatic Number language precisely when the string $G$ encodes a valid undirected graph, and $G$ is $k$-colorable. For optimization problems, the solution size is the "standard parameter". One might also parameterize by some structural parameter, say maximum degree or treewidth. In general, choosing an interesting parameter is an art.

Traditionally, an algorithm is thought to be efficient if it runs in time polynomial in the input size. The following gives similar rough intuition for an algorithm solving a parameterized problem.

**Definition 2.5.** A parameterized problem $L \subseteq \Sigma^* \times \mathbb{N}$ is called fixed-parameter tractable (FPT) if there exists an algorithm $\mathcal{A}$ (called a *fixed-parameter algorithm*), a computable nondecreasing function $f : \mathbb{N} \to \mathbb{N}$, 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 $f(k) \cdot |x|^c$. The complexity class containing all fixed-parameter tractable problems is called $\mathsf{FPT}$.

Of course, precisely like an algorithm running in time $n^{100}$ is not efficient in practice, an algorithm solving a parameterized problem in time $10^{10^k} n$ is not efficient in practice either. However, it is perhaps fair to say that when a problem is deemed to be FPT, faster FPT algorithms can be found, e.g., the function $f(k)$ can be improved upon.

Consider then the following problem known as Clique: given an undirected graph $G$ and an integer $k$, decide whether $G$ contains a clique on $k$ vertices. It is easy to observe the problem is solvable in polynomial time for any fixed $k$. Indeed, using the solution size $k$ as a parameter, the obvious algorithm tries all the $\binom{n}{k}$ vertex subsets, thus running in time $\Theta(n^k)$. The following definition captures such parameterized algorithms.

**Definition 2.6.** A parameterized problem $L \subseteq \Sigma^* \times \mathbb{N}$ is called *slice-wise polynomial* (XP) if there exists an algorithm $\mathcal{A}$ and two computable nondecreasing functions $f, g : \mathbb{N} \to \mathbb{N}$ 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 $f(k) \cdot |x|^{g(k)}$. The complexity class containing all slice-wise polynomial time problems is called $\mathsf{XP}$.

The brute-force algorithm places Clique in $\mathsf{XP}$. However, there are several problems, such as Clique, for which we do not know FPT algorithms for. In fact, it is common belief that there are problems, including Clique, for which no FPT algorithm exists (under reasonable complexity assumptions). This raises a question: how does one show Clique or some other parameterized problem is (unlikely) to be FPT? To this end, let us mention *parameterized reductions*. A parameterized reduction from a parameterized decision problem $L$ to a parameterized decision problem $L'$ is an algorithm that transforms an instance $(I, k)$ of $L$ into an instance $(I', g(k))$ of $L'$ in time $f(k)|I|^{O(1)}$ where $f$ and $g$ are computable functions such that $(I, k)$ is a YES-instance of $L$ if and only if $(I', g(k))$ is a YES-instance of $L'$. The class of problems reducible to Clique under parameterized reductions is denoted by $\mathsf{W}[1]$. We define hardness and completeness analogously to classical complexity, but assume parameterized reductions. A problem is said to be $\mathsf{W}[1]$-*hard* if Clique (and thus each problem in $\mathsf{W}[1]$) can be reduced to it by a parameterized reduction. It is widely believed that $\mathsf{FPT} \neq \mathsf{W}[1]$, while $\mathsf{FPT} \neq \mathsf{XP}$ is known to be true (see [27]). For a more thorough introduction to fixed-parameter intractability, we again refer the reader to [22].

Finally, we do not believe every problem would even be in XP for a given parameter. Let us go back to the problem Chromatic Number. It is well-known the problem is NP-complete for every $k \geq 3$. Thus, it is easy to imagine the consequences of an XP algorithm for Chromatic Number: if we could solve Chromatic Number in time $f(k) \cdot n^{g(k)}$ for an $n$-vertex graph, then $\mathsf{P} = \mathsf{NP}$.

## 2.5 Lower bounds on exact algorithms

Every problem in NP can be solved in time exponential in the input size by a brute-force algorithm. Indeed, unless $\mathsf{P} \neq \mathsf{NP}$, no NP-complete problem has a polynomial-time algorithm. However, many NP-complete problems have exponential-time algorithms that are considerable faster than a naive brute-force algorithm. For example, consider the Subset Sum problem: given a set of $n$ integers $U$ and a target integer $t$, is there a subset $U' \subseteq U$ such that the sum of the elements in $U'$ equals $t$? The obvious algorithm runs in $O^*(2^n)$ time by trying all of the $2^n$ subsets. However, already in the early 70s Horowitz and Sahni [41] gave the following algorithm running in $O^*(2^{n/2})$ time. Let us give the idea behind their algorithm. First, split the input set $U$ into two parts of size roughly $n/2$ both. Then, let $Q$ and $S$ contain all possible $2^{n/2}$ sums of the first part and the second part, respectively. Finally, sort $S$, and for each sum $q \in Q$, perform a binary search for the integer $t - q$ in $S$. Correctness of the algorithm is straightforward to establish. A natural question arises: can one do even better? More generally, assuming $\mathsf{P} \neq \mathsf{NP}$, how fast of an algorithm can one hope to have for a particular problem?

For exponential lower bounds, it seems unavoidable to introduce a conjecture stronger than $\mathsf{P} \neq \mathsf{NP}$. For concreteness, suppose we have an algorithm running in $O^*(2^n)$ time for some NP-complete problem. If we only assume $\mathsf{P} \neq \mathsf{NP}$, how could we rule out an algorithm running in $O^*(c^n)$ time, for some $c < 2$? To this end, let us introduce the Exponential Time Hypothesis (ETH) of Impagliazzo and Paturi [43].

**Conjecture 2.7** (Exponential Time Hypothesis (ETH), [43])**.** *There exists a constant $c > 0$, such that there is no algorithm for solving* 3-SAT *in time $O^*(2^{cn})$, where $n$ is the number of variables.*

How does introducing ETH help in excluding algorithms that are considerably faster than the obvious algorithm? First, let us observe the consequences of a linear-size reduction from 3-SAT to a target problem. That is, such a reduction outputs an instance $I'$ of the target problem whose size is $O(n + m)$, where $n$ denotes the number of variables, and $m$ the number of clauses. Now, observe that if the target problem could be solved in time $2^{o(|I'|)}$, we could solve 3-SAT in time $2^{o(n+m)}$. Let us explain how this contradicts ETH. It can be seen that a 3-SAT instance can have up to $\Theta(n^3)$ distinct clauses. Thus, it seems that any reduction from 3-SAT unavoidably outputs an instance of a target problem that has size at least cubic in the number of input variables. Fortunately, a sparsification lemma of Impagliazzo, Paturi, and Zane [44] makes it possible to sparsify a given 3-SAT formula such that the number of clauses is linear in the number of variables. We skip many details, but it follows that we can safely assume that when reducing from 3-SAT, the number of clauses is linear in the number of variables (for details, see e.g., [22]). Thus, if we could solve 3-SAT in time $2^{o(n+m)}$, we would break ETH. Finally, let us explain the general scheme of obtaining hardness results under ETH. That is, suppose a reduction takes an instance $I$ of 3-SAT and outputs an instance of a target problem of size at most $g(|I|)$ for some nondecreasing function $g$. Then, a $O^*(2^{o(f(|I|))})$-time algorithm for the target problem implies a $O^*(2^{o(f(g(|I|)))})$-time algorithm for 3-SAT. Such an algorithm is obtained by composing the reduction with an algorithm for the target problem.

Previously, assuming ETH, it has been shown that there is no $2^{o(n \log n)}$-time algorithm for Channel Assignment [71], as well as for many embedding problems including Subgraph Isomorphism and Graph Minor [21]. For a survey on lower bounds under ETH, we refer the reader to [62].

Finally, yet another conjecture is based on the following Set Cover problem. In this problem, we are given an integer $f$ and a family of sets $\mathcal{S}$ over the universe $U = \bigcup \mathcal{S}$ with $n = |U|$ and $m = |\mathcal{S}|$. The goal is to decide whether there is a subfamily of at most $f$ sets $S_1, S_2, \ldots, S_f \in \mathcal{S}$ such that $U = \bigcup_{i=1}^{f} S_i$.

**Conjecture 2.8** (Set Cover Conjecture, [20])**.** *There is no algorithm for the* Set Cover *problem that runs in time $(2-\varepsilon)^n (nm)^{O(1)}$ for any $\varepsilon > 0$.*

In fact, a dynamic programming algorithm of Fomin, Kratsch, and Woeginger [32] (designed to solve the minimum dominating set problem on split graphs) decides Set Cover in time $O^*(2^n)$. An improved algorithm running in time $O^*((2-\varepsilon)^n)$, for any $\varepsilon > 0$, has been deemed a major breakthrough after decades of research on the problem. It turns out that under this assumption, we can establish tight lower bounds for several well-known parameterized problems. In particular, the following is known.

**Theorem 2.9** ([20, 2])**.** *Unless the Set Cover Conjecture fails,*

- Steiner Tree *cannot be solved in time $O^*((2-\varepsilon)^\ell)$ for any $\varepsilon > 0$, where $\ell$ is the target size of the tree;*
- Connected Vertex Cover *cannot be solved in time $O^*((2-\varepsilon)^k)$ for any $\varepsilon > 0$, where $k$ is the target size of the solution;*
- Set Partitioning *cannot be solved in time $O^*((2-\varepsilon)^n)$ for any $\varepsilon > 0$, where $n$ is the size of the universe;*
- Subset Sum *cannot be solved in time $O^*((2-\varepsilon)^m)$ for any $\varepsilon > 0$, where $m$ is the number of bits of the encoding of the target sum $t$; and*
- Graph Motif *cannot be solved in time $O^*((2-\varepsilon)^k)$ for any $\varepsilon > 0$, where $k$ is the size of the solution.*

For each of the above problems, non-trivial algorithms with matching upper bounds are known (see [20] for the references). All of these algorithms are based on dynamic programming, except for the one for Graph Motif, which takes an algebraic approach. For this reason, it is suggested in [20] that Set Cover is the "canonical" dynamic programming problem. Informally, it seems the Set Cover Conjecture is a suitable conjecture for deriving exponential lower bounds for problems admitting natural dynamic programming algorithms. Naturally, it is of considerable interest to find supporting evidence for Conjecture 2.8 (some evidence is provided in [20]). However, even if Conjecture 2.8 is false, the lower bounds of Theorem 2.9 are meaningful. That is, instead of trying to improve upon the fastest known algorithms for the problems of Theorem 2.9, one should focus on the more basic Set Cover problem.

3

# Hardness of finding rainbow paths

Suppose we have a computational problem, and we prove it to be NP-complete. Why is it interesting to seek further NP-completeness results for the problem on restricted inputs? From a practical viewpoint, it is worth asking: when will it be the case that the input has no structure, i.e., is truly general? The answer is almost never. For instance, graphs modeling road networks, railway systems, electric printed circuits, or chemical molecules are (typically) planar. Such structural information might enable us to devise a practical polynomial-time algorithm for a problem that would otherwise be intractable.

Another point is the construction of further hardness results for other problems. Of course, we can always build a valid reduction from 3-SAT to show NP-hardness of a target problem. However, depending on the target problem, such a reduction might be far from obvious, or very tedious to describe. Thus, it is beneficial to have a wide selection of source problems to reduce from, to make the "easy reduction". This is perhaps particularly so when we are trying to show NP-hardness of a problem on some restricted input.

In this chapter, we focus on the problem of verifying whether a given graph is rainbow-connected. The aim is to pinpoint as precisely as possible the "hardness barrier", that is, the strongest possible restriction of the input for which the problem remains NP-complete. These hardness results will then be complemented by FPT algorithms. We conclude the chapter by considering lower bounds for some FPT algorithms, and ask: how fast of an algorithm can one hope for?

## 3.1 Hardness barriers

The following four problems are the main focus of this section. We first define the two problems defined on edge-colored graphs.

Rainbow Connectivity (RCon)

**Instance:** A connected graph $G = (V, E)$, and an edge-coloring $\zeta : E \to C$, where $C$ is a set of colors.

**Problem:** Is $G$ rainbow-connected under $\zeta$?

Strong Rainbow Connectivity (SRCon)

**Instance:** A connected graph $G = (V, E)$, and an edge-coloring $\zeta : E \to C$, where $C$ is a set of colors.

**Problem:** Is $G$ strongly rainbow-connected under $\zeta$?

The two vertex variants are then defined analogously.

Rainbow Vertex Connectivity (RVCon)

**Instance:** A connected graph $G = (V, E)$, and a vertex-coloring $\psi : V \to C$, where $C$ is a set of colors.

**Problem:** Is $G$ rainbow vertex-connected under $\psi$?

Strong Rainbow Vertex Connectivity (SRVCon)

**Instance:** A connected graph $G = (V, E)$, and a vertex-coloring $\psi : V \to C$, where $C$ is a set of colors.

**Problem:** Is $G$ strongly rainbow vertex-connected under $\psi$?

Clearly, if the number of colors $k = |C|$ is bounded by a constant, each of the four problems can be solved in polynomial time. Indeed, we get an upper bound on the length of a rainbow path, and the brute-force algorithm will run in polynomial time.

The Rainbow Connectivity problem was shown to be NP-complete by Chakraborty, Fischer, Matsliah, and Yuster [7]. For the vertex variant, namely Strong Rainbow Vertex Connectivity, hardness was established by Chen, Li, and Shi [15]. A more fine-grained study into the complexity of the problems for restricted graph classes was performed by Uchizawa, Aoki, Ito, Suzuki, and Zhou [74]. In particular, they gave a reduction from a variant of 3-SAT, known as 3-Occurrence 3-SAT. In this variant, we have the additional constraint that each variable appears at most three times in the given formula. For this problem, it is crucial to let clauses be of size two *and* three. In fact, if every clause was of size three, the instance would always be satisfiable as shown by Tovey [73].

We capitalize on the reduction idea of Uchizawa *et al.* [74], and obtain an even more thorough view of the hardness barrier of the problem. Most of our reductions are heavily inspired by their reduction for Rainbow Connectivity. In addition, our results have consequences for the (non)existence of FPT algorithms for various structural parameters.

The key idea behind the reductions is the following. Given a 3-Occurrence 3-SAT formula $\varphi$, we construct a variable gadget for each variable, and a clause gadget for each clause. Each gadget is colored so that regardless of the satisfiability of $\varphi$, each vertex

**Table 3.1:** Summary of known complexity results for rainbow connectivity problems. Results originating from this thesis are marked with ★.

| Graph class | RCon | SRCon | RVCon | SRVCon |
|---|---|---|---|---|
| All | NPC | NPC | NPC | NPC★ |
| Bipartite planar | NPC | NPC | NPC★ | NPC★ |
| Cactus | P | P | P | P★ |
| Interval | NPC★ | NPC★ | NPC★ | NPC★ |
| Interval block | NPC★ | P★ | P★ | P★ |
| Interval outerplanar | NPC★ | NPC★ | P | ? |
| $k$-regular, $k \geq 3$ | NPC★ | NPC★ | NPC★ | NPC★ |
| Outerplanar | NPC | NPC | P | ? |
| Series-parallel | NPC | NPC | NPC | NPC★ |
| Split | ? | P★ | ? | P★ |
| Tree | P | P | P | P |
| Unit interval | NPC★ | NPC★ | ? | NPC★ |

pair in a gadget is rainbow-connected. In fact, each vertex pair will be rainbow-connected except for a specific vertex pair $s$ and $t$. Informally, we set up the gadgets in a path-like manner, and $s$ and $t$ are the respective endpoints of this path-like graph. Then, the constructed graph is (strongly) rainbow-connected if and only if there is a rainbow (shortest) path between $s$ and $t$. This approach establishes the following results.

**Theorem 3.1.** *Both of the problems* Rainbow Connectivity *and* Strong Rainbow Connectivity *are* NP*-complete for interval outerplanar graphs and $k$-regular graphs for $k \geq 3$.*

**Theorem 3.2.** *Both of the problems* Rainbow Vertex Connectivity *and* Strong Rainbow Vertex Connectivity *are* NP*-complete for*

- *bipartite planar graphs of maximum degree 3,*
- *interval graphs,*
- *$k$-regular graphs for $k \geq 3$, and*
- *series-parallel graphs.*

Prior to our results, no graph class was known for which the complexity of Rainbow Connectivity and Strong Rainbow Connectivity would differ (the same is true for the vertex variants). We show the following.

**Theorem 3.3.** *For block graphs,* Rainbow Connectivity *is* NP*-complete while* Strong Rainbow Connectivity *is in* P*. Moreover,* Rainbow Connectivity *remains* NP*-complete for interval block graphs.*

The known complexity results along with our new results are summarized in Table 3.1.

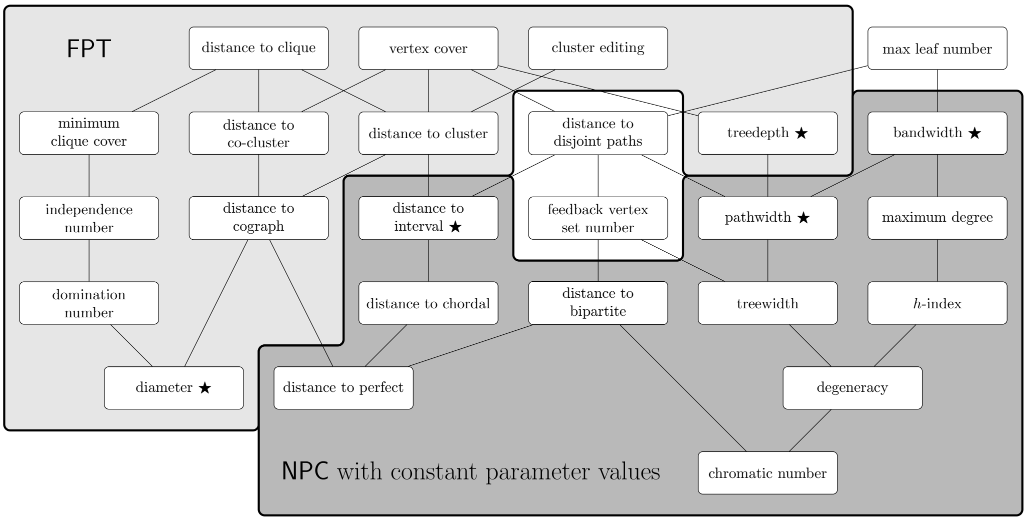


**Figure 3.1:** The FPT landscape for the strong rainbow connectivity problems. A line between two parameters means the parameter below can be polynomially upper-bounded in the parameter above. Thus, positive results propagate upwards, while negative results spread downwards. Results originating from this thesis are marked with ★.

## 3.2 A charting of the FPT landscape

The reductions of the previous section have negative consequences for parameterized algorithms. By inspecting the constructions, we observe the following.

**Theorem 3.4.** *Each of the problems* Rainbow Connectivity*,* Strong Rainbow Connectivity*,* Rainbow Vertex Connectivity*, and* Strong Rainbow Vertex Connectivity *is* NP*-complete for graphs of*

- *bandwidth $b \geq 3$, and*
- *pathwidth $p \geq 3$.*

*For* Rainbow Connectivity *and* Strong Rainbow Connectivity*, the same is true for $b \geq 2$ and $p \geq 2$.*

Graphs of pathwidth 1 are precisely the $(K_3, K_{1,3})$-free graphs, i.e., disjoint unions of paths (for more, we refer the reader to [25]). Moreover, it is well-known graphs of treewidth 1 are exactly forests. We obtain the following dichotomy result.

**Corollary 3.5.** *The problems* Rainbow Connectivity *and* Strong Rainbow Connectivity *are solvable in polynomial time when the input graph has treewidth 1, and are* NP*-complete otherwise. The same is true when treewidth is replaced with pathwidth.*

We summarize our complexity results for Strong Rainbow Connectivity and Strong Rainbow Vertex Connectivity in a Hasse diagram in Figure 3.1. In the figure, the parameters "distance to $X$" measure the number of vertices to be deleted to obtain a graph belonging to class $X$. Similarly, we visualize our results for Rainbow Connectivity and Rainbow Vertex Connectivity in Figure 3.2.

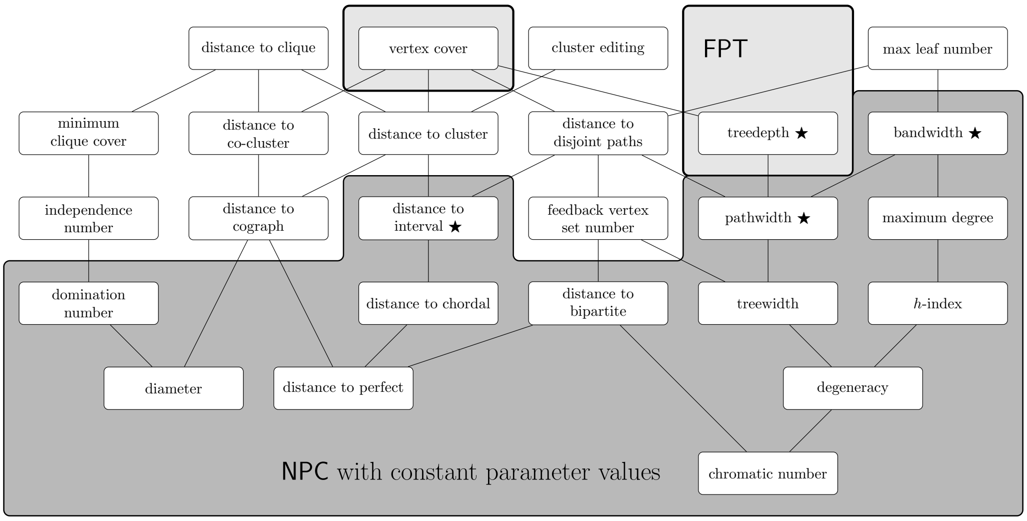


**Figure 3.2:** The FPT landscape for the rainbow connectivity problems. A line between two parameters means the parameter below can be polynomially upper-bounded in the parameter above. Thus, positive results propagate upwards, while negative results spread downwards. Results originating from this thesis are marked with ★.

It can be observed that all four problems are in XP parameterized by the number of colors $k$. Indeed, any rainbow path is of length at most $k$ when there are $k$ colors in the coloring. It is natural to ask whether the problems are FPT as well. The question was answered in the affirmative by Uchizawa *et al.* [74]. In particular, they gave a dynamic programming algorithm running in $O^*(2^k)$ time and exponential space. Let us explain briefly the idea behind their algorithm. The plan is to compute, for each vertex $v \in V$, rainbow walks of length at most $k$ originating from $v$. In particular, there is a rainbow walk from $v$ to $v'$ with length $\ell$ if and only if there is a vertex $u \in N(v')$ such that there is a rainbow walk from $v$ to $u$ of length $\ell - 1$ that does not use color assigned to $uv'$. For each distance (up to $k$), the algorithm computes families of at most $2^k$ color sets using dynamic programming. A rainbow path, if existing, can then be obtained as a sub-walk of a rainbow walk.

The crucial observation is that the number of colors $k$ gives us an upper bound on the length of a walk to compute. Moreover, we observe that if certain parameters are bounded, we similarly get an upper bound on the length of any rainbow walk (and thus, rainbow path). For instance, it is shown in [67] that the length of a longest path in an (undirected) graph $G$ is upper bounded by $2\,\mathrm{td}(G) - 2$, where $\mathrm{td}(G)$ stands for the treedepth of $G$. We arrive at the following observation.

**Theorem 3.6.** *Each of the problems* Rainbow Connectivity, Strong Rainbow Connectivity, Rainbow Vertex Connectivity, *and* Strong Rainbow Vertex Connectivity *is FPT parameterized by the treedepth of the input graph.*

In a similar spirit, bounding the diameter gives us an upper bound of the length of a longest shortest path. Thus, the following is immediate.

**Theorem 3.7.** *Both problems* Strong Rainbow Connectivity *and* Strong Rainbow Vertex Connectivity *are FPT parameterized by the diameter of the input graph.*

This implies polynomial-time solvability of the strong variants for e.g., split graphs and $P_4$-free graphs (also known as *cographs*).

## 3.3 On fast algorithms for solving rainbow connectivity

Uchizawa *et al.* [74] gave algorithms running in $2^k n^{O(1)}$ time and exponential space for all of the four problems considered, namely, Rainbow Connectivity, Strong Rainbow Connectivity, Rainbow Vertex Connectivity, and Strong Rainbow Vertex Connectivity. However, the major downside of their approach is that it uses exponential space. Especially for a practical implementation, exponential space usage is prohibitive. We improve upon their algorithms by showing that Rainbow Connectivity and Rainbow Vertex Connectivity can be solved within the same time bound and in polynomial space. In particular, our algorithm will solve the $s$-$t$ version of the problems, that is, it will decide whether there is a rainbow path between two vertices $s$ and $t$. Let us call this problem Rainbow $st$-Connectivity. Finally, we will consider the possibility of obtaining an $O^*(c^k)$-time algorithm for the problem, for any $c < 2$.

Our algorithm will be based on a polynomial-space algorithm for the following Edge-Colorful Walk problem: given a $k$-edge-colored graph $G$ and two vertices $s$ and $t$, is there a colorful $s$-$t$ walk, that is, a walk using each of the $k$ colors exactly once? We observe the following.

**Lemma 3.8.** Rainbow $st$-Connectivity *with parameter $k$ reduces in polynomial time to* Edge-Colorful Walk *with parameter $k$, where $k$ is the number of colors.*

Using the inclusion-exclusion principle, we then show the following.

**Theorem 3.9.** Edge-Colorful Walk *can be solved in $k2^k m$ deterministic time and $O(n)$ space.*

Finally, we get the following by combining the previous theorem with Lemma 3.8.

**Theorem 3.10.** Rainbow $st$-Connectivity *can be solved in $k2^k(m+k^2)$ deterministic time and $O(n+k)$ space.*

We employ a similar approach for the vertex variant of Rainbow $st$-Connectivity, known as Rainbow Vertex $st$-Connectivity. In this problem, the input is a vertex-colored graph $G$ and two vertices $s$ and $t$. The goal is to verify whether there is a vertex rainbow path between $s$ and $t$. We show the problem reduces in polynomial time to Colorful Path. By an algorithm similar to the one given in Theorem 3.9, we arrive at the following.

**Theorem 3.11.** Rainbow Vertex $st$-Connectivity *can be solved in $k2^k(m+k^2)$ deterministic time and $O(n+k)$ space.*

Given these positive results, it is natural to ask whether one can have an even faster algorithm, perhaps regardless of its space complexity. While we are unable to settle this question, we can at least show that one is unlikely to obtain a faster algorithm by speeding-up our algorithm for Edge-Colorful Walk. We recall the classical Set Cover problem. In this problem, we are given an integer $f$ and a family of sets $\mathcal{S}$ over the universe $U = \bigcup \mathcal{S}$ with $n = |U|$ and $m = |\mathcal{S}|$. The goal is to decide whether there is a subfamily of at most $f$ sets $S_1, S_2, \ldots, S_f \in \mathcal{S}$ such that $U = \bigcup_{i=1}^{f} S_i$. To establish the following result, we construct a polynomial-time reduction from Set Cover to Edge-Colorful Walk, such that the number of colors $k$ is equal to $n + f + O(1)$. Given such a reduction, the result follows from the work of Cygan *et al.* [20]. Indeed, it is observed in [2] that although stated differently in [20], it follows that if Set Cover can be solved in $(2-\varepsilon)^{n+f}(nm)^{O(1)}$ time for some $\varepsilon > 0$, then it can also be solved in $(2-\varepsilon')^{n}(nm)^{O(1)}$ time, for some $\varepsilon' > 0$. In addition, there are many variants of Edge-Colorful Walk that the following theorem captures as well.

**Theorem 3.12.** *For every $\varepsilon > 0$,* Edge-Colorful *st*-Path *does not admit a $(2-\varepsilon)^k n^{O(1)}$-time algorithm, unless* Set Cover *admits a $(2-\varepsilon')^n(nm)^{O(1)}$-time algorithm, for some $\varepsilon' > 0$. The same applies to other variants of the problem* Edge-Colorful *st*-Path*, i.e., for any problem $A$-*Colorful *$BC$, where $A \in \{$*Edge*, *Vertex*$\}$, $B \in \{st, \lambda\}$, and $C \in \{$*Path*, *Walk*$\}$, where $\lambda$ denotes the empty string.*

4

# Algorithmic aspects of rainbow coloring graphs

For this chapter, we turn our attention to finding colorings, instead of “verifying” them. We begin by obtaining new negative results for rainbow coloring. In particular, we will show NP-completeness results for restricted graph classes. As by-products, we also obtain hardness results for approximating the rainbow connection numbers. Then, we move on to lower bound results which hold under the Exponential Time Hypothesis (ETH).

Motivated by such strong hardness results, we consider rainbow coloring from a parameterized perspective. We show that the problems become tractable when certain structural parameters of the input graphs are bounded. We will also consider the parameterized dual problem, i.e., the problem of saving $k$ colors from the trivial upper bound.

Finally, we conclude the chapter by considering problem variants, where we relax the constraint that all vertex pairs need to be rainbow-connected.

## 4.1 Hardness and lower bounds for rainbow coloring

In this section, we consider the problem of rainbow coloring graphs. We begin by formally defining the problems we study. Note that in contrast to rainbow connectivity problems, the input graphs are uncolored.

Rainbow $k$-Coloring ($k$-RC)

**Instance:** A connected undirected graph $G = (V, E)$.
**Problem:** Is $\mathrm{rc}(G) \leq k$?

Strong Rainbow $k$-Coloring ($k$-SRC)

**Instance:** A connected undirected graph $G = (V, E)$.
**Problem:** Is $\mathrm{src}(G) \leq k$?

The two vertex variants, namely Rainbow Vertex $k$-Coloring ($k$-RVC) and Strong Rainbow Vertex $k$-Coloring ($k$-SRVC), are defined analogously for $\mathrm{rvc}(G)$ and $\mathrm{srvc}(G)$, respectively. We will also consider generalized versions of these four problems, where $k$ is given as part of the input.

Rainbow Coloring (RC)

**Instance:** A connected undirected graph $G = (V, E)$, and a positive integer $k$.
**Problem:** Is $\mathrm{rc}(G) \leq k$?

The three other problems (where $k$ is given as part of the input) — namely SRC, RVC, and SRVC — are defined analogously.

Caro, Lev, Roditty, Tuza, and Yuster [6] conjectured that $k$-RC is NP-complete for $k = 2$. The conjecture was settled in the affirmative by Chakraborty *et al.* [7]. In fact, Ananth, Nasre, and Sarpatwar [1] observed their construction shows hardness for every even $k \geq 2$, and complemented this by showing hardness for odd values of $k$ as well. While the reduction of [7] requires a few intermediate steps, a more direct proof of hardness for every $k \geq 2$ is provided by Le and Tuza [55].

Through a series of reductions, Chakraborty *et al.* [7] showed that the following Subset Rainbow $k$-Coloring problem is NP-complete: given a graph $G = (V, E)$ and a set of pairs $P \subseteq V(G) \times V(G)$, decide if the edges of $G$ can be colored in 2 colors such that all pairs $(u, v) \in P$ are rainbow-connected. Then, they reduced this problem to $k$-RC for $k = 2$. In particular, a useful idea was the following: for every $(u, v) \notin P$, add a new vertex $x_{uv}$, and create a path $u, x_{uv}, v$. The key idea is that this path will not make any pair in $P$ rainbow-connected, but we can color it in 2 colors to rainbow-connect $u$ and $v$. This idea has been applied by different authors for showing hardness results on rainbow coloring problems.

By a reduction from Chromatic Number, Ananth *et al.* [1] showed Subset Rainbow $k$-Coloring is NP-complete for every $k \geq 3$ when $G$ is restricted to be a star. Thus, one can without loss assume the input graph $G$ of Subset Rainbow $k$-Coloring is a star. We combine this strengthening of the problem with the above idea of Chakraborty *et al.* [7], and obtain the following.

**Theorem 4.1.** *The problem $k$-SRC is NP-complete for every $k \geq 3$ when restricted to split graphs with a dominating vertex.*

*Proof.* Let $I = (S, P, k)$ be an instance of the Subset Rainbow $k$-Coloring problem, where $S = (V, E)$ is a star, both $p$ and $q$ in each $(p, q) \in P$ are leaves of $S$, and $k \geq 3$ is an integer. We construct an instance $I' = (G')$ of $k$-SRC, where $G' = (V', E')$ is a split graph such that $I$ is a YES-instance of Subset Rainbow $k$-Coloring iff $I'$ is a YES-instance of $k$-SRC.

Let $a$ be the central vertex of $S$. For every vertex $v \in V \setminus \{a\}$, we add a new vertex $x_v$, and for every pair of leaves $(u, v) \in (V \times V) \setminus P$, we add a new vertex $x_{(u,v)}$. Formally, we construct $G' = (V', E')$ such that

- $V' = V \cup \{x_v \mid v \in V \setminus \{a\}\} \cup \{x_{(u,v)} \mid (u, v) \in (V \times V) \setminus P\}$,
- $E' = E \cup E_1 \cup E_2 \cup E_3$,

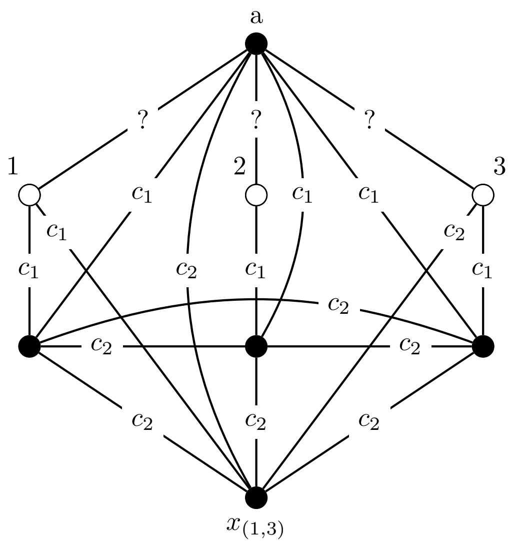


**Figure 4.1:** A star graph $S$ on the vertex set $\{a, 1, 2, 3\}$ transformed to a split graph $G'$ with $P = \{(1, 2), (2, 3)\}$. The white vertices form an independent set while the black vertices form a clique. The symbol ? marks an edge-coloring $\chi$ of $S$ with $k$ colors under which the pairs in $P$ are connected by a rainbow path.

- $E_1 = \{vx_v, ax_v \mid v \in V \setminus \{a\}\}$,
- $E_2 = \{ux_{(u,v)}, vx_{(u,v)}, ax_{(u,v)} \mid (u, v) \in (V \times V) \setminus P\}$, and
- $E_3 = \{xx' \mid x, x' \in V' \setminus V\}$.

Let us then verify $G'$ is a split graph. Observe the leaves of $S$ form an independent set in $G'$. The remaining vertices $\{a\} \cup (V' \setminus V)$ form a clique, proving $G'$ is split. Moreover, $a$ is a dominating vertex. An example illustrating the construction is given in Figure 4.1.

We will now prove $G'$ is strongly rainbow-colorable with $k$ colors if and only if $(S, P)$ is $k$-subset strongly rainbow-connected. First, suppose $(S, P)$ is not $k$-subset strongly rainbow-connected; we will show $G'$ is not strongly rainbow-colorable with $k$ colors. Observe that for each $(p, q) \in P$, there is a unique shortest path between $p$ and $q$ in $S$. Moreover, the same holds for $G'$. Therefore, any strong rainbow coloring using $k$ colors must make this path strongly rainbow-connected in $G'$. But because the pairs in $P$ cannot be strongly rainbow-connected with $k$ colors in $S$, the graph $G'$ cannot be strongly rainbow-connected with $k$ colors.

Finally, suppose $(S, P)$ is $k$-subset strongly rainbow-connected under some edge-coloring $\chi : E \to \{c_1, \ldots, c_k\}$. We will describe an edge-coloring $\chi'$ given to $G'$ by extending $\chi$. We retain the original coloring on the edges of $S$, that is, $\chi'(e) = \chi(e)$, for every $e \in E$. The rest of the edges are colored as follows:

- $\chi'(e) = c_1$, for all $e \in E_1$,
- $\chi'(e) = c_2$, for all $e \in E_3$, and
- $\chi'(ux_{(u,v)}) = c_1$, $\chi'(vx_{(u,v)}) = c_2$, and $\chi'(ax_{(u,v)}) = c_2$ for all $(u, v) \notin P$.

It is straightforward to verify $G'$ is indeed strongly rainbow-connected under $\chi'$, completing the proof. □

We remark the above problem is also NP-complete for $k = 2$ when restricted to split graphs. Indeed, $k$-RC is NP-complete for $k = 2$ for split graphs, as shown by Chandran, Rajendraprasad, and Tesař [10]. As $\text{rc}(G) = 2$ if and only if $\text{src}(G) = 2$, the claim follows.

In Theorem 4.1, we have a chain of reductions from Chromatic Number to Subset Rainbow $k$-Coloring, which we finally reduce to $k$-SRC. The size of the split graph resulting from the above construction remains quadratic in the size of the input graph of the Chromatic Number instance. Moreover, it is known that the chromatic number of an $n$-vertex graph cannot be approximated within a factor of $n^{1-\varepsilon}$, for any $\varepsilon > 0$, unless $\mathsf{P} = \mathsf{NP}$ [76]. The following corollary is then immediate.

**Corollary 4.2.** *The strong rainbow connection number of an $n$-vertex split graph cannot be approximated within a factor of $n^{1/2-\varepsilon}$, for any $\varepsilon > 0$, unless $\mathsf{P} = \mathsf{NP}$.*

We reuse many of the same ideas to show hardness for the strong vertex variant as well. In particular, a similar approach shows $k$-SRVC is NP-complete for every $k \geq 3$. To show hardness for $k = 2$, we take a more direct approach, and reduce directly from $k$-SRC. It will be useful to state this is a separate lemma.

**Lemma 4.3.** *There exists a polynomial-time algorithm which, given an instance $G = (V, E)$ of $k$-SRC, creates an instance $G' = (V', E')$ of $k$-SRVC such that $G$ is a YES-instance of $k$-SRC if and only if $G'$ is a YES-instance of $k$-SRVC. Moreover, $G'$ has size linear in the size of $G$.*

The above lemma combined with another construction establishes the following.

**Theorem 4.4.** *The problem $k$-SRVC is NP-complete for every $k \geq 2$ when restricted to graphs of diameter 3.*

Again, the following corollary is immediate, as we started from Chromatic Number, and controlled the size of the graph resulting from the construction.

**Corollary 4.5.** *The strong rainbow vertex connection number of an $n$-vertex graph cannot be approximated within a factor of $n^{1/2-\varepsilon}$, for any $\varepsilon > 0$, unless $\mathsf{P} = \mathsf{NP}$.*

In fact, the more direct approach we took in Lemma 4.3 for showing $k$-SRVC is NP-complete for $k = 2$ has useful properties. In particular, the same approach establishes also a reduction from $k$-RC to $k$-RVC, for every $k \geq 2$.

**Corollary 4.6.** *There exists a polynomial-time algorithm which, given an instance $G = (V, E)$ of $k$-RC, creates an instance $G' = (V', E')$ of $k$-RVC such that $G$ is a YES-instance of $k$-RC if and only if $G'$ is a YES-instance of $k$-RVC. Moreover, $G'$ has size linear in the size of $G$.*

Conceptually, this is a considerable simplification over the chain of reductions used by Chen, Li, and Shi [15] and Chen, Li, and Lian [14] to show $k$-RVC is hard for $k \geq 2$. Furthermore, it was shown by Chandran and Rajendraprasad [9] that the rainbow connection number of a graph cannot be approximated within a factor of less than 2 unless $\mathsf{P} = \mathsf{NP}$. Again, an easy observation leads us to the following.

**Table 4.1:** Hardness of approximation results for rainbow coloring assuming $\mathsf{P} \neq \mathsf{NP}$. Results originating from this thesis are marked with ★.

| Graph invariant | Result | Graph class | Reference |
|---|---|---|---|
| $\mathrm{rc}(G)$ | No 2-approximation | Bipartite | [9] |
| $\mathrm{rvc}(G)$ | No 2-approximation | General | [P4] ★ |
| $\mathrm{src}(G)$ | No $n^{1/2-\varepsilon}$ for any $\varepsilon > 0$ | Bipartite | [1] |
| $\mathrm{src}(G)$ | No $n^{1/2-\varepsilon}$ for any $\varepsilon > 0$ | Split | [Corollary 4.2] ★ |
| $\mathrm{srvc}(G)$ | No $n^{1/2-\varepsilon}$ for any $\varepsilon > 0$ | Bounded diameter | [P4] ★ |

**Corollary 4.7.** *The rainbow vertex connection number of a graph cannot be approximated within a factor of less than 2 unless* $\mathsf{P} = \mathsf{NP}$.

We summarize the known hardness of approximation results for all four variants of the rainbow coloring problem in Table 4.1.

A natural question arises from the inapproximability results: how fast of an exact algorithm can we hope for? For concreteness, consider $k$-RC, which is $\mathsf{NP}$-complete for every $k \geq 2$. The naive algorithm generates all $k^m$ edge-colorings, and runs the $2^k n^{O(1)}$-time algorithm of Uchizawa *et al.* [74] for each. This results in a runtime of $k^m 2^k n^{O(1)}$, which already in the simplest variant of just two colors, i.e., $k = 2$, takes $2^{O(n^2)}$ time for dense graphs. The question is, is there a considerably faster algorithm?

Recall that ETH (Conjecture 2.7) states that there is no algorithm for deciding 3-SAT in time $2^{o(n)}$. Let us first state the following result that holds under ETH, and then discuss it.

**Theorem 4.8.** *Assuming ETH, there is no algorithm for solving* $k$-RC *running in time* $2^{o(n^{3/2})}$ *for any* $k \geq 2$.

The above result is obtained through a series of reductions using the same intermediate steps as the work of Chakraborty *et al.* [7]. There, the authors start from 3-SAT, and finally output an instance of $k$-RC with $\Theta(n^4 + m^4)$ vertices and edges, where $n$ and $m$ denote the number of variables and clauses, respectively. Indeed, in a typical $\mathsf{NP}$-hardness reduction, there is a polynomial blowup for the size of the resulting instance. In contrast, in case of Theorem 4.8, we must achieve *compression.* That is, to exclude an algorithm running in time $2^{o(n^{3/2})}$ for $k$-RC, our reduction must output instances with $O(n^{2/3})$ vertices, where $n$ is the number of variables in the original 3-SAT instance. To see this, observe that by composing such a reduction with a hypothetical $2^{o(n^{3/2})}$-time algorithm for $k$-RC, one immediately obtains a $2^{o(n)}$-time algorithm for 3-SAT violating ETH. Moreover, to the best of our knowledge, this is the first natural $\mathsf{NP}$-complete graph problem for which the existence of a $2^{o(n^{1+\varepsilon})}$-time algorithm is excluded under reasonable complexity assumptions, for any $\varepsilon > 0$.

Through Corollary 4.6, we also obtain hardness under ETH for the vertex variant of the problem.

**Corollary 4.9.** *Assuming ETH, there is no algorithm for solving* $k$-RVC *running in time* $2^{o(n^{3/2})}$ *for any* $k \geq 2$.

**Table 4.2:** A summary of known complexity results for rainbow coloring. Results originating from this thesis are marked with ★.

| Graph class | $k$-RC | $k$-SRC | $k$-RVC | $k$-SRVC |
|---|---|---|---|---|
| All | $k \geq 2$:NPC | $k \geq 2$:NPC | $k \geq 2$:NPC | $k \geq 2$:NPC ★ |
| AT-free | ? | ? | P ★ | ? |
| Bipartite | $k \geq 3$:NPC | $k \geq 3$:NPC | ? | ? |
| Block | ? | P ★ | P ★ | P ★ |
| Chordal | $k \geq 2$:NPC | $k \geq 2$:NPC | ? | ? |
| Complete | P | P | P | P |
| Complete bipartite | P | P | P | P |
| Interval | ? | ? | P ★ | ? |
| Planar | P ★ | P ★ | P ★ | P ★ |
| Split | $k = 2, 3$:NPC | $k \geq 2$:NPC ★ | P ★ | ? |
| Threshold | P | ? | P ★ | ? |
| Tree | P | P | P | P |

Again, recall that $\text{rc}(G) = 2$ if and only if $\text{src}(G) = 2$. Thus, the result of Theorem 4.8 holds for $k$-SRC for $k = 2$. By Lemma 4.3, the same is true for $k$-SRVC for $k = 2$. It seems reasonable to expect that for both strong variants of the problem, the result holds for every $k \geq 2$. Indeed, we conjecture the following.

**Conjecture 4.10.** *Assuming ETH, there is no algorithm for solving either* $k$-SRC *or* $k$-SRVC *running in time* $2^{o(n^{3/2})}$ *for any* $k \geq 2$.

We summarize the known complexity results for all four variants for well-known graph classes in Table 4.2.

## 4.2 Graphs with bounded structural parameters

In this section, we consider rainbow coloring graphs with bounded structural parameters. In particular, we will focus on graphs of small treewidth, and also on graphs of small vertex cover.

Informally, treewidth is a measure of how close a graph is to being a tree. For instance, forests are precisely the graphs of treewidth 1, and outerplanar graphs have treewidth at most 2. Treewidth is not the only possible way of measuring tree-likeness. However, due to certain algorithmic properties the definition enjoys, it has became increasingly popular in recent years. For more on treewidth and its algorithmic aspects, we refer the interested reader to [22].

When working with treewidth, a useful classification tool arises from *monadic second order logic*, or $\text{MSO}_2$. Let us first define the language of $\text{MSO}_2$, and then explain why it is useful in this context. We have an infinite supply of individual variables, denoted by lowercase letters $x$, $y$, and $z$, and an infinite supply of set variables, denoted by uppercase latters $X$, $Y$, and $Z$. A *formula* of $\text{MSO}_2$ is constructed from atomic formulas $I(x, y)$, $x \in X$, and $x = y$ using the connectives $\neg$ (negation), $\wedge$ (conjunction), and existential quantification $\exists x$ over individual variables, as well as existential quantification $\exists X$ over set variables. Individual variables range over vertices and edges, and set variables range either over sets of vertices or over sets of edges. The atomic formula $I(x, y)$ expresses that a vertex $x$ is incident to an edge $y$, $x = y$ expresses equality, and $x \in X$ expresses membership. The semantics of $\mathrm{MSO}_2$ logic is defined in the standard way (see e.g., [22]).

A restricted logic of $\mathrm{MSO}_2$, known as $\mathrm{MSO}_1$, is defined similarly with the following distinctions. Individual variables range only over vertices, and set variables only range over sets of vertices. The atomic formula $I(x, y)$ is replaced by $E(x, y)$, expressing that a vertex $x$ is adjacent to a vertex $y$. In $\mathrm{MSO}_1$, we are only able to refer to the vertices of a graph, roughly speaking.

We define *free and bound variables* of a formula in the usual way. A *sentence* is a formula without free variables. It is well-known that $\mathrm{MSO}_2$ formulas can be efficiently checked when the graph has bounded treewidth. Similarly, $\mathrm{MSO}_1$ formulas can be checked efficiently when the graph has bounded *cliquewidth.* The following theorems capture these facts.

**Theorem 4.11** ([18])**.** *Let $\phi$ be a fixed* $\mathrm{MSO}_2$ *sentence and $p \in \mathbb{N}$ be a constant. Given an $n$-vertex graph $G$ of treewidth at most $p$, it is possible to decide whether $G \models \phi$ in time $O(n)$.*

**Theorem 4.12** ([19, 36])**.** *Let $\phi$ be a fixed* $\mathrm{MSO}_1$ *sentence and $p \in \mathbb{N}$ be a constant. Given an $n$-vertex graph $G$ of clique-width at most $p$, it is possible to decide whether $G \models \phi$ in time $O(n^3)$.*

It should be noted that the algorithms arising from the theorems are not practical. In other words, the constants hidden by the asymptotic notation are huge. However, the theorems provide us with a useful classification tool. That is, if a problem can be expressed in $\mathrm{MSO}_2$, it is fixed-parameter tractable parameterized by treewidth. Put differently, it is solvable in polynomial time when the input graph has bounded treewidth.

By giving a suitable $\mathrm{MSO}_2$ formula for each of the four rainbow coloring problems, we arrive at the following.

**Theorem 4.13.** *Let $p \in \mathbb{N}$ be fixed. Then, the problems $k$-*RC*, $k$-*SRC*, $k$-*RVC*, and $k$-*SRVC *can be solved in time $O(n)$ on $n$-vertex graphs of treewidth at most $p$. Furthermore, $k$-*RVC *and $k$-*SRVC *can be solved in time $O(n^3)$ on $n$-vertex graphs of cliquewidth at most $p$.*

The above theorem also has consequences for rainbow coloring planar graphs. In particular, it was shown by Eppstein [30] that a planar graph of diameter $d$ has treewidth $O(d)$. In all of the four problems $k$-RC, $k$-SRC, $k$-RVC, and $k$-SRVC, it can be assumed the input graph has diameter at most $k$. If this was not the case, we would be dealing with a NO-instance as $\mathrm{diam}(G) \leq \mathrm{rc}(G) \leq \mathrm{src}(G)$ and $\mathrm{diam}(G) - 1 \leq \mathrm{rvc}(G) \leq \mathrm{srvc}(G)$ holds for every connected graph $G$. The following is then immediate.

**Theorem 4.14.** *The problems $k$-*RC*, $k$-*SRC*, $k$-*RVC*, and $k$-*SRVC *can be solved in time $O(n)$ on $n$-vertex planar graphs.*

Theorem 4.13 shows that for the combined parameter $k + p$, where $p$ denotes treewidth, all four problems are FPT. But what happens when the parameter is $k$ alone? In this case, we are unable to settle the complexity of the problem. In fact, we conjecture the following.

**Conjecture 4.15.** *The problems* RC*,* SRC*,* RVC*, and* SRVC *are* W[1]*-hard parameterized by treewidth.*

When a problem turns out to be hard for treewidth, it is natural to consider a parameter stronger than treewidth. One such popular parameter is the vertex cover number. It seems fair to say that most problems hard for treewidth become tractable for vertex cover number (for a study of treewidth versus vertex cover number, see [31]).

For the following result, we use a so-called *win-win approach*, often used to obtain positive parameterized results. The general idea is the following: we (efficiently) check to see if some structural condition holds.[1] If it does, we can immediately answer YES/NO, depending on the problem. If the particular condition *does not* hold, there is a reason for it. In particular, this reason manifests itself as a combinatorial obstacle. If we can show this obstacle can be exploited algorithmically, we can usually obtain the desired parameterized algorithm.

To illustrate the win-win approach, let $G$ be a graph and let $p$ be its vertex cover number, i.e., the size of a smallest vertex cover of $G$. It can be shown $G$ can be rainbow vertex-connected in $2p$ colors, proving that $\mathrm{rvc}(G) \leq 2p$. Consider then the problem RVC parameterized by $p$. If $k$ —the queried upper bound on the number of colors —is greater than $2p$, we can immediately output YES. Otherwise, we use Theorem 4.13 combined with the fact that the vertex cover number is an upper bound on treewidth to compute a solution in $O(n)$ time. A similar approach establishes the following result for three of the four coloring problems.

**Theorem 4.16.** *Let $p \in \mathbb{N}$ be fixed. Then the problems* RC*,* RVC*, and* SRVC *can be solved in time $O(n)$ on $n$-vertex graphs of vertex cover number at most $p$.*

## 4.3 Variants of rainbow coloring through parameterization

In this section, we consider the parameterized dual problem of rainbow coloring, i.e., the problem of saving $k$ colors from the trivial upper bound. Moreover, it seems natural to consider a problem variant where we relax the condition that every vertex pair needs to be rainbow-connected. For instance, perhaps we wish to rainbow-connect only a certain subset of pairs, or to maximize the number of pairs connected.

Formally, we define the parameterized dual problem of $k$-RC, called $k$-SavingRC, as follows. Given a connected graph $G$, decide whether $\mathrm{rc}(G) \leq m - k$. In other words, the goal is to decide if $G$ can be rainbow-connected with $k$ colors less than the trivial upper bound. The input of $k$-SavingRVC is equivalent, but now we ask whether $\mathrm{rvc}(G) \leq n - k$. The strong variants are defined analogously. The same problem has been investigated in the context of Chromatic Number by Chor, Fellows, and Juedes [17]. Here, the authors showed the problem is FPT parameterized by the number of colors to be saved. In fact, it has been observed that typically, the parametric dual of a problem has complimentary parameterized complexity (see e.g., [50] for more discussion).

We confirm such complimentary nature for rainbow coloring. Using a win-win approach, we show both saving problems are FPT parameterized by $k$, i.e., the number of colors saved.

[1]Often, we might be interested in verifying if treewidth is bounded, but the condition can be something else as well.

**Theorem 4.17.** *For each $k \in \mathbb{N}$, the problems $k$-*SavingRC *and $k$-*SavingRVC *can be solved in time $O(n)$.*

Our win-win approach depends on a non-trivial $\text{MSO}_2$ formulation. It is not clear whether a similar approach works for the strong variants of the saving problems.

Let us then consider the following relaxation of rainbow coloring. In the Maximum Subset Rainbow $k$-Coloring problem we are given a graph $G$, a set of anti-edges $S$, and an integer $q$. The goal is to decide whether there is a coloring of $E(G)$ with $q$ colors such that at least $q$ pairs in $S$ are rainbow-connected. The following observation gives us an approximation algorithm which will be a useful subroutine for other algorithms to follow. We say a vertex pair $(u, v) \in S$ is *feasible* if the distance between $u$ and $v$ is at most $k$.

**Proposition 4.18.** *If all the pairs in $S$ are feasible, then for every $k \geq 2$* Maximum Subset Rainbow $k$-Coloring *admits a deterministic polynomial-time algorithm which finds a coloring that satisfies at least $\frac{k!}{k^k}|S|$ pairs from $S$. In particular, this is a $k!/k^k$-approximation algorithm.*

For a proof of the above statement, we start with a random $k$-edge-coloring. That is, we assign one of the $k$ colors available to an edge with equal and independent probability. As each pair $(u, v) \in S$ is feasible, the pair is rainbow-connected with probability at least $k!/k^k$. By linearity of expectation, the expected number of rainbow-connected anti-edges from $S$ is at least $\frac{k!}{k^k}|S|$. Using the standard method of conditional expectation (see e.g., [75]), the algorithm can be derandomized to run in polynomial time. Clearly, the same approach works for the vertex variant of the problem.

A closely related variant known as Maximum Rainbow $k$-Coloring was introduced by Ananth *et al.* [1]. In this problem, we are given a graph $G = (V, E)$, an integer $q$, and asked whether there is a coloring of $E$ that rainbow-connects at least $q$ non-adjacent pairs of vertices from $V$. Observe that any coloring of $E$ satisfies at least $m$ pairs of vertices. Thus, we parameterize the problem by the number of anti-edges $q$. It was shown by Ananth *et al.* [1] that for $k = 2$, Maximum Rainbow $k$-Coloring is FPT parameterized by $q$. Using Proposition 4.18 together with other techniques, we extend their result to hold for every $k \geq 2$.

**Theorem 4.19.** Maximum Rainbow $k$-Coloring *parameterized by the number of anti-edges $q$ is FPT for every $k \geq 2$.*

In addition, the problem admits a *kernel* of linear size, i.e., there is a polynomial-time algorithm that returns an equivalent instance with $O(q)$ vertices (see [22] for more).

# 5

# Bounds on the rainbow connection numbers

For this chapter, we leave the realm of fixed-parameter tractability. Instead, we take a more graph-theoretic view of rainbow coloring. Indeed, a deeper understanding of the combinatorial nature of the problem is often crucial for the development of faster algorithms, and additional hardness results as well.

## 5.1 Upper bounds via colorings and domination

Because rainbow coloring is hard in general, it is natural to consider tractable special cases, approximation, and FPT algorithms for the problem. Likewise, it is interesting to ask whether any of the four rainbow connection numbers can be upper bounded in terms of other parameters.

For an illuminating example, consider RAINBOW $k$-COLORING which is NP-complete already for $k = 2$. Thus, it would be interesting to upper bound the rainbow connection number of a graph of diameter two in terms of some other well-known graph invariant. Quite similarly to the chromatic number of a graph, we define the *chromatic index* of a graph $G$, denoted by $\chi'(G)$, to be the minimum number of colors needed to color the edges of $G$ in such a way that two adjacent edges receive distinct colors. The following is an easy observation.

**Proposition 5.1.** *Let $G$ be a graph such that* $\mathrm{diam}(G) = 2$. *Then,* $\mathrm{rc}(G) \leq \chi'(G)$.

*Proof.* Give $G$ a proper edge-coloring $c$ using $\chi'(G)$ colors. We claim that any two vertices $u$ and $v$ are rainbow-connected. It suffices to consider $u$ and $v$ such that $d(u, v) = 2$. Any $u$-$v$ path of length two passes through a vertex $w$, using the edges $uw$ and $wv$. Because $c$ is a proper edge-coloring, every edge incident to $w$ has received a distinct color. In particular, this implies $c(uw) \neq c(wv)$, so the claim follows. □

It is well-known that $\chi'(G) \leq \Delta(G) + 1$. Thus, every graph of diameter two can be (strongly) rainbow-connected using $\Delta(G) + 1$ colors. Moreover, one can generalize the idea above for any fixed diameter $d$. For $d = 3$, the upper bound would be in terms of the *strong chromatic index* (see e.g., [63]), and so on. A similar idea works for the vertex variants of the problem as well. However, an upper bound $\mathrm{rc}(G) \leq \chi'(G)$ is not a strong one. In fact, it is known that when $G$ is bridgeless and has diameter two, then $\mathrm{rc}(G) \leq 5$, and this is tight [56]. However, this should make the idea clear: if we can bound a graph invariant in terms of another one, maybe we can obtain approximation algorithms, find tractable special cases, or find improved bounds.

It seems fair to say the edge variants of rainbow coloring have received most of the attention in the literature. To balance the situation from the viewpoint of restricted graph classes, let us make the following observation regarding the rainbow vertex connection number. A graph $G$ is a *diametral path graph* if it contains a shortest path of length $\mathrm{diam}(G)$ that is also a dominating set. For instance, every AT-free graph, unit interval graph, and threshold graph is a diametral path graph. It has turned out that domination is a useful concept when deriving upper bounds on $\mathrm{rc}(G)$ (see e.g., [6, 53, 8]). In the following, we observe this is also the case when dealing with the vertex variants.

**Proposition 5.2.** *Let $G$ be a diametral path graph. Then,* $\mathrm{rvc}(G) = \mathrm{diam}(G) - 1$.

*Proof.* For any connected graph $G$, we have that $\mathrm{diam}(G) - 1 \leq \mathrm{rvc}(G)$. To prove our claim, it suffices to show that $\mathrm{rvc}(G) \leq \mathrm{diam}(G) - 1$. Let $P$ be a dominating diametral path of $G$. Color the internal vertices of $P$ in distinct colors. Without introducing additional colors, color the remaining vertices in $V(G) \setminus V(P)$ arbitrarily. Each vertex has been colored, and we have used precisely $\mathrm{diam}(G) - 1$ colors. To see that two vertices $u$ and $v$ are rainbow vertex-connected, observe that there is a $u$-$v$ path whose internal vertices are contained completely in $V(P)$. Thus, the claim follows. □

Consequently, we have that $k$-RVC is in P for diametral path graphs for every fixed $k$. We believe the same can be shown to be true for $k$-SRVC.

It also seems fair to claim that the rainbow connection number is much more heavily studied than the strong rainbow connection number. In fact, it has been suggested the investigation of the strong rainbow connection number is harder. In [58], the authors write: "The investigation of strong rainbow connection number is much harder than that of rainbow connection number." The reason given is that the strong rainbow connection number is not a monotone graph property. Here, a property is monotone if it does not increase when adding edges. Thus, despite the interest, there have been less results concerning the strong rainbow connection number. In the following section, we give the first polynomial-time algorithm for optimally strongly rainbow-connecting any non-trivial subclass of graphs.

## 5.2 Rainbow coloring block graphs

In this section, we consider rainbow and strong rainbow coloring subclasses of chordal graphs, in particular block graphs. Recall a graph is a block graph if each of its maximal biconnected component is a clique. In fact, chordal graphs have received significant attention in the context of rainbow coloring. But why would one find rainbow coloring chordal graphs particularly interesting? The following gives a possible reason.

A family $\mathcal{C}$ of cliques of a graph $G$ is a *clique covering* if every edge of $G$ is in at least one member of $\mathcal{C}$. The cardinality of a minimum clique covering is called the *clique covering*

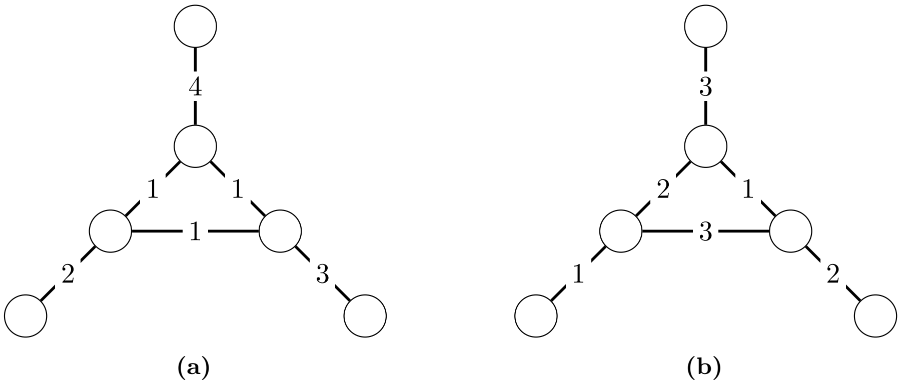


**Figure 5.1:** A graph rainbow-connected in **(a)** four colors, and **(b)** optimally in three colors.

*number* of $G$, denoted by $\mathrm{cc}(G)$. If $G$ has no edges, an empty set is a clique covering of $G$, and thus $\mathrm{cc}(G) = 0$. Clearly, we have that $\mathrm{cc}(G) \leq m$. In terms of $n$, a tight upper bound of $\mathrm{cc}(G) \leq n^2/4$ follows from Mantel's theorem. For more results on the clique covering numbers, we refer the reader to [69, 66]. By observing no shortest path between two vertices $u$ and $v$ uses two edges from a maximal clique, the following is easy to show.

**Proposition 5.3.** *Let $G$ be a connected graph. Then,* $\mathrm{src}(G) \leq \mathrm{cc}(G)$.

In particular for a chordal graph $G$, the above suggests a straightforward linear-time algorithm for rainbow coloring $G$. That is, compute the maximal cliques[1] of $G$, and introduce one color per clique. Finally, color the edges of a clique with the color associated with it. An edge belonging to multiple cliques can be given a color associated with any of the cliques containing it. It is easy to see such a coloring is optimal for infinitely many chordal graphs. When does such a coloring cease to be optimal? A small example is obtained by taking a triangle, and by adding a pendant vertex to each of its vertices (see Figure 5.1). For such a graph $G$, it holds that $\mathrm{rc}(G) = 3$, but $G$ has four maximal cliques. In fact, $G$ has a unique rainbow 3-coloring, up to the permutation of colors. Moreover, $G$ is a block graph. This motivates the investigation of rainbow coloring block graphs in particular.

In what is to follow, our goal is to show block graphs can be strongly rainbow-connected with the optimal number of colors in linear time. Furthermore, we will provide a polynomial-time characterization of bridgeless block graphs with small rainbow connection numbers.

**Clique trees of chordal graphs**. To achieve our goals, let us first apply tools from the theory of chordal graphs to block graphs. In particular, we derive some structural properties that hold for a *clique tree* of a block graph. Here, a clique tree of a (connected) chordal graph $G$ is any tree $T$ whose vertices are the maximal cliques of $G$ such that for every two maximal cliques $C, C'$, each clique on the path from $C$ to $C'$ in $T$ contains $C \cap C'$. Chordal graphs are precisely the class of graphs that admit a clique tree representation [37].

The following concept is central to all chordal graphs, and will provide useful for our algorithms to come. A set $S \subseteq V(G)$ disconnects a vertex $a$ from vertex $b$ in a graph $G$ if

[1]For a chordal graph, this can be done in $O(n + m)$ time (see e.g., [35]).

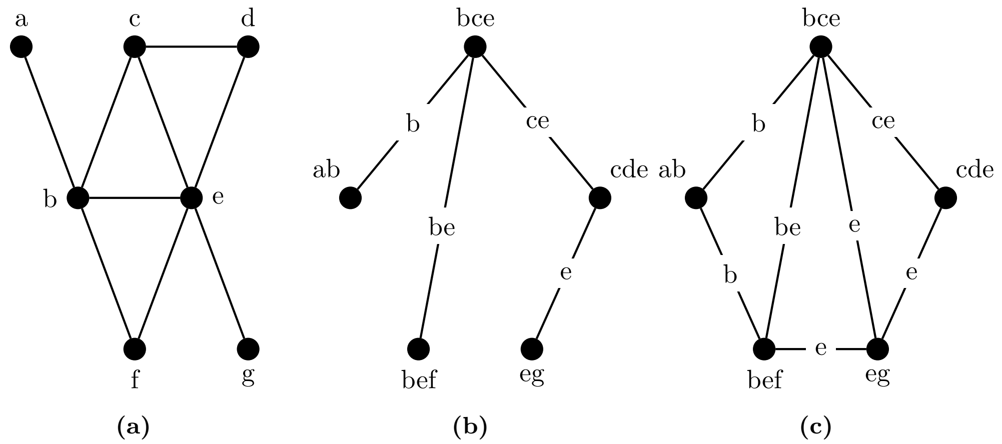


**Figure 5.2:** **(a)** A chordal graph $G$, **(b)** a clique tree of $G$, and **(c)** the reduced clique graph $\mathcal{C}_r(G)$.

every path of $G$ between $a$ and $b$ contains a vertex from $S$. A non-empty set $S \subseteq V(G)$ is a *minimal separator* of $G$ if there exists $a$ and $b$ such that $S$ disconnects $a$ from $b$ in $G$, and no proper subset of $S$ disconnects $a$ from $b$ in $G$. If we want to identify the vertices that $S$ disconnects, we may also refer to $S$ as a *minimal a-b separator*. Two maximal cliques $C_i$ and $C_j$ of $G$ form a *separating pair* if $C_i \cap C_j$ is non-empty and every path in $G$ from a vertex of $C_i \setminus C_j$ to a vertex of $C_j \setminus C_i$ contains a vertex of $C_i \cap C_j$. We denote this separating pair by $S_{i,j}$.

The *reduced clique graph* [39] of a chordal graph $G$ captures all possible clique tree representations of $G$. It is obtained by taking the maximal cliques of $G$ as vertices, and by putting edges between those vertices for which the corresponding cliques intersect in a minimal separator that separates them. The reduced clique graph of $G$ is denoted by $\mathcal{C}_r(G)$. In other words, the reduced clique graph $\mathcal{C}_r(G)$ is the union of all clique trees of $G$ [39]. An example of a chordal graph, a corresponding clique tree, and the corresponding reduced clique graph are given in Figure 5.2.

We may label each edge in $\mathcal{C}_r(G)$ by the minimal separator that separates its endpoints. Let $C$ be a vertex in $\mathcal{C}_r(G)$. For each edge in $\mathcal{C}_r(G)$ incident to $C$, consider its label. The *labeled degree* of the vertex $C$, denoted by $\lambda_{\text{deg}}(C)$, is the number of edges incident to $C$ with distinct labels. Notice that the labeled degree of a vertex is different from the degree of a vertex. Consider the following example illustrated in Figure 5.2. Let $C_1 = \{b, c, e\}$, $C_2 = \{e, g\}$ and $C_3 = \{c, d, e\}$ be vertices in $\mathcal{C}_r(G)$. Notice that $S_{1,2} = \{e\}$, and so the label on the edge $C_1C_2$ is $\{e\}$. Also, $S_{1,3} = \{c, e\}$, thus the label on the edge $C_1C_3$ is $\{c, e\}$. We have that $\deg(C_1) = 4$, and $\lambda_{\text{deg}}(C_1) = 4$. However, $\deg(C_2) = 3$, but $\lambda_{\text{deg}}(C_2) = 1$.

If a graph $G$ has exactly one shortest path between any pair of vertices, $G$ is said to be *geodetic*. It was shown by Stemple and Watkins [72] that a connected graph $G$ is geodetic if and only if every block of $G$ is geodetic. The following observation is later exploited by our algorithms.

**Theorem 5.4.** *Every block graph is geodetic.*

The reduced clique graph $\mathcal{C}_r(G)$ is a useful tool in reasoning about a chordal graph $G$.

However, it is not a linear representation of $G$. For instance, the reduced clique graph of an $n$-vertex star graph uses $\Theta(n^2)$ space. Since a chordal graph on $n$ vertices admits at most $n$ maximal cliques, the size of a clique tree is always bounded from above by $n$. To save space and simplify our algorithms, we will show that we do not need to explicitly compute $\mathcal{C}_r(G)$, but instead that any clique tree of $G$ will do. More specifically, we will show that no matter what clique tree $T$ of $\mathcal{C}_r(G)$ we use, the labeled degree of a vertex in $\mathcal{C}_r(G)$ is preserved in $T$. We first present two results due to Galinier *et al.* [35].

**Lemma 5.5** (Triangle Lemma, [35])**.** *Let $\{C_1, C_2, C_3\}$ be a triangle in $\mathcal{C}_r(G)$ and let $S_{1,2}, S_{1,3}, S_{2,3}$ be the associated minimal separators of $G$. Then 2 of these 3 minimal separators are equal and included in the third.*

**Lemma 5.6** (Weak Triangulation Lemma, [35])**.** *Let $\{C_1, \ldots, C_k\}$, $k \geq 4$, be a path in a clique tree $T$ of a chordal graph $G$. If $C_1C_k$ is an edge of $\mathcal{C}_r(G)$, then either $C_2C_k$ or $C_1C_{k-1}$ is an edge of $\mathcal{C}_r(G)$.*

Recall that in a block graph $G$, the blocks intersect in at most one vertex, which is a cut vertex of $G$. This cut vertex is a minimal separator, so in a block graph, the size of every minimal separator is 1. We claim that for block graphs, the triangle lemma implies that $S_{1,2} = S_{1,3} = S_{2,3}$. If this was not the case, for example, if $S_{1,2} = s$, $S_{1,3} = t$ and $s \neq t$, then $S_{2,3}$ would have to be $s$ or $t$. Without loss we may assume $S_{2,3} = s$. Then the separator $s$ would have to be included in $S_{1,3}$. In other words, $S_{1,3} = \{s, t\}$, and now $S_{1,3}$ has size 2. Thus we have the following lemma.

**Lemma 5.7.** *If $\{C_1, C_2, C_3\}$ is a triangle in the reduced clique graph of a block graph, then $S_{1,2} = S_{1,3} = S_{2,3}$.*

**Theorem 5.8.** *Let $T$ be a clique tree of a block graph $G$, let $\mathcal{C}_r(G)$ be the corresponding reduced clique graph, and let $C_1$ be the same vertex in each. If $e_1, e_2, \ldots, e_l$ are all labeled edges in $\mathcal{C}_r(G)$ incident to $C_1$ with the label $s$, then at least one of these edges must be in $T$.*

*Proof.* Suppose not. Let $C_1$ and $C_k$ be adjacent vertices in $\mathcal{C}_r(G)$ with minimal separator $s$, so that $C_1C_k$ is not an edge in $T$. Let $C_2$ be another vertex in $\mathcal{C}_r(G)$ adjacent to $C_1$, and suppose $C_1C_2$ is an edge of $T$. Then this edge is not labeled $s$, so let $t$ denote the label of this edge. However, because $T$ is a spanning tree, $C_2$ and $C_k$ are connected. Let $\{C_2, C_3, \ldots, C_k\}$ be the path in $T$ from $C_2$ to $C_k$. If this path is simply an edge, then we have a triangle $\{C_1, C_2, C_k\}$ of $\mathcal{C}_r(G)$. Thus, $S_{1,2} = S_{2,k} = S_{1,k}$. However, $S_{1,2} = t$ and $S_{1,k} = s$, so this is a contradiction to Lemma 5.7. Thus there are at least 3 vertices on this path. Then the path $\{C_1, C_2, \ldots, C_k\}$ has at least 4 vertices. Consider this path. Because $C_1C_k$ is an edge of $\mathcal{C}_r(G)$, then by Lemma 5.6, either $C_2C_k$ or $C_1C_{k-1}$ is an edge of $\mathcal{C}_r(G)$. We have already shown $C_2C_k$ is not an edge, so $C_1C_{k-1}$ is an edge of $\mathcal{C}_r(G)$. Because $\{C_1, C_k, C_{k-1}\}$ is a triangle and $S_{1,k} = s$, we have $S_{1,k-1} = s$.

Now consider the path $\{C_1, C_2, \ldots, C_{k-1}\}$. We assume this path has at least 4 vertices, for if it was a triangle, we would have a contradiction to $S_{1,k-1} = s$ and $S_{1,2} = t$. Because $C_1C_{k-1}$ is an edge of $\mathcal{C}_r(G)$, then by Lemma 5.6, either $C_2C_{k-1}$ or $C_1C_{k-2}$ is an edge of $\mathcal{C}_r(G)$. If $C_2C_{k-1}$ was an edge, then $\{C_1, C_2, C_{k-1}\}$ would be a triangle with $S_{1,2} = t$ and $S_{1,k-1} = s$. This contradicts Lemma 5.7, so $C_2C_{k-1}$ is not an edge. Thus $C_1C_{k-2}$ is an edge of $\mathcal{C}_r(G)$, and we have the triangle $\{C_1, C_{k-1}, C_{k-2}\}$. Because $S_{1,k-1} = s$, it follows that $S_{1,k-2} = s$.

We can continue this process, showing that all of the edges

$$C_1C_{k-1}, C_1C_{k-2}, C_1C_{k-3}, \ldots, C_1C_3$$

are in $\mathcal{C}_r(G)$ and have label $s$. But now we have the triangle $\{C_1, C_2, C_3\}$, and since $S_{1,2} = t$ and $S_{1,3} = s$, we have a contradiction to Lemma 5.7. Thus, $T$ has at least one edge incident to $C_1$ with the label $s$. $\square$

**Corollary 5.9.** *Let $G$ be a block graph. Then any pair of clique trees $T_1$ and $T_2$ of $G$ has the property that every vertex in $\mathcal{C}_r(G)$ has the same labeled degree in $T_1$ as it does in $T_2$.*

*Proof.* Let $C$ be a vertex in $\mathcal{C}_r(G)$. Denote the set of edges incident to $C$ with the label $x$ as $I_x$. Then if $C$ has $l$ distinct minimal separators, $C$ has the following incident edges: $I_1, I_2, \ldots, I_l$. By Theorem 5.8, any clique tree of $G$ contains at least one edge from each of $I_1, I_2, \ldots, I_l$. Thus $C$ has labeled degree $l$ in any clique tree of $G$. $\square$

**Strongly rainbow-connecting block graphs in linear time**. We are then in a position to determine exactly the strong rainbow connection number of block graphs. In particular, we present an exact linear-time algorithm for constructing a strong rainbow coloring using $\mathrm{src}(G)$ colors for a given block graph $G$.

Let $C$ be a block in a block graph $G$ whose edges are colored by using colors from the set $R = \{c_1, \ldots, c_r\}$. Then we say that $C$ is *colored* and $C$ is *associated* with each color $c_1, \ldots, c_r$. Furthermore, any color from $R$ can be used as a representative for the color of $C$. Thus we may say that $C$ has been colored $c_i$ for any $i \in \{1, \ldots, r\}$.

**Lemma 5.10.** *Let $G$ be a block graph, let $T$ be a clique tree of $G$, let $C$ be a vertex of $T$ that is associated with the color $c$, let $uv$ be an edge in $G$ such that $u, v \notin C$, and let $y$ be the minimal $a$-$b$ separator for any $a \in C \setminus \{y\}$ and $b \in \{u, v\}$. If no shortest $y$-$u$ path or shortest $y$-$v$ path contains $uv$, then by coloring $uv$ with the color $c$, any shortest path between $u$ or $v$ and $w \in C$ contains at most one edge of color $c$.*

*Proof.* Any shortest path between $u$ or $v$ and $y$ does not contain the edge $uv$, and does not contain any edges in $C$, so these paths do not have any edges of color $c$. Any shortest path between $y$ and $w$ is just an edge of color $c$. $\square$

The algorithm for strongly rainbow-connecting a block graph is presented in Algorithm 5.1. Given a block graph $G$, the algorithm first computes a clique tree $T$ of $G$. Next, it partitions the vertices of $T$ into two sets $V_{<3}$ and $V_{\geq 3}$ based on their labeled degree. If the labeled degree of a vertex is less than 3, it is added to $V_{<3}$. Otherwise, it is added to $V_{\geq 3}$. Then, for each vertex in $V_{<3}$, a distinct color is used to color the edges of the block the vertex corresponds to in $G$. At the final step the algorithm goes through every vertex $C_j \in V_{\geq 3}$. Let $N_\lambda(C_j)$ denote the set of vertices adjacent to $C_j$ via distinct labels. Fix 3 distinct vertices $C_1$, $C_2$, and $C_3$ in $N_\lambda(C_j)$. Observe that in $T \setminus C_j$, we would have at least 3 connected components, and $C_1$, $C_2$, and $C_3$ would be in different connected components. Suppose $C_j$ was removed, and from each connected component $C_1$, $C_2$, and $C_3$ is in, find a vertex in $V_{<3}$. The picked three vertices are each associated with a distinct color. These colors are used to color the edges of the block $C_j$ corresponds to.

The correctness of Algorithm 5.1 is established by an invariant, which says that we always maintain the property that if the shortest path between two vertices is colored, then it is rainbow. We refer to this property as the *shortest rainbow path* property.

**Algorithm 5.1** Algorithm for strongly rainbow-connecting a block graph

```
Input: A block graph $G$
Output: A strong rainbow coloring of $G$
 1: $T :=$ a clique tree of $G$
 2: $V_{<3} := \{U \mid U \in V(T) \wedge \lambda_{\deg}(U) < 3\}$
 3: $V_{\geq 3} := V(T) \setminus V_{<3}$
 4: for all $U \in V_{<3}$ do
 5:     Color edges in $U$ with a fresh distinct color
 6: end for
 7: for all $C_j \in V_{\geq 3}$ do
 8:     Let $C_1$, $C_2$, $C_3$ be distinct vertices in $N_\lambda(C_j)$
 9:     Let $S_{j,1} = x_1$, $S_{j,2} = x_2$, $S_{j,3} = x_3$ be the corresponding minimal separators
10:     Assume $C_j$ is removed
11:     From each connected component $C_1$, $C_2$, $C_3$ is in, find a vertex in $V_{<3}$
12:     Let $c_1, c_2, c_3$ be the respective colors associated with the found vertices
13:     Color all edges not incident to $x_1$ with color $c_1$
14:     Color all edges incident to $x_1$, except $x_1x_2$, with color $c_2$
15:     Color the edge $x_1x_2$ with color $c_3$
16: end for
```

**Theorem 5.11.** *At every step, Algorithm 5.1 maintains the shortest rainbow path property.*

*Proof.* Before the execution of the first loop, nothing is colored so the claim is trivially true. Furthermore, the first loop obviously maintains the property. To see this, consider any shortest path of length $l \geq 1$ at any step. The path consists of $l$ edges that are in $l$ distinct blocks. Since each colored block has received a distinct color, the shortest path is rainbow. This establishes the base step for the correctness of the second loop.

Assume after iteration $i-1$ of the second loop, if the shortest path between any two vertices is colored, then it is rainbow. We show that this property is maintained after iteration $i$ of the second loop. Consider any edge $uv$ in $C_j$ not incident to $x_1$, and let $y \in C_1$ be the minimal $a$-$b$ separator for any $a \in C_1 \setminus \{y\}$ and $b \in \{u, v\}$. The algorithm states that $uv$ will be colored with color $c_1$. Because $u$ and $v$ are both at a distance 1 from $x_1$, it follows that neither shortest path $y$-$u$ or $y$-$v$ contains $uv$. Thus by Lemma 5.10, if the shortest $w$-$u$ path, for $w \in C_1$ is colored, then it is rainbow-connected. (The same is true for the shortest $w$-$v$ path). Therefore, by coloring $uv$ with color $c_1$, the shortest rainbow path property is maintained.

Consider any edge $uv$ in $C_j$ not incident to $x_2$, and let $y \in C_2$ be the minimal $a$-$b$ separator for any $a \in C_2 \setminus \{y\}$ and $b \in \{u, v\}$. By Lemma 5.10, this edge can be colored with $c_2$ to maintain the shortest rainbow path property. Notice that $u$ and $v$ are both at a distance 1 from $x_2$, so it follows that $x_1$ must be one of these vertices (i.e., either $u = x_1$ or $v = x_1$). So we conclude that every edge incident to $x_1$, except $x_1x_2$, can be colored with $c_2$ to maintain the shortest rainbow path property.

Now the only uncolored edge in $C_j$ is the edge $x_1x_2$. Because $x_1$ and $x_2$ are both at a distance 1 from $x_3$, Lemma 5.10 assures us that by coloring $x_1x_2$ with color $c_3$, the shortest rainbow path property is maintained. □

We will then consider the complexity of Algorithm 5.1. It is an easy observation that lines 1 to 6 take linear time. Observe that on lines 8 to 11, we essentially perform reachability queries of the form *given a vertex $v \in V(T)$, return any vertex of degree less than 3 of $T$ that is reachable from $v$ with a path including a given edge $uv$, and no other edges incident to $v$.* In our context, $v$ is $C_j$, and $u$ is $C_i$, where $C_i \in \{C_1, C_2, C_3\}$. The naive way of answering such queries is to start a depth-first search (DFS) from each $C_i$, and halt when a suitable vertex is found. However, such an implementation uses $\Theta(d)$ time, where $d$ is the diameter of the input graph $G$. Using elementary techniques, we can preprocess the clique tree $T$ after line 1 using linear time to answer such queries in $O(1)$ time. Thus, the total runtime will be linear as the for-loop on line 7 loops $O(n)$ times.

**Theorem 5.12.** *Algorithm 5.1 constructs a strong rainbow coloring in $O(n + m)$ time.*

We will now show that the strong rainbow coloring produced by Algorithm 5.1 is optimal. This is done by first showing that we need at least $k$ colors, where $k$ is the number of vertices with labeled degree less than 3 in any clique tree $T$ of $G$. This is then shown to be sufficient as well by a matching upper bound. Recall from Corollary 5.9 that the labeled degree of a vertex of $\mathcal{C}_r(G)$ is preserved in any clique tree $T$ of $G$.

**Theorem 5.13.** *Let $G$ be a block graph, and let $k$ be the number of vertices with labeled degree less than 3 in any clique tree $T$ of $G$. Then src$(G) \geq k$.*

*Proof.* Let $A$ be a set of $k$ edges in $G$, one from each block with labeled degree less than 3, selected as follows. For each vertex $C \in T$, if $\lambda_{\text{deg}}(C) = 1$, pick an edge incident to the minimal separator. If $\lambda_{\text{deg}}(C) = 2$, pick the edge connecting the 2 minimal separators. We claim that if we are to strongly rainbow-connect $G$, then the edges in $A$ must all receive distinct colors.

Suppose there are 2 edges in $A$ that are of the same color, say $ux \in E(C_i)$ and $vy \in E(C_j)$. Without loss, we may assume that $u$ and $v$ are the minimal separators of $C_i$ and $C_j$, respectively, such that $d(u, v)$ is minimized. Then the shortest $x$-$y$ path is unique by Theorem 5.4, and it contains two edges of the same color. □

**Theorem 5.14.** *Let $G$ be a block graph, and let $k$ be the number of vertices with labeled degree less than 3 in any clique tree $T$ of $G$. Then src$(G) = k$.*

*Proof.* It is shown in the proof of Theorem 5.11 that any vertex $C \in V(T)$ with labeled degree at least 3 can be colored using the colors associated with vertices of labeled degree less than 3. Thus we need at most $k$ colors to color $G$. This establishes a matching upper bound for Theorem 5.13, so it follows that $\text{src}(G) = k$. □

To summarize, Theorems 5.11 and 5.14 show that Algorithm 5.1 is correct, and always finds an optimal solution.

If an explicit coloring is not required, then it is easy to see that there is a linear-time algorithm for computing $\text{src}(G)$, where $G$ is a block graph. This is obtained by computing a clique tree $T$ of $G$, and counting the number of vertices with labeled degree less than 3 in $T$.

**Corollary 5.15.** *There is an algorithm such that given a block graph $G$, it computes* $\text{src}(G)$ *in $O(n + m)$ time.*

**Block graphs with small rainbow connection numbers**. Let us then consider the rainbow connection number of block graphs. Using known results, we begin by deriving a tight linear-time computable upper bound on the rainbow connection number. Finally, we prove a polynomial-time characterization of bridgeless block graphs with rainbow connection number at most 4.

Using a technique of [8], we derive an upper bound on the rainbow connection number of a block graph as follows. Recall the connected domination number of a graph $G$, denoted by $\gamma_c(G)$, is the size of a smallest connected dominating set in $G$. The following is known.

**Theorem 5.16** ([8])**.** *For every connected graph $G$, with $\delta(G) \geq 2$,*

$$\mathrm{rc}(G) \leq \gamma_c(G) + 2.$$

Further, the following has been determined.

**Theorem 5.17** ([16])**.** *Let $G$ be a connected block graph, $S$ the set of minimal separators of $G$, and $l$ the number of blocks in $G$. Then*

$$\gamma_c(G) = \begin{cases} 1 & \text{for } l = 1, \\ |S| & \text{for } l \geq 2. \end{cases}$$

Combining the two previous theorems, we get the following.

**Theorem 5.18.** *Let $G$ be a connected block graph with at least two blocks and $\delta(G) \geq 2$. Then $\mathrm{rc}(G) \leq |S| + 2$, where $S$ is the set of minimal separators of $G$. Moreover, this is tight.*

We remark that the above upper bound can be computed in linear time for a given block graph.

We will then characterize the bridgeless block graphs having the rainbow connection number 1, 2, 3, or 4. Before proceeding, we make the following observation that will be useful later on. Recall the *eccentricity* of a vertex $v$ is the maximum distance between $v$ and any other vertex in a graph. A *peripheral vertex* is a vertex of maximum eccentricity.

**Lemma 5.19.** *Let $G$ be a block graph with at least 3 blocks, and let $x$ and $y$ be two peripheral vertices in distinct blocks. If $G$ has a minimal separator $s$ adjacent to $x$ and $y$, then $\mathrm{rc}(G) > \mathrm{diam}(G)$.*

*Proof.* Suppose not. That is, assume $\mathrm{rc}(G) = \mathrm{diam}(G)$. Let $C_x$ and $C_y$ be the two distinct blocks $x$ and $y$ are in, respectively. Choose a vertex $z \in C_z$ such that $d(x, z) = \mathrm{diam}(G)$, where $C_z$ is a block different from $C_x$ and $C_y$. Rainbow color the shortest $x$-$z$ path. Without loss, suppose the edge $xs$ was colored with color $c_1$. Then consider each uncolored edge incident to $s$ in $C_x$. Notice we must color each such edge with color $c_1$, for otherwise $G$ would not be rainbow-connected. Finally, consider the edges incident to $s$ in $C_y$. Again, each such edge must receive color $c_1$. But now $x$ and $y$ are not connected by a rainbow path. Thus, $\mathrm{rc}(G) > \mathrm{diam}(G)$. □

We are then ready to proceed with the claimed characterization.

**Theorem 5.20.** *Let $G$ be a bridgeless block graph, and let $k$ be a positive integer such that $k \leq 4$. Deciding whether* $\mathrm{rc}(G) = k$ *is in* $\mathsf{P}$.

*Proof.* As $k \leq 4$, it is enough to consider bridgeless block graphs with diameter $d = \mathrm{diam}(G) \leq 4$. In what follows, we show how such graphs are optimally colored.

- Case $d = 1$. Trivial.
- Case $d = 2$. If $G$ has exactly 2 blocks, it is easy to see that $\mathrm{rc}(G) = 2$. Moreover, if the graph has $\mathrm{rc}(G) = 2$, it must have exactly 2 blocks. Suppose this is was not the case, i.e. $G$ has at least 3 blocks and $\mathrm{rc}(G) = 2$. By an argument similar to Lemma 5.19, this leads to a contradiction. Thus, $\mathrm{rc}(G) = 2$ if and only if $G$ has exactly 2 blocks. When $G$ consists of 3 or more blocks, we will show that $\mathrm{rc}(G) = 3$. Let $\mathcal{K}$ be the set of all blocks of $G$, and let $a$ be the unique central vertex of $G$. For each $K \in \mathcal{K}$, color one edge incident to $a$ with the color $c_1$, and every other incident edge with the color $c_2$. Then color every uncolored edge of $G$ with the color $c_3$. To see this is a rainbow coloring of $G$, observe there is a rainbow path from any vertex to the central vertex $a$ avoiding a particular color in $\{c_1, c_2, c_3\}$.
- Case $d = 3$. The graph $G$ consists of a unique central clique, and at least 2 other blocks. If $G$ has altogether 3 blocks, then $\mathrm{rc}(G) = \mathrm{src}(G) = 3$. If $G$ has 4 blocks, there are two cases: either $G$ has a cut vertex adjacent to two peripheral vertices in distinct blocks (then $\mathrm{rc}(G) \geq 4$ by Lemma 5.19) or it does not (then $\mathrm{rc}(G) = \mathrm{src}(G) = 3$). Otherwise, $G$ has at least 5 blocks, and by an argument similar to Lemma 5.19, we have that $\mathrm{rc}(G) \geq 4$. We will then color every block that is not the central clique with 3 colors exactly as in the case $d = 2$, and color every edge of the central clique with a fresh distinct color $c_4$ proving $\mathrm{rc}(G) = 4$.
- Case $d = 4$. Let us call the set of blocks which contain the central vertex $a$ the *core* of the graph $G$. The set of blocks not in the core is the *outer layer*. First, suppose the core contains exactly 2 blocks, and the outer layer at most 4 blocks. Furthermore, suppose the condition of Lemma 5.19 does not hold (otherwise we would have $\mathrm{rc}(G) > 4$ immediately). When the outer layer contains 2 or 3 blocks, we have that $\mathrm{rc}(G) = \mathrm{src}(G) = 4$. Suppose the outer layer contains exactly 4 blocks. First, consider the case where a core block is adjacent to 3 blocks in the outer layer. Because the condition of Lemma 5.19 does not hold, it must be the case that at least one of the core blocks is not a $K_3$. Clearly, every two vertices $x$ and $y$, such that $d(x, y) = \mathrm{diam}(G)$, have to be connected by a rainbow shortest path. By an argument similar to Theorem 5.13, we have that $\mathrm{rc}(G) > 4$. Otherwise, when a core block is not adjacent to 3 blocks in the outer layer, $\mathrm{rc}(G) = \mathrm{src}(G) = 4$. Now suppose the outer layer has at least 5 blocks. As above, by an argument similar to Theorem 5.13, we have that $\mathrm{rc}(G) > 4$. Finally, suppose the core has 3 or more blocks. We argue that in this case, $\mathrm{rc}(G) = 4$ if and only if the outer layer contains exactly 2 blocks. For the sake of contradiction, suppose $\mathrm{rc}(G) = 4$, and that the outer layer has 3 or more blocks. If the condition of Lemma 5.19 holds, we have an immediate contradiction. Otherwise, by an argument similar to Lemma 5.19, we arrive at a contradiction. When the outer layer contains exactly 2 blocks, we will show $\mathrm{rc}(G) = 4$. Let $B_1$ and $B_2$ be the blocks in the outer layer. We color every edge of $B_1$ with color $c_1$, and every edge of $B_2$ with color $c_4$. Then color $b_1 a$ with $c_2$, and $ab_2$ with $c_3$, where $a$ is the central vertex of $G$, and $b_1$ and $b_2$ are the cut

vertices in $B_1$ and $B_2$, respectively. For every block $B_i$ in the core, let $Q_i$ denote the set of edges in $B_i$ incident to $a$. Color the uncolored edges of $Q_i$ with either $c_2$ or $c_3$, such that both colors appear at least once in $Q_i$. Then, color every uncolored edge of the block that contains both $a$ and $b_2$ with color $c_1$. Every other uncolored edge of $G$ receives color $c_4$. We can now verify $G$ is indeed rainbow-connected under the given coloring.

□

Given that the strong rainbow connection number of a block graph $G$ can be efficiently computed, it is interesting to ask when $\mathrm{rc}(G) = \mathrm{src}(G)$, or if the difference between $\mathrm{src}(G)$ and $\mathrm{rc}(G)$ would always be small. It is at least easy to observe that the difference between $\mathrm{src}(G)$ and $\mathrm{rc}(G)$ can be made arbitrarily large: attach $n$ triangles to a $K_n$, one to each vertex of the $K_n$. As $n$ increases, the rainbow connection number remains 4 by Theorem 5.20, while the strong rainbow connection number increases by Theorem 5.14.

## 5.3 Rainbow coloring chordal diametral path graphs

Another subclass of chordal graphs is given by *chordal diametral path graphs*. That is, such graphs are chordal, and have a dominating diametral path. At first, such graphs might feel contrived. However, chordal diametral path graphs generalize well-known subclasses of chordal graphs, including interval graphs and threshold graphs.

It follows from Theorem 5.16 that for any chordal diametral path graph $G$ we have that $\mathrm{rc}(G) \leq \mathrm{diam}(G) + 3$, provided that $\delta(G) \geq 2$. The same holds true for AT-free graphs, which form a subclass of diametral path graphs. For some graph classes, including AT-free graphs, no tight examples achieving $\mathrm{rc}(G) = \gamma_c(G) + 2$ were found in [8]. In fact, it was explicitly mentioned a deeper investigation into the rainbow connection number of these graphs could be interesting.

Using combinatorial methods, we show that a chordal diametral path graph $G$ with $\delta \geq 2$ has $\mathrm{rc}(G) \leq \mathrm{diam}(G) + 1$. Moreover, we will show this bound is tight. The proof of the claim is split into two lemmas.

**Lemma 5.21.** *Let $G$ be a chordal diametral path graph such that $\delta(G) \geq 2$ and $\mathrm{diam}(G) \geq 3$. Then, $\mathrm{rc}(G) \leq \mathrm{diam}(G) + 1$.*

*Proof.* Denote $d = \mathrm{diam}(G)$. In what follows, we describe an algorithm for constructing a rainbow $d+1$-coloring $c : E \to [d+1]$ for $G$.

1. **Diametral path cliques**. Choose an arbitrary dominating diametral path $D = \{v_1, v_2, \ldots, v_{d+1}\}$ from $G$, which exists by definition. Define a set $\mathcal{D}$ of maximal cliques as follows. For $2 \leq i \leq d-1$, choose a maximal clique containing the edge $v_i v_{i+1}$, and add it to $\mathcal{D}$. For each $i$, the uncolored edges of the maximal clique are colored $i$.

2. **Edges incident to inner vertices**. We partition the vertices of $G$ into two sets. Let $V(\mathcal{D})$ denote the set of vertices of elements of $\mathcal{D}$. We say a vertex is an *inner vertex* if it is in $V(\mathcal{D})$; otherwise it is an *outer vertex*. Consider each maximal

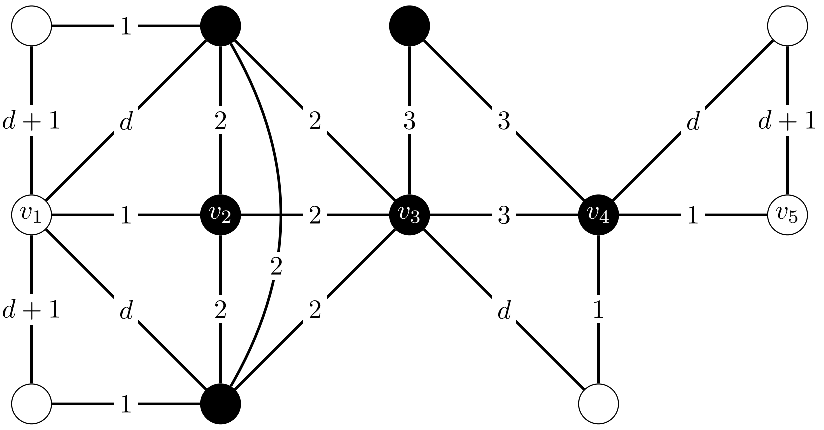


**Figure 5.3:** A chordal diametral path graph $G$ with $\delta = 2$ and $\operatorname{diam}(G) = 4$ colored by the algorithm of Lemma 5.21. The black vertices are inner vertices, while $D = \{v_1, \ldots, v_5\}$.

clique $C \notin \mathcal{D}$. As $\delta \geq 2$, we have $|C| \geq 3$, i.e., there are no pendant vertices. Let $C \cap V(\mathcal{D}) = I_C$. If $I_C = \{x\}$, color edges $xc$, where $c \in C$, such that at least one of the edges is colored 1, and at least one of the edges is colored $d$. Otherwise, $|I_C| \geq 2$. For each $c \in C$, color edges $cx$, where $x \in I_C$, such that at least one of the edges is colored 1, and at least one the edges is colored $d$.

3. **Outer vertices**. Let $u$ and $v$ be outer vertices. Color the edge $uv$ with color $d+1$.

Only remaining uncolored edges (if any) are between two inner vertices $x$ and $x'$. Such edges receive an arbitrary color from $[d+1]$. Every edge is now colored. This finishes the construction. An example is shown in Figure 5.3.

The following key properties are immediate from the construction. (The numbering corresponds to the numbering of the steps of the algorithm).

($P_1$) Two inner vertices $u$ and $v$ are rainbow-connected by a path on which colors 1, $d$, and $d+1$ do not appear.

($P_2$) For an edge $xu$, we have $c(xu) = 1$ or $c(xu) = d$, where $x$ is an inner vertex and $u$ an outer vertex.

($P_3$) An edge between two outer vertices has color $d+1$.

We claim that two vertices $u$ and $v$ are rainbow-connected under the constructed edge-coloring $c$. When both $u$ and $v$ are inner vertices, the claim is true by ($P_1$). When exactly one of $u$ and $v$ is an inner vertex and the other an outer vertex, the claim is true by combining ($P_1$) with ($P_2$). Thus, it remains to be shown that two outer vertices $u$ and $v$ are rainbow-connected. Before proceeding, we introduce three additional claims that will simplify the proof.

**Claim 1.** *Let $u$ and $v$ be two outer vertices. Either $u$ or $v$ is adjacent to an inner vertex.*

*Proof.* Observe that the claim holds for any outer vertex $u$ that is not adjacent to $v_1$ or $v_{d+1}$, as $u$ is dominated by some other (inner) vertex in $V(\mathcal{D})$. This is so because

$V(\mathcal{D})$ is constructed from $D$ by removing only $v_1$ and $v_{d+1}$, and by adding other vertices. Therefore, it suffices to consider the claim when both $u$ and $v$ are adjacent to the same endpoint of $D$, or adjacent to different endpoints of $D$. In other words, it is either the case (without loss) that $uv_1, vv_1 \in E$, or $uv_1, vv_{d+1} \in E$.

First, suppose $uv_1, vv_1 \in E$. By the above, we have for every $y \in D \setminus \{v_1, v_{d+1}\}$ that $uy, vy \notin E$. As $\delta \geq 2$, both $u$ and $v$ have degree at least two. Clearly, we have that $uv \notin E$, for otherwise $d(v, v_{d+1}) = d + 1$ contradicting the fact that $D$ is a diametral path (by symmetry, the same is true for $u$). Because $uy, vy \notin E$ and $\delta \geq 2$, it must be the case that there are vertices $u', v' \in V$ such that $uu', vv' \in E$. So consider $u'$. As $D$ is dominating shortest path, $u'$ is dominated by some $y' \in D$. Observe that there is no edge $u'v_j$ for any $j \geq 5$ as otherwise we get a path shorter than $d$ from $v_1$ to $v_{d+1}$ (by symmetry, the same is true for $v'$). However, as $\delta \geq 2$, it must be the case that one of the edges $u'v_1$, $u'v_2$, $u'v_3$, or $u'v_4$ is present in $E$ (by symmetry, the same is true for $v'$).

First, we claim that $u'v_4 \notin E$. So suppose this is not the case, i.e., that $u'v_4 \in E$. We have a 6-cycle on the vertices $v_1$, $u$, $u'$, $v_4$, $v_3$, and $v_2$. As there is no chord $uy$ where $y \in D$ and no chord $v_1v_j$ for $j \geq 3$, it must be the case that $v_1u'$ is a chord (for otherwise there is a chordless cycle of length at least 4, namely, $u$, $u'$, $v_2$, $v_1$ or $u$, $u'$, $v_3$, $v_2$, $v_1$). But there is no chord $v_1u'$ for otherwise we would have a path from $v_1$ to $v_4$ of length less than three contradicting the fact that $D$ is a shortest path. Thus, $u'v_4 \notin E$, and by symmetry, $v'v_4 \notin E$.

We will then prove that the statement is true in the remaining three cases.

- $u'v_3 \in E$: The only possible chords are $v_1u'$ and $v_2u'$. In fact, both must be present for otherwise $G$ would contain a chordless cycle of length 4 (either $v_1, u, u', v_2$ or $v_1, u', v_3, v_2$). Now, consider $v'$. As $D$ is a dominating set, there is a $y' \in D$ that dominates $v'$. More precisely, we have that $y' \in \{v_1, v_2, v_3\}$ as there is no edge $v'v_j$ for $j \geq 4$ by the previous case. Symmetrically, $v_1v'$ must be an edge, for otherwise we would have a chordless 4-cycle $v, v', v_2, v_1$. Moreover, it must be the case either $v'v_2 \in E$ or $v'v_3 \in E$ (otherwise $d(v', v_{d+1}) = d + 1$ contradicting the fact that $D$ is a diametral path). If $v'v_2 \in E$, then $d(v, v_{d+1}) = d + 1$, again, a contradiction. It follows that both $v'v_3$ and $v'v_2$ must be edges. Now there is a chordless 4-cycle $v', v_3, u', v_1$. Obviously, there is no chord $v_1v_3$, so we must have the chord $u'v'$. In other words, $u'$, $v'$, $v_2$, and $v_3$ form a $K_4$. Thus, by definition, $u'$ and $v'$ are inner vertices, and we have that $uu', vv' \in E$, which is what we wanted to show.

- $u'v_2 \in E$: There is no chord $uv_2$, so we must have the chord $v_1u'$. Furthermore, it must be the case that $u'v_3 \in E$, for otherwise $d(u, v_{d+1}) = d + 1$ leading to a contradiction. But now we reach the same case as above, and by the same reasoning conclude that $uu', vv' \in E$, which is what we wanted to show.

- $u'v_1 \in E$: By the same argument as above, it follows that $u'v_2$ and $u'v_3$ must also be edges. Again, we conclude that $uu', vv' \in E$ where $u'$ and $v'$ are inner vertices, which is what we wanted to show.

Finally, suppose $uv_1, vv_{d+1} \in E$. Furthermore, suppose the statement of the claim does not hold, i.e., that $d(u, x) > 1$ and $d(v, x') > 1$, where $x, x' \in V(\mathcal{D})$ (with possibly $x = x'$). Because $ux, vx' \notin E$, we have that $d(u, v) = \text{diam}(G) + 2$ contradicting $G[D]$ being of length $\text{diam}(G)$. Thus, $ux \in E$ or $vx' \in E$, and the claim follows. □

The two following claims are structural, and essentially follow by chordality of $G$.

**Claim 2.** *Let $u$ be an outer vertex. If $u$ is adjacent to an inner vertex $x$, then $u, x$ forms a triangle with a third vertex $u'$.*

*Proof.* As $\delta \geq 2$, we have that $u$ is adjacent to $u'$. For the sake of contradiction, suppose $u'x \notin E$. Thus, $u'$ is dominated by some other vertex $y \in D$ such that $y \neq x$. As $D$ is a dominating shortest path, there is an $x$-$y$ path $P_{xy}$ of length at least 1 that lies completely in $D$. Thus, the concatenation of $P_{xy}$, $xu$, $uu'$, and $u'y$ is a cycle of length at least 4. Let $y' \in P_{xy} \setminus \{x\}$. Because $D$ is a shortest path, there is no chord $y'u$ or $y'u'$. Thus, $G$ contains a chordless cycle of length at least 4, a contradiction. So $u'x \in E$, and thus $x, u, u'$ is a triangle. ■

**Claim 3.** *Let $u$ be an outer vertex. If $u$ is not adjacent to an inner vertex $x$, then there are two additional vertices $u', u''$ such that $x, u', u, u''$ form a 4-cycle (containing a chord).*

*Proof.* We proceed with an argument similar to that in Claim 2. Because $ux \notin E$, there is some (outer) vertex $u'$ such that $uu', u'x \in E$. Moreover, as $\delta \geq 2$, there is an (outer) vertex $u'' \neq u'$ such that $u''u \in E$. We claim that $u''x \in E$. Suppose this is not the case. Then $u''$ is dominated by some other vertex $y \in D$ such that $y \neq x$. Again, let $P_{xy}$ be the shortest $x$-$y$ path of length at least 1 that lies completely in $D$. The concatenation of $P_{xy}$, $xu'$, $u'u$, $uu''$, and $u''y$ forms a cycle of length at least 5. Let $y' \in P_{xy} \setminus \{x\}$. Because $P_{xy}$ is a shortest path, there is no chord $y'u$ or $y'u'$. Thus, $G$ contains a chordless cycle of length at least 4, a contradiction. So $xu'$ must be an edge. Furthermore, as $x, u', u, u''$ is a 4-cycle, it holds that $u'u''$ is a chord. The claim follows. ■

We are then ready to proceed with the final step of the proof. We will use all of the three previous claims in the proof.

**Claim 4.** *Let $u$ and $v$ be outer vertices. Then, $u$ and $v$ are rainbow-connected.*

*Proof.* Suppose there are two inner vertices $x$ and $x'$ (not necessarily distinct) such that $u$ is adjacent to $x$, and $v$ is adjacent to $x'$. If $c(ux) \neq c(x'v)$, we are done by combining ($P_1$) with ($P_2$). Otherwise, suppose $c(ux) = c(x'v)$. By Claim 2, there is a vertex $u'$ adjacent to both $x$ and $u$. By construction, $c(ux) \neq c(u'x)$. Now, suppose $u'$ is an outer vertex. Then, by combining ($P_3$) with ($P_2$), we have that $c(uu') = d + 1 \neq c(u'x)$. As $c(ux) = c(x'v)$, we have constructed a rainbow $u$-$v$ path of length $d(u, v) + 1$. Finally, suppose $u'$ is an inner vertex. We have $c(ux) = c(vx')$ by assumption, so by combining ($P_1$) with ($P_2$), we have $c(uu') \neq c(ux)$. In particular, it follows $c(uu') \neq c(vx')$, so we have a rainbow $u$-$v$ path of length $d(u, v) + 1$.

Finally, by Claim 1, it suffices to show $u$ and $v$ are rainbow-connected when exactly one of $u$ and $v$ is adjacent to an inner vertex $x$. Without loss, suppose $ux \notin E$ and $vx' \in E$, where $x'$ is an inner vertex (not necessarily distinct from $x$). By Claim 3, there are two outer vertices $u', u''$ such that $u, u', x, u''$ is a 4-cycle (with the chord $u'u''$). By ($P_2$) and ($P_3$), there is a path of length 2 from $u$ to $x$ using colors $\{1, d + 1\}$, and also using colors $\{d, d+1\}$. Moreover, by ($P_2$), we have $c(vx') = 1$ or $c(vx') = d$, where $x'$ is an inner vertex (with possibly $x = x'$). Thus, by ($P_1$) we have that $u$ and $v$ are rainbow-connected. ■

All cases are exhausted, so we conclude the proof. □

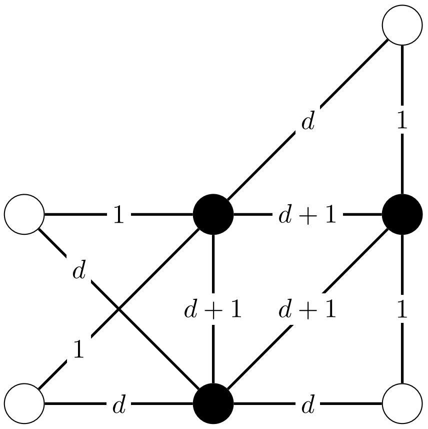


**Figure 5.4:** A chordal diametral path graph $G$ with $\delta = 2$ and $\text{diam}(G) = 2$ colored by the algorithm of Lemma 5.23. The black vertices form a dominating clique.

In the above, if $\text{diam}(G) = 2$, it would be the case that $\mathcal{D}$ is empty. In other words, there would be no inner vertices. For this reason, we handle the case $\text{diam}(G) = 2$ separately using the following result of Kratsch, Damaschke, and Lubiw [52].

**Theorem 5.22** ([52])**.** *A chordal graph $G$ has a dominating clique if and only if* $\text{diam}(G) \leq 3$.

**Lemma 5.23.** *Let $G$ be a chordal diametral path graph such that $\delta(G) \geq 2$ and* $\text{diam}(G) = 2$*. Then,* $\text{rc}(G) \leq \text{diam}(G) + 1$.

*Proof.* By Theorem 5.22, $G$ contains a dominating clique $D$. Color every edge in $G[D]$ with color $d + 1$. Assuming the terminology of Lemma 5.21, a vertex $x \in D$ is an inner vertex, and a vertex $u \notin D$ is an outer vertex. Execute steps 2 and 3 of the algorithm given in Lemma 5.21. An example is shown in Figure 5.4. By an argument similar to Claim 4, we have that $G$ is rainbow-connected. □

By combining Lemma 5.21 and Lemma 5.23, we have proven the following.

**Theorem 5.24.** *Let $G$ be a chordal diametral path graph such that $\delta(G) \geq 2$. Then,* $\text{rc}(G) \leq \text{diam}(G) + 1$.

To see that the above is tight, one can consider a sufficiently large fan, that is, a path with one universal vertex added (see e.g., [42]). One can also consider the graph $G$ obtained as the join of $K_2$ and $n$ isolated vertices (see e.g., [8, Example 3]). When $n = 4$, we have that $\text{rc}(G) = 2$. But when $n \geq 5$, we have $\text{rc}(G) = 3$.

Finally, we remark the algorithm given in Lemma 5.21 can be implemented to run in polynomial time. The runtime is dominated by that of finding a dominating diametral path. By the result of [23], this can be done in $O(n^3 m)$ time. The remaining steps can be performed within the same time bound.

6

# Conclusions

We conclude the thesis by exploring some questions left open by our work. In particular, we will pose both conjectures and open problems. Conjectures are precise statements, whereas open problems are more open-ended research directions we find interesting.

## 6.1 Conjectures and open problems

The presented open questions respect the thematic order of the thesis. That is, we look at rainbow connectivity first. Next, we proceed to rainbow coloring problems, and conclude with questions with a flavor of pure mathematics. Naturally, a strict categorization is challenging. For instance, a polynomial-time algorithm computing the rainbow connection number of a restricted graph class might establish both the tractability of $k$-RC, and give improved bounds for a particular graph class.

**Rainbow connectivity**. Arguably the most interesting problem left open is the complexity of finding a rainbow $s$-$t$ path in a $k$-edge-colored graph. We proved that under the Set Cover Conjecture, there is no $(2-\varepsilon)^k n^{O(1)}$-time algorithm for Edge-Colorful Path for any $\varepsilon > 0$. Furthermore, we showed a rainbow $s$-$t$ path can be found in $2^k n^{O(1)}$-time and polynomial space, but is there a matching lower bound? We conjecture this is true, perhaps similarly under the Set Cover Conjecture.

**Conjecture 6.1.** *For every $\varepsilon > 0$ there is no $(2-\varepsilon)^k n^{O(1)}$-time algorithm for* Rainbow *st*-Connectivity *under reasonable complexity assumptions.*

If the conjecture is true, can it be extended for the strong variants of the problem?

Our results provide a thorough hardness classification of the problems Rainbow Connectivity, Strong Rainbow Connectivity, Rainbow Vertex Connectivity, and Strong Rainbow Vertex Connectivity. However, some gaps remain (see Table 3.1). In particular, we find the investigation of Rainbow Connectivity and Rainbow Vertex Connectivity on split graphs interesting. For Rainbow Vertex Connectivity, it is clear that the problem is easy if the split graph given as input has diameter (at most) two. But what happens on split graphs of diameter three? We believe techniques from our construction for showing NP-completeness of Rainbow Connectivity for block graphs can be useful.

**Conjecture 6.2.** *Both* Rainbow Connectivity *and* Rainbow Vertex Connectivity *are* NP*-complete for split graphs.*

We showed that block graphs constitute a graph class for which Rainbow Connectivity is hard, but Strong Rainbow Connectivity is easy. If the above conjecture is true, it would provide us with an example of such phenomenon for the vertex variants.

**Rainbow coloring**. We have several examples of restricted graph classes for which $k$-RC, $k$-SRC, $k$-RVC, and $k$-SRVC are hard. But what can be said about the problem variants in which the number of colors $k$ is given as part of the input? To the best of our knowledge, no graph class is known for which say $k$-RC is easy for every fixed $k$, but RC is NP-complete. We firmly believe block graphs are a good candidate for such a graph class. Thus, we conjecture the following.

**Conjecture 6.3.** *The problem* RC *is* NP*-complete for block graphs.*

We also believe planar graphs are a good candidate for such a problem. By our earlier observation (Theorem 4.14), all of the four problems are in P for fixed $k$ on planar graphs. This is in contrast to bridgeless block graphs, for which we know this to be the case only for $k \leq 4$.

**Conjecture 6.4.** *The problem* RC *is* NP*-complete for planar graphs.*

Recall that it is rather trivial to rainbow-connect trees and complete graphs optimally. What happens on graphs that are close to being trees, or close to being complete? In this sense, we find treewidth and cliquewidth the most interesting parameters. In particular, we believe that when the parameter is treewidth (or cliquewidth) alone, rainbow coloring is hard.

**Conjecture 6.5.** *The problem* RC *is* W[1]*-hard parameterized by treewidth. Moreover, the same is true when treewidth is replaced with cliquewidth.*

We proved $k$-SRVC is NP-complete for every $k \geq 2$. Moreover, this holds for graphs of bounded diameter. In fact, it seems that there are several examples of graph classes for which the edge variants are hard, but the vertex variants remain easy. This raises a natural question: when does say $k$-SRVC cease to be hard?

**Open problem 6.6.** Investigate the complexity of rainbow vertex coloring. In particular, for what restricted graph classes do $k$-RVC and $k$-SRVC remain NP-complete?

Our results also show the strong rainbow vertex connection number of an $n$-vertex split graph cannot be approximated within a factor of $n^{1/2-\varepsilon}$, for any $\varepsilon > 0$, unless $\mathsf{P} = \mathsf{NP}$. Is there a matching upper bound?

**Open problem 6.7.** Investigate the approximability of rainbow coloring. In particular, are there matching upper bounds for the known lower bounds?

**Graph-theoretic considerations**. From a combinatorial perspective, there are still several questions to be investigated. To a modest extent, we explored the possibility of computer-assisted and automated conjecture-making in the context of rainbow coloring. To this end, we used the system named GraPHedron [65]. On a high-level, GraPHedron keeps a large database of dozens of interesting graph parameters, computed for all non-isomorphic graphs of up to $n$ vertices (typically, $n$ is at most 12). The user chooses two parameters, and the system investigates their relationship. In particular, GraPHedron is able to automatically suggest conjectures in form of inequalities. For instance, the following was suggested by GraPHedron. Moreover, it was verified for all graphs with at most 8 vertices.

**Conjecture 6.8** (GraPHedron)**.** *A connected graph $G$ with $n$ vertices and chromatic number $\chi(G)$ has* $\mathrm{src}(G) \leq n + 1 - \chi(G)$.

A weaker version of the above is the statement that $\mathrm{src}(G) \leq n + 1 - \omega(G)$, and an even weaker statement is that $\mathrm{src}(G) \leq n - 1$. We have made partial progress in settling these two weaker statements in [49], but their status remains open.

Relating to Open Problem 6.7, let us mention that the edge-splitting lemma of Kamčev, Krivelevich, and Sudakov [46] might be a useful starting point. In particular, they proved that if $G$ has two connected spanning subgraphs $G_1 = (V, E_1)$ and $G_2 = (V, E_2)$ such that $|E_1 \cap E_2| \leq c$, then $\mathrm{rc}(G) \leq \mathrm{diam}(G_1) + \mathrm{diam}(G_2) + c$. Obviously, if $c = 0$ and $\mathrm{diam}(T_1) = \mathrm{diam}(T_2) = \mathrm{diam}(G)$, we have obtained a rainbow coloring using at most twice the optimal number of colors. In this sense, it could be interesting to consider spanning subtrees. Specifically, can we characterize the graphs which have two edge-disjoint *diameter-preserving* spanning trees? Here, a spanning tree $T$ is said to be diameter-preserving if $\mathrm{diam}(T) = \mathrm{diam}(G)$. A more general question is thus the following.

**Open problem 6.9.** Characterize the graphs which have $k$ edge-disjoint diameter-preserving spanning trees for $k = 2$.

It appears the problem is open [33], and we think the answer could be of independent interest. In addition, “diameter-preserving” could be replaced with “minimum diameter” as well (for work in this direction, see e.g., [34]). Of course, such a characterization would also be interesting for graphs whose spanning trees share $c = O(1)$ edges. We remark that a characterization for $k = 1$ is provided by Buckley and Lewinter [5]. Moreover, a diameter-preserving spanning tree can be found in polynomial time [40].

# Appendices

A

# A compendium of common problems

For completeness, we provide the reader with a compact, comprehensive list of the computational problems considered in this work. Many of the problems listed are often assumed to be common knowledge in the field. We divide the problems into three categories: rainbow connectivity problems, rainbow coloring problems, and other problems.

To avoid confusion, we stress that in the rainbow connectivity problems (Section A.1), the connected input graph is colored. In contrast, in the rainbow coloring problems (Section A.2), the connected input graph is uncolored.

## A.1 Rainbow connectivity

Rainbow Connectivity (RCon)

**Instance:** A connected graph $G = (V, E)$, and an edge-coloring $\zeta : E \to C$, where $C$ is a set of colors.

**Problem:** Is $G$ rainbow-connected under $\zeta$?

Strong Rainbow Connectivity (SRCon)

**Instance:** A connected graph $G = (V, E)$, and an edge-coloring $\zeta : E \to C$, where $C$ is a set of colors.

**Problem:** Is $G$ strongly rainbow-connected under $\zeta$?

Rainbow Vertex Connectivity (RVCon)

**Instance:** A connected graph $G = (V, E)$, and a vertex-coloring $\psi : V \to C$, where $C$ is a set of colors.

**Problem:** Is $G$ rainbow vertex-connected under $\psi$?

Strong Rainbow Vertex Connectivity (SRVCon)

**Instance:** A connected graph $G = (V, E)$, and a vertex-coloring $\psi : V \to C$, where $C$ is a set of colors.
**Problem:** Is $G$ strongly rainbow vertex-connected under $\psi$?

## A.2 Rainbow coloring

Rainbow $k$-Coloring ($k$-RC)

**Instance:** A connected graph $G = (V, E)$.
**Problem:** Is $\mathrm{rc}(G) \leq k$?

Strong Rainbow $k$-Coloring ($k$-SRC)

**Instance:** A connected graph $G = (V, E)$.
**Problem:** Is $\mathrm{src}(G) \leq k$?

Rainbow Vertex $k$-Coloring ($k$-RVC)

**Instance:** A connected graph $G = (V, E)$.
**Problem:** Is $\mathrm{rvc}(G) \leq k$?

Strong Rainbow Vertex $k$-Coloring ($k$-SRVC)

**Instance:** A connected graph $G = (V, E)$.
**Problem:** Is $\mathrm{srvc}(G) \leq k$?

Rainbow Coloring (RC)

**Instance:** A connected graph $G = (V, E)$, and a positive integer $k$.
**Problem:** Is $\mathrm{rc}(G) \leq k$?

Strong Rainbow Coloring (SRC)

**Instance:** A connected graph $G = (V, E)$, and a positive integer $k$.
**Problem:** Is $\mathrm{src}(G) \leq k$?

Rainbow Vertex Coloring (RVC)

**Instance:** A connected graph $G = (V, E)$, and a positive integer $k$.
**Problem:** Is $\mathrm{rvc}(G) \leq k$?

Strong Rainbow Vertex Coloring (SRVC)

**Instance:** A connected graph $G = (V, E)$, and a positive integer $k$.
**Problem:** Is $\text{srvc}(G) \leq k$?

$k$-SavingRC

**Instance:** A connected graph $G = (V, E)$, and a positive integer $k$.
**Problem:** Is $\text{rc}(G) \leq |E| - k$?

$k$-SavingRVC

**Instance:** A connected graph $G = (V, E)$, and a positive integer $k$.
**Problem:** Is $\text{rvc}(G) \leq |V| - k$?

Maximum Subset Rainbow $k$-Coloring

**Instance:** A graph $G = (V, E)$, a set of anti-edges $S$, and an integer $q$.
**Problem:** Is there an edge-coloring $c : E \to [q]$ such that at least $q$ pairs in $S$ are rainbow-connected?

Maximum Rainbow $k$-Coloring

**Instance:** A graph $G = (V, E)$, and an integer $q$.
**Problem:** Is there an edge-coloring $c : E \to [q]$ such that at least $q$ anti-edges of $G$ are rainbow-connected?

## A.3 Other problems

3-Occurrence 3-SAT

**Instance:** A CNF formula $\varphi$ where each clause is of size at most three, and each variable appears at most three times in $\varphi$.
**Problem:** Is there a satisfying assignment for $\varphi$, i.e., an assignment of truth values to the variables of $\varphi$ such that $\varphi$ evaluates to true?

Channel Assignment

**Instance:** A graph $G = (V, E)$, a weight function $w : E \to \mathbb{N}$, and a positive integer $B$.
**Problem:** Is there a function assigning each vertex $v$ a channel $\phi(x)$ from $[B]$ such that $|\phi(v) - \phi(u)| \geq w(uv)$ for each edge $uv$?

Chromatic Number

**Instance:** A graph $G = (V, E)$, and an integer $k$.
**Problem:** Is there a function $c : V \to [k]$ such that $c(u) \neq c(v)$ for every $uv \in E$?

Clique

**Instance:** A graph $G = (V, E)$, and an integer $k$.
**Problem:** Is there a set $S \subseteq V$ of pairwise adjacent vertices such that $|S| \geq k$?

Colorful Path

**Instance:** A graph $G = (V, E)$ and a (possibly non-proper) $k$-vertex-coloring $c : V \to [k]$.
**Problem:** Does $G$ contain a path of length $k - 1$ on which each of the $k$ colors occur exactly once?

Edge-Colorful Path

**Instance:** A graph $G = (V, E)$ and a (possibly non-proper) $k$-edge-coloring $c : E \to [k]$.
**Problem:** Does $G$ contain a path of length $k$ on which each of the $k$ colors occur exactly once?

Connected Vertex Cover

**Instance:** A graph $G = (V, E)$, and an integer $k$.
**Problem:** Is there a set $S \subseteq V$ such that $|S| \leq k$, the induced subgraph $G[S]$ is connected, and $S \cap e \neq \emptyset$ holds for every edge $e \in E$?

Graph Minor

**Instance:** Two graphs $G = (V, E)$, and $G' = (V', E')$
**Problem:** Is $G$ a minor of $G'$?

Graph Motif

**Instance:** A vertex-colored graph $G = (V, E)$, and a multiset $M$ of colors.
**Problem:** Does $G$ contain a connected subgraph whose colors agree with $M$?

Set Cover

**Instance:** A universe $U = [n]$, a set system $\mathcal{S} \subseteq 2^U$, and an integer $f$.
**Problem:** Does $\mathcal{S}$ have a set cover of size at most $f$, i.e., a subset $\mathcal{C} \subseteq \mathcal{S}$ with $|\mathcal{C}| \leq t$ such that $\bigcup_{S \in \mathcal{C}} S = U$?

Set Partitioning

**Instance:** A universe $U = [n]$, a set system $\mathcal{F} \subseteq 2^U$, and an integer $t$.
**Problem:** Does $\mathcal{F}$ have a set partitioning of size at most $t$, i.e., a set cover $\mathcal{C}$ such that for every $S, S' \in \mathcal{C}$ with $S \neq S'$, we have $S \cap S' = \emptyset$?

Steiner Tree

**Instance:** A graph $G = (V, E)$, an integer $k$, and a set of terminals $S \subseteq V$.
**Problem:** Is there a tree $T$ in $G$ with at most $k$ edges such that $S \subseteq V(T)$?

Subgraph Homomorphism

**Instance:** Two graphs $G = (V, E)$, and $G' = (V', E')$
**Problem:** Does $G$ have a subgraph homomorphic to $G'$?

Subgraph Isomorphism

**Instance:** Two graphs $G = (V, E)$, and $G' = (V', E')$
**Problem:** Does $G$ contain a subgraph isomorphic to $G'$?

Subset Sum

**Instance:** A set of $n$ integers $U$, and an integer $t$.
**Problem:** Is there a subset $U' \subseteq U$ such that the sum of elements in $U'$ equals $t$?

# Bibliography

[1] Prabhanjan Ananth, Meghana Nasre, and Kanthi K. Sarpatwar. Rainbow Connectivity: Hardness and Tractability. In: *Proceedings of the 31st IARCS Annual Conference on Foundations of Software Technology and Theoretical Computer Science, FSTTCS 2011, Mumbai, India, December 12–14*. 2011, pp. 241–251.

[2] Andreas Björklund, Petteri Kaski, and Łukasz Kowalik. Probably Optimal Graph Motifs. In: *Proceedings of the 30th International Symposium on Theoretical Aspects of Computer Science, STACS 2013, Kiel, Germany, February 27–March 2*. Vol. 20. 2013, pp. 20–31.

[3] Hans L. Bodlaender. A partial $k$-arboretum of graphs with bounded treewidth. *Theoretical Computer Science* 209(1–2) (1998), pp. 1–45.

[4] Hans L. Bodlaender, John R. Gilbert, Hjálmtyr Hafsteinsson, and Ton Kloks. Approximating Treewidth, Pathwidth, Frontsize, and Shortest Elimination Tree. *Journal of Algorithms* 18(2) (1995), pp. 238–255.

[5] Fred Buckley and Martin Lewinter. A note on graphs with diameter-preserving spanning trees. *Journal of Graph Theory* 12(4) (1988), pp. 525–528.

[6] Yair Caro, Arie Lev, Yehuda Roditty, Zsolt Tuza, and Raphael Yuster. On rainbow connection. *Electron. J. Combin* 15(1) (2008), R57.

[7] Sourav Chakraborty, Eldar Fischer, Arie Matsliah, and Raphael Yuster. Hardness and algorithms for rainbow connection. *Journal of Combinatorial Optimization* 21(3) (2009), pp. 330–347.

[8] L. Sunil Chandran, Anita Das, Deepak Rajendraprasad, and Nithin M. Varma. Rainbow connection number and connected dominating sets. *Journal of Graph Theory* 71(2) (2012), pp. 206–218.

[9] L. Sunil Chandran and Deepak Rajendraprasad. Inapproximability of Rainbow Colouring. In: *Proceedings of the 33rd IARCS Annual Conference on Foundations of Software Technology and Theoretical Computer Science, FSTTCS 2013, Guwahati, India, December 12–14*. 2013, pp. 153–162.

[10] L. Sunil Chandran, Deepak Rajendraprasad, and Marek Tesař. Rainbow colouring of split graphs. *Discrete Applied Mathematics* (2015). To appear.

[11] Gary Chartrand, Garry L. Johns, Kathleen A. McKeon, and Ping Zhang. Rainbow connection in graphs. *Mathematica Bohemica* 133(1) (2008).

[12] Gary Chartrand, Futaba Okamoto, and Ping Zhang. Rainbow trees in graphs and generalized connectivity. *Networks* 55(4) (2010), pp. 360–367.

[13] Gary Chartrand and Ping Zhang. *Chromatic graph theory*. CRC press, 2008.

[14] Lily Chen, Xueliang Li, and Huishu Lian. Further hardness results on the rainbow vertex-connection number of graphs. *Theoretical Computer Science* 481 (2013), pp. 18–23.

[15] Lily Chen, Xueliang Li, and Yongtang Shi. The complexity of determining the rainbow vertex-connection of a graph. *Theoretical Computer Science* 412(35) (2011), pp. 4531–4535.

[16] Xue-gang Chen, Liang Sun, and Hua-ming Xing. Characterization of graphs with equal domination and connected domination numbers. *Discrete Mathematics* 289(1–3) (2004), pp. 129–135.

[17] Benny Chor, Mike Fellows, and David Juedes. Linear Kernels in Linear Time, or How to Save $k$ Colors in $O(n^2)$ Steps. In: *Proceedings of the 30th International Workshop in Graph-Theoretic Concepts in Computer Science, WG 2004, Bad Honnef, Germany, June 21–23.* 2005, pp. 257–269.

[18] Bruno Courcelle. The Monadic Second-Order Logic of Graphs I. Recognizable Sets of Finite Graphs. *Information and Computation* (1990), pp. 12–75.

[19] Bruno Courcelle, Johann A. Makowsky, and Udi Rotics. Linear time solvable optimization problems on graphs of bounded clique-width. *Theory of Computing Systems* 33(2) (2000), pp. 125–150.

[20] Marek Cygan, Holger Dell, Daniel Lokshtanov, Dániel Marx, Jesper Nederlof, Yoshio Okamoto, Ramamohan Paturi, Saket Saurabh, and Magnus Wahlström. On Problems as Hard as CNF-SAT. *ACM Transactions on Algorithms* 12(3) (2016), 41:1–41:24.

[21] Marek Cygan, Fedor V. Fomin, Alexander Golovnev, Alexander S. Kulikov, Ivan Mihajlin, Jakub Pachocki, and Arkadiusz Socała. Tight Bounds for Graph Homomorphism and Subgraph Isomorphism. In: *Proceedings of the 27th Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2016, Arlington, VA, USA, January 10-12.* 2016, pp. 1643–1649.

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

[23] Jitender S. Deogun and Dieter Kratsch. Diametral Path Graphs. In: *Proceedings of the 21st International Workshop in Graph-Theoretic Concepts in Computer Science, WG 1995, Aachen, Germany, June 20–22.* 1995, pp. 344–357.

[24] Reinhard Diestel. *Graph Theory.* Springer-Verlag Heidelberg, 2005.

[25] Guoli Ding and Stan Dziobiak. On 3-Connected Graphs of Path-Width at Most Three. *SIAM Journal on Discrete Mathematics* 27(3) (2013), pp. 1514–1526.

[26] Paul Dorbec, Ingo Schiermeyer, Elżbieta Sidorowicz, and Éric Sopena. Rainbow connection in oriented graphs. *Discrete Applied Mathematics* 179 (2014), pp. 69–78.

[27] Rodney G. Downey and Michael R. Fellows. *Fundamentals of Parameterized Complexity.* Springer, 2013.

[28] Richard J. Duffin. Topology of series-parallel networks. *Journal of Mathematical Analysis and Applications* 10(2) (1965), pp. 303–318.

[29] Jan Ekstein, Přemysl Holub, Tomáš Kaiser, Maria Koch, Stephan Matos Camacho, Zdeněk Ryjáček, and Ingo Schiermeyer. The rainbow connection number of 2-connected graphs. *Discrete Mathematics* 313(19) (2013), pp. 1884–1892.

[30] David Eppstein. Diameter and Treewidth in Minor-Closed Graph Families. *Algorithmica* 27(3) (2000), pp. 275–291.

[31] Jiří Fiala, Petr A. Golovach, and Jan Kratochvíl. Parameterized complexity of coloring problems: Treewidth versus vertex cover. *Theoretical Computer Science* 412(23) (2011), pp. 2513–2523.

[32] Fedor V. Fomin, Dieter Kratsch, and Gerhard J. Woeginger. Exact (Exponential) Algorithms for the Dominating Set Problem. In: *Proceedings of the 30th International Workshop in Graph-Theoretic Concepts in Computer Science, WG 2004, Bad Honnef, Germany, June 21–23.* 2005, pp. 245–256.

[33] András Frank. Personal communication. January 14th, 2015.

[34] András Frank and László Szegő. Constructive characterizations for packing and covering with trees. *Discrete Applied Mathematics* 131(2) (2003), pp. 347–371.

[35] Philippe Galinier, Michel Habib, and Christophe Paul. Chordal graphs and their clique graphs. In: *Proceedings of the 21st International Workshop in Graph-Theoretic Concepts in Computer Science, WG 1995, Aachen, Germany, June 20–22.* 1995, pp. 358–371.

[36] Robert Ganian and Petr Hliněný. On parse trees and Myhill-Nerode-type tools for handling graphs of bounded rank-width. *Discrete Applied Mathematics* 158(7) (2010), pp. 851–867.

[37] Fănică Gavril. The intersection graphs of subtrees in trees are exactly the chordal graphs. *Journal of Combinatorial Theory, Series B* 16(1) (1974), pp. 47–56.

[38] Paul C. Gilmore and Alan J. Hoffman. A characterization of comparability graphs and of interval graphs. *Canadian Journal of Mathematics* 16(539–548) (1964), p. 4.

[39] Michel Habib and Juraj Stacho. Reduced clique graphs of chordal graphs. *European Journal of Combinatorics* 33(5) (2012), pp. 712–735.

[40] Refael Hassin and Arie Tamir. On the minimum diameter spanning tree problem. *Information Processing Letters* 53(2) (1995), pp. 109–111.

[41] Ellis Horowitz and Sartaj Sahni. Computing partitions with applications to the knapsack problem. *Journal of the ACM (JACM)* 21(2) (1974), pp. 277–292.

[42] Xiaolong Huang, Xueliang Li, Yongtang Shi, Jun Yue, and Yan Zhao. Rainbow connections for outerplanar graphs with diameter 2 and 3. *Applied Mathematics and Computation* 242 (2014), pp. 277–280.

[43] Russell Impagliazzo and Ramamohan Paturi. On the Complexity of k-SAT. *Journal of Computer and System Sciences* 62(2) (2001), pp. 367–375.

[44] Russell Impagliazzo, Ramamohan Paturi, and Francis Zane. Which Problems Have Strongly Exponential Complexity? *Journal of Computer and System Sciences* 63(4) (2001), pp. 512–530.

[45] Shaun N. Joseph and Lisa C. DiPippo. Pseudo-scheduling: A New Approach to the Broadcast Scheduling Problem. In: *Algorithms for Sensor Systems: 8th International Symposium on Algorithms for Sensor Systems, Wireless Ad Hoc Networks and Autonomous Mobile Entities, ALGOSENSORS 2012, Ljubljana, Slovenia, September 13-14.* 2013, pp. 93–104.

[46] Nina Kamčev, Michael Krivelevich, and Benny Sudakov. Some Remarks on Rainbow Connectivity. *Journal of Graph Theory* (2015). To appear.

[47] Haim Kaplan and Ron Shamir. Pathwidth, Bandwidth, and Completion Problems to Proper Interval Graphs with Small Cliques. *SIAM Journal on Computing* 25(3) (1996), pp. 540–561.

[48] Melissa Keranen and Juho Lauri. Computing Minimum Rainbow and Strong Rainbow Colorings of Block Graphs. Submitted.

[49] Melissa Keranen and Juho Lauri. Upper Bounds on the Strong Rainbow Connection Number via Covers and Forests. In preparation.

[50] Subhash Khot and Venkatesh Raman. Parameterized complexity of finding subgraphs with hereditary properties. *Theoretical Computer Science* 289(2) (2002), pp. 997–1008.

[51] Jon Kleinberg and Éva Tardos. *Algorithm Design.* Pearson Education, 2006.

[52] Dieter Kratsch, Peter Damaschke, and Anna Lubiw. Dominating cliques in chordal graphs. *Discrete Mathematics* 128(1) (1994), pp. 269–275.

[53] Michael Krivelevich and Raphael Yuster. The rainbow connection of a graph is (at most) reciprocal to its minimum degree. *Journal of Graph Theory* 63(3) (2010), pp. 185–191.

[54] Juho Lauri and Henri Riihimäki. A tight upper bound on the rainbow connection number of chordal diametral path graphs. Submitted.

[55] Van Bang Le and Zsolt Tuza. *Finding optimal rainbow connection is hard.* Tech. rep. CS-03-09. Universität Rostock, 2009.

[56] Hengzhe Li, Xueliang Li, and Sujuan Liu. Rainbow connection of graphs with diameter 2. *Discrete Mathematics* 312(8) (2012), pp. 1453–1457.

[57] Xueliang Li, Yaping Mao, and Yongtang Shi. The strong rainbow vertex-connection of graphs. *Utilitas Mathematica* 93 (2014), pp. 213–223.

[58] Xueliang Li, Yongtang Shi, and Yuefang Sun. Rainbow Connections of Graphs: A Survey. *Graphs and Combinatorics* 29(1) (2012), pp. 1–38.

[59] Xueliang Li and Yuefang Sun. Rainbow connection numbers of line graphs. *Ars Combinatoria* 100 (2011), pp. 449–463.

[60] Xueliang Li and Yuefang Sun. *Rainbow connections of graphs.* Springer, 2012.

[61] Xueliang Li and Yuefang Sun. Upper Bounds for the Rainbow Connection Numbers of Line Graphs. *Graphs and Combinatorics* 28(2) (2011), pp. 251–263.

[62] Daniel Lokshtanov, Dániel Marx, and Saket Saurabh. Lower bounds based on the Exponential Time Hypothesis. *Bulletin of the EATCS* (105) (2011), pp. 41–72.

[63] Mohammad Mahdian. On the computational complexity of strong edge coloring. *Discrete Applied Mathematics* 118(3) (2002), pp. 239–248.

[64] Dániel Marx. What's Next? Future Directions in Parameterized Complexity. In: *The Multivariate Algorithmic Revolution and Beyond: Essays Dedicated to Michael R. Fellows on the Occasion of His 60th Birthday.* 2012, pp. 469–496.

[65] Hadrien Mélot. Facet defining inequalities among graph invariants: The system GraPHedron. *Discrete Applied Mathematics* 156(10) (2008), pp. 1875–1891.

[66] Sylvia D. Monson, Norman J. Pullman, and Rolf Rees. A survey of clique and biclique coverings and factorizations of (0,1)-matrices. *Bulletin of the Institute of Combinatorial Mathematics and its Applications* 14 (1995), pp. 17–86.

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

[68] Rolf Niedermeier. *Invitation to fixed-parameter algorithms.* Oxford University Press Oxford, 2006.

[69] Fred S. Roberts. Applications of edge coverings by cliques. *Discrete Applied Mathematics* 10(1) (1985), pp. 93–109.

[70] Fred S. Roberts. Indifference graphs. In: *Proof techniques in graph theory.* 1969, pp. 139–146.

[71] Arkadiusz Socała. Tight lower bound for the channel assignment problem. In: *Proceedings of the 26th Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2015, San Diego, CA, USA, January 4-6.* 2015, pp. 662–675.

[72] Joel G. Stemple and Mark E. Watkins. On planar geodetic graphs. *Journal of Combinatorial Theory* 4(2) (1968), pp. 101–117.

[73] Craig A. Tovey. A simplified NP-complete satisfiability problem. *Discrete Applied Mathematics* 8(1) (1984), pp. 85–89.

[74] Kei Uchizawa, Takanori Aoki, Takehiro Ito, Akira Suzuki, and Xiao Zhou. On the Rainbow Connectivity of Graphs: Complexity and FPT Algorithms. *Algorithmica* 67(2) (2013), pp. 161–179.

[75] Vijay V. Vazirani. *Approximation algorithms.* Springer, 2013.

[76] David Zuckerman. Linear Degree Extractors and the Inapproximability of Max Clique and Chromatic Number. *Theory of Computing* 3(6) (2007), pp. 103–128.

# Paper 1

Juho Lauri.

**Further hardness results on rainbow and strong rainbow connectivity.**

© Elsevier B.V., 2016. Discrete Applied Mathematics 201 (2016), pp. 191–200.

doi:10.1016/j.dam.2015.07.041

Discrete Applied Mathematics 201 (2016) 191–200

Contents lists available at ScienceDirect

Discrete Applied Mathematics

journal homepage: www.elsevier.com/locate/dam

# Further hardness results on rainbow and strong rainbow connectivity

Juho Lauri

*Department of Mathematics, Tampere University of Technology, Korkeakoulunkatu 1, 33720 Tampere, Finland*

ARTICLE INFO

*Article history:*
Received 28 July 2014
Received in revised form 5 May 2015
Accepted 29 July 2015
Available online 31 August 2015

*Keywords:*
Rainbow connectivity
Computational complexity

ABSTRACT

A path in an edge-colored graph is *rainbow* if no two edges of it are colored the same. The graph is said to be *rainbow connected* if there is a rainbow path between every pair of vertices. If there is a rainbow shortest path between every pair of vertices, the graph is *strong rainbow connected*. We consider the complexity of the problem of deciding if a given edge-colored graph is rainbow or strong rainbow connected. These problems are called Rainbow connectivity and Strong rainbow connectivity, respectively. We prove both problems remain NP-complete on interval outerplanar graphs and $k$-regular graphs for $k \geq 3$. Previously, no graph class was known where the complexity of the two problems would differ. We show that for block graphs, which form a subclass of chordal graphs, Rainbow connectivity is NP-complete while Strong rainbow connectivity is in P. We conclude by considering some tractable special cases, and show for instance that both problems are in XP when parameterized by tree-depth.

## 1. Introduction

Let $G$ be an edge-colored undirected graph that is simple and finite. A path in $G$ is *rainbow* if no two edges of it are colored the same. The graph $G$ is *rainbow connected* if there is a rainbow path between every pair of vertices. If there is a rainbow shortest path between every pair of vertices, $G$ is *strong rainbow connected*. Clearly, a strong rainbow connected graph is also rainbow connected. The minimum number of colors needed to make $G$ rainbow connected is known as the *rainbow connection number* and is denoted by $\mathrm{rc}(G)$. Likewise, the minimum number of colors needed to make $G$ strong rainbow connected is known as the *strong rainbow connection number* and is denoted by $\mathrm{src}(G)$. The concept of rainbow connectivity was introduced by Chartrand et al. [4] in 2008, and it has applications in data transfer and networking. The *diameter* of a graph, denoted by $\mathrm{diam}(G)$, is the largest distance between two vertices of $G$. Clearly, $\mathrm{diam}(G)$ is a lower bound for $\mathrm{rc}(G)$. On the other hand, a trivial upper bound for $\mathrm{rc}(G)$ is $m$, where $m$ is the number of edges in $G$. Finally, because each strong rainbow connected graph is also rainbow connected, we have that $\mathrm{diam}(G) \leq \mathrm{rc}(G) \leq \mathrm{src}(G) \leq m$. For less trivial bounds and more, we refer the reader to the books [5,18], or the recent survey [17].

A similar concept was introduced for vertex-colored graphs by Krivelevich and Yuster [14]. A vertex-colored graph $H$ is *rainbow vertex-connected* if every pair of vertices is connected by a path whose internal vertices have distinct colors. The minimum number of colors needed to make $H$ rainbow vertex-connected is known as the *rainbow vertex-connection number* and is denoted by $\mathrm{rvc}(H)$. Li et al. [16] investigated the *strong rainbow vertex-connection number* as a natural variant. A vertex-colored graph is *strong rainbow vertex-connected* if every pair of vertices is connected by a shortest path whose internal vertices have distinct colors. The minimum number of colors needed to make $H$ strong rainbow vertex-connected

*E-mail address:* juho.lauri@tut.fi.

http://dx.doi.org/10.1016/j.dam.2015.07.041
0166-218X/© 2015 Elsevier B.V. All rights reserved.

is known as the *strong rainbow vertex-connection number* and is denoted by $\text{srvc}(H)$. For rainbow vertex-connection numbers or other rainbow connection numbers outside of our scope we refer the reader to [17].

Rainbow connectivity can be motivated by the following example from the domain of networking. Suppose we have a network of agents represented as a graph. Each vertex in the graph represents an agent, and an edge between two agents is a link. An agent in the network wishes to communicate with every other agent in the network by sending messages. A message sent from agent $A$ to agent $B$ is routed through other agents that act as intermediaries. This communication path uses links between agents, and each link uses a channel. For the message to get through, we require that each link on the communication path receives a distinct channel. Given a network of agents $G$, our objective is to ensure each pair of agents can establish a communication path, while also minimizing the number of channels needed. The minimum number of channels we need is exactly $\text{rc}(G)$.

Chakraborty et al. [2] showed that it is NP-complete to decide if $\text{rc}(G) \leq k$ for $k = 2$. Ananth et al. [1] proved the problem remains hard for $k \geq 3$ as well. Chandran and Rajendraprasad [3] proved there is no polynomial time algorithm to rainbow color graphs with less than twice the optimum number of colors, unless P = NP. Computing the strong rainbow connection number is known to be hard as well. Chartrand et al. [4] proved $\text{rc}(G) = 2$ if and only if $\text{src}(G) = 2$, so deciding if $\text{src}(G) \leq k$ is NP-complete for $k = 2$. Ananth et al. [1] showed the problem remains NP-complete for $k \geq 3$ even when $G$ is bipartite [1]. In the same paper, they also showed there is no polynomial time algorithm for approximating the strong rainbow connection number of an $n$-vertex graph within a factor of $n^{1/2-\epsilon}$, where $\epsilon > 0$ unless NP = ZPP.

Given that it is hard to compute both the rainbow and the strong rainbow connection number, it is natural to ask if it is easier to verify if a given edge-colored graph is rainbow or strong rainbow connected. In this paper, we are concerned with the complexity of the following two decision problems:

Rainbow connectivity
**Instance:** An undirected graph $G = (V, E)$, and an edge-coloring $\chi : E \to C$, where $C$ is a set of colors
**Question:** Is $G$ rainbow connected under $\chi$?

Strong rainbow connectivity
**Instance:** An undirected graph $G = (V, E)$, and an edge-coloring $\chi : E \to C$, where $C$ is a set of colors
**Question:** Is $G$ strong rainbow connected under $\chi$?

Out of these two problems, Rainbow connectivity has gained considerably more attention in the literature. Chakraborty et al. [2] observed the problem is easy when the number of colors $|C|$ is bounded from above by a constant. However, they proved that for an arbitrary coloring, the problem is NP-complete. Building on their result, Li et al. [15] proved Rainbow connectivity remains NP-complete for bipartite graphs. Furthermore, the problem is NP-complete even for bipartite planar graphs as shown by Huang et al. [11]. Recently, Uchizawa et al. [23] complemented these results by showing Rainbow connectivity is NP-complete for outerplanar graphs, and even for series–parallel graphs. In the same paper, the authors also gave some positive results. Namely, they showed the problem is in P for cactus graphs, which form a subclass of outerplanar graphs. Furthermore, they settled the precise complexity of the problem from a viewpoint of graph diameter by showing the problem is in P for graphs of diameter 1, but NP-complete already for graphs of diameter greater than or equal to 2. To the best of our knowledge, Uchizawa et al. [23] were the only ones to consider Strong rainbow connectivity. They showed the problem is in P for cactus graphs, but NP-complete for outerplanar graphs. We shortly mention similar hardness results are known for deciding if a given vertex-colored is rainbow vertex-connected (see e.g. [6,15,23]).

A *fixed-parameter algorithm* (FPT) solves a problem with an input instance of size $n$ and a parameter $k$ in $f(k) \cdot n^{O(1)}$ time for some computable function $f$ depending solely on $k$. That is, for every fixed parameter value it yields a solution in polynomial time and the degree of the polynomial is independent from $k$. Uchizawa et al. [23] gave FPT algorithms for both problems on general graphs when parameterized by the number of colors $k = |C|$. These algorithms run in $O(k2^k mn)$ time and $O(k2^k n)$ space, where $n$ and $m$ are the number of vertices and edges in the input graph, respectively. These algorithms imply both Rainbow connectivity and Strong rainbow connectivity are solvable in polynomial time for any $n$-vertex graph if $|C| = O(\log n)$.

In this paper, we prove both Rainbow connectivity and Strong rainbow connectivity remain NP-complete for interval outerplanar graphs. We then consider the class of block graphs, which form a subclass of chordal graphs. Interestingly, for block graphs Rainbow connectivity is NP-complete, while Strong rainbow connectivity is in P. To the best of our knowledge, this is the first graph class known for which the complexity of these two problems differ. Both problems are easy on 2-regular graphs. However, we show that both problems become NP-complete on cubic graphs, and further generalize this for $k$-regular graphs, where $k > 3$. This completely settles the complexity of both problems from the viewpoint of regularity.

## 2. Preliminaries

All graphs in this paper are simple, finite, and undirected. We begin by defining the graph classes we consider in this work. For graph theoretic concepts not defined here, we refer the reader to [8]. For an integer $n$, we write $[n] = \{1, 2, \ldots, n\}$.

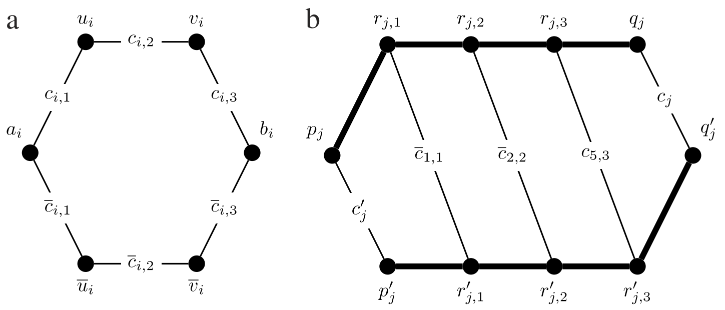


**Fig. 1.** (a) A variable gadget $X_i$ for the variable $x_i$, and (b) a clause gadget $C_j$ for the clause $c_j = (x_1 \vee x_2 \vee \neg x_5)$, where $x_1$ is the first literal of $x_1$, $x_2$ is the second literal of $x_2$, and $\neg x_5$ is the third literal of $x_5$.

A *chord* is an edge joining two non-consecutive vertices in a cycle. A graph is *chordal* if every cycle of length 4 or more has a chord. Equivalently, a graph is chordal if it contains no induced cycle of length 4 or more. A *cut vertex* is a vertex whose removal will disconnect the graph. A *biconnected graph* is a connected graph having no cut vertices. A *block graph* is an undirected graph where every maximal biconnected component, known as a *block*, is a clique. In a block graph $G$, different blocks intersect in at most one vertex, which is a cut vertex of $G$. In other words, every edge of $G$ lies in a unique block, and $G$ is the union of its blocks. It is easy to see that a block graph is chordal. Another well-known subclass of chordal graphs is formed by *interval graphs*. To define such graphs, we will first introduce the notion of *clique trees*. A clique tree of a connected chordal graph $G$ is any tree $T$ whose vertices are the maximal cliques of $G$ such that for every two maximal cliques $C_i$, $C_j$, each clique on the path from $C_i$ to $C_j$ in $T$ contains $C_i \cap C_j$. Chordal graphs are precisely the class of graphs that admit a clique tree representation [9]. As shown by Gilmore and Hoffman [10], a graph is an interval graph if and only if it admits a clique tree that is a path. A graph is *planar* if it can be embedded in the plane without crossing edges. A graph is *outerplanar* if it has a crossing-free embedding in the plane such that all vertices are on the same face. Finally, the *degree* of a vertex is the number of edges incident to it. A graph is $k$-regular if the degree of each of its vertices is exactly $k$. Specifically, a 3-regular graph is known as a *cubic graph*.

The 3-Occurrence 3-SAT problem is a variant of the 3-SAT problem where every variable occurs at most three times. The NP-completeness of Rainbow connectivity and Strong rainbow connectivity for outerplanar graphs were shown by a reduction from the 3-Occurrence 3-SAT problem by Uchizawa et al. [23]. Often, it does not matter if one refers by 3-SAT to the variant of 3-SAT where each clause has exactly 3 literals, or the variant where each clause has at most 3 literals since both are NP-complete. However, for 3-Occurrence 3-SAT this distinction is crucial. The variant where every clause has exactly 3 literals is in P because every such instance is satisfiable as shown by Tovey [22]. The variant where every clause has at most 3 literals is however NP-complete [19].

This distinction is not explicitly made by Uchizawa et al. [23]. However, it is not hard to modify their clause gadgets to allow for less than 3 literals. In other words, this does not affect the correctness of their reductions. The clause gadgets corresponding to clauses of size one and two can be found from the Appendix of this article. Their reductions greatly inspire ours, and thus we also reduce from the 3-Occurrence 3-SAT problem, where each clause has at most 3 literals. We begin by describing their construction as our reductions are based on it. We tighten their result slightly by observing the resulting graph is both bipartite and outerplanar.

**Theorem 1** (*Uchizawa et al.* [23]). Rainbow connectivity *is* NP*-complete when restricted to the class of bipartite outerplanar graphs.*

**Construction**: We first observe the problem is in NP with the certificate being a set of colored paths, one for each pair of vertices. It is then simple to decide if a given path is rainbow. Given a 3-Occurrence 3-SAT formula $\phi = \bigwedge_{j=1}^{m} c_i$ over variables $x_1, x_2, \ldots, x_n$, we construct a graph $G_\phi$ and an edge-coloring $\chi$ such that $\phi$ is satisfiable if and only if $G_\phi$ is rainbow connected under $\chi$. We first describe the construction of $G_\phi$, and then the edge-coloring $\chi$ of $G_\phi$.

For each variable $x_i$, $i \in [n]$, we build a *variable gadget* $X_i$. A variable gadget $X_i$ is a cycle graph $C_6$ embedded in the plane on vertices $a_i, u_i, v_i, b_i, \bar{v}_i, \bar{u}_i$ in clockwise order. For each clause $c_j, j \in [m]$, we build a *clause gadget* $C_j$. A clause gadget $C_j$ is built by starting from a cycle graph $C_{10}$ embedded in the plane on vertices $p_j, r_{j,1}, r_{j,2}, r_{j,3}, q_j, q'_j, r'_{j,3}, r'_{j,2}, r'_{j,1}, p'_j$ in clockwise order, and by adding chords $(r_{j,1}, r'_{j,1})$, $(r_{j,2}, r'_{j,2})$, and $(r_{j,3}, r'_{j,3})$. These chords correspond to the three literals the clause $c_j$ has. Both a variable gadget and a clause gadget are shown in Fig. 1.

We connect $X_i$ with $X_{i+1}$ by adding an edge $(b_i, a_{i+1})$ for each $1 \le i < n$. Then, we connect $C_j$ with $C_{j+1}$ by adding an edge $(q'_j, p_{j+1})$ for each $1 \le j < m$. Likewise, we connect the two components together by adding the edge $(b_n, p_1)$. We then add one vertex $t$, and the edge $(q'_m, t)$. Finally, we build a path of length $m$ on vertices $s_1, s_2, \ldots, s_m$, and connect it to $G_\phi$ by adding the edge $(s_m, a_1)$. This completes the construction of $G_\phi$. We can verify $G_\phi$ is indeed a bipartite outerplanar graph.

We now describe the edge-coloring $\chi$ given to the edges of $G_\phi$. Notice there are exactly two paths between $a_i$ and $b_i$ in a variable gadget $X_i$. Intuitively, taking the path from $a_i$ to $b_i$ through $u_i$ and $v_i$ corresponds to setting $x_i = 1$ in the formula $\phi$.

**Table 1**
Summary of known complexity results for Rainbow connectivity and Strong rainbow connectivity.

| Graph class | Rainbow connectivity | Strong rainbow connectivity |
|---|---|---|
| Bounded diameter $\geq 2$ | NP-complete[a] | P [ Theorem 11] |
| Series–parallel | NP-complete[a] | NP-complete[a] |
| Bipartite planar | NP-complete[b] | NP-complete[a] |
| Bipartite outerplanar | NP-complete[a] | NP-complete[a] |
| Interval outerplanar | NP-complete [ Theorem 4] | NP-complete [ Corollary 5] |
| Cactus | P[a] | P[a] |
| $k$-regular, $k \geq 3$ | NP-complete [ Theorem 9] | NP-complete [ Corollary 10] |
| Block | NP-complete | P [ Corollary 13] |
| Interval block | NP-complete [ Theorem 6] | P |
| Tree | P | P |

[a] Stands for [23].
[b] Stands for [11].

We refer to this path as the *positive $X_i$ path*. We color the three edges $(a_i, u_i)$, $(u_i, v_i)$, and $(v_i, b_i)$ with colors $c_{i,1}$, $c_{i,2}$, and $c_{i,3}$, respectively. Taking the path from $a_i$ to $b_i$ through $\overline{u}_i$ and $\overline{v}_i$ corresponds to setting $x_i = 0$ in the formula $\phi$. We refer to this path as the *negative $X_i$ path*. The three edges $(a_i, \overline{u}_i)$, $(\overline{u}_i, \overline{v}_i)$ and $(\overline{v}_i, b_i)$ receive the colors $\overline{c}_{i,1}$, $\overline{c}_{i,2}$ and $\overline{c}_{i,3}$, respectively. The coloring of a variable gadget $X_i$ is illustrated in Fig. 1(a).

Recall a variable $x_i$ appears at most three times in $\phi$. We refer to the first occurrence of $x_i$ as the *first literal of* $x_i$, the second occurrence of $x_i$ as the *second literal of* $x_i$, and finally the third occurrence of $x_i$ as the *third literal of* $x_i$. If a clause has two or three literals of a same variable, the tie is broken arbitrarily. In a clause gadget $C_j$, we color the edge $(p_j, p'_j)$ with the color $c'_j$, and the edge $(q_j, q'_j)$ with the color $c_j$. For each $k \in [3]$, we denote the $k$th literal in the $j$th clause by $l_{j,k}$. We color the edge $(r_{j,k}, r'_{j,k})$ as follows:

$$\chi((r_{j,k}, r'_{j,k})) = \begin{cases} \overline{c}_{i,1} & \text{if } l_{j,k} \text{ is a positive literal and the first literal of } x_i \\ \overline{c}_{i,2} & \text{if } l_{j,k} \text{ is a positive literal and the second literal of } x_i \\ \overline{c}_{i,3} & \text{if } l_{j,k} \text{ is a positive literal and the third literal of } x_i \\ c_{i,1} & \text{if } l_{j,k} \text{ is a negative literal and the first literal of } x_i \\ c_{i,2} & \text{if } l_{j,k} \text{ is a negative literal and the second literal of } x_i \\ c_{i,3} & \text{if } l_{j,k} \text{ is a negative literal and the third literal of } x_i. \end{cases}$$

The edge $(q'_j, p_{j+1})$, for each $1 \leq j < m$, receives the color $c'_j$, while the edge $(q'_m, t)$ is colored with $c'_m$. The coloring of a clause gadget $C_j$ is shown in Fig. 1(b).

Finally, we color each edge $(s_j, s_{j+1})$ with the color $c_j$ for each $1 \leq j < m$. The edge $(s_m, a_1)$ is colored with the color $c_m$. Every other uncolored edge of $G_\phi$ receives a fresh new color, that does not appear in $G_\phi$. Formally, these are precisely the edges in $U \cup W$, where $U = \{(b_i, a_{i+1}) \mid 1 \leq i < n\} \cup \{(b_n, p_1)\}$ and

$$W = \{(p_j, r_{j,1}), (r_{j,1}, r_{j,2}), (r_{j,2}, r_{j,3}), (r_{j,3}, q_j) \mid 1 \leq j \leq m\} \cup \{(q'_j, r'_{j,3}), (r'_{j,3}, r'_{j,2}), (r'_{j,2}, r'_{j,1}), (r'_{j,1}, p'_j) \mid 1 \leq j \leq m\}.$$

The edges in $W$ correspond precisely to the edges drawn with thick lines in Fig. 1(b), for each clause gadget $C_j$. This completes the edge-coloring $\chi$ of $G_\phi$. The following claim is true for $G_\phi$, and it furthermore proves Theorem 1.

**Lemma 2** (*Uchizawa et al. [23]*). *The graph $G_\phi$ is rainbow connected under $\chi$ if and only if $G_\phi$ has a rainbow path between the vertices $s_1$ and $t$. Furthermore, there is a rainbow path between $s_1$ and $t$ if and only if the formula $\phi$ is satisfiable.*

In the previous reduction, by observing every pair of vertices is rainbow connected by a rainbow shortest path given a satisfiable instance of $\phi$, Uchizawa et al. [23] also got the following.

**Theorem 3** (*Uchizawa et al. [23]*). Strong rainbow connectivity *is* NP-*complete when restricted to the class of bipartite outerplanar graphs.*

## 3. Hardness results

In this section, we give new hardness results for both Rainbow connectivity and Strong rainbow connectivity. All of our hardness results will follow by a reduction from the 3-Occurrence 3-SAT problem, and will essentially be based on Theorem 1. For the sake of brevity, and similarly to Theorem 1, we will present our constructions assuming each clause is of size three. The clause gadgets corresponding to clauses of size one and two can be found in the Appendix for each graph class.

We summarize the known complexity results for both problems in Table 1 along with our new results.

## *3.1. Rainbow and strong rainbow connectivity are* NP*-complete for interval outerplanar graphs*

In this subsection, we prove Rainbow connectivity and Strong rainbow connectivity remain NP-complete for interval outerplanar graphs.

**Theorem 4.** Rainbow connectivity *is* NP*-complete when restricted to the class of interval outerplanar graphs.*

**Proof.** We assume the same terminology as in Theorem 1. Given a 3-Occurrence 3-SAT instance $\phi$, we first build a graph $G_\phi$ along with its edge-coloring $\chi$ precisely as in Theorem 1. For clarity, we then rename $G_\phi$ to $G_\phi^M$, and $\chi$ to $\chi_M$.

A variable gadget $X_i^M$ is obtained from $X_i$ by adding three chords $(u_i, \overline{u}_i)$, $(u_i, \overline{v}_i)$, and $(v_i, \overline{v}_i)$, and coloring each with a new color $\overline{c}_i$. Next, a clause gadget $C_j^M$ is obtained from $C_j$ by adding four chords $(r_{j,1}, p_j')$, $(r_{j,2}, r_{j,1}')$, $(r_{j,3}, r_{j,2}')$, and $(q_j, r_{j,3}')$. Each of these four chords receive the color $c_j'$. Finally, we recolor each edge in $U = \{(b_i, a_{i+1}) \mid 1 \le i < n\} \cup \{(b_n, p_1)\}$ with the color $\overline{c}_i$. We can now verify that $G_\phi^M$ is indeed a chordal outerplanar graph. Furthermore, it is easy to see $G_\phi^M$ admits a clique tree that is a path. Thus, $G_\phi^M$ is both interval and outerplanar.

We then show these modifications do not contradict Lemma 2. First, observe the distance between $a_i$ and $b_i$ for each $1 \le i \le n$ remains unchanged. However, we introduce additional paths between $a_i$ and $b_i$. But because every edge in $U$ is a bridge and has the color $\overline{c}_i$, it still holds that any rainbow path from $s_1$ to $t$ must, in every $X_i^M$, take precisely either the positive $X_i^M$ path or the negative $X_i^M$ path.

Similarly, we also establish additional paths between $p_j$ and $q_j'$. However, because each edge in $\{(q_j', p_{j+1}) \mid 1 \le j < m\} \cup \{(q_m', t)\}$ is a bridge and has the color $c_j'$, none of the newly added chords can be on a rainbow path from $s_1$ to $t$. Finally, observe also the distance between $p_j$ and $q_j'$ for each $1 \le j \le m$ remains unchanged. This implies any rainbow path from $s_1$ to $t$ must still, in every $C_j^M$, use precisely one of the edges $(r_{j,1}, r_{j,1}')$, $(r_{j,2}, r_{j,2}')$, or $(r_{j,3}, r_{j,3}')$. Thus, Lemma 2 still holds, and we have the theorem. □

Similarly to Theorem 1, given a satisfiable instance of $\phi$, we can observe there is a rainbow shortest path between every pair of vertices. Thus we get the following.

**Corollary 5.** Strong rainbow connectivity *is* NP*-complete when restricted to the class of interval outerplanar graphs.*

## *3.2. Rainbow connectivity is* NP*-complete for interval block graphs*

In this subsection, we prove Rainbow connectivity is NP-complete for interval block graphs, which form a subclass of chordal graphs, and also generalize trees. It is worth noting that unlike in Theorems 1 and 4, the reduction we give next does not show hardness of Strong rainbow connectivity for block graphs.

**Theorem 6.** Rainbow connectivity *is* NP*-complete when restricted to the class of interval block graphs.*

**Proof.** We assume the same terminology as in Theorem 4. Given a 3-Occurrence 3-SAT instance $\phi$, we first build a graph $G_\phi^M$ along with its edge-coloring $\chi_M$ precisely as in Theorem 4. For clarity, we rename $G_\phi^M$ to $G_\phi^B$, and $\chi_M$ to $\chi_B$.

We obtain an $X_i^B$ by adding to $X_i^M$ all the possible chords, that is, the edges $(a_i, \overline{v}_i)$, $(a_i, b_i)$, $(a_i, v_i)$, $(u_i, b_i)$, $(v_i, \overline{u}_i)$, and $(b_i, \overline{u}_i)$. Each of these chords receive the color $\overline{c}_i$. We also add all possible chords to every $C_j^M$, and thus obtain the clause gadget $C_j^B$. Formally, we add to $G_\phi^B$ the edges in

$$\begin{aligned} Z = {} & \{(p_j, r_{j,2}), (p_j, r_{j,3}), (p_j, q_j), (p_j, q_j'), (p_j, r_{j,3}'), (p_j, r_{j,2}'), (p_j, r_{j,1}') \mid 1 \le j \le m\} \\ & \cup \{(r_{j,1}, r_{j,3}), (r_{j,1}, q_j), (r_{j,1}, q_j'), (r_{j,1}, r_{j,3}'), (r_{j,1}, r_{j,2}') \mid 1 \le j \le m\} \\ & \cup \{(r_{j,2}, q_j), (r_{j,2}, q_j'), (r_{j,2}, r_{j,3}'), (r_{j,2}, p_j') \mid 1 \le j \le m\} \\ & \cup \{(r_{j,3}, q_j'), (r_{j,3}, r_{j,1}'), (r_{j,3}, p_j') \mid 1 \le j \le m\} \\ & \cup \{(q_j, r_{j,2}'), (q_j, r_{j,1}'), (q_j, p_j') \mid 1 \le j \le m\} \\ & \cup \{(q_j', r_{j,2}'), (q_j', r_{j,1}'), (q_j', p_j') \mid 1 \le j \le m\} \\ & \cup \{(r_{j,3}', r_{j,1}'), (r_{j,3}', p_j') \mid 1 \le j \le m\} \\ & \cup \{(r_{j,2}', p_j') \mid 1 \le j \le m\}. \end{aligned}$$

Each edge in $Z$ receives the color $c_j'$. This completes the construction of $G_\phi^B$. Clearly, $G_\phi^B$ is now a block graph, with each block being a $K_2$, a $K_6$, or a $K_{10}$. Furthermore, it is easy to see $G_\phi^B$ admits a clique tree that is path. Thus, $G_\phi^B$ is both interval and block.

By an argument similar to Theorem 4, none of the newly added chords can be on a rainbow path from $s_1$ to $t$. Thus, Lemma 2 still holds, and we have the theorem. □

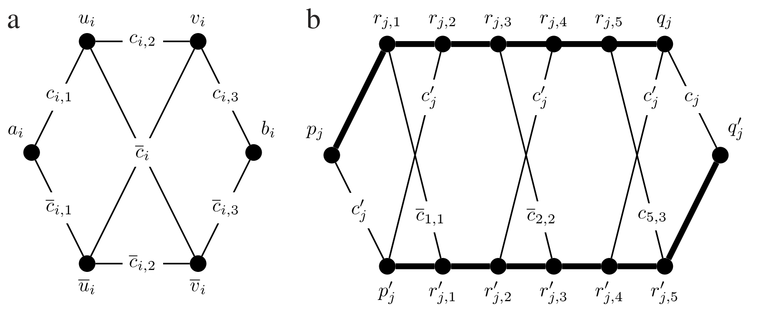


**Fig. 2.** (a) A variable gadget $X_i^{\Delta}$ for the variable $x_i$, and (b) a clause gadget $C_j^{\Delta}$ for the clause $c_j = (x_1 \vee x_2 \vee \neg x_5)$, where $x_1$ is the first literal of $x_1$, $x_2$ is the second literal of $x_2$, and $\neg x_5$ is the third literal of $x_5$.

In the previous construction, the key difference to Theorem 4 is that the distance between any pair of vertices in $X_i^B$ is one, as is the distance between any pair of vertices in $C_j^B$. Therefore, given a positive instance of $\phi$, it is not true that every pair of vertices in $G_\phi^B$ would be connected by a rainbow shortest path.

### *3.3. Rainbow and strong rainbow connectivity are* NP*-complete for k-regular graphs*

In this subsection, we prove both Rainbow connectivity and Strong rainbow connectivity remain NP-complete for $k$-regular graphs, for $k \geq 3$. We begin by proving hardness for cubic graphs, that is, for $k = 3$. We use this construction as a building block for proving hardness for $k$-regular graphs, where $k > 3$.

**Theorem 7.** Rainbow connectivity *is* NP-*complete when restricted to the class of cubic graphs.*

**Proof.** We assume the terminology of Theorem 1. Given a 3-Occurrence 3-SAT instance $\phi$, we first construct a graph $G_\phi^{\Delta}$, and then its edge-coloring $\chi_\Delta$.

We begin very similarly to Theorem 1. A variable gadget $X_i^{\Delta}$ is built for every variable $x_i$, $i \in [n]$, by starting from an $X_i$ and adding two chords $(u_i, \overline{v_i})$ and $(\overline{u}_i, v_i)$. For each clause $c_j, j \in [m]$, we build a clause gadget $C_j^{\Delta}$. A clause gadget $C_j^{\Delta}$ is built by starting from a cycle graph $C_{14}$ embedded in the plane on vertices $p_j, r_{j,1}, r_{j,2}, r_{j,3}, r_{j,4}, r_{j,5}, q_j, q'_j, r'_{j,5}, r'_{j,4}, r'_{j,3}, r'_{j,2}, r'_{j,1}, p'_j$ in clockwise order, and by adding chords $(r_{j,1}, r'_{j,1})$, $(r_{j,3}, r'_{j,3})$, $(r_{j,5}, r'_{j,5})$, $(p'_j, r_{j,2})$, $(r'_{j,2}, r_{j,4})$, and $(r'_{j,4}, q_j)$. The chords $(r_{j,1}, r'_{j,1})$, $(r_{j,3}, r'_{j,3})$, and $(r_{j,5}, r'_{j,5})$ correspond to the three literals each clause has. Both a variable gadget and a clause gadget are shown in Fig. 2.

We then construct a *tail gadget*, which is done by starting with two path graphs on $m-1$ vertices $s_1, \dots, s_{m-1}$ and $s'_1, \dots, s'_{m-1}$, respectively. Then, we add the edges $(s_j, s'_j)$ for each $3 \leq j \leq m-1$, and three edges $(s_1, s'_1)$, $(s'_1, s_2)$, and $(s_1, s'_2)$. Finally, we add a vertex $a_0$, and two edges $(s_{m-1}, a_0)$ and $(s'_{m-1}, a_0)$. The last gadget we build is a *head gadget*. A head gadget is built by starting from a $K_4$ on vertices $t_1$, $t_2$, $t_3$, and $t_4$ with the edge $(t_1, t_2)$ removed. We then add the vertex $t_0$, and finally the edges $(t_0, t_1)$ and $(t_0, t_2)$. Both a tail gadget and a head gadget are shown in Fig. 3.

We connect $X_i^{\Delta}$ with $X_{i+1}^{\Delta}$ by adding an edge $(b_i, a_{i+1})$ for each $1 \leq i < n$. Then, we connect $C_j^{\Delta}$ with $C_{j+1}^{\Delta}$ by adding an edge $(q'_j, p_{j+1})$ for each $1 \leq j < m$. These two components are connected by adding the edge $(b_n, p_1)$. The head gadget is connected to $G_\phi^{\Delta}$ by adding the edge $(t_0, q'_m)$, and the tail gadget by adding the edge $(a_0, a_1)$. This completes the construction of $G_\phi^{\Delta}$. We can now verify that $G_\phi^{\Delta}$ is indeed cubic.

We then describe the edge-coloring $\chi_\Delta$ of $G_\phi^{\Delta}$. The positive $X_i^{\Delta}$ path and the negative $X_i^{\Delta}$ path are colored precisely as in Theorem 1. The two chords $(u_i, \overline{v}_i)$ and $(\overline{u}_i, v_i)$ receive the color $\overline{c}_i$, as does each edge in $U = \{(b_i, a_{i+1}) \mid 1 \leq i < n\} \cup \{(b_n, p_1)\}$. The coloring of a variable gadget $X_i^{\Delta}$ is illustrated in Fig. 2(a).

In a clause gadget $C_j^{\Delta}$, we color the edge $(p_j, p'_j)$ with the color $c'_j$, and the edge $(q_j, q'_j)$ with the color $c_j$. The three chords $(p'_j, r_{j,2})$, $(r'_{j,2}, r_{j,4})$, and $(r'_{j,4}, q_j)$ are colored with the color $c'_j$. For each $k \in \{1, 2, 3\}$, we color the edge $(r_{j,2k-1}, r'_{j,2k-1})$ as follows:

$$\chi_\Delta((r_{j,k}, r'_{j,k})) = \begin{cases} \overline{c}_{i,1} & \text{if } l_{j,k} \text{ is a positive literal and the first literal of } x_i \\ \overline{c}_{i,2} & \text{if } l_{j,k} \text{ is a positive literal and the second literal of } x_i \\ \overline{c}_{i,3} & \text{if } l_{j,k} \text{ is a positive literal and the third literal of } x_i \\ c_{i,1} & \text{if } l_{j,k} \text{ is a negative literal and the first literal of } x_i \\ c_{i,2} & \text{if } l_{j,k} \text{ is a negative literal and the second literal of } x_i \\ c_{i,3} & \text{if } l_{j,k} \text{ is a negative literal and the third literal of } x_i. \end{cases}$$

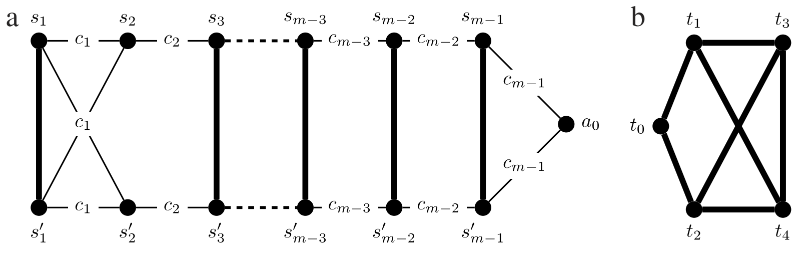


**Fig. 3.** (a) A tail gadget, and (b) a head gadget.

The edge $(q'_j, p_{j+1})$, for each $1 \le j < m$, receives the color $c'_j$, while the edge $(q'_m, t_0)$ is colored with $c'_m$. The coloring of a clause gadget $C^{\Delta}_j$ is shown in Fig. 2(b).

For each $1 \le j < m - 1$, we color the edge $(s_j, s_{j+1})$ with the color $c_j$, and also the edge $(s'_j, s'_{j+1})$ with the color $c_j$. The edges $(s_1, s'_2)$ and $(s_2, s'_1)$ both receive the color $c_1$. The edges $(s_{m-1}, a_0)$ and $(s'_{m-1}, a_0)$ both receive the color $c_{m-1}$. The bridge $(a_0, a_1)$ receives the color $c_m$. The coloring of a tail gadget is shown in Fig. 3(a). Every other uncolored edge of $G^{\Delta}_{\phi}$ receives a fresh new color, that does not appear in $G^{\Delta}_{\phi}$. Formally, these are precisely the edges in

$$\begin{aligned} Q = {} & \{(p_j, r_{j,1}), (r_{j,1}, r_{j,2}), (r_{j,2}, r_{j,3}), (r_{j,3}, r_{j,4}), (r_{j,4}, r_{j,5}), (r_{j,5}, q_j) \mid 1 \le j \le m\} \\ & \cup \{(q'_j, r'_{j,5}), (r'_{j,5}, r'_{j,4}), (r'_{j,4}, r'_{j,3}), (r'_{j,3}, r'_{j,2}), (r'_{j,2}, r'_{j,1}), (r'_{j,1}, p'_j) \mid 1 \le j \le m\} \\ & \cup \{(s_j, s'_j) \mid 3 \le j \le m - 1\} \\ & \cup \{(s_1, s'_1)\} \\ & \cup \{(t_0, t_1), (t_0, t_2), (t_1, t_3), (t_1, t_4), (t_2, t_3), (t_2, t_4), (t_3, t_4)\}. \end{aligned}$$

The edges in $Q$ correspond precisely to the edges drawn with thick lines in Figs. 2 and 3. This completes the edge-coloring $\chi_{\Delta}$ of $G^{\Delta}_{\phi}$.

Let us rename $t_0$ as $t$. By an argument similar to Theorem 4, we can show there is a rainbow path between $s_1$ and $t$ (and similarly between $s'_1$ and $t$) if and only if $\phi$ is satisfiable. $\square$

Again, in the positive case, every pair of vertices has a rainbow shortest path between them further giving us the following.

**Corollary 8.** Strong rainbow connectivity *is* NP-*complete when restricted to the class of cubic graphs.*

We are now ready to prove the hardness of both problems for $k$-regular graphs, where $k > 3$.

**Theorem 9.** Rainbow connectivity *is* NP-*complete when restricted to the class of* $k$*-regular graphs, where* $k > 3$.

**Proof.** We assume the terminology of Theorem 7. Given a 3-Occurrence 3-SAT instance $\phi$, we construct a graph $G^*_{\phi}$, and its edge-coloring $\chi_*$.

We first construct $k - 2$ copies of the cubic graph $G^{\Delta}_{\phi}$. Let us denote these copies as $G^{\Delta}_{\phi,h}$, where $h \in [k - 2]$. Each $G^{\Delta}_{\phi,h}$ retains its original coloring as defined in Theorem 7. That is, each $G^{\Delta}_{\phi,h}$ has precisely the same coloring. Let us assign a labeling $1, \dots, |V(G^{\Delta}_{\phi})|$ on the vertices of $G^{\Delta}_{\phi}$, and use the same labeling for each $G^{\Delta}_{\phi,h}$. By $v_{h,l}$ we denote the vertex in subgraph $G^{\Delta}_{\phi,h}$ with the label $l$, where $l \in [|V(G^{\Delta}_{\phi})|]$. We then form a clique between the vertices $v_{h,l}$ for each $h$ and $l$ by adding all possible $\binom{k-2}{2}$ edges. These newly added edges are precisely the uncolored edges of $G^*_{\phi}$, and all of them receive the fresh new color $c^*$. Because $G^{\Delta}_{\phi}$ is cubic, we can verify $G^*_{\phi}$ is now $k$-regular. This completes the construction of both $G^*_{\phi}$, and its edge-coloring $\chi_*$.

We will then show $G^*_{\phi}$ is rainbow connected if and only if $\phi$ is satisfiable. Recalling the naming of vertices from Theorem 7, without loss let us rename $s_1$ in $G^{\Delta}_{\phi,1}$ as $s$, and $t_0$ in $G^{\Delta}_{\phi,1}$ as $t$. First suppose $\phi$ is satisfiable. Then because there is a rainbow path between $s$ and each vertex of $G^{\Delta}_{\phi,1}$ by Theorem 7, the graph $G^*_{\phi}$ is rainbow connected. Finally, suppose $\phi$ is unsatisfiable. Observe that any rainbow path from $s$ to $t$ must only consist of edges in $G^{\Delta}_{\phi,1}$. But since $s$ and $t$ are not rainbow connected, it follows that $G^*_{\phi}$ is not rainbow connected. Thus, we have the theorem. $\square$

Again, the following is immediate from the previous construction.

**Corollary 10.** Strong rainbow connectivity *is* NP-*complete when restricted to the class of* $k$*-regular graphs, where* $k > 3$.

## 4. Polynomial time solvable cases

In this section, we consider Strong rainbow connectivity from a structural perspective. We observe some graph classes for which the problem is easy. We begin by showing bounding the diameter of the input graph makes Strong rainbow connectivity tractable, while this not so for Rainbow connectivity [23].

**Theorem 11.** Strong rainbow connectivity *is solvable in* $O(n^{d+3})$ *time for graphs of bounded diameter* $d \geq 1$*, where n is the number of vertices in the input graph.*

**Proof.** For $d = 1$, the problem is trivial. So suppose $d \geq 2$, and let $n$ denote the number of vertices in $G$. Let $u$ and $v$ be two arbitrary vertices of $G$, and let $P = ut_1t_2 \cdots t_{d-1}v$ be a shortest path from $u$ to $v$. Because there are less than $n$ choices for each $t_i$ where $i \in [d-1]$, it follows that there are at most $n^d$ shortest $u$–$v$ paths of length no more than $d$. We can then check all of these paths of length exactly $d(u, v)$, and verify if at least one such path is rainbow. Clearly, it takes $O(d)$ time to check one path. Because we have $\binom{n}{2}$ pairs of vertices to check and $d$ is fixed, it follows that Strong rainbow connectivity can be decided in $O(n^{d+3})$ time for graphs of bounded diameter. □

If a graph $G$ has exactly one shortest path between any pair of vertices, $G$ is said to be *geodetic*. A graph is *k-geodetic* if there are at most $k$ shortest paths between any pair of vertices. In fact, it is an easy observation that the brute-force algorithm that checks every shortest path between a pair of vertices runs in polynomial time for $k$-geodetic graphs.

**Theorem 12.** Strong rainbow connectivity *is solvable in polynomial time when restricted to the class of k-geodetic graphs, where* $k = O(\text{poly}(n, m))$*, and n and m are the number of vertices and edges in the input graph, respectively.*

As shown by Stemple and Watkins [21], a connected graph $G$ is geodetic if and only if every block of $G$ is geodetic. By observing that a complete graph is geodetic, we get the following corollary.

**Corollary 13.** Strong rainbow connectivity *is solvable in polynomial time when restricted to the class of block graphs.*

Finally, we make some observations about the reductions built in this work, and describe consequences for parameterized complexity. It follows from the work of Uchizawa et al. [23] that both Rainbow connectivity and Strong rainbow connectivity remain NP-complete when parameterized by *treewidth*. Informally, treewidth is a measure of how close a graph is to being a tree. *Pathwidth* of a graph measures the closeness to a path. Pathwidth of a graph $G$ can be defined to being one less than the maximum clique size in an interval supergraph of $G$. The interval outerplanar graph we construct in Theorem 4 has maximum clique size 3. It follows both Rainbow connectivity and Strong rainbow connectivity are NP-complete for graphs of pathwidth 2. But we can be slightly more general, and show hardness for graphs of pathwidth $p \geq 2$. To see this, observe we can connect a clique of size at least 3 to the graph constructed in Theorem 4, and color its edges with a fresh new color. This might break the property of being outerplanar, but the graph definitely remains interval. Thus, we observe the following.

**Corollary 14.** *Both* Rainbow connectivity *and* Strong rainbow connectivity *are* NP-*complete for graphs of pathwidth p, for every* $p \geq 2$.

By the result of Kaplan and Shamir [12], the *bandwidth* of a graph $G$ is one less than the maximum clique size of any *proper interval* supergraph of $G$, chosen to minimize its clique number. Proper interval graphs are exactly the *claw-free* interval graphs [20], where a claw is the complete bipartite graph $K_{1,3}$. The interval outerplanar graph we construct in Theorem 4 can be observed to be claw-free. Combining this observation with the argument above, we can again be slightly more general.

**Corollary 15.** *Both* Rainbow connectivity *and* Strong rainbow connectivity *are* NP-*complete for graphs of bandwidth b, for every* $b \geq 2$.

Recall a problem is said to be in XP if it can be solved in $O(n^{f(k)})$ time, where $n$ is the input size, $k$ a parameter, and $f$ some computable function. Theorem 11 proves Strong rainbow connectivity is in XP when parameterized by the diameter of the graph. This implies the problem is in XP for several other parameters, such as domination number, independence number, minimum clique cover, distance to clique, distance to cograph, distance to co-cluster, vertex cover number, distance to cluster, and cluster editing (see e.g. [13] for a relationship of some parameters). Corollary 14 extends the known hardness barrier from treewidth to pathwidth. Pathwidth is upper bounded by *tree-depth*, which is informally a measure of how close a graph is to being a star (that is, the $K_{1,n}$). As shown by Nešetřil and Ossona de Mendez [7], the length of a longest path for every undirected graph $G$ is upper bounded by $2^{\text{td}(G)} - 2$, where $\text{td}(G)$ denotes the tree-depth of $G$. Combining this result with Theorem 11, we obtain the following.

**Corollary 16.** *Both* Rainbow connectivity *and* Strong rainbow connectivity *are in* XP *when parameterized by tree-depth.*

## Acknowledgments

The author thanks Henri Hansen, Mikko Lauri, and Keijo Ruohonen for helpful comments, and acknowledges the idea of Henri Hansen that led to Theorem 9. The author also thanks the referees for their useful comments.

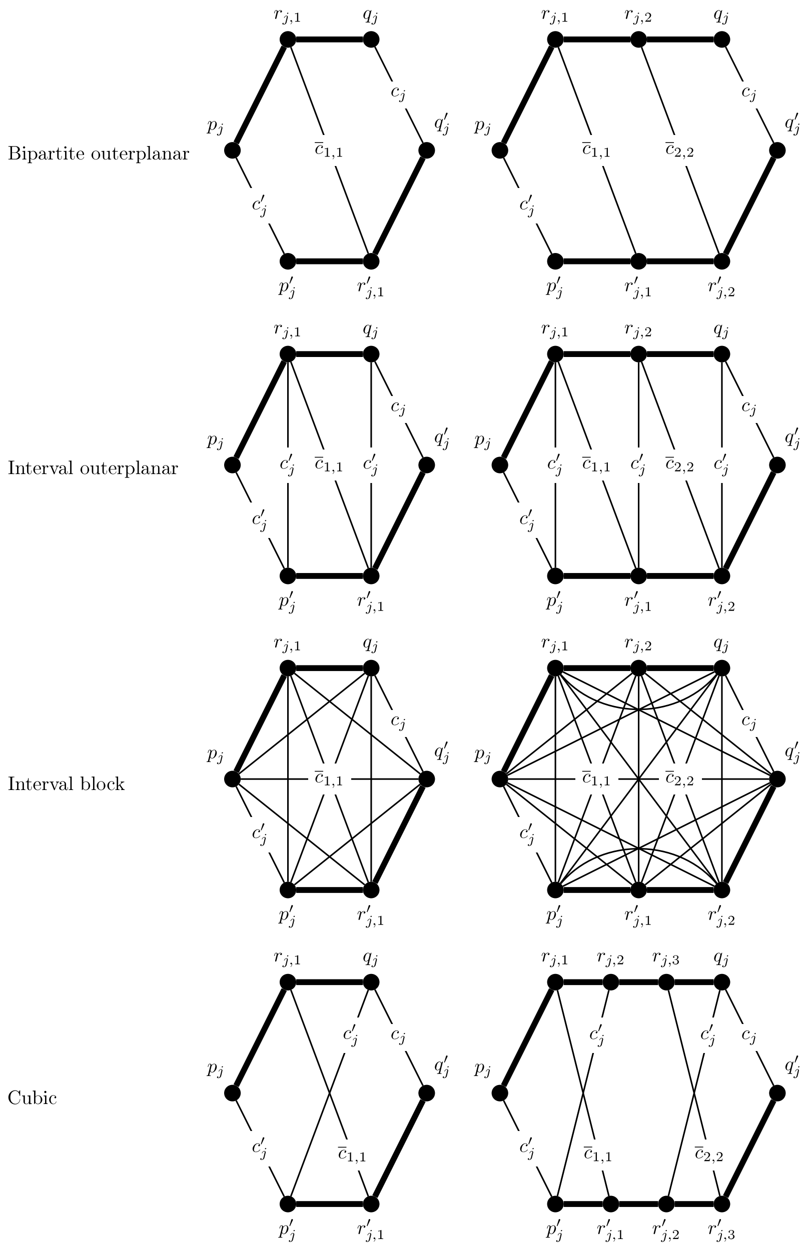


**Fig. 4.** Clause gadgets corresponding to clauses of size one and two for different graph classes.

## Appendix

In Section 2, we presented a reduction from the 3-Occurrence 3-SAT problem to Rainbow connectivity due to Uchizawa et al. [23]. In this appendix, we give the missing critical details of their proof, as discussed in the beginning of the section. Namely, we show how clause gadgets corresponding to clauses of size one and two can be built in Theorem 1. For completeness, we describe similar gadgets for Theorems 4, 6 and 7.

The clause gadgets for four different graph classes are shown in Fig. 4. The first column denotes the graph class. The second column shows a clause gadget corresponding to a clause containing one literal, while the third column does the same for a clause having two literals. For clarity, the edges denoted by thin lines having no labels on row three correspond to chords colored with the color $c'_j$ (refer to Theorem 6 for details). See the respective theorems for an explanation of other colors appearing on the edges.

## References

[1] P. Ananth, M. Nasre, K.K. Sarpatwar, Rainbow connectivity: Hardness and tractability, in: IARCS Annual Conference on Foundations of Software Technology and Theoretical Computer Science, FSTTCS 2011, 2011, pp. 241–251.
[2] S. Chakraborty, E. Fischer, A. Matsliah, R. Yuster, Hardness and algorithms for rainbow connection, J. Comb. Optim. 21 (2009) 330–347.
[3] L.S. Chandran, D. Rajendraprasad, Inapproximability of rainbow colouring, in: IARCS Annual Conference on Foundations of Software Technology and Theoretical Computer Science, FSTTCS 2013, 2013, pp. 153–162.
[4] G. Chartrand, G. Johns, K. McKeon, P. Zhang, Rainbow connection in graphs, Math. Bohem. 133 (2008).
[5] G. Chartrand, P. Zhang, Chromatic Graph Theory, CRC press, 2008.
[6] L. Chen, X. Li, Y. Shi, The complexity of determining the rainbow vertex-connection of a graph, Theoret. Comput. Sci. 412 (2011) 4531–4535.
[7] J. Nešetřil, P. Ossona de Mendez, Grad and classes with bounded expansion I. Decompositions, European J. Combin. 29 (2008) 760–776.
[8] R. Diestel, Graph Theory, Springer-Verlag, Heidelberg, 2005.
[9] F. Gavril, The intersection graphs of subtrees in trees are exactly the chordal graphs, J. Comb. Theory B 16 (1974) 47–56.
[10] P.C. Gilmore, A.J. Hoffman, A characterization of comparability graphs and of interval graphs, Canad. J. Math 16 (1964) 4.
[11] X. Huang, X. Li, Y. Shi, Note on the hardness of rainbow connections for planar and line graphs, Bull. Malays. Math. Sci. Soc. (2014) 1–7.
[12] H. Kaplan, R. Shamir, Pathwidth, bandwidth, and completion problems to proper interval graphs with small cliques, SIAM J. Comput. 25 (1996) 540–561.
[13] C. Komusiewicz, R. Niedermeier, New races in parameterized algorithmics, in: Mathematical Foundations of Computer Science 2012, in: Lecture Notes in Computer Science, vol. 7464, Springer, Berlin, Heidelberg, 2012, pp. 19–30.
[14] M. Krivelevich, R. Yuster, The rainbow connection of a graph is (at most) reciprocal to its minimum degree, J. Graph Theory 63 (2010) 185–191.
[15] S. Li, X. Li, Y. Shi, Note on the complexity of deciding the rainbow (vertex-)connectedness for bipartite graphs, Appl. Math. Comput. 258 (2015) 155–161.
[16] X. Li, Y. Mao, Y. Shi, The strong rainbow vertex-connection of graphs, Util. Math. 93 (2014) 213–223.
[17] X. Li, Y. Shi, Y. Sun, Rainbow connections of graphs: a survey, Graphs Combin. 29 (2012) 1–38.
[18] X. Li, Y. Sun, Rainbow Connections of Graphs, Springer, 2012.
[19] C.H. Papadimitriou, Computational Complexity, Addison-Wesley, 1994.
[20] F.S. Roberts, Indifference graphs, in: Proof Techniques in Graph Theory, Academic Press, New York, 1969, pp. 139–146.
[21] J.G. Stemple, M.E. Watkins, On planar geodetic graphs, J. Combin. Theory 4 (1968) 101–117.
[22] C.A. Tovey, A simplified NP-complete satisfiability problem, Discrete Appl. Math. 8 (1984) 85–89.
[23] K. Uchizawa, T. Aoki, T. Ito, A. Suzuki, X. Zhou, On the rainbow connectivity of graphs: complexity and FPT algorithms, Algorithmica 67 (2013) 161–179.

# Paper 2

Juho Lauri.

**Complexity of rainbow vertex connectivity problems for restricted graph classes.**

# Complexity of Rainbow Vertex Connectivity Problems for Restricted Graph Classes[*]

Juho Lauri[†]

October 10, 2016

**Abstract**

A path in a vertex-colored graph $G$ is *vertex rainbow* if all of its internal vertices have a distinct color. The graph $G$ is said to be *rainbow vertex connected* if there is a vertex rainbow path between every pair of its vertices. Similarly, the graph $G$ is *strongly rainbow vertex connected* if there is a shortest path which is vertex rainbow between every pair of its vertices. We consider the complexity of deciding if a given vertex-colored graph is rainbow or strongly rainbow vertex connected. We call these problems Rainbow Vertex Connectivity and Strong Rainbow Vertex Connectivity, respectively. We prove both problems remain NP-complete on very restricted graph classes including bipartite planar graphs of maximum degree 3, interval graphs, and $k$-regular graphs for $k \geq 3$. We settle precisely the complexity of both problems from the viewpoint of two width parameters: pathwidth and tree-depth. More precisely, we show both problems remain NP-complete for bounded pathwidth graphs, while being fixed-parameter tractable parameterized by tree-depth. Moreover, we show both problems are solvable in polynomial time for block graphs, while Strong Rainbow Vertex Connectivity is tractable for cactus graphs and split graphs.

**Keywords:** rainbow connectivity, computational complexity

## 1 Introduction

Krivelevich and Yuster [1] introduced the concept of rainbow vertex connectivity. A path in a vertex-colored graph $G$ is said to be *vertex rainbow* if all of its internal vertices have a distinct color. The graph $G$ is said to be *rainbow vertex connected* if there is a vertex rainbow path between every pair of its vertices. The minimum number of colors needed to make $G$ rainbow vertex connected is known as the *rainbow vertex connection number*, and it is denoted by $\mathrm{rvc}(G)$. Recall the *diameter* of a graph $G$, denoted by $\mathrm{diam}(G)$, is the length of a longest shortest path in $G$. It is easy to see two vertices $u$ and $v$ are rainbow vertex connected regardless of the underlying vertex-coloring if their distance $d(u, v)$ is at most 2. Thus, we have that $\mathrm{rvc}(G) \geq \mathrm{diam}(G) - 1$, with equality if the diameter is 1 or 2. Similarly, an easy to see upper bound is $\mathrm{rvc}(G) \leq n - 2$, as long as we disregard the singleton graph. In other words, complete graphs are precisely the graphs with rainbow vertex connection number 0; for all other graphs we require at least 1 color.

Li et al. [2] introduced the strong variant of rainbow vertex connectivity. We say the vertex-colored graph $G$ is *strongly rainbow vertex connected* if there is, between every pair of vertices, a shortest path whose internal vertices have a distinct color. The minimum number of colors needed to make $G$ strongly rainbow vertex connected is known as the *strong rainbow vertex connection number*, and it is denoted by $\mathrm{srvc}(G)$. As each strong vertex rainbow coloring is also a rainbow vertex coloring, we have that $\mathrm{diam}(G) - 1 \leq \mathrm{rvc}(G) \leq \mathrm{srvc}(G) \leq n - 2$.

Prior to the work of Krivelevich and Yuster [1], the concept of rainbow connectivity (for edge-colored graphs) was introduced by Chartrand et al. [3] as an interesting way to strengthen the connectivity property. Indeed, the notion has proven to be useful in the domain of networking [4] and anonymous communication [5]. Rainbow coloring and connectivity problems have been subject to considerable

[*]Work partially supported by the Emil Aaltonen Foundation

[†]Tampere University of Technology, Finland. E-mail: `juho.lauri@tut.fi`

interest and research during the past years. For additional applications, we refer the reader to the survey [6]. A comprehensive introduction is also provided by the books [7, 8].

It is computationally difficult to determine either $\text{rvc}(G)$ or $\text{srvc}(G)$ for a given graph $G$. Indeed, through the work of Chen et al. [9] and Chen et al. [10] it is known that deciding if $\text{rvc}(G) \leq k$ is NP-complete for every $k \geq 2$. Likewise, Eiben et al. [11] showed deciding if $\text{srvc}(G) \leq k$ is NP-complete for every $k \geq 3$. In the same paper, the authors also proved that the strong rainbow vertex connection number of an $n$-vertex graph of bounded diameter cannot be approximated within a factor of $n^{1/2-\epsilon}$, for any $\epsilon > 0$, unless P = NP. Given such strong intractability results, it is interesting to ask whether the following problem is easier.

Rainbow Vertex Connectivity (Rvc)
**Instance:** A connected undirected graph $G = (V, E)$, and a vertex-coloring $\psi : V \to C$, where $C$ is a set of colors
**Question:** Is $G$ rainbow vertex connected under $\psi$?

However, Rainbow Vertex Connectivity was shown to be NP-complete by Chen et al. [9]. Later on, Huang et al. [12] showed the problem remains NP-complete even when the input graph is a line graph. A more systematic study into the complexity of Rainbow Vertex Connectivity was performed by Uchizawa et al. [13]. They proved the problem remains NP-complete for both series-parallel graphs, and graphs of bounded diameter. In contrast, they showed the problem is in P for outerplanar graphs. Furthermore, they showed the problem is fixed-parameter tractable for the $n$-vertex $m$-edge general graph parameterized by the number of colors in the vertex-coloring. That is, they gave an algorithm running in time $O(k2^k mn)$ such that given a graph vertex-colored with $k$ colors, it decides whether $G$ is rainbow vertex connected.

We mention two related problems, defined on edge-colored undirected graphs. A path in an edge-colored graph $H$ is *rainbow* if no two edges of it are colored the same. The graph $H$ is said to be *rainbow connected* if there is a rainbow path between every pair of its vertices. Likewise, the graph $H$ is said to be *strongly rainbow connected* if there is a shortest path which is rainbow between every pair of its vertices. Formally, the two problems are defined as follows.

Rainbow Connectivity (Rc)
**Instance:** A connected undirected graph $H = (V, E)$, and an edge-coloring $\zeta : E \to C$, where $C$ is a set of colors
**Question:** Is $H$ rainbow connected under $\zeta$?

Strong Rainbow Connectivity (Src)
**Instance:** A connected undirected graph $H = (V, E)$, and an edge-coloring $\zeta : E \to C$, where $C$ is a set of colors
**Question:** Is $H$ strongly rainbow connected under $\zeta$?

It was shown by Chakraborty et al. [4] that Rainbow Connectivity is NP-complete. Later on, the complexity of both edge variants was studied by Uchizawa et al. [13]. For instance, the authors showed both problems remain NP-complete for outerplanar graphs, and that Rainbow Connectivity is NP-complete already on graphs of diameter 2. A further study into the complexity of the edge variant problems was done in our earlier work [14]. For instance, it was shown that both problems remain NP-complete on interval outerplanar graphs, $k$-regular graphs for $k \geq 3$, and on graphs of bounded pathwidth. In addition, block graphs were identified as a class for which the complexity of the two problems Rainbow Connectivity and Strong Rainbow Connectivity differ. Indeed, it was shown that for block graphs, Rainbow Connectivity is NP-complete, while Strong Rainbow Connectivity is in P.

In this paper, we introduce as a natural variant of Rainbow Vertex Connectivity the following problem.

Strong Rainbow Vertex Connectivity (Srvc)
**Instance:** A connected undirected graph $G = (V, E)$, and a vertex-coloring $\psi : V \to C$, where $C$ is a set of colors
**Question:** Is $G$ strongly rainbow vertex connected under $\psi$?

We present several new complexity results for both Rainbow Vertex Connectivity and Strong Rainbow Vertex Connectivity.

- In Section 3, we focus on negative results. In particular, we prove both problems remain NP-complete for bipartite planar graphs of maximum degree 3 (Subsection 3.2), interval graphs (Subsection 3.3), triangle-free cubic graphs (Subsection 3.4), and $k$-regular graphs for $k \geq 4$ (Subsection 3.5).
- In Section 4, we show both problems are solvable in polynomial time when restricted to the class of block graphs. Furthermore, we extend the algorithm of Uchizawa et al. [13] for deciding Rainbow Vertex Connectivity on cactus graphs to decide Strong Rainbow Vertex Connectivity for the same graph class.
- In Subsection 4.2, we consider the implications of our constructions of Section 3 for parameterized complexity. For instance, we remark both problems remain NP-complete on graphs of pathwidth $p$, where $p \geq 3$, and also on graphs of bandwidth $b$, where $b \geq 3$. For positive results, we show Strong Rainbow Vertex Connectivity is FPT parameterized by the diameter of the input graph, implying polynomial-time solvability for the class of split graphs. Moreover, exploiting known results on tree-depth, we observe all four problems investigated are FPT parameterized by tree-depth.

## 2 Preliminaries

All graphs we consider in this work are simple, undirected, and finite. We begin by defining the graph classes we consider in this work, along with some terminology and graph invariants. For graph-theoretic concepts not defined here, we refer the reader to [15]. For an integer $n$, we write $[n] = \{1, 2, \ldots, n\}$.

A *coloring* of a graph $G$ is an assignment of colors to the vertices of $G$ such that no two adjacent vertices receive the same color. A graph $G$ is said to be *$k$-colorable* if there exists a coloring using $k$ colors for it. A 2-colorable graph is *bipartite.* A *complete graph* on $n$ vertices, denoted by $K_n$, has all the possible $\binom{n}{2}$ edges. In particular, we will call $K_3$ a *triangle.* A *complete bipartite graph* consists of two non-empty independent sets $X$ and $Y$ with $(x, y)$ being an edge whenever $x \in X$ and $y \in Y$. A complete bipartite graph is denoted by $K_{n,m}$, and it has $n + m = |X| + |Y|$ vertices. In particular, we will call $K_{1,3}$ a *claw.* A complete subgraph of $G$ is a *clique.* The *clique number* of a graph $G$, denoted by $\omega(G)$, is the size of a largest clique in $G$.

A graph is said to be *planar* if it can be embedded in the plane with no crossing edges. Equivalently, a graph is planar if it is $(K_{3,3}, K_5)$-minor-free. A graph is *outerplanar* if it has a crossing-free embedding in the plane such that all vertices are on the same face. Clearly, each outerplanar graph is planar. Another superclass of outerplanar graphs is formed by *series-parallel graphs.* Series-parallel graphs are exactly the $K_4$-minor-free graphs [16]. In a *cactus graph*, every edge is in at most one cycle. Cactus graphs form a subclass of outerplanar graphs.

A *chord* is an edge joining two non-consecutive vertices in a cycle. A graph is *chordal* if every cycle of length 4 or more has a chord. Equivalently, a graph is chordal if it contains no induced cycle of length 4 or more. Chordal graphs are precisely the class of graphs admitting a *clique tree* [17]. A clique tree of a connected chordal graph $G$ is any tree $T$ whose vertices are the maximal cliques of $G$ such that for every two maximal cliques $C_i, C_j$, each clique on the path from $C_i$ to $C_j$ in $T$ contains $C_i \cap C_j$. A subclass of chordal graphs is formed by *interval graphs.* A graph is an interval graph if and only if it admits a clique tree that is path [18]. A *cut vertex* is a vertex whose removal will disconnect the graph. A *biconnected graph* is a connected graph with no cut vertices. In a *block graph*, every maximal biconnected component, known as a *block*, is a clique. In other words, every edge of a block graph $G$ lies in a unique block, and $G$ is the union of its blocks. It is easy to see that block graphs are also chordal. An *independent set* in

Table 1: Complexity results for rainbow connectivity problems along with some of our new results marked by ★. The symbol † stands for [13] and the symbol ‡ for [14].

| Graph class | RVC | SRVC | RC | SRC |
|---|---|---|---|---|
| Block | P ★ | P ★ | NPC ‡ | P ‡ |
| Bounded bandwidth | NPC ★ | NPC ★ | NPC ‡ | NPC ‡ |
| Bounded diameter | NPC † | FPT ★ | NPC † | FPT ★ |
| Bounded pathwidth | NPC ★ | NPC ★ | NPC ‡ | NPC ‡ |
| Bounded tree-depth | FPT ★ | FPT ★ | FPT ★ | FPT ★ |
| Cactus | P † | P ★ | P † | P † |
| Interval | NPC ★ | NPC ★ | NPC ‡ | NPC ‡ |
| $k$-regular, $k \geq 3$ | NPC ★ | NPC ★ | NPC ‡ | NPC ‡ |
| Outerplanar | P † | ? | NPC † | NPC † |
| Series-parallel | NPC † | NPC ★ | NPC † | NPC † |
| Split | ? | P ★ | ? | P ★ |
| Tree | P | P | P | P |

a graph is a set of pairwise non-adjacent vertices. A graph whose vertex set can be partitioned into a clique and an independent set is known as a *split graph*. It is easy to see that a split graph is chordal.

The *degree* of a vertex $v$ is the number of edges incident to $v$. A graph is *$k$-regular* if every vertex has degree exactly $k$. In particular, we will call a 3-regular graph *cubic*. A connected 2-regular graph is a *cycle graph*. A cycle graph on $n$ vertices is denoted by $C_n$.

A *proper interval graph* is a graph that is both interval and claw-free (see [19]). The *bandwidth* of a graph $G$, denoted by $\text{bw}(G)$, is one less than the minimum clique number of any proper interval graph having $G$ as a subgraph [20]. The *pathwidth* of a graph $G$, denoted by $\text{pw}(G)$, is one less than the minimum clique number of any interval graph having $G$ as a subgraph. The *treewidth* of a graph $G$, denoted by $\text{tw}(G)$, is one less than the minimum clique number of any chordal graph having $G$ as a subgraph. Indeed, for a graph $G$, we have that $\text{tw}(G) \leq \text{pw}(G) \leq \text{bw}(G)$ (for a proof, see [21]). Finally, a $(C_4, P_4)$-free graph is *trivially perfect*. The *tree-depth* of a graph $G$, denoted by $\text{td}(G)$, is the minimum clique number of any trivially perfect graph having $G$ as a subgraph. Here, we have that $\text{pw}(G) \leq \text{td}(G) - 1$ (for a proof, see [22]).

Finally, we say a problem is *fixed-parameter tractable* (FPT) if it can be solved in time $f(k) \cdot n^{O(1)}$, where $f$ is some computable function depending solely on some parameter $k$, and $n$ is the input size. Similarly, a problem is said to be in XP if it can be solved in $n^{f(k)}$ time. For a more comprehensive treatment on parameterized complexity, we refer the reader to the books [23, 24].

## 3 Hardness results

In this section, we will give a number of hardness results for both Rainbow Vertex Connectivity and Strong Rainbow Vertex Connectivity for very restricted graph classes. It is interesting to compare the obtained complexity results against those of the edge variants, namely Rainbow Connectivity and Strong Rainbow Connectivity. Indeed, we summarize the known complexity results for all four variants in Table 1 along with our new results.

### 3.1 Overview of the reductions

In this subsection, we give an overview of our reductions. Let us remark that all of the four problems considered are in NP with the certificate being a set of colored paths, one path for each pair of vertices. All of our reductions are from the 3-Occurrence 3-SAT problem, which is a variant of the classical 3-SAT problem. In the 3-Occurrence 3-SAT problem, we have a restriction that every variable occurs at most three times, and each clause has at most 3 literals. The problem is known to be NP-complete [25].

All of our reductions are greatly inspired by those of Uchizawa et al. [13], who gave hardness results for both Rainbow Connectivity and Rainbow Vertex Connectivity. Let us explain the gist of their reduction on a high-level. Given a 3-Occurrence 3-SAT formula $\phi$, a variable gadget is constructed

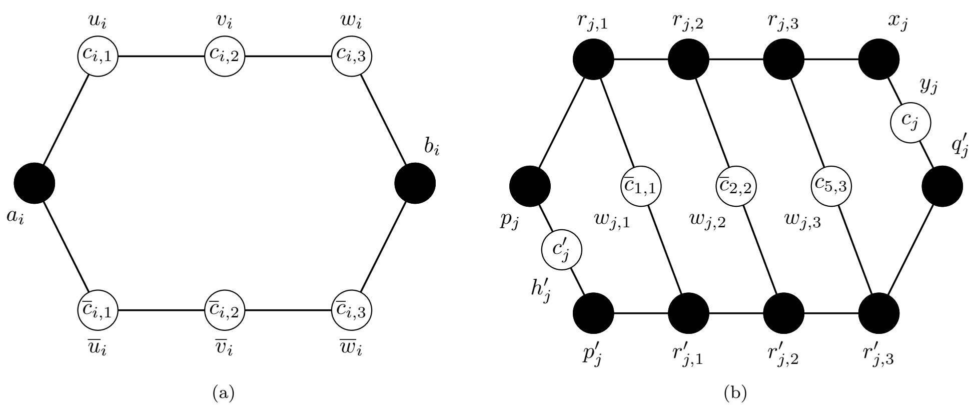


Figure 1: **(a)** A variable gadget $X_i$ for the variable $x_i$, and **(b)** a clause gadget $C_j$ for the clause $c_j = (x_1 \vee x_2 \vee \neg x_5)$, where $x_1$ is the first literal of $x_1$, $x_2$ is the second literal of $x_2$, and $\neg x_5$ is the third literal of $x_5$. The vertices receiving fresh distinct colors are drawn as solid circles.

for each variable, and a clause gadget is built for each clause. Moreover, a certain vertex-coloring is constructed for each gadget. A key idea is that regardless of the satisfiability of $\phi$, every vertex pair in a gadget is rainbow (vertex) connected. Moreover, regardless of the satisfiability of $\phi$, the whole graph will be rainbow (vertex) connected except for a specific vertex pair $s$ and $t$. Informally, the gadgets are set up in a path-like manner, and the special vertices $s$ and $t$ act as endpoints of this path-like graph. The idea is illustrated in Figure 2 (the corresponding construction is given in Theorem 1).

Clearly, a strongly rainbow (vertex) connected graph is also rainbow (vertex) connected. Therefore, it is desirable to construct the gadgets such that each vertex pair is always (regardless of $\phi$) connected by a rainbow (vertex) shortest path. This allows one to obtain a hardness result for the strong problem variant as well. Indeed, the first hardness result we present (Theorem 1) will be of this flavor. However, we are not always able to do this, or doing so will overly complicate the construction in question. We will always explicitly mark whether or not this is the case, i.e., if a hardness result for the strong variant follows as well.

Finally, for the sake of presentation, all of our constructions assume the given 3-Occurrence 3-SAT formula $\phi$ only has clauses with exactly 3 literals. However, as shown by Tovey [26], every such instance is satisfiable. Therefore, we will present clause gadgets corresponding to clauses of size 2 in the appendix for each graph class (note that clauses of size 1 can be safely removed by unit propagation).

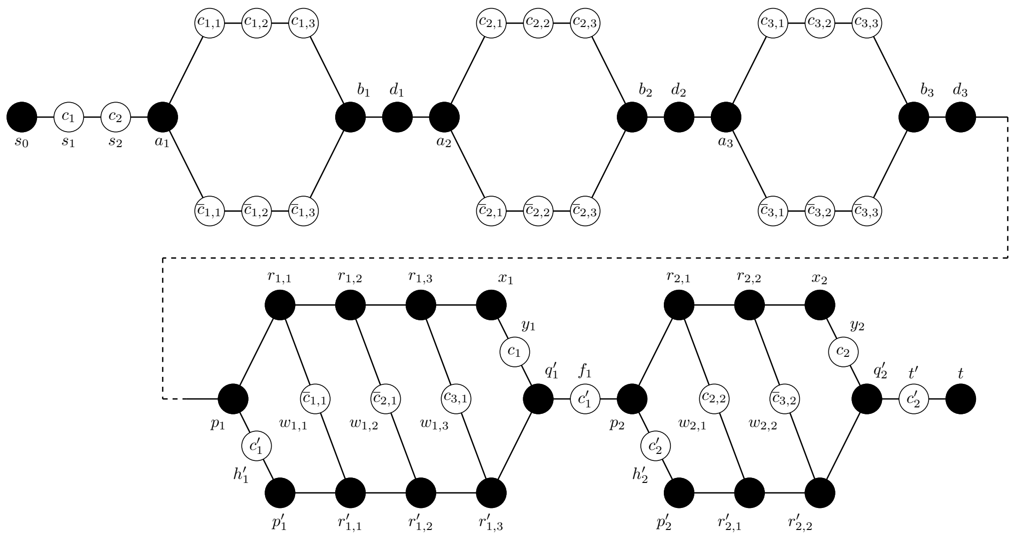
## 3.2 Bipartite planar graphs

In this subsection, we will prove that both Rainbow Vertex Connectivity and Strong Rainbow Connectivity remain NP-complete on bipartite planar graphs of maximum degree 3. We remark that this is a very restricted graph class, generalizing the class of bipartite claw-free graphs. A bipartite claw-free graph consists of disjoint cycles and paths. It is easy to see both Rainbow Vertex Connectivity and Strong Rainbow Connectivity are solvable in polynomial time for the class of bipartite claw-free graphs.

**Theorem 1.** Rainbow Vertex Connectivity *is* NP*-complete when restricted to the class of bipartite planar graphs of maximum degree 3.*

**Construction**: Given a 3-Occurrence 3-SAT formula $\phi = \bigwedge_{j=1}^{m} c_i$ over variables $x_1, x_2, \ldots, x_n$, we construct a graph $G_\phi$ and a vertex-coloring $\psi$ such that $\phi$ is satisfiable if and only if $G_\phi$ is rainbow vertex connected under $\psi$. We first describe the construction of $G_\phi$, and then the vertex-coloring $\psi$ of $G_\phi$.

We will construct for each variable $x_i$, where $i \in [n]$, a *variable gagdet* $X_i$. A variable gadget $X_i$ is the cycle graph $C_8$ embedded in the plane on the vertices $a_i$, $u_i$, $v_i$, $w_i$, $b_i$, $\overline{w}_i$, $\overline{v}_i$, $\overline{u}_i$ in clockwise order. For each clause $c_j$, where $j \in [m]$, we construct a *clause gadget* $C_j$. A clause gadget $C_j$ is built



$c_{1,1}$ $c_{1,2}$ $c_{1,3}$ $c_{2,1}$ $c_{2,2}$ $c_{2,3}$ $c_{3,1}$ $c_{3,2}$ $c_{3,3}$
$b_1$ $d_1$ $b_2$ $d_2$ $b_3$ $d_3$
$c_1$ $c_2$
$s_0$ $s_1$ $s_2$ $a_1$ $a_2$ $a_3$
$\overline{c}_{1,1}$ $\overline{c}_{1,2}$ $\overline{c}_{1,3}$ $\overline{c}_{2,1}$ $\overline{c}_{2,2}$ $\overline{c}_{2,3}$ $\overline{c}_{3,1}$ $\overline{c}_{3,2}$ $\overline{c}_{3,3}$
$r_{1,1}$ $r_{1,2}$ $r_{1,3}$ $x_1$ $r_{2,1}$ $r_{2,2}$ $x_2$
$y_1$ $c_1$ $y_2$ $c_2$
$q'_1$ $f_1$ $c'_1$ $q'_2$ $t'$ $c'_2$ $t$
$\overline{c}_{1,1}$ $\overline{c}_{2,1}$ $\overline{c}_{3,1}$ $\overline{c}_{2,2}$ $\overline{c}_{3,2}$
$p_1$ $w_{1,1}$ $w_{1,2}$ $w_{1,3}$ $p_2$ $w_{2,1}$ $w_{2,2}$
$c'_1$ $h'_1$ $c'_2$ $h'_2$
$p'_1$ $r'_{1,1}$ $r'_{1,2}$ $r'_{1,3}$ $p'_2$ $r'_{2,1}$ $r'_{2,2}$

Figure 2: A planar bipartite graph $G_\phi$ of maximum degree 3 constructed for the formula $\phi = (x_1 \vee x_2 \vee \neg x_3) \wedge (\neg x_2 \vee x_3)$. For brevity, some vertex labels are not shown.

by starting from the cycle graph $C_{12}$ embedded in the plane on the vertices $p_j$, $r_{j,1}$, $r_{j,2}$, $r_{j,3}$, $x_j$, $y_j$, $q'_j$, $r'_{j,3}$, $r'_{j,2}$, $r'_{j,1}$, $p'_j$, and $h'_j$ in clockwise order, and by adding chords $(r_{j,1}, r'_{j,1})$, $(r_{j,2}, r'_{j,2})$, and $(r_{j,3}, r'_{j,3})$. For $\ell \in [3]$, the added chord $(r_{j,\ell}, r'_{j,\ell})$ is subdivided by a new vertex $w_{j,\ell}$. The vertices $w_{j,\ell}$ correspond to the three literals the clause $c_j$ has. Both a variable gadget and a clause gadget are shown in Figure 1.

For each $1 \le i < n$, we connect $X_i$ with $X_{i+1}$ by adding a new vertex $d_i$ along with two edges $(b_i, d_i)$ and $(d_i, a_{i+1})$. Similarly, we connect $C_j$ with $C_{j+1}$ by adding a new vertex $f_j$ along with two edges $(q'_j, f_j)$ and $(f_j, p_{j+1})$ for each $1 \le j < m$. The two components are connected together by adding the vertex $d_n$ with the edges $(b_n, d_n)$ and $(d_n, p_1)$. We then add two vertices $t'$ and $t$ along with the edges $(q'_m, t')$ and $(t', t)$. Finally, we construct a path of length $m + 1$ on vertices $s_0, s_1, \ldots, s_m$, and connect it with $G_\phi$ by adding the edge $(s_m, a_1)$. This completes the construction of $G_\phi$. We can verify $G_\phi$ is indeed a bipartite planar graph of maximum degree 3.

We then describe the vertex-coloring $\psi$ given to the vertices of $G_\phi$. Observe that in a variable gadget $X_i$, there are precisely two paths between $a_i$ and $b_i$. Intuitively, taking the path from $a_i$ to $b_i$ through $u_i$, $v_i$, and $w_i$ corresponds to setting $x_i = 1$ in the formula $\phi$; we refer to this path as the *positive $X_i$ path.* We color the three vertices $u_i$, $v_i$, and $w_i$ with colors $c_{i,1}, c_{i,2}$, and $c_{i,3}$, respectively. Taking the path from $a_i$ to $b_i$ through $\overline{u}_i$, $\overline{v}_i$, and $\overline{w}_i$ corresponds to setting $x_i = 0$ in the formula $\phi$; we refer to this path as the *negative $X_i$ path.* The three vertices $\overline{u}_i$, $\overline{v}_i$, and $\overline{w}_i$ receive colors $\overline{c}_{i,1}, \overline{c}_{i,2}$ and $\overline{c}_{i,3}$, respectively. The coloring of a variable gadget $X_i$ is illustrated in Figure 1 (a).

Recall that a variable $x_i$ appears at most three times in $\phi$. We refer to the first occurrence of $x_i$ as the *first literal of $x_i$*, the second occurrence of $x_i$ as the *second literal of $x_i$*, and finally the third occurrence of $x_i$ as the *third literal of $x_i$*. If a clause has two or three literals of a same variable, the tie is broken arbitrarily. In a clause gadget $C_j$, we color vertex $h'_j$ with color $c'_j$, and vertex $y_j$ with color $c_j$. For each $k \in [3]$, we denote the $k$th literal in the $j$th clause by $l_{j,k}$. We color vertex $w_{j,\ell}$ as follows:

$$\psi(w_{j,\ell}) = \begin{cases} \overline{c}_{i,1} & \text{if } l_{j,k} \text{ is a positive literal and the first literal of } x_i \\ \overline{c}_{i,2} & \text{if } l_{j,k} \text{ is a positive literal and the second literal of } x_i \\ \overline{c}_{i,3} & \text{if } l_{j,k} \text{ is a positive literal and the third literal of } x_i \\ c_{i,1} & \text{if } l_{j,k} \text{ is a negative literal and the first literal of } x_i \\ c_{i,2} & \text{if } l_{j,k} \text{ is a negative literal and the second literal of } x_i \\ c_{i,3} & \text{if } l_{j,k} \text{ is a negative literal and the third literal of } x_i \end{cases}$$

The vertex $f_j$, for each $1 \leq j < m$, receives color $c'_j$, while vertex $t'$ is colored with $c'_m$. The coloring of a clause gadget $C_j$ is shown in Figure 1 (b).

Finally, for each $1 \leq j \leq m$, we color vertex $s_j$ with color $c_j$. Every other uncolored vertex of $G_\phi$ receives a fresh new color that does not appear in $G_\phi$. Formally, these are precisely the vertices in

$$\begin{aligned} U = &\{a_i, b_i, d_i \mid 1 \leq i \leq n\} \\ &\cup \{p_j, r_{j,1}, r_{j,2}, r_{j,3}, x_j, q'_j, r'_{j,3}, r'_{j,2}, r'_{j,1}, p'_j \mid 1 \leq j \leq m\} \\ &\cup \{s_0, t\}. \end{aligned}$$

Vertices in $U$ shown in Figure 1 are drawn as solid circles. This completes the vertex-coloring $\psi$ of $G_\phi$. An example is shown in Figure 2. The proof of Theorem 1 is obtained via the following two lemmas, which also make precise the intuition provided in Section 3.1. The arguments essentially follow from [13], but we describe them for completeness. The reader should observe the two following lemmas prove a slightly stronger statement than necessary, by talking about *strong* rainbow vertex connectedness instead of rainbow vertex connectedness.

**Lemma 2.** *The graph $G_\phi$ is strongly rainbow vertex connected under the vertex-coloring $\psi$ if and only if $G_\phi$ has a vertex rainbow shortest path between the vertices $s_0$ and $t$.*

*Proof.* Trivially, it suffices show that if $s_0$ and $t$ are strongly rainbow vertex connected, then $G_\phi$ is strongly rainbow vertex connected. For convenience, we partition the vertex set $V$ into three groups. Indeed, let $V = S \cup A \cup L$, where $S = \{s_1, \ldots, s_m\}$, $A = \bigcup_{i=1}^{n} V(X_i)$, and $L = \bigcup_{j=1}^{m} V(C_j)$. Let $u$ and $v$ be two distinct vertices in $V$, and we will show they are strongly rainbow vertex connected. It is straightforward to verify $u$ and $v$ are strongly rainbow vertex connected when they are in the same group. So let us consider the three possible cases of $u$ and $v$ being in distinct groups.

- **Case 1:** $u \in S$ and $v \in A$ are strongly rainbow vertex connected.

  *Proof.* No two vertices in $S$ and $A$ share colors, so the claim follows. ■

- **Case 2:** $u \in S$ and $v \in L$ are strongly rainbow vertex connected.

  *Proof.* By our assumption, there is a rainbow shortest path $P$ from $s_0$ and $t$. Observe that $P$ must use every color $c_1, \ldots, c_m$, and also every color $c'_1, \ldots, c'_m$. Therefore, it must be the case that $P$ uses the vertex $w_{j,\ell}$ for some $\ell \in [3]$ for every $j \in [m]$. So suppose the vertex $v$ is contained in a clause gadget $C_j$. By the above reasoning, it is clear that $p_j$ is reachable from $u \in S$ by a rainbow shortest path $P'$, which is a subpath of $P$. Finally, we can construct a shortest path $P''$ from $p_j$ to $v$ such that $y_j$ is not an internal vertex of $P''$. The concatenation of $P'$ and $P''$ gives us a rainbow shortest path between $u$ and $v$, so the claim follows. ■

- **Case 3:** $u \in A$ and $v \in L$ are strongly rainbow vertex connected.

  *Proof.* Observe that we can always choose a shortest $u$-$v$ path $P$ so that none of the vertices $w_{j,\ell}$ appear as an internal vertex in $P$, for any $j \in [m]$ and $\ell \in [3]$. Thus, the claim follows. ■

This completes the proof. □

**Lemma 3.** *There is a vertex rainbow shortest path between $s_0$ and $t$ if and only if the formula $\phi$ is satisfiable.*

*Proof.* Suppose there is a vertex rainbow shortest path $P$ between $s_0$ and $t$, and we will show the formula $\phi$ is satisfiable. It is clear that $P$ must choose from every variable gadget $X_i$ either the positive or the negative $X_i$ path. Indeed, let us construct a truth assignment $\boldsymbol{\alpha} = (\alpha_1, \ldots, \alpha_n)$ for $\phi$ as follows. For every $X_i$, if $P$ is using the positive $X_i$ path, we set $\alpha_i = 1$. Otherwise, $P$ is using the negative $X_i$ path and we let $\alpha_i = 0$. We will then argue $\boldsymbol{\alpha}$ is a satisfying assignment for $\phi$. Consider a clause gadget $C_j$, where $j \in [m]$. It is easy to verify the vertex rainbow shortest path $P$ must use exactly one of the vertices $w_{j,\ell}$, where $\ell \in [3]$, in every $C_j$. Indeed, if two or more of the vertices $w_{j,\ell}$ were chosen, the path $P$ would not be a shortest path. So consider the vertex $w_{j,\ell}$ chosen by $P$ in some clause gadget $C_j$.

Furthermore, suppose $w_{j,\ell}$ has received color $c_{i,\delta}$, for some $i \in [n]$ and $\delta \in [3]$ (recall a variable occurs at most three times in $\phi$). By construction, the literal $l_{j,k}$ corresponding to $w_{j,\ell}$ is a negative literal of the variable $x_i$. Moreover, color $c_{i,\delta}$ also appears on the positive $X_i$ path. Because $P$ contains $w_{j,\ell}$ colored $c_{i,\delta}$, it follows $P$ chooses the $X_i$ negative path. Thus, we have $\alpha_i = 0$, and the literal $l_{j,k}$ is set true by $\boldsymbol{\alpha}$. The proof is symmetric for the case $w_{j,\ell}$ having color $\overline{c}_{i,\delta}$.

For the other direction, suppose $\phi$ is satisfiable under the assignment $\boldsymbol{\alpha} = (\alpha_1, \ldots, \alpha_n)$. We construct a vertex rainbow shortest path $P$ between $s_0$ and $t$ as the concatenation of two paths $P_V$ and $P_C$. To construct $P_V$, we proceed as follows. For each variable gadget $X_i$, if $\alpha_i = 1$ we choose the positive $X_i$ path; otherwise $\alpha_i = 0$ and we choose the $X_i$ negative path. Clearly, $P_V$ is a vertex rainbow shortest path from $s_0$ to $b_n$. We will then show that for every clause gadget $C_j$, there is a vertex $w_{j,\ell}$ such that its color does not appear on $P_V$. It will then be straightforward to construct the path $P_C$. Because $\boldsymbol{\alpha}$ is a satisfying assignment for $\phi$, each clause has a literal which is made true by $\boldsymbol{\alpha}$. Let $l_{j,k}$ be such a literal for a clause gadget $C_j$. Suppose $l_{j,k}$ is a positive literal of the variable $x_i$, for some $i \in [n]$. By construction, the vertex $w_{j,l}$ has received color $\overline{c}_{i,\delta}$, where $\delta \in [3]$. Because $l_{j,k}$ is a positive literal of $x_i$ and $l_{j,k}$ is made true by $\boldsymbol{\alpha}$, we have that $\alpha_i = 1$. Moreover, the path $P_V$ has taken the positive $X_i$ path, meaning it is using color $c_{i,1}$, $c_{i,2}$, and $c_{i,3}$. In other words, color $\overline{c}_{i,\delta}$ does not appear in $P_V$. Thus, the concatenation of $P_V$ and $P_C$ indeed gives us a vertex rainbow shortest path between $s_0$ and $t$. This completes the proof. □

For proving Theorem 1, the two above lemmas are slightly stronger than necessary. That is, given a positive instance of $\phi$, every pair of vertices in $G_\phi$ is not only rainbow vertex connected, but *strongly* rainbow vertex connected. In other words, we have also proven the following.

**Theorem 4.** Strong Rainbow Vertex Connectivity *is* NP*-complete when restricted to the class of bipartite planar graphs of maximum degree 3.*

## 3.3 Interval graphs

In this subsection, we investigate the complexity of both Rainbow Vertex Connectivity and Strong Rainbow Vertex Connectivity on chordal graphs. We will show that both problems remain NP-complete on interval graphs, which form a well-known subclass of chordal graphs. In fact, we will prove a stronger result for Strong Rainbow Vertex Connectivity by showing it remains NP-complete for proper interval graphs. A *caterpillar* is a tree that has a dominating path. One can observe caterpillars form a subclass of interval graphs. Moreover, both problems are solvable in polynomial time on caterpillars.

**Theorem 5.** Rainbow Vertex Connectivity *is* NP*-complete when restricted to the class of interval graphs.*

*Proof.* We assume the terminology of Theorem 1. Given a 3-Occurrence 3-SAT instance $\phi = \bigwedge_{j=1}^{m} c_i$ over variables $x_1, x_2, \ldots, x_n$, we follow a strategy similar to Theorem 1. We will first describe how variable and clause gadgets of a graph $G_\phi^I$ are built along with their vertex-colorings.

A variable gadget $X_i^I$ is built by starting from the cycle graph $C_{20}$ on the vertices $v_{i,\ell}$ in clockwise order, where $\ell \in [20]$. For convenience (and to match Theorem 1), we rename $v_{i,1}$ to $a_i$ and $v_{i,11}$ to $b_i$. We will then describe the altogether 19 chords added to $X_i^I$. First, we add the chords $(v_{i,2}, v_{i,19})$, $(v_{i,2}, v_{i,20})$, $(v_{i,3}, v_{i,18})$, $(v_{i,3}, v_{i,19})$, $(v_{i,3}, v_{i,20})$, $(v_{i,4}, v_{i,18})$, and $(v_{i,4}, v_{i,19})$. Then, we add the chords $(v_{i,5}, v_{i,18})$, $(v_{i,6}, v_{i,16})$, $(v_{i,6}, v_{i,17})$, $(v_{i,6}, v_{i,18})$, $(v_{i,7}, v_{i,16})$, $(v_{i,8}, v_{i,14})$, $(v_{i,8}, v_{i,15})$, $(v_{i,8}, v_{i,16})$, $(v_{i,9}, v_{i,14})$, $(v_{i,10}, v_{i,14})$, $(v_{i,10}, v_{i,13})$, and $(v_{i,10}, v_{i,12})$. This completes the construction of a variable gadget $X_i^I$. A variable gadget $X_i^I$ is shown in Figure 3. It is straightforward to verify $X_i^I$ admits a clique tree that is a path, and thus $X_i^I$ is an interval graph.

We will then describe the vertex-coloring of $X_i^I$. For each $X_i^I$, we introduce 6 new colors $c_{i,a}$, $c_{i,b}$, $c_{i,c}$, $c_{i,d}$, $c_{i,e}$, and $c_{i,f}$. We color both vertices $v_{i,2}$ and $v_{i,16}$ with color $c_{i,a}$, both $v_{i,3}$ and $v_{i,14}$ with color $c_{i,b}$, both $v_{i,4}$ and $v_{i,12}$ with color $c_{i,c}$, both $v_{i,20}$ and $v_{i,6}$ with color $c_{i,d}$, both $v_{i,19}$ and $v_{i,8}$ with color $c_{i,e}$, and both $v_{i,18}$ and $v_{i,10}$ with color $c_{i,f}$. The vertices $v_{i,5}$, $v_{i,7}$, and $v_{i,9}$ receive colors $c_{i,1}$, $c_{i,2}$, and $c_{i,3}$, respectively. Similarly, the vertices $v_{i,17}$, $v_{i,15}$, and $v_{i,13}$ receive colors $\overline{c}_{i,1}$, $\overline{c}_{i,2}$, and $\overline{c}_{i,3}$, respectively. Conceptually, these two sets of three vertices correspond to the positive and the negative $X_i$ path of Theorem 1. The vertex-coloring of a variable gadget $X_i^I$ is shown in Figure 3.

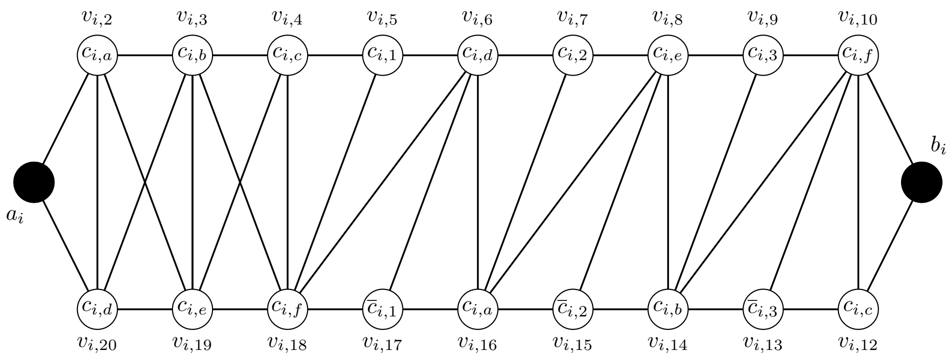


Figure 3: A variable gadget $X_i^I$.

A clause gadget $C_j^I$ is built by starting from a clause gadget $C_j$, and by adding the altogether 15 chords $(r_{j,1}, h'_j)$, $(r_{j,1}, p'_j)$, $(r_{j,2}, p'_j)$, $(r_{j,2}, w_{j,1})$, $(r_{j,2}, r'_{j,1})$, $(r_{j,3}, r'_{j,1})$, $(r_{j,3}, w_{j,2})$, $(r_{j,3}, r'_{j,2})$, $(x_j, r'_{j,2})$, $(x_j, w_{j,3})$, $(x_j, r'_{j,3})$, $(y_j, r'_{j,3})$, $(p'_j, w_{j,1})$, $(r'_{j,1}, w_{j,2})$, and $(r'_{j,2}, w_{j,3})$. This completes the construction of a clause gadget $C_j^I$. A clause gadget $C_j^I$ is shown in Figure 4. It can be verified $C_j^I$ admits a clique tree that is a path, and thus $C_j^I$ is an interval graph.

We will then describe the vertex-coloring of $C_j^I$. We color vertices $h'_j$ and $y_j$, and the three vertices $w_{j,\ell}$, for $\ell \in [3]$, exactly as in Theorem 1. Moreover, for each $C_j^I$, we introduce four new colors $c_{j,u}$, $c_{j,v}$, $c_{j,w}$, and $c_{j,z}$. We color both vertices $r_{j,1}$ and $p'_j$ with color $c_{j,u}$, both $r_{j,2}$ and $r'_{j,1}$ with color $c_{j,v}$, both $r_{j,3}$ and $r'_{j,2}$ with color $c_{j,w}$, and both $x_j$ and $r'_{j,3}$ with color $c_{j,z}$. The vertex-coloring of a clause gadget $C_j^I$ is shown in Figure 4.

The variable and clause gadgets are joined together precisely as in Theorem 1. Furthermore, we also add vertices $s, s_1, \ldots, s_m$, $t'$, and $t$ exactly as in Theorem 1. The remaining uncolored vertices receive a fresh new color that does not appear in $G_\phi^I$. Formally, these are precisely the vertices in

$$\begin{aligned} U = &\{a_i, b_i, d_i \mid 1 \le i \le n\} \\ &\cup \{p_j, q'_j \mid 1 \le j \le m\} \\ &\cup \{s_0, t\}. \end{aligned}$$

Informally, disregarding the vertex-colorings, the graph $G_\phi^I$ differs from the graph $G_\phi$ of Theorem 1 only in the way in which the gadgets are built.

We will then show these modifications do not contradict Lemma 3. Since we only modified the variable and clause gadgets, it suffices to inspect them. Consider a variable gadget $X_i^I$. We claim that any vertex rainbow path $R$ from $a_i$ to $b_i$ must still pass through all the vertices in either $P = \{v_{i,5}, v_{i,7}, v_{i,9}\}$ or $N = \{v_{i,17}, v_{i,15}, v_{i,13}\}$. Observe that the path $R$ must choose at least one vertex from each set of two vertices at a distance 1, 2, and 3 from $a_i$. Similarly, by the way in which the vertices are colored, the path $R$ must choose exactly one vertex from the set of two vertices at a distance 5, 7, and 9 from $a_i$. Thus, $R$ cannot choose more than three vertices from $\{v_{i,\ell}, v_{i,16+\ell} \mid 2 \le \ell \le 4\}$. It is then straightforward to verify that $R$ must pass through all the vertices in either $P$ or $N$. In other words, there are exactly two choices how the path $R$ can traverse from $a_i$ to $b_i$.

Finally, consider a clause gadget $C_j^I$. The addition of chords establishes additional paths between $p_j$ and $q'_j$. However, as each vertex in $\{f_j \mid 1 \le j < m\} \cup \{t'\}$ is a cut vertex colored with color $c'_j$, no vertex rainbow path $R'$ from $s_0$ to $t$ can use vertex $h'_j$. It can be verified that $R'$ must still, in every $C_j^I$, use at least one of the vertices $w_{j,1}$, $w_{j,2}$, or $w_{j,3}$. Thus, Lemma 3 still holds, and we have the theorem. □

We will then prove a stronger result for Strong Rainbow Vertex Connectivity.

**Theorem 6.** Strong Rainbow Vertex Connectivity *is* NP*-complete when restricted to the class of proper interval graphs.*

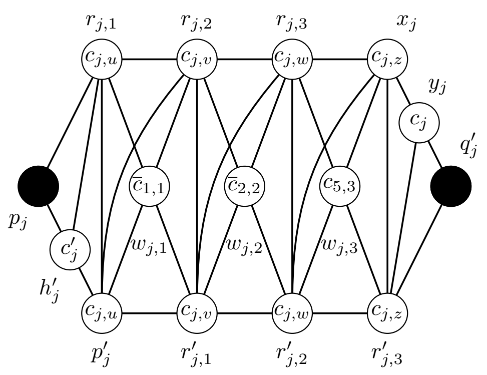


Figure 4: A clause gadget $C_j^I$.

*Proof.* We assume the terminology of Theorem 5. Given a 3-Occurrence 3-SAT instance $\phi$, we construct the graph $G_\phi^I$ exactly as in Theorem 5; we will only slightly change the variable gadgets $X_i^I$ to prove our claim. Indeed, we delete the chords $(v_{i,6+2k}, v_{i,18-2k})$ and add the chords $(v_{i,5+2k}, v_{i,17-2k})$, where $0 \leq k \leq 2$.

First, observe that this modification does not break the property of $G_\phi^I$ being interval. Furthermore, we can now verify $G_\phi^I$ is also claw-free. Then, consider a vertex rainbow shortest path $R$ from $a_i$ to $b_i$ after the deletion and addition of new chords. The distance $d(a_i, b_i)$ is now 10, so $R$ cannot use any of the newly added chords $(v_{i,5+2k}, v_{i,17-2k})$, where $0 \leq k \leq 2$. By an argument similar to that of Theorem 5, any $R$ must use either exactly all vertices in $N$, or all vertices $P$. Finally, any $R$ must choose from every $C_j^I$ exactly one of the vertices $w_{j,1}$, $w_{j,2}$, or $w_{j,3}$. Thus, the theorem follows. $\square$

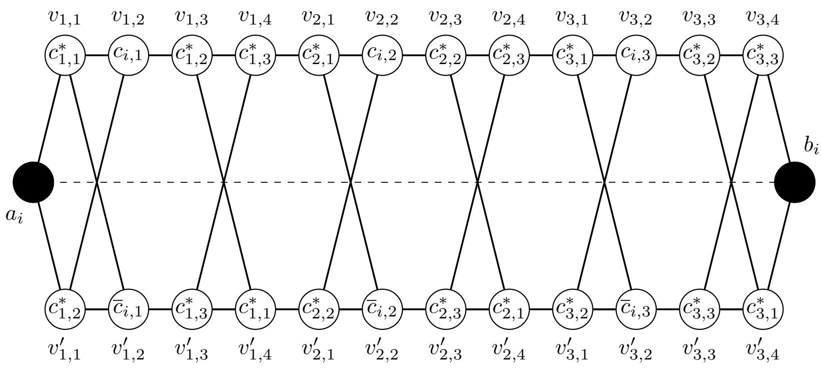
It is worth observing the modification of the variable gadget in the above theorem does not extend for Rainbow Vertex Connectivity (Theorem 5). Indeed, if the path $R$ from $a_i$ to $b_i$ is not required to be a shortest path, it is possible to construct $R$ such that a particular color from $\{c_{i,\ell}, \overline{c}_{i,\ell} \mid 1 \leq \ell \leq 3\}$ is avoided, possibly breaking Lemma 3.

## 3.4 Cubic graphs

In this subsection, we turn our attention to regular graphs. It is easy to see that both Rainbow Vertex Connectivity and Strong Rainbow Vertex Connectivity are solvable in polynomial time on 2-regular graphs. Therefore, we will consider 3-regular graphs, i.e., cubic graphs. In contrast to previous constructions, we will need additional gadgets. Strictly speaking, the gadgets we introduce in the following are not cubic. However, when the gadgets are connected together, the resulting graph will be cubic.

Indeed, before proceeding, we will describe a parametric gadget that will serve different purposes in a construction to follow. This parametric gadget $T_k$, where $k \geq 1$, is a cycle graph of length $8k+2$. We choose two vertices $v_s$ and $v_t$ such that $d(v_s, v_t) = 4k+1$. The two $v_s$-$v_t$ paths of length $4k+1$ are broken down into $4k$ vertices $v_{i,\ell}$ and $v'_{i,\ell}$, respectively, where $i \in [k]$ and $\ell \in [4]$. The construction is finished by adding the chords $(v_{k,1}, v'_{k,2})$, $(v_{k,2}, v'_{k,1})$, $(v_{k,3}, v'_{k,4})$, and $(v_{k,4}, v'_{k,3})$, for each $k$. An example of a $T_k$ for $k = 3$ is shown in Figure 5. For each $k$, we introduce a set of three "blocking" colors $\{c^*_{k,1}, c^*_{k,2}, c^*_{k,3}\}$ and color the vertices as follows: both vertices $v_{k,1}$ and $v'_{k,4}$ receive color $c^*_{k,1}$, both vertices $v_{k,3}$ and $v'_{k,1}$ receive color $c^*_{k,2}$, and both vertices $v_{k,4}$ and $v'_{k,3}$ receive color $c^*_{k,3}$. Both $v_s$ and $v_t$ receive a fresh new color that does not appear elsewhere. Exactly $2k$ vertices are now left uncolored: depending on the situation, we will color these vertices differently. However, we can still argue the following about a vertex rainbow path traversing $T_k$.

**Lemma 7.** *Let $R$ be a vertex rainbow path from $v_s$ to $v_t$ in a parametric gadget $T_k$, where $k \geq 1$. There are no $v_{\ell,i}$ and $v'_{\ell',j}$ in $R$ with $1 \leq i \leq j \leq 4$ and $\ell, \ell' \in [k]$.*



$v_{1,1}$ $v_{1,2}$ $v_{1,3}$ $v_{1,4}$ $v_{2,1}$ $v_{2,2}$ $v_{2,3}$ $v_{2,4}$ $v_{3,1}$ $v_{3,2}$ $v_{3,3}$ $v_{3,4}$
$c^*_{1,1}$ $c_{i,1}$ $c^*_{1,2}$ $c^*_{1,3}$ $c^*_{2,1}$ $c_{i,2}$ $c^*_{2,2}$ $c^*_{2,3}$ $c^*_{3,1}$ $c_{i,3}$ $c^*_{3,2}$ $c^*_{3,3}$
$b_i$
$a_i$
$c^*_{1,2}$ $\bar{c}_{i,1}$ $c^*_{1,3}$ $c^*_{1,1}$ $c^*_{2,2}$ $\bar{c}_{i,2}$ $c^*_{2,3}$ $c^*_{2,1}$ $c^*_{3,2}$ $\bar{c}_{i,3}$ $c^*_{3,3}$ $c^*_{3,1}$
$v'_{1,1}$ $v'_{1,2}$ $v'_{1,3}$ $v'_{1,4}$ $v'_{2,1}$ $v'_{2,2}$ $v'_{2,3}$ $v'_{2,4}$ $v'_{3,1}$ $v'_{3,2}$ $v'_{3,3}$ $v'_{3,4}$

Figure 5: A variable gadget $X_i^\Delta = T_3$. The vertex $v_s$ has been renamed to $a_i$, and the vertex $v_t$ to $b_i$. The dashed horizontal line divides the gadget conceptually into two segments: no vertex rainbow path from $a_i$ to $b_i$ will cross the dashed line by Lemma 7.

*Proof.* For every $k$, the path $R$ must choose either $v_{k,1}$ or $v'_{k,1}$. Similarly, either $v_{k,4}$ or $v'_{k,4}$ must be chosen. If $v_{k,1}$ is chosen, then $v'_{k,4}$ cannot be chosen, as they share the same color $c^*_{k,1}$ by construction. Then, if $v'_{k,1}$ is chosen, $v'_{k,3}$ must be chosen. But then $v'_{k,3}$ and $v_{k,4}$ share the same color $c^*_{k,2}$ by construction. It follows that if $v_{k,1}$ is chosen, $v_{k,4}$ must be chosen. Symmetrically, if $v'_{k,1}$ is chosen, $v'_{k,4}$ must be chosen. □

The above lemma is illustrated in Figure 5.

Informally, the color scheme described above allows us to enforce "choose all" type of constraints. For a clause gadget, we wish to enforce "choose at least one" type of constraints. Indeed, the reader should be aware that in the following, while a clause gadget is structurally a parametric gadget $T_k$, its vertex-coloring will be different.

We will also mention that a parametric gadget $T_k$ with $k = 3$ will be constructed for each variable. Here, the reader should note we do not distinguish between say vertex $v_{1,1}$ in the first variable gadget, and the vertex $v_{1,1}$ in the second variable gadget. We feel the danger for confusion is not large enough to warrant the notational burden. We are then ready to proceed with the following.

**Theorem 8.** Rainbow Vertex Connectivity *is* NP*-complete when restricted to the class of triangle-free cubic graphs.*

*Proof.* We assume the terminology of Theorem 1. Given a 3-Occurrence 3-SAT instance $\phi = \bigwedge_{j=1}^m c_i$ over variables $x_1, x_2, \ldots, x_n$, we follow a strategy similar to Theorem 1. We will first describe how variable and clause gadgets of a graph $G_\phi^\Delta$ are built along with their vertex-colorings.

A variable gadget $X_i^\Delta$ is a parametric gadget $T_k$, where $k = 3$. To match Theorem 5 we shall rename, for each $i \in [n]$, the vertex $v_s$ to $a_i$ and the vertex $v_t$ to $b_i$ in a variable gadget $X_i^\Delta$. The uncolored vertices $v_{1,2}$, $v_{2,2}$, and $v_{3,2}$ receive colors $c_{i,1}$, $c_{i,2}$, and $c_{i,3}$, respectively. Similarly, the vertices $v'_{1,2}$, $v'_{2,2}$, and $v'_{3,2}$ receive colors $\bar{c}_{i,1}$, $\bar{c}_{i,2}$, and $\bar{c}_{i,3}$, respectively. Conceptually, these two sets of three vertices correspond to the positive and the negative $X_i$ path of Theorem 1. A variable gadget $X_i^\Delta$ along with its vertex-coloring is shown in Figure 5.

A clause gadget $C_j^\Delta$ is a parametric gadget $T_k$, where $k = 2$. To match Theorem 5 we shall rename, for each $j \in [m]$, the vertex $v_s$ to $p_j$ and the vertex $v_t$ to $q'_j$ in a clause gadget $C_j^\Delta$. We will then describe how each vertex of $C_j^\Delta$ is colored; note that we do not follow the usual coloring scheme of $T_k$ here. For convenience, let us rename $v_{1,1}$ to $r_{j,1}$, $v_{1,2}$ to $r_{j,2}$, $v_{1,3}$ to $r_{j,3}$, $v_{2,1}$ to $r_{j,4}$, and $v'_{2,3}$ to $r_{j,5}$. Also, let us rename $v'_{1,2}$ to $w_{j,1}$, $v'_{1,4}$ to $w_{j,2}$, and $v'_{2,2}$ to $w_{j,3}$. Then, the vertex $w_{j,\ell}$ for $\ell \in [3]$ is colored precisely as in Theorem 1. The vertex $v'_{1,1}$ receives color $c'_j$, and the vertex $v_{2,4}$ color $c_j$. We introduce a set of three "blocking" colors $\{c^*_{j,x}, c^*_{j,y}, c^*_{j,z}\}$, and color both vertices $v_{1,4}$ and $v'_{1,3}$ with $c^*_{j,x}$, both $v_{2,2}$ and $v'_{2,1}$ with $c^*_{j,y}$, and both $v_{2,3}$ and $v'_{2,4}$ with $c^*_{j,z}$. A clause gadget along with its vertex-coloring is shown in Figure 6 (a).

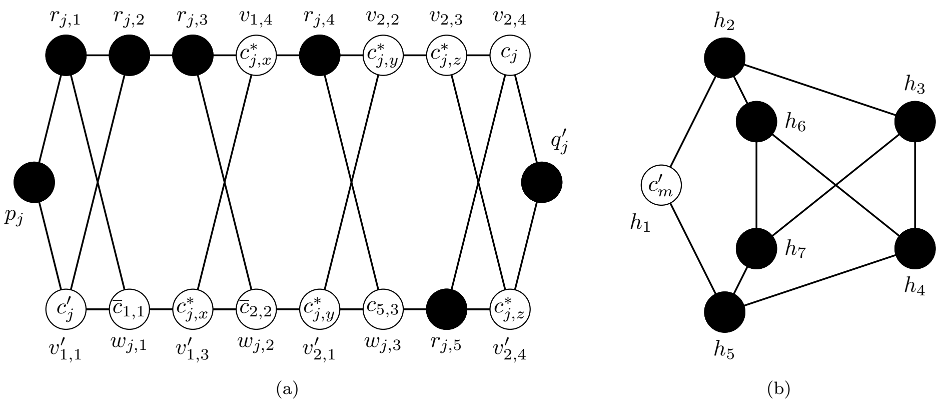


Figure 6: **(a)** A clause gadget $C_j^\Delta = T_2$ with some vertices renamed, and **(b)** the head gadget.

We will then describe how variable and clause gadgets are connected together, along with some additional gadgets. After describing the additional gadgets, we will explain how they are vertex-colored. For each $1 \leq i < n$, we connect $X_i^\Delta$ with $X_{i+1}^\Delta$ by adding the edge $(b_i, a_{i+1})$. Similarly, for each $1 \leq j < m$, we connect $C_i^\Delta$ with $C_{i+1}^\Delta$ by adding the edge $(q'_j, p_{j+1})$. Let us then subdivide the edge $(q'_j, p_{j+1})$ by a new vertex $f_j$. For each $f_j$, we introduce the following *dummy gadget*. A dummy gadget is constructed by starting from the cycle graph $C_5$ on the vertices $h_{j,q}$ in clockwise order, where $q \in [5]$, and two additional vertices $h_{j,6}$ and $h_{j,7}$. The construction of a dummy gadget is finished by adding the edges $(h_{j,2}, h_{j,6})$, $(h_{j,3}, h_{j,7})$, $(h_{j,4}, h_{j,6})$, $(h_{j,5}, h_{j,7})$, and $(h_{j,6}, h_{j,7})$. The vertex $f_j$ is made adjacent to $h_{j,1}$ by adding the edge $(f_j, h_{j,1})$. Finally, the two components are connected by adding the edge $(b_n, p_1)$.

We will then construct a *tail gadget*, which is a parametric gadget $T_k$ with $k = m$. In addition, we construct a single dummy gadget, and connect its degree two vertex $h_1$ with $G_\phi^\Delta$ by adding the edge $(q'_m, h_1)$. For convenience, we will refer to this dummy gadget as the *head gadget*. The head gadget is shown in Figure 6 (b).

We will then describe the vertex-coloring of the remaining vertices. In the tail gadget, both vertices $v_{j,2}$ and $v'_{j,2}$ receive color $c_j$, for every $j \in [m]$. Other vertices in a tail gadget follow the coloring scheme described for a $T_k$ in the beginning of Section 3.4. For each $1 \leq j < m$, we color vertex $f_j$ with color $c'_j$. Then, in the head gadget, the vertex $h_1$ receives color $c'_m$. Every other uncolored vertex of $G_\phi^\Delta$ receives a fresh new color that does not appear elsewhere. Formally, these are exactly the vertices in

$$\begin{aligned} Z = &\{a_i, b_i \mid 1 \leq i \leq n\} \\ &\cup \{p_j, q'_j, r_{j,1}, r_{j,2}, r_{j,3}, r_{j,4}, r_{j,5} \mid 1 \leq j \leq m\} \\ &\cup \{h_q \mid 2 \leq q \leq 7\} \\ &\cup \{h_{j,q} \mid 1 \leq j < m \land 1 \leq q \leq 7\}. \end{aligned}$$

As each gadget is cubic and triangle-free, the graph $G_\phi^\Delta$ is cubic and triangle-free. Consider a clause gadget $C_j^\Delta$, and a vertex rainbow path $R$ traversing from $p_j$ to $q'_j$ in it. It can be observed that because $R$ cannot choose either $v'_{1,1}$ or $v_{2,4}$, it must choose at least one of the vertices $w_{j,\ell}$, where $\ell \in [3]$. Then, Lemma 7 together with an argument similar to Lemma 3 gives the theorem. □

Furthermore, given a positive instance $\phi$ of 3-Occurrence 3-SAT, it can be observed every pair of vertices is connected by a vertex rainbow shortest path, giving us the following.

**Theorem 9.** Strong Rainbow Vertex Connectivity *is* NP*-complete when restricted to the class of triangle-free cubic graphs.*

## 3.5 $k$-regular graphs

In this subsection, we show both Rainbow Vertex Connectivity and Strong Rainbow Vertex Connectivity remain NP-complete on $k$-regular graphs, where $k \geq 4$. Our plan is to use the construc-

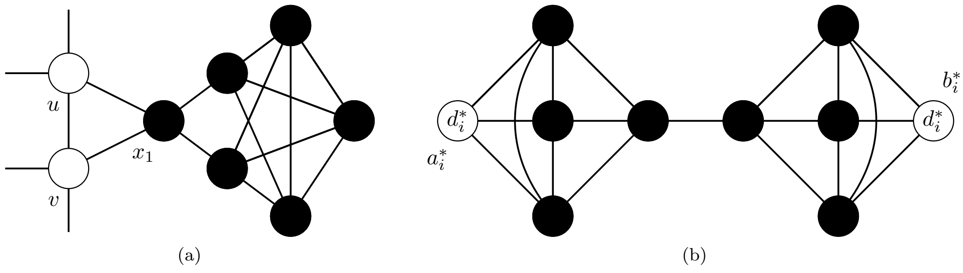


Figure 7: **(a)** The degree increment operation applied to $\{u, v\}$ with $d = 1$, where $u$ and $v$ have degree 3. The unlabeled vertices correspond to $w_1^{x_1}, \ldots, w_5^{x_1}$. **(b)** A detour gadget $D_{4,5}$ of diameter 5.

tion of Section 3.4, but add dummy vertices in a controlled manner to increase the degree of each vertex. In particular, we will need two operations detailed next.

Let $u$ and $v$ be two adjacent vertices such that $\deg(u) = \deg(v) = 3$. Let $d \geq 1$ be a constant, and consider the following *degree increment* operation. We introduce a set of vertices $X = \{x_1, \ldots, x_d\}$ along with the edges $\{(u, x), (v, x), (x, x') \mid x, x' \in X\}$. In other words, the vertices $\{u, v\} \cup X$ form a clique of size $d + 2$. For each $x \in X$, we introduce a clique $W_x$ on $d + 4$ new vertices $w_1^x, \ldots, w_{d+4}^x$ with an edge removed, say $(w_1^x, w_2^x) \notin W_x$. Finally, for each $x \in X$, we add the edges $(x, w_1^x)$ and $(x, w_2^x)$. We can then verify both $u$ and $v$ have degree $d + 3$. Furthermore, every new vertex we added has degree $d + 3$. The degree increment with $d = 1$ applied to two vertices $u$ and $v$ is illustrated in Figure 7 (a).

The degree increment operation suffices to show Rainbow Vertex Connectivity is NP-complete on $k$-regular graphs for $k \geq 4$. However, for Strong Rainbow Vertex Connectivity we need to be careful not to change certain distances in our construction. For this reason, we will need an additional *detour gadget $D_{d,l}$*. A building block $B$ of a detour gadget $D_{d,l}$ is the complete graph $K_{d-1}$ with two universal vertices added. The graph $B$ has $d - 1$ vertices of degree $d$, and two vertices of degree $d - 1$. By chaining such graphs $B$ together by adding an edge between the vertices of degree $d - 1$, we obtain a detour gadget $D_{d,l}$ for which it holds that the degree of every vertex is $d$ except for two vertices that have degree $d - 1$, and the diameter is $l = 2 + 3p$, for some $p \in \mathbb{N}^+$. A detour gadget $D_{4,5}$ is shown in Figure 7 (b).

We are then ready to proceed with our claim.

**Theorem 10.** *Both* Rainbow Vertex Connectivity *and* Strong Rainbow Vertex Connectivity *are* NP*-complete when restricted to the class of $k$-regular graphs, for every $k \geq 4$.*

*Proof.* Consider the vertex-colored cubic graph $G_\phi^\Delta$ constructed in the proof of Theorem 8. Through degree increment operations and addition of detour gadgets, we will transform the cubic graph $G_\phi^\Delta$ into a $k$-regular graph $G_\phi^*$, for any $k \geq 4$. Consider a variable gadget $X_i^\Delta$. We divide the vertices $v_{1,1}, v_{1,2}, \ldots, v_{3,4}$ into six pairs $\{v_{1,1}, v_{1,2}\}, \ldots, \{v_{3,3}, v_{3,4}\}$. Similarly, the vertices $v'_{1,1}, v'_{1,2}, \ldots, v'_{3,4}$ are divided into six pairs $\{v'_{1,1}, v'_{1,2}\}, \ldots, \{v'_{3,3}, v'_{3,4}\}$. For each of the altogether 12 pairs, we apply the degree increment operator with $d = k - 3$. We repeat this for each variable gadget in $G_\phi^\Delta$, and color each vertex arising from the operation with a fresh new color. Finally, consider the vertices $a_i$ and $b_i$ in a variable gadget $X_i^\Delta$. As the distance $d(a_i, b_i) = 13$, we introduce a detour gadget $D_{k,11}$ whose vertices of degree $k - 1$ are named $a_i^*$ and $b_i^*$. By adding the edges $(a_i, a_i^*)$ and $(b_i, b_i^*)$ we ensure $d(a_i, b_i)$ remains equal to 13. For each detour gadget $D_{k,11}$, we introduce a new color $d_i^*$ that does not appear anywhere else. We color both $a_i^*$ and $b_i^*$ with color $d_i^*$. Every vertex other than $a_i^*$ and $b_i^*$ receives a fresh distinct color. This ensures that an argument similar to Lemma 3 holds: no vertex rainbow path can pass through a detour gadget, as both $a_i^*$ and $b_i^*$ have the same color. For Lemma 2 to hold, it is enough to observe no vertex rainbow (shortest) path needs to have both $a_i^*$ and $b_i^*$ as its internal vertices. Indeed, for the remainder of the construction, each detour gadget will follow the same coloring scheme.

Let us then consider a clause gadget $C_j^\Delta$ of $G_\phi^\Delta$. Without loss, we can assume the given 3-Occurrence 3-SAT formula $\phi$ only contains clauses of size two and three. Indeed, clauses of size one can be removed by unit propagation. Consider the vertices $p_j$ and $q'_j$ in $C_j^\Delta$. When the corresponding clause is of size two, $d(p_j, q'_j) = 7$. Thus, similarly as above with a variable gadget, we add a detour gadget $D_{k,5}$, and connect it to $p_j$ and $q'_j$. Otherwise, the corresponding clause is of size three, and $d(p_j, q'_j) = 9$. In the obvious way, we can extend the length of the clause gadget $C_j^{\Delta}$ such that $d(p_j, q'_j) = 10$ by breaking the triangle-freeness of the gadget. The two vertices added to the clause gadget for this purpose receive fresh distinct colors. Then, we add a detour gadget $D_{k,8}$, and connect it with the clause gadget in the already described manner.

Consider then a vertex $f_j$ connecting two clause gadgets, for $j \in [m-1]$. We divide $f_j$ along with its dummy gadget into four pairs of vertices, and apply the degree increment operation for each with $d = k - 3$. In a similar fashion, we increase the degree of each vertex in the tail gadget, also possibly extending its length to accommodate for a detour gadget. For simplicity, we replace the head gadget as follows. We delete the vertices $h_q$ for $2 \leq q \leq 7$, and identify $h_1$ with $p_{j+1}$ of a new clause gadget $C_{j+1}^{\Delta}$. Each vertex of $C_{j+1}^{\Delta}$ receives a fresh new color (so $h_1$ still has color $c'_m$). As above, we increase the degree of each vertex in $C_{j+1}^{\Delta}$.

At this point, for the obtained graph $G_{\phi}^{*}$, it holds that every vertex has degree $k$, except for two vertices $v_s$ in the tail gadget, and $q'_{j+1}$ in the clause gadget $C_{j+1}^{\Delta}$ replacing the head gadget. To finish the construction, we connect a detour gadget with $v_s$ and $q'_{j+1}$, extending $C_{j+1}^{\Delta}$ in the obvious way if necessary. This completes the proof. □

## 4 Tractability considerations

In this section, we consider both Rainbow Vertex Connectivity and Strong Rainbow Connectivity from a structural viewpoint. We pinpoint graph classes for which both problems can be solved in polynomial time. Furthermore, we consider implications of our hardness results for parameterized algorithms, along with some positive parameterized results.

### 4.1 Polynomial time solvable cases

A graph is said to be *geodetic* if there is a unique shortest path between every pair of its vertices. It was proven by Stemple and Watkins [27] that a connected graph $G$ is geodetic if and only if every block of $G$ is geodetic. Indeed, we have the following.

**Observation 11.** *A block graph is geodetic.*

This immediately leads us to the following result.

**Corollary 12.** Strong Rainbow Vertex Connectivity *is solvable in polynomial time when restricted to the class of block graphs.*

More generally, a graph is said to be *$k$-geodetic* if there are at most $k$ shortest paths between every pair of vertices. Quite trivially, Strong Rainbow Vertex Connectivity is solvable in polynomial time on such graphs. This includes e.g., bigeodetic graphs [28] (that is, $k = 2$).

It is known that Rainbow Connectivity is NP-complete for the class of block graphs. However, it turns out this is not the case for Rainbow Vertex Connectivity. Indeed, the following lemma suggests a straightforward algorithm for the problem.

**Lemma 13.** *Two distinct vertices $s$ and $t$ are rainbow vertex connected in a vertex-colored block graph if and only if each cut vertex on the unique $s$-$t$ shortest path has a distinct color.*

*Proof.* The vertices $s$ and $t$ are rainbow vertex connected regardless of the underlying vertex coloring if $d(s, t) \leq 2$. So we can assume $d(s, t) \geq 3$. Recall that by Observation 11, the shortest path between $s$ and $t$ is unique. We will then show that if $s$ and $t$ are rainbow vertex connected, then each cut vertex on the unique shortest $s$-$t$ path $P$ has received a different color. Suppose not, i.e., $s$ and $t$ are rainbow vertex connected, but at least two cut vertices on $P$ share the same color. But because any $s$-$t$ path uses every cut vertex on $P$, we have a contradiction. The other direction is trivial. □

In other words, a vertex-colored block graph is rainbow vertex connected if and only if it is strongly rainbow vertex connected. Thus, the previous lemma establishes the following.

**Corollary 14.** Rainbow Vertex Connectivity *is solvable in polynomial time when restricted to the class of block graphs.*

In the $st$-version of Strong Rainbow Vertex Connectivity, the input has two additional vertices $s$ and $t$. The task is to decide whether there is a vertex rainbow shortest path between $s$ and $t$ in the graph $G$. Let us refer to this problem as Strong Rainbow Vertex $st$-Connectivity. We define the problem Rainbow Vertex $st$-Connectivity analogously.

**Lemma 15.** *The* Strong Rainbow Vertex $st$-Connectivity *problem for cactus graphs reduces to the* Rainbow Vertex $st$-Connectivity *problem for cactus graphs.*

*Proof.* Let $I = (G, \psi, s, t)$ be an instance of Strong Rainbow Vertex $st$-Connectivity, where $G$ is a cactus graph, and $\psi$ its vertex-coloring. In polynomial time, we will construct an instance $I' = (G', \psi', s, t)$ of Rainbow Vertex $st$-Connectivity where $G'$ is a cactus graph such that $I$ is a YES-instance of Strong Rainbow Vertex $st$-Connectivity if and only if $I'$ is a YES-instance of Rainbow Vertex $st$-Connectivity.

To construct $I'$, we first let $G' = G$ and $\psi' = \psi$. Then, we delete from $G'$ every vertex $w$ such that $d(s,w) + d(w,t) \neq d(s,t)$. This is achieved by running two breadth-first searches; one from $s$ and one from $t$, recording the distance to every other vertex. In other words, $G'$ contains only vertices that appear on some shortest $s$-$t$ path. Clearly, the property of being a cactus graph is closed under vertex deletion. Thus, $G'$ is a cactus graph. By observing precisely cycles of even length are preserved in $G'$, it is straightforward to verify that $I$ is a YES-instance of Strong Rainbow Vertex $st$-Connectivity if and only if $I'$ is a YES-instance of Rainbow Vertex $st$-Connectivity. □

It is shown by Uchizawa et al. [13] that Rainbow Vertex $st$-Connectivity can be solved in polynomial time for outerplanar graphs, which form a superclass of cacti. By applying the above reduction to each pair of vertices, we obtain the following.

**Corollary 16.** Strong Rainbow Vertex Connectivity *is solvable in polynomial time when restricted to the class of cactus graphs.*

## 4.2 Consequences for parameterized algorithms

It is known that both Rainbow Connectivity and Rainbow Vertex Connectivity remain NP-complete for graphs of bounded diameter. However, Strong Rainbow Connectivity is in XP parameterized by the diameter on the input graph [14]. Indeed, by the same argument as in [14, Theorem 11], we establish a similar result for the strong vertex variant.

**Observation 17.** Strong Rainbow Vertex Connectivity *is in* XP *parameterized by the diameter of the input graph.*

This implies Strong Rainbow Vertex Connectivity is in XP for several other structural parameters including domination number, independence number, minimum clique cover, distance to cograph, distance to cluster, distance to co-cluster, distance to clique, and vertex cover. We refer the reader to Komusiewicz and Niedermeier [29] for a visualization of the relationships of many graph parameters. In fact, it can be observed the diameter of any split graph is at most three. Thus, we obtain the following for both strong variants of the problem.

**Corollary 18.** *Both* Strong Rainbow Connectivity *and* Strong Rainbow Vertex Connectivity *are solvable in polynomial time when restricted to the class of split graphs.*

It follows from the work of Uchizawa et al. [13] that Rainbow Vertex Connectivity is NP-complete for graphs of bounded treewidth. The pathwidth of an interval graph $G$ is $\omega(G) - 1$, i.e., one less than the size of the maximum clique in $G$. We can observe the maximum clique in the graph $G^I_\phi$ constructed in Theorem 5 is of size 4. Thus, hardness of both problems for bounded pathwidth graphs follow. Furthermore, we can connect a clique of size at least 5 to $G^I_\phi$, and color each of its vertices with a fresh new color. Thus, we obtain the following.

**Theorem 19.** *Both* Rainbow Vertex Connectivity *and* Strong Rainbow Vertex Connectivity *remain* NP-*complete when restricted to the class of graphs with pathwidth $p$, for every $p \geq 3$.*

Recall the bandwidth of a graph $G$ is one less than the maximum clique size of any proper interval supergraph of $G$, chosen to minimize its clique number. In Theorem 6, the graph constructed is already a proper interval graph. Moreover, we can verify its maximum clique size is 4. But we started the construction from the graph built in Theorem 5. Thus, we can connect a clique of any size colored with fresh new colors to either one of the graphs, and observe the following.

**Theorem 20.** *Both* Rainbow Vertex Connectivity *and* Strong Rainbow Vertex Connectivity *remain* NP*-complete when restricted to the class of graphs with bandwidth $b$, for every $b \geq 3$.*

Finally, one can observe Theorem 19 also implies hardness for bounded treewidth graphs. Thus, it is interesting to consider a parameter stronger than pathwidth. Indeed, tree-depth is an upper bound on the pathwidth of a graph. It was shown by Nešetřil and Ossona de Mendez [30] that the length of a longest path in an undirected graph $G$ is upper bounded by $2\,\mathrm{td}(G) - 2$. Using this fact in combination with the argument given in [14, Theorem 11], we have the following.

**Observation 21.** *Both* Rainbow Vertex Connectivity *and* Strong Rainbow Vertex Connectivity *are in* XP *parameterized by the tree-depth of the input graph.*

The previous observation raises a natural question: is either problem FPT for tree-depth? Similarly, in the light of Observation 17, it is interesting to ask whether Strong Rainbow Vertex Connectivity or Strong Rainbow Connectivity is FPT parameterized by the diameter of the input graph. In the following, we remark these questions have a positive answer.

Uchizawa et al. [13] gave a dynamic programming algorithm for solving all four problems in $2^k n^{O(1)}$ time and exponential space, where $k$ is the number of colors used in the coloring of the input graph. Their algorithm decides whether there is a rainbow walk from an arbitrary vertex $s$ to each vertex $v \in V \setminus \{s\}$. The crucial property is that any rainbow $s$-$v$ walk is of length at most $k$, for otherwise a color would have to repeat. We remark that their algorithm is also an FPT algorithm for any parameter that bounds the longest (shortest) path length. Indeed, if the diameter is bounded, this gives us an upper bound on the length of a walk to compute in the strong variant. The observation is similar for tree-depth.

**Theorem 22.** *All problems* Rainbow Connectivity, Strong Rainbow Connectivity, Rainbow Vertex Connectivity, *and* Strong Rainbow Vertex Connectivity *are* FPT *parameterized by the tree-depth of the input graph.*

**Theorem 23.** *Both* Strong Rainbow Connectivity *and* Strong Rainbow Vertex Connectivity *are* FPT *parameterized by the diameter of the input graph.*

## 5 Concluding remarks

We gave several complexity results for both Rainbow Vertex Connectivity and Strong Rainbow Vertex Connectivity (see Table 1). The goal was to investigate whether the complexity results for the edge variants in [14] could be extended for the vertex variants. As the results in Table 1 show, it is not a priori obvious how complexity is affected when considering the vertex variants for a particular graph class. This is showcased by e.g., block graphs. In the process, we obtained further negative results for the edge variants, and positive parameterized results for all four problems.

Previously, it was shown in [13] that Rainbow Vertex Connectivity is NP-complete for series-parallel graphs. We remark the same is true for Strong Rainbow Vertex Connectivity. Indeed, we follow precisely the reduction given in [13], but reduce from Strong Rainbow Connectivity instead of Rainbow Connectivity.

From a parameterized perspective, it seems the strong variants are more tractable. Moreover, Table 1 suggests the vertex variants are never harder than the edge variants. Is there a graph class for which say Rainbow Vertex Connectivity is hard, but Rainbow Connectivity easy? It is also interesting to consider the complexity of the weak problem variants for split graphs. In particular, the vertex variant is trivial for split graphs of diameter 2, but what about split graphs of diameter 3?

## References

[1] M. Krivelevich, R. Yuster, The rainbow connection of a graph is (at most) reciprocal to its minimum degree, Journal of Graph Theory 63 (2010) 185–191.

[2] X. Li, Y. Mao, Y. Shi, The strong rainbow vertex-connection of graphs, Utilitas Mathematica 93 (2014) 213–223.

[3] G. Chartrand, G. Johns, K. McKeon, P. Zhang, Rainbow connection in graphs, Mathematica Bohemica 133 (2008).

[4] S. Chakraborty, E. Fischer, A. Matsliah, R. Yuster, Hardness and algorithms for rainbow connection, Journal of Combinatorial Optimization 21 (2009) 330–347.

[5] P. Dorbec, I. Schiermeyer, E. Sidorowicz, E. Sopena, Rainbow connection in oriented graphs, Discrete Applied Mathematics 179 (2014) 69–78.

[6] X. Li, Y. Shi, Y. Sun, Rainbow Connections of Graphs: A Survey, Graphs and Combinatorics 29 (2012) 1–38.

[7] X. Li, Y. Sun, Rainbow connections of graphs, Springer, 2012.

[8] G. Chartrand, P. Zhang, Chromatic graph theory, CRC press, 2008.

[9] L. Chen, X. Li, Y. Shi, The complexity of determining the rainbow vertex-connection of a graph, Theoretical Computer Science 412 (2011) 4531–4535.

[10] L. Chen, X. Li, H. Lian, Further hardness results on the rainbow vertex-connection number of graphs, Theoretical Computer Science 481 (2013) 18–23.

[11] E. Eiben, R. Ganian, J. Lauri, On the complexity of rainbow coloring problems, in: Z. Lipták, W. F. Smyth (Eds.), Combinatorial Algorithms - 26th International Workshop, IWOCA 2015, Verona, Italy, October 5-7, 2015, Revised Selected Papers, volume 9538 of *Lecture Notes in Computer Science*, Springer, 2015, pp. 209–220.

[12] X. Huang, X. Li, Y. Shi, Note on the hardness of rainbow connections for planar and line graphs, Bulletin of the Malaysian Mathematical Sciences Society (2014) 1–7.

[13] K. Uchizawa, T. Aoki, T. Ito, A. Suzuki, X. Zhou, On the Rainbow Connectivity of Graphs: Complexity and FPT Algorithms, Algorithmica 67 (2013) 161–179.

[14] J. Lauri, Further hardness results on rainbow and strong rainbow connectivity, Discrete Applied Mathematics 201 (2016) 191–200.

[15] R. Diestel, Graph Theory, Springer-Verlag Heidelberg, 2005.

[16] R. J. Duffin, Topology of series-parallel networks, Journal of Mathematical Analysis and Applications 10 (1965) 303–318.

[17] F. Gavril, The intersection graphs of subtrees in trees are exactly the chordal graphs, Journal of Combinatorial Theory, Series B 16 (1974) 47–56.

[18] P. C. Gilmore, A. J. Hoffman, A characterization of comparability graphs and of interval graphs, Canad. J. Math 16 (1964) 4.

[19] F. S. Roberts, Indifference graphs, in: Proof techniques in graph theory, Academic Press, New York, 1969, pp. 139–146.

[20] H. Kaplan, R. Shamir, Pathwidth, bandwidth, and completion problems to proper interval graphs with small cliques, SIAM Journal on Computing 25 (1996) 540–561.

[21] H. L. Bodlaender, A partial k-arboretum of graphs with bounded treewidth, Theoretical Computer Science 209 (1998) 1–45.

[22] H. Bodlaender, J. Gilbert, H. Hafsteinsson, T. Kloks, Approximating treewidth, pathwidth, frontsize, and shortest elimination tree, Journal of Algorithms 18 (1995) 238–255.

[23] R. G. Downey, M. R. Fellows, Fundamentals of Parameterized Complexity, Springer, 2013.

[24] M. Cygan, F. V. Fomin, Ł. Kowalik, D. Lokshtanov, D. Marx, M. Pilipczuk, M. Pilipczuk, S. Saurabh, Parameterized Algorithms, Springer, 2015.

[25] C. H. Papadimitriou, Computational complexity, Addison-Wesley, 1994.

[26] C. A. Tovey, A simplified NP-complete satisfiability problem, Discrete Applied Mathematics 8 (1984) 85–89.

[27] J. G. Stemple, M. E. Watkins, On planar geodetic graphs, Journal of Combinatorial Theory 4 (1968) 101–117.

[28] N. Srinivasan, J. Opatrny, V. Alagar, Bigeodetic graphs, Graphs and Combinatorics 4 (1988) 379–392.

[29] C. Komusiewicz, R. Niedermeier, New races in parameterized algorithmics, in: Mathematical Foundations of Computer Science 2012, volume 7464 of *Lecture Notes in Computer Science*, Springer Berlin Heidelberg, 2012, pp. 19–30.

[30] J. Nešetřil, P. Ossona de Mendez, Grad and classes with bounded expansion I. Decompositions, European Journal of Combinatorics 29 (2008) 760–776.

## Appendix

As mentioned in Subsection 3.1, all of our reductions from 3-Occurrence 3-SAT assume each clause of the input formula has exactly three literals. For completeness, we present here clause gadgets corresponding to clauses of size two for each graph class considered.

The clause gadgets for different graph classes are shown in Figure 8. The first column denotes the graph class. The second column shows a clause gadget corresponding to a clause containing two literals. See the respective theorems for an explanation of the colors appearing on the vertices.

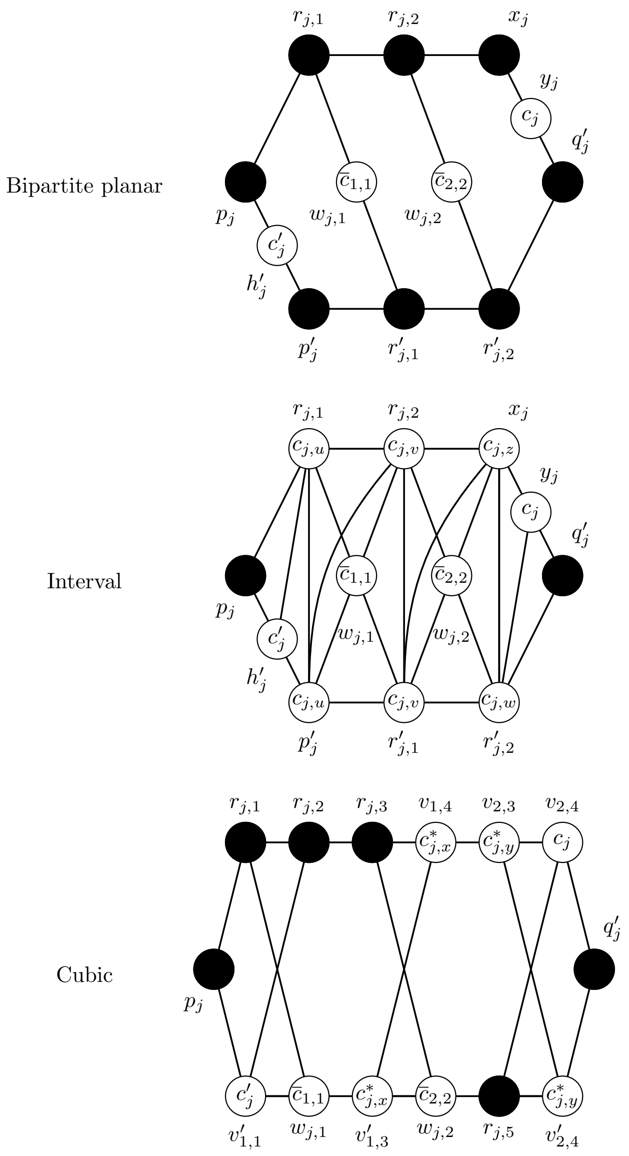


Figure 8: Clause gadgets corresponding to clauses of size two for different graph classes.

# Paper 3

Łukasz Kowalik and Juho Lauri.

**On finding rainbow and colorful paths.**

3

© Elsevier B.V., 2016. Theoretical Computer Science 628 (2016), pp. 110–114.

doi:10.1016/j.tcs.2016.03.017

Theoretical Computer Science 628 (2016) 110–114

Contents lists available at ScienceDirect

Theoretical Computer Science

www.elsevier.com/locate/tcs

Note

# On finding rainbow and colorful paths [☆]

Łukasz Kowalik [a], Juho Lauri [b,*]

[a] University of Warsaw, Poland
[b] Tampere University of Technology, Finland

A R T I C L E   I N F O

*Article history:*
Received 26 January 2016
Received in revised form 21 February 2016
Accepted 10 March 2016
Available online 21 March 2016
Communicated by F.V. Fomin

*Keywords:*
Colorful path
Rainbow connectivity
Graph algorithms
Computational complexity

A B S T R A C T

In the Colorful Path problem we are given a graph $G = (V, E)$ and an arbitrary vertex coloring function $c : V \to [k]$. The goal is to find a *colorful path*, i.e., a path on $k$ vertices, that visits each color. This problem has been introduced in the classical work of Alon et al. (1995) [1], and the authors proposed a dynamic programming algorithm that runs in time $2^k n^{O(1)}$ and uses $O(2^k)$ space. Since then the only progress obtained is reducing the space size to a polynomial at the cost of using randomization. In this work we show that a progress in time complexity is unlikely: if Colorful Path can be solved in time $(2-\varepsilon)^k n^{O(1)}$, then Set Cover admits a $(2-\varepsilon')^n (nm)^{O(1)}$-time algorithm. The same applies to other versions of the problem: when edges are colored instead of vertices, or we ask for a walk instead of a path, or when the requested path/walk has specified endpoints.

We study also a second, very related problem. In Rainbow $st$-Connectivity, we are given a $k$-edge-colored graph and two vertices $s$ and $t$. The goal is to decide whether there is a rainbow path between $s$ and $t$, that is, a path on which no color repeats. In its vertex variant (Rainbow Vertex $st$-Connectivity) the input graph is $k$-vertex-colored, and a rainbow path is defined analogously. Uchizawa et al. (2011) [14] show that both variants can be solved in $2^k n^{O(1)}$ time and exponential space. We show that the space size can be reduced to a polynomial, while keeping the same running time. In contrast to the polynomial space algorithm for Colorful Path, our algorithm for finding rainbow paths is deterministic.

## 1. Introduction

Finding a path between two vertices is one of the most fundamental graph problems. However, in many applications, we often seek to find such a path with additional properties. For example, the textbook of Kleinberg and Tardos [10] (Exercise 8.12) describes the following application in monitoring. A company has a website that has both subscribers and nonsubscribers. All content is shown to subscribers, but access for nonsubscribers is limited. More specifically, nonsubscribers can view any page, but the maximum number of pages viewed in a single session is limited. The website is modeled by a (directed) graph $G = (V, E)$, in which the vertices correspond to pages, and edges to hyperlinks. The website has a front page, which a particular vertex $s \in V$ corresponds to. To track a user session, the vertex set $V$ is divided into color classes $Z_1, Z_2, \dots, Z_k \subseteq V$, where each color class $Z_i$ represents a zone. A navigation path of a nonsubscriber starting

☆ Work partially supported by the National Science Centre of Poland, grant number 2013/09/B/ST6/03136 (Ł.K.), and by the Emil Aaltonen Foundation (J.L.).

* Corresponding author.
*E-mail addresses:* kowalik@mimuw.edu.pl (Ł. Kowalik), juho.lauri@tut.fi (J. Lauri).

http://dx.doi.org/10.1016/j.tcs.2016.03.017
0304-3975/© 2016 Elsevier B.V. All rights reserved.

from $s$ is restricted to include at most one page from each zone $Z_i$. Otherwise, the user's session is terminated, and an ad is shown suggesting the user becomes a subscriber. A question the company asks is whether it is possible for a nonsubscriber to navigate from the front page $s$ to some other page $t$ in a single session, i.e., whether there is an $s$-$t$ path that passes each zone at most once.

The problem above is modelled by the Rainbow Vertex $st$-Connectivity problem. We are given a graph, a coloring function $c : V \to [k]$, and two vertices $s$ and $t$. The goal is to decide whether there is a *rainbow path* between $s$ and $t$, that is, a path on which no color repeats.[1] Somewhat more studied is the edge version, called Rainbow $st$-Connectivity, where the edges are colored in $k$ colors and again, a path is rainbow when no color repeats on it. The problems were shown NP-complete by Chen, Li, and Shi [5] and Chakraborty, Fischer, Matsliah, and Yuster [4], respectively. Moreover, Uchizawa, Aoki, Ito, Suzuki, and Zhou [14] performed a more fine-grained study into the complexity of the problems for restricted graph classes; for instance, they show the edge version remains NP-complete on graphs of bounded treewidth, and on graphs of diameter 2. Building on the results of [14], additional hardness results are given for $k$-regular graphs for every $k \geq 3$, interval outerplanar graphs, and block graphs among others in [12]. Moreover, Uchizawa et al. [14] showed that both edge and vertex variant of the problem can be solved by a $2^k n^{O(1)}$-time dynamic programming algorithm that uses $O(k2^k n)$ space.

A *colorful path* is a special case of a rainbow path, where all the $k$ colors are supposed to show up on the path. In the Colorful Path problem we are given a graph $G = (V, E)$ and an arbitrary vertex coloring function $c : V \to [k]$. The goal is to find a *colorful path*. The problem can be seen to be NP-complete by a simple reduction from $k$-Path. Colorful Path has been introduced in the classical work of Alon, Yuster, and Zwick [1], and the authors proposed a dynamic programming algorithm that runs in time $2^k n^{O(1)}$ and uses $O(2^k)$ space. Yet another related problem is called Colorful Graph Motif: instead of a path we ask for a *subtree* where colors do not repeat. Guillemot and Sikora [9] proposed a $2^k n^{O(1)}$-time and $O(kn)$-space Monte-Carlo randomized algorithm for Colorful Graph Motif. It is easy to see that their algorithm can be simplified to solve Colorful Path within the same time and space bounds (see also Exercise 10.17 in the textbook of Cygan et al. [7]).

**Our results** A major downside of the algorithm of Uchizawa et al. [14] for Rainbow $st$-Connectivity is that it uses exponential space. Especially for a practical implementation, exponential space complexity is prohibitive. Our first result is a deterministic algorithm which solves the decision version of Rainbow $st$-Connectivity for an $n$-vertex and $m$-edge input graph in time $O(k2^k(m + k^2))$ and space $O(n + k)$. The actual path can be found at the cost of additional $O(k \log n)$ running time overhead. The same results apply to the vertex version of the problem.

Our second result explains the modest progress on Colorful Path over the last twenty years. Namely, we show that if for some $\varepsilon > 0$ problem Colorful Path can be solved in time $(2 - \varepsilon)^k n^{O(1)}$, then there is $\varepsilon' > 0$ such that Set Cover admits a $(2 - \varepsilon')^n (nm)^{O(1)}$-time algorithm that solves any instance with $m$ sets over an universe of size $n$. The same applies to other versions of the problem: when edges are colored instead of vertices, or we ask for a walk instead of a path, or when the requested path/walk has specified endpoints. The existence of a $(2 - \varepsilon')^n (nm)^{O(1)}$-time algorithm for Set Cover would be a major breakthrough. The Set Cover Conjecture (see [7]) says that for every $\varepsilon < 1$, there is an integer $k$ such that Set Cover with $m$ sets of size at most $k$ cannot be computed in time $2^{\varepsilon n}(mk)^{O(1)}$. Note that the Set Cover Conjecture implies that there is no $(2 - \varepsilon')^n (mn)^{O(1)}$-time algorithm for Set Cover, for any $\varepsilon' > 0$. Under the Set Cover Conjecture Cygan et al. [6] show exponential lower bounds for Steiner Tree, Connected Vertex Cover, and Subset Sum. We note that even if the Set Cover Conjecture if false, these lower bounds and the lower bound of the present paper are meaningful: instead of trying to reduce the time of the best known algorithms for Colorful Path or Steiner Tree, one should rather focus on the more basic Set Cover.

**Notation** For standard graph-theoretic notation not defined here, we refer the reader to [8]. All graphs we consider in this paper are simple and undirected. For a positive integer $k$, we write $[k]$ to denote the set $\{1, 2, \ldots, k\}$. If $I$ and $J$ are instances of decision problems $P$ and $R$, respectively, then we say that $I$ and $J$ are *equivalent*, when either both $I$ and $J$ are YES-instances or both are NO-instances.

## 2. Rainbow *st*-connectivity in polynomial space

In this section, we show the problems Rainbow $st$-Connectivity and Rainbow Vertex $st$-Connectivity can be solved in $2^k n^{O(1)}$-time and polynomial space, where $k$ is the number of colors. Although Rainbow $st$-Connectivity can be easily reduced to Rainbow Vertex $st$-Connectivity (basically by replacing the original graph by its line graph), we study these problems separately in order to avoid unnecessary polynomial overhead in the running time caused by the instance size blow-up in the reduction.

Our plan is to reduce the problems to Edge-Colorful Walk and Colorful Path, respectively. In Edge-Colorful Walk the input is a graph $G = (V, E)$, a coloring $c : E \to [k]$, and a pair of vertices $s, t$. The goal is to verify if there is a colorful $s$-$t$ walk in $G$. We give the reductions below.

[1] In previous works (see e.g., [11]), the definition of a rainbow path in a vertex-colored graph differs slightly: it is required that the colors do not repeat only on the internal vertices of the path. However, the decision problems are easily seen to be computationally equivalent (in one direction: remove vertices colored by $c(s)$ and $c(t)$; in the other: color $s$ and $t$ with two new colors $k + 1$ and $k + 2$).

**Lemma 1.** *For any instance $(G, c)$ of* Rainbow *st*-Connectivity *where $G = (V, E)$ and $c : E \to [k]$ we can build in polynomial time an equivalent instance $(G' = (V', E'), c')$ of* Edge-Colorful Walk, *where $|V'| = |V| + k$, $|E'| = |E| + O(k^2)$, and $c'$ is a coloring of $E'$ using $k$ colors.*

**Proof.** Let $(G, c, s, t)$ be an instance of Rainbow *st*-Connectivity. We construct an instance $(G', c', s, t)$ of Edge-Colorful Walk as follows. Begin with $G' = G$, and $c' = c$. Add a clique on $k$ vertices $K = \{v_1, \dots, v_k\}$. For every pair of integers $i, j$, where $1 \le i < j \le k$, the edge $v_i v_j$ is colored with $j$. For every $i = 1, \dots, k$, add an edge $sv_i$ colored with $i$. Finally, for every $i = 1, \dots, k$, and a vertex $w \in N_G(s)$ add an edge $v_i w$ colored with $c(sw)$. This ends the construction. We claim that $(G, c, s, t)$ is a YES-instance of Rainbow *st*-Connectivity iff $(G', c', s, t)$ is a YES-instance of Edge-Colorful Walk.

Suppose there is a rainbow path $P$ from $s$ to $t$ in $G$. We show that there is a colorful $st$-path $P'$ in $G'$. If $P$ is colorful, we are done. Otherwise, let $M = [k] \setminus c(E(P))$ be the set of colors missing in $P$. Let $c_1, \dots, c_{|M|}$ be the colors of $M$ in increasing order. Let $P = s, x_1, \dots, x_\ell$, where $x_\ell = t$. Then $P' = (s, v_{c_1}, \dots, v_{c_{|M|}}, x_1, \dots, x_\ell)$ is a required colorful path.

For the other direction, suppose there is an edge-colorful walk $W$ from $s$ to $t$ in $G'$. Then $W$ goes through at least one edge between $\{s\} \cup K$ and $N_G(s)$. Let $e = (x, y)$ be the last such edge visited by $W$, where $x \in \{s\} \cup K$ and $y \in N_G(s)$. Then $W$ decomposes into $W'eW''$, where $W''$ does not visit any vertex in $\{s\} \cup K$. Consider the walk $W_1 = (s, y, W'')$. Then $W_1$ is a walk in $G$. Since $c(sy) = c'(xy)$, walk $W_1$ has the same color sequence as walk $eW$ in $G'$, hence $W_1$ is rainbow. By removing cycles from $W_1$ we obtain a rainbow path in $G$. $\square$

Now we present an analogous result for the vertex version.

**Lemma 2.** *For any instance $(G, c)$ of* Rainbow Vertex *st*-Connectivity *where $G = (V, E)$ and $c : E \to [k]$ we can build in polynomial time an equivalent instance $(G' = (V', E'), c')$ of* Colorful Path, *where $|V'| = |V| + k - 1$, $|E'| = |E| + O(k^2)$, and $c'$ is a coloring of $E'$ using $k$ colors.*

**Proof.** Let $(G, c, s, t)$ be an instance of Rainbow Vertex *st*-Connectivity. We construct an instance $(G', c', s', t)$ of Colorful Path very similarly as in Lemma 1, as follows. Begin with $G' = G$, and $c' = c$. Remove $s$ and add a clique on $k$ vertices $K = \{v_1, \dots, v_k\}$. For every $i = 1, \dots, k$ color $v_i$ with $i$. Finally, for every pair of vertices $v \in K$ and $w \in N_G(s)$ add an edge $vw$. Denote $s' = v_{c(s)}$. This ends the construction. Analogously as in Lemma 1 one can show that $(G, c, s, t)$ is a YES-instance of Rainbow Vertex *st*-Connectivity iff $(G', c', s', t)$ is a YES-instance of Colorful Path. $\square$

Now we give efficient algorithms for Edge-Colorful Walk and Colorful Path. We are going to use the intersection version of the inclusion-exclusion principle (see e.g., [7]), stated below.

**Theorem 3** *(Inclusion–exclusion principle, intersection version). Let $A_1, \dots, A_n \subseteq U$, where $U$ is a finite set. Denote $\bigcap_{i\in\emptyset}(U \setminus A_i) = U$. Then*

$$\Big|\bigcap_{i\in[n]} A_i\Big| = \sum_{X\subseteq[n]} (-1)^{|X|} \Big|\bigcap_{i\in X}(U \setminus A_i)\Big|.$$

**Theorem 4.** *Assume the input graph has $n$ vertices, $m$ edges, and its vertices (or edges) are colored in $k$ colors. Decision problems* Edge-Colorful Walk *and* Colorful Path *can be solved in $k2^k m$ deterministic time and $O(n)$ space.*

**Proof.** We focus on Colorful Path; the edge-colored version is analogous. Assume the goal is to decide whether there is a colorful path from $s$ to $t$. Note that this is equivalent to checking whether there is a colorful *walk* from $s$ to $t$. For every color $i \in [k]$, define $A_i$ to be the set of $k$-vertex walks from $s$ to $t$ that visit a vertex colored by $i$. Our plan is to compute $|\bigcap_{i\in[k]} A_i|$ and return YES iff its value is positive. By Theorem 3 the claim boils down to showing that there is an algorithm running in time $O(km)$ which, given a subset $X \subseteq [k]$, finds the number of $k$-vertex walks from $s$ to $t$ that *do not* visit any edge colored by a color from $X$. However, this is equivalent to finding the number of $k$-vertex walks from $s$ to $t$ in the subgraph induced by the colors in $[k] \setminus X$. This is a routine task which can be done in time $O(km)$ by a simple dynamic programming algorithm using only space $O(n)$. $\square$

Note that the theorem above solves only the decision problems (incidentally, also counting problems), but not the finding version of the problems. However, the actual walks can be found my means of self-reducibility. In particular, observe that Theorem 4 gives an algorithm that decides whether there is a colorful $st$-path in a subgraph $G'$ of the input graph $G$ induced by a subset of edges $E'$. Thus we can retrieve the solution walk by removing edges and using the decision algorithm to test whether the solution still exists. However, this requires $O(m)$ calls to the decision algorithm. Björklund, Kaski, and Kowalik [3] show that group testing methods can be used to decrease the number of calls to $O(k \log n)$.

From Lemmas 1, 2, and Theorem 4 we get the following result.

**Corollary 5.** *Assume the input graph has $n$ vertices, $m$ edges, and its vertices (or edges) are colored in $k$ colors. Then, both* Rainbow *st*-Connectivity *and* Rainbow Vertex *st*-Connectivity *can be solved in $k2^k(m + k^2)$ deterministic time and space $O(n + k)$.*

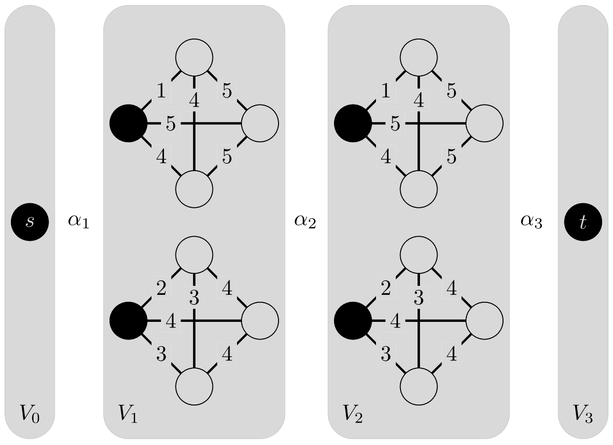


**Fig. 1.** An instance of Edge-Colorful $st$-Path constructed from an instance $(U, \mathcal{S}, r)$ of Set Cover, where $U = [5]$, $\mathcal{S} = \{\{1, 4, 5\}, \{2, 3, 4\}\}$, and $r = 2$. Black vertices are portals. To reduce clutter, the edges between two layers are not shown.

## 3. Finding a colorful path is hard

Recall that in the Set Cover problem we are given an integer $r$ and a family of sets $\mathcal{S} = \{S_1, S_2, \ldots, S_m\}$ over the universe $U = \bigcup_{j=1}^{m} S_j$ with $n = |U|$. The task is to decide whether there is a subfamily of $r$ sets $S_{i_1}, S_{i_2}, \ldots, S_{i_r}$ such that $U = \bigcup_{j=1}^{r} S_{i_j}$. In this section, we prove that under reasonable complexity assumptions, it is unlikely that Edge-Colorful $st$-Path has an algorithm running in time $(2 - \varepsilon)^r (mn)^{O(1)}$, for any $\varepsilon > 0$. The proof boils down to the following lemma.

**Lemma 6.** *There exists a polynomial time algorithm which given an instance $I = (\mathcal{S}, t)$ of* Set Cover *creates an instance $I' = (G, s, t)$ of* Edge-Colorful $st$-Path, *where $G$ is edge-colored with $n + r + O(1)$ colors, such that $I$ is a YES-instance of* Set Cover *iff $I'$ is a YES-instance of* Edge-Colorful $st$-Path. *The same applies to other variants of the problem* Edge-Colorful $st$-Path, *i.e., for any problem $A$-*Colorful *$BC$, where $A \in \{$*Edge, Vertex*$\}$, $B \in \{st, \varepsilon\}$, and $C \in \{$*Path, Walk*$\}$.*

**Proof.** Let $I = (U, \mathcal{S}, r)$ be an instance of Set Cover. In polynomial time, we create the following instance $I' = (G, c, s, t)$ of Edge-Colorful $st$-Path, where the coloring $c$ uses $n + r + 1$ colors from the set $U \cup \{\alpha_1, \ldots, \alpha_{r+1}\}$. We first describe a gadget used for a set $S_i \in \mathcal{S}$, and then the whole graph $G = (V, E)$.

**Set gadget** We construct the following gadget $X_i$ for each set $S_i = \{e_1, e_2, \ldots, e_{|S_i|}\} \in \mathcal{S}$. The gadget $X_i$ is a clique with vertex set $\{0, \ldots, |S_i|\}$. The vertex 0 is called the *portal of* $X_i$. For every $i$, $j$, such that $0 \le i < j \le |S_i|$, the edge $ij$ is colored with color $e_j$. The following claim is immediate.

**Claim 1.** *For every subset $Q \subseteq S_i$, there is a path $P_Q$ in $X_i$ that begins in the portal* 0 *and uses all the colors from $Q$, each exactly once.*

**Construction and correctness** The graph $G = (V, E)$ is constructed as follows. The vertex set $V$ is partitioned into $r + 2$ layers $V_0, V_1, \ldots, V_{r+1}$. There are two special vertices $s$ and $t$ such that $V_0 = \{s\}$ and $V_{r+1} = \{t\}$. The vertex $t$ is called the portal of $V_{r+1}$. For each $j \in [r]$, the layer $V_j$ has $m$ copies of set gadgets $X_i$, one for each set $S_i \in \mathcal{S}$. For every $j \in \{1, \ldots, r + 1\}$ and for every vertex $v \in V_{j-1}$ we put an edge colored $\alpha_j$ from $v$ to every portal of $V_j$. An example illustrating the construction is shown in Fig. 1. Let us then prove $I$ is a YES-instance of Set Cover iff $I'$ is a YES-instance of Edge-Colorful $st$-Path.

Suppose $I$ is a YES-instance of Set Cover, and let $\mathcal{S}' = \{S_{i_1}, \ldots, S_{i_r}\}$ be some solution, i.e., a subfamily of $r$ sets such that $\bigcup \mathcal{S}' = U$. For every $j = 1, \ldots, r$ define

$$Z_j = S_{i_j} \setminus \bigcup_{\ell < j} S_{i_\ell},$$

i.e., $Z_j$ is the set of new elements covered by $S_{i_j}$, after picking the previous sets. For every $j = 1, \ldots, r$ consider the path $P_{Z_j}$ (defined in Claim 1) in the set gadget $X_{i_j}$ of the layer $V_j$. Clearly, the paths $\{P_{Z_j}\}_{j=1,\ldots,r}$ use all the colors from $U$, each exactly once. Then we join $s, P_{Z_1}, \ldots, P_{Z_r}, t$, in this order, into one path $P$. The connecting $r + 1$ edges use the remaining colors $\alpha_1, \ldots, \alpha_{r+1}$. Hence $P$ is colorful.

For the other direction, suppose $I'$ is a YES-instance of Edge-Colorful $st$-Path, and let $P$ be a colorful path in $G$. Consider a sequence $\alpha = (\alpha_{j_1}, \ldots, \alpha_{j_q})$ of colors from $\{\alpha_1, \ldots, \alpha_{r+1}\}$, that appear on $P$ when going from $s$ to $t$. Recall that for any

$j \in [r]$ vertices of layer $V_j$ are incident with edges of only two colors from $\{\alpha_1, \dots, \alpha_{r+1}\}$, namely $\alpha_j$ and $\alpha_{j+1}$. It follows that for every $\ell$, we have $|j_{\ell+1} - j_\ell| = 1$. Since $j_1 = 1$ and $P$ is colorful, we infer that $\alpha = (\alpha_1, \dots, \alpha_{r+1})$. It follows that $P$ enters each layer $V_j$ exactly once. For every $j \in [r]$, since within layer $V_j$ there are no edges between set gadgets, $P$ visits exactly one set gadget, say $X_{i_j}$ in $V_j$. We claim that $\mathcal{S}' = \{S_{i_1}, \dots, S_{i_r}\}$ is a cover of $U$. Indeed, for every $e \in U$ the path $P$ has an edge $f$ colored with $e$, and hence $f$ belongs to some set gadget $X_{i_j}$, which implies that $e \in S_{i_j}$, as required.

Now let us consider variants of Edge-Colorful $st$-Path, i.e., the problem $A$-Colorful $BC$, where $A \in \{\text{Edge}, \text{Vertex}\}$, $B \in \{st, \varepsilon\}$ and $C \in \{\text{Path}, \text{Walk}\}$.

In the vertex variant, i.e., when $A = \text{Vertex}$, the construction differs only slightly, in particular we use the same graph $G$. The graph $G$ is then colored using $n+r+2$ colors from the set $U \cup \{\alpha_0, \dots, \alpha_{r+1}\}$. Vertex $s$ is colored with $\alpha_0$ and $t$ is colored with $\alpha_{r+1}$. For every $j \in [k]$, for every $i \in [m]$ consider the set gadget $X_i$ in $V_j$, corresponding to a set $S_i = \{e_1, e_2, \dots, e_{|S_i|}\}$. The portal 0 gets color $\alpha_j$, while for each $\ell = 1, \dots, |S_i|$ the vertex $\ell$ of $X_i$ gets color $e_\ell$. The arguments for the equivalence are essentially the same as before.

We note that in the edge variant all edges colored with $\alpha_1$ are incident with $s$ and all edges colored with $\alpha_{r+1}$ are incident with $t$. Since $s$ is incident only with edges colored $\alpha_1$ and $t$ is incident only with edges colored $\alpha_{r+1}$, $G$ has a colorful path iff $G$ has a colorful $st$-path. By a similar argument, the equivalence holds also in the vertex variant. It follows that the result generalizes to $B = \varepsilon$.

Finally, both in edge and vertex variant, $G$ contains a colorful $st$-path iff $G$ contains a colorful $st$-walk, so the result generalizes to $C = \text{Walk}$. $\square$

The following result is given in [6].

**Theorem 7** *(Cygan et al. [6]). If* Set Cover *can be solved in* $(2-\epsilon)^{n+r}(nm)^{O(1)}$ *time for some* $\epsilon > 0$ *then it can also be solved in* $(2-\epsilon')^n(nm)^{O(1)}$ *time, for some* $\epsilon' > 0$.

By combining Lemma 6 with Theorem 7, we have proven the following.

**Theorem 8.** *For every* $\varepsilon > 0$, Edge-Colorful $st$-Path *does not admit a* $(2-\varepsilon)^k n^{O(1)}$*-time algorithm, unless* Set Cover *admits a* $(2-\varepsilon')^r(mn)^{O(1)}$*-time algorithm, for some* $\varepsilon' > 0$*. The same applies to other variants of the problem* Edge-Colorful $st$-Path*, i.e., for any problem* $A$-Colorful $BC$*, where* $A \in \{\text{Edge}, \text{Vertex}\}$, $B \in \{st, \varepsilon\}$*, and* $C \in \{\text{Path}, \text{Walk}\}$.

## 4. Further research

Our result on Colorful Path motivates a related question on Rainbow $st$-Connectivity: can one show that Rainbow $st$-Connectivity does not admit an algorithm running in time $(2-\varepsilon)^k n^{O(1)}$, assuming the Set Cover Conjecture, or some other well-motivated complexity assumption? We leave this as an interesting open problem.

It would be also interesting to identify more natural problems where the goal is to find some structure in a colored graph. We remark here that the technique from Theorem 4 can be used for solving the Colorful Graph Motif problem in $2^k n^{O(1)}$ time and polynomial space *deterministically*. It suffices to replace walks by branching walks (see Nederlof [13] or Björklund et al. [2]).

## References

[1] N. Alon, R. Yuster, U. Zwick, Color-coding, J. ACM 42 (4) (1995) 844–856.

[2] A. Björklund, P. Kaski, L. Kowalik, Probably optimal graph motifs, in: 30th International Symposium on Theoretical Aspects of Computer Science, STACS 2013, February 27, March 2, 2013, Kiel, Germany, in: LIPIcs. Leibniz Int. Proc. Inform., vol. 20, Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2013, pp. 20–31.

[3] A. Björklund, P. Kaski, L. Kowalik, Fast witness extraction using a decision oracle, in: Algorithms – ESA 2014 – Proceedings of the 22th Annual European Symposium, Wroclaw, Poland, September 8–10, 2014, in: Lecture Notes in Comput. Sci., Springer, 2014, pp. 149–160.

[4] S. Chakraborty, E. Fischer, A. Matsliah, R. Yuster, Hardness and algorithms for rainbow connection, J. Comb. Optim. 21 (3) (2009) 330–347.

[5] L. Chen, X. Li, Y. Shi, The complexity of determining the rainbow vertex-connection of a graph, Theoret. Comput. Sci. 412 (35) (2011) 4531–4535.

[6] M. Cygan, H. Dell, D. Lokshtanov, D. Marx, J. Nederlof, Y. Okamoto, R. Paturi, S. Saurabh, M. Wahlstrom, On problems as hard as CNF-SAT, in: IEEE 27th Annual Conference on Computational Complexity, CCC, 2012, pp. 74–84.

[7] M. Cygan, F.V. Fomin, Ł. Kowalik, D. Lokshtanov, D. Marx, M. Pilipczuk, M. Pilipczuk, S. Saurabh, Parameterized Algorithms, Springer, 2015.

[8] R. Diestel, Graph Theory, Springer-Verlag, Heidelberg, 2010.

[9] S. Guillemot, F. Sikora, Finding and counting vertex-colored subtrees, Algorithmica 65 (4) (2013) 828–844.

[10] J. Kleinberg, É. Tardos, Algorithm Design, Pearson Education, 2006.

[11] M. Krivelevich, R. Yuster, The rainbow connection of a graph is (at most) reciprocal to its minimum degree, J. Graph Theory 63 (3) (2010) 185–191.

[12] J. Lauri, Further hardness results on rainbow and strong rainbow connectivity, Discrete Appl. Math. 201 (2016) 191–200.

[13] J. Nederlof, Fast polynomial-space algorithms using inclusion-exclusion, Algorithmica 65 (4) (2013) 868–884.

[14] K. Uchizawa, T. Aoki, T. Ito, A. Suzuki, X. Zhou, On the rainbow connectivity of graphs: complexity and FPT algorithms, Algorithmica 67 (2) (2013) 161–179.

# Paper 4

Eduard Eiben, Robert Ganian, and Juho Lauri.

**On the complexity of rainbow coloring problems.**

4

# On the Complexity of Rainbow Coloring Problems[*]

Eduard Eiben[†] Robert Ganian[‡] Juho Lauri[§]

August 16, 2016

**Abstract**

An edge-colored graph $G$ is said to be *rainbow connected* if between each pair of vertices there exists a path which uses each color at most once. The *rainbow connection number*, denoted by $\mathrm{rc}(G)$, is the minimum number of colors needed to make $G$ rainbow connected. Along with its variants, which consider vertex colorings and/or so-called strong colorings, the rainbow connection number has been studied from both the algorithmic and graph-theoretic points of view.

In this paper we present a range of new results on the computational complexity of computing the four major variants of the rainbow connection number. In particular, we prove that the Strong Rainbow Vertex Coloring problem is NP-complete even on graphs of diameter 3, and also when the number of colors is restricted to 2. On the other hand, we show that if the number of colors is fixed then all of the considered problems can be solved in linear time on graphs of bounded treewidth. Moreover, we provide a linear-time algorithm which decides whether it is possible to obtain a rainbow coloring by saving a fixed number of colors from a trivial upper bound. Finally, we give a linear-time algorithm for computing the exact rainbow connection numbers for three variants of the problem on graphs of bounded vertex cover number.

**Keywords.** Rainbow coloring; Rainbow connection number; Graph algorithms; Treewidth.

## 1 Introduction

The concept of rainbow connectivity was introduced by Chartrand, Johns, McKeon, and Zhang in 2008 [8] as an interesting connectivity measure motivated by recent developments in the area of secure data transfer. Over the past years, this strengthened notion of connectivity has received a significant amount of attention in the research community. The applications of rainbow connectivity are discussed in detail for instance in the recent survey [25], and various bounds are also available in [10, 26].

An edge-colored graph $G$ is said to be *rainbow connected* if between each pair of vertices $a, b$ there exists an $a-b$ path which uses each color at most once; such a path is called *rainbow*. The minimum number of colors needed to make $G$ rainbow connected is called the *rainbow connection number* (rc), and the Rainbow Coloring problem asks to decide if the rainbow

[*]A shortened version of this paper has appeared in the proceedings of the 26th International Workshop on Combinatorial Algorithms (IWOCA) [17].

[†]TU Wien, Vienna, Austria

[‡]TU Wien, Vienna, Austria

[§]Tampere University of Technology, Tampere, Finland connection number is upper-bounded by a number specified in the input. Precise definitions are given in Section 2.

The rainbow connection number and RAINBOW COLORING have been studied from both the algorithmic and graph-theoretic points of view. On one hand, the exact rainbow connection numbers are known for a variety of simple graph classes, such as wheel graphs [8], complete multipartite graphs [8], unit interval graphs [29], and threshold graphs [6]. On the other hand, RAINBOW COLORING is a notoriously hard problem. It was shown by Chakraborty *et al.* [5] that already deciding if $\text{rc}(G) \leq 2$ is $\mathsf{NP}$-complete, and Ananth *et al.* [1] showed that for any $k > 2$ deciding $\text{rc}(G) \leq k$ is $\mathsf{NP}$-complete. In fact, Chandran and Rajendraprasad [6] strengthened this result to hold for chordal graphs. In the same paper, the authors gave a linear time algorithm for rainbow coloring split graphs which form a subclass of chordal graphs with at most one more color than the optimum. Basavaraju *et al.* [2] gave an $(r+3)$-factor approximation algorithm to rainbow color a general graph of radius $r$. Later on, the inapproximability of the problem was investigated by Chandran and Rajendraprasad [7]. They proved that there is no polynomial time algorithm to rainbow color graphs with less than twice the minimum number of colors, unless $\mathsf{P} = \mathsf{NP}$. For chordal graphs, they gave a 5/2-factor approximation algorithm, and proved that it is impossible to do better than 5/4, unless $\mathsf{P} = \mathsf{NP}$.

Several variants of the notion of rainbow connectivity have also been considered. Indeed, a similar concept was introduced for vertex-colored graphs by Krivelevich and Yuster [22]. A vertex-colored graph $H$ is *rainbow vertex connected* if there is a path whose internal vertices have distinct colors between every pair of vertices, and this gives rise to the *rainbow vertex connection number* (rvc). The *strong rainbow connection number* (src) was introduced and investigated also by Chartrand *et al.* [10]; an edge-colored graph $G$ is said to be *strong rainbow connected* if between each pair of vertices $a, b$ there exists a shortest $a - b$ path which is rainbow. The combination of these two notions, *strong rainbow vertex connectivity* (srvc), was studied in a graph theoretic setting by Li *et al.* [24].

Not surprisingly, the problems arising from the strong and vertex variants of rainbow connectivity are also hard. Chartrand *et al.* showed that $\text{rc}(G) = 2$ if and only if $\text{src}(G) = 2$ [8], and hence deciding if $\text{src}(G) \leq k$ is $\mathsf{NP}$-complete for $k = 2$. The problem remains $\mathsf{NP}$-complete for $k > 2$ for bipartite graphs [1], and also for split graphs [21]. Furthermore, the strong rainbow connection number of an $n$-vertex bipartite graph cannot be approximated within a factor of $n^{1/2-\epsilon}$, where $\epsilon > 0$ unless $\mathsf{NP} = \mathsf{ZPP}$ [1], and the same holds for split graphs [21]. The computational aspects of the rainbow vertex connection numbers have received less attention in the literature. Through the work of Chen *et al.* [12] and Chen *et al.* [11], it is known that deciding if $\text{rvc}(G) \leq k$ is $\mathsf{NP}$-complete for every $k \geq 2$. However, to the best of our knowledge, the complexity of deciding whether $\text{srvc}(G) \leq k$ (the $k$-SRVC problem) has not been previously considered.

In this paper, we present new positive and negative results for all four variants of the rainbow coloring problems discussed above.

- In Section 3, we prove that $k$-SRVC is $\mathsf{NP}$-complete for every $k \geq 3$ even on graphs of diameter 3. Our reduction relies on an intermediate step which proves the $\mathsf{NP}$-hardness of a more general problem, the $k$-SUBSET STRONG RAINBOW VERTEX COLORING problem. We also provide bounds for approximation algorithms (under established complexity assumptions), see Corollary 6, and tighten the hardness result to additionally cover 2-SRVC.

- In Section 4, we show that all of the considered problems can be formulated in monadic second order (MSO) logic. In particular, this implies that for every fixed $k$, all of the considered problems can be solved in linear time on graphs of bounded treewidth, and the vertex variants can be solved in cubic time on graphs of bounded clique-width.
- In Section 5, we investigate the problem from a different perspective: we ask whether, given an $n$-vertex graph $G$ and an integer $k$, it is possible to color $G$ using $k$ colors less than the known upper bound. Here we employ a win-win approach and show that this problem can be solved in time $\mathcal{O}(n)$ for any fixed $k$.
- In the final Section 6, we show that in the general case when $k$ is not fixed, three of the considered problems admit linear-time algorithms on graphs of bounded vertex cover number. This is also achieved by exploiting a win-win approach, where we show that either $k$ is bounded by a function of the vertex cover number and hence we can apply the result of Section 4, or $k$ is sufficiently large which allows us to exploit the structure of the graph and solve the problem directly.

## 2 Preliminaries

### 2.1 Graphs and Rainbow Connectivity

We refer to [15] for standard graph-theoretic notions. We use $[i]$ to denote the set $\{1, 2, \ldots, i\}$. All graphs considered in this paper are simple and undirected. The *degree* of a vertex is the number of its incident edges, and a vertex is a *pendant* if it has degree 1. We will often use the shorthand $ab$ for the edge $\{a, b\}$. For a vertex set $X$, we use $G[X]$ to denote the subgraph of $G$ induced on $X$.

A *vertex coloring* of a graph $G = (V, E)$ is a mapping from $V$ to $\mathbb{N}$, and similarly an *edge coloring* of $G$ is a mapping from $E$ to $\mathbb{N}$; in this context, we will often refer to the elements of $\mathbb{N}$ as colors. An *$a - b$ path $P$ of length $p$* is a finite sequence of the form $(a = v_0, e_0, v_1, e_1, \ldots b = v_p)$, where $v_0, v_1, \ldots v_p$ are distinct vertices and $e_0, \ldots e_{p-1}$ are distinct edges and each edge $e_j$ is incident to $v_j$ and $v_{j+1}$. An $a - b$ path of length $p$ is a *shortest path* if every $a - b$ path has length at least $p$. The *diameter* of a graph $G$ is the length of its longest shortest path, denoted by $\operatorname{diam}(G)$. Given an edge (vertex) coloring $\alpha$ of $G$, a color $x \in \mathbb{N}$ *occurs* on a path $P$ if there exists an edge (an internal vertex) $z$ on $P$ such that $\alpha(z) = x$.

A vertex or edge coloring of $G$ is *rainbow* if between each pair of vertices $a, b$ there exists an $a - b$ path $P$ such that each color occurs at most once on $P$; in this case we say that $G$ is *rainbow connected* or *rainbow colored.* We denote by $\operatorname{rc}(G)$ the minimum $i \in \mathbb{N}$ such that there exists a rainbow edge coloring $\alpha : E \to [i]$. Similarly, $\operatorname{rvc}(G)$ denotes the minimum $i \in \mathbb{N}$ such that there exists a rainbow vertex coloring $\alpha : V \to [i]$. Furthermore, an edge or vertex coloring of $G$ is a *strong rainbow coloring* if between each pair of vertices $a, b$ there exists a shortest $a - b$ path $P$ such that each color occurs at most once on $P$. We denote by $\operatorname{src}(G)$ ($\operatorname{srvc}(G)$) the minimum $i \in \mathbb{N}$ such that there exists a strong rainbow edge (vertex) coloring $\alpha : E \to [i]$ ($\alpha : V \to [i]$).

Let $G$ and $H$ be two graphs with $n$ and $n'$ vertices, respectively. The *corona* of $G$ and $H$, denoted by $G \circ H$, is the disjoint union of $G$ and $n$ copies of $H$ where the $i$-th vertex of $G$ is connected by an edge to every vertex of the $i$-th copy of $H$. Clearly, the corona $G \circ H$ has $n(1 + n')$ vertices. Coronas of graphs were first studied by Frucht and Harary [18].

## 2.2 Problem Statements

Here we formally state the problems studied in this work.

Rainbow $k$-Coloring ($k$-RC)
**Instance:** A connected undirected graph $G = (V, E)$.
**Question:** Is $\text{rc}(G) \leq k$?

Strong Rainbow $k$-Coloring ($k$-SRC), Rainbow Vertex $k$-Coloring ($k$-RVC) and Strong Rainbow Vertex $k$-Coloring ($k$-SRVC) are then defined analogously for $\text{src}(G)$, $\text{rvc}(G)$, and $\text{srvc}(G)$, respectively. We also consider generalized versions of these problems, where $k$ is given as part of the input.

Rainbow Coloring (RC)
**Instance:** A connected undirected graph $G = (V, E)$, and a positive integer $k$.
**Question:** Is $\text{rc}(G) \leq k$?

The problems SRC, RVC, and SRVC are also defined analogously. In Section 5 we consider the "saving" versions of the problem, which ask whether it is possible to improve upon the trivial upper bound for the number of colors.

Saving $k$ Rainbow Colors ($k$-SavingRC)
**Instance:** A connected undirected graph $G = (V, E)$.
**Question:** Is $\text{rc}(G) \leq |E| - k$?

Saving $k$ Rainbow Vertex Colors ($k$-SavingRVC)
**Instance:** A connected undirected graph $G = (V, E)$.
**Question:** Is $\text{rvc}(G) \leq |V| - k$?

## 2.3 Structural Measures

Several of our results utilize certain structural measures of graphs. We will mostly be concerned with the *treewidth* and the *vertex cover number* of the input graph. Section 4 also mentions certain implications of our results for graphs of bounded *clique-width*, the definition of which can be found for instance in [14].

A *tree decomposition* of $G$ is a pair $(T, \{X_i : i \in I\})$ where $X_i \subseteq V$, $i \in I$, and $T$ is a tree with elements of $I$ as nodes such that:

1. for each edge $uv \in E$, there is an $i \in I$ such that $\{u, v\} \subseteq X_i$, and
2. for each vertex $v \in V$, $T[\{\, i \in I \mid v \in X_i \,\}]$ is a (connected) tree with at least one node.

The *width* of a tree decomposition is $\max_{i \in I} |X_i| - 1$. The *treewidth* [28] of $G$ is the minimum width taken over all tree decompositions of $G$ and it is denoted by $\text{tw}(G)$.

**Fact 1** ([4]). *There exists an algorithm which, given a graph $G$ and an integer $p$, runs in time $2^{p^{\mathcal{O}(1)}} \cdot (|V(G)| + |E(G)|)$, and either outputs a tree decomposition of $G$ of width at most $p$ or correctly determines that $\text{tw}(G) > p$.*

A *vertex cover* of a graph $G = (V, E)$ is a set $X \subseteq V$ such that each edge in $G$ has at least one endvertex in $X$. The cardinality of a minimum vertex cover in $G$ is denoted as $\tau(G)$. Given a vertex cover $X$, a *type* $T$ is a subset of $V \setminus X$ such that any two vertices in $T$ have the same neighborhood; observe that any graph contains at most $2^{|X|}$ many distinct types.

### 2.4 Monadic Second Order Logic

We assume that we have an infinite supply of individual variables, denoted by lowercase letters $x, y, z$, and an infinite supply of set variables, denoted by uppercase letters $X, Y, Z$. *Formulas* of MSO$_2$ logic are constructed from atomic formulas $I(x, y)$, $x \in X$, and $x = y$ using the connectives $\neg$ (negation), $\wedge$ (conjunction) and existential quantification $\exists x$ over individual variables as well as existential quantification $\exists X$ over set variables. Individual variables range over vertices and edges, and set variables range either over sets of vertices or over sets of edges. The atomic formula $I(x, y)$ expresses that vertex $x$ is incident to edge $y$, $x = y$ expresses equality, and $x \in X$ expresses that $x$ is in the set $X$. From this, we define the semantics of MSO$_2$ logic in the standard way.

MSO$_1$ logic is defined similarly as MSO$_2$ logic, with the following distinctions. Individual variables range only over vertices, and set variables only range over sets of vertices. The atomic formula $I(x, y)$ is replaced by $E(x, y)$, which expresses that vertex $x$ is adjacent to vertex $y$.

*Free and bound variables* of a formula are defined in the usual way. A *sentence* is a formula without free variables. It is known that MSO$_2$ formulas can be checked efficiently as long as the graph has bounded tree-width.

**Fact 2** ([13]). *Let $\phi$ be a fixed* MSO$_2$ *sentence and $p \in \mathbb{N}$ be a constant. Given an $n$-vertex graph $G$ of treewidth at most $p$, it is possible to decide whether $G \models \phi$ in time $\mathcal{O}(n)$.*

Similarly, MSO$_1$ formulas can be checked efficiently as long as the graph has bounded *clique-width* [14] (or, equivalently, *rank-width* [19]). In particular, while the formula can be checked in linear time if a suitable rank- or clique-decomposition is provided, current algorithms for finding (or approximating) such a decomposition require cubic time.

**Fact 3** ([14,19]). *Let $\phi$ be a fixed* MSO$_1$ *sentence and $p \in \mathbb{N}$ be a constant. Given an $n$-vertex graph $G$ of clique-width at most $p$, it is possible to decide whether $G \models \phi$ in time $\mathcal{O}(n^3)$.*

## 3 Hardness of Strong Rainbow Vertex $k$-Coloring

It is easy to see that $\text{srvc}(G) = 1$ if and only if $\text{diam}(G) = 2$. We will prove that deciding if $\text{srvc}(G) \leq k$ is NP-complete for every $k \geq 3$ already for graphs of diameter 3. This is done by first showing hardness of an intermediate problem, described below. Later on, we will show that deciding $\text{srvc}(G) \leq 2$ is in fact also NP-complete.

In the $k$-Subset Strong Rainbow Vertex Coloring problem ($k$-SSRVC) we are given a graph $G$ which is a corona of a complete graph and $K_1$, and a set $P$ of pairs of pendants in $G$. The goal is to decide if the vertices of $G$ can be colored with $k$ colors such that each pair in $P$ is connected by a vertex rainbow shortest path. We will first show this intermediate problem is NP-complete by reducing from the classical *vertex $k$-coloring problem*: given a graph $G$, decide if there is an assignment of $k$ colors to the vertices of $G$ such that adjacent vertices receive a different color. The smallest $k$ for which this is possible is known as the *chromatic number* of $G$. The vertex $k$-coloring problem is well-known to be $\mathsf{NP}$-complete for every $k \geq 3$.

**Lemma 4.** *The $k$-SSRVC problem is $\mathsf{NP}$-complete for every $k \geq 3$.*

*Proof.* Let $G = (V, E)$ be an instance of the vertex $k$-coloring problem, where $k \geq 3$. We will construct an instance $\langle G', P\rangle$ of the $k$-SSRVC problem such that $\langle G', P\rangle$ is a YES-instance if and only if $G$ is vertex $k$-colorable.

The graph $G' = (V', E')$ along with the set of pairs $P$ are constructed as follows:

- $V' = V \cup \{p_v \mid v \in V\}$,
- $E' = \{uv \mid u, v \in V \wedge u \neq v\} \cup \{vp_v \mid v \in V\}$, and
- $P = \{\{p_u, p_v\} \mid uv \in E\}$.

Clearly, $G' = K_{|V|} \circ K_1$. This completes the construction of $G'$.

Suppose $G$ is vertex $k$-colorable. Let $c$ be the color assigned to vertex $v$ in $V$. We assign the color $c$ to both $v$ and $p_v$ in $G'$. Observe that the shortest path between any pair of vertices in $G'$ is unique. It is then straightforward to verify that any pair in $P$ is strong rainbow vertex connected.

For the other direction, suppose there is a vertex coloring of $G'$ using $k$ colors under which there is a vertex rainbow shortest path between every pair in $P$. Since any two vertices $\{p_u, p_v\} \in P$ are strong rainbow vertex connected, the two internal vertices on the unique $p_u - p_v$ shortest path have distinct colors. Thus by assigning to the vertex $v \in V$ the color on the corresponding vertex $v' \in (V' \setminus \{p_v \mid v \in V\})$ we get a proper vertex coloring of $G$. This completes the proof. □

We are now ready to prove the following.

**Theorem 5.** *The $k$-SRVC problem is $\mathsf{NP}$-complete for every integer $k \geq 3$, even when the input is restricted to graphs of diameter 3.*

*Proof.* Let $k \geq 3$ and $\langle G = (V, E), P\rangle$ be an instance of the $k$-SSRVC problem. We will construct a graph $G' = (V', E')$ that is strong rainbow vertex colorable with $k$ colors if and only if $\langle G = (V, E), P\rangle$ is a YES-instance of $k$-SSRVC.

Let $V_1$ denote the set of pendant vertices in $G$. For every vertex $v \in V_1$ we introduce a new vertex $x_v$. For every pair of pendant vertices $\{u, v\} \notin P$, we add two vertices $x^1_{uv}$ and $x^2_{uv}$. We also add two new vertices $s$ and $t$. In the following, we denote by $k_v$, where $v \in V_1$, the unique vertex that $v$ is adjacent to in $G$. In addition, for a set $A$, we write $\binom{A}{2}$ to denote the Cartesian product $A \times A$. Formally, we construct a graph $G' = (V', E')$ such that:

- $V' = V \cup \{x_v \mid v \in V_1\} \cup \{x^1_{uv}, x^2_{uv} \mid \{u, v\} \in \binom{V_1}{2} \setminus P\} \cup \{s, t\}$,
- $E' = E \cup E_1 \cup E_2 \cup E_3 \cup E_4$,
- $E_1 = \{vx_v, sx_v, tx_v \mid v \in V_1\}$,
- $E_2 = \{ux^1_{uv}, x^1_{uv}x^2_{uv}, x^2_{uv}v \mid \{u, v\} \in \binom{V_1}{2} \setminus P\}$,
- $E_3 = \{sx^1_{uv}, tx^2_{uv}, k_u x^1_{uv}, k_v x^2_{uv} \mid \{u, v\} \in \binom{V_1}{2} \setminus P\}$, and

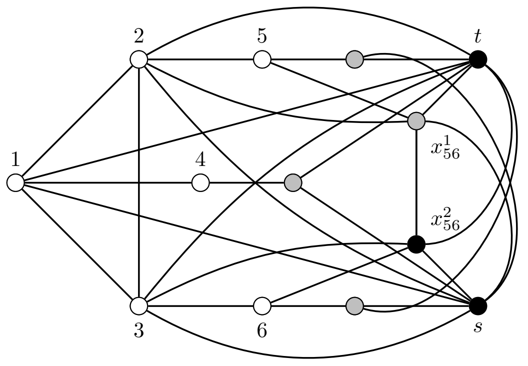


Figure 1: The graph $K_3 \circ K_1$ transformed to a graph of diameter 3 with $P = \{\{4,5\},\{4,6\}\}$. The color $c_1$ is represented with grey, and the color $c_2$ with black. White vertices represent an unknown vertex coloring under which the pairs in $P$ are strong rainbow vertex connected.

- $E_4 = \{sy, ty \mid y \in V \setminus V_1\}$.

This completes the construction of $G'$. It is straightforward to verify $\text{diam}(G') = 3$, and this is realized between any pair of vertices in $V_1$. An example illustrating the reduction is shown in Figure 1.

First, suppose $G'$ admits a strong rainbow vertex coloring $\phi$ using $k$ colors. Observe that for each $\{u,v\} \in P$, the shortest path between $u$ and $v$ in $G'$ is unique. Therefore $k_u$ and $k_v$ must receive distinct colors by $\phi$. Hence the restriction of $\phi$ to $V$ witnesses that $\langle G, P\rangle$ is a YES-instance of $k$-SSRVC.

On the other hand, suppose $\langle G, P\rangle$ is $k$-subset strong rainbow vertex connected under some coloring $\phi : V \to \{c_1, \ldots, c_k\}$. We will describe an extended vertex $k$-coloring $\phi'$ under which $G'$ is strong rainbow vertex connected. We retain the original coloring on the vertices of $G$, i.e., $\phi'(v) = \phi(v)$ for every $v \in V$. The remaining vertices in $G'$ receive colors as follows:

- $\phi'(x_v) = c_1$, for every $v \in V_1$,
- $\phi'(x^1_{uv}) = c_1$, $\phi'(x^2_{uv}) = c_2$, for every $\{u,v\} \in \binom{V_1}{2} \setminus P$, and
- $\phi'(s) = c_2$, and $\phi'(t) = c_2$.

Since each pair of vertices $\{a,b\} \in V'$ at distance at most 2 are always strong rainbow vertex connected regardless of the chosen coloring, and each pair of vertices $\{u,v\} \in \binom{V_1}{2} \setminus P$ are connected by the path through $x^1_{uv}, x^2_{uv}$, it is straightforward to verify that $G'$ is indeed strong rainbow vertex connected under $\phi'$. □

It can be observed that the size of the above reduction does not depend on $k$, the number of colors. In fact, if the instance of the vertex $k$-coloring problem has $n$ vertices, then the graph $G'$ we build in Theorem 5 has no more than $\mathcal{O}(n^2)$ vertices. Furthermore, a strong rainbow vertex coloring of $G'$ gives us a solution to the vertex $k$-coloring problem. Since the

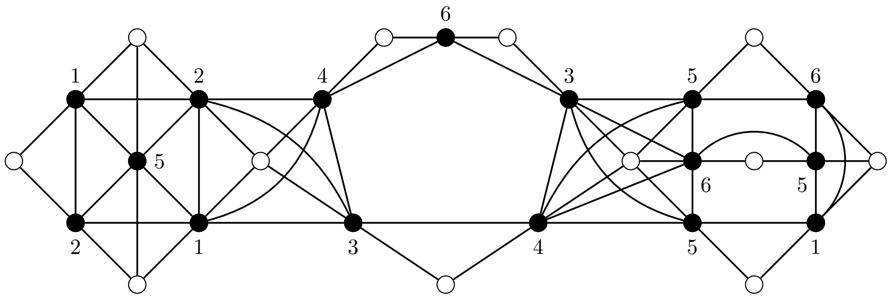


Figure 2: A 12-vertex graph $G$ after the construction of Lemma 7. The white vertices correspond to the original 12 vertices of $G$, and the black vertices to the subdivided edges. An optimal strong rainbow coloring with 6 colors is shown.

chromatic number of an $n$-vertex graph cannot be approximated within a factor of $n^{1-\epsilon}$ for any $\epsilon > 0$ unless $\mathsf{P} = \mathsf{NP}$ [31], we obtain the following corollary.

**Corollary 6.** *There is no polynomial time algorithm for approximating the strong rainbow vertex connection number of an $n$-vertex graph of bounded diameter within a factor of $n^{1/2-\epsilon}$ for any $\epsilon > 0$, unless $\mathsf{P} = \mathsf{NP}$.*

Each of $k$-RC, $k$-SRC, and $k$-RVC is known to be $\mathsf{NP}$-complete for every $k \geq 2$. In this light, Theorem 5 raises a natural question: is $k$-SRVC also $\mathsf{NP}$-complete already for $k = 2$? Indeed, the following lemma will establish that this is the case. In contrast to Theorem 5, we employ a more direct reduction from $k$-SRC.

**Lemma 7.** *There exists a polynomial time algorithm which, given an instance $G = (V, E)$ of $k$-SRC, creates an instance $G' = (V', E')$ of $k$-SRVC such that $G$ is a YES-instance of $k$-SRC if and only if $G'$ is a YES-instance of $k$-SRVC.*

*Proof.* Let $G = (V, E)$ be an instance of the $k$-SRC problem for any $k \geq 2$. In polynomial time, we will construct an instance $G' = (V', E')$ of the $k$-SRVC problem such that $\mathrm{src}(G) = k$ if and only if $\mathrm{srvc}(G') = k$.

The graph $G'$ is obtained from $G$ by subdividing each edge $e \in E$ by a new vertex $w_e$. Then, for every $e, f \in E$ such that $e \neq f$, we make $w_e$ and $w_f$ adjacent in $G'$ if the edges $e$ and $f$ were adjacent in $G$. Formally, we let $V' = V \cup W$, where $W = \{w_e \mid e \in E\}$, and $E' = \{aw_e, bw_e \mid ab = e \in E\} \cup \{w_e w_f \mid e, f \text{ are adjacent in } G\}$. This completes the construction of $G'$, which is illustrated in Figure 2. Let us then prove that $\mathrm{src}(G) = k$ if and only if $\mathrm{srvc}(G') = k$.

Suppose $\mathrm{src}(G) = k$, and consider a edge coloring $c : E \to [k]$ under which $G$ is strong rainbow connected. We construct a vertex $k$-coloring $c' : V' \to [k]$ such that $c'(w_{ij}) = c(ij)$, for every $ij \in E$. The remaining vertices receive an arbitrary color, say $c'(v) = 1$ for every $v \in V$. We claim that any two vertices $u$ and $v$ are strong rainbow vertex connected under $c'$ in $G'$. There are three cases to consider: both $u$ and $v$ are in $V$, neither $u$ nor $v$ is in $V$, and exactly one of $u$ and $v$ is in $V$. We will show this only for the first case, as the remaining cases follow easily by a similar argument. Without loss of generality, suppose $u$ and $v$ are strong rainbow connected in $G$ via the edges $e_1, \ldots, e_\ell$, where $\ell \leq k$. Then, it is easy to see the path $u, w_1, \ldots, w_\ell, v$ is a rainbow vertex shortest path in $G'$. A strong rainbow coloring of $G$ transformed into a strong rainbow vertex coloring of $G'$ is shown in Figure 2.

For the other direction, suppose there is a vertex $k$-coloring $c'$ under which $G'$ is strong rainbow vertex connected. We construct an edge $k$-coloring $c : E \to [k]$ for $G$ such that $c(ij) = c'(w_{ij})$, for every $ij \in E$. Observe that for each nonadjacent $u, v \in V(G')$ and each shortest $u$-$v$ path $P_{uv}$, it holds that every internal vertex of $P_{uv}$ is in $W$. Thus, because $c'$ is a strong rainbow vertex coloring of $G'$, we have that $c$ is a strong rainbow coloring of $G$. This completes the proof. □

By the above, we strengthen Theorem 5 to cover the case $k = 2$.

**Corollary 8.** *The $k$-SRVC problem is NP-complete for every integer $k \geq 2$.*

In fact, we remark that Lemma 7 holds also for the non-strong variants of the problems. That is, the lemma establishes also a reduction from $k$-RC to $k$-RVC. In the literature, the hardness of $k$-RVC was established in two stages: Chen *et al.* [12] first showed 2-RVC is NP-complete by a chain of reductions originating from 3-SAT. Then, by reducing from 2-RVC, Chen *et al.* [11] proved that $k$-RVC is NP-complete for every $k \geq 2$. Conceptually, Lemma 7 is a considerable simplification over this chain of reductions. Moreover, it was shown by Chandran and Rajendraprasad [7] that the rainbow connection number of a graph cannot be approximated within a factor of less than 2 unless P = NP. Thus, by combining our remark on Lemma 7 with an argument similar to Corollary 6, we obtain the following.

**Corollary 9.** *There is no polynomial time algorithm for approximating the rainbow vertex connection number of a graph within a factor less than 2 unless* P = NP.

## 4 MSO Formulations

This section will present formulations of the $k$-coloring variants of rainbow connectivity in MSO logic, along with their algorithmic implications.

**Lemma 10.** *For every $k \in \mathbb{N}$ there exists a $\mathrm{MSO}_1$ formula $\phi_k$ such that for every graph $G$, it holds that $G \models \phi_k$ iff $G$ is a YES-instance of $k$-RVC. Similarly, for every $k \in \mathbb{N}$ there exists a $\mathrm{MSO}_2$ formula $\psi_k$ such that for every graph $G$, it holds that $G \models \psi_k$ iff $G$ is a YES-instance of $k$-RC.*

*Proof.* In the case of $k$-RC, we wish to partition the edges of the graph $G = (V, E)$ into $k$ color classes $C_1, \ldots, C_k$ such that each pair of vertices is connected by a rainbow path. Let us consider the following $\mathrm{MSO}_2$ formula $\psi_k$.

$$\begin{aligned}\psi_k :=\ & \exists C_1, \ldots, C_k \subseteq E\Big(\forall e \in E\Big(e \in C_1 \vee \cdots \vee e \in C_k\Big)\Big)\\ & \wedge \Big(\forall i, j \in [k], i \neq j : (C_i \cap C_j = \emptyset)\Big)\\ & \wedge \Big(\forall u, v \in V\Big((u \neq v) \implies \bigvee_{1 \leq i \leq k} \Big(\exists e_1, \ldots, e_i \in E\big(\mathrm{Path}(u, v, e_1, \ldots, e_i)\\ & \wedge \mathrm{Rainbow}(e_1, \ldots, e_i)\big)\Big)\Big)\Big),\end{aligned}$$

where the auxiliary predicates are defined as

$$\begin{aligned}\mathrm{Path}(u, v, e_1, \ldots, e_\ell) \coloneqq \exists v_1, \ldots, v_{\ell-1} \in V \\ \Big(\forall i, j \in [\ell-1], i \neq j : (v_i \neq v_j)\Big) \\ \wedge \mathrm{I}(e_1, u) \wedge \mathrm{I}(e_\ell, v) \\ \wedge \Big(\forall i \in [\ell-1] : (\mathrm{I}(e_i, v_i) \wedge \mathrm{I}(e_{i+1}, v_i))\Big)\end{aligned}$$

and

$$\mathrm{Rainbow}(e_1, \ldots, e_\ell) \coloneqq \forall i \in [\ell]\ \exists j \in [k] : \Big(e_i \in C_j \wedge (\forall p \neq i : e_p \notin C_j)\Big).$$

Here, $\mathrm{Path}(u, v, e_1, \ldots, e_\ell)$ expresses that the edges $e_1, \ldots, e_\ell$ form a path between the vertices $u$ and $v$. The predicate $\mathrm{Rainbow}(e_1, \ldots, e_\ell)$ expresses that the edges $e_1, \ldots, e_\ell$ are each in precisely one color class.

In the case of $k$-RVC, the MSO$_1$ formula $\phi_k$ is defined analogously, with the following distinctions:

1. instead of edges, we partition the vertices of $G$ into color classes;
2. the predicate Path speaks of vertices instead of edges and uses the adjacency relation instead of the incidence relation; and
3. the predicate Rainbow tests the coloring of vertices instead of edges. □

Using a similar approach, we obtain an analogous result for the strong variants of these problems.

**Lemma 11.** *For every $k \in \mathbb{N}$ there exists a MSO$_1$ formula $\phi_k$ such that for every graph $G$, it holds that $G \models \phi_k$ iff $G$ is a YES-instance of $k$-*SRVC*. Similarly, for every $k \in \mathbb{N}$ there exists a MSO$_2$ formula $\psi_k$ such that for every graph $G$, it holds that $G \models \psi_k$ iff $G$ is a YES-instance of $k$-*SRC*.*

*Proof.* In the case of $k$-SRC, we wish to partition the edges of the graph $G = (V, E)$ into $k$ color classes $C_1, \ldots, C_k$ such that each pair of vertices is connected by a rainbow shortest path. We will assume the predicates $\mathrm{Path}(u, v, e_1, \ldots, e_\ell)$ and $\mathrm{Rainbow}(e_1, \ldots, e_\ell)$ are defined precisely as in Lemma 10.

Let us then construct the following MSO$_2$ formula $\psi_k$:

$$\begin{aligned}\psi_k \coloneqq\ & \exists C_1, \ldots, C_k \subseteq E\Big(\forall e \in E\Big(e \in C_1 \vee \cdots \vee e \in C_k\Big)\Big) \\ & \wedge \Big(\forall i, j \in [k], i \neq j : (C_i \cap C_j = \emptyset)\Big) \\ & \wedge \Big(\forall u, v \in V\Big((u \neq v) \implies \Big(\exists i \in [k]\ \exists e_1, \ldots, e_i \in E\big(\mathrm{Path}(u, v, e_1, \ldots, e_i) \\ & \wedge \mathrm{Rainbow}(e_1, \ldots, e_i) \\ & \wedge \forall j \in [i-1]\ \neg(\exists w_1, \ldots, w_j \in E\ \ \mathrm{Path}(u, v, w_1, \ldots, w_j))\big)\Big)\Big)\Big).\end{aligned}$$

To capture the property of being a shortest path, we require there to be a $u-v$ path of length $i$, and no paths of length less than $i$. Furthermore, observe that no path of length greater than $k$ can be rainbow. The construction for $k$-SRVC then uses the same ideas, with the same distinctions as those specified in Lemma 10. □

**Theorem 12.** *Let $p \in \mathbb{N}$ be fixed. Then the problems $k$-RC, $k$-SRC, $k$-RVC, and $k$-SRVC can be solved in time $\mathcal{O}(n)$ on $n$-vertex graphs of treewidth at most $p$. Furthermore, $k$-RVC and $k$-SRVC can be solved in time $\mathcal{O}(n^3)$ on $n$-vertex graphs of clique-width at most $p$.*

*Proof.* The proof follows from Lemma 10 and Lemma 11 in conjunction with Fact 2 and Fact 3. □

In the language of parameterized complexity [16,27], Theorem 12 implies that these problems are fixed-parameter tractable (FPT) parameterized by treewidth, and their vertex variants are FPT parameterized by clique-width.

## 5 The Complexity of Saving Colors

This section focuses on the saving versions of the rainbow coloring problems introduced in Subsection 2.2, and specifically gives linear-time algorithms for $k$-SavingRC and $k$-SavingRVC. Our results make use of the following facts.

**Fact 13** ([20]). *There is a $\mathrm{MSO}_1$ predicate* VertexConnects *such that on a graph $G = (V, E)$* VertexConnects$(S, u, v)$ *is true iff $S \subseteq V$ is a set of vertices of $G$ such that there is a path from $u$ to $v$ that lies entirely in $S$.*

The above is easily modified to give us the following.

**Fact 14.** *There is a $\mathrm{MSO}_2$ predicate* EdgeConnects *such that on a graph $G = (V, E)$* EdgeConnects$(X, u, v)$ *is true iff $X \subseteq E$ is a set of edges of $G$ such that there is path from $u$ to $v$ that lies entirely in $X$.*

**Theorem 15.** *For each $k \in \mathbb{N}$, the problem $k$-SavingRC can be solved in time $\mathcal{O}(n)$ on $n$-vertex graphs.*

*Proof.* Observe that by coloring each edge of a spanning tree of $G$ with a distinct color we have that $\mathrm{rc}(G) \leq n - 1$. Thus, if $m \geq n + k - 1$, we have a YES-instance of $k$-SavingRC. Otherwise, suppose $m < n + k - 1$. Then $G$ has a feedback edge set of size at most $k - 1$, and hence, $G$ has treewidth at most $k$. Furthermore, we assume that $m > 2k$, since otherwise the instance can be solved by brute force in time independent of $n$. We construct a $\mathrm{MSO}_2$ formula $\psi_k$ such that it holds that $G \models \psi_k$ is true iff $G$ is a YES-instance of $k$-SavingRC. Using Fact 14, we construct $\psi_k$ as follows:

$$\begin{aligned}
\psi_k := {} & \exists R_1, \ldots, R_k \subseteq E\Big(\forall i, j \in [k], i \neq j : (R_i \cap R_j = \emptyset)\Big) \\
& \wedge \Big(\forall i \in [k] : \big(\exists e \in E(e \in R_i)\big)\Big) \wedge |R_1 \cup R_2 \cup \cdots \cup R_k| \geq 2k \\
& \wedge \Big(\forall u, v \in V\Big((u \neq v) \implies \Big(\exists X \subseteq E\Big(\text{EdgeConnects}(X, u, v) \\
& \wedge \forall e_1, e_2 \in X\Big(\forall i \in [k] : (e_1 \in R_i \wedge e_2 \in R_i) \implies (e_1 = e_2)\Big)\Big)\Big)\Big)\Big).
\end{aligned}$$

In the above, the expression $|A| \geq 2k$ is shorthand for the existence of $2k$ pairwise-distinct edges in $A$, which can be expressed by a simple but lengthy $\mathrm{MSO}_2$ expression. The formula $\psi_k$ expresses that there exist $k$ disjoint sets $R_1, \ldots, R_k$ of edges (each representing a different color set with at least 1 edge) such that their union contains at least $2k$ edges, with the following property: there is a path using at most one edge from each set $R_1, \ldots, R_k$ between every pair of vertices. Formally, this property is stated as the existence of an edge-set $X$ for each pair of vertices $u, v$ such that the graph $(V, X)$ contains an $u-v$ path that cannot repeat edges from any $R_i$.

Let us argue that $G \models \psi_k$ is true iff $G$ is a YES-instance of $k$-SavingRC. Assume $G$ contains sets $R_1, \ldots, R_k$ as per $\psi_k$; then assigning a unique color to each $R_i$ and a unique additional color to each edge in $E \setminus (R_1 \cup \cdots \cup R_k)$ results in a rainbow coloring of $G$ which uses at most $m-k$ colors. On the other hand, in any rainbow $(m-k)$-coloring of $G$, there must exist $j$ reappearing colors in $G$ (where $2 \leq j \leq k$) and the number of edges with these $j$ colors is at least $j+k$. Let us consider an assignment of edges to $R_1, \ldots, R_j$ such that $R_i$ receives all edges with reappearing color $i$; for the remaining $k-j$ sets $R$, we then use arbitrarily selected edges (i.e., they represent arbitrary non-reappearing colors). Since we started with a rainbow coloring, rows 3 and 4 of $\psi_k$ must be satisfied, and since $|R_1, \ldots, R_j| \geq k+j$ it follows that row 2 must also be satisfied.

By the above, it indeed holds that $G \models \psi_k$ is true iff $G$ is a YES-instance of $k$-SavingRC. The proof then follows by Fact 2. $\square$

To prove a similar result for $k$-SavingRVC, we will use the following result.

**Fact 16** ([3]). *If the treewidth of a connected graph $G$ is at least $2k^3$, then $G$ has a spanning tree with at least $k$ vertices with degree 1.*

**Theorem 17.** *For each $k \in \mathbb{N}$, the problem $k$-SavingRVC can be solved in time $\mathcal{O}(n)$ on $n$-vertex graphs.*

*Proof.* Using Fact 1, we will test if the treewidth of $G$ is at least $2k^3$. If it is, then by Fact 16 the graph $G$ has a spanning tree with at least $k$ vertices of degree 1. Each of these $k$ vertices can receive the same color, and we conclude we have a YES-instance. Otherwise, suppose the treewidth of $G$ is less than $2k^3$, and we construct a MSO$_1$ formula $\phi_k$ such that it holds that $G \models \phi_k$ is true iff $G$ is a YES-instance of $k$-SavingRVC. The construction is analogous to Theorem 15, but instead of EdgeConnects we use VertexConnects from Fact 13. The proof then follows by Fact 2. $\square$

## 6 Rainbow Coloring Graphs with Small Vertex Covers

In this section we turn our attention to the more general problem of determining whether the rainbow connection number is below a number specified in the input. Specifically, we show that RC, RVC, and SRVC admit linear time algorithms on graphs of bounded vertex cover number. In particular, this implies that RC, RVC, and SRVC are FPT parameterized by $\tau(G)$.

**Lemma 18.** *Let $G = (V, E)$ be a connected graph and $p = \tau(G)$. Then $\operatorname{rvc}(G) \leq 2p-1$ and $\operatorname{srvc}(G) \leq \frac{p^2+p}{2}$.*

*Proof.* Let us fix a vertex cover $X$ of cardinality $p$. For the first claim, let $S$ be a spanning tree of $G$ with a minimum number of internal vertices, and observe that the number of internal vertices in $S$ is at most $2p-1$; indeed, one only needs to add at most $p-1$ vertices to the vertex cover in order to get a connected subgraph. Let $Z$ be the internal vertices of $S$. Let $\alpha$ be a vertex coloring which assigns a unique color from $[|Z|]$ to each vertex in $Z$, and then assigns the color 1 to each vertex in $V \setminus Z$. Then $\alpha$ is a rainbow vertex coloring: for any choice of $a$ and $b$, there exists an $a-b$ path whose internal vertices are a subset of $Z$.

For the second claim, consider the set $Q$ constructed as follows: for each distinct $a, b \in X$, if there exists a vertex $v$ in $V \setminus X$ adjacent to both $a$ and $b$, we choose an arbitrary such $v$ and add it into $Q$. Let $Z = Q \cup X$, and observe that $|Z| \leq p + \frac{p\cdot(p-1)}{2} = \frac{p^2+p}{2}$. Once again, let $\alpha$ be a vertex coloring which assigns a unique color from $[|Z|]$ to each vertex in $Z$, and then assigns the color 1 to each vertex in $V \setminus Z$. We claim that $\alpha$ is a strong rainbow vertex coloring. Indeed, consider any $a, b \in V$ and let $P$ be an arbitrary shortest $a-b$ path. Then for every internal vertex $v_i$ of $P$ such that $v_i \notin X$, it must hold that $v_{i-1} \in X$ and $v_{i+1} \in X$. Consider the path $P'$ obtained from $P$ by replacing each internal vertex $v_i \notin X$ by $v_i'$, where $v_i'$ is an element of $Q$ which is adjacent to $v_{i-1}$ and $v_{i+1}$. Since $P'$ has the same length as $P$ and $P'$ is rainbow colored by $\alpha$, the claim follows. ☐

The following lemma will be useful in the proof of Lemma 20, a key component of our approach for dealing with RC on the considered graph classes. A *bridge* is an edge $e$ such that deleting $e$ separates the connected component containing $e$ into two connected components.

**Lemma 19.** *Let $G = (V, E)$ be a graph and $X \subseteq V$ be a minimum vertex cover of $G$. Then there exist at most $2|X| - 2$ bridges which are not incident to a pendant outside of $X$.*

*Proof.* We prove the lemma by induction on $p = |X|$. If $p = 1$, then the graph is a star and the lemma holds (in a star, every bridge is incident to a pendant).

So, assume the lemma holds for $p-1$ and assume $G$ has a vertex cover $X$ of size $p$. Let $S$ be the set of all bridges in $G$ which are not incident to a pendant outside of $X$. If $S$ contains a bridge $e$ whose both endpoints lie in $X$, then $e$ separates $X$ into two non-empty subsets $X_1$ and $X_2$ and every other bridge has both endpoints either in $X_1$ or in $X_2$. Let $G_1$ and $G_2$ be the connected components of $G - e$ containing $X_1$ and $X_2$, respectively. Observe that $X_i$ is a vertex cover of $G_i$ for $i \in [2]$. Since $|X_1| < p$ and $|X_2| < p$, by our inductive assumption it follows that $G_1$ contains at most $2|X_1| - 2$ bridges which are not incident to a pendant outside of $X$, and similarly $G_2$ contains at most $2|X_2| - 2$ bridges which are not incident to a pendant outside of $X$. Since each bridge in $G$ is either $e$ or a bridge in $G_1$ or $G_2$, it follows that $|S| = 1 + 2|X_1| - 2 + 2|X_2| - 2 = 1 + 2p - 4 < 2p - 2$, and hence in this case the lemma holds.

On the other hand, assume $S$ contains a bridge $e = ax$ where $x \in X, a \notin X$. Since the connected component of $G - e$ containing $a$ is not a pendant, it follows that $a$ has a neighbor in $G$ which is different from $x$, and hence this connected component (say $G_1$) contains at least one vertex from $X$. Let $X_1 = X \cap V(G_1)$, $X_2 = X \setminus X_1$ and $G_2$ be the connected component of $G - e$ containing $X_2$. This implies that in this case $e$ also separates $X$ into two non-empty subsets $X_1$ and $X_2$. Furthermore, if there exists another $e' \in S$ which separates $X$ into the same sets $X_1$ and $X_2$ as $e$, then $e'$ must also be incident to $a$ and in particular this other edge $e'$ is unique; every other bridge in $S$ has both endpoints either in $X_1$ or in $X_2$. Since $|X_1| < p$ and $|X_2| < p$, by our inductive assumption it follows that $G_1$ contains at most $2|X_1| - 2$ bridges which are not incident to a pendant outside of $X$, and similarly $G_2$ contains at most $2|X_2| - 2$ bridges which are not incident to a pendant outside of $X$. Since each bridge in $G$ is either $e$ or $e'$ or a bridge in $G_1$ or $G_2$, it follows that $|S| \leq 2 + 2|X_1| - 2 + 2|X_2| - 2 = 2 + 2p - 4 \leq 2p - 2$, and hence in this case the lemma also holds. ☐

For ease of presentation, we define the function $\beta$ as $\beta(p) = 2p - 2 + p \cdot (p^2 + 2p \cdot 2^p)$. The next Lemma 20 will represent one part of our win-win strategy, as it allows us to precisely compute $\mathrm{rc}(G)$ when the number of bridges is sufficiently large. We remark that an analogous claim does not hold for $\mathrm{src}(G)$ (regardless of the choice of $\beta$).

**Lemma 20.** *Let $G = (V, E)$ be a connected graph and $p = \tau(G)$. Let $z$ be the number of bridges in $G$. If $z \geq \beta(p)$, then $\mathrm{rc}(G) = z$.*

*Proof.* Let us fix a vertex cover $X$ of cardinality $p$. It is known that the number of bridges is a lower bound for $\mathrm{rc}(G)$ [9], i.e., $\mathrm{rc}(G) \geq z$. We will show that $z$ is also an upper bound for $\mathrm{rc}(G)$.

Consider the edge $z$-coloring $\alpha$ constructed as follows. Since $X$ is a vertex cover and, by Lemma 19 in conjunction with our assumption on $z$, there are at least $p\cdot(p^2+2p\cdot 2^p)$ leaves in $G$, it follows that there must exist some $x \in X$ adjacent to at least $z' = p^2 + 2p \cdot 2^p$ pendants. Let $\{e_1, \ldots, e_{z'}\}$ be the edges incident to both $x$ and a pendant vertex, and let $\{e_{z'+1}, \ldots, e_z\}$ be all the remaining bridges; then for each bridge we set $\alpha(e_i) = i$.

Let $f_1, \ldots, f_q$ be the edges of $G[X]$ which are not bridges; for each such edge we set $\alpha(f_i) = z' - i$. Observe that for each such $f_i$ we have $\alpha(f_i) > 2p \cdot 2^p$.

Consider the set $\tau = \{\, T_i \mid T_i \text{ is a type in } G \text{ and } |N(T_i)| > 1\,\}$. Let $Q_i = \{2pi+1, \ldots, 2pi+2p\}$. For each $T_i \in \tau$, we let $G_i$ be the subgraph of $G$ on $T_i \cup N(T_i)$ which contains exactly the edges incident to $T_i$. Then $G_i$ is bipartite, and furthermore can be rainbow colored using at most $2p$ colors as follows: we pick an arbitrary $y \in T_i$ and uniquely color all edges in $G_i$ incident to $y$ using colors $c_1, \ldots, c_p$, and for every other vertex in $T_i$ we color all edges in $G_i$ incident to $y'$ using colors $c_{1+p}, \ldots, c_{2p}$. For each type $T_i \in \tau$, we let $\alpha$ color the edges incident to $T_i$ in this manner using the colors from $Q_i$.

We will proceed by arguing that $\alpha$ is a rainbow $z$-coloring of $G$, but before that we make three key observations. First, there are only two cases when $\alpha$ can use the same color for two distinct edges $e, f$: either one of $e, f$ is an edge between $x$ and a pendant, or both $e, f$ occur in some $G_i$. Second, if $|T_i| > 1$, then for every $u \in N(T_i)$ and every $v \in V(G_i)$ and every color $c$, there exists a rainbow $u - v$ path in $G_i$ under $\alpha$ which does not use $c$. Third, each $G_i$ is rainbow colored by $\alpha$.

We now make the following case distinction.

1. Let $a, b \in V$ be such that neither is a pendant adjacent to $x$. Consider an arbitrary $a - b$ path $P$ such that the number of pairs of edges in $P$ assigned the same color by $\alpha$ is minimized. If $P$ contains two edges $e, f$ such that $\alpha(e) = \alpha(f)$, then both $e$ and $f$ must occur in some $G_i$. Let $t$ and $u$ be the first and last vertex in $V(G_i)$ on $P$, respectively. Since $G_i$ is rainbow colored by $\alpha$, there exists a $t - u$ rainbow path $P^*$ in $G_i$. Let $P'$ be obtained from $P$ by replacing the path segment between $t$ and $u$ by $P^*$; by the key observation made above, it follows that $P'$ has a strictly lower number of pairs of edges in $P$ which are assigned the same color by $\alpha$, hence contradicting our choice of $P$.

2. Let $a$ be a pendant adjacent to $x$, and $b \in V$. Let $c = \alpha(xa)$. Consider an arbitrary $a - b$ path $P$ such that the number of pairs of edges in $P$ assigned the same color by $\alpha$ is minimized. If $P$ contains two edges $e, f$ such that $\alpha(e) = \alpha(f) \neq c$, then both $e$ and $f$ must occur in some $G_i$ s.t. $|T_i| > 1$. Let $t$ and $u$ be the first and last vertex in $V(G_i)$ on $P$, respectively. Since $t \in X \cap N(T_i)$, by our observations above it follows that there exists a rainbow $t - u$ path $P^*$ in $G_i$ which avoids $c$. Hence the path obtained by replacing the path segment between $t$ and $u$ by $P^*$ once again contradicts our choice of $P$. On the other hand, if $P$ contains an edge $e$ such that $\alpha(e) = c$, then either $e$ is an

edge in $G[X]$ or $e$ is incident to some $T_i$. In the latter case, the same argument can be used to contradict our choice of $P$. In the former case it follows by construction of $\alpha$ that $c$ only occurs on the edge $(x, a)$ and on $e$, and furthermore $e$ is contained in some 2-edge-connected component $D$ of $G$. Let $d, w$ be the first and last vertex, respectively, in $D$ which occurs in $P$, and let $P'$ be the path obtained from $P$ by replacing the path segment between $d$ and $w$ by an arbitrary rainbow path segment in $D$ which does not contain $e$. It is readily verified that the colors which occur in $D$ are only repeated on edges between $x$ and pendants, and in particular such edges cannot occur on $P'$. Hence $P'$ again contradicts our choice of $P$.

To summarize, for any $a, b \in V$ there exists a rainbow $a - b$ path under $\alpha$, and hence $\alpha$ witnesses that $\mathrm{rc}(G) \leq z$. We conclude that $\mathrm{rc}(G) = z$. □

**Lemma 21.** *Let $G = (V, E)$ be a graph with a vertex cover $X \subseteq V$ of cardinality $p$. Let $z$ be the number of bridges in $G$. If $z < \beta(p)$, then $\mathrm{rc}(G) \leq \beta(p) + p^2 + 2^p \cdot 2p$.*

*Proof.* Consider the following edge coloring $\alpha$ which assigns a unique color to each edge in $G[X]$ and to each edge incident to a pendant. For each nonempty type $T_i$, we choose an arbitrary vertex $y_i$ and let $\alpha$ assign a unique color for each of the at most $p$ edges incident to $y_i$. Finally, for each type $T_i$ and each $x \in X$ adjacent to (the vertices of) $T_i$, $\alpha$ uses a single new color for all edges between $x$ and the vertices in $T_i$. It is readily verified that $\alpha$ uses no more than $z + p^2 + 2^p \cdot 2p$ colors.

We argue that $\alpha$ is rainbow. Let $G_i$ be the subgraph of $G$ on $T_i \cup N(T_i)$ which contains exactly the edges incident to $T_i$, and observe that each $G_i$ is rainbow colored by $\alpha$. Consider any $a, b \in V$ and let $P$ be an $a - b$ path such that the number of pairs of edges in $P$ assigned the same color by $\alpha$ is minimized. By construction of $\alpha$, two edges $e, f$ in $P$ may only have the same color if $e, f$ are both incident to some $T_i$. Let $t$ and $u$ be the first and last vertex in $V(G_i)$ on $P$, respectively. Since $G_i$ is rainbow colored by $\alpha$, there exists a $t - u$ rainbow path $P^*$ in $G_i$ under $\alpha$. Let $P'$ be obtained from $P$ by replacing the path segment between $t$ and $u$ by $P^*$. Then $P'$ has a strictly lower number of pairs of edges in $P$ with the same color, which contradicts our choice of $P$. □

**Theorem 22.** *Let $p \in \mathbb{N}$ be fixed. Then the problems* RC, RVC, *and* SRVC *can be solved in time $\mathcal{O}(n)$ on $n$-vertex graphs of vertex cover number at most $p$.*

*Proof.* For RVC and SRVC, we first observe that if $k$ (the queried upper bound on the number of colors) is greater than $2p - 1$ and $\frac{p^2+p}{2}$, respectively, then the algorithm can immediately output YES by Lemma 18. Otherwise we use Theorem 12 and the fact that the vertex cover number is an upper bound on the treewidth to compute a solution in $\mathcal{O}(n)$ time.

For RC, it is well known that the total number of bridges in $G$, say $z$, can be computed in linear time on graphs of bounded treewidth. If $z \geq \beta(p)$, then by Lemma 20 we can correctly output YES when $z \leq k$ and NO when $z > k$. On the other hand, if $z < \beta(p)$, then by Lemma 21 the value $\mathrm{rc}(G)$ is upper-bounded by a function of $p$. We compare $k$ and this upper bound on $\mathrm{rc}(G)$; if $k$ exceeds the upper bound on $\mathrm{rc}(G)$, then we output YES, and otherwise we can use Theorem 12 along with the fact that the vertex cover number is an upper bound on the treewidth to compute a solution in $\mathcal{O}(n)$ time. □

## 7 Concluding Notes

We presented new positive and negative results for the most prominent variants of rainbow coloring. We believe that the techniques presented above, and in particular the win-win approaches used in Section 5 and Section 6, can be of use also for other challenging connectivity problems.

It is worth noting that our results in Section 4 leave open the question of whether Rainbow Coloring or its variants can be solved in (uniformly) polynomial time on graphs of bounded treewidth. Hardness results for related problems [23,30] do not imply that finding an optimal coloring of a bounded-treewidth graph is hard, and it seems that new insights are needed to determine the complexity of these problems on graphs of bounded treewidth. Finally, the complexity of the SRC problem still remains open on graphs of bounded vertex cover number.

## 8 Acknowledgments

The authors would like to thank the reviewers for their useful comments. In addition, the authors would like to thank Martin Derka for his helpful comments and discussion at the IWOCA'15 conference that took place in Verona, Italy.

This work was partially supported by the Austrian Science Fund (FWF), projects P26696 and W1255-N23 (E.E., R.G.), and by the Emil Aaltonen Foundation (J.L.). Robert Ganian is also affiliated with FI MU, Brno, Czech Republic.

## References

[1] P. Ananth, M. Nasre, and K. K. Sarpatwar. Rainbow connectivity: Hardness and tractability. In *IARCS Annual Conference on Foundations of Software Technology and Theoretical Computer Science, FSTTCS 2011*, pages 241–251, 2011.

[2] M. Basavaraju, L. Chandran, D. Rajendraprasad, and A. Ramaswamy. Rainbow connection number and radius. *Graphs and Combinatorics*, pages 1–11, 2012.

[3] H. Bodlaender. On linear time minor tests with depth-first search. *Journal of Algorithms*, 14(1):1–23, 1993.

[4] H. L. Bodlaender and T. Kloks. Efficient and constructive algorithms for the pathwidth and treewidth of graphs. *Journal of Algorithms*, 21(2):358–402, 1996.

[5] S. Chakraborty, E. Fischer, A. Matsliah, and R. Yuster. Hardness and algorithms for rainbow connection. *Journal of Combinatorial Optimization*, 21(3):330–347, 2009.

[6] L. S. Chandran and D. Rajendraprasad. Rainbow Colouring of Split and Threshold Graphs. *Computing and Combinatorics*, pages 181–192, 2012.

[7] L. S. Chandran and D. Rajendraprasad. Inapproximability of rainbow colouring. In *IARCS Annual Conference on Foundations of Software Technology and Theoretical Computer Science, FSTTCS 2013*, pages 153–162, 2013.

[8] G. Chartrand, G. Johns, K. McKeon, and P. Zhang. Rainbow connection in graphs. *Mathematica Bohemica*, 133(1), 2008.

[9] G. Chartrand, F. Okamoto, and P. Zhang. Rainbow trees in graphs and generalized connectivity. *Networks*, 55(4):360–367, 2010.

[10] G. Chartrand and P. Zhang. *Chromatic graph theory.* CRC press, 2008.

[11] L. Chen, X. Li, and H. Lian. Further hardness results on the rainbow vertex-connection number of graphs. *Theoretical Computer Science*, 481(0):18–23, 2013.

[12] L. Chen, X. Li, and Y. Shi. The complexity of determining the rainbow vertex-connection of a graph. *Theoretical Computer Science*, 412(35):4531–4535, 2011.

[13] B. Courcelle. The Monadic Second-Order Logic of Graphs I. Recognizable Sets of Finite Graphs. *Information and Computation*, pages 12–75, 1990.

[14] B. Courcelle, J. A. Makowsky, and U. Rotics. Linear time solvable optimization problems on graphs of bounded clique-width. *Theory of Computing Systems*, 33(2):125–150, 2000.

[15] R. Diestel. *Graph Theory.* Springer-Verlag Heidelberg, 2010.

[16] R. G. Downey and M. R. Fellows. *Fundamentals of Parameterized Complexity.* Springer, 2013.

[17] E. Eiben, R. Ganian, and J. Lauri. On the complexity of rainbow coloring problems. In *Proceedings of the 26th International Workshop on Combinatorial Algorithms, IWOCA 2015, Verona, Italy, October 5–7*, pages 209–220, 2015.

[18] R. Frucht and F. Harary. On the corona of two graphs. *Aequationes Mathematicae*, 4(3):322–325, 1970.

[19] R. Ganian and P. Hliněný. On parse trees and Myhill-Nerode-type tools for handling graphs of bounded rank-width. *Discrete Applied Mathematics*, 158(7):851–867, 2010.

[20] G. Gottlob and S. T. Lee. A logical approach to multicut problems. *Information Processing Letters*, 103(4):136–141, 2007.

[21] M. Keranen and J. Lauri. Computing minimum rainbow and strong rainbow colorings of block graphs. *arXiv preprint arXiv:1405.6893*, 2014.

[22] M. Krivelevich and R. Yuster. The rainbow connection of a graph is (at most) reciprocal to its minimum degree. *Journal of Graph Theory*, 63(3):185–191, 2010.

[23] J. Lauri. Further hardness results on rainbow and strong rainbow connectivity. *Discrete Applied Mathematics*, 201:191–200, 2016.

[24] X. Li, Y. Mao, and Y. Shi. The strong rainbow vertex-connection of graphs. *Utilitas Mathematica*, 93:213–223, 2014.

[25] X. Li, Y. Shi, and Y. Sun. Rainbow Connections of Graphs: A Survey. *Graphs and Combinatorics*, 29(1):1–38, 2012.

[26] X. Li and Y. Sun. *Rainbow connections of graphs.* Springer, 2012.

[27] R. Niedermeier. *Invitation to fixed-parameter algorithms.* Oxford University Press Oxford, 2006.

[28] N. Robertson and P. Seymour. Graph minors. II. Algorithmic aspects of tree-width. *Journal of Algorithms*, 7(3):309–322, 1986.

[29] L. Sunil Chandran, A. Das, D. Rajendraprasad, and N. M. Varma. Rainbow connection number and connected dominating sets. *Journal of Graph Theory*, 71(2):206–218, 2012.

[30] K. Uchizawa, T. Aoki, T. Ito, A. Suzuki, and X. Zhou. On the Rainbow Connectivity of Graphs: Complexity and FPT Algorithms. *Algorithmica*, 67(2):161–179, 2013.

[31] D. Zuckerman. Linear degree extractors and the inapproximability of max clique and chromatic number. *Theory of Computing*, 3(6):103–128, 2007.

# Paper 5

Łukasz Kowalik, Juho Lauri, and Arkadiusz Socała.

**On the fine-grained complexity of rainbow coloring.**

© 2016 the authors (licensed under Creative Commons License CC-BY). In: Proceedings of the 24th Annual European Symposium on Algorithms, ESA 2016, Aarhus, Denmark, August 22-24. 2016, 58:1–58:16.

doi:10.4230/LIPIcs.ESA.2016.58

# On the Fine-Grained Complexity of Rainbow Coloring*

**Łukasz Kowalik[1], Juho Lauri[2], and Arkadiusz Socała[3]**

1 **University of Warsaw, Warsaw, Poland**
`kowalik@mimuw.edu.pl`
2 **Tampere University of Technology, Tampere, Finland**
`juho.lauri@tut.fi`
3 **University of Warsaw, Warsaw, Poland**
`a.socala@mimuw.edu.pl`

## Abstract

The Rainbow $k$-Coloring problem asks whether the edges of a given graph can be colored in $k$ colors so that every pair of vertices is connected by a rainbow path, i.e., a path with all edges of different colors. Our main result states that for any $k \geq 2$, there is no algorithm for Rainbow $k$-Coloring running in time $2^{o(n^{3/2})}$, unless ETH fails. Motivated by this negative result we consider two parameterized variants of the problem. In the Subset Rainbow $k$-Coloring problem, introduced by Chakraborty *et al.* [STACS 2009, J. Comb. Opt. 2009], we are additionally given a set $S$ of pairs of vertices and we ask if there is a coloring in which all the pairs in $S$ are connected by rainbow paths. We show that Subset Rainbow $k$-Coloring is FPT when parameterized by $|S|$. We also study Maximum Rainbow $k$-Coloring problem, where we are additionally given an integer $q$ and we ask if there is a coloring in which at least $q$ anti-edges are connected by rainbow paths. We show that the problem is FPT when parameterized by $q$ and has a kernel of size $O(q)$ for every $k \geq 2$, extending the result of Ananth *et al.* [FSTTCS 2011]. We believe that our techniques used for the lower bounds may shed some light on the complexity of the classical Edge Coloring problem, where it is a major open question if a $2^{O(n)}$-time algorithm exists.

**1998 ACM Subject Classification** F.2.2 Nonnumerical Algorithms and Problems

**Keywords and phrases** graph coloring, computational complexity, lower bounds, exponential time hypothesis, FPT algorithms

**Digital Object Identifier** 10.4230/LIPIcs.ESA.2016.58

## 1 Introduction

The Rainbow $k$-Coloring problem asks whether the edges of a given graph can be colored in $k$ colors so that every pair of vertices is connected by a rainbow path, i.e., a path with all edges of different colors. A minimum such $k$, called the *rainbow connection number* can be viewed as yet another measure of graph connectivity. The concept of rainbow coloring was introduced by Chartrand, Johns, McKeon, and Zhang [7] in 2008, while also featured in an earlier book of Chartrand and Zhang [8]. Chakraborty, Fischer, Matsliah, and Yuster [3] describe an interesting application of rainbow coloring in telecommunications. The problem

* Work partially supported by the National Science Centre of Poland, grant number 2013/09/B/ST6/03136 (Ł.K., A.S.), and by the Emil Aaltonen Foundation (J.L.). This is an extended abstract; for the full version see the technical report at arxiv [17].

24th Annual European Symposium on Algorithms (ESA 2016).
Editors: Piotr Sankowski and Christos Zaroliagis; Article No. 58; pp. 58:1–58:16
Leibniz International Proceedings in Informatics
Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

is intensively studied from the combinatorial perspective, with over 100 papers published by now (see the survey of Li, Shi, and Sun [20] for an overview). However, the computational complexity of the problem seems less explored. It was conjectured by Caro, Lev, Roditty, Tuza, and Yuster [2] that the Rainbow $k$-Coloring problem is NP-complete for $k = 2$. This conjecture was confirmed by Chakraborty *et al.* [3]. Ananth, Nasre, and Sarpatwar [1] noticed that the proof of Chakraborty *et al.* in fact proves NP-completeness for every even $k > 1$, and complemented this by showing NP-completeness of the odd cases as well. An alternative hardness proof for every $k > 1$ was provided by Le and Tuza [19]. For complexity results on restricted graph classes, see e.g., [4, 5, 6, 12].

For many NP-complete graph problems there are algorithms running in time $2^{O(n)}$ for an $n$-vertex graph. This is obviously the case for problems asking for a set of vertices, like Clique or Vertex Cover, or more generally, for problems which admit polynomially (or even subexponentially) checkable $O(n)$-bit certificates. However, there are $2^{O(n)}$-time algorithms also for some problems for which such certificates are not known, including e.g., Hamiltonicity [13] and Vertex Coloring [18]. Unfortunately it seems that the best known worst-case running time bound for Rainbow $k$-Coloring is $k^m 2^k n^{O(1)}$, where $m$ is the number of edges, which is obtained by checking each of the $k^m$ colorings by a simple $2^k n^{O(1)}$-time dynamic programming algorithm [23]. Even in the simplest variant of just two colors, i.e., $k = 2$, this algorithm takes $2^{O(n^2)}$ time if the input graph is dense. It raises a natural question: is this problem really much harder than, say, Hamiltonicity, or have we just not found the right approach yet? Questions of this kind have received considerable attention recently. For example, the existence of a $2^{O(n)}$-time algorithm for Edge Coloring is a notorious question, appearing in numerous open problem lists. On the other hand, it was shown that unless the Exponential Time Hypothesis fails, there is no algorithm running in time $2^{o(n \log n)}$ for Channel Assignment [21], Subgraph Homomorphism, and Subgraph Isomorphism [9]. Let us recall the precise statement of the Exponential Time Hypothesis (ETH).

▶ **Hypothesis 1** (Exponential Time Hypothesis [14]). *There exists a constant $c > 0$, such that there is no deterministic algorithm solving* 3-SAT *in time $O^*(2^{cn})$.*

Note that some kind of a complexity assumption, like ETH, is hard to avoid when we prove exponential lower bounds, unless one aims at proving P $\neq$ NP.

**Main Result.** Our main result is the following theorem.

▶ **Theorem 2.** *For any $k \geq 2$,* Rainbow $k$-Coloring *can be solved neither in $2^{o(n^{3/2})}$ nor $2^{o(m/\log m)}$ time where $n$ and $m$ are the number of vertices and edges respectively, unless ETH fails.*

Hence, this is an NP-complete graph problem which does not admit a $2^{o(n^{1+\epsilon})}$-time algorithm (under reasonable complexity assumptions), for an $\epsilon > 0$. Such lower bounds are fairly rare in the literature. The best known algorithm for Rainbow $k$-Coloring just verifies all possible colorings and thus it runs in time $2^{O(m)}$ for any fixed $k$. Our lower bounds mean that one cannot hope for substantial improvements in this running time.

**Remaining Lower Bounds.** We also study a natural generalized problem, called Subset Rainbow $k$-Coloring, introduced by Chakraborty *et al.* [3] as a natural intermediate step in reductions from 3-SAT to Rainbow $k$-Coloring. In Subset Rainbow $k$-Coloring, we are given a connected graph $G$, and a set of pairs of vertices $S \subseteq \binom{V(G)}{2}$. Elements of $S$

are called *requests*. For a given coloring of $E(G)$ we say that a request $\{u, v\}$ is *satisfied* if $u$ and $v$ are connected by a rainbow path. The goal in Subset Rainbow $k$-Coloring is to determine whether there is a $k$-coloring of $E(G)$ such that every pair in $S$ is satisfied. Our main result implies that Subset Rainbow $k$-Coloring admits no algorithm running in time $2^{o(n^{3/2})}$, under ETH. We show also two more lower bounds, as follows.

▶ **Theorem 3.** *For any $k \geq 2$,* Subset Rainbow $k$-Coloring *can be solved neither in time $2^{o(n^{3/2})}$, nor in time $2^{o(m)}$, nor in time $2^{o(s)}$ where $n$ is the number of vertices, $m$ is the number of edges, and $s$ is the number of requests, unless ETH fails.*

An interesting feature here is that for $k = 2$ the $2^{o(m)}$ and $2^{o(s)}$ bounds are *tight* up to a polynomial factor (a $2^m n^{O(1)}$ algorithm is immediate, and a $2^{|S|}n^{O(1)}$-time algorithm is discussed in the next paragraph).

**New Algorithms.** In the context of the hardness results mentioned above it is natural to ask for FPT algorithms for Subset Rainbow $k$-Coloring. We show that for every fixed $k$, Subset Rainbow $k$-Coloring parameterized by $|S|$ is FPT:

▶ **Theorem 4.** *For every integer $k$,* Subset Rainbow $k$-Coloring *is FPT and it has an algorithm running in time $|S|^{O(|S|)}n^{O(1)}$.*

For the 2 color case we are able to show a different, faster algorithm running in time $2^{|S|}n^{O(1)}$, which is tight up to a polynomial factor.

We also study the Maximum Rainbow $k$-Coloring problem, introduced by Ananth, Nasre, and Sarpatwar [1]. Intuitively, the idea is to parameterize the problem by the number of pairs to satisfy. However, all pairs of adjacent vertices are trivially satisfied by any edge-coloring. Hence, we parameterize by the number of anti-edges to satisfy. More formally, in Maximum Rainbow $k$-Coloring we are given a graph $G = (V, E)$, an integer $q$, and asked whether there is a coloring of $E$ that satisfies at least $q$ anti-edges. First, we show that the maximization version of the problem (find maximum such $q$) admits a constant factor approximation algorithm for every fixed value of $k$. Second, we show that Maximum Rainbow $k$-Coloring is FPT for every $k \geq 2$, which generalizes the result of Ananth *et al.* [1] who showed this claim for the $k = 2$ case. Our algorithm runs in time $2^{q \log q}n^{O(1)}$ for any $k$, which is faster than the algorithm of Ananth *et al.* for 2 colors. For 2 colors we give an even faster algorithm, running in time $8^q n^{O(1)}$. We also show that the problem admits a kernel size $O(q)$, i.e., that there is a polynomial-time algorithm that returns an equivalent instance with $O(q)$ vertices. (For more background on kernelization see e.g., [10].) Before, this was known only for $k = 2$ (due to Ananth *et al.* [1]). Our main results for Maximum Rainbow $k$-Coloring are summarized in the following theorem.

▶ **Theorem 5.** Maximum Rainbow $k$-Coloring *parameterized by the number of anti-edges $q$ is FPT for every $k \geq 2$. Moreover, it admits a kernel of linear size.*

**Notation.** For standard graph-theoretic notions, we refer the reader to [11]. All graphs we consider in this paper are simple and undirected. We denote $\Delta_1(G) = \max\{\Delta(G), 1\}$.

By $\bar{E}$ we denote the set of anti-edges, i.e., $\bar{E} = \binom{V}{2} \setminus E$. When $G = (V, E)$ is a graph then $\bar{G} = (V, \bar{E})$ is its *complement graph*. By $x^{\underline{k}}$ we denote the falling factorial, i.e., $x^{\underline{k}} = x(x-1)\cdots(x-k+1)$. For an integer $k$, we denote $[k] = \{1, \dots, k\}$. For a (partial) function $c$, by $\mathrm{Dom}(c)$ we denote its domain.

If $I$ and $J$ are instances of decision problems $P$ and $R$, respectively, then we say that $I$ and $J$ are *equivalent*, when either both $I$ and $J$ are YES-instances or both are NO-instances.

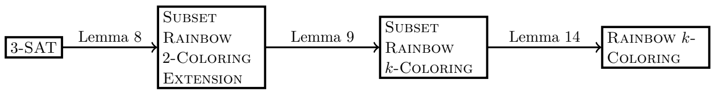

3-SAT
Lemma 8
Subset Rainbow 2-Coloring Extension
Lemma 9
Subset Rainbow $k$-Coloring
Lemma 14
Rainbow $k$-Coloring

**Figure 1** A simplified road map of our reductions.

**Organization of the paper.** In Section 2 we present our hardness results. The main difficulties we encountered are sketched at the beginning of that section. Due to space constraints proofs of the claims marked by ★ are skipped and can be found in the full version [17]. Next, in Section 3 we present our algorithms for Subset Rainbow $k$-Coloring. Again because of the space limitations, our algorithms for Maximum Rainbow $k$-Coloring are skipped in this extended abstract and are available in the full version [17].

## 2 Hardness of rainbow coloring

### 2.1 Overview

The main goal of this section is to show that for any $k \geq 2$ Rainbow $k$-Coloring does not admit an algorithm running in time $2^{o(n^{3/2})}$, unless the Exponential Time Hypothesis fails. Let us give a high-level overview of our proof. A natural idea would be to begin with a 3-SAT formula $\phi$ with $n$ variables and then transform it in time $2^{o(n)}$ to an equivalent instance $G = (V, E)$ of Rainbow $k$-Coloring with $O(n^{2/3})$ vertices. Then indeed a $2^{o(|V|^{3/2})}$-time algorithm that solves Rainbow 2-Coloring can be used to decide 3-SAT in time $2^{o(n)}$. Note that in a typical NP-hardness reduction, we observe some polynomial blow-up of the instance size. For example, one can verify that in the reduction of Chakraborty *et al.* [3], the initial 3-SAT formula with $n$ variables and $m$ clauses is transformed into a graph with $\Theta(n^4 + m^4)$ vertices and edges. In our case, instead of a blow-up we aim at *compression*: the number of vertices needs to be much smaller than the number of variables in the input formula $\phi$. As usual in reductions, variables and clauses in $\phi$ are going to correspond to some structures in $G$, called gadgets. The compression requirement means that our gadgets need to share vertices. To make our lives slightly easier, we apply the following well-known Sparsification Lemma, which allows for assuming that the number of clauses is $O(n)$.

▶ **Lemma 6** (Sparsification Lemma [15]). *For each $\varepsilon > 0$ there exist a constants $c_\varepsilon$, such that any 3-SAT formula $\varphi$ with $n$ variables can be expressed as $\varphi = \vee_{i=1}^{t} \psi_i$, where $t \leq 2^{\varepsilon n}$ and each $\psi_i$ is a 3-SAT formula with the same variable set as $\varphi$, but contains at most $c_\varepsilon n$ clauses. Moreover, this disjunction can be computed in time $O^*(2^{\varepsilon n})$.*

Note that by using the Sparsification Lemma we tweak our general plan a bit: instead of creating one equivalent instance, we are going to create $2^{\varepsilon n}$ instances (for arbitrarily small $\epsilon$), each with $O(n^{2/3})$ vertices. The following lemma further simplifies the instance.

▶ **Lemma 7** ([22]). *Given a 3-SAT formula $\varphi$ with $m$ clauses one can transform it in polynomial time into a formula $\varphi'$ with $O(m)$ variables and $O(m)$ clauses, such that $\varphi'$ is satisfiable iff $\varphi'$ is satisfiable, and moreover each clause of $\varphi'$ contains exactly three different variables and each variable occurs in at most 4 clauses of $\varphi'$.*

Now our goal is to transform a 3-SAT formula $\phi$ with $n$ variables such that every variable occurs in at most 4 clauses, to a graph with $O(n^{2/3})$ vertices — an equivalent instance of Rainbow $k$-Coloring. We do it in three steps (see Fig 1).

In the first step we transform $\phi$ to an instance $I = (G, S, c_0)$ of Subset Rainbow 2-Coloring Extension, which is a generalization of Subset Rainbow 2-Coloring, where $c_0$, called a *precoloring*, is a partial coloring of the edges of $G$ into two colors and the goal is to determine if there is an edge-coloring of $E(G)$ which extends $c_0$ and such that all pairs of $S$ are satisfied. The first step is crucial, because here the compression takes place: $|V(G)| = O(n^{2/3})$ and $E(G) = O(n)$. The major challenge in the construction is avoiding interference between gadgets that share a vertex: to this end we define various conflict graphs and we show that they can be vertex-colored in a few colors. This reduction is described in Section 2.2.

In the second step (Lemma 9) we reduce Subset Rainbow 2-Coloring Extension to Subset Rainbow $k$-Coloring, for every $k \geq 2$. In fact, in the full version this is done in two sub-steps, via Subset Rainbow $k$-Coloring Extension. The number of the vertices in the resulting instance does not increase more than by a constant factor. This step is rather standard, though some technicalities appear because we need to guarantee additional properties of the output instance, which are needed by the reduction in the third step.

The last step (Section 2.3), where we reduce an instance $(G = (V, E), S)$ of Subset Rainbow $k$-Coloring to an instance $G'$ of Rainbow $k$-Coloring, is yet another challenge. We would like to get rid of the set of requests somehow. For simplicity, let us focus on the $k = 2$ case now. Here, the natural idea, used actually by Chakraborty *et al.* [3] is to create, for every $\{u, v\} \notin S$, a path $(u, x_{uv}, v)$ through a new vertex $x_{uv}$. Such a path cannot help any of the requests $\{u', v'\} \in S$ to get satisfied (since if it creates a new path $P'$ between $u'$ and $v'$, then $P'$ has length at least 3), and by coloring it into two different colors we can satisfy $\{u, v\}$. Unfortunately, in our case we cannot afford for creating a new vertex for every such $\{u, v\}$, because that would result in a quadratic blow up in the number of vertices. However, one can observe that for any biclique (a complete bipartite subgraph) in the graph $(V, \binom{V}{2} \setminus S)$ it is sufficient to use just one such vertex $x$ (connected to all the vertices of the biclique). By applying a result of Jukna [16] we can show that in our specific instance of Subset Rainbow 2-Coloring which results from a 3-SAT formula, the number of bicliques needed to cover all the pairs in $\binom{V}{2} \setminus S$ is small enough. We show a $2^{|V(G)|}|V(G)|^{O(1)}$-time algorithm to find such a cover. Although this algorithm does not seem fast, in our case $|V(G)| = O(n^{2/3})$, so this complexity is *subexponential* in the number of variables of the input formula, which is enough for our goal. The case of $k \geq 3$ is similar, i.e., we also use the biclique cover. However, the details are much more technical because for each biclique we need to introduce a much more complex gadget.

## 2.2 From 3-SAT to Subset Rainbow $k$-Coloring

Let Subset Rainbow $k$-Coloring Extension be a generalization of Subset Rainbow $k$-Coloring, where $c_0$ is a partial $k$-coloring of the edges of $G$ and the goal is to determine if there is an edge-coloring of $E(G)$ which extends $c_0$ and such that all pairs of $S$ are satisfied. In this section we show a reduction (Lemma 8) from 3-SAT to Subset Rainbow 2-Coloring Extension.

For an instance $I = (G, S, c_0)$ of Subset Rainbow $k$-Coloring Extension (for any $k \geq 2$), let us define a *precoloring conflict graph* $CG_I$. Its vertex set is the set of colored edges, i.e., $V(CG_I) = \mathrm{Dom}(c_0)$. Two different colored edges $e_1$ and $e_2$ (treated as vertices of $CG_I$) are adjacent in $CG_I$ when they are incident in $G$ or there is a pair of endpoints $u \in e_1$ and $v \in e_2$ such that $uv \in E(G) \cup S$.

In what follows the reduction in Lemma 8 is going to be pipelined with further reductions going through Subset Rainbow $k$-Coloring Extension and Subset Rainbow $k$-

Coloring to Rainbow $k$-Coloring. In these three reductions we need to keep the instance small. To this end, the instance of Subset Rainbow 2-Coloring Extension resulting in Lemma 8 has to satisfy some additional properties, which are formulated in the claim of Lemma 8. Their role will become clearer later on.

▶ **Lemma 8.** *Given a* 3-SAT *formula $\varphi$ with $n$ variables such that each clause of $\varphi$ contains exactly three variables and each variable occurs in at most four clauses, one can construct in polynomial time an equivalent instance $(G, S, c_0)$ of* Subset Rainbow 2-Coloring Extension *such that $G$ has $O(n^{2/3})$ vertices and $O(n)$ edges. Moreover, $\Delta(G) = O(n^{1/3})$, $\Delta(V(G), S) = O(n^{1/3})$, $|\mathrm{Dom}(c_0)| = O(n^{2/3})$ and along with the instance $I = (G, S, c_0)$ the algorithm constructs a proper vertex 4-coloring of $(V(G), E \cup S)$ (so also of $(V(G), S)$) and a proper vertex $O(n^{1/3})$-coloring of the precoloring conflict graph $CG_I$.*

**Proof.** Let $m$ denote the number of clauses in $\varphi$. Observe that $m \leq \frac{4}{3}n$. Let Var and Cl denote the sets of variables and clauses of $\varphi$. For more clarity, the two colors of the partial coloring $c_0$ will be called $T$ and $F$. Let us describe the graph $G$ along with a set of anti-edges $S$. Graph $G$ consists of two disjoint vertex subsets: the variable part and the clause part. The intuition is that in any 2-edge coloring of $G$ that extends $c_0$ and satisfies all pairs in $S$

- edge colors in the variable part represent an assignment of the variables of $\varphi$,
- edge colors in the clause part represent a choice of literals that satisfy all the clauses, and
- edge colors between the two parts make the values of the literals from the clause part consistent with the assignment represented by the variable part.

**The variable part.** The vertices of the variable part consist of the *middle set* $M$ and $\lceil n^{1/3} \rceil$ layers $L_1 \cup L_2 \cdots \cup L_{\lceil n^{1/3} \rceil}$. The middle set $M$ consists of vertices $m_i$ for each $i = 1, \ldots, \lceil n^{2/3} \rceil + 9$. For every $i = 1, \ldots, \lceil n^{1/3} \rceil$ the layer $L_i$ consists of two parts: upper $L_i^{\uparrow} = \{u_{i,j} \ : \ j = 1, \ldots, \lceil n^{1/3} \rceil + 3\}$ and lower $L_i^{\downarrow} = \{l_{i,j} \ : \ j = 1, \ldots, \lceil n^{1/3} \rceil + 3\}$.

We are going to define four functions: mid : Var $\to M$, lay, up, low : Var $\to [\lceil n^{1/3} \rceil]$. Then, for every variable $x \in$ Var we add two edges $u_{\mathrm{lay}(x),\mathrm{up}(x)}\mathrm{mid}(x)$ and $\mathrm{mid}(x)l_{\mathrm{lay}(x),\mathrm{low}(x)}$. Moreover, we add the pair $p_x = \{u_{\mathrm{lay}(x),\mathrm{up}(x)}, l_{\mathrm{lay}(x),\mathrm{low}(x)}\}$ to $S$. In other words, $x$ corresponds to the 2-path $u_{\mathrm{lay}(x),\mathrm{up}(x)}\mathrm{mid}(x)l_{\mathrm{lay}(x),\mathrm{low}(x)}$. Now we describe a careful construction of the four functions, that guarantee several useful properties (for example edge-disjointness of paths corresponding to different variables).

Let us define the *variable conflict graph* $G_V = (\mathrm{Var}, E_{G_V})$, where for two variables $x, y \in$ Var we have $xy$ are adjacent iff they both occur in the same clause. Since every variable occurs in at most 4 clauses, $\Delta(G_V) \leq 8$. It follows that there is a proper vertex 9-coloring $\alpha : Var \to [9]$ of $G_v$, and it can be found by a simple linear time algorithm. Next, each of the 9 color classes $\alpha^{-1}(i)$ is partitioned into $\lceil |\alpha^{-1}(i)| / \lceil n^{1/3} \rceil \rceil$ disjoint groups, each of size at most $\lceil n^{1/3} \rceil$. It follows that the total number $n_g$ of groups is at most $\lceil n^{2/3} \rceil + 9$. Let us number the groups arbitrarily from 1 to $n_g$ and for every variable $x \in$ Var, let $g(x)$ be the number of the group that contains $x$. Then we define $\mathrm{mid}(x) = m_{g(x)}$. Since any group contains only vertices of the same color we can state the following property:

$(\mathbf{P_1})$ If variables $x$ and $y$ occur in the same clause then $\mathrm{mid}(x) \neq \mathrm{mid}(y)$.

Now, for every variable $x$ we define its layer, i.e., the value of the function $\mathrm{lay}(x)$. Recall that for every $i = 1, \ldots, \lceil n^{2/3} \rceil + 9$ the $i$-th group $\mathrm{mid}^{-1}(m_i)$ contains at most $\lceil n^{1/3} \rceil$ variables. Inside each group, number the variables arbitrarily and let $\mathrm{lay}(x)$ be the number of variable $x$ in its group, $\mathrm{lay}(x) \in [n^{1/3}]$. This implies another important property.

$(\mathbf{P_2})$ If variables $x$ and $y$ belong to the same layer then $\mathrm{mid}(x) \neq \mathrm{mid}(y)$.

Observe that every layer gets assigned at most $\lceil n^{2/3} \rceil + 9$ variables. For every layer $L_i$ pick any injective function $h_i : \mathrm{lay}^{-1}(i) \to [\lceil n^{1/3} \rceil + 3]^2$. Then, for every variable $x \in \mathrm{Var}$ we put $(\mathrm{up}(x), \mathrm{low}(x)) = h_{\mathrm{lay}(x)}(x)$. Note that by ($\mathrm{P}_2$) we have the following.

**($\mathbf{P}_3$)** For every variable $x$ there is exactly one 2-path in $G$ connecting $p_x$, namely $(u_{\mathrm{lay}(x),\mathrm{up}(x)}, \mathrm{mid}(x), l_{\mathrm{lay}(x),\mathrm{low}(x)})$.

**($\mathbf{P}_4$)** For every pair of variables $x, y$ the two unique paths connecting $p_x$ and $p_y$ are edge-disjoint.

Although we are going to add more edges and vertices to $G$, none of these edges has any endpoint in $\bigcup_i L_i$, so $P_3$ will stay satisfied.

**The clause part.** The vertices of the clause part are partitioned into $O(m^{1/3})$ clusters. Similarly as in the case of variables, each clause is going to correspond to a pair of vertices in the same cluster. Again, the assignment of clauses to clusters has to be done carefully. To this end we introduce the *clause conflict graph* $G_C = (\mathrm{Cl}, E_{G_C})$. Two different clauses $C_1$ and $C_2$ are adjacent in $G_C$ if $C_1$ contains a variable $x_1$ and $C_2$ contains a variable $x_2$ such that $\mathrm{mid}(x_1) = \mathrm{mid}(x_2)$. Fix a variable $x_1$. Since $|\mathrm{mid}^{-1}(\mathrm{mid}(x_1))| \le \lceil n^{1/3} \rceil$, there are at most $\lceil n^{1/3} \rceil$ variables $x_2$ such that $\mathrm{mid}(x_1) = \mathrm{mid}(x_2)$. Since every clause contains 3 variables, and each of them is in at most 4 clauses, $\Delta(G_C) \le 12 \lceil n^{1/3} \rceil$. It follows that in polynomial time we can find a proper coloring $\beta$ of the vertices of $G_C$ into at most $12 \lceil n^{1/3} \rceil + 1$ colors. Moreover, if for any color $j$ its color class $\beta^{-1}(j)$ is larger than $\lceil n^{2/3} \rceil$ we partition it into $\lceil |\beta^{-1}(j)| / \lceil n^{2/3} \rceil \rceil$ new colors. Clearly, in total we produce at most $\frac{4}{3} \lceil n^{1/3} \rceil$ new colors in this way because $m \le \frac{4}{3} n \le \frac{4}{3} \lceil n^{1/3} \rceil \cdot \lceil n^{2/3} \rceil$. Hence, in what follows we assume that each color class of $\beta$ is of size at most $\lceil n^{2/3} \rceil$, and the total number of colors $s \le 14 \lceil n^{1/3} \rceil + 1$. In what follows we construct $s$ clusters $Q_1, \ldots, Q_s$. Every clause $C \in \mathrm{Cl}$ is going to correspond to a pair of vertices in the cluster $Q_{\beta(C)}$.

Fix $i = 1, \ldots, s$. Let us describe the subgraph induced by cluster $Q_i$. Define *cluster conflict graph* $G_i = (\beta^{-1}(i), E_{G_i})$. Two different clauses $C_1, C_2 \in \beta^{-1}(i)$ are adjacent in $G_i$ if there are three variables $x_1$, $x_2$, and $x_3$ such that (*i*) $C_1$ contains $x_1$, (*ii*) $C_2$ contains $x_2$, (*iii*) $(\mathrm{lay}(x_1), \mathrm{up}(x_1)) = (\mathrm{lay}(x_3), \mathrm{up}(x_3))$ and (*iv*) $\mathrm{mid}(x_2) = \mathrm{mid}(x_3)$. Fix a variable $x_1$ which appears in a clause $C_1 \in \beta^{-1}(i)$. By our construction, there are at most $\lceil n^{1/3} \rceil + 2$ other variables $x_3$ that map to the same pair as $x_1$ by functions lay and up. For each such $x_3$ there are at most $\lceil n^{1/3} \rceil$ variables $x_2$ such that $\mathrm{mid}(x_2) = \mathrm{mid}(x_3)$; however, at most one of these variables belongs to a clause $C_2$ from the same cluster $\beta^{-1}(i)$, by the definition of the coloring $\beta$. It follows that $\Delta(G_i) \le 12(\lceil n^{1/3} \rceil + 2)$. Hence in polynomial time we can find a proper coloring $\gamma_i$ of the vertices of $G_i$ into at most $12(\lceil n^{1/3} \rceil + 2) + 1$ colors. Similarly as in the case of the coloring $\beta$, we can assume that each of the color classes of $\gamma_i$ has at most $\lceil n^{1/3} \rceil$ clauses, at the expense of at most $\lceil n^{1/3} \rceil$ additional colors. It follows that we can construct in polynomial time a function $g : \mathrm{Cl} \to [\lceil n^{1/3} \rceil]$ such that for every cluster $i = 1, \ldots, s$ and for every color class $S$ of $\gamma_i$ $g$ is injective on $S$. Let $n_i \le 13 \lceil n^{1/3} \rceil + 25$ be the number of colors used by $\gamma_i$. For notational convenience, let us define a function $\gamma : \mathrm{Cl} \to [\max_i n_i]$ such that for any clause $C$ we have $\gamma(C) = \gamma_{\beta(C)}(C)$.

We are ready to define the vertices and edges of $Q_i$. It is a union of three disjoint vertex sets $A_i$, $B_i$, and $C_i$. We have $A_i = \{a_{i,j} \;:\; j = 1, \ldots, \lceil n^{1/3} \rceil\}$, $B_i = \{b^k_{i,j} \;:\; j = 1, \ldots, n_i, k = 1, 2, 3\}$, and $C_i = \{c_{i,j} \;:\; j = 1, \ldots, n_i\}$. For every $j = 1, \ldots, n_i$ and for every $k = 1, 2, 3$ we add edge $c_{i,j} b^k_{i,j}$ to $G$, and we color it by $c_0$ to color $F$. (These are the only edges pre-colored in the whole graph $G$.) For every clause $C \in \beta^{-1}(i)$ we do the following. For each $k = 1, 2, 3$, add the edge $(a_{i,g(C)}, b^k_{i,\gamma(C)})$ to $G$. Finally, add the pair $\{a_{i,g(C)}, c_{i,\gamma(C)}\}$ to $S$. Clearly, the following holds:

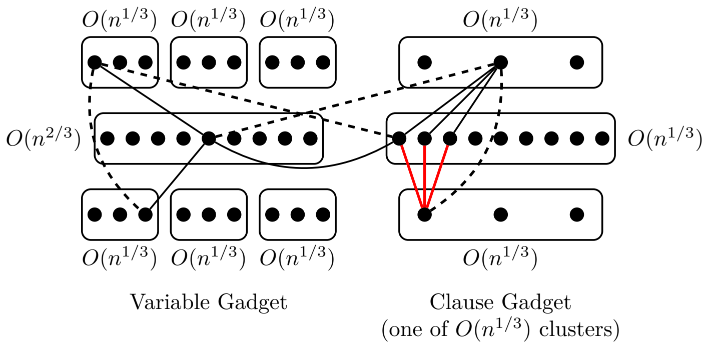


**Figure 2** A simplified view of the obtained instance. Edges (solid lines) and requests (dashed lines) representing one variable and one clause that contains this variable are presented on the picture.

**($P_5$)** Let $C$ be any clause. Let $i = \beta(C)$ and let $j = g(C)$. Then there are exactly three 2-paths between $a_{\beta(C),g(C)}$ and $c_{\beta(C),\gamma(C)}$, each going through $b^k_{\beta(C),\gamma(C)}$ for $k = 1, 2, 3$.

**Connections between the two parts.** Consider a clause $C = \{\ell_1, \ell_2, \ell_3\}$ and its $k$-th literal $\ell_k$ for each $k = 1, 2, 3$. Then for some variable $x$ we have $\ell_k = x$ or $\ell_k = \bar{x}$. We add the edge $b^k_{\beta(C),\gamma(C)}\mathrm{mid}(x)$ and we add the pair $\{\mathrm{mid}(x), a_{\beta(C),g(C)}\}$ to $S$. If $\ell_k = x$, we also add the pair $\{b^k_{\beta(C),\gamma(C)}, u_{\mathrm{lay}(x),\mathrm{up}(x)}\}$ to $S$; otherwise we add the pair $\{b^k_{\beta(C),\gamma(C)}, l_{\mathrm{lay}(x),\mathrm{low}(x)}\}$ to $S$. We claim the following.

**($P_6$)** Every edge between the two parts was added exactly once, i.e., for every edge $uv$ such that $u$ is in the clause part and $v$ is in the variable part, there is exactly one clause $C$ and exactly one literal $\ell_k \in C$ such that $u = b^k_{\beta(C),\gamma(C)}$ and $v = \mathrm{mid}(x)$, where $x$ is the variable in $\ell_k$.

Indeed, assume for a contradiction that there is a clause $C_1$ with its $k_1$-th literal containing $x_1$ and a clause $C_2$ with its $k_2$-th literal containing $x_2$ such that $b^{k_1}_{\beta(C_1),\gamma(C_1)} = b^{k_2}_{\beta(C_2),\gamma(C_2)}$ and $\mathrm{mid}(x_1) = \mathrm{mid}(x_2)$. Then $C_1 \neq C_2$ by ($P_1$). Since $\mathrm{mid}(x_1) = \mathrm{mid}(x_2)$, $C_1$ and $C_2$ are adjacent in the clause conflict graph $G_C$. It follows that $\beta(C_1) \neq \beta(C_2)$, so two different clusters share a vertex, a contradiction.

This finishes the description of the instance $(G, S, c_0)$. (See Fig. 2.)

**From an assignment to a coloring.** Let $\xi : \mathrm{Var} \to \{T, F\}$ be a satisfying assignment of $\varphi$. We claim that there is a coloring $c$ of $E(G)$ which extends $c_0$ and satisfies all pairs in $S$. We define $c$ as follows. Denote $\overline{F} = T$, $\overline{T} = F$ and $\xi(\overline{x}) = \overline{\xi(x)}$. For every variable $x \in \mathrm{Var}$ we put $c(u_{\mathrm{lay}(x),\mathrm{up}(x)}\mathrm{mid}(x)) = \xi(x)$ and $c(\mathrm{mid}(x)l_{\mathrm{lay}(x),\mathrm{low}(x)}) = \overline{\xi(x)}$. By ($P_3$) and ($P_4$) each edge is colored exactly once. Note that it satisfies all the pairs in $S$ between vertices in the variable part.

For each clause $C$ and each of its literals $\ell_k$ do the following. Let us color the edge $a_{\beta(C),g(C)}b^k_{\beta(C),\gamma(C)}$ with the color $\xi(\ell_k)$. Since $g$ is injective on color classes of $\gamma_{\beta(C)}$, after processing all the literals in all the clauses, no edge is colored more than once. Recall that for every clause $C$ we added exactly one pair to $S$, namely $\{a_{\beta(C),g(C)}, c_{\beta(C),\gamma(C)}\}$. Pick any of $C$'s satisfied literals, say $\ell_k$. Note that the pair $\{a_{\beta(C),g(C)}, c_{\beta(C),\gamma(C)}\}$ is then satisfied, because edge $a_{\beta(C),g(C)}b^k_{\beta(C),\gamma(C)}$ is colored by $T$ and $b^k_{\beta(C),\gamma(C)}c_{\beta(C),\gamma(C)}$ is colored by $F$. Hence all the pairs in $S$ between vertices in the clause part are satisfied.

Now let us color the edges between the clause part and the variable part. Consider any such edge $uv$, i.e., $u$ is in the clause part and $v$ is in the variable part. By ($P_6$), there is exactly one clause $C$ and exactly one literal $\ell_k \in C$ such that $u = b^k_{\beta(C),\gamma(C)}$ and $v = \mathrm{mid}(x)$, where $x$ is the variable in $\ell_k$. Color the edge $b^k_{\beta(C),\gamma(C)}\mathrm{mid}(x)$ with the color $\overline{\xi(\ell_k)}$. Then the pair $\{\mathrm{mid}(x), a_{\beta(C),g(C)}\}$ is satisfied by the path $(\mathrm{mid}(x), b^k_{\beta(C),\gamma(C)}, a_{\beta(C),g(C)})$, since $c(b^k_{\beta(C),\gamma(C)}a_{\beta(C),g(C)}) = \xi(\ell_k)$. Assume $\ell_k = x$. Then the pair $\{b^k_{\beta(C),\gamma(C)}, u_{\mathrm{lay}(x),\mathrm{up}(x)}\}$ is satisfied by the path $(b^k_{\beta(C),\gamma(C)}, \mathrm{mid}(x), u_{\mathrm{lay}(x),\mathrm{up}(x)})$, since its first edge is colored by $\overline{\xi(\ell_k)} = \overline{\xi(x)}$ and its second edge is colored by $\xi(x)$. Analogously, when $\ell_k = \bar{x}$, then the pair $\{b^k_{\beta(C),\gamma(C)}, l_{\mathrm{lay}(x),\mathrm{low}(x)}\}$ is satisfied by the path $(b^k_{\beta(C),\gamma(C)}, \mathrm{mid}(x), l_{\mathrm{lay}(x),\mathrm{low}(x)})$, since its first edge is colored by $\overline{\xi(\ell_k)} = \xi(x)$ and its second edge is colored by $\overline{\xi(x)}$.

It follows that we colored all the edges and all the pairs in $S$ are satisfied, so $(G, S, c_0)$ is a YES-instance, as required.

**From a coloring to an assignment.** Let $c : E(G) \to \{T, F\}$ be a coloring which extends $c_0$ and satisfies all pairs in $S$. Consider the following variable assignment: for every $x \in \mathrm{Var}$, we put $\xi(x) = c(u_{\mathrm{lay}(x),\mathrm{up}(x)}\mathrm{mid}(x))$. We claim that $\xi$ satisfies all the clauses of $\varphi$. Consider an arbitrary clause $C = \{\ell_1, \ell_2, \ell_3\}$.

Since the pair $\{a_{\beta(C),g(C)}, c_{\beta(C),\gamma(C)}\}$ is satisfied, there is a 2-color 2-path $P$ between $a_{\beta(C),g(C)}$ and $c_{\beta(C),\gamma(C)}$. Recall that $N(c_{\beta(C),\gamma(C)}) = \{b^k_{\beta(C),\gamma(C)} : k = 1, 2, 3\}$, so there is $k = 1, 2, 3$ such that $b^k_{\beta(C),\gamma(C)}$ is the internal vertex on $P$. Since $c$ extends $c_0$ and $c_0(b^k_{\beta(C),\gamma(C)}c_{\beta(C),\gamma(C)}) = F$, we infer that $c(a_{\beta(C),g(C)}b^k_{\beta(C),\gamma(C)}) = T$. Let $x$ be the variable in the literal $\ell_k$.

Since the pair $\{\mathrm{mid}(x), a_{\beta(C),g(C)}\}$ is satisfied, there is a 2-color 2-path $Q$ between $\mathrm{mid}(x)$ and $a_{\beta(C),g(C)}$. Then the internal vertex of $Q$ is $b^{k'}_{\beta(C'),\gamma(C')}$, for some clause $C'$ and integer $k' = 1, 2, 3$. Let $y$ be the variable in the $k'$-th literal of $C'$. Since there is an edge between $\mathrm{mid}(x)$ and $b^{k'}_{\beta(C'),\gamma(C')}$, from ($P_6$) we infer that $\mathrm{mid}(y) = \mathrm{mid}(x)$. If $C = C'$ and $k' \neq k$, then by ($P_1$) we get that $\mathrm{mid}(x) \neq \mathrm{mid}(y)$, a contradiction. If $C \neq C'$, since $\mathrm{mid}(y) = \mathrm{mid}(x)$, the clauses $C$ and $C'$ are adjacent in the clause conflict graph $G_C$, so $\beta(C') \neq \beta(C)$. However, then the edge $b^{k'}_{\beta(C'),\gamma(C')}a_{\beta(C),g(C)}$ of $Q$ goes between two clusters, a contradiction. Hence $C' = C$ and $k' = k$, i.e., $Q = (\mathrm{mid}(x), b^k_{\beta(C),\gamma(C)}, a_{\beta(C),g(C)})$. Since $c(b^k_{\beta(C),\gamma(C)}a_{\beta(C),g(C)}) = T$, we get $c(\mathrm{mid}(x)b^k_{\beta(C),\gamma(C)}) = F$. Now assume w.l.o.g. that $\ell_k = x$, the case $\ell_k = \bar{x}$ is analogous.

Since the pair $\{b^k_{\beta(C),\gamma(C)}, u_{\mathrm{lay}(x),\mathrm{up}(x)}\}$ is satisfied, there is a 2-color 2-path $R$ between $b^k_{\beta(C),\gamma(C)}$ and $u_{\mathrm{lay}(x),\mathrm{up}(x)}$. Then the internal vertex $z$ of $R$ belongs to $M$. By ($P_6$) there is a literal $\ell_k$ which belongs to a clause $C_2$ and contains a variable $x_2$ such that $z = \mathrm{mid}(x_2)$ and $b^k_{\beta(C),\gamma(C)} = b^k_{\beta(C_2),\gamma(C_2)}$. In particular, $\beta(C) = \beta(C_2)$ and $\gamma(C) = \gamma(C_2)$. Assume $C_2 \neq C$. There is a variable, say $x_3$, corresponding to edge $\mathrm{mid}(x_2)u_{\mathrm{lay}(x),\mathrm{up}(x)}$, i.e., $\mathrm{mid}(x_2) = \mathrm{mid}(x_3)$ and $u_{\mathrm{lay}(x),\mathrm{up}(x)} = u_{\mathrm{lay}(x_3),\mathrm{up}(x_3)}$. It follows that $C$ and $C_2$ are adjacent in $G_{\beta(C)}$, which contradicts the fact that $\gamma(C) = \gamma(C_2)$. Hence $C_2 = C$, i.e., there is exactly one 2-path between $b^k_{\beta(C),\gamma(C)}$ and $u_{\mathrm{lay}(x),\mathrm{up}(x)}$, and it goes through $\mathrm{mid}(x)$. Since $c(\mathrm{mid}(x)b^k_{\beta(C),\gamma(C)}) = F$ and the path is 2-color, we get that $c(u_{\mathrm{lay}(x),\mathrm{up}(x)}\mathrm{mid}(x)) = T$. Hence $\xi(\ell_k) = \xi(x) = T$, so clause $C$ is satisfied, as required.

It finishes the proof. (The analysis of the size of the resulting instance and its other properties described in the claim is not immediate; because of the space constraints we skip them here.) ◀

The proof of the following lemma is non-trivial, but standard (see lemmas 4 and 5 in the full version [17].)

▶ **Lemma 9** (★). *For any fixed $k \geq 2$, there is a polynomial time algorithm which given an instance $I = (G = (V, E), S, c_0)$ of* Subset Rainbow 2-Coloring Extension *constructs an equivalent instance $(G' = (V', E'), S')$ of* Subset Rainbow $k$-Coloring *such that $|V'| = O(k|V|k^2\ell)$, $|E'| = |E| + O(k|V|) + |\mathrm{Dom}(c_0)| + O(k^2\ell)$, $|S'| = |S| + |E| + 2|\mathrm{Dom}(c_0)| + O(k^2\ell)$. Let $G_S = (V, S)$ and $G_{S'} = (V', S')$. Then $\Delta(G_{S'}) = O(\Delta(G_S) + \Delta(G) + |\mathrm{Dom}(c_0)|/\ell)$. Moreover if we are given a proper vertex $p$-coloring of the graph $G_S = (V, S)$ then we can output also a proper vertex $(p + 3)$-coloring of the graph $G_{S'} = (V', S')$.*

## 2.3 From Subset Rainbow $k$-Coloring to Rainbow $k$-Coloring

The basic idea of our reduction from Subset Rainbow $k$-Coloring to Rainbow $k$-Coloring is to modify the graph so that the pairs of vertices from $\bar{E} \setminus S$ can be somehow trivially satisfied, without affecting the satisfiability of $S$. To this end we use a notion of biclique covering number (called also bipartite dimension). The *biclique covering number* $\mathrm{bc}(G)$ of a graph $G$ is the smallest number of biclique subgraphs of $G$ that cover all edges of $G$. The following proposition is well-known.

▶ **Proposition 10** (Folklore). *It holds that $\mathrm{bc}(K_n) = \lceil \log n \rceil$, and the corresponding cover can be constructed in polynomial time.*

**Proof.** Assume $V(K_n) = \{0, \ldots, n-1\}$. The $i$-th biclique contains edges between the vertices that have 0 at the $i$-th bit and the vertices that have 1 at the $i$-th bit. ◀

Let $G = (V_1, V_2, E)$ be a bipartite graph. Then $\hat{G}$ denotes the bipartite complement of $G$, i.e, the bipartite graph $(V_1, V_2, \{v_1 v_2 \,:\, v_1 \in V_1,\, v_2 \in V_2, \text{ and } v_1 v_2 \notin E\})$. We will use the following result of Jukna. Recall that we denote $\Delta_1(G) = \max\{\Delta(G), 1\}$.

▶ **Theorem 11** (Jukna [16]). *If $G$ is an $n$-vertex bipartite graph, then $\mathrm{bc}(\hat{G}) = O(\Delta_1(G) \log n)$.*

Let us call the cover from Theorem 11 the *Jukna cover*. In our application we need to be able to *compute* the Jukna cover fast.

▶ **Lemma 12.** *The Jukna cover can be constructed in (i) expected polynomial time, or (ii) deterministic $2^n n^{O(1)}$ time.*

**Proof.** Denote $\Delta = \Delta(G)$. If $\Delta = 0$ the claim follows from Proposition 10, so in what follows assume $\Delta \geq 1$. Jukna [16] shows a simple worst-case linear time algorithm which samples a biclique in $G$. Then it is proved that after sampling $t$ bicliques, the probability that there is an edge not covered by one of the bicliques is at most $n^2 e^{-t/(\Delta e)}$. It follows that the probability that more than $\Delta e(2 \ln n + 1)$ samples are needed is at most $e^{-1}$. If after $\Delta e(2 \ln n + 1)$ samples some edges is not covered, we discard all the bicliques found and repeat the whole algorithm from the scratch. The expected number of such restarts is $1/(1 - e^{-1}) = O(1)$.

Now we proceed to the second part of the claim. Let $G = (V_1, V_2, E)$. For every subset $A \subseteq V_1$ we define the biclique $B_A = (A, B, E_A)$, where $B$ is the set of vertices of $V_2$ adjacent in $\hat{G}$ to all vertices of $A$. Clearly, $B_A$ is a subgraph of $\hat{G}$ and for every subset $A \subseteq V_1$ it can be found in time linear in the size of $\hat{G}$. Our deterministic algorithm works as follows: as long as not all edges of $\hat{G}$ are covered, it picks the biclique $B_A$ which maximizes the number of new covered edges of $\hat{G}$. Since all the bicliques in the set $\{B_A \,:\, A \subseteq V_1\}$ can be listed in time $O(2^n|E(\hat{G})|)$, the total running time is $t2^n n^{O(1)}$, where $t$ is the size of the returned cover. It suffices to show that $t = O(\Delta \log n)$.

Jukna [16] shows that if set $A$ is chosen by picking every vertex of $V_1$ independently with probability $\frac{1}{\Delta}$, then for any edge $uv \in E(\hat{G})$, $\Pr[uv \in E_A] \geq \frac{1}{\Delta e}$. Consider any step of our algorithm and let $R \subseteq E(\hat{G})$ be the set of the edges of $\hat{G}$ which are not covered yet. By the bound on $\Pr[uv \in E_A]$ and the linearity of expectation a set $A$ sampled as described above covers at least $|R|/(\Delta e)$ new edges in expectation. In particular, it implies that there exists a set $A \subseteq V_1$ that covers at least $|R|/(\Delta e)$ new edges. Let $\alpha = (1 - \frac{1}{\Delta e})^{-1}$. By the Taylor expansion of $\log(1-x)$, it follows that $t = O(\log_\alpha |E(\hat{G})|) = O(\log n/ \log \alpha) = O(\Delta \log n)$. ◀

▶ **Lemma 13.** *Let $G$ be an $n$-vertex graph with a given proper vertex $p$-coloring. Then the edges of $\bar{G}$ can be covered by $O(p^2 \Delta_1(G) \log n)$ bicliques from $\bar{G}$ so that any edge of $G$ and any biclique have at most one common vertex. This cover can be constructed in (i) expected polynomial time, or (ii) deterministic $2^n n^{O(1)}$ time.*

**Proof.** The edges of $\bar{G}$ between the vertices of any color class form a clique, so by Proposition 10 we can cover its edges using $O(\log n)$ bicliques. If an edge of $G$ has both endpoints in such a biclique, these endpoints have the same color, contradiction. For two different colors $i$ and $j$ the edges of $G$ between their color classes form a bipartite graph of maximum degree at most $\Delta(G)$. Hence by Lemma 12 we can cover the edges of its bipartite complement using $O(\Delta_1(G) \log n)$ bicliques. If an edge $uv$ of $G$ has both endpoints in such a biclique, then either (i) these endpoints have the same color, contradiction, or (ii) these endpoints belong to two different parts of the biclique, so $uv$ is in the biclique and hence $uv \in E(\bar{G})$, a contradiction. Summing over all color classes and pairs of color classes, we use $O(p^2 \Delta_1(G) \log n)$ bicliques, as required. ◀

Now we proceed to the actual reduction.

▶ **Lemma 14.** *Given an instance $(G = (V, E), S)$ of* Subset Rainbow 2-Coloring *together with a proper $p$-coloring of the graph $G_S = (V, S)$, one can construct an equivalent instance $G'$ of* Rainbow 2-Coloring *such that $|V(G')| = O(|V| + kp^2 \Delta_1(G_S) \log |V|)$, $|E(G')| = O(|E(G)| + (|V| + p^2 \Delta_1(G_S) \log |V|) \cdot p^2 \Delta_1(G_S) \log |V|)$. The construction algorithm can run in (i) expected polynomial time, or (ii) deterministic $2^{|V|}|V|^{O(1)}$ time.*

**Proof.** Here we focus on the $k = 2$ case. The $k \geq 3$ case is significantly more technical — see the details in the full version. Let us consider a biclique covering of the complement of the graph $G_S$ with $q = O(p^2 \Delta_1(G_S) \log n)$ bicliques $(U_1, V_1; E_1), (U_2, V_2; E_2), \ldots, (U_q, V_q; E_q)$ as in Lemma 13. Let $W = \{w_1, w_2, \ldots, w_q\}$, $T = \{t_1, t_2, t_3\}$, $V(G') = V \cup W \cup T$ and $E(G') = E(G) \cup (W \times W) \cup (T \times T) \cup (\{t_2\} \times W) \cup (\{t_3\} \times (V \cup W)) \cup \left(\bigcup_{1 \leq i \leq q} \{w_i\} \times (U_i \cup V_i)\right)$ (we abuse the notation assuming that $\times$ operator returns *unordered* pairs minus loops). Because of the space limitation the equivalence proof is deferred to the full version. ◀

## 2.4 Putting everything together

By pipelining lemmas 7, 8, and 9 we get the following corollary.

▶ **Corollary 15.** *Fix $k \geq 2$. Given a* 3-SAT *formula $\varphi$ with $m$ clauses one can construct in polynomial time an equivalent instance $(G = (V, E), S)$ of* Subset Rainbow $k$-Coloring *such that $|V| = O(m^{2/3})$, $|E| = O(m)$, $\Delta((V, S)) = O(m^{1/3})$, and the graph $G_S = (V, S)$ is $O(1)$-colorable.*

Note that in Corollary 15 we have $|S| = |V|\Delta((V, S)) = O(m)$. It follows that the Sparsification Lemma (Lemma 6) and Corollary 15 imply Theorem 3.

Pipelining Corollary 15 and Lemma 14 gives the following corollary.

▶ **Corollary 16.** *Fix $k \geq 2$. Given a 3-SAT formula $\varphi$ with $O(m)$ clauses one can construct an equivalent instance $G$ of* RAINBOW $k$-COLORING *with $O(m^{2/3})$ vertices and $O(m \log m)$ edges. The construction algorithm can run in (i) expected polynomial time, or (ii) deterministic $2^{O(m^{2/3})}$ time.*

Again, the above and the Sparsification Lemma immediately imply Theorem 2.

## 3 Algorithms for Subset Rainbow $k$-Coloring

In this section we study FPT algorithms for SUBSET RAINBOW $k$-COLORING parameterized by $|S|$. We provide two such algorithms, based on different approaches: one for $k = 2$ case, and one (slightly slower) for the general case. Consider an instance $(G, S)$ of the SUBSET RAINBOW $k$-COLORING problem. Note that we can assume that $S \subseteq \bar{E}$, since any constraint $\{u, v\} \in E$ is satisfied in every edge coloring. Moreover, we say that a pair $\{u, v\}$ is *feasible* when the distance between $u$ and $v$ is at most $k$. The set of all feasible pairs is denoted by $F(G)$. Clearly, when $S$ contains a request which is not feasible, then $(G, S)$ is a trivial NO-instance. Hence, throughout this section we assume $S \subseteq \bar{E} \cap F(G)$.

### 3.1 The $k = 2$ case

For any $X \subseteq S$ let $\mathcal{P}_X$ be the set of all 2-edge paths between the pairs of vertices in $X$. Denote $E(\mathcal{P}_X) = \bigcup_{P \in \mathcal{P}_X} E(P)$. For two edges $e_1, e_2 \in E(G)$ we say that $e_1$ and $e_2$ are *linked by $X$*, denoted as $e_1 \sim_X e_2$ when there are two paths $P_1, P_2 \in \mathcal{P}_X$ (possibly $P_1 = P_2$) such that $e_1 \in E(P_1)$, $e_2 \in E(P_2)$ and $E(P_1) \cap E(P_2) \neq \emptyset$. Let $\approx_X$ be the transitive closure of $\sim_X$. Then $\approx_X$ is an equivalence relation. Recall that $E(G)/\approx_X$ denotes the quotient set of the relation $\approx_X$. The main observation of this section is the following theorem.

▶ **Theorem 17.** *The number of 2-colorings of $E(G)$ that satisfy all the pairs in $S$ is equal to $\sum_{X \subseteq S} (-1)^{|X|} 2^{|E(G)/\approx_X|}$.*

In the proof we make use of the well-known inclusion-exclusion principle. Below we state it in the intersection version (see, e.g., [10])

▶ **Theorem 18** (Inclusion–exclusion principle, intersection version)**.** *Let $A_1, \ldots, A_n \subseteq U$, where $U$ is a finite set. Denote $\bigcap_{i \in \emptyset}(U \setminus A_i) = U$. Then*

$$|\bigcap_{i \in [n]} A_i| = \sum_{X \subseteq [n]} (-1)^{|X|} |\bigcap_{i \in X} (U \setminus A_i)|.$$

**Proof of Theorem 17.** Let us define, for every pair $\{u, v\} \in S$ (say, $u < v$), the set $A_{u,v}$ of 2-edge colorings of $G$ that satisfy $\{u, v\}$. Note that the number of rainbow 2-colorings of $G$ that satisfy all the pairs in $S$ is equal to $|\bigcap_{\{u,v\} \in S} A_{u,v}|$. By Theorem 18 it suffices to show that, for any subset $X \subseteq S$, the number $\#_X$ of 2-colorings such that *none* of the pairs in $X$ is satisfied, equals $2^{|E(G)/\approx_X|}$.

Fix any coloring $c$ that does not satisfy any pair from $X$. Then every path from $\mathcal{P}_X$ has both edges of the same color. Hence, for two edges $e_1, e_2 \in E(G)$, if $e_1 \sim_X e_2$ then $e_1$ and $e_2$ are colored by $c$ with the same color. It follows that for any equivalence class $A$ of $\approx_X$, all edges of $A$ are have the same color in $c$. This proves that $\#_X \leq 2^{|E(G)/\approx_X|}$.

For every function $c_0 : (E(G)/\approx_X) \to \{1,2\}$ we can define the coloring $c : E(G) \to \{1,2\}$ by putting $c(e) = c_0([e]_{\approx_X})$ for every edge $e \in E(G)$. (Note that the edges that do not belong to any path in $\mathcal{P}_X$ form singleton equivalence classes.) Then, $c$ does not satisfy any pair from $X$, because if some pair $\{u,v\}$ is satisfied then there is a 2-color path $uxv$; but $ux \sim_X xv$, so $[ux]_{\approx_X} = [xv]_{\approx_X}$ and $c(ux) = c(xv)$, a contradiction. It follows that $\#_X \geq 2^{|E(G)/\approx_X|}$. ◀

Since it is a standard exercise to compute the relation $X \subseteq S$ in $O(|E| + |S| \cdot |V|)$ time (see the full version), we get the following corollary. (Let us remark here that the algorithm from Corollary 19 only decides whether the coloring exists, without finding it. However, by a minor modification of the algorithm it can construct the coloring; see the full version.)

▶ **Corollary 19.** *For any graph $G = (V,E)$ and a set of requests $S$ the number of 2-colorings of $E$ that satisfy all the pairs in $S$ can be computed in $O(2^{|S|}(|E| + |S| \cdot |V|))$ time and polynomial space. In particular,* Subset Rainbow 2-Coloring *can be decided within the same time.*

## 3.2 The general case

In this section we use partial colorings. For convenience, a partial coloring is represented as a function $c : E \to [k] \cup \{\perp\}$, where the value $\perp$ corresponds to an uncolored edge. By $\mathrm{Dom}(c)$ we denote the domain of the corresponding partial function, i.e., $\mathrm{Dom}(c) = c^{-1}([k])$. The partial coloring which does not color anything, i.e., is constantly equal to $\perp$ is denoted by $c_\perp$.

For a graph $G = (V,E)$ consider a partial edge coloring $c : E \to [k] \cup \{\perp\}$. A *guide function* is any function of the form $f : S \to \binom{\mathrm{Dom}(c)}{\leq k}$, i.e., any function that assigns sets of at most $k$ colored edges to all requests in $S$. A constant guide function equal to $\emptyset$ for every request in $S$ is denoted by $g_{S,\emptyset}$. Pick any pair $\{u,v\} \in S$. We say that a walk $W$ connecting $u$ and $v$ is $f$-guided if every color appears at most once on $W$, and $f(\{u,v\}) \subseteq E(W)$. We say that a coloring $c$ is $(f,S)$-rainbow when for every pair $\{u,v\} \in S$ there is an $f$-guided walk between $u$ and $v$. Note that $(G,S)$ is a YES-instance of Subset Rainbow $k$-Coloring iff there is an $(g_{S,\emptyset}, S)$-rainbow coloring. Indeed, every rainbow walk contains a rainbow path.

The following lemma is going to be useful in our branching algorithm.

▶ **Lemma 20.** *Let $G = (V,E)$ be a graph, and let $S$ be a set of requests. Let $c_0 : E \to [k]$ be a partial edge coloring and let $f : S \to \binom{\mathrm{Dom}(c_0)}{\leq k}$ be a guide function. Then, given a pair $\{u,v\} \in S$ in time $2^k n^{O(1)}$ one can find an $f$-guided $u$-$v$ walk of length at most $k$, if it exists.*

**Proof.** The algorithm is as follows. We can assume that $f(\{u,v\})$ does not contain two edges of the same color, for otherwise the requested walk does not exist. For every $e \in f(\{u,v\})$ we remove all the edges of color $c_0(e)$. Next, we put back edges of $f(\{u,v\})$. Then it suffices to find in the resulting graph $G'$ any $u$-$v$ path of length at most $k$ and with no repeated colors that visits all the colors of the edges in $f(\{u,v\})$. This is done using dynamic programming. For every vertex $x \in V$, subset $X \subseteq [k]$ and integer $\ell = 0, \ldots, k$ we find the boolean value $T[x, X, \ell]$ which is true iff there is a $u$-$x$ walk of length $\ell$ which does not repeat colors and visits all the colors from $X$, but not more. We initialize $T[u, \emptyset, 0] = \texttt{true}$ and $T[x, \emptyset, 0] = \texttt{false}$ for every $x \neq u$. Next we iterate through the remaining triples $(x, X, \ell)$, in the nondecreasing order of $\ell$ and $X$'s cardinalities. The value of $T[x, X, \ell]$ is then computed using the formula

$$T[x, X, \ell] = \bigvee_{yx \in c_0^{-1}(X \cup \{\perp\}) \cap E(G')} T[y, X \setminus \{c_0(yx)\}, \ell - 1].$$

**Pseudocode 1:** FindColoring$(S_0, c_0, f)$

```
if $S_0 = \emptyset$ then
    return $c_0$
if for some $r \in S_0$ there are edges $e_1, e_2 \in f(r)$ with $c_0(e_1) = c_0(e_2)$ then
    return null
Pick any $\{u, v\} \in S_0$;
Find any $f$-guided $u$-$v$ walk $W$ of length at most $k$ using Lemma 20;
if $W$ does not exist then
    return null
Let $c_1$ be obtained from $c_0$ by coloring the uncolored edges of $W$ to get a rainbow walk;
if FindColoring$(S_0 \setminus \{u, v\}, c_1, f|_{S_0 \setminus \{u,v\}}) \neq$ null then return the coloring found;
for $e \in E(W) \setminus \mathrm{Dom}(c_0)$ do
    for $\alpha \in [k]$ do
        for $r \in S_0 \setminus \{\{u, v\}\}$ do
            Let $c_{e,\alpha}$ be obtained from $c_0$ by coloring $e$ with $\alpha$;
            Let $f_{e,r}$ be obtained from $f$ by putting $f(r) := f(r) \cup \{e\}$;
            if FindColoring$(S_0, c_{e,\alpha}, f_{e,r}) \neq$ null then return the coloring found;
return null
```

The requested walk exists iff $T[v, X, \ell] = \texttt{true}$ for any $\ell = 0, \ldots, k$ and $X$ such that $c_0(f(\{u, v\})) \subseteq X$. The walk is retrieved using standard DP methods. ◀

Now we are ready to describe our branching algorithm. Let $(G = (V, E), S)$ be the input instance. Our algorithm consists of a recursive procedure FindColoring which gets three parameters: $S_0$ (a set of requests), $c_0 : E \to [k] \cup \{\perp\}$ (a partial coloring), and a guide function $f : S \to \binom{\mathrm{Dom}(c_0)}{\leq k}$. It is assumed that for every request $r \in S$, every pair of different edges $e_1, e_2 \in f(r)$ is colored differently by $c_0$. The goal of the procedure FindColoring is to find an $(f, S_0)$-rainbow coloring $c : E \to [k]$ which extends $c_0$. Thus the whole problem is solved by invoking FindColoring$(S, c_\perp, g_{S,\emptyset})$. A rough description of FindColoring is as follows. We pick any pair $\{u, v\} \in S_0$ and we find any $f$-guided $u$-$v$ walk $W$ of length at most $k$ using Lemma 20. Let $c_1$ be obtained from $c_0$ by coloring the uncolored edges of $W$ to get a rainbow walk. If FindColoring$(S_0 \setminus \{u, v\}, c_1, f|_{S_0 \setminus \{u,v\}})$ returns a coloring, we are done. But if no such coloring exists then we know that we made a wrong decision: coloring some of the uncolored edges $e$ of $W$ into $c_1(e)$ (instead of some color $\alpha$) makes some other request $r \in S_0 \setminus \{\{u, v\}\}$ impossible to satisfy. For every possible triple $(e, \alpha, r)$ we invoke FindColoring with the same set of requests $S_0$, partial coloring $c_0$ extended by coloring $e$ with $\alpha$, and the guide function $f$ extended by putting $f(r) := f(r) \cup \{e\}$.

A precise description of procedure FindColoring can be found in Pseudocode 1. The following lemma proves its correctness.

▶ **Lemma 21.** *Procedure* FindColoring *invoked with parameters $(S_0, c_0, f)$ finds an $(f, S_0)$-rainbow coloring $c : E \to [k]$ which extends $c_0$, whenever it exists.*

**Proof.** The proof is by induction on the sum of $|S_0|$ and the number of uncolored edges. It is clear that if $|S_0| = 0$ or all the edges are colored then the algorithm behaves correctly. In the induction step, the only non-trivial thing to check is whether any of the calls in lines 10 or 16 returns a coloring, provided that there is a solution, i.e., an $(f, S_0)$-rainbow coloring $c : E \to [k]$ which extends $c_0$. Assume that no coloring is returned in Line 16. Then for every edge $e \in E(W) \setminus \mathrm{Dom}(c_0)$, and request $r \in S_0 \setminus \{\{u, v\}\}$ coloring $c$ is not a $(f_{e,r}, S_0)$-rainbow coloring, for otherwise the call FindColoring$(S_0, c_{e,c(e)}, f_{e,r})$ returns a coloring. If follows that for every edge $e \in E(W) \setminus \mathrm{Dom}(c_0)$ and request $r \in S_0 \setminus \{\{u, v\}\}$ the walk that realizes the request $r$ in the coloring $c$ does not contain $e$. Hence, the following coloring

$$c'(e) = \begin{cases} c(e) & \text{if } e \notin E(W), \\ c_1(e) & \text{if } e \in E(W). \end{cases}$$

is another $(f, S_0)$-rainbow coloring, and it extends $c_1$. It follows that the call in Line 10 returns a coloring, as required. ◀

▶ **Theorem 22.** *For every integer $k$, there is an FPT algorithm for* Subset Rainbow $k$-Coloring *parameterized by $|S|$. The algorithm runs in time $(k^2|S|)^{k|S|}2^k n^O(1)$, in particular in $|S|^{O(|S|)}n^{O(1)}$ time for every fixed $k$.*

**Proof.** By Lemma 21 Subset Rainbow $k$-Coloring is solved by invoking FindColoring( $S, c_\perp, g_{S,\emptyset}$). Note that whenever we go deeper in the recursion either some request of $S_0$ gets satisfied, or $|f(r)|$ increases for some $r \in S_0$. When $|f(r)|$ increases to $k+1$, the corresponding recursive call returns `null` immediately (because the condition in Line 3 holds). It follows that the depth of the recursion is at most $|S|k$. Since in every call of Subset Rainbow $k$-Coloring the algorithm uses time $2^k n^{O(1)}$ (by Lemma 21) and branches into at most $1 + k^2(|S| - 1) \leq k^2|S|$ recursive calls, the total time is $(k^2|S|)^{k|S|}2^k n^O(1)$, as required. ◀

## 4 Further Work

We believe that this work only initiates the study of fine-grained complexity of variants of Rainbow $k$-Coloring. In particular, many open questions are still unanswered. The ultimate goal is certainly to get tight bounds. We pose the following two conjectures.

▶ **Conjecture 23.** *For any integer $k \geq 2$, there is no $2^{o(|E|)}n^{O(1)}$-time algorithm for* Rainbow $k$-Coloring*, unless ETH fails.*

▶ **Conjecture 24.** *For any integer $k \geq 2$, there is no $2^{o(n^2)}n^{O(1)}$-time algorithm for* Rainbow $k$-Coloring*, unless ETH fails.*

Note that in this work we have settled Conjecture 23 for Subset Rainbow $k$-Coloring, and for Rainbow $k$-Coloring we showed a slightly weaker, $2^{o(|E|/\log|E|)}n^{O(1)}$ bound. However, avoiding this $\log|E|$ factor seems to constitute a considerable technical challenge.

In this paper we gave two algorithms for Subset Rainbow $k$-Coloring parameterized by $|S|$, one working in $2^{|S|}n^{O(1)}$ time for $k = 2$ and another, working in time $|S|^{O(|S|)}n^{O(1)}$ for every fixed $k$. We conjecture that there exists an algorithm running in time $2^{O(|S|)}n^{O(1)}$ for every fixed $k$.

Finally, we would like to propose yet another parameterization of Rainbow $k$-Coloring. Assume we are given a graph $G = (V, E)$ and a subset of vertices $S \subseteq V$. In the Steiner Rainbow $k$-Coloring problem the goal is to determine whether there is a rainbow $k$-coloring such that every pair of vertices in $S$ is connected by a rainbow path. By our Theorem 2, Steiner Rainbow $k$-Coloring has no algorithm running in time $2^{o(|S|^{3/2})}$, under ETH. On the other hand, our algorithm for Subset Rainbow $k$-Coloring implies that Steiner Rainbow $k$-Coloring parameterized by $|S|$ admits an FPT algorithm with running time of $2^{O(|S|^2\log|S|)}n^{O(1)}$. It would be interesting make the gap between these bounds smaller.

## References

1 Prabhanjan Ananth, Meghana Nasre, and Kanthi K. Sarpatwar. Rainbow connectivity: Hardness and tractability. In *IARCS Annual Conference on Foundations of Software Technology and Theoretical Computer Science (FSTTCS 2011)*, pages 241–251, 2011.

2 Yair Caro, Arie Lev, Yehuda Roditty, Zsolt Tuza, and Raphael Yuster. On rainbow connection. *Electron. J. Combin*, 15(1):R57, 2008.

3 Sourav Chakraborty, Eldar Fischer, Arie Matsliah, and Raphael Yuster. Hardness and algorithms for rainbow connection. *J. of Combinatorial Optimization*, 21(3):330–347, 2009.

4 L. Sunil Chandran and Deepak Rajendraprasad. Rainbow Colouring of Split and Threshold Graphs. *Computing and Combinatorics*, pages 181–192, 2012.

5 L. Sunil Chandran and Deepak Rajendraprasad. Inapproximability of rainbow colouring. In *IARCS Annual Conference on Foundations of Software Technology and Theoretical Computer Science (FSTTCS 2013)*, pages 153–162, 2013.

6 L. Sunil Chandran, Deepak Rajendraprasad, and Marek Tesař. Rainbow colouring of split graphs. *Discrete Applied Mathematics*, 2015. doi:10.1016/j.dam.2015.05.021.

7 Gary Chartrand, Garry L. Johns, Kathleen A. McKeon, and Ping Zhang. Rainbow connection in graphs. *Mathematica Bohemica*, 133(1), 2008.

8 Gary Chartrand and Ping Zhang. *Chromatic graph theory.* CRC press, 2008.

9 Marek Cygan, Fedor V. Fomin, Alexander Golovnev, Alexander S. Kulikov, Ivan Mihajlin, Jakub Pachocki, and Arkadiusz Socała. Tight bounds for graph homomorphism and subgraph isomorphism. In *Proc. of the 27th Annual ACM-SIAM Symp. on Discrete Algorithms, SODA 2016, Arlington, VA, USA, January 10-12, 2016*, pages 1643–1649, 2016.

10 Marek Cygan, Fedor V Fomin, Łukasz Kowalik, Dániel Lokshtanov, Daniel Marx, Marcin Pilipczuk, Michał Pilipczuk, and Saket Saurabh. *Parameterized Algorithms.* Springer, 2015.

11 Reinhard Diestel. *Graph Theory.* Springer-Verlag Heidelberg, 2010.

12 Eduard Eiben, Robert Ganian, and Juho Lauri. On the complexity of rainbow coloring problems. In *Proceedings of the Twenty-Sixth International Workshop on Combinatorial Algorithms, IWOCA 2015, Verona, Italy, October 5-7*, pages 209–220, 2015. URL: http://arxiv.org/abs/1510.03614, doi:10.1007/978-3-319-29516-9_18.

13 Michael Held and Richard M. Karp. A dynamic programming approach to sequencing problems. *Journal of SIAM*, 10:196–210, 1962.

14 Russell Impagliazzo and Ramamohan Paturi. On the Complexity of k-SAT. *J. Comput. Syst. Sci.*, 62(2):367–375, 2001. doi:10.1006/jcss.2000.1727.

15 Russell Impagliazzo, Ramamohan Paturi, and Francis Zane. Which problems have strongly exponential complexity? *J. Comput. Syst. Sci.*, 63(4):512–530, 2001. doi:10.1006/jcss.2001.1774.

16 Stasys Jukna. On set intersection representations of graphs. *Journal of Graph Theory*, 61(1):55–75, 2009. doi:10.1002/jgt.20367.

17 Lukasz Kowalik, Juho Lauri, and Arkadiusz Socala. On the fine-grained complexity of rainbow coloring. *CoRR*, abs/1602.05608, 2016. URL: http://arxiv.org/abs/1602.05608.

18 Eugene L. Lawler. A note on the complexity of the chromatic number problem. *Information Processing Letters*, 5(3):66–67, 1976.

19 Van Bang Le and Zsolt Tuza. Finding optimal rainbow connection is hard. Technical Report CS-03-09, Universität Rostock, 2009.

20 Xueliang Li, Yongtang Shi, and Yuefang Sun. Rainbow Connections of Graphs: A Survey. *Graphs and Combinatorics*, 29(1):1–38, 2012.

21 Arkadiusz Socała. Tight lower bound for the channel assignment problem. In *Proceedings of the Twenty-Sixth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2015, San Diego, CA, USA, January 4-6, 2015*, pages 662–675, 2015.

22 Craig A. Tovey. A simplified NP-complete satisfiability problem. *Discrete Applied Mathematics*, 8(1):85–89, 1984. doi:10.1016/0166-218X(84)90081-7.

23 Kei Uchizawa, Takanori Aoki, Takehiro Ito, Akira Suzuki, and Xiao Zhou. On the Rainbow Connectivity of Graphs: Complexity and FPT Algorithms. *Algorithmica*, 67(2):161–179, 2013.

Tampereen teknillinen yliopisto
PL 527
33101 Tampere

Tampere University of Technology
P.O.B. 527
FI-33101 Tampere, Finland

ISBN 978-952-15-3836-0
ISSN 1459-2045