FACULTY
OF MATHEMATICS
AND PHYSICS
Charles University

**DOCTORAL THESIS**

Jan Bok

# Structure and complexity of homomorphisms

Computer Science Institute of Charles University

Supervisor of the doctoral thesis: prof. RNDr. Jaroslav Nešetřil, DrSc.
Study programme: Computer Science - Theory of Computing, Discrete Models and Optimization

Prague 2022

I declare that I carried out this doctoral thesis independently, and only with the cited sources, literature and other professional sources. It has not been used to obtain another or the same degree.

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

In Prague, ........................ ..................................
Author's signature

# Acknowledgments

First, I would like to thank Professor Jaroslav Nešetřil for being my PhD advisor and for all the verbal, financial, and other support.

The second thanks go to my co-authors, not only to those from the papers appearing in this thesis but to all of my co-authors. The list is quite long; see the list of publications at the end of this thesis.

My gratitude also goes to my friends, family, and fellow students and colleagues of the Department of Applied Mathematics and the Computer Science Institute.

A special thanks go to Professor Pavol Hell and to Rick Brewster. The months spent in Canada have been a highlight of my studies and have given me a lot. I am thankful that we are still working together, even with the nine-hour time difference.

Finally, I would like to thank Nikola Jedličková, my dearest co-author.

**Title:** Structure and complexity of homomorphisms

**Author:** Jan Bok

**Institute:** Computer Science Institute of Charles University

**Supervisor:** prof. RNDr. Jaroslav Nešetřil, DrSc.

**Supervisor's Department:** Computer Science Institute of Charles University

**Keywords:** graphs, graph homomorphism, complexity, signed graphs, covering projections, colouring, acyclic colouring

**Abstract:**

This thesis is concerned with computational complexity aspects of graph homomorphisms and related concepts. We are mainly interested in various polynomial time versus NP-complete dichotomies. These results are especially popular thanks to the seminal result of Hell and Nešetřil providing the complexity dichotomy for graph homomorphism problems and the recent breakthrough result proving the complexity dichotomy for constraint satisfaction problems.

The thesis is divided into three parts, all unified by the common goal to provide complexity classifications of various graph homomorphism problems. The first part is about list homomorphism problems for signed graphs. We study the complexity of such problems and obtain a structural description and dichotomy first for the case of targets being signed trees and then for the so-called separable graphs.

The second part focuses on graph covering projections, also known as locally bijective homomorphisms. To the best of our knowledge, we are the first to initiate cataloguing the complexity of the corresponding problems for (mutli)graphs with semi-edges. We have three larger goals here. (1) Providing the complete dichotomy for one- and two-vertex target graphs. (2) Discuss and propose the right definition of graph cover in the case of disconnected targets. (3) Explore what happens when we introduce lists into the problem.

The final part is dedicated to acyclic colourings, which can be viewed as special constrained colourings and hence homomorphisms to complete graphs. We study the effect of restricting the class of input graphs to those with a forbidden induced subgraph by providing a partial complexity dichotomy in the case where the number of colours is a part of the input and the full dichotomy when the number of colours is fixed.

# Contents

# Chapter 1

# Introduction

The theme of *graph homomorphisms* is a very active and fruitful area of mathematics and, more concretely, combinatorics and theoretical computer science. This is certainly not an exaggeration, considering that the concept generalises well-known and classical combinatorics problems, with maybe the most notable example being *graph colouring*. Furthermore, graph homomorphisms are a special case of the *constrained satisfaction problems* (CSPs). While the complexity classification for graph homomorphism problems is now a standard result [102], the classification for CSPs took a few decades. The study of CSP is undoubtedly an important and very live topic spanning general algebra, theoretical computer science, combinatorics, logic, and many other areas, and no doubt the effort behind proving the CSP dichotomy conjecture (now a theorem [43, 173]) was a driving force behind its popularity and appeal.

This introduction certainly cannot aim to provide a comprehensive introduction to graph homomorphisms. The reason is quite simple: there is already a very nice, coherent, complex, and relatively recent one written by Pavol Hell and Jaroslav Nešetřil [103] (with the second edition in preparation). Let us at least provide a brief minimum to make this thesis self-contained.

For the sake of generality and simplicity, we shall define graph homomorphisms for *digraphs*. *Digraphs* are a pair consisting of a set $V$, called vertices, and a binary relation $E$ on $V$, called edges. A mapping $f$ between the vertices of two digraphs $G$ and $H$ is then called a *digraph homomorphism* if $f(u)f(v) \in E(H)$ whenever $uv \in E(G)$. In accordance with [102], *graphs* are those digraphs with edge relation being irreflexive and symmetric (those are also frequently denoted as *simple graphs* or *graphs without loops and multiedges*).

However, for this thesis, the preceding definition of digraph homomorphism is still not general enough (with the sole exception of Chapter 9). The reason is rather simple. Both Part I and Part II of this thesis deal with special kinds of graphs; in Part I, they are *signed graphs*, and in Part II, they are *graphs with semi-edges*. These types of graphs are certainly not new and unknown, but on the other hand, they are also not standard ones, especially when it comes to the study of graph homomorphisms. Therefore, the definition of a graph is always strictly tied to the corresponding part of the thesis. Programmers would say that the scopes of the definitions are not *global* but *local* to the given part.

Finally, this is the reason why all the exhaustive literature review, motivations, and connections are described in the corresponding chapters (Chapters 2, 5, and 9) rather than in this brief introduction.

While each part pursues slightly different goals, there is one common motivation behind our efforts, providing a unifying umbrella to all of these results. *We want to explore boundaries between efficient and non-efficient, in other words, polynomial versus NP-complete dichotomies, for various generalisations and variations of the classical graph homomorphism problem.* I sincerely believe that the reader will find our endeavour described in this thesis interesting.

**Note on P versus NP and NP-hardness.** In this thesis and its results and dichotomies, we assume that P is not equal to NP. We also often omit the obvious argumentation that a given problem belongs to NP, and thus in proving NP-completeness results, we focus only on NP-hardness.

## Papers used in this thesis

The contents of this thesis is based on the following papers:

1. [23] Jan Bok, Richard C. Brewster, Tomás Feder, Nikola Jedličková, and Pavol Hell: *List Homomorphism Problems for Signed Graphs.* In 45th International Symposium on Mathematical Foundations of Computer Science, MFCS 2020, volume 170 of Leibniz International Proceedings in Informatics (LIPIcs), pages 170:20:1–20:14, 2020.

2. [24] Jan Bok, Richard C. Brewster, Tomás Feder, Pavol Hell, and Nikola Jedličková: *List Homomorphism Problems for Signed Graphs.* Submitted, 2021. `https://arxiv.org/abs/2005.05547`

3. [22] Jan Bok, Richard C. Brewster, Tomás Feder, Pavol Hell, and Nikola Jedličková: *List homomorphisms to separable signed graphs.* In Algorithms and Discrete Applied Mathematics - 8th International Conference, CALDAM 2022, volume 13179 of Lecture Notes in Computer Science, pages 22–35, 2022. `https://doi.org/10.1007/978-3-030-95018-7_3`

4. [27] Jan Bok, Jiří Fiala, Petr Hliněný, Nikola Jedličková, and Jan Kratochvíl: *Computational Complexity of Covering Multigraphs with Semi-Edges: Small Cases.* In 46th International Symposium on Mathematical Foundations of Computer Science, MFCS 2021, volume 202 of Leibniz International Proceedings in Informatics (LIPIcs), pages 21:1–21:15, 2021.

5. [29] Jan Bok, Jiří Fiala, Nikola Jedličková, Jan Kratochvíl, and Michaela Seifrtová: *Computational Complexity of Covering Disconnected Multigraphs.* In Fundamentals of Computation Theory, FCT 2021, volume 12867 of Lecture Notes in Computer Science, pages 85–89, 2021.

6. [28] Jan Bok, Jiří Fiala, Nikola Jedličková, Jan Kratochvíl, and Paweł Rzążewski: *List covering of regular multigraphs.* In Combinatorial Algorithms - 33rd International Workshop, IWOCA 2022, volume 13270 of Lecture Notes in Computer Science, pages 228–242, 2022. `https://doi.org/10.1007/978-3-031-06678-8_17`
7. [32] Jan Bok, Nikola Jedličková, Barnaby Martin, Daniël Paulusma, and Siani Smith: *Acyclic, Star and Injective Colouring: A Complexity Picture for $H$-Free Graphs.* In 28th Annual European Symposium on Algorithms, ESA 2020, volume 173 of Leibniz International Proceedings in Informatics (LIPIcs), pages 173:22:1–22:22, 2020.
8. [31] Jan Bok, Nikola Jedličková, Barnaby Martin, Pascal Ochem, Daniël Paulusma, and Siani Smith: *Acyclic, Star and Injective Colouring: A Complexity Picture for $H$-Free Graphs.* Submitted, 2021. `https://arxiv.org/abs/2008.09415`

The papers are often substantially extended, unified better together, and based on the respective journal versions either submitted or in preparation. At the beginning of each chapter, we provide a list of publications we base on.

All of the results contained in this thesis have been included here with the approval of every respective co-author. The contribution of the author of this thesis to each paper is proportional to the number of co-authors.

# Structure of the thesis

We shall now shortly describe the contents of this thesis in a rather informal manner. We provide more detailed and formal introductions at the beginning of each part.

It is worth noting that each part can be read without reading any other part. Also, regarding Part I and Part II, the corresponding introductory chapters (Chapter 2 and 5) contain all necessary preliminaries to understand any subsequent chapter in the given part.

## Part I

The first part of the thesis studies the complexity of list homomorphism problems for signed graphs.

The complexity of homomorphism (and list homomorphism) problems is a popular topic [103, 104]. For undirected graphs, it was shown in [102] that the problem of deciding the existence of homomorphisms from an input graph to a fixed graph $H$ is polynomial if $H$ is bipartite or has a loop and is NP-complete otherwise. For general structures $H$, the corresponding problem leads to the so-called CSP dichotomy conjecture [69, 112], which was only recently established [44, 173]. In the list homomorphism problem for $H$, the input contains with each input graph also lists of allowed images for each vertex. The list homomorphism problems generally have a nicer behaviour than the homomorphism problems because the lists facilitate recursion to subproblems.

Signed graphs are related to graphs with two symmetric binary relations; in addition, they are equipped with an operation of *switching*. The possibility of switching poses challenges when classifying the complexity of homomorphisms, as the problem no longer appears to be a homomorphism problem for relational structures. Nevertheless, it can be shown that it is equivalent to such a problem and hence the results from [44, 173] imply that there these problems also enjoy a dichotomy of polynomial versus NP-complete. For homomorphisms of signed graphs (without lists), a concrete dichotomy classification was conjectured in [39], and proved in [41]. Interestingly, for signed graphs, the list version no longer seems easier to classify.

In Chapter 2 we properly introduce the area of homomorphisms of signed graphs and speak in detail about the above-mentioned efforts. We also introduce crucial tools for proving NP-completeness.

The next chapter, Chapter 3, provides a complete graph-theoretical classification of complexity for the case of targets being signed trees. We first provide a classification for irreflexive (no loops) and reflexive (a loop on each vertex) signed trees. Then we generalise further to the mixed (some vertices have loops, some do not) signed trees by providing a suitable majority polymorphism and expanding our list of forbidden induced subgraphs.

Finally, in Chapter 4 we focus on one particular class of target graphs — bipartite irreflexive signed graphs in which the unicoloured edges form simple structures, such as paths and cycles — and provide a full classification of the complexity of the corresponding list homomorphism problem. In particular, our results confirm a recent conjecture of Kim and Siggers [117] for this class of signed graphs.

## Part II

The second part of the thesis is about graph coverings. A graph covering projection, also known as a locally bijective homomorphism, is a mapping between vertices and edges of two graphs which preserves incidences and is a local bijection. This notion stems from topological graph theory but has also found applications in general combinatorics and theoretical computer science.

The main goal of this part's four chapters is to start considering graph coverings from the computational point of view for multigraphs with loops and possible *semi-edges* (informally edges with only one endpoint, see Figure 1.1 for an initial example). These graphs naturally appear in algebraic graph theory but so far never received treatment in algorithmic graph theory.

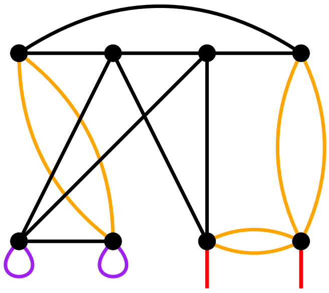


Figure 1.1: An example of graphs we consider in Part II of the thesis. Multiedges are in orange, loops in purple, and semi-edges in red colour.

Chapter 5 first surveys the motivations and literature on graph coverings and then focuses on carefully introducing all the needed definitions. Chapter 6 aims to provide the full complexity dichotomy for one-vertex and two-vertex target graphs. Moreover, we aim (and succeed) at providing a strong dichotomy in the sense that NP-hardness results have their inputs restricted to simple graphs (or even simple bipartite graphs), while polynomial results have as their inputs the most general graphs (in this context they are multigraphs with loops and semi-edges).

Chapter 7 focuses on a long-neglected question of what should be the right definition of graph cover in the case of disconnected targets. While this is easy to dismiss for the classical graph homomorphism problem, it is not the case here. We introduce three different ways and show that one particular way is, in our opinion, the best one. Our complexity results support this.

Chapter 8 concludes this part and stems from a similar motivation as the one for Part I. What happens when we add lists to the problem? Our main result is that List-$H$-Cover problem is NP-complete for every $k$-regular multigraph $H$, if $k > 2$ and if it contains at least one vertex incident with no loops, with no multiple edges and with at most one semi-edge. We apply the result for NP-co/polytime dichotomy of the computational complexity of List-$H$-Cover of cubic multigraphs.

## Part III

The third part of the thesis, consisting of Chapter 9, is about the complexity of *acyclic colourings*. It is a standard knowledge that graph colourings (assigning natural numbers to vertices so that adjacent vertices differ in assigned numbers) are, in fact, a special case of graph homomorphism. Namely, they correspond to the homomorphisms into complete graphs. Acyclic colourings are those in which the subgraph induced by any two colour classes does not contain a bicoloured cycle (hence the term acyclic). See Figure 1.2 for an introductory example.

As the introductory section of Chapter 9 suggests, the notion was intensively

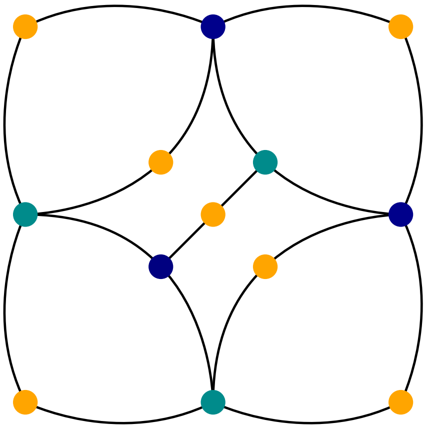


Figure 1.2: An example of an acyclically coloured graph on 13 vertices. In classical colouring, two colours would be enough, but now three colours are needed.

studied, even from the computational point of view. However, the complexity for $H$-free graphs (graphs with forbidden induced subgraph $H$) was not studied, contrasting with the classical colouring [120]. We fill this gap by providing a complete dichotomy for the case when the number of colours is fixed, and we provide an almost complete dichotomy for the case when the number of colours is given on input. We also provide a short overview of related results on special types of acyclic colourings: *star colouring* and *injective colouring*.

# Part I

# List homomorphisms of signed graphs

# Chapter 2

# Introduction to list homomorphism problems for signed graphs

This chapter is based on:

- [23] Jan Bok, Richard C. Brewster, Tomás Feder, Nikola Jedličková, and Pavol Hell: *List Homomorphism Problems for Signed Graphs.* In 45th International Symposium on Mathematical Foundations of Computer Science, MFCS 2020, volume 170 of Leibniz International Proceedings in Informatics (LIPIcs), pages 170:20:1–20:14, 2020.
- [24] Jan Bok, Richard C. Brewster, Tomás Feder, Pavol Hell, and Nikola Jedličková: *List Homomorphism Problems for Signed Graphs.* Submitted, 2021. `https://arxiv.org/abs/2005.05547`
- [22] Jan Bok, Richard C. Brewster, Tomás Feder, Pavol Hell, and Nikola Jedličková: *List homomorphisms to separable signed graphs.* In Algorithms and Discrete Applied Mathematics - 8th International Conference, CALDAM 2022, volume 13179 of Lecture Notes in Computer Science, pages 22–35, 2022. `https://doi.org/10.1007/978-3-030-95018-7_3`

## 2.1 Motivation

We investigate a problem at the confluence of two popular topics: graph homomorphisms and signed graphs. Their interplay was first considered in an unpublished manuscript of Guenin [97], and has since become an established field of study [148].

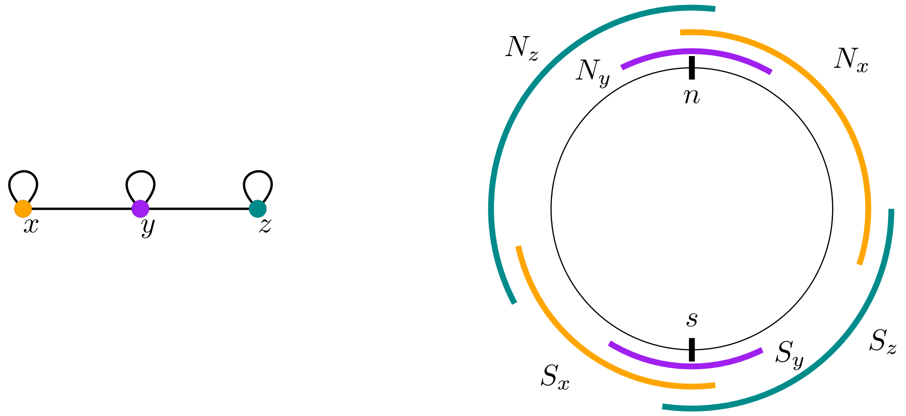


Figure 2.1: A reflexive path on three vertices and its bi-arc model. The colours of the vertices of the graph correspond to the appropriate arcs in the model.

We now introduce the two topics separately. In the study of computational aspects of graph homomorphisms, the central problem is one of existence – does an input graph $G$ admit a homomorphism to a fixed target graph $H$? (The graphs considered here are undirected graphs with possible loops but no parallel edges.) This is known as the *graph homomorphism problem*. It was shown in [102] that this problem is polynomial-time solvable when $H$ has a loop or is bipartite, and is NP-complete otherwise. This is known as the *dichotomy* of graph homomorphisms (see [103]). The *core* of a graph $H$ is a subgraph of $H$ with the smallest number of vertices to which $H$ admits a homomorphism; note that such a subgraph is unique up to isomorphism. A graph with a loop has a vertex with a loop as its core, and a (non-empty) bipartite graph has an edge as its core. Thus an equivalent way of stating the graph dichotomy result is that the problem is polynomial-time solvable when the core of $H$ has at most one edge, and is NP-complete otherwise.

Before we proceed and add lists to the problem, we first need to introduce bi-arc graphs.

**Definition 2.1.** *Let $C$ be a fixed circle with two specified points $n$ and $s$. A* bi-arc graph *is a graph $H$ such that each vertex $v \in V(H)$ can be associated with a pair of intervals $N_v, S_v$ where $N_v$ contains $n$ but not $s$ and $S_v$ contains $s$ but not $n$ satisfying the following conditions:*

1. *$N_v$ intersects $S_w$ if and only if $S_v$ intersects $N_w$, and*
2. *$N_v$ intersects $S_w$ if and only if $vw$ is not an edge of $H$.*

For an illustration, see Figure 2.1. Bi-arc graphs have turned out to be an interesting class of graphs; for instance, this class of graphs includes all interval graphs: a reflexive graph (each vertex has a loop) is a bi-arc graph if and only if it is an interval graph. Moreover, an irreflexive graph is a bi-arc graph if and only if it is bipartite and its complement is a circular arc graph [64].

Now suppose the input graph $G$ is equipped with lists, $L(v) \subseteq V(H), v \in V(G)$, and we ask if there is a homomorphism $f$ of $G$ to $H$ such that each $f(v) \in L(v)$. This is known as the *graph list homomorphism problem.* This problem also has a dichotomy of possible complexities [65] — it is polynomial-time solvable when $H$ is a bi-arc graph and it is NP-complete otherwise.

These kinds of complexity questions found their most general formulation in the context of constraint satisfaction problems. The Feder-Vardi dichotomy conjecture [69] claimed that every constraint satisfaction problem with a fixed template $H$ is polynomial-time solvable or NP-complete. After a quarter century of concerted effort by researchers in theoretical computer science, universal algebra, logic, and graph theory, the conjecture was proved in 2017, independently by Bulatov [44] and Zhuk [173]. This exciting development focused research attention on additional homomorphism type dichotomies, including ones for signed graphs [39, 41, 85].

The study of signed graphs goes back to [100, 101], and has been most notably investigated in the papers of Zaslavsky [165, 166, 167, 168, 169], from the point of view of colourings, matroids, or embeddings. Notably, they are of particular interest in nowhere-zero flows for graphs embedded in non-orientable surfaces [114]. Following Guenin, homomorphisms of signed graphs have been pioneered in [40] and [147]. The computational aspects of existence of homomorphisms in signed graphs — given a fixed signed graph $(H, \pi)$, does an input signed graph $(G, \sigma)$ admit a homomorphism to $(H, \pi)$ — were studied in [39, 85], and eventually a complete dichotomy classification was obtained in [41]. It is surprisingly similar to the second way we stated the graph dichotomy result above, see Theorem 2.6, and the discussion following it.

Although typically homomorphism problems tend to be easier to classify with lists than without lists (lists allow for recursion to subgraphs), the complexity of the list homomorphism problem for signed graphs appears difficult to classify. If the analogy to (unsigned) graphs holds again, then the tractable cases of the problem should identify an interesting class of signed graphs, generalizing bi-arc graphs. In the following chapter, we begin the exploration of this concept, first focusing on the case of signed trees. We find that there is interesting structure to the tractable cases.

## 2.2 Terminology and notation

A *signed graph* is a graph $G$, with possible loops and multiple edges (at most two loops per vertex and at most two edges between a pair of vertices), together with a mapping $\sigma\colon E(G) \to \{+, -\}$, assigning a sign ($+$ or $-$) to each edge and each loop of $G$, so that different loops at a vertex have different signs, and similarly for different edges between the same two vertices. For convenience, we shall usually consider an edge to mean an edge or a loop, and to emphasize otherwise we

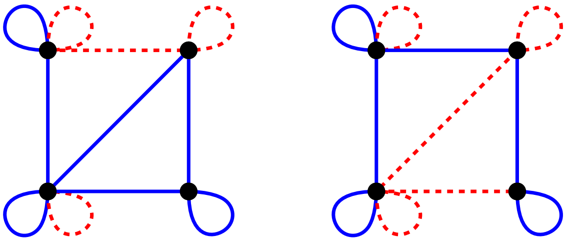


Figure 2.2: Two switching equivalent signed graphs. To obtain from the graph on the left the graph on the right, it suffices to switch at both left vertices of the graph (in any order).

shall call it a *non-loop edge*. Thus we can say, for example, that each edge of a signed graph has a sign, meaning both loops and non-loop edges. We denote a signed graph by $(G, \sigma)$, and call $G$ its *underlying graph* and $\sigma$ its *signature*. When the signature name is not needed, we denote the signed graph $(G, \sigma)$ by $\widehat{G}$ to emphasize that it has a signature even though we do not give it a name.

We also provide an alternative equivalent definition. A *signed graph* $\widehat{G}$ consists of a set $V(G)$ and two symmetric binary relations $+, -$. We also view $\widehat{G}$ as a graph $G$ with the vertex set $V(G)$, the edge set $+ \cup -$ (the *underlying graph of* $\widehat{G}$), and a mapping $\sigma\colon E(G) \to \{+, -\}$, assigning a sign ($+$ or $-$) to each edge of $G$. (A loop is considered to be an edge.)

We now define the *switching* operation. This operation can be applied to any vertex of a signed graph and it negates the signs of all its incident non-loop edges. The signs of loops are unchanged by switching since the loop is incident twice to a given vertex and thus its sign is negated twice as well. We say that two signatures $\sigma_1, \sigma_2$ of a graph $G$ are *switching equivalent* if we can obtain $(G, \sigma_2)$ from $(G, \sigma_1)$ by a sequence of switchings. In that case we also say that the two signed graphs $(G, \sigma_1)$ and $(G, \sigma_2)$ are switching equivalent. See an example in Figure 2.2. We note a sequence of switchings may also be realized by negating all the edges of a single edge cut. In a very formal way, a signed graph is an equivalence class under the switching equivalence, and we sometimes use the notation $\widehat{G}$ to mean the entire class.

We will usually view signs of edges as colours, and call positive edges *blue*, and negative edges *red*. It will be convenient to call a red-blue pair of edges with the same endpoint(s) a *bicoloured edge* (this includes loops as well as non-loop edges); however, formally they are two distinct edges. By contrast, we call edges that are not part of such a pair *unicoloured*; moreover, when we refer to an edge as blue or red we shall always mean the edge is unicoloured blue or red. We also call an edge *at least blue* if it is either blue or bicoloured, and similarly for *at least red* edges. The terms *at least positive* and *at least negative* are used in the same sense.

Treating a pair of red-blue edges as one bicoloured edge is advantageous in many

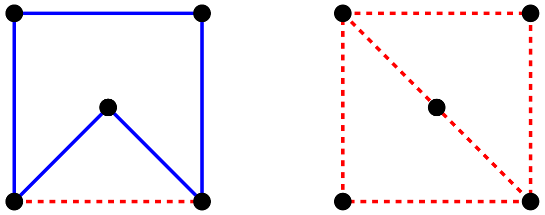


Figure 2.3: The signed graph on the left is neither balanced nor anti-balanced. The signed graph on the right is anti-balanced and also balanced, despite having all edges negative.

descriptions, but introduces an ambiguity when discussing walks, since a walk in a signed graph could be seen as a sequence of incident vertices and edges, and so selecting just one edge from a red-blue pair, or it could be interpreted as a sequence of consecutively adjacent vertices, and hence contain some bicoloured edges. This creates particular problem for cycles, since in the former view, a bicoloured edge would be seen as a cycle of length two, with one red edge and one blue edge. In the literature, the former approach is more common, but here we take the latter approach. Of course, the two views coincide if only walks of unicoloured edges are considered. The sign of a walk consisting of unicoloured edges $\widehat{G}$ is the product of the signs of its edges. Thus a walk of unicoloured edges is *negative* if it has an odd number of negative (red) edges, and *positive* if it has an even number of negative (red) edges. In the case of unicoloured cycles, we also call a negative cycle *unbalanced* and a positive cycle *balanced*. Note that a vertex with a red loop is a cycle with one negative edge, and hence is unbalanced.

A signed graph is *balanced* if all its cycles (if any) have an even number of red edges and it is *anti-balanced* if each cycle has an even number of blue edges. Since bicoloured edge can be viewed as a cycle of length two and thus is neither balanced, nor anti-balanced, it may be useful to have a weaker notion. A *weakly balanced signed graph* is a signed graph in which all unicoloured cycles (if any) have an even number of red edges and a *weakly anti-balanced signed graph* is a signed graph in which each unicoloured cycle has an even number of blue edges.[1] For an illustration, see Figure 2.3 and Figure 2.4.

It was proved by Zaslavsky [166] that two signatures of $G$ are switching equivalent if and only if they define exactly the same set of negative (or positive) cycles. It is easy to conclude that a signed graph is balanced if it is switching equivalent to one without red edges (and bicoloured edges), and is anti-balanced if it is switching equivalent to one without blue edges (and bicoloured edges); here we view a bicoloured edge as both blue and red. Similarly, a weakly balanced signed

[1] We were only recently notified by Reza Naserasr that the term of weak balancedness actually collides with the term *weakly bipartite signed graph* used by e.g. Guenin [96], Schrijver [156], and others. (Where bipartite signed graph means actually balanced signed graph, not that the underlying graph is bipartite.) In the future, we might decide to change the naming.

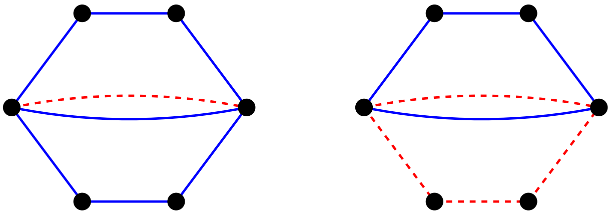


Figure 2.4: The signed graph on the left is both weakly balanced and weakly anti-balanced. The signed graph on the right is neither weakly balanced nor weakly anti-balanced.

graph is switching equivalent to a signed graph with all edges and loops at least blue, and a weakly anti-balanced signed graph is switching equivalent to a signed graph with all edges and loops at least red. Thus we have a symmetry to viewing the signs as colours, in particular $\widehat{G}$ is weakly balanced if and only if $\widehat{G}'$, obtained from $\widehat{G}$ by exchanging the colour of each edge, is weakly anti-balanced.

We now consider homomorphisms of signed graphs. Since signed graphs $\widehat{G}, \widehat{H}$ can be viewed as equivalence classes, a homomorphism of signed graphs $\widehat{G}$ to $\widehat{H}$ should be a homomorphism of one representative $(G, \sigma)$ of $\widehat{G}$ to one representative $(H, \pi)$ of $\widehat{H}$. It is easy to see that this definition can be simplified by prescribing any fixed representative $(H, \pi)$ of $\widehat{H}$. In other words, we now consider mapping all possible representatives $(G, \sigma')$ of $\widehat{G}$ to one fixed representative $(H, \pi)$ of $\widehat{H}$. At this point, a homomorphism $f$ of one concrete $(G, \sigma')$ to $(H, \pi)$ is just a homomorphism of the underlying graph $G$ to the underlying graph $H$ preserving the edge colours. Since there are multiple edges, we can either consider $f$ to be a mapping of vertices to vertices and edges to edges, preserving vertex-edge incidences and edge-colours, as in [148], or simply state that blue edges map to edges that are at least blue, red edges map to edges that are at least red, and bicoloured edges map to bicoloured edges. Formally, we state it as follows.

**Definition 2.2.** *We say that a mapping $f\colon V(G) \to V(H)$ is a* homomorphism *of the signed graph $(G, \sigma)$ to the signed graph $(H, \pi)$, written as $f\colon (G, \sigma) \to (H, \pi)$, if there exists a signed graph $(G, \sigma')$, switching equivalent to $(G, \sigma)$, such that whenever the edge $uv$ is at least positive in $(G, \sigma')$, then $f(u)f(v)$ is an edge that is at least positive in $(H, \pi)$, and whenever the edge $uv$ is at least negative in $(G, \sigma')$, then $f(u)f(v)$ is an edge that is at least negative in $(H, \pi)$.*

There is an equivalent alternative definition (see [148]). A homomorphism of the signed graph $(G, \sigma)$ to the signed graph $(H, \pi)$ is a homomorphism $f$ of the underlying graph $G$ to the underlying graph $H$, which maps bicoloured edges of $(G, \sigma)$ to bicoloured edges of $(H, \pi)$, and which for any closed walk $W$ in $(G, \sigma)$ with only unicoloured edges for which the image walk $f(W)$ has also only unicoloured edges, the sign of $f(W)$ in $(H, \pi)$ is the same as the sign of $W$ in $(G, \sigma)$. (In other words, negative closed walks map to negative closed walks and positive closed walks map to positive closed walks.) This definition does not require switching the input graph before mapping it. The equivalence of the two definitions follows from the theorem of Zaslavsky [166] cited above. That result is constructive, and the actual switching required to produce the switching equivalent signed graph $(G, \sigma')$ can be found in polynomial time [148].

We deduce the following fact.

**Lemma 2.3.** *Suppose $(G, \sigma)$ and $(H, \pi)$ are signed graphs, and $f$ is a mapping of the vertices of $G$ to the vertices of $H$. Then $f$ is a homomorphism of the signed graph $(G, \sigma)$ to the signed graph $(H, \pi)$ if and only if $f$ is a homomorphism of the underlying graph $G$ to the underlying graph $H$, which moreover maps bicoloured edges of $(G, \sigma)$ to bicoloured edges of $(H, \pi)$, and for any closed walk $W$ in $(G, \sigma)$ with only unicoloured edges for which the image walk $f(W)$ has also only unicoloured edges, the signs of $W$ and $f(W)$ are the same.*

Note that each negative closed walk contains a negative cycle, and in particular an irreflexive tree $(H, \pi)$ has no negative closed walks except for those using bicoloured edges. Thus if $(H, \pi)$ is an irreflexive tree, then the condition simplifies to having no negative cycle of $(G, \sigma)$ mapped to unicoloured edges in $(H, \pi)$ (because the image would be a positive closed walk). For reflexive trees, the condition requires that no negative cycle of $(G, \sigma)$ maps to a positive closed walk in $(H, \pi)$, and no positive cycle of $(G, \sigma)$ maps to a negative closed walk.

For our purposes, the simpler Definition 2.2 is sufficient. Note that whether an edge is unicoloured or bicoloured is independent of switching, and that a homomorphism can map a unicoloured edge or loop in $\widehat{G}$ to a bicoloured edge or loop in $\widehat{H}$ but not conversely.

Let $\widehat{H}$ be a fixed signed graph. The *homomorphism problem* S-Hom($\widehat{H}$) takes as input a signed graph $\widehat{G}$ and asks whether there exists a homomorphism of $\widehat{G}$ to $\widehat{H}$. The formal definition of the list homomorphism problems for signed graphs is very similar.

**Definition 2.4.** *Let $\widehat{H}$ be a fixed signed graph. The* list homomorphism problem *List-S-Hom($\widehat{H}$) takes as input a signed graph $\widehat{G}$ with lists $L(v) \subseteq V(H)$ for every $v \in V(G)$, and asks whether there exists a homomorphism $f$ of $\widehat{G}$ to $\widehat{H}$ such that $f(v) \in L(v)$ for every $v \in V(G)$.*

We note that when $\widehat{H}$ and $\widehat{H'}$ are switching equivalent signed graphs, then any homomorphism of an input signed graph $\widehat{G}$ to $\widehat{H}$ is also a homomorphism to $\widehat{H'}$, and therefore the problems S-Hom($\widehat{H}$) and S-Hom($\widehat{H'}$), as well as the problems List-S-Hom($\widehat{H}$) and List-S-Hom($\widehat{H'}$), are equivalent.

We call a signed graph $\widehat{H}$ *connected* if the underlying graph $H$ is connected. We call $\widehat{H}$ *reflexive* if each vertex of $H$ has a loop, and *irreflexive* if no vertex has a loop. We call $\widehat{H}$ a *signed tree* if $H$, with any existing loops removed, is a tree.

We may assume that the target signed graph $\widehat{H}$ is connected. This implies no loss of generality for list homomorphism problems, as each component of an input signed graph $\widehat{G}$ can only be mapped to one component of a target signed graph $\widehat{H}$.

## 2.3 More background and connections to CSP

We now briefly introduce the constraint satisfaction problems (CSP), in the format used in [69]. A *relational system* $G$ consists of a set $V(G)$ of vertices and a family of relations $R_1, R_2, \dots, R_k$ on $V(G)$. Assume $G$ is a relational system with relations $R_1, R_2, \dots, R_k$ and $H$ a relational system with relations $S_1, S_2, \dots, S_k$, where the arity of the corresponding relations $R_i$ and $S_i$ is the same for all $i = 1, 2, \dots, k$. A *homomorphism* of $G$ to $H$ is a mapping $f\colon V(G) \to V(H)$ that preserves all relations, i.e., satisfies $(v_1, v_2, \dots) \in R_i \implies (f(v_1), f(v_2), \dots) \in S_i$, for all $i = 1, 2, \dots, k$. The *constraint satisfaction problem* with fixed template $H$ asks whether or not an input relational system $G$, with the same arities of corresponding relations as $H$, admits a homomorphism to $H$.

Note that when $H$ has a single relation $S$, which is binary and symmetric, then we obtain the graph homomorphism problem referred to at the beginning of Section 2.1. When $H$ has a single relation $S$, which is an arbitrary binary relation, we obtain the *digraph homomorphism problem* [13] which is in a certain sense [69] as difficult to classify as the general constraint satisfaction problem. When $H$ has two relations $+, -$, then we obtain a problem that is superficially similar to the homomorphism problem for signed graphs, except that switching is not allowed. This problem is called the *edge-coloured graph homomorphism problem* [38], and it turns out to be similar to the digraph homomorphism problem in that it is difficult to classify [39]. On the other hand, the homomorphism problem for signed graphs [39, 41, 85], seems easier to classify, and exhibits a dichotomy similar to the graph dichotomy classification, see Theorem 2.6.

List homomorphism problems are also special cases of constraint satisfaction problems, as lists can be replaced by unary relations. Consider first the case of graphs. Suppose $H$ is a fixed graph, and form the relational system $H^{\#}$ with vertices $V(H)$ and the following relations: one binary relation $E(H)$ (this is a symmetric relation corresponding to the undirected edges of the graph $H$), and $2^{|V(H)|} - 1$ unary relations $R_X$ on $V(H)$, each consisting of a different non-empty subset $X$ of $V(H)$. The constraint satisfaction problem with template $H^{\#}$ has inputs $G$ with a symmetric binary relation $E(G)$ (a graph) and unary relations $S_X, X \subseteq V(H)$, and the question is whether or not a homomorphism exists. If a vertex $v \in V(G)$ is in the relation $S_X$ corresponding to $R_X$, then any mapping preserving the relations must map $v$ to a vertex in $X$; thus imposing the relation $S_X$ on $v \in V(G)$ amounts to setting $L(v) = X$. Therefore the list homomorphism problem for the graph $H$ is formulated as the constraint satisfaction problem for the template $H^{\#}$.

Such a translation is also possible for homomorphism of signed graphs. Brewster and Graves [40] introduced a useful construction. The *switching graph* $(H^+, \pi^+)$ has two vertices $v_1, v_2$ for each vertex $v$ of $(H, \pi)$, and each edge $vw$ of $(H, \pi)$ gives rise to edges $v_1w_1, v_2w_2$ of colour $\pi(vw)$ and edges $v_1w_2, v_2w_1$ of the opposite colour. (This definition applies also for loops, i.e., when $v = w$.) Then each homomorphism of the signed graph $(G, \sigma)$ to the signed graph $(H, \pi)$ corresponds to a homomorphism of the edge-coloured graph $(G, \sigma)$ to the edge-coloured graph $(H^+, \pi^+)$ and conversely. For list homomorphisms of signed graphs, we can use the same transformation, modifying the lists of the input signed graph. If $(G, \sigma)$ has lists $L(v), v \in V(G)$, then the new lists $L^+(v), v \in V(G)$, are defined as follows: for any $x \in L(v)$ with $v \in V(G)$, we place both $x_1$ and $x_2$ in $L^+(v)$. It is easy to see that the signed graph $(G, \sigma)$ has a list homomorphism to the signed graph $(H, \pi)$ with respect to the lists $L$ if and only if the edge-coloured graph $(G, \sigma)$ has a list homomorphism to the edge-coloured graph $(H^+, \pi^+)$ with respect to the lists $L^+$. The new lists $L^+$ are *symmetric sets in* $H^+$, meaning that for any $x \in V(H), v \in V(G)$, we have $x_1 \in L^+(v)$ if and only if we have $x_2 \in L^+(v)$. Thus we obtain the list homomorphism problem for the edge-coloured graph $(H^+, \pi^+)$, restricted to input instances $(G, \sigma)$ with lists $L$ that are symmetric in $H^+$. As above, we can transform this list homomorphism problem for the edge-coloured graph $(H^+, \pi^+)$, to a constraint satisfaction problem. The details are similar to the construction of $H^{\#}$, except this time the new template $(H^+, \pi^+)^*$ is obtained by adding unary relations $R_X = X$ only for sets $X \subseteq V(H^+)$ that are symmetric in $H^+$.

We conclude that our problems List-S-Hom($\widehat{H}$) fit into the general constraint satisfaction framework, and therefore it follows from [44, 173] that dichotomy holds for problems List-S-Hom($\widehat{H}$). We therefore ask which problems List-S-Hom($\widehat{H}$) are polynomial-time solvable and which are NP-complete.

The solution of the Feder-Vardi dichotomy conjecture involved an algebraic classification of the complexity pioneered by Jeavons [112]. A key role in this is played by the notion of a polymorphism of a relational structure $H$. If $H$ is a digraph, then a *polymorphism* of $H$ is a homomorphism $f$ of some power $H^t$ to $H$, i.e., a function $f$ that assigns to each ordered $t$-tuple $(v_1, v_2, \ldots, v_t)$ of vertices of $H$ a vertex $f(v_1, v_2, \ldots, v_t)$ such that two coordinate-wise adjacent tuples obtain adjacent images. For general templates, all relations must be similarly preserved. A polymorphism of order $t = 3$ is a *majority* if $f(v, v, w) = f(v, w, v) = f(w, v, v) = v$ for all $v, w$. A *Siggers polymorphism* is a polymorphism of order $t = 4$, if $f(a, r, e, a) = f(r, a, r, e)$ for all $a, r, e$. One formulation of the dichotomy theorem proved by Bulatov [44] and Zhuk [173] states that the constraint satisfaction problem for the template $H$ is polynomial-time solvable if $H$ admits a Siggers polymorphism, and is NP-complete otherwise. Majority polymorphisms are less powerful, but it is known [69] that if $H$ admits a majority then the constraint satisfaction problem for the template $H$ is polynomial-time solvable. Moreover, it was shown in [65] that a graph $H$ is a bi-arc graph if and only if the associated relational system $H^*$ admits a majority polymorphism. Thus the list homomorphism problem for a graph $H$ with possible loops is polynomial-time solvable if $H^*$ admits a majority polymorphism, and is NP-complete otherwise. It was observed in [117] that this is not true for signed graphs.

There is a convenient way to think of polymorphisms $f$ of the relational system $(H^+, \pi^+)^*$. A mapping $f$ is a polymorphism of $(H^+, \pi^+)^*$ if and only if it is a polymorphism of the edge-coloured graph $(H^+, \pi^+)$ and if, for any symmetric set $X \subseteq V(H^+)$, we have $x_1, x_2, \dots, x_t \in X$ then also $f(x_1, x_2, \dots, x_t) \in X$. We call such polymorphisms of $(H^+, \pi^+)$ *semi-conservative.*

We can apply the dichotomy result of [44, 173] to obtain an algebraic classification.

**Theorem 2.5.** *For any signed graph $(H, \pi)$, the problem* List-S-Hom$(H, \pi)$ *is polynomial-time solvable if $(H^+, \pi^+)$ admits a semi-conservative Siggers polymorphism, and is NP-complete otherwise.*

As mentioned above, one can not replace the semi-conservative Siggers polymorphism by a semi-conservative majority polymorphism [117]. We focus here on seeking a graph theoretic classification, at least for some classes of signed graphs.

## 2.4 Basic facts

We first mention the dichotomy classification of the problems S-Hom$(\widehat{H})$ by Brewster and Siggers [41], previously conjectured by Brewster, Foucaud, Hell, and Naserasr [39]. A subgraph $\widehat{G}$ of the signed graph $\widehat{H}$ is the *s-core* (or also *signed core*) of $\widehat{H}$ if there is homomorphism $f \colon \widehat{H} \to \widehat{G}$, and every homomorphism $\widehat{G} \to \widehat{G}$ is a bijection on $V(G)$. The letter $s$ stands for *signed.* It is again easy to see that the s-core is unique up to isomorphism and switching equivalence. In counting edges we count each unicoloured edge as one and each bicoloured edge as two.

**Theorem 2.6.** *[41] The problem S-Hom$(\widehat{H})$ is polynomial-time solvable if the s-core of $\widehat{H}$ has at most two edges, and is NP-complete otherwise.*

When the signature $\pi$ has all edges positive, the problem S-Hom$(H, \pi)$ is equivalent to the unsigned graph homomorphism problem, and the s-core of $(H, \pi)$ is just the core of $H$. To compare Theorem 2.6 with the graph dichotomy theorem of [102] as discussed at the beginning Section 2.1, we observe that the core of a graph cannot have exactly two edges, as a core must be either a single vertex (possibly with a loop), or a single edge, or a graph with at least three edges. Thus Theorem 2.6 is stronger than the graph dichotomy theorem from [102], which states that the graph homomorphism problem to $H$ is polynomial-time solvable if the core of $H$ has at most one edge and is NP-complete otherwise. (However, we note that the proof of Theorem 2.6 in [41] uses the graph dichotomy theorem [102].)

Observe that an instance of the problem S-Hom($\widehat{H}$) can be also viewed as an instance of List-S-Hom($\widehat{H}$) with all lists $L(v) = V(H)$, therefore if S-Hom($\widehat{H}$) is NP-complete, then so is List-S-Hom($\widehat{H}$). Moreover, if $\widehat{H}'$ is an induced subgraph of $\widehat{H}$, then any instance of List-S-Hom($\widehat{H}'$) can be viewed as an instance of List-S-Hom($\widehat{H}$) (with the same lists), therefore if the problem List-S-Hom($\widehat{H}'$) is NP-complete, then so is the problem List-S-Hom($\widehat{H}$). This yields the NP-completeness of List-S-Hom($\widehat{H}$) for all signed graphs ($\widehat{H}$) that contain an induced subgraph $\widehat{H}'$ whose s-core has more than two edges. Furthermore, when the signed graph $\widehat{H}$ is weakly balanced, then we may assume that all edges are at least blue, and the list homomorphism problem for $H$ can be reduced to List-S-Hom($\widehat{H}$). In particular, we emphasize that *List-S-Hom($\widehat{H}$) is NP-complete if $\widehat{H}$ is a weakly balanced signed graph (or, by a symmetric argument, a weakly anti-balanced signed graph), and the underlying graph $H$ is not a bi-arc graph* [65].

Next we focus on the class of signed graphs that have no bicoloured loops and no bicoloured edges. In this case, the following simple dichotomy — initially announced in [26] — describes the classification. It follows from our earlier remarks that these signed graphs are balanced if and only if they are weakly balanced, and similarly they are anti-balanced if and only if they are weakly anti-balanced.

**Theorem 2.7.** *Suppose $\widehat{H}$ is a connected signed graph without bicoloured loops and edges. If the underlying graph $H$ is a bi-arc graph, and $\widehat{H}$ is balanced or anti-balanced, then the problem List-S-Hom($\widehat{H}$) is polynomial-time solvable. Otherwise, the problem is NP-complete.*

*Proof.* The polynomial cases follow from Feder et al. [65], by the following argument. Suppose $\widehat{H}$ is balanced; we may assume all edges are blue. In [166], there is a polynomial-time algorithm to decide if the input signed graph $\widehat{G}$ is balanced. It it is not balanced, there is no homomorphism of $\widehat{G}$ to $\widehat{H}$. Otherwise, we may assume that $\widehat{G}$ has also all edges blue and hence there is a homomorphism of $\widehat{G}$ to $\widehat{H}$ if and only if there is a homomorphism of $G$ to $H$. Since $H$ is a bi-arc graph, this can be decided in polynomial time by the algorithm in [65]. The argument is similar if $\widehat{H}$ is anti-balanced. Otherwise, $\widehat{H}$ contains a cycle which cannot be switched to a blue cycle and a cycle which cannot be switched to a red cycle, in which case the s-core of $\widehat{H}$ contains at least three edges. (This is true even if the cycles are just loops.) □

We have observed that List-S-Hom($\widehat{H}$) is NP-complete if the s-core of $\widehat{H}$ has more than two edges. Thus we will focus on signed graphs $\widehat{H}$ whose s-cores have at most two edges. This is not as simple as it sounds, as there are many complex signed graphs with this property, including, for example, all irreflexive

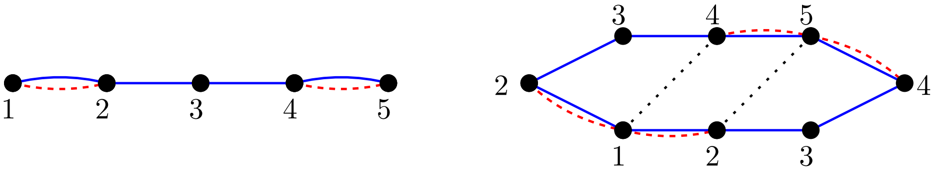


Figure 2.5: An example of a signed graph and of a chain in it (on the right).

bipartite signed graphs that contain a bicoloured edge, and all signed graphs that contain a bicoloured loop. That these cases are not easy underlines the fact that the assumptions in Theorem 2.7 cannot be weakened without significant new breakthroughs. Consider, for example, allowing bicoloured edges but not bicoloured loops. In this situation, we may focus on the case when there is a bicoloured edge (else Theorem 2.7 applies), and so if there is any loop at all, the s-core would have more than two edges. Thus we consider irreflexive signed graphs. The s-core is still too big if the underlying graph has an odd cycle. So in this case it remains to classify the irreflexive bipartite signed graphs that contain a bicoloured edge. Even this case is complex. We explore homomorphisms to irreflexive bipartite signed graphs in Chapter 4.

## 2.5 Tools

We now introduce basic tools for proving our NP-completeness results: *chains* and *invertible pairs*. While invertible pairs have been introduced earlier in the literature, chains are a completely new concept.

### 2.5.1 Chains

**Definition 2.8.** *Let $(U, D)$ be two walks in $\widehat{H}$ of equal length, say $U$, with vertices $u = u_0, u_1, \dots, u_k = v$ and $D$, with vertices $u = d_0, d_1, \dots, d_k = v$. We say that $(U, D)$ is a* chain, *provided $uu_1, d_{k-1}v$ are unicoloured edges and $ud_1, u_{k-1}v$ are bicoloured edges, and for each $i$, $1 \leq i \leq k-2$, we have*

1. *both $u_iu_{i+1}$ and $d_id_{i+1}$ are edges of $\widehat{H}$ while $d_iu_{i+1}$ is not an edge of $\widehat{H}$, or*
2. *both $u_iu_{i+1}$ and $d_id_{i+1}$ are bicoloured edges of $\widehat{H}$ while $d_iu_{i+1}$ is not a bicoloured edge of $\widehat{H}$.*

See an example of a chain in Figure 2.5.

Let us remind that Not-All-Equal 3-SAT is a problem in which a formula is given such that each clause has three unnegated variables, and we seek a truth assignment in which at least one variable is true and at least one is false, in each clause.

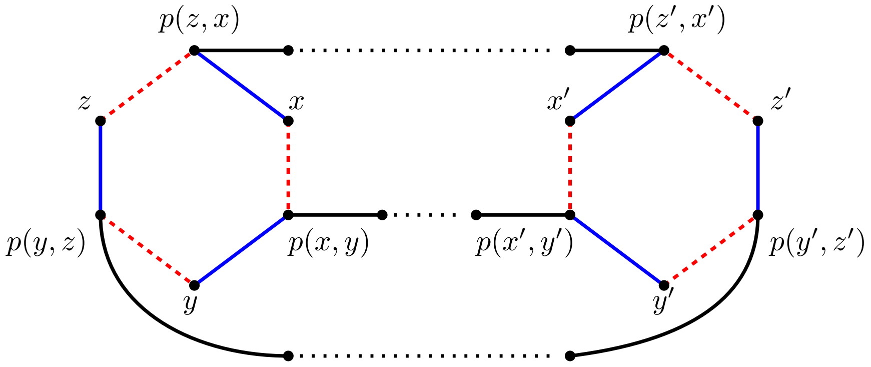


Figure 2.6: The clause gadget for clause $(x \vee y \vee z)$ in Theorem 2.9.

**Theorem 2.9.** *If a signed graph $\widehat{H}$ contains a chain, then* List-S-Hom($\widehat{H}$) *is NP-complete.*

*Proof.* Suppose that $\widehat{H}$ has a chain $(U, D)$ as specified above. We shall reduce from Not-All-Equal 3-SAT. For each clause $x \vee y \vee z$, we take three vertices $x, y, z$, each with list $\{u\}$, and three vertices $x', y', z'$, each with list $\{v\}$. For the triple $x, y, z$, we add three new vertices $p(x, y)$, $p(y, z)$, and $p(z, x)$, each with list $\{u_1, d_1\}$, and for the triple $x', y', z'$, we add three new vertices $p(x', y'), p(y', z'), p(z', x')$, each with list $\{u_{k-1}, d_{k-1}\}$. We connect these vertices as follows:

- $p(x, y)$ adjacent to $x$ by a red edge and to $y$ by a blue edge,
- $p(y, z)$ adjacent to $y$ by a red edge and to $z$ by a blue edge,
- $p(z, x)$ adjacent to $z$ by a red edge and to $x$ by a blue edge.

Analogously, the hexagon $x', p(x', y'), y', p(y', z'), z', p(z', x')$ will also be alternating in blue and red colours, with (say) $p(x', y')$ adjacent to $x'$ by a red edge.

Moreover, we join each pair of vertices $p(x, y)$ and $p(x', y')$ by a separate path $P(x, y)$ of length $k - 1$, say $p(x, y) = a_1, a_2, \ldots, a_{k-2}, a_{k-1} = p(x', y')$, where $a_i$ has list $\{u_i, d_i\}$ and the edge $a_i a_{i+1}$ is blue unless both $u_i u_{i+1}$ and $d_i d_{i+1}$ are bicoloured, in which case $a_i a_{i+1}$ is also bicoloured. Paths $P(z, x)$ and $P(y, z)$ are defined analogously. See Figure 2.6 for an illustration.

We observe for future reference that the path $x, p(x, y) = a_1, a_2, \ldots, a_{k-2}, a_{k-1} = p(x', y'), x'$, when considered by itself, admits a list homomorphism both to $U$ and to $D$. (To see this invoke Zaslavsky's theorem characterizing switching equivalent signatures, and use the fact that both $U$ and $D$ contain a bicoloured edge.) Further we note there is no list homomorphism to any other subgraph of $U \cup D$ where $p(x, y)$ maps to $d_1$ and $p(x', y')$ maps to $u_{k-1}$. (This follows from the conditions in the definition of chain.)

If $x$ occurs in several clauses, we link the occurrences by a new vertex $p(x)$ with the list $\{u_1\}$ and blue edges to all occurrences of $x$. Since the edge $uu_1$ is unicoloured, this will ensure that all occurrences of the vertex $x$ are switched or no occurrence of $x$ is switched. Hence, the variable $x$ will take on the same truth value in all clauses.

We denote the resulting graph $(G, \sigma)$. We now claim that this instance of Not-All-Equal 3-SAT is satisfiable if and only if $(G, \sigma)$ admits a list homomorphism to $(H, \pi)$.

Let $\widehat{G}(x, y, z)$ denote the subgraph of $(G, \sigma)$ induced by $P(x, y), P(z, x), P(y, z)$ and $x, y, z, x', y', z'$. We claim that

(i) *any list homomorphism of $\widehat{G}(x, y, z)$ to $U \cup D$ must switch at either one or two of the vertices $x, y, z$,* and that

(ii) *there are list homomorphisms of $\widehat{G}(x, y, z)$ to $U \cup D$ that switch at any one or any two of the vertices $x, y, z$.*

Once this claim is proved, we can associate with every truth assignment a list homomorphism of $\widehat{G}(x, y, z)$ to $U \cup D$ where a vertex corresponding to a variable is switched if and only if that variable is true, and conversely, setting a variable true if its corresponding vertex was switched in the list homomorphism. (Recall that we have ensured that all occurrences of a variable take on the same truth value.)

We now prove (i). Since the lists are so restrictive, any list homomorphism is fully described by what happens to the paths $P(x, y)$, $P(z, x)$, $P(y, z)$, and whether or not the vertices corresponding to $x, y, z$ (and $x', y', z'$) are switched. Note that $U$ begins with a unicoloured edge and $D$ ends with a unicoloured edge. If neither $x$ nor $y$ or both $x$ and $y$ are switched, the edges $xp(x, y)$ and $p(x, y)y$ are different colours, and in any list homomorphism of $\widehat{G}(x, y, z)$ to $U \cup D$ we must map $P(x, y)$ to $D$. In particular, $x'p(x', y')$ and $p(x', y')y'$ map to a unicoloured edge and must have the same colour. Thus, if none or all of the vertices $x, y, z$ were switched, then the hexagon $x', p(x', y'), y', p(y', z'), z', p(z', x')$ has an even number of red and even number of blue edges, which is impossible. (It started with an odd number of each.)

For (ii), it remains to show that one or two of the vertices $x, y, z$ can be switched under a list homomorphism of $\widehat{G}(x, y, z)$ to $U \cup D$. Suppose first that only one was switched; by symmetry assume it was $x$ (so $y$ and $z$ were not switched). Now edges $x, p(x, y)$ and $p(x, y), y$ have the same colour, and $z, p(z, x)$ and $p(z, x), x$ have the same colour. By the above observation, we can map $P(x, y)$ and $P(z, x)$ to $U$, and map $P(y, z)$ to $D$. Note that the switchings necessary for these list homomorphisms affect disjoint sets of vertices (the paths $P(x, y), P(y, z), P(z, x)$), so the observation applies. If two vertices, say $y$ and $z$ were switched, the argument is almost the same and we omit it. □

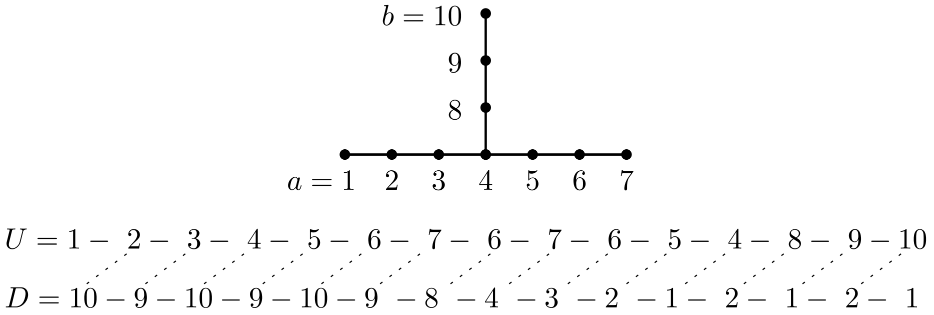

$b = 10$
9
8
$a = 1$
2
3
4
5
6
7
$U = 1 - 2 - 3 - 4 - 5 - 6 - 7 - 6 - 7 - 6 - 5 - 4 - 8 - 9 - 10$
$D = 10 - 9 - 10 - 9 - 10 - 9 - 8 - 4 - 3 - 2 - 1 - 2 - 1 - 2 - 1$

Figure 2.7: The graph $F_1$.

### 2.5.2 Invertible pairs

Invertible pairs played an important role in classifying complexity of list homomorphism problems for graphs. An *invertible pair* in an undirected graph $H$ is a pair of vertices $a, b$, with two walks $U, D$ of the same length, where $U$ has vertices $a = u_0, u_1, \ldots, u_k = b, u_{k+1}, \ldots, u_t = a$, and $D$ has vertices $b = d_0, d_1, \ldots, d_k = a, d_{k+1}, \ldots, d_t = b$, such that for each $i$, $1 \leq i \leq k-2$, both $u_i u_{i+1}$ and $d_i d_{i+1}$ are edges of $H$, while $d_i u_{i+1}$ is not an edge of $\widehat{H}$. For simplicity, we say that a signed graph has an invertible pair if its underlying graph has an invertible pair. It follows from [23, 63, 105, 67] that we have the following observation.

**Theorem 2.10.** *If $\widehat{H}$ has an invertible pair, then the list homomorphism problem for $\widehat{H}$ is NP-complete.*

Figure 2.7 shows the graph $F_1$, with an invertible pair $1, 10$. The walks $U, D$ begin as indicated, then continue from $7, 10$ to $7, 1$ in a similar manner, and then to $10, 1$, and similarly for the second half, from $10, 1$ to $1, 10$.

# Chapter 3

# List homomorphism problems for signed trees

This chapter is based on:

- [23] Jan Bok, Richard C. Brewster, Tomás Feder, Nikola Jedličková, and Pavol Hell: *List Homomorphism Problems for Signed Graphs.* In 45th International Symposium on Mathematical Foundations of Computer Science, MFCS 2020, volume 170 of Leibniz International Proceedings in Informatics (LIPIcs), pages 170:20:1–20:14, 2020.
- [24] Jan Bok, Richard C. Brewster, Tomás Feder, Pavol Hell, and Nikola Jedličková: *List Homomorphism Problems for Signed Graphs.* Submitted, 2021. `https://arxiv.org/abs/2005.05547`

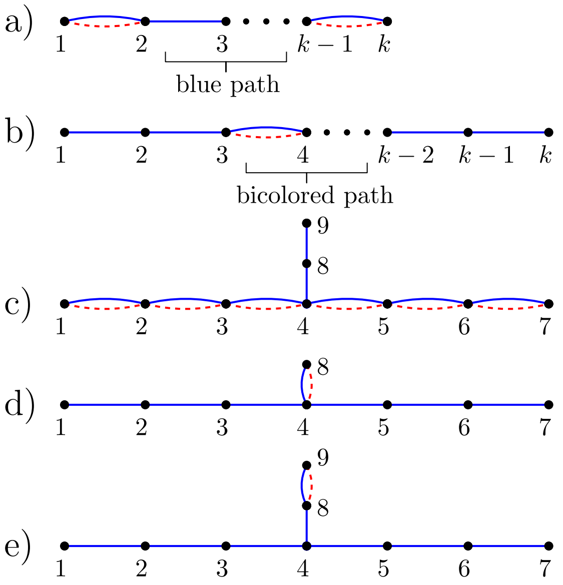
In this chapter, we study the list homomorphism problem to a fixed target signed tree. This will illustrate the difficulty of the general problem, since classifying the complexity of the problem when $H$ is a tree (with possible loops) is already rather complicated.

The structure of the signed trees in the polynomial cases is interesting, suggesting that the class of general signed graphs for which the problems are polynomial may have nice structure, analogous to bi-arc graphs (which characterized the polynomial cases of list homomorphisms to unsigned graphs).

We shall first treat the case of irreflexive trees, then the reflexive trees, and finally we will consider the general case.

## 3.1 Irreflexive trees

In this section, $\widehat{H}$ will always be an irreflexive tree. As trees do not have any cycles, $\widehat{H}$ is trivially weakly balanced, and hence we may assume that all edges



Special case of e) of family $\mathcal{J}$ in Figure 3.8.
$k \geq 4$

$U = 3 - 2 - 1 - 2 - \dots\dots\dots\dots\dots - (k-2)$
$D = 3 - \dots - (k-1) - k - (k-1) - (k-2)$
$k \geq 6$

$U = 4 - 8 - 9 - 8 - 4 - 5 - 6 - 7 - 6 - 5 - 4$
$D = 4 - 3 - 2 - 1 - 2 - 3 - 4 - 8 - 9 - 8 - 4$

$U = 4 - 3 - 2 - 1 - 2 - 3 - 4 - 8 - 4$
$D = 4 - 8 - 4 - 5 - 6 - 7 - 6 - 5 - 4$

$U = 8 - 4 - 5 - 6 - 7 - 6 - 5 - 4 - 8 - 9 - 8$
$D = 8 - 9 - 8 - 4 - 3 - 2 - 1 - 2 - 3 - 4 - 8$

Figure 3.1: The family $\mathcal{F}$ of signed irreflexive trees with NP-complete problems.

are at least blue.

**Lemma 3.1.** *If the underlying graph $H$ contains the graph $F_1$ in Figure 2.7, then* List-S-Hom$(\widehat{H})$ *is NP-complete.*

*Proof.* If the underlying graph $H$ contains the graph $F_1$ in Figure 2.7, then $H$ contains an invertible pair, whence List-S-Hom$(\widehat{H})$ is NP-complete by Theorem 2.10. □

**Lemma 3.2.** *If $\widehat{H}$ contains one of the signed graphs in family $\mathcal{F}$ from Figure 3.1 as an induced subgraph, then* List-S-Hom$(\widehat{H})$ *is NP-complete.*

*Proof.* For each signed tree in the family $\mathcal{F}$ we specify a chain either directly in Figure 3.1, or, in case a), indirectly by reference to a more general situation addressed in Figure 3.8. By Theorem 2.9, these signed trees yield NP-complete list homomorphism problems. Thus any signed graph $\widehat{H}$ that contains one of them as an induced subgraph has also the problem List-S-Hom$(\widehat{H})$ NP-complete. □

An irreflexive tree $H$ is a 2-*caterpillar* if it contains a path $P = v_1 v_2 \dots v_k$, such that each vertex of $H$ is either on $P$, or is a child of a vertex on $P$, or is a *grandchild* of a vertex on $P$, i.e., is adjacent to a child of a vertex on $P$. We also say that $H$ is a 2-caterpillar *with respect to the spine* $P$. (Note that the same tree $H$ can be a 2-caterpillar with respect to different spines $P$.) In such a situation, let $T_1, T_2, \dots, T_\ell$ be the connected components of $H \setminus P$. Each $T_i$ is a star adjacent to a unique vertex $v_j$ on $P$. The tree $T_i$ together with the edge joining it to $v_j$ is called a *rooted subtree* of $H$ (with respect to the spine $P$), and is considered to be rooted at $v_j$. Note that there can be several rooted subtrees with the same root vertex $v_j$ on the spine, but each rooted subtree at $v_j$ contains a unique child of $P$ (and possibly no grandchildren, or possibly several grandchildren).

We also use the term 2-*caterpillar* for any signed graph to mean that the underlying graph, with loops removed, is a 2-caterpillar.

If $H$ is a 2-caterpillar with respect to the spine $P$, and additionally the bicoloured edges of $\widehat{H}$ form a connected subgraph, and there exists an integer $d$, with $1 \leq d \leq k$, such that:

- all edges on the path $v_1 v_2 \ldots v_d$ are bicoloured, and all edges on the path $v_d v_{d+1} \ldots v_k$ are blue,
- the edges of all subtrees rooted at $v_1, v_2, \ldots, v_{d-1}$ are bicoloured, except possibly edges incident to leaves, and
- the edges of all subtrees rooted at $v_{d+1}, \ldots, v_k$ are all blue,

then we call $\widehat{H}$ a *good* 2-*caterpillar* with respect to $P = v_1 v_2 \ldots v_k$.

The vertex $v_d$ is called the *dividing vertex* of $\widehat{H}$. Note that the subtrees rooted at $v_d$ are not limited by any condition except the connectivity of the subgraph formed by the bicoloured edges. A typical example of a good 2-caterpillar is depicted in Figure 3.2.

**Lemma 3.3.** *Let $\widehat{H}$ be an irreflexive signed tree. Then $\widehat{H}$ is a good 2-caterpillar if and only if it does not contain any of the graphs from family $\mathcal{F}$ in Figure 3.1 as an induced subgraph, and the underlying graph $H$ does not contain the graph $F_1$ in Figure 2.7.*

*Proof.* It is easy to check that none of the depicted signed graphs admits a suitable spine, and hence they are not good 2-caterpillars. So assume $\widehat{H}$ does not contain any of the graphs in Figure 3.1 as an induced subgraph, and the underlying graph $H$ does not contain the graph $F_1$ in Figure 2.7 as a subgraph. If $H$ is not a 2-caterpillar with respect to any spine, then the underlying graph $H$ contains the tree in Figure 2.7. The bicoloured edges of $\widehat{H}$ induce a connected subgraph, since there is no subgraph of type a) from Figure 3.1. Similarly, the unicoloured edges between two non-leaf vertices of $\widehat{H}$ induce a connected subgraph, since there is no subgraph of type b) in Figure 3.1. It follows that there exists a path $P = v_1 v_2 \ldots v_k$ with a dividing vertex $v_d$ as specified. The absence of classes b)

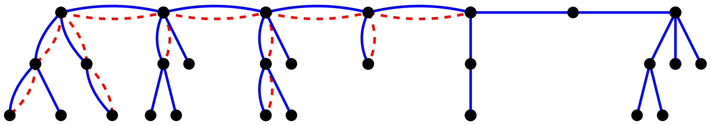


Figure 3.2: An example of a good 2-caterpillar.

and c) from Figure 3.1 also ensures that, there is a suitable spine $P = v_1v_2 \dots v_k$ with bicoloured edges on $v_1v_2 \dots v_d$ and blue edges on $v_dv_{d+1} \dots v_k$, and with all subtrees of height two rooted at $v_1, \dots, v_{d-1}$ attached to $P$ with a bicoloured edge. (For this, we note that in the case c), as long as the edges 34 and 45 are bicoloured, the subtree still yields an NP-complete problem even with any of the edges $12, 23, 56, 67$ unicoloured.) The absence of graphs d) and e) in Figure 3.1 ensures that all subtrees rooted at $v_{d+1}, \dots, v_k$ have all edges blue. □

**Theorem 3.4.** *Let $\widehat{H}$ be an irreflexive tree. If $\widehat{H}$ is a good 2-caterpillar, then* List-S-Hom($\widehat{H}$) *is polynomial-time solvable. Otherwise, $H$ contains a copy of $F_1$, or $\widehat{H}$ contains one of the signed graphs in family $\mathcal{F}$ as an induced subgraph, and the problem is NP-complete.*

The second claim follows from Lemmas 3.1, 3.2 and 3.3. We prove the first claim in a sequence of lemmas. Suppose that $\widehat{H}$ is a good 2-caterpillar with respect to the spine $P = v_1v_2 \dots v_k$. Since $H$ is bipartite, we may distinguish its vertices as *black* and *white*. We may assume that the input signed graph $\widehat{G}$ is connected and bipartite, and the lists of the black vertices of $G$ contain only black vertices of $H$, and similarly for white vertices. (Since $G$ is connected, there are only two possible assignments of black and white colours to its vertices, and we consider each separately.) For now, assume that $v_1$ is white.

We distinguish four types of rooted subtrees of $\widehat{H}$ with respect to the spine $P$.

- Type $T_1$: a bicoloured edge $v_ix$;
- Type $T_2$: a bicoloured edge $v_ix$, bicoloured edges $xz_j$ for a set of vertices $z_j$, and blue edges $xt_j$ for another set of vertices $t_j$;
- Type $T_3$: a blue edge $v_ix$ and blue edges $xt_j$ for a set of vertices $t_j$; and
- Type $T_4$: a blue edge $v_ix$.

In the types $T_2$ and $T_3$ we assume that they are not of type $T_1$ or $T_4$, i.e., that at least some $z_j$ or $t_j$ exist; but we allow in $T_2$ either the set of $z_j$ or the set of $t_j$ to be empty.

Recall that we assume that all edges of $\widehat{H}$ are at least blue.

A *bipartite min ordering* of the bipartite graph $H$ is a pair $<_b, <_w$, where $<_b$ is a linear ordering of the black vertices and $<_w$ is a linear ordering of the white vertices, such that for white vertices $x <_w x'$ and black vertices $y <_b y'$, if $xy', x'y$ are both edges in $H$, then $xy$ is also an edge in $H$. (This is also called *underbar property* and the reason for it is perhaps better understood by Figure 3.3.) It is known [69] that if a bipartite graph $H$ has a bipartite min ordering, then the list homomorphism problem for $H$ can be solved in polynomial time as follows. First apply the *arc consistency* test, which repeatedly visits edges $xy$ and removes from $L(x)$ any vertex of $H$ not adjacent to some vertex of $L(y)$, and similarly removes from $L(y)$ any vertex of $H$ not adjacent to some vertex of $L(x)$. After arc

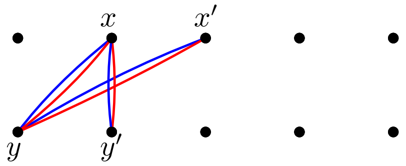


Figure 3.3: A visualisation of the underbar property.

consistency, if there is an empty list, no list homomorphism exists, and if all lists are non-empty, choosing the minimum element of each list, according to $<_b$ or $<_w$, defines a list homomorphism as required. We call a bipartite min ordering of the signed irreflexive tree $\widehat{H}$ *special* if for any black vertices $x, x'$ and white vertices $y, y'$, if $xy$ is bicoloured and $xy'$ is blue, then $y <_w y'$, and if $xy$ is bicoloured and $x'y$ is blue, then $x <_b x'$. In other words, the bicoloured neighbours of any vertex appear before its unicoloured neighbours, both in $<_b$ and in $<_w$.

**Lemma 3.5.** *Every good* 2*-caterpillar* $\widehat{H}$ *admits a special bipartite min ordering.*

*Proof.* Let us first observe that any 2-caterpillar admits a bipartite min ordering $<_b, <_w$ with $v_1 <_w v_3 <_w v_5, \ldots$ and $v_2 <_b v_4 <_b v_6, \ldots$, in which the vertices of each subtree rooted at a vertex $v_i$ are placed as follows: all non-leaf children of $v_i$, as well as all leaf children of $v_i$ adjacent to $v_i$ by bicoloured edges, are ordered between $v_{i-1}$ and $v_{i+1}$, all leaf children of $v_i$ adjacent to $v_i$ by unicoloured edges are ordered between $v_{i+1}$ and $v_{i+3}$, and all grandchildren of $v_i$ are ordered between $v_i$ and $v_{i+2}$. Moreover, we ensure that the order of the grandchildren conforms to the order of the children, i.e., if a child $a$ of $v_i$ is ordered before a child $b$ of $v_i$ then the children of $a$ are all ordered before the children of $b$. Finally, all children of $v_i$ are ordered after all the grandchildren of $v_{i-1}$. See Figure 3.4 for an illustration.

It remains to ensure that the bipartite min ordering we choose is in fact a special bipartite min ordering, i.e., that each vertex has its neighbours joined by bicoloured edges ordered before its neighbours joined by unicoloured edges. Therefore the subtrees rooted at each $v_i$ are handled as follows. We will order first the vertices of subtrees of type $T_1$, one at a time, then order the vertices of subtrees of type $T_2$, one at a time, then the vertices of subtrees of type $T_3$, one at a time, and finally the vertices of subtrees of type $T_4$, one at a time. Each subtree of type $T_1$ consists of only one bicoloured edge, and we order these consecutively

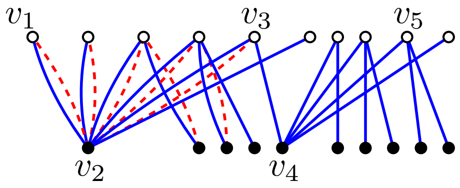


Figure 3.4: An example of special bipartite min ordering.

between $v_{i-1}$ and $v_{i+1}$. Next in order will come the children of $v_i$ in subtrees of type $T_2$, still before $v_{i+1}$, and in each of these subtrees we order first the grandchildren of $v_i$ incident to a bicoloured edge before those incident to a unicoloured edge. We order the subtrees of type $T_3$ similarly. Note that by the definition of a good 2-caterpillar, the subtrees of type $T_3$ can only be rooted at vertices $v_i$ with $d \leq i \leq k$. Thus, if we have a blue child of $v_i$ ordered before $v_{i+1}$, then $v_iv_{i+1}$ is unicoloured. Finally, for subtrees of type $T_4$, we order their vertices (each a child of $v_i$) right after $v_{i+1}$. □

**Lemma 3.6.** *If a signed irreflexive tree $\widehat{H}$ admits a special bipartite min ordering, then* List-S-Hom$(\widehat{H})$ *is polynomial-time solvable.*

*Proof.* We describe a polynomial-time algorithm. Suppose $\widehat{G}$ is the input signed graph; we may assume $\widehat{G}$ is connected, bipartite, and such that the black vertices have lists with only the black vertices of $\widehat{H}$, and similarly for the white vertices. The first step is to perform the arc consistency test for the existence of a homomorphism of the underlying graphs $G$ to $H$, using the special bipartite min ordering $<_b, <_w$. We also perform the *bicoloured arc consistency* test, which repeatedly visits bicoloured edges $xy$ of $G$ and removes from $L(x)$ any vertex of $H$ not adjacent to some vertex of $L(y)$ by a bicoloured edge, and similarly removes from $L(y)$ any vertex of $H$ not adjacent to some vertex of $L(x)$ by a bicoloured edge. If this yields an empty list, there is no list homomorphism of the underlying graphs, and hence no list homomorphism of signed graphs. Otherwise, the minima of all lists define a list homomorphism $f\colon G \to H$ of the underlying graphs, by [69]. By the bicoloured arc consistency test, the minimum choices imply that the image of a bicoloured edge under $f$ is also a bicoloured edge. According to Lemma 2.3 and the remark following it, $f$ is also a list homomorphism of signed graphs unless a negative cycle $C$ of unicoloured edges of $\widehat{G}$ maps to a closed walk $f(C)$ of blue edges in $\widehat{H}$. Now we make use of the properties of special bipartite min ordering to repair the situation, if possible. Note that the fact that we choose minimum possible values for $f$ means that we cannot map $C$ lower in the orders $<_b, <_w$. We consider three possible cases.

- *At least one of the edges of $f(C)$ is in a subtree $T$ of type $T_2$ rooted at some $v_i$, with $i \leq d$:*
  In this case, all edges of $f(C)$ must be in $T$, since the edge of $T$ incident to $v_i$ is bicoloured. Assume without loss of generality that $v_i$ is white, $x$ is the unique child of $v_i$ in $T$, and $xt_1, \ldots, xt_m$ are the blue edges of $T$, where $x$ is black and $t_1, \ldots, t_m$ are white. Since $f(C)$ is included in the edges $xt_1, \ldots, xt_m$ and $v_i$ precedes in $<_w$ all vertices $t_1, \ldots, t_m$, the final lists of the white vertices in $C$ do not include $v_i$ (since we assigned the minimum value in each list). Therefore under any homomorphism the image of the connected graph $C$ either is included in the set of edges $xt_1, \ldots, xt_m$, or is disjoint from this set of edges. Since we have already explored the first possibility, we can delete the vertices $t_1, \ldots, t_m$ from the lists of all white

vertices of $C$ and repeat the arc consistency test. This will check whether there is possibly another list homomorphism of graphs $G \to H$, which is also a homomorphism of signed graphs $\widehat{G} \to \widehat{H}$.

- *At least one of the edges of $f(C)$ is in a subtree of type $T_4$ rooted at some $v_i, i \leq d-1$:*
  In this case, all edges of $f(C)$ must be in subtrees of type $T_4$ rooted at the same $v_i$. Assume again, without loss of generality, that $v_i$ is white and the subtrees consist of the blue edges $v_ix_1, v_ix_2, \ldots, v_ix_m$, with each $x_j$ black. Since $<_b, <_w$ is a special bipartite min ordering, all vertices adjacent to $v_i$ by a bicoloured edge are smaller in $<_b$ than $x_1, \ldots, x_m$. Therefore no such vertex can be in a list of a black vertex in $C$. This again means that the image of $C$ is either included in the set of edges $v_ix_1, v_ix_2, \ldots, v_ix_m$, or is disjoint from this set of edges. We can delete all vertices $x_1, \ldots, x_m$ from the lists of all black vertices of $C$ and repeat as above.

- *The edges of $f(C)$ are included in the set of edges on the path $v_dv_{d+1}\ldots v_k$ and in the subtrees of types $T_3$ or $T_4$ rooted at $v_d, \ldots, v_k$:*
  In this case, the vertices in the lists of the cycle $C$ are joined only by blue edges, and there is no homomorphism of signed graphs $\widehat{G} \to \widehat{H}$.

After we modified the image of one negative cycle $C$ of $\widehat{H}$, we proceed to modify another, until we either obtain a homomorphism of signed graph, or find that no such homomorphism exists. The algorithm is polynomial, because arc consistency can be performed in linear time [69], and each modification removes at least one vertex of $H$ from the list of at least one vertex of $G$. Recall that the graph $H$ is fixed, and hence its number of vertices is a constant $k$. If $G$ has $n$ vertices, then this step will be performed at most $kn$ times. □

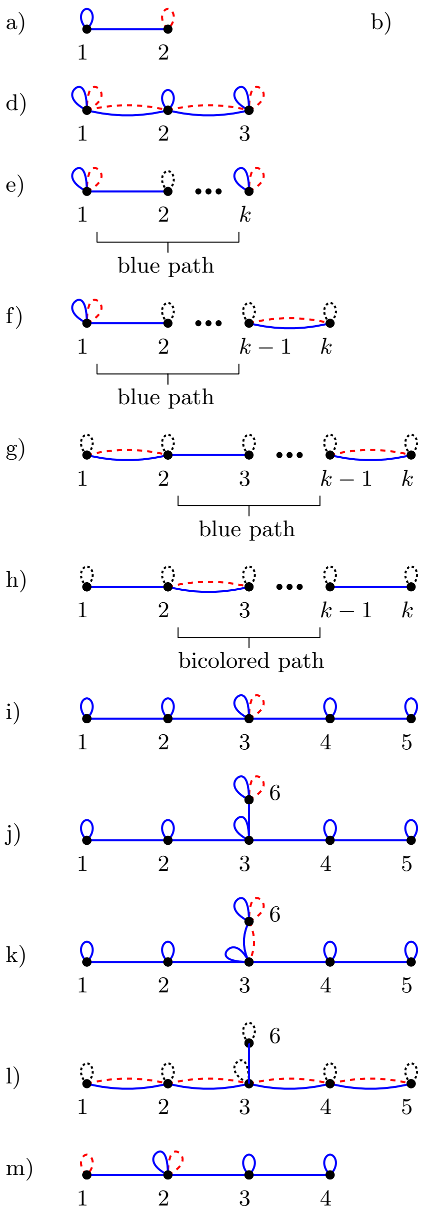
This concludes the main result of this section, Theorem 3.4.

## 3.2 Reflexive trees

We now turn to reflexive trees, and hence in this section, $\widehat{H}$ will always be a reflexive tree. It may have red, blue, or bicoloured loops, but we may again assume that all non-loop unicoloured edges are of the same colour (blue or red).

**Lemma 3.7.** *If $\widehat{H}$ contains one of the reflexive trees from the family $\mathcal{G}$ in Figure 3.5 as an induced subgraph, then* List-S-Hom$(\widehat{H})$ *is NP-complete.*

*Proof.* The signed trees in a), b) and c) are themselves s-cores with more than two edges, so it follows from Theorem 2.6 that they yield NP-complete problems. The signed trees in d) and h) have chains indicated in Figure 3.5, and hence also yield NP-complete problems by Theorem 2.9. The remaining cases are again handled in more general context in the next section, as indicated in Figure 3.5. □



$U = 2 - 2 - 3 - 2$
$D = 2 - 1 - 2 - 2$

Special case of g) of family $\mathcal{J}$ in Figure 3.8.
$k \geq 2$

Special case of f) of family $\mathcal{J}$ in Figure 3.8.
$k \geq 3$

Special case of e) in family $\mathcal{J}$ in Figure 3.8.
$k \geq 4$

$U = 2 - 1 - 1 - 2 - \ldots - (k-1)$
$D = 2 - 3 - \ldots - k - k - (k-1)$
$k \geq 4$

Special case of b) of family $\mathcal{J}$ in Figure 3.8.

Special case of c) of family $\mathcal{J}$ in Figure 3.8.

Special case of k) of family $\mathcal{J}$ in Figure 3.8.

Special case of h) of family $\mathcal{J}$ in Figure 3.8.

Special case of d) of family $\mathcal{J}$ in Figure 3.8.

Figure 3.5: The family $\mathcal{G}$ of signed reflexive trees with NP-complete problems. (The dotted loops can be either blue, red or bicoloured.)

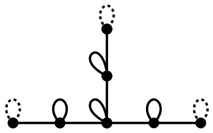


Figure 3.6: The graph $F_2$.

The next lemma is used to prove that in all polynomial cases $\widehat{H}$ is a caterpillar. Although this section is restricted to reflexive graphs, we will prove it in greater generality for future use in a later section. To that end let $F_2$ be the graph in Figure 3.6 where each loop on the three leaves may or may not be present. Thus, $F_2$ represents a family of graphs, but we will abuse notation and simply refer to $F_2$ as any member of that family.

**Lemma 3.8.** *If the underlying graph $H$ contains the graph $F_2$ in Figure 3.6, then the problem* List-S-Hom$(\widehat{H})$ *is NP-complete.*

*Proof.* Deciding if there exists a list homomorphism (of an unsigned graph) to the graph $F_2$ is NP-complete, as stated in [65] (and proved using results in [64] and [66]). It would be natural to attempt a direct reduction of List-Hom$(F_2)$ to List-S-Hom$(\widehat{H})$, as we have done here for the proof of Lemma 3.1. However, this is complicated by the fact that the loops in $\widehat{H}$ can be red, blue, or bicoloured. Therefore, below we proceed on a different path, adapting to our setting the proof of the reflexive case from from [63] (see Theorem 2.3 in that paper).

Suppose that $\widehat{F_2}$ is a subgraph of $\widehat{H}$ with underlying graph $F_2$, and suppose that $\widehat{F_2}$ has been switched so that all non-loop edges are at least blue. Label the leaves of $\widehat{F_2}$ by $0, 1, 2$, and their respective neighbours by $0^+, 1^+, 2^+$, and finally label the central vertex by $c$.

If all the unicoloured loops in $\widehat{F_2}$ are blue, then we may restrict the input to blue (there is no advantage to switching). Thus we may reduce the NP-complete problem List-Hom$(F_2)$ [65] to List-S-Hom$(\widehat{H})$.

Similarly, if all unicoloured loops in $\widehat{F_2}$ are red, then we can switch all non-loop edges to red and apply the same logic. Thus we assume that there are both blue and red unicoloured loops in $\widehat{F_2}$.

We first prove that if some edge $ci^+, i \in \{0, 1, 2\}$ is not bicoloured, then the problem List-S-Hom$(\widehat{F_2})$ is NP-complete by showing that the copy of $\widehat{F_2}$ contains a member of the family $\mathcal{G}$ or $\mathcal{J}$.

Note, any path between a blue loop and a red loop must have a vertex with a bicoloured loop; otherwise, $\widehat{F_2}$ contains a) or b) from family $\mathcal{G}$. Thus at least one of $c, 0^+, 1^+, 2^+$ has a bicoloured loop.

Next, if none of the edges $ci^+$ are bicoloured, then we either have a copy of c) from family $\mathcal{J}$ when there is a bicoloured loop at some $i^+$ or a copy of b) when there is bicoloured loop at $c$. If one of the $ci^+$ edges is bicoloured, then we have a copy of k) from family $\mathcal{J}$. Finally if two of the edges are bicoloured, then we have a copy of h) from family $\mathcal{J}$. (We note that the chain in h) is applicable even if the edges 12 or 45 are unicoloured.) Thus, all edges $ci^+$ are bicoloured.

We now finish the proof using a modification of the proof in [63]. Given distinct $i$ and $j$ in $\{0, 1, 2\}$ and distinct subsets $I$ and $J$ of $\{0, 1, 2\}$, an $(i, I, j, J)$*-chooser* is a path $\widehat{P}$ with endpoints $a$ and $b$, together with a list assignment $L$, such that the following statement holds. For each list homomorphism $f$ from $\widehat{P}$ to $\widehat{F}_2$, either $f(a) = i$ and $f(b) \in I$ or $f(a) = j$ and $f(b) \in J$. Moreover, for each $i' \in I$ and $j' \in J$, there are list homomorphisms $g_1, g_2$ from $\widehat{P}$ to $\widehat{F}_2$ such that $g_1(a) = i, g_1(b) = i'$ and $g_2(a) = j, g_2(b) = j'$.

Suppose $\widehat{P}$ is a $(0, \{0, 1\}, 1, \{1, 2\})$-chooser, $\widehat{P}'$ is $(0, \{1, 2\}, 1, \{2, 0\})$-chooser, and $\widehat{P}''$ is a $(0, \{2, 0\}, 1, \{0, 1\})$-chooser. Let $\widehat{T}$ be the tree obtained by identifying the $b$ vertices in the three choosers and labelling the leaves respectively as $a, a', a''$. It is easy to verify that $\widehat{T}$ admits a list-homomorphism to $\widehat{F}_2$ if, and only if, the triple $(a, a', a'')$ does not map to either $(0, 0, 0)$ or $(1, 1, 1)$. Consequently, we can reduce an instance of Not-All-Equal 3-SAT to List-S-Hom($\widehat{F}_2$). For each clause in the instance, create a copy of $\widehat{T}$ and identify the vertices $(a, a', a'')$ with the three literals in the clause.

It remains to construct the choosers. First, we build a $(0, \{0, 2\}, 1, \{1, 2\})$-chooser. By symmetry we then have $(i, \{i, k\}, j, \{j, k\})$-choosers for any distinct $i, j, k \in \{0, 1, 2\}$. Let $Q$ be a path on $q_0, q_1, \ldots, q_{10}$ with lists

$$\begin{array}{ll}
L(q_0) = \{0, 1\} & L(q_6) = \{0^+, 2^+, 1\} \\
L(q_1) = \{0^+, 1^+\} & L(q_7) = \{0^+, c, 1^+\} \\
L(q_2) = \{0, 1^+\} & L(q_8) = \{0, 2^+, 1\} \\
L(q_3) = \{0^+, c, 1^+\} & L(q_9) = \{0^+, 2^+, 1^+\} \\
L(q_4) = \{0, 2^+, 1\} & L(q_{10}) = \{0, 2, 1\} \\
L(q_5) = \{0^+, 2, 1^+\} &
\end{array}$$

The path $\widehat{Q}$ has all edges blue. In mapping $\widehat{Q}$ to $\widehat{F_2}$ first suppose $q_0$ maps to 0. Then $q_{10}$ either maps to 0, in which case the loop $0^+$ is traversed twice, or $q_{10}$ maps to 2, in which case the loop at $0^+$ and the loop at $2^+$ are each traversed once. In the both cases if the loop at $0^+$ is unicoloured red, then switch at $q_6$. In the latter case, if there is a red loop at $2^+$, then we switch at $q_8$. Note in the latter case the bicoloured edges $0^+c$ and $c2^+$ allow the edges $q_6q_7$ and $q_7q_8$ to be of either colour. A similar reasoning shows $\widehat{Q}$ can map to $\widehat{F_2}$ with $q_0$ mapping to 1 and $q_{10}$ mapping to either 1 or 2 but not to 0. Thus $\widehat{Q}$ is a $(0, \{0, 2\}, 1, \{1, 2\})$-chooser.

The $(0, \{0\}, 1, \{2\})$-chooser $\widehat{R}$ is a path with vertices $r_0, \ldots, r_6$ and lists

$$\{0, 1\}, \{0^+, 1^+\}, \{0, 1^+\}, \{0^+, c\}, \{0, 2^+\}, \{0^+, 2^+\}, \{0, 2\}.$$

All edges are blue. When $\widehat{R}$ maps to the edge $00^+$, no switching is required as $00^+$ is at least blue. When $\widehat{R}$ maps to the path $1, 1^+, 1^+, c, 2^+, 2^+, 2$, switching at $r_2$ (respectively $r_4$) is required when there is a unicoloured red loop at $1^+$ (respectively $2^+$).

The required choosers are defined as follows. First, $\widehat{P}$ is the $(0, \{0\}, 1, \{2\})$-chooser followed by the $(0, \{0, 1\}, 2, \{1, 2\})$-chooser. Next $\widehat{P}'$ is the concatenation of the $(0, \{0\}, 1, \{2\})$-chooser, the $(0, \{1\}, 2, \{2\})$-chooser, the $(1, \{1\}, 2, \{0\})$-chooser, and the $(1, \{1, 2\}, 0, \{0, 2\})$-chooser. Finally $\widehat{P}''$ is the concatenation of the $(0, \{2\}, 1, \{1\})$-chooser and the $(2, \{0, 2\}, 1, \{0, 1\})$-chooser. □

A tree $H$ is a *caterpillar* if it contains a path $P = v_1 \dots v_k$ such that each vertex of $H$ is on $P$ or is adjacent to a vertex of $P$. Note that the path $P$, which we again call the *spine* of $H$, is not unique, and we sometimes make it explicit by saying that $H$ is a caterpillar *with spine $P$*. A vertex $x$ not on $P$ is adjacent to a unique neighbour $v_i$ on $P$, and we call the edge $v_i x$ (with the loop at $x$) *the subtree rooted at $v_i$*. A vertex on the spine can have more than one subtree rooted at it. We say that a signed graph $\widehat{H}$ whose underlying graph $H$ is a reflexive caterpillar is a *good caterpillar with respect to the spine $v_1 \dots v_k$* if the bicoloured edges of $\widehat{H}$ form a connected subgraph, the unicoloured non-loop edges all have the same colour $c$, and there exists an integer $d$, with $1 \leq d \leq k$, such that

- all edges on the path $v_1 v_2 \dots v_d$ are bicoloured, and all edges on the path $v_d v_{d+1} \dots v_k$ are unicoloured with colour $c$,
- all loops at the vertices $v_1, \dots, v_{d-1}$ and all non-loop edges of the subtrees rooted at these vertices are bicoloured,
- all loops at the vertices $v_{d+1}, \dots, v_k$ and all edges and loops of the subtrees rooted at these vertices are unicoloured with colour $c$,
- if $v_d$ has a bicoloured loop, then all children of $v_d$ with bicoloured loops are adjacent to $v_d$ by bicoloured edges,
- if $v_d$ has a unicoloured loop of colour $c$, then all children of $v_d$ have unicoloured loops of colour $c$, and are adjacent to $v_d$ by unicoloured edges, and
- if $d < k$, then the loops of all children of $v_d$ adjacent to $v_d$ by unicoloured edges also have colour $c$.

The vertex $v_d$ will again be called the *dividing vertex*. We also say that $\widehat{H}$ is a good caterpillar *with preferred colour $c$*. Figure 3.7 (on the left) shows an example of good caterpillar with preferred colour blue. We emphasize that in the case $d = k$ (depicted in Figure 3.7 on the right), it is possible (if $v_d$ has a bicoloured loop) that $v_d$ has some children with red loops and some with blue loops, adjacent to $v_d$ by unicoloured edges.

Let $\mathcal{G}$ be the family of signed graphs depicted in Figure 3.5, together with the family of complementary signed graphs where all unicoloured edges and loops are red, rather than blue, and vice versa. Note that the complementary signed graphs

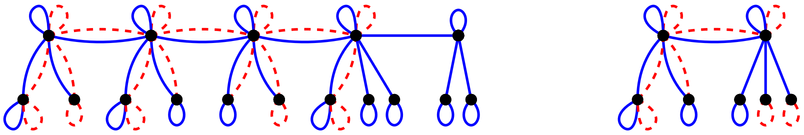


Figure 3.7: Two good caterpillars with preferred colour blue: with $d < k$ (left), with $d = k$ (right).

are not switching equivalent to the original signed graphs because switching does not change the colour of loops.

**Lemma 3.9.** *Let $\widehat{H}$ be a reflexive signed tree. Then $\widehat{H}$ is a good caterpillar if and only if it does not contain any of the graphs in the family $\mathcal{G}$ as an induced subgraph, and the underlying graph $H$ does not contain the graph $F_2$.*

*Proof.* It is easy to see that none of the signed reflexive trees in Figure 3.5 is a good caterpillar. By symmetry, the same is true for their complementary signed graphs. It is also clear that the graph $F_2$ (from Figure 3.6) is not a caterpillar. We proceed to show that if the signed reflexive trees from family $\mathcal{G}$ in Figure 3.5 are excluded as induced subgraphs, then $\widehat{H}$ is a good caterpillar with preferred colour blue. (The complementary exclusions produce a good caterpillar with preferred colour red.) Since the graphs g) are absent, the bicoloured non-loop edges induce a connected subgraph. The exclusion of family h) similarly ensures that all unicoloured non-loop edges induce a connected subgraph. By grouping all bicoloured non-loop edges before all unicoloured non-loop edges, we conclude that there exists a spine $P = v_1 \dots v_k$, and a dividing vertex $v_d$. Thus, all edges between $v_1, \dots, v_{d-1}$, and (since l) is excluded) all edges to their children, are bicoloured. The exclusion of b) and c) ensures each bicoloured non-loop edge has a bicoloured loop on (at least) one of its endpoints. Forbidding the family d) ensures the vertices $v_1, \dots, v_{d-1}$ all have bicoloured loops. The exclusion of b) and c) ensures each bicoloured non-loop edge has a bicoloured loop on (at least) one of its endpoints. Forbidding the family d) ensures the vertices $v_1, \dots, v_{d-1}$ all have bicoloured loops.

The subgraph induced by $v_{d+1}, \dots, v_k$ and their children must contain only blue edges since the edge $v_d v_{d+1}$ is blue and the bicoloured edges induce a connected subgraph. Forbidding a), e), and f) implies all the loops in this subgraph are also blue. (Recall that when we say blue we always mean unicoloured blue.)

Now we distinguish two cases. If $v_d$ has a blue loop then by excluding families a), b), c) and d) we conclude that all edges to its children are blue and all loops of its children are also blue. In the case $v_1 = v_d$, if there is a bicoloured loop on exactly one leaf of $v_1$ (respectively $v_k$), we renumber the vertices so that this leaf becomes the first vertex of the spine, $v_1$. (If it was a leaf of $v_1$, this involves a small shift of subscripts, if it was a leaf of $v_k$, it also involves a reversal of the ordering of subscripts.)

Now suppose that $v_d$ has a bicoloured loop. Excluding family e) ensures that any child of $v_d$ with a bicoloured loop must be adjacent to $v_d$ by a bicoloured edge.

Finally if $d < k$, case m) implies that we can choose the spine so that no child of $v_d$ has a red loop.

Families i), j), and k) ensure when there is a single bicoloured loop or a single bicoloured non-loop edge, the spine can be chosen to begin with this loop or edge. □

**Theorem 3.10.** *Let $\widehat{H}$ be a reflexive tree. If $\widehat{H}$ is a good caterpillar, then the problem* List-S-Hom$(\widehat{H})$ *is polynomial-time solvable. Otherwise, $H$ contains $F_2$ from Figure 3.6, or $\widehat{H}$ contains one of the signed graphs in family $\mathcal{G}$ as an induced subgraph, and the problem is NP-complete.*

Suppose that $\widehat{H}$ is not a good caterpillar. If $H$ is not a caterpillar, then it contains $F_2$ from Figure 3.6, and the problem is NP-complete by Lemma 3.8. Otherwise, $\widehat{H}$ contains an induced subgraph from $\mathcal{G}$, and the problem is NP-complete by Lemma 3.7.

We prove the first statement. Thus assume that $\widehat{H}$ is a good caterpillar, with spine $v_1 \dots v_k$ and dividing vertex $v_d$. By symmetry, we may assume it is a good caterpillar with preferred colour blue. We distinguish three types of rooted subtrees.

- Type $T_1$: a bicoloured edge $v_i x$ with a bicoloured loop on $x$;
- Type $T_2$: a bicoloured edge $v_i x$ with a unicoloured loop on $x$;
- Type $T_3$: a blue edge $v_i x$ with a unicoloured loop on $x$.

There is a general version of min ordering we can use in this context. A *min ordering* of a graph $H$ is a linear ordering $<$ of the vertices of $H$, such that for vertices $x < x', y < y'$, if $xy', x'y$ are both edges in $H$, then $xy$ is also an edge in $H$. It is again the case that if a graph $H$ admits a min ordering, then the list homomorphism problem for $H$ can be solved in polynomial time by arc consistency followed by making the minimum choice in each list [69]. Suppose again that $\widehat{H}$ is a good caterpillar with spine $v_1 \dots v_k$ and preferred colour blue. A *special min ordering* of $\widehat{H}$ is a min ordering of the underlying graph $H$ such that for any vertices $v_i, x, x'$ with edges $v_i x, v_i x'$ we have $x < x'$ if

- the edge $v_i x$ is bicoloured and the edge $v_i x'$ is blue, or
- $x$ has a bicoloured loop and $x'$ a unicoloured loop, or
- $x$ has a blue loop and $x'$ has a red loop.

**Lemma 3.11.** *Every good caterpillar $\widehat{H}$ admits a special min ordering.*

*Proof.* It is again easy to see that the ordering $v_1 < v_2 < \ldots < v_k$ of $V(\widehat{H})$ in which the children of each $v_i$ are ordered between $v_i$ and $v_{i+1}$ is a min ordering of the underlying graph $H$. We may again assume that $\widehat{H}$ has preferred colour blue. To ensure that $<$ is a special min ordering of $\widehat{H}$, we make sure that after each vertex $v_i$ with $i = 1, 2, \ldots, d-1$, we first list the leaves of subtrees of type $T_1$, then the leaves of subtrees of type $T_2$ with blue loop, and last the leaves of subtrees of type $T_2$ with red loop. If $d = k$, then we proceed the same way also after $v_d$, and then we list the leaves of subtrees of type $T_3$ with blue loop, and last the leaves of subtrees of type $T_3$ with red loop. If $d < k$, we list after $v_d$ first the leaves of subtrees of type $T_1$, then the leaves of subtrees of type $T_2$ with blue loop, then the leaves of subtrees of type $T_2$ with red loop, and last the leaves of subtrees of type $T_3$. For vertices $v_i, i > d$, there are only subtrees of type $T_3$, and their leaves can be listed in any order. □

We now describe our polynomial-time algorithm. As in the irreflexive case, we first perform the arc consistency test to check for the existence of a homomorphism of the underlying graphs ($G$ to $H$). Then we also perform the bicoloured arc consistency test. If we obtain an empty list, there is no list homomorphism. Otherwise, taking again the minima of all lists (in the special min ordering $<$) defines a list homomorphism $f\colon G \to H$ of the underlying graphs by [69], and again by bicoloured arc consistency test we have that $f$ maps bicoloured edges of $\widehat{G}$ to bicoloured edges of $\widehat{H}$. Therefore, by Lemma 2.3 and the remarks following it, $f$ is also a list homomorphism of the signed graphs $\widehat{G} \to \widehat{H}$, unless a negative cycle $C$ of unicoloured edges of $\widehat{G}$ maps to a positive closed walk $f(C)$ of unicoloured edges in $\widehat{H}$, or a positive cycle $C$ of unicoloured edges of $\widehat{G}$ maps to a negative closed walk $f(C)$ of unicoloured edges in $\widehat{H}$. The minimum choices in all lists imply that no vertex $x$ of $C$ can be mapped to an image $y$ with $y < f(x)$. We proceed to modify the images of such cycles $C$ one by one, in the order of increasing smallest vertex in $f(C)$ (in the ordering $<$), until we either obtain a homomorphism of signed graphs, or we find that no such homomorphism exists.

Let $w$ be the leaf of the last subtree of type $T_2$ rooted at $v_d$ (we let $w = v_d$ if $v_d$ has no subtree of type $T_2$). We note that if $d < k$, then all edges and loops amongst the vertices that follow $w$ in $<$ are blue, by the properties of a special min ordering. Also note that since the edges of $f(C)$ are unicoloured, they do not include a bicoloured loop on $v_d$ (if there is one). We distinguish three possible cases.

- *At least one vertex $y$ of $f(C)$ satisfies $y \leq w$:*
  The only unicoloured closed walks including $y$ are (red or blue) loops, so $f$ maps the entire cycle $C$ to $y$. As in the reflexive case, we may remove $y$ from all lists of vertices of $C$ and continue seeking a better homomorphism of the underlying graphs ($G$ to $H$).
- *All vertices of $f(C)$ except for $v_d$ follow $w$ in the order $<$ and $d < k$, or $d = k$ and $v_d$ does not have a subtree of Type $T_3$ with red loop:*

In this case $C$ is a negative cycle of unicoloured edges. The subgraph of $\widehat{H}$ induced by the vertices after $w$ (in the order $<$) has only blue edges and loops. Thus there is no homomorphism of signed graphs mapping $\widehat{G} \to \widehat{H}$.

- *All vertices of $f(C)$ except for $v_d$ follow $w$ in the order $<$, $d = k$ and $v_d$ has a subtree of Type $T_3$ with red loop:*
  In this case a fairly complex situation may arise because $f(C)$ can be a closed walk using both red and blue loops, along with blue edges; see below.

We now consider the final case in detail. Since $f$ chooses minimum possible values of images (under $<$), we could only modify $f$ by mapping some vertices of $C$ that were taken by $f$ to a vertex with a blue loop, to vertex with a red loop instead, if lists allow it. We show how to reduce this problem to solving a system of linear equations modulo two, which can then be solved in polynomial time by (say) Gaussian elimination. We begin by considering the pre-image (under $f$) of all vertices in the subtrees of type $T_3$ rooted at $v_d$. We denote by $P$ the set of vertices $v \in V(G)$ with $f(v)$ equal to a vertex with a blue loop and by $N$ the set of vertices $v \in V(G)$ with $f(v)$ equal to a vertex with a red loop. We say that a vertex $x$ of $G$ is a *boundary point* if $f(x) = v_d$. The set of boundary points is denoted by $B$. Thus the pre-image of the subtrees of type $T_3$ rooted at $v_d$ is the disjoint union $B \cup P \cup N$. We now focus on the subgraph $\widehat{G}'$ of $\widehat{G}$ induced by $B \cup P \cup N$. A *region* is a connected component of $\widehat{G}' \setminus B$ together with all its boundary points, i.e. between any pair of vertices in a region there is a path with no boundary point as an internal vertex.

Given a region $r$ and boundary points $x$ and $y$ (not necessarily distinct), we construct (possibly several) boolean equations on the corresponding variables, using the same symbols $x, y$, and $r$. The variables $x, y$ indicate whether or not the corresponding boundary vertices $x$ and $y$ should be switched before mapping them with $f$ (true corresponds to switching), and the variable $r$ indicates whether the region $r$ will be mapped by $f$ to a blue loop or a red loop (true corresponds to a blue loop). The equations depend of the parity and the sign of walks between the two vertices. If $c$ and $d$ denote parities (even or odd), we say a walk $W$ from $x$ to $y$ in $\widehat{G}'$ is a *$(c, d)$-walk* if it contains no boundary points other than $x$ and $y$, the parity of the number of blue edges in $W$ is $c$, and the parity of the number of red edges in $W$ is $d$. The equations generated by the $(c, d)$-walks are as follows.

- *(odd,odd)-walk:* We add the equation $x = y + 1$. This ensures that exactly one of the boundary vertices has to be switched, in particular $x$ and $y$ must be distinct. The image of the walk must be weakly balanced or weakly anti-balanced (as the whole walk maps to exactly one subtree of type $T_3$). An even length walk with an odd number of red edges is neither. However, if we switch at exactly one of the endpoints, we can freely map all of the non-boundary points to a blue loop or a red loop.

- *(even,even)-walk:* We add the equation $x = y$. The reasoning is similar to the previous case.

- *(odd,even)-walk:* We add the equation $x = y + r + 1$. The image of the walk is a closed walk of odd length and positive sign. Thus if both or neither of $x$ and $y$ are switched, then the walk remains positive and $r = 1$. Conversely, switching exactly one of $x$ or $y$ makes the walk negative, and $r = 0$.
- *(even,odd)-walk:* We add the equation $x = y + r$. The argument is analogous to the previous case.

It is possible that there are several kinds of walks between the same $x, y$, but we only need to list one of each kind, so the number of equations is polynomial in the size of $G$. A simple labelling procedure can be used for determining which kinds of walks exist, for given boundary points $x$ and $y$ and a region $r$. We start at the vertex $x$, and label its neighbours $n_x$ by the appropriate pairs $(c, d)$, determined by the signs of the edges $xn_x$. Once a vertex is labelled by a pair $(c, d)$, we correspondingly label its neighbours; a vertex is only given a label $(c, d)$ once even if it is reached with that label several times. Thus a vertex has at most four labels. Any time a vertex receives a new label its neighbours are checked again. The process ends in polynomial time (in the size of the region) as each edge of the region is traversed at most four times. The result is inherent in the labels obtained by $y$.

Finally, for each region we examine the connected component of the non-boundary vertices. Since the arc consistency procedure was done in the first step of the algorithm, all lists of non-boundary points for a given region are the same. Also, by the ordering $<$, these lists must only contain leaves of $v_d$. Thus, the non-boundary vertices of the region must map to a single loop. We ensure the choice of the loop is consistent with the lists of each region. If the lists of vertices of some region do not contain a vertex with a red loop, then we add the equation $r = 1$ for the region. Similarly, if the lists do not contain a blue loop, then we add the equation $r = 0$.

Such a system of boolean linear equations can be solved in polynomial time. Also, the system itself is of polynomial size measured by the size of $\widehat{G}$. This completes the proof.

## 3.3 General trees

In this section we handle signed trees $\widehat{H}$ in general, i.e., trees in which some vertices have loops while others do not. In homomorphism problems, reflexive and irreflexive bipartite target graphs $H$ tend to share some similarities, cf. e.g. [12, 63, 64], and also both tend to be simpler. For instance, the general version of list homomorphisms for graphs with possible loops [65] is significantly more involved than both the reflexive and irreflexive bipartite cases [63, 64]. Similarly, considering general signed trees with possible loops introduces an additional level of difficulty.

To simplify the descriptions, we assume, without loss of generality, that *all non-loop unicoloured edges are blue,* unless noted otherwise. In Figure 3.8 we introduce our main NP-complete cases.

We first focus on signed trees $\widehat{H}$ without bicoloured non-loop edges. If there are no bicoloured loops either, then Theorem 2.7 implies that List-S-Hom($\widehat{H}$) is NP-complete when $\widehat{H}$ has both a red loop and a blue loop, or when the underlying graph is not a bi-arc tree. We now introduce NP-complete cases when bicoloured loops are allowed.

**Lemma 3.12.** *If $\widehat{H}$ contains any of the graphs a)-d) in the family $\mathcal{J}$ in Figure 3.8, then the problem List-S-Hom($\widehat{H}$) is NP-complete.*

*Proof.* For each of the signed graphs a), b), and c) in family $\mathcal{J}$, we can apply Theorem 2.9. The figure lists a chain for each of these forbidden subgraphs.

In the final case d), we reduce Not-All-Equal 3-SAT to List-S-Hom$(H, \pi)$ where $(H, \pi)$ is the signed graph d) in family $\mathcal{J}$. Let $(T', \sigma')$ be the signed graph with the list assignments and signature shown in Figure 3.9. For each clause $(x, y, z)$ in the instance of Not-All-Equal 3-SAT, we create a copy of $(T', \sigma')$ identifying the leaves $x, y, z$ in $T'$ with the variables in the clause.

We claim that $(T', \sigma')$ has a list homomorphism to $(H, \pi)$ if and only if we switch at exactly one or two elements of $\{x, y, z\}$. We can then view the switching at one of $\{x, y, z\}$ as setting the variable to true and, conversely, no switching as setting to false.

Consider a mapping of $(T', \sigma')$ to $(H, \pi)$. It is easy to see that either both $x$ and $m$ are switched or neither is switched. We also observe that if $m$ maps to 1, then exactly one of $m$ or $y$ must be switched. On the other hand, if $m$ maps to 3, then neither or both of $m$ and $y$ is switched. (In the first case the image of the path is a negative walk, while in the second case it is a positive walk.) Thus, when $m$ maps to 1, exactly one of $x$ or $y$ is switched, and when $m$ maps to 3, either both or neither $x$ and $y$ is switched. Finally, if $m$ maps to 1, then we are free to switch or not switch at $z$. On the other hand, if $m$ maps to 3, then we must switch at $z$ if and only if we do not switch at $m$. In conclusion, with $m$ mapping to 1 the following truth values are possible for $x, y, z$ respectively: $1, 0, 0; 1, 0, 1; 0, 1, 0; 0, 1, 1$, and with $m$ mapping to 3 we obtain the possible triples $1, 1, 0$ and $0, 0, 1$ for the variables $x, y, z$. These are precisely the not-all-equal values as claimed. □

If bicoloured edges are present, we use the following result.

**Lemma 3.13.** *If $\widehat{H}$ contains any of the graphs e)-n) in family $\mathcal{J}$ in Figure 3.8, then the problem List-S-Hom($\widehat{H}$) is NP-complete.*

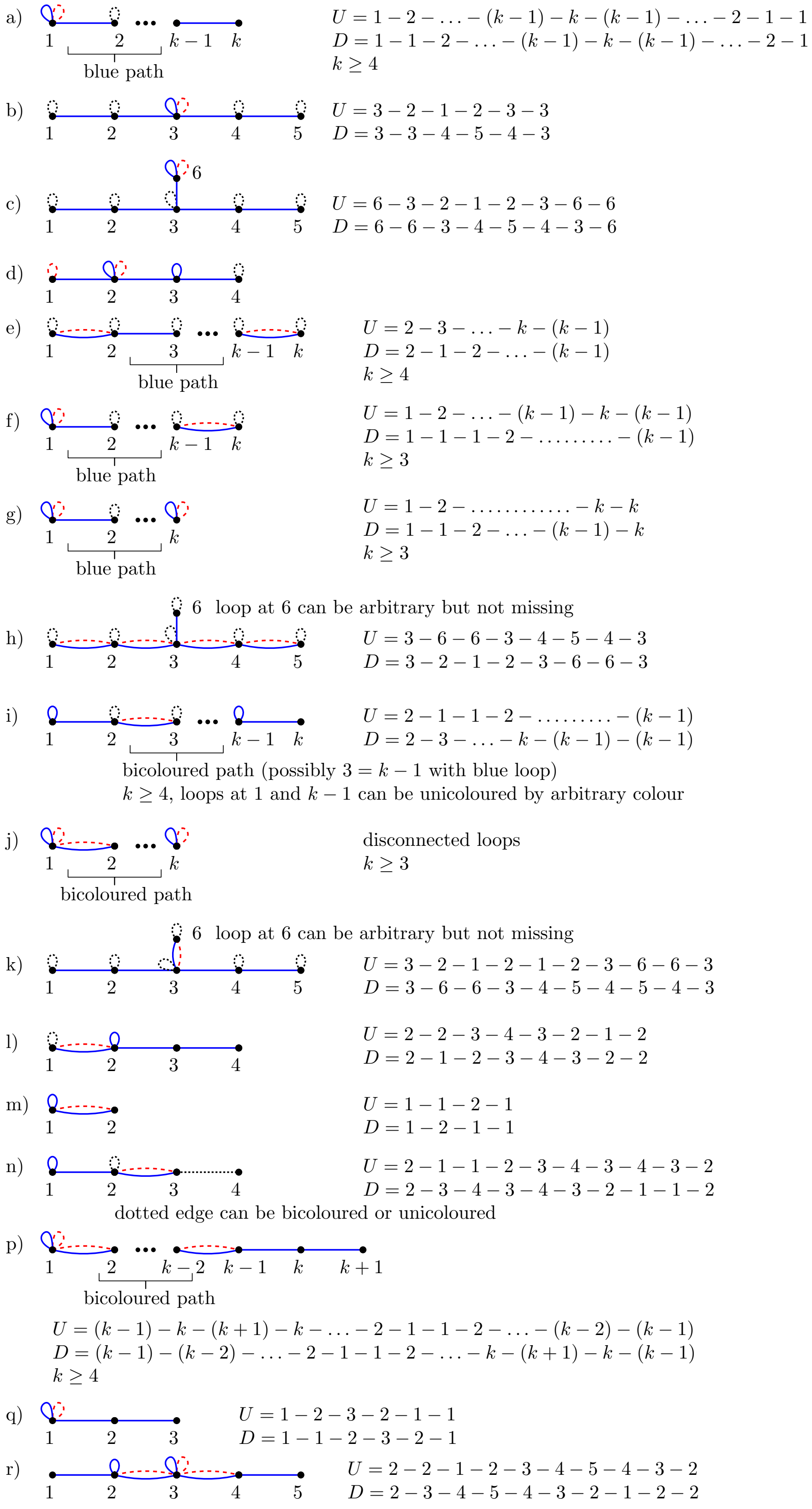

a)
1 2 $k-1$ $k$
blue path
$U = 1 - 2 - \ldots - (k-1) - k - (k-1) - \ldots - 2 - 1 - 1$
$D = 1 - 1 - 2 - \ldots - (k-1) - k - (k-1) - \ldots - 2 - 1$
$k \geq 4$
b)
1 2 3 4 5
$U = 3 - 2 - 1 - 2 - 3 - 3$
$D = 3 - 3 - 4 - 5 - 4 - 3$
c)
6
1 2 3 4 5
$U = 6 - 3 - 2 - 1 - 2 - 3 - 6 - 6$
$D = 6 - 6 - 3 - 4 - 5 - 4 - 3 - 6$
d)
1 2 3 4
e)
1 2 3 $k-1$ $k$
blue path
$U = 2 - 3 - \ldots - k - (k-1)$
$D = 2 - 1 - 2 - \ldots - (k-1)$
$k \geq 4$
f)
1 2 $k-1$ $k$
blue path
$U = 1 - 2 - \ldots - (k-1) - k - (k-1)$
$D = 1 - 1 - 1 - 2 - \ldots\ldots\ldots - (k-1)$
$k \geq 3$
g)
1 2 $k$
blue path
$U = 1 - 2 - \ldots\ldots\ldots\ldots - k - k$
$D = 1 - 1 - 2 - \ldots - (k-1) - k$
$k \geq 3$
6 loop at 6 can be arbitrary but not missing
h)
1 2 3 4 5
$U = 3 - 6 - 6 - 3 - 4 - 5 - 4 - 3$
$D = 3 - 2 - 1 - 2 - 3 - 6 - 6 - 3$
i)
1 2 3 $k-1$ $k$
$U = 2 - 1 - 1 - 2 - \ldots\ldots\ldots - (k-1)$
$D = 2 - 3 - \ldots - k - (k-1) - (k-1)$
bicoloured path (possibly $3 = k-1$ with blue loop)
$k \geq 4$, loops at 1 and $k-1$ can be unicoloured by arbitrary colour
j)
1 2 $k$
bicoloured path
disconnected loops
$k \geq 3$
6 loop at 6 can be arbitrary but not missing
k)
1 2 3 4 5
$U = 3 - 2 - 1 - 2 - 1 - 2 - 3 - 6 - 6 - 3$
$D = 3 - 6 - 6 - 3 - 4 - 5 - 4 - 5 - 4 - 3$
l)
1 2 3 4
$U = 2 - 2 - 3 - 4 - 3 - 2 - 1 - 2$
$D = 2 - 1 - 2 - 3 - 4 - 3 - 2 - 2$
m)
1 2
$U = 1 - 1 - 2 - 1$
$D = 1 - 2 - 1 - 1$
n)
1 2 3 4
$U = 2 - 1 - 1 - 2 - 3 - 4 - 3 - 4 - 3 - 2$
$D = 2 - 3 - 4 - 3 - 4 - 3 - 2 - 1 - 1 - 2$
dotted edge can be bicoloured or unicoloured
p)
1 2 $k-2$ $k-1$ $k$ $k+1$
bicoloured path
$U = (k-1) - k - (k+1) - k - \ldots - 2 - 1 - 1 - 2 - \ldots - (k-2) - (k-1)$
$D = (k-1) - (k-2) - \ldots - 2 - 1 - 1 - 2 - \ldots - k - (k+1) - k - (k-1)$
$k \geq 4$
q)
1 2 3
$U = 1 - 2 - 3 - 2 - 1 - 1$
$D = 1 - 1 - 2 - 3 - 2 - 1$
r)
1 2 3 4 5
$U = 2 - 2 - 1 - 2 - 3 - 4 - 5 - 4 - 3 - 2$
$D = 2 - 3 - 4 - 5 - 4 - 3 - 2 - 1 - 2 - 2$

Figure 3.8: The family $\mathcal{J}$. (The dotted loops can be arbitrary or missing, unless stated otherwise.)

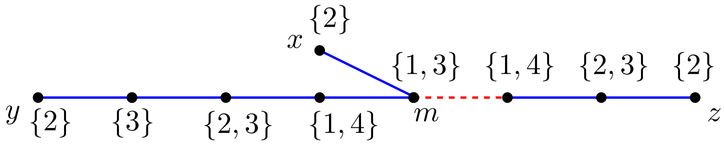


Figure 3.9: The gadget $(T', \sigma')$ for the case d) in family $\mathcal{J}$.

*Proof.* For each of the signed graphs e)-n) in family $\mathcal{J}$, except for the case j), we can apply Theorem 2.9. The figure lists a chain for each of these forbidden subgraphs. The case j) follows from a result in [68] implying that the problem is NP-complete if the vertices with loops of any colour are disconnected. Thus any signed graph $\widehat{H}$ that contains one of the signed graphs in the cases e)-n) of the family $\mathcal{J}$ as an induced subgraph has the problem List-S-Hom($\widehat{H}$) NP-complete. □

In cases p), q), and r) in Figure 3.8 we present three additional NP-complete trees we will use. Note that the case q) is a special case of p), with $k = 2$. The case q) is also a special case of a) in family $\mathcal{J}$, where the chain is specified in general. However, the chain for the case p) when $k > 2$ is different, as shown in the figure. (Note that the absence of a loop at 2 is crucial for the chain in p), and for r), the absence of a loop at 4 is crucial, while the edges 12 or 45 could be blue or bicoloured and the given chain would still apply.)

Thus we have the following lemma.

**Lemma 3.14.** *If $\widehat{H}$ contains any of the graphs p), q), r) in family $\mathcal{J}$ in Figure 3.8, then the problem* List-S-Hom($\widehat{H}$) *is NP-complete.*

If $\widehat{H}$ is a signed graph, the *bicoloured part* of $\widehat{H}$ is the graph $D_{\widehat{H}}$ (with possible loops) consisting of all those edges and loops that occur as bicoloured edges and loops in $\widehat{H}$, and all the vertices they contain. (Thus vertices of $\widehat{H}$ not incident with a bicoloured edge or loop are deleted.) Similarly, the *blue part* of $\widehat{H}$ is the graph $B_{\widehat{H}}$ with possible loops consisting of all those edges (and loops) that are at least blue in $\widehat{H}$. Since we assume all non-loop edges of $\widehat{H}$ are blue, every vertex of $\widehat{H}$ is included in $B_{\widehat{H}}$. (We may think of $B$ as standing for "blue" and $D$ as standing for "double", in the sense of having both colours.)

We now denote by $\mathcal{T}$ the union of all the NP-complete tree families $\mathcal{F}, \mathcal{G}, \mathcal{J}$. There are further cases that cause the problem to be NP-complete. Theorem 2.7 implies, in the context of trees, that the problem is NP-complete if there are no bicoloured edges or loops and there is both a red loop and a blue loop. Any signed graph $\widehat{H}$ which is not irreflexive and has a bicoloured edge but no bicoloured loops yields an NP-complete homomorphism (and hence list homomorphism) problem by Theorem 2.6, since the s-core contains at least one unicoloured loop and one bicoloured edge (counted as two edges). As discussed earlier, if the vertices with loops of any fixed colour induce a disconnected graph, the problem is NP-complete by [68]. Finally, as mentioned earlier, if the bicoloured part $D_{\widehat{H}}$ yields an NP-complete list homomorphism problem, then so does $\widehat{H}$, since for bicoloured inputs, this is the only part of $\widehat{H}$ that can be used. Thus List-S-Hom($\widehat{H}$) is also NP-complete if the unsigned graph $D_{\widehat{H}}$ is not a bi-arc tree, i.e., contains one of the trees in Figures 3 and 4 of [65]. Moreover, if $\widehat{H}$ contains no red loops, then it is also true that if the blue part $B_{\widehat{H}}$ yields an NP-complete list homomorphism problem, then so does $\widehat{H}$. Indeed, if there are no red loops (or edges) in $\widehat{H}$, then for an input signed graph $\widehat{G}$ that has only blue edges, there is no cause for switching. In other words a blue input $\widehat{G}$ admits a signed list homomorphism to $\widehat{H}$ if and only if $G$ admits a list edge-coloured homomorphism to $H$. This is a reduction from the list homomorphism problem for $B_{\widehat{H}}$ to the signed list homomorphism problem for $\widehat{H}$.

We say that a signed tree is *colour-connected* if each of the following subgraphs is connected: the subgraph spanned by non-loop edges that are at least blue, the subgraph spanned by non-loop edges that are at least red, the subgraph spanned by non-loop edges that are bicoloured, the subgraph induced by the vertices with loops that are at least blue, the subgraph induced by the vertices with loops that are at least red, and the subgraph induced by the vertices with loops that are bicoloured.

We call a signed tree $\widehat{H}$ a *good signed tree* if it satisfies the following conditions.

1. If $\widehat{H}$ has no bicoloured edge, then all the loops are of the same colour (red or blue).
2. If $\widehat{H}$ has a bicoloured non-loop edge, then it also has a bicoloured loop, or it has no loops at all.
3. $\widehat{H}$ is colour-connected.
4. The blue part $B_{\widehat{H}}$ is a bi-arc tree.
5. $\widehat{H}$ contains no signed tree from the family $\mathcal{T}$.

### 3.3.1 Assuming no red loops

In this subsection, we assume that $\widehat{H}$ has no red loops. It follows from the previous section, that if such $\widehat{H}$ is not good, then List-S-Hom($\widehat{H}$) is NP-complete. In particular, $\widehat{H}$ is colour-connected, since (as observed before), [68] implies that the problem is NP-complete if the vertices with loops of any colour are disconnected, and the family e) in $\mathcal{J}$ implies that the problem is NP-complete if the subgraph spanned by non-loop edges that are bicoloured is not connected. Also recall that all unicoloured non-loop edges are assumed to be blue, and thus all non-loop edges that are at least red are in fact bicoloured. In the next subsection, we prove this fact (that signed trees that are not good have NP-complete problems) is true if we allow red loops as well.

We first analyse the structure of good signed trees without red loops.

Let $\widehat{H}$ be a good signed tree with no red loops and at least one bicoloured loop. Since the blue part $B_{\widehat{H}}$ is a bi-arc tree, we can use the results of [65] and [66], which together characterize bi-arc trees as trees in which vertices with loops induce a connected subgraph, and which are either obtained from a reflexive caterpillar by deleting the loops at a (possibly empty) subset of leaves (illustrated in Figure 3.10, repeated from Figure 5 of [65]), or obtained from an irreflexive 2-caterpillar in one of the following ways:

1. (possibly) adding a loop at a good vertex $v$, or
2. adding a loop at a good vertex $v$ and on one neighbour $w$ of $v$ which has the property that each neighbour of $w$ other than $v$ is a leaf, or
3. adding a loop at a good vertex $v$ and on a (possibly empty) set of neighbours of $v$ that are leaves.

Here a *good vertex* is a vertex $v$ for which there does not exist a path $P$ of length six with middle vertex $u$ connected to $v$ by a path (possibly of length zero) which is disjoint from $P$. It is easy to see that if $v$ is a good vertex, then there exists a spine in which $v$ is the first vertex, $v = v_1$ (and, in case (2), the vertex $w$ is a child of $v$, not on the spine; similarly in case (3) the leaves of $v$ to which loops have been added are children of $v$ not on the spine). These cases are illustrated in Figure 3.11, repeated here from Figure 6 in [65]. The two 2-caterpillars in that figure will be called Type (a) and Type (b), as shown.

**Proposition 3.15.** *Let $\widehat{H}$ be a good signed tree without red loops but with at least one bicoloured loop.*

*Then $\widehat{H}$ is either*

- *obtained from a good reflexive caterpillar (with spine $v_1, v_2, \ldots, v_k$) by*
  - *removing loops at a subset $S$ of leaves, and*
  - *optionally replacing any bicoloured edges $v_iu$ by blue edges for these leaves $u \in S$, or*
- *is a signed 2-caterpillar (with spine $v_1, v_2, \ldots, v_k$) obtained from a bi-arc tree by*
  - *replacing each edge and loop by a bicoloured edge and loop (respectively),*
  - *optionally, for Type (b) 2-caterpillars, adding a blue loop at a leaf adjacent to $v_1$, and*
  - *optionally adding, at a spine vertex $v_i$ or at a loopless child of a $v_i$, a blue edge leading to a new (loopless) leaf.*

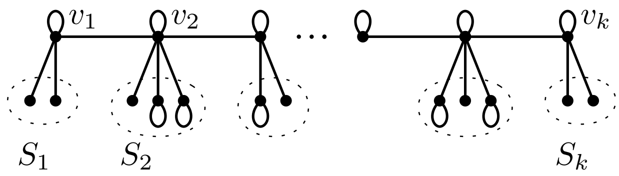


Figure 3.10: Bi-arc caterpillars from [65].

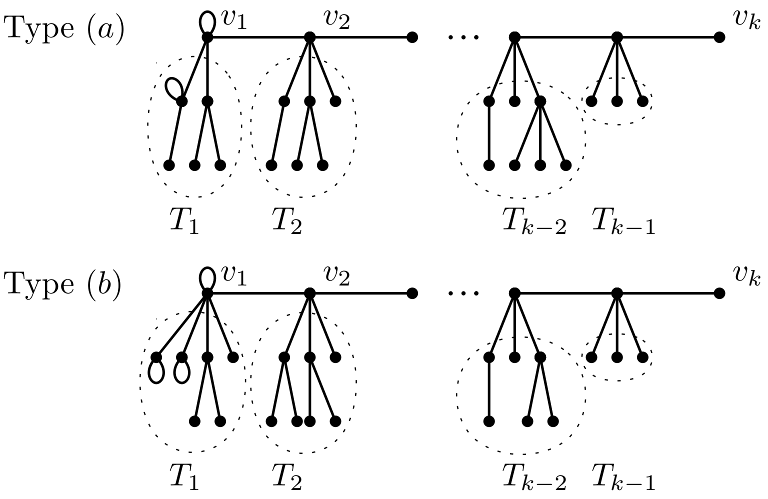


Figure 3.11: Bi-arc 2-caterpillars from [65].

*Proof.* Assume first that the blue part $B_{\widehat{H}}$ is a bi-arc tree of the first type, in other words, a caterpillar with loops on all vertices on the spine and possibly some leaves. We now proceed analogously to the proof of Lemma 3.9, using the absence of signed trees from the family $\mathcal{J}$ instead of those from the family $\mathcal{G}$. We sketch the analogy, and leave the detailed proof to the reader. For this, it helps to refer to the annotations in Figure 3.5 relating the cases of the family $\mathcal{G}$ to the more general trees in the family $\mathcal{J}$. It is also helpful to point out that the case h) of the family $\mathcal{G}$ is closely related to the case i) of the family $\mathcal{J}$ (as well as to b) of the family $\mathcal{F}$). There is a common generalization to all three, but it has a technical formulation we chose to omit, because other cases of the family $\mathcal{J}$ cover the same situations; in particular the reader should note the case n), which is also helpful in the omitted proof. The principal difference from the proof in the reflexive case is caused by the requirement that certain loops in cases h) and l) in $\mathcal{G}$ have to remain present in the corresponding cases in family $\mathcal{J}$. This results in the fact that some vertices $v_1, v_2, \ldots, v_{d-1}$ can have incident blue edges off the spine, as long as they lead to vertices without loops, as enforced by the absence of the signed trees from the family $\mathcal{J}$.

Thus $\widehat{H}$ is indeed a caterpillar obtained from a reflexive signed caterpillar by removing loops at some leaves, and optionally replacing the bicoloured edges by blue edges to some of those leaves.

It remains to consider the case when the blue part $B_{\widehat{H}}$ is a 2-caterpillar obtained by adding suitable loops to an irreflexive bi-arc tree, cf. the two bi-arc trees in Figure 3.11. Specifically, there are two cases to consider.

In the first case, the blue part $B_{\widehat{H}}$ has two loops, at least one of which is bicoloured. According to [66], we may choose the spine so that one loop of $B_{\widehat{H}}$ is at $v_1$ and the other at its child $u$. If the loop at $v_1$ is bicoloured in $\widehat{H}$, then the absence of p) and q) in family $\mathcal{J}$ implies that we may assume the spine consists of bicoloured edges only, and if a vertex $v_i$ on the spine has a (necessarily loopless) neighbour $w$ that is not a leaf, then the edge $v_iw$ is bicoloured. The neighbour $u$ has a loop and needs to be considered separately. We first claim that the edge $uv_1$ must be bicoloured, else $\widehat{H}$ contains the subtree n) from the family $\mathcal{J}$ (with 2 corresponding to $v_1$), or g) from the family $\mathcal{J}$ (with $k = 2$). Moreover, if the loop at $u$ is unicoloured, then $u$ must be a leaf, otherwise $\widehat{H}$ would contain r) from the family $\mathcal{J}$. If the loop at $v_1$ is unicoloured, then the loop at $u$ must be bicoloured (we assumed that a bicoloured loop exists). Now, unless $\widehat{H}$ arose from a reflexive caterpillar, it must contain a) from the family $\mathcal{J}$ (if $uv_1$ is blue), or l) from family $\mathcal{J}$ (if $uv_1$ is bicoloured). (Note that in both cases, the chain applies even if the edges $23, 34$ are bicoloured.) In conclusion, in this case we either have both loops at $v_1$ and $u$ (as well as the edge joining them) bicoloured, or the loop at $v_1$ and the edge $uv_1$ is bicoloured, the loop at $u$ is blue and a leaf. The former situation is depicted on the left of Figure 3.13 (with $u$ depicted on the spine), and the latter situation is a special case of the tree on the right, with only one child ($u$) of $v_1$ having a (blue) loop. Thus going from $D_{\widehat{H}}$ to $\widehat{H}$ we only added a blue loop on a leaf $u$ adjacent to $v_1$, and then added some blue edges leading to leaves from any spine vertex $v_i$, or from any child of $v_2, v_3, \ldots, v_k$, or from any child of $v_1$ other than $u$.

In the second case, the blue part $B_{\widehat{H}}$ has one loop at $v_1$ and possibly several other loops at leaf children of $v_1$. If the loop at $v_1$ is bicoloured, and possibly some of the loops at its children are also bicoloured, then the proof proceeds exactly as in the previous case, concluding that any edge joining two vertices with loops must be bicoloured (else there would be a copy of the subtree n) from the family $\mathcal{J}$) and the children of $v_1$ with loops are leaves. If, say, leaf $u$ has a bicoloured loop and all other loops, including the loop at $v_1$, are blue in $\widehat{H}$, we again obtain a contradiction to the absence of a) from family $\mathcal{J}$ or l) from family $\mathcal{J}$, unless $\widehat{H}$ arose from a reflexive caterpillar. In conclusion, in this case, going from $D_{\widehat{H}}$ to $\widehat{H}$ involved only the addition of blue loops on leaves adjacent to $v_1$, and a possible addition of some blue edges from spine vertices or from non-loop children of spine vertices, leading to leaves as described. □

### 3.3.2 Allowing red loops

We now consider signed graphs $\widehat{H}$ in which red loops are allowed. We denote by $\widehat{H}'$ the signed tree obtained from $\widehat{H}$ by deleting all vertices with red loops.

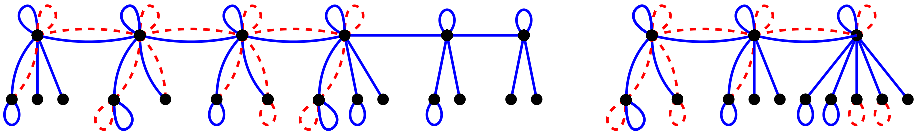


Figure 3.12: Good signed trees obtained from a good reflexive tree by deleting loops.

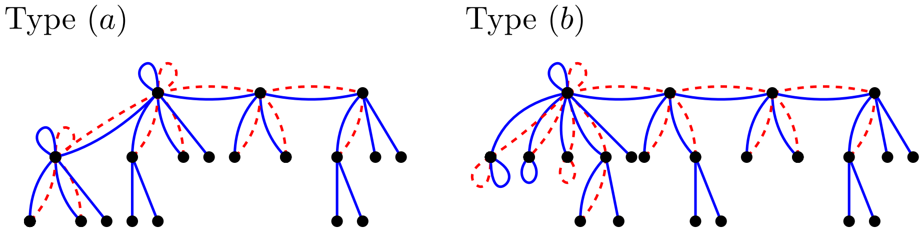


Figure 3.13: Good signed trees obtained from a bi-arc tree as described in Proposition 3.16 .

We focus on the blue part $B_{\widehat{H'}}$ instead of $B_{\widehat{H}}$ because $\widehat{H'}$ has no red loops and satisfies the assumptions of Proposition 3.15.

**Proposition 3.16.** *Let $\widehat{H}$ be a good signed tree with at least one bicoloured loop. Then $\widehat{H}$ is either*

- *obtained from a good reflexive caterpillar (with spine $v_1, v_2, \ldots, v_k$) by*
    - *removing loops at a subset $S$ of leaves, and*
    - *optionally replacing any bicoloured edges $v_iu$ by blue edges for these leaves $u \in S$, or*
- *is a signed 2-caterpillar (with spine $v_1, v_2, \ldots, v_k$) obtained from a bi-arc tree by*
    - *replacing each edge and loop by bicoloured edge and loop (respectively),*
    - *optionally, for Type (b) 2-caterpillars, adding a unicoloured loop at any leaf adjacent to $v_1$, and*
    - *optionally adding, at a spine vertex $v_i$ or at a loopless child of a $v_i$, a blue edge leading to a new (loopless) leaf.*

*Proof.* Since $\widehat{H'}$ (defined above) satisfies the assumptions of Proposition 3.15, the tree $\widehat{H'}$ is described by the proposition, and we now consider where can the vertices of $B_{\widehat{H}} - B_{\widehat{H'}}$ be added, without violating any of the assumptions on $\widehat{H}$. Since the vertices with red loops must form a connected subgraph, they must be adjacent to each other and then to vertices with bicoloured loops. We now take in turn each case in the previous proof.

For the first case, when $\widehat{H'}$ is a caterpillar with reflexive spine vertices, adding a vertex with a red loop adjacent to a vertex on the spine, results in another good reflexive caterpillar with some loops on leaves removed. Adding a vertex with a red loop to a leaf with a red loop either creates a copy of $F_2$, or results in another good caterpillar with a different spine, and possibly different preferred colour, from which some loops at leaves have been removed.

For the second case, we may add a vertex $w$ with a red loop joined to $v_1$ by a bicoloured edge. If we tried to add a vertex $w$ with a red loop adjacent to a child $u$ of $v_1$, then both $uv_1$ and $wu$ would need to be bicoloured, and $u$ would have to have a bicoloured loop or a red loop. In this case, the signed tree either is a caterpillar with reflexive spine and we are in the previous case, or we would obtain, in red, a path with three loops and two non-loops. □

In both cases of the proof above, we note that when the red loops are deleted (without deleting their vertices), we obtain a signed graph which also satisfies the assumptions of Proposition 3.15. Moreover, if the red loops are all changed to be blue, the same conclusion holds. These observations justify the following corollary.

**Corollary 3.17.** *Suppose $\widehat{H}$ is a signed tree. If the blue part $B_{\widehat{H}}$ is not a bi-arc tree, then* LIST-S-HOM*($\widehat{H}$) is NP-complete. If the underlying unsigned tree is not a bi-arc tree, then* LIST-S-HOM*($\widehat{H}$) is NP-complete.*

It follows from the first statement of Corollary 3.17 that if a signed tree is not good then LIST-S-HOM($\widehat{H}$) is NP-complete even if there are red loops in $\widehat{H}$.

We now state our main theorem of this section.

**Theorem 3.18.** *If $\widehat{H}$ is a good signed tree, then* LIST-S-HOM*($\widehat{H}$) is polynomial-time solvable.*

We can explicitly state the dichotomy classification as follows.

**Corollary 3.19.** *Let $\widehat{H}$ be a signed tree.*

*If any of the following conditions apply, then* LIST-S-HOM*($\widehat{H}$) is NP-complete.*

1. *$\widehat{H}$ has no bicoloured loop, but there is a bicoloured (non-loop) edge and a unicoloured loop.*
2. *$\widehat{H}$ has no bicoloured edge, but there is a red loop and a blue loop.*
3. *The bicoloured part $D_{\widehat{H}}$ is not a bi-arc tree, i.e., contains a subgraph from Figures 3 or 4 of [65].*
4. *The blue part $B_{\widehat{H}}$ is not a bi-arc tree, i.e., contains a subgraph from Figures 3 or 4 of [65].*

5. *$\widehat{H}$ contains a signed tree from the family $\mathcal{T}$.*

6. *The set of vertices of $\widehat{H}$ with red (respectively blue, or at least blue, or bicoloured) loops induces a disconnected graph.*

*If none of the conditions apply, then* List-S-Hom($\widehat{H}$) *polynomial-time solvable.*

We also state the result in the more usual complementary way, where the polynomial cases are enumerated first. Note that here all the conditions are required to be satisfied to yield a polynomial case.

**Corollary 3.20.** *Let $\widehat{H}$ be a signed tree. If all of the following conditions apply, then* List-S-Hom($\widehat{H}$) *is polynomial-time solvable.*

1. *If $\widehat{H}$ has a bicoloured non-loop edge, then it has a bicoloured loop, or it has no loops at all.*

2. *If $\widehat{H}$ has no bicoloured edge, then all unicoloured loops are of the same colour.*

3. *The bicoloured part $D_{\widehat{H}}$ is a bi-arc tree.*

4. *The blue part $B_{\widehat{H}}$ is a bi-arc tree.*

5. *$\widehat{H}$ contains no signed tree from the family $\mathcal{T}$.*

6. *The vertices with red (respectively blue, respectively bicoloured) loops induce a connected subgraph of $\widehat{H}$.*

*If at least one of the conditions fails, then* List-S-Hom($\widehat{H}$) *is NP-complete.*

We now return to the proof of Theorem 3.18.

*Proof.* We show that for a good signed tree $\widehat{H}$, the problem List-S-Hom($\widehat{H}$) is polynomial-time solvable. We may assume there is a bicoloured loop; otherwise the result follows from Theorems 2.7 and 3.4. By Proposition 3.16 we distinguish two cases.

For the first case, let $\widehat{H}$ be a good signed tree obtained from a good reflexive caterpillar (with spine $v_1, v_2, \dots, v_k$) by removing loops at a subset $S$ of leaves, and optionally replacing any bicoloured edges $v_iu$ by blue edges for the leaves $u \in S$. (See an illustration in Figure 3.12.) As in the case of reflexive trees, we use a special min ordering of $\widehat{H}$. This means that if a vertex $v_i$ (with $1 \le i \le d-1$) has a non-loop neighbour $u$ connected by unicoloured edge, then $u$ is ordered to come after $v_{i+1}$ in the special min ordering. Now we can use our algorithm for reflexive trees, with the observation that if there is a negative cycle $C$ mapped to a unicoloured edge $v_iu$, then we can remove $u$ from lists of all vertices in $C$ and continue in modifying the images of such cycles.

For the second case, let $\widehat{H}$ be a good signed 2-caterpillar (with spine $v_1, v_2, \ldots, v_k$), obtained from a bi-arc tree by replacing edges and loops by bicoloured edges and loops (respectively), optionally adding unicoloured loops at leaves of $v_1$, and then adding blue edges from the spine or children of the spine to loopless leaves. We set $T = V(\widehat{H})$ and $T' = V(D_{\widehat{H}})$; moreover, we set $L = T \setminus T'$. It follows from Corollary 3.19 that all vertices of $L$ are loopless leaves in $T$ incident with exactly one blue edge. We also note that if two distinct vertices $a$ and $b$ in $T'$ are adjacent in $\widehat{H}$, then they are adjacent by a bicoloured edge.

To prove List-S-Hom($\widehat{H}$) is polynomial-time solvable, we shall construct a suitable majority polymorphism. Recall that for a signed graph $\widehat{H}$, the switching graph $S(\widehat{H})$ is constructed as follows. We represent $\widehat{H}$ as $(H, \pi)$ where the signature $\pi$ has all unicoloured non-loop edges blue (positive), and define $S(\widehat{H})$ to be the edge-coloured graph $(H^+, \pi^+)$ in which each vertex $x$ of $H$ gives rise to two vertices $x, x'$ of $H^+$ and each edge $xy$ of $H$ gives rise to edges $xy, x'y'$ of the colour $\pi(xy)$ in $H^+$ and edges $xy', x'y$ of the opposite colour; this definition also applies to loops, by letting $x = y$. For any vertex $x$ of $H$, we shall denote by $x^*$ one of $x, x'$, and by $s(x^*)$ the other one of $x, x'$. A majority polymorphism of $(H^+, \pi^+)$ is a ternary mapping $F$ on the vertices of $H^+$ such that $F(x^*, y^*, z^*)$ is adjacent to $F(u^*, v^*, w^*)$ in blue (red) provided $x^*$ is adjacent to $u^*$ in blue (red), $y^*$ is adjacent to $v^*$ in blue (red), and $z^*$ is adjacent to $w^*$ in blue (red, respectively), and such that if two arguments from $x^*, y^*, z^*$ are equal, then the assigned value $F(x^*, y^*, z^*)$ is also equal to it. A semi-conservative majority polymorphism assigns $F(x^*, y^*, z^*)$ to be one of the values $x^*, y^*, z^*, s(x^*), s(y^*), s(z^*)$, and a conservative majority polymorphism assigns $F(x^*, y^*, z^*)$ to be one of the values $x^*, y^*, z^*$. As outlined in Section 2.3, if the edge-coloured graph $(H^+, \pi^+)$ admits a semi-conservative majority polymorphism, then the signed list homomorphism problem for $(H, \pi)$ is polynomial-time solvable. We shall in fact construct a conservative majority polymorphism of $(H^+, \pi^+)$.

First, to construct a conservative majority polymorphism $F(x^*, y^*, z^*)$ for triples $(x^*, y^*, z^*)$ from $V(H^+)$, we will of course define values of triples with repetition to be the repeated value,

$$F(x^*, y^*, y^*) = F(y^*, x^*, y^*) = F(y^*, y^*, x^*) = y^*.$$

Now we partition the triples $(x^*, y^*, z^*)$ of distinct vertices of $V(H^+)$ into two sets $R_1$ and $R_2$, where $R_1$ consists of those triples $(x^*, y^*, z^*)$ for which at most one of $x, y, z$ is in $L$, and $R_2$ consists of triples that have at least two of $x, y, z$ in $L$. (The vertices $x, y, z$ of $\widehat{H}$ need not be distinct, as long as $x^*, y^*, z^*$ are distinct.) Note that two triples $(x_1^*, y_1^*, z_1^*), (x_2^*, y_2^*, z_2^*)$ that are coordinate-wise adjacent in $\widehat{H}$ cannot both be in $R_2$, and if they are both in $R_1$, then there is a coordinate $t \in \{x, y, z\}$ such that $t_1 = t_2$, or the edge $t_1t_2$ is bicoloured in $\widehat{H}$.

The definition of $F(x^*, y^*, z^*)$ will differ for triples with $(x, y, z) \in R_1$, where we explicitly describe the value $F(x^*, y^*, z^*)$, and for triples with $(x, y, z) \in R_2$, where we merely prove that a suitable value $F(x^*, y^*, z^*)$ exists.

First we consider the underlying unsigned tree of $\widehat{H}$. It clearly contains all edges of $B_{\widehat{H}}$, but it also contains loops that are red in $\widehat{H}$. By Corollary 3.17, this tree (with vertex set $T$), which we also denote by $T$, is a bi-arc tree, and hence has a majority polymorphism $f$ [65, 66].

We now describe the polymorphism $f$ from [65], assuming, as above, that the bi-arc tree $T$ is one of the trees in Figure 3.11.

In both cases, $T$ is a 2-caterpillar with spine $v_1, v_2, \ldots, v_k$, on which only $v_1$ has a loop, and either there is only one additional loop on a child of $v_1$ (which may have children), or any number of loops on children of $v_1$ which must be leaves. Following (a slight modification of) the notation of [65], we denote by $T_i$ the subtree rooted at the vertex $v_i$ of the spine as shown in Figure 3.13. (Note this differs from our notation above in Section 3.1 where there is a rooted subtree for each child of each vertex of the spine.) Each $T_i$ is ordered by depth first search (in the case of $T_1$ giving higher priority to vertices with loops), and a total ordering of $T$ is obtained by concatenating these DFS orderings from $T_1$ to $T_2$ and so on. We also colour the vertices of $T$ by two colours, in a proper colouring ignoring the self-adjacencies due to the loops. Below we refer to vertices in $T_i$ other than the root $v_i$ as being *inside* $T_i$. The value $f(x, y, z)$ is defined as the majority of $x, y, z$ if two of the arguments $x, y, z$ are equal, and otherwise it is defined according to the following rules.

**Rule (A).** Assume $x, y, z$ are distinct and in the same colour class. Let $r(x)$, $r(y)$, $r(z)$ be the (not necessarily distinct) roots of the trees containing $x, y, z$ respectively, and let $v_m$ be the median of these vertices on the spine. Then $f(x, y, z)$ is the vertex from amongst $x, y, z$ in the tree $T_m$, and if there are more than one in $T_m$, it is the first vertex in the DFS ordering unless one of the following occurs, in which case it is the second vertex in the DFS ordering.

- All three vertices $x, y, z$ lie inside $T_m$ with $m \geq 2$;
- all three vertices $x, y, z$ lie inside $T_1$ and at most one of them has a loop;
- exactly two of $x, y, z$ lie inside $T_1$ and exactly one of them has a loop;
- exactly two of $x, y, z$ lie inside $T_1$, neither has a loop, exactly one of them is adjacent to the unique neighbour of $v_1$ with a loop, and the third vertex of $x, y, z$ is not $v_1$.

**Rule (B).** Assume $x, y, z$ are distinct but not all in the same colour class.

Then $f(x, y, z)$ is the first vertex in the DFS ordering of the two vertices in the same colour class, except when $\{x, y, z\}$ contains $v_1$ and at least one of its leaf neighbours with a loop, in which case $f(x, y, z) = v_1$.

We now use the above conservative majority $f$ on $T$ to define a conservative majority $F$ on triples in $R_1$. We say that a vertex $y$ *dominates* a vertex $x$ in $\widehat{H}$ if any blue (or red) neighbour of $x$ is also blue (red respectively) neighbour of $y$.

**Rule (1).** Assume that at least two of $x^*, y^*, z^*$ are equal, say $y^* = z^*$. As mentioned earlier, we define $F(x^*, y^*, y^*)$ to be the repeated value,

$$F(x^*, y^*, y^*) = F(y^*, x^*, y^*) = F(y^*, y^*, x^*) = y^*.$$

**Rule (2).** Assume that $x^*, y^*, z^*$ are distinct but two of $x, y, z$ are equal. Then for triples $(x^*, y^*, z^*)$ we define the value $F$ to be the first version of the repeated vertex, i.e.,

$$F(x^*, s(x^*), y^*) = F(x^*, y^*, s(x^*)) = F(y^*, x^*, s(x^*)) = x^*,$$

unless $x \in L$ or $x$ has a unicoloured loop in $\widehat{H}$ and $y$ dominates $x$ in $\widehat{H}$, in which case

$$F(x^*, s(x^*), y^*) = F(x^*, y^*, s(x^*)) = F(y^*, x^*, s(x^*)) = y^*.$$

(For example $F(x, x', y) = x'$ and $F(x', x, y) = x'$, but $F(x, x', y) = F(x', x, y) = y$ if $y$ dominates $x$ and $x$ has a unicoloured loop or is in $L$.)

**Rule (3).** Assume $x^*, y^*, z^*$ are distinct and also $x, y, z$ are distinct. For triples $(x^*, y^*, z^*)$, we define $F(x^*, y^*, z^*)$ to be the argument in the same coordinate as $f(x, y, z)$, except if $f(x, y, z) \in L$ and another vertex $t \in \{x, y, z\}$ dominates $f(x, y, z)$, in which case we define $F(x^*, y^*, z^*)$ to be the argument in the same coordinate as $t$.

It is easy to check that if two triples $(x^*, y^*, z^*)$ and $(u^*, v^*, w^*)$ are coordinatewise adjacent in blue (red) in $S(\widehat{H})$, then $(x, y, z)$ is adjacent to $(u, v, w)$ in $T$ and hence $f(x, y, z)$ is adjacent to $f(u, v, w)$ in $T$. We now check that we can also conclude that $F(x^*, y^*, z^*)$ is adjacent to $F(u^*, v^*, w^*)$ in blue (red respectively).

**Case 1.** *Elements $x, y, z$ are distinct and $u, v, w$ are distinct.*

If $f(x, y, z)$ and $f(u, v, w)$ choose the same coordinate, then $F(x^*, y^*, z^*)$ and $F(u^*, v^*, w^*)$ also choose the same coordinate, and hence the values are adjacent in the right colour. (This remains true even if one or both of the choices $F(x^*, y^*, z^*)$ and $F(u^*, v^*, w^*)$ were modified by domination.) Otherwise, suppose without loss of generality that $f(x, y, z) = x$ and $f(u, v, w) = v$. Then the vertex $x$ is adjacent in $T$ to both $u$ and $v$ and hence is not a loop-free leaf, and similarly $v$ is not a loop-free leaf. If $x \neq v$, this means that the edge $xv$ is bicoloured in $\widehat{H}$ and hence $F(x^*, y^*, z^*) = x^*$ is adjacent to $F(u^*, v^*, w^*) = v^*$ in both colours. If $x = v$, the same argument applies if the loop $xv$ is bicoloured, so let us assume it is unicoloured. In this situation, Proposition 3.16 implies that the vertex $x = v$ must be a leaf child of $v_1$ in $T$, and $u = y = v_1$. This is governed by the special case of Rule (B) in the definition of the majority polymorphism $f$, which implies that we would have $f(x,y,z)=v_1$ contradicting $f(x,y,z)=x$, so this case does not occur.

**Case 2.** *Elements $u,v,w$ are distinct but $x,y,z$ are not distinct, say $x=y$ (but perhaps $x^*\neq y^*$).*

This means that $f(u,v,w)$ is adjacent to $x=f(x,y,z)$ in $T$, and $x$ is not a loop-free leaf since it is adjacent to both $u$ and $v$. We now observe that if $F(u^*,v^*,w^*)$ was chosen in the same coordinate as $f(u,v,w)$, then it is in $T'$, and otherwise it was chosen in the same coordinate as some $t\in T'$ (we used Rule (3)). Hence $f(u,v,w)$ or $t$ is adjacent to $x$ by a bicoloured edge in $\widehat{H}$, and $F(u^*,v^*,w^*)$ adjacent to $x^*$ and to $s(x^*)$ in both colours. (Note that $F(x^*,y^*,z^*)$ is $x^*$ regardless of whether $y^*=x^*$ or $y^*=s(x^*)$.)

**Case 3.** *Each triple $x,y,z$ and $u,v,w$ has exactly one repetition.*

Suppose first that the repetition is in different positions, say $x=y$ and $v=w$.

Then $f(x,y,z)=x$ is adjacent to $f(u,v,w)=v$ in $T$. If $x\neq v$, then the edge $xv$ is bicoloured in $\widehat{H}$, and $F(x^*,y^*,z^*)=x^*$ or $F(x^*,y^*,z^*)=s(x^*)$ and $F(u^*,v^*,w^*)=v^*$ or $F(u^*,v^*,w^*)=s(v^*)$ are adjacent in both colours. If $x=v$, then the same argument applies if the loop is bicoloured, and if it is unicoloured, then Proposition 3.16 implies that $u=z$ and $u$ has a bicoloured loop and dominates $x$, whence $F(x^*,y^*,z^*)=z^*$ and $F(u^*,v^*,w^*)=u^*$ and the adjacency is correct. On the other hand, if the repetition is in the same positions, say $x=y, u=v$, then $F(x^*,y^*,z^*)=x^*$ is adjacent to $F(u^*,v^*,w^*)=u^*$ by the definition of $F$, and hence the edge has the correct colour.

**Case 4.** *One triple has all vertices the same, say, $x=y=z$ (but possibly $x^*\neq y^*$).*

If we also have $u=v=w$, then by the pigeon principle some coordinate contains both $F(x^*,y^*,z^*)$ in $(x^*,y^*,z^*)$ and $F(u^*,v^*,w^*)$ in $(u^*,v^*,w^*)$, and so we have the correct adjacency. If $x$ is joined to $u$ by a bicoloured edge in $\widehat{H}$, then $F(x^*,y^*,z^*)$ is joined to $F(u^*,v^*,w^*)$ with the correct adjacency (even in the domination case of Rule 3). As both $x,u\notin L$, the only way for the edge joining them to be unicoloured, is $x=u$ and the edge is a unicoloured loop. In this case by Proposition 3.16, $w$ dominates $u$ and is joined to $u=x$ with a bicoloured edge in $\widehat{H}$, again ensuring the right adjacency.

Now we prove that one can extend the definition of $F$ to $R_2$ so that it remains a polymorphism. (It is of course possible to define each $F(u^*,v^*,w^*)$ for $(u^*,v^*,w^*)\in R_2$ directly, but we found the arguments become more transparent if we only verify that a suitable choice for $F(u^*,v^*,w^*)$ is always possible.)

Consider first values $F(x^*, y^*, z^*)$ with all three vertices $x, y, z$ in $L$. This means that $x$ is incident in $\widehat{H}$ with only one (necessarily blue) edge, say $xx_1$, and similarly for blue edges $yy_1, zz_1$. Thus in the switching graph $S(\widehat{H})$ the vertex $x$ is incident with only one blue edge, namely $xx_1$, and one red edge, namely $xx_1'$, and similarly for $x'$ and for $y, y', z, z'$. Note that $(x_1, y_1, z_1) \in R_1$ because two vertices of $L$ are never adjacent. To choose the value of $F(x^*, y^*, z^*)$, we only need to take into account the existing values of $F(x_1^{**}, y_1^{**}, z_1^{**})$, where $x_1^{**}$ is also either $x_1$ or $x_1'$, and similarly for $y_1^{**}, z_1^{**}$. For example, $(x, y', z')$ is coordinate-wise adjacent in blue only to $(x_1, y_1', z_1')$ and in red only to $(x_1', y_1, z_1)$, and the choices of $F(x_1, y_1', z_1')$ and $F(x_1', y_1, z_1)$ occur in the same coordinate, by the definition of $F$ on $R_1$; if, say, $F(x_1, y_1', z_1') = x_1$ and $F(x_1', y_1, z_1) = x_1'$, then setting $F(x, y', z') = x$ ensures that $F(x, y', z') = x$ is adjacent to $F(x_1, y_1', z_1') = x_1$ in blue and to $F(x_1', y_1, z_1) = x_1'$ in red, as required. Thus in general we can choose the value $F(x^*, y^*, z^*)$ in the same coordinate as $F(x_1^{**}, y_1^{**}, z_1^{**})$, and satisfy the polymorphism property. (Note that this argument applies even if the vertices $x, y, z$ are not distinct.)

It remains to consider the case when exactly two of $x, y, z$ belong to $L$, say $x \in L$ and $y \in L$, with unique (blue) neighbours $x_1$ and $y_1$ in $\widehat{H}$, and $z \notin L$, with neighbours $z_1, \ldots, z_p$. We want to show that there is a suitable value for each $F(x^*, y^*, z^*)$ that maintains the polymorphism property. In the proofs below, we use the fact that $(x^*, y^*, z^*)$ is coordinate-wise adjacent in at least blue to each $(x_1^*, y_1^*, z_i^*)$ and possibly also $(x_1^*, y_1^*, s(z_i^*))$ (if the edge $zz_i$ is bicoloured), and adjacent in at least red to each $(s(x_1^*), s(y_1^*), s(z_i^*))$ and possibly also $(s(x_1^*), s(y_1^*), z_i^*)$ (if the edge $zz_i$ is bicoloured). In any event, we again denote the relevant triples by $(x_1^{**}, y_1^{**}, z_1^{**})$.

Suppose that $x, y, z$ are of the same colour. We observe that $x$ and $y$ cannot lie on the spine, since they are in $L$.

Consider first the case that $x, y$ lie inside the same tree $T_r$. Recall that we say *inside* to mean $x$ and $y$ are not on the spine; thus vertices $x_1$ and $y_1$ also belong to $T_r$, and so $T_r$ is the median tree. If $z$ also lies inside $T_r$, the argument is similar to previous cases where $F(x^*, y^*, z^*)$ is chosen according to the unique neighbours of $x^*, y^*, z^*$. If no neighbour $z_i$ of $z$ is in $T_r$, then each value $F(x_1^{**}, y_1^{**}, z_i^{**})$ is either $x_1^{**}$ or $y_1^{**}$ independently of the location of $z_i$, and hence choosing correspondingly $F(x^*, y^*, z^*) = x^*$ or $= y^*$ will ensure the polymorphism property. If some neighbour $z_i$ of $z$ lies in $T_r, r > 1$, then $z$ is the root of $T_{r-1}$, or of $T_r$, or of $T_{r+1}$, and in this case, we can choose $F(x^*, y^*, z^*) = z^*$. Indeed, in this case, $zz_i$ is bicoloured, and $z_i = x_1 = y_1$ (if $z$ is in $T_{r-1}$ or $T_{r+1}$) or $zx_1, zy_1$ are also bicoloured (if $z$ is in $T_r$). If $r = 1$, then in addition to the previous case the vertices $z, z_1, \ldots, z_p$ can have loops (the vertices $x, y, x_1, y_1$ do not have loops). Since $x_1$ and $y_1$ have the same colour, if the colour of $z_i$ is different (when $z = z_i$), we have the value of $F(x_1^{**}, y_1^{**}, z_i^{**})$ equal to the first or second coordinate, and we can choose $F(x^*, y^*, z^*)$ accordingly. Note that these arguments apply also when $y^* = s(x^*)$, i.e., $x = y$.

If $x \in T_r, y \in T_s$ with $r \neq s$, the arguments are similar. If no neighbour $z_j$ of $z$ lies in $T_r$ or $T_s$, then we can choose $F(x^*, y^*, z^*) = x^*$ or $= y^*$ or $z^*$, depending on which is the median tree, and if some $z_j$ lies in, say, $T_r$, then we can choose $F(x^*, y^*, z^*) = x^*$ or $= z^*$ as above.

Next we consider the case when $x, y, z$ do not have the same colour. We may assume that $x, y$ have different colours, since if $x, y$ have the same colour and $z$ has a different colour then any relevant $F(x_1^{**}, y_1^{**}, z_i^{**})$ is $x_1^{**}$ or $y_1^{**}$ regardless of $z_i$ and we can choose $F(x^*, y^*, z^*)$ accordingly. However, if $z$ has a loop, then we need to consider also $F(x_1^{**}, y_1^{**}, z^{**})$, which could be $z^{**}$ or $s(z^{**})$ if $z$ is the vertex $v_1$; in that case we can set $F(x^*, y^*, z^*) = z^*$.

Thus assume without loss of generality that $x, z$ have the same colour, but the colour of $y$ is different. Then unless $z$ has a loop, $F(x_1^{**}, y_1^{**}, z_i^{**})$ is $x_1^{**}$ or $z_i^{**}$, depending on whether $z_i$ precedes or follows $x_1$ in the DFS ordering. If all neighbours $z_i$ precede $x_1$, or all follow $x_1$, the uniform choice of $F(x_1^{**}, y_1^{**}, z_i^{**})$ allows one to choose $F(x^*, y^*, z^*)$ accordingly. The only situation when some $z_i$ precedes $x_1$ and another $z_j$ follows $x_1$ occurs when $z$ is the root of the tree $T_r$ containing $x$. It is easy to see that in that case we can set $F(x^*, y^*, z^*) = z^*$. Finally, when $z$ has a loop, then we also need to consider $F(x_1^{**}, y_1^{**}, z^{**}) = z^{**}$ and we set $F(x^*, y^*, z^*) = z^*$. □

With this, the dichotomy is fully established. The next chapter will explore signed graphs further beyond the class of signed trees.

# Chapter 4

# List homomorphism problems for separable signed graphs

This chapter is based on:

- [22] Jan Bok, Richard C. Brewster, Tomás Feder, Pavol Hell, and Nikola Jedličková: *List homomorphisms to separable signed graphs.* In Algorithms and Discrete Applied Mathematics - 8th International Conference, CALDAM 2022, volume 13179 of Lecture Notes in Computer Science, pages 22–35, 2022. `https://doi.org/10.1007/978-3-030-95018-7_3`

Understanding the complexity of finding list homomorphisms into irreflexive signed graphs seems to be the heart of the problem and Kim and Siggers [117] have conjectured a classification for these signed graphs. We focus on a special case of irreflexive signed graphs, namely those in which the unicoloured edges form a spanning path or cycle, and classify the complexity of list homomorphisms to these signed graphs. In particular, our results confirm the conjecture of Kim and Siggers for this class of signed graphs.

## 4.1 Introduction and motivation

For weakly balanced irreflexive signed graphs, Kim and Siggers [117] suggest the following conjecture.

**Conjecture 4.1.** *[117] For a weakly balanced irreflexive signed graph $\widehat{H}$, the list homomorphism problem is polynomial-time solvable if $\widehat{H}$ has a special min ordering; otherwise $\widehat{H}$ contains a chain or an invertible pair and the problem is NP-complete.*

In fact, the formulation in the mentioned paper is slightly different: the conjecture is stated as an equivalence of the existence of a special min ordering, the existence of a semi-conservative weak near-unanimity polymorphism, and the absence of invertible pairs and chains; however the aim of the conjecture is the above reformulation as a conjectured dichotomy classification.

Authors there prove that the existence of a special min ordering implies the existence of a semi-conservative majority which means that the problem is polynomial-time solvable; so to confirm their conjecture, one needs to prove that the remaining cases are NP-complete.

In the previous chapter in Theorem 3.4 we confirmed the above conjecture (both in our formulation and in the original formulation from [117]), when $\widehat{H}$ is an irreflexive signed tree: the only NP-complete cases have a chain or an invertible pair.

We now introduce the main definition of this chapter.

**Definition 4.2.** *We say that an irreflexive signed graph $\widehat{H}$ is* path-separable (cycle-separable) *if the unicoloured edges of $\widehat{H}$ form a spanning path (cycle) in the underlying graph of $\widehat{H}$. We shall also call such paths and cycles* hamiltonian. *For brevity, we also say a signed graph is* separable *if it is path-separable or cycle-separable.*

**Main results.** In this chapter we explicitly classify the complexity of the list homomorphism problem for separable signed graphs $\widehat{H}$ — see Theorems 4.3 and 4.5. The descriptions suggest that the polynomial cases are rather rare and very nicely structured. In particular, we confirm Conjecture 4.1 (in both formulations) in the special case when $\widehat{H}$ is a separable signed graph. Moreover, in our results we do not assume that $\widehat{H}$ is weakly balanced.

## 4.2 Path-separable signed graphs

We first observe that for any irreflexive signed graph $\widehat{H}$, the list homomorphism problem for $\widehat{H}$ is NP-complete if the underlying graph $H$ contains an odd cycle, since then the s-core of $\widehat{H}$ has at least three edges and we can apply Theorem 2.6. There is a natural transformation of each general problem to a problem for a bipartite irreflexive signed graph, akin to what is done for unsigned graphs in [65]. This is explained in more detail in [117].

However, for bipartite $H$, we don't have a combinatorial classification beyond the case of trees $H$, except in the case $\widehat{H}$ has no bicoloured edges or loops (when Theorem 2.7 applies), or when $\widehat{H}$ has no unicoloured edges or loops (when the problem essentially concerns unsigned graphs and thus is solved by [65]). Therefore we may assume that both bicoloured and unicoloured edges are present. In this section, we consider irreflexive signed graphs in which the unicoloured edges form a spanning path.

We may assume the edges of $P$ are all blue. In other words, all the edges of the hamiltonian path $P$ are blue, and all the other edges of $\widehat{H}$ are bicoloured. Recall that the distinction between unicoloured and bicoloured edges is independent of switching, thus such a hamiltonian path $P = v_1v_2 \dots v_n$ is unique, if it exists.

We now show that the list homomorphism problem for $\widehat{H}$ is also NP-complete if $H$ contains an induced cycle of length greater than four. Indeed, it suffices to prove this if $H$ is an even cycle of length $k > 4$. If all edges of $H$ are unicoloured, then the problem is NP-complete by Theorem 2.7, since an irreflexive cycle of length $k > 4$ is not a bi-arc graph. If all edges of the cycle $H$ are bicoloured, then we can easily reduce from the previous case. If $H$ contains both unicoloured and bicoloured edges, then $\widehat{H}$ contains an induced subgraph of type a) or b) in the family $\mathcal{F}$ in Figure 3.1, and the problem is NP-complete by Theorem 2.9. (There are cases when the subgraphs are not induced, but the chains from the proof of Theorem 2.9 are still applicable.)

We further identify two additional cases of $\widehat{H}$ with NP-complete list homomorphism problems. An *alternating 4-cycle* is a 4-cycle $v_1v_2v_3v_4$ in which the edges $v_1v_2, v_3v_4$ are bicoloured and the edges $v_2v_3, v_4v_1$ unicoloured. A *4-cycle pair* consists of 4-cycles $v_1v_2v_3v_4$ and $v_1v_5v_6v_7$, sharing the vertex $v_1$, in which the edges $v_1v_2, v_1v_5$ are bicoloured, and all other edges are unicoloured. An alternating 4-cycle has the chain $U = v_1, v_4, v_3; D = v_1, v_2, v_3$, and a 4-cycle pair has the chain $U = v_1, v_4, v_3, v_2, v_1; D = v_1, v_5, v_6, v_7, v_1$. Therefore, if a signed graph $\widehat{H}$ contains an alternating 4-cycle or a 4-cycle pair as an induced subgraph, then the list homomorphism problem for $\widehat{H}$ is NP-complete. Note that the latter chain requires only $v_2v_6$ and $v_3v_5$ to be non-edges. The problem remains NP-complete as long as these edges are absent; all other edges with endpoints in different 4-cycles can be present. If both $v_2v_6$ and $v_3v_5$ are bicoloured edges, then there is an alternating 4-cycle $v_2v_3v_5v_6$. Thus we conclude that the problem is NP-complete if $\widehat{H}$ contains a 4-cycle pair as a subgraph (not necessarily induced), unless exactly one of $v_3v_5$ or $v_2v_6$ is a bicoloured edge.

From now on we will assume that $\widehat{H}$ is a path-separable signed graph with the unicoloured edges (all blue) forming a hamiltonian path $P = v_1, \dots, v_n$. We will assume further that the list homomorphism problem for $\widehat{H}$ is not NP-complete, and derive information on the structure of $\widehat{H}$. In particular, the underlying graph $H$ is bipartite and does not contain any induced cycles of length greater than 4, and $\widehat{H}$ does not contain an alternating 4-cycle or a 4-cycle pair; more generally,

$\widehat{H}$ does not contain a chain. If $\widehat{H}$ has no bicoloured edges (and hence no edges not on $P$), then the list homomorphism problem for $\widehat{H}$ is polynomially solvable by Theorem 2.7, since a path is a bi-arc graph. If there is a bicoloured edge in $\widehat{H}$, then we may assume there is an edge $v_iv_{i+3}$, otherwise there is an induced cycle of length greater than 4.

A *block* in a path-separable signed graph $\widehat{H}$ is a subpath $v_iv_{i+1}v_{i+2}v_{i+3}$ of $P$, with the bicoloured edge $v_iv_{i+3}$. The previous paragraph concluded that $\widehat{H}$ must contain a block. Note that if $v_iv_{i+1}v_{i+2}v_{i+3}$ is a block, then $v_{i+1}v_{i+2}v_{i+3}v_{i+4}$ cannot be a block: in fact, $v_{i+1}v_{i+4}$ cannot be a bicoloured edge, otherwise $\widehat{H}$ would contain an alternating 4-cycle. However, $v_{i+2}v_{i+3}v_{i+4}v_{i+5}$ can again be a block, and so can $v_{i+4}v_{i+5}v_{i+6}v_{i+7}$, etc. If both $v_iv_{i+1}v_{i+2}v_{i+3}$ and $v_{i+2}v_{i+3}v_{i+4}v_{i+5}$ are blocks then $v_iv_{i+5}$ must be a bicoloured edge, otherwise $v_iv_{i+3}v_{i+2}v_{i+5}$ would induce a signed graph of type a) in family $\mathcal{F}$ from Figure 3.1. A *segment* in $\widehat{H}$ is a maximal subpath $v_iv_{i+1}\dots v_{i+2j+1}$ of $P$ with $j \geq 1$ that has all bicoloured edges $v_{i+e}v_{i+e+3}$, where $e$ is even, $0 \leq e \leq 2j-2$. (A *maximal* subpath is not properly contained in another such subpath.) Thus each subpath $v_{i+e}v_{i+e+1}v_{i+e+2}v_{i+e+3}$ of the segment is a block, and the segment is a consecutively intersecting sequence of blocks; note that it can consist of just one block. Two segments can touch as the second and third segment in Figure 4.1, or leave a gap as the first and second segment in the same figure.

In a segment $v_iv_{i+1}\dots v_{i+2j+1}$ we call each vertex $v_{i+e}$ with $0 \leq e \leq 2j-2$ a *forward source*, and each vertex $v_{i+o}$ with $3 \leq o \leq 2j+1$ a *backward source.* Thus forward sources are the beginning vertices of blocks in the segment, and the backward sources are the ends of blocks in the segment. If $a < b$, we say the edge $v_av_b$ is a *forward edge* from $v_a$ and a *backward edge* from $v_b$. In this terminology, each forward source has a forward edge to its corresponding backward source. Because of the absence of a signed graph of type a) in family $\mathcal{F}$ from Figure 3.1, we can in fact conclude, by the same argument as in the previous paragraph, that each forward source in a segment has forward edges to all backward sources in the segment.

We say that a segment $v_iv_{i+1}\dots v_{i+2j+1}$ is *right-leaning* if $v_{i+e}v_{i+e+o}$ is a bicoloured edge for all $e$ is even, $0 \leq e \leq 2j-2$, and *all* odd $o \geq 3$; and we say it is *left-leaning* if $v_{i+2j+1-e}v_{i+2j+1-e-o}$ is a bicoloured edge for all $e$ even, $0 \leq e \leq 2j-2$ and all odd $o \geq 3$. Thus in a right-leaning segment each forward source has *all* possible forward edges (that is, all edges to vertices of opposite colour in the bipartition, including vertices with subscripts greater than $i+2j+1$). The concepts of left-leaning segments, backward sources and backward edges are defined similarly.

We say that a path-separable signed graph $\widehat{H}$ is *right-segmented* if all segments are right-leaning, and there are no edges other than those mandated by this fact. In other words, each forward source has all possible forward edges, and each vertex which is not a forward source has no forward edges. Similarly, we say that a path-separable signed graph $\widehat{H}$ is *left-segmented* if all segments are left-leaning,

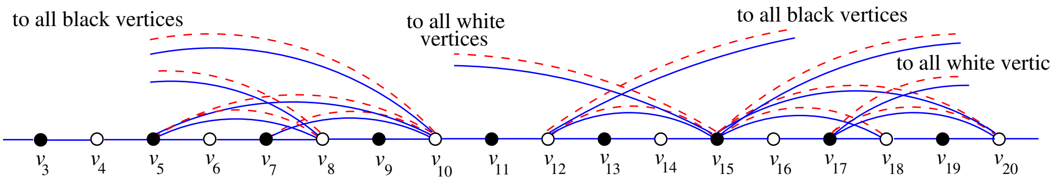


Figure 4.1: An example of a left-right-segmented signed graph. The additional bicoloured edges from all white vertices before $v_{12}$ to all black vertices after $v_{15}$ are not shown.

and there are no edges other than those mandated by this fact. In other words, each backward source has all possible backward edges, and each vertex which is not a backward source has no backward edges. Finally, $\widehat{H}$ is left-right-segmented if there is a unique segment $v_i v_{i+1} \dots v_{i+2j+1}$ that is both left-leaning and right-leaning, all segments preceding it are left-leaning, all segments following it are right-leaning, and moreover there are *additional* bicoloured edges $v_{i-e}v_{i+2j+o}$ for all even $e \geq 2$ and all odd $o \geq 3$, but no other edges. In other words, vertices $v_1, v_2, \dots, v_{i+2j+1}$ induce a left-segmented graph, vertices $v_i, v_{i+1}, \dots, v_n$ induce a right-segmented graph, and in addition to the edges this requires there are all the edges joining $v_{i-e}$ from $v_1, \dots, v_{i-1}$ to $v_{i+o}$ from $v_{i+2j+2}, \dots, v_n$, with even $e$ and odd $o$. A *segmented graph* is a path-separable signed graph that is right-segmented or left-segmented or left-right-segmented. We also refer to vertices with even subscripts as *white* and vertices with odd subscripts as *black*.

See Figure 4.1 for a non-trivial example. There are three segments in the figure, the left-leaning segment $v_5v_6v_7v_8v_9v_{10}$, the left- and right-leaning segment $v_{12}v_{13}v_{14}v_{15}$, and the right-leaning segment $v_{15}v_{16}v_{17}v_{18}v_{19}v_{20}$. Thus this is a left-right-segmented signed graph.

We are now ready to formulate the first of the two main results of this chapter.

**Theorem 4.3.** *Let $\widehat{H}$ be a path-separable signed graph. Then the list homomorphism problem for $\widehat{H}$ is polynomial-time solvable if $\widehat{H}$ is switching equivalent to a segmented signed graph $\widehat{H}$. Otherwise, the problem is NP-complete.*

The rest of the section will be devoted to the proof of this theorem.

### 4.2.1 NP-complete cases

Assume as above that $\widehat{H}$ is a path-separable signed graph for which the list homomorphism problem is not NP-complete, with the unicoloured edges all blue and forming the hamiltonian path $P = v_1, \dots, v_n$. As noted above, there are bicoloured edges forming segments, and possibly some other bicoloured edges.

Consider two consecutive segments, a segment $S$, ending with the block $v_i$, $v_{i+1}$, $v_{i+2}$, $v_{i+3}$, and a segment $S'$ beginning with the block $v_j$, $v_{j+1}$, $v_{j+2}$, $v_{j+3}$, where

$i + 3 \leq j$. Note that there can be no bicoloured edge joining two vertices from the set $\{v_{i+1}, \ldots, v_{j+2}\}$, because the segments $S, S'$ are maximal and consecutive. (If there is any edge joining two vertices of that set, there would have to be one forming a 4-cycle with the unicoloured edges, since the underlying graph has no induced cycles longer than 4.) We emphasize that this crucial observation will be repeatedly used in the arguments in the next three paragraphs, usually without specifically mentioning it.

We claim that either $v_i$ (the last forward source of the segment $S$) has forward edges to all $v_{i+3}, v_{i+5}, v_{i+7}, \ldots, v_s$ for some $s > j + 1$, or symmetrically, $v_{j+3}$ (the first backward source of the next segment $S'$) has a backward edges to all $v_{j-2}, v_{j-4}, \ldots, v_t$ for some $t < i + 2$. In the former case we say that $S$ *precedes* $S'$, in the latter case we say that $S'$ *precedes* $S$.

In case that $S$ precedes $S'$, we now show that $v_i$ (the last forward source of $S$) has forward edges to all $v_{i+3}, v_{i+5}, v_{i+7}, \ldots, v_s$ with $s > j + 1$. (When $S'$ precedes $S$, the argument symmetrically shows that $v_{j+3}$ (the first backward source of $S'$) has a backward edges to all $v_{j-2}, v_{j-4}, \ldots, v_t$ with $t < i + 2$.)

Suppose first that $i$ and $j$ have the same parity (are both even or both odd). There must be other bicoloured edges, otherwise there is a signed graph of type a) from the family $\mathcal{F}$ in Figure 3.1 induced on the vertices $v_i, v_{i+3}, v_{i+4}, \ldots, v_j, v_{j+3}$, and hence a chain in $\widehat{H}$. Therefore, using the above crucial observation, there must be extra edges incident with $v_i$ or $v_{j+3}$. Moreover, as long as there is no edge $v_i v_{j+3}$, there would always be an induced subgraph of type a) from the family $\mathcal{F}$. On the other hand, if $v_i v_{j+3}$ is an edge, then there is a chain with $U = v_i, v_{i+1}, v_{i+2}, v_{i+3}, v_i, v_{j+3}$ and $D = v_i, v_{j+3}, v_{j+2}, v_{j+1}, v_{j+2}, v_{j+3}$, unless $v_{i+2} v_{j+3}$ or $v_i v_{j+1}$ is an edge. (There is no edge $v_{i+3} v_{j+2}$ by our crucial observation above.) Note that both $v_{i+2} v_{j+3}$ and $v_i v_{j+1}$ cannot be edges, because of the chain $U = v_i, v_{i+1}, v_{i+2}, v_{j+3}$, $D = v_i, v_{j+1}, v_{j+2}, v_{j+3}$. Assume that $v_i v_{j+1}$ is an edge. Now we can repeat the argument: there would be a chain with $U = v_i, v_{i+1}, v_{i+2}, v_{i+3}, v_i, v_{j+1}$ and $D = v_i, v_{j+1}, v_j, v_{j-1}, v_j, v_{j+1}$, unless $v_i v_{j-1}$ is an edge. (In this case, we don't need to consider $v_{i+2} v_{j+1}$, since both lie in $\{v_{i+1}, \ldots, v_{j+2}\}$.) We can continue this way until this argument implies the already existing edge $v_i v_{i+3}$, and conclude that $v_i$ is adjacent to all $v_{j+3}, v_{j+1}, v_{j-1}, \ldots, v_{i+3}$, which proves the claim with $s = j + 3$. If $v_{i+2} v_{j+3}$ is an edge, we conclude symmetrically that the claim holds for $t = i$.

Now assume that the parity of $i$ and $j$ is different. This happens, for instance, when $i + 3 = j$: in this case, we would have a 4-cycle pair unless one of $v_i v_{j+2}, v_{i+1} v_{j+3}$ is an edge. Both cannot be edges, as there would be an alternating 4-cycle. This verifies the claim when $i + 3 = j$. Otherwise, there again is a signed graph of type a) from the family $\mathcal{F}$ in Figure 3.1, induced on the vertices $v_i, v_{i+3}, v_{i+4}, \ldots, v_j, v_{j+3}$, so there must be extra edges incident with $v_i$ or $v_{j+3}$. Moreover, there would always remain such an induced subgraph unless there is a vertex $v_p$ with $i + 3 \leq p \leq j$ that is adjacent to both $v_i$ and $v_{j+3}$. Thus let $v_p$ be such a vertex.

We first show that $p$ can be chosen to be $j$ or $i+3$, i.e., that $v_j$ is adjacent to $v_i$, or $v_{j+3}$ is adjacent to $v_{i+3}$. Indeed, if $v_iv_j, v_{i+3}v_{j+3}$ are not edges, then $v_iv_{j+2}$ must also not be an edge (else we obtain a cycle of length greater than 4), and we have the chain

$$U = v_i, v_{i+1}, v_{i+2}, v_{i+3}, v_i, v_p, v_{j+3}, \quad D = v_i, v_p, v_{j+3}, v_{j+2}, v_{j+1}, v_{j+2}, v_{j+3}.$$

(Recall that we are still using the crucial observation that there is no bicoloured edge joining two vertices from the set $\{v_{i+1}, \dots, v_{j+2}\}$.)

Consider now the case that $p = j$ (or symmetrically $p = i+3$). Since $v_iv_p$ is an edge and there are no induced cycles of length greater than 4, the vertex $v_i$ must also be adjacent to $v_{j-2}, v_{j-4}, \dots, v_{i+5}$. Moreover, using again the crucial observation, $v_i$ is also adjacent to $v_{j+2}$, as otherwise we would have the chain $U = v_i, v_{i+1}, v_{i+2}, v_{i+3}, v_i, v_j$ and $D = v_i, v_j, v_{j+1}, v_{j+2}, v_{j+1}, v_j$. Thus the claim holds, with $s = j+2$. In the case $p = i+3$, we obtain a symmetric situation, proving the claim with $t = i+1$.

We conclude that for any two consecutive segments, exactly one precedes the other. For technical reasons, we also introduce two *auxiliary* segments, calling the other segments *normal.* If the first normal segment $S$ of $\widehat{H}$ starts at $v_i$ with $i > 2$, we introduce the *left end-segment* to consist of the vertices $v_1, v_2, \dots, v_i$. We say that the left end-segment *precedes* $S$ if there is no edge $v_{i-2}v_{i+3}$, and we say that $S$ precedes the left end-segment if $v_{i-2}v_{i+3}$ is an edge. Similarly, if the last normal segment $S'$ ends at $v_k$ with $k < n-1$, the *right end-segment* consists of the vertices $v_k, v_{k+1}, \dots, v_n$. The right end-segment *precedes* $S'$ if $v_{k-3}v_{k+2}$ is not an edge, and $S'$ precedes the right end-segment if $v_{k-3}v_{k+2}$ is an edge. Then it is still true that for any two consecutive segments, one precedes the other.

Suppose that we have the special situation where each segment (including the end-segments) precedes the next segment. Consider again the last normal segment $S'$, ending in block $v_{k-3}, v_{k-2}, v_{k-1}, v_k$. We first note that since there are no induced cycles of length greater than 4, and no blocks after the block $v_{k-3}, v_{k-2}, v_{k-1}, v_k$, there cannot be any forward edges from $v_{k-1}, v_k, v_{k+1}, \dots$. By the same argument and the absence of alternating 4-cycles, there are no forward edges from $v_{k-2}$ either.

Our special assumption implies that $v_{k-3}v_{k+2}$ is an edge. Then $v_{k-3}$ has also an edge to $v_{k+4}$, otherwise we have a signed graph of type b) from family $\mathcal{F}$ in Figure 3.1, induced on the vertices $v_{k-1}, v_{k-2}, v_{k-3}, v_{k+2}, v_{k+3}, v_{k+4}$. It is easy to check that the subgraph is induced because otherwise there would be either another block, or an alternating 4-cycle. Then we argue similarly that $v_{k-3}$ has also an edge to $v_{k+6}$, and so on, concluding by induction that the last forward source $v_{k-3}$ has all possible forward edges, and that no vertex after $v_{k-3}$ has any forward edges. Because of the absence of alternating 4-cycles, also the vertex

$v_{k-4}$ has no forward edges, so if $v_{k-5}$ is a forward source in $S'$, we can use the same arguments to conclude it has all forward edges. Thus each forward source $v_{k-o}$ of $S'$ (with odd $o > 3$) has all possible forward edges. Thus the last segment $S'$ is right-leaning, and there are no other forward edges (starting in its vertices or later) than those mandated by this fact.

We proceed by induction from the last segment to the first segment to show that in this special situation all segments are right-leaning and there are no other forward edges at all. The proof is analogous to the preceding paragraph. Consider for instance a segment $S$ ending in block $v_i, v_{i+1}, v_{i+2}, v_{i+3}$: since it precedes the next block, its last forward source, $v_i$ has all possible forward edges until $v_s$ where $s > j + 1$ and the next segment begins with $v_j$. Now the arguments can be repeated, starting avoiding a signed graph of type b) from family $\mathcal{F}$ in Figure 3.1 induced on the vertices $v_{i+2}, v_{i+1}, v_i, v_s, v_{s+1}, v_{s+2}$. Finally, the vertices in the left end-segment cannot have any forward edges, as the absence of other blocks and of induced cycles of length greater than 4 implies there would have to be an edge $v_{f-5}v_f$ where $v_f$ is the first backward source, contrary to the assumption that the left end-segment precedes the first normal segment.

Thus we have proved that if each segment precedes the next segment, then $\widehat{H}$ is a right-segmented graph. By symmetric arguments, we obtain the case of left-segmented signed graphs by assuming that each segment precedes the previous segment. It remains to consider the cases where some segment precedes, or is preceded by, both its left and right neighbours.

It turns out that the case where two segments $S_1$ and $S_3$ both precede the intermediate segment $S_2$ is impossible. Suppose $S_2$ has vertices $v_a, v_{a+1}, \dots, v_b$; in particular, this implies that $v_a v_b$ is an edge. Since each segment before $S_2$ precedes the next segment, the previous arguments apply to the portion of the vertices before $S_2$, and in particular the vertex $v_{a-2}$ is not a forward source, hence has no forward edges; therefore $v_{a-2}$ is not adjacent to $v_b$. By a symmetric argument, there is no edge $v_a v_{b+2}$. There is also no edge $v_{a-1}v_{b+1}$ because it would form an alternating 4-cycle with $v_a v_b$. Therefore, $v_{a-2}, v_{a-1}, v_a, v_b, v_{b+1}, v_{b+2}$ induce a signed graph of type b) from family $\mathcal{F}$ in Figure 3.1, a contradiction.

We conclude that if $S_1$ precedes the next segment $S_2$, then $S_2$ must precede the following segment $S_3$, and so on, and similarly if $S_1$ precedes the previous segment $S_2$.

Hence it remains only to consider the situation where a unique segment $S_2$, with vertices $v_a, v_{a+1}, \dots, v_b$ precedes both its left neighbour $S_1$ and its right neighbour $S_3$ and to the left of $S_1$ each segment precedes its left neighbour, and to the right of $S_3$ each segment precedes its right neighbour. This implies that all segments before $S_2$ are left-leaning, all segments after $S_2$ are right-leaning, while $S_2$ is both left-leaning and right-leaning. To prove that in this situation the signed graph $\widehat{H}$ is left-right-segmented, we show that all edges $v_{a-e}v_{a+o}$ are present, with $e$ even and $o$ odd. This is obvious when $e = 0$ and $o \leq b - a$, by the observations following the definition of a segment. We also have the edges $v_a v_{b+2}, v_{a-2}v_b$ since the segment $S_2$ is both left-leaning and right-leaning. Then we must have the edge $v_{a-2}v_{b+2}$ else there would be the chain $U = v_a, v_{a-1}, v_{a-2}, v_b, D = v_a, v_{b+2}, v_{b+1}, v_b$. We have the edge $v_a v_{b+4}$ since $v_a$ is a forward source, and thus we must also have the edge $v_{a-2}v_{b+4}$, otherwise there is the chain $U = v_a, v_{a-1}, v_{a-2}, v_{b+2}, D = v_a, v_{b+4}, v_{b+3}, v_{b+2}$. Continuing this way by induction on $e+o$ we conclude that all edges $v_{a-e}v_{a+o}$, with $e$ even and $o$ odd, must be present. This completes the proof of NP-completeness for any path-separable signed graph that is not segmented.

### 4.2.2 Polynomial cases

To show that the problem is polynomial when $\widehat{H}$ is a segmented signed graph, we first show that $\widehat{H}$ has a special bipartite min ordering.

We first describe a special bipartite min ordering of the vertices for the case of a right-segmented signed graph, with the unicoloured path $P$. Consider two white vertices $u$ and $v$ that are forward sources such that $u$ precedes $v$ on $P$. Then all forward neighbours of $v$ are also forward neighbours of $u$, and all backward neighbours of $u$ are also backward neighbours of $v$. White vertices $z$ that are not forward sources have edges from all black forward sources $w$ that precede $z$ on $P$. We now construct a bipartite min ordering $<$: order the white vertices that are forward sources in the forward order, then order the remaining white vertices in the backward order. The same ordering is applied on black vertices. It now follows from our observations above that this is a special min ordering. For left-segmented graphs, the ordering is similar.

We now describe *a special min ordering $<$ for the case of a left-right-segmented signed graph $\widehat{H}$*; we consider its vertices in the order of the unicoloured path $P$. We may assume the left-right-leaning segment begins with a white vertex $a$ and ends with a black vertex $b$. We denote by $a'$ the (black) successor of $a$ on $P$, and by $b'$ the (white) successor of $a'$ on $P$. We also denote by $L$ the set of backward sources and by $R$ the set of forward sources of $\widehat{H}$, and denote by $U$ the portion of $P$ from its first vertex to $a'$, and by $V$ the remaining portion of $P$, from $b'$ to its last vertex. Then the min ordering $<$ we construct has white vertices ordered as follows: first the vertices in $U \cap L$ listed in backward order on $P$, then the vertices of $U \setminus L$ listed in forward order on $P$, followed by the vertices in $V \cap R$ in forward order, and then the vertices of $V \setminus R$ in backward order. Similarly, the black vertices are ordered as follows: first the vertices of $V \cap R$ in forward order, then vertices of $V \setminus R$ in backward order, then vertices of $U \cap L$ in backward order, and finally the vertices of $U \setminus L$ in forward order.

To check that $<$ is a min ordering we consider where could a violation of the min ordering property lie. A violation would consist of white vertices $x < y$ and black vertices $s < t$ such that $xt, ys$ are edges of $\widehat{H}$ but $xs$ is not.

We observe that all white vertices in $U$ join all black vertices in $V$, and no black vertex in $U$ joins a white vertex in $V$ with the exception of $a'$ joining $b'$.

First assume that $x$ lies in $U \cap L$: then $x$ is adjacent to all black vertices in $V$, all black vertices before $x$ on $P$ and to its immediate successor and predecessor on $P$. As $x$ belongs to $L$, its predecessor and its successor belong to $U \backslash L$ and are its only unicoloured neighbours. Therefore the non-neighbours of $x$ are in $U \backslash L$, and ordered in $<$ later than its neighbours. A violation cannot occur. Moreover, we observe that its bicoloured neighbours all precede its unicoloured neighbours in $<$ and hence we also verify the special property of a min ordering for $x$.

Next consider the case that $x$ lines in $U \backslash L$: this implies $s \in U$ and since $s < t$, $t \in U$ also. If $s$ follows $x$ on $P$, then $s \in U \backslash L$ giving $t \in U \backslash L$ which means $xt$ cannot be an edge. On the other hand, if $s$ precedes $x$ on $P$, the edge $ys$ implies $y \in U$ and the ordering $x < y$ implies $y \in U \backslash L$ giving $ys$ cannot be an edge. It is easy to check that the special property holds in this case also.

Now assume $x \in V \backslash R$. Since $x < y$, we must have $y \in V \backslash R$ as well and $y$ precedes $x$ on $P$. The only black vertices adjacent to $y$ with a bicoloured edge must lie to the left of $y$ on $P$ and therefore to the left of $x$. Such vertices join $x$ with a bicoloured edge as well. Thus $s \notin R$ and must be adjacent to $y$ by a unicoloured edge. Since $s < t$ and $t \in V$ (no black vertex in $U$ can join $x$), we have $t$ precedes $s$ on $P$ and $t \notin L$. Now $tx$ cannot be an edge. The special property is straightforward to verify.

Finally consider the case when $x \in V \cap R$. Since $x$ is a forward source $s$ must precede $x$ on $P$. If $s \in V$, then $s \in V \backslash R$ as $sx$ is not an edge. Since $xt$ is an edge, $t \in V$ and $s < t$ implies $t \in V \backslash R$ contradicting $xt$ is an edge. If $s \in U$, then $t \in U$ contradicting the existence of at least one of $ys$ or $xt$.

It is easy to check that the special property holds in all cases, and that it also holds for all black vertices. Thus our ordering is a special min ordering and we can use the result from [117] which asserts that the existence of a special min ordering ensures the existence of a polynomial-time algorithm. However, it is not difficult to give a direct proof in this case and we will do so now.

### 4.2.3 Alternative polynomial-time algorithm

We offer the following alternative polynomial-time algorithm for a superclass of segmented graphs. We shall use the special bipartite min ordering of the segmented signed graph $\widehat{H}$ we just described.

**Theorem 4.4.** *Suppose $\widehat{H}$ is a weakly balanced irreflexive signed graph, which contains a unicoloured spanning subgraph. If $\widehat{H}$ has a special bipartite min ordering, then* List-S-Hom*($\widehat{H}$) is polynomial-time solvable.*

*Proof.* We may assume all unicoloured edges in $\widehat{H}$ are blue. We first argue that the special min ordering $<$ has the following additional property: if $xy$ is an bicoloured edge and $uv$ is an edge with $u \leq x, v \leq y$, then the edge $uv$ is also bicoloured. To show this, assume $uv$ is a unicoloured edge. By the special property of our min ordering, $u \neq x, v \neq y$. Now consider a unicoloured path $R$ from $x$ to $u$, which exists by assumption. After $x$ the path $R$ must visit a vertex $z > y$, otherwise the special property is violated. Thus there is an edge $cd$ that crosses $xy$, in the sense that $x < c, d < y$ or $c < x, y < d$, and this is true for any bicoloured edge $x'y', x' \leq x, y' \leq y$. We may assume that $x'y'$ is a bicoloured edge such that no bicoloured edge $x''y''$ has $x'' \leq x', y'' < y'$, and suppose that $c'd'$ crosses $x'y'$ with (say) $x' < c', d' < y'$. Then $x'd'$ must be an edge because $<$ is a min ordering, and it must be bicoloured by the special property at $x'$. This contradicts the minimality of $x'y'$.

The additional property means we can use the following standard polynomial-time list homomorphism algorithm [103]: we perform the consistency test (see [103] Algorithm 4) for blue edges, and then choose in each remaining list the minimum element in the order $<$. (If any list becomes empty there is no list homomorphism [103].) By the standard arguments the properties of a min ordering ensure that this is a list homomorphism of the blue edges [103], Corollary 5.25 . Finally, the additional property ensures that this mapping also preserves bicoloured edges, since if there is any bicoloured edge between two lists, then there is a bicoloured edges joining their minimum elements in $<$. Therefore a list homomorphism exists if and only the consistency test succeeds, and hence the problem is polynomial-time solvable [103]. □

## 4.3 Cycle-separable signed graphs

As an application of Theorem 4.3, we now consider irreflexive signed graphs in which the unicoloured edges form a spanning cycle. Recall that we say that an irreflexive signed graph $\widehat{H}$ is *cycle-separable* if the unicoloured edges of $\widehat{H}$ form a hamiltonian cycle $C$ in the underlying graph $H$. In other words, we have a hamiltonian cycle $C$ whose edges are all unicoloured, and all the other edges of $\widehat{H}$ are bicoloured. In contrast to the path-separable signed graphs, we cannot assume the edges of $C$ are all blue. (The hamiltonian cycle $P = v_1v_2 \ldots v_n$ is again unique.)

We first introduce three cycle-separable signed graphs for which the list homomorphism problem will turn out to be polynomial-time solvable. The signed graph $\widehat{H_0}$ is the 4-cycle with all edges unicoloured blue. The signed graph $\widehat{H_1}$ consists of a a blue path $b = t_0, t_1, t_2, t_3 = w$, a red path $b, s_1, s_2, w$, together with a bicoloured edge $bw$. The signed graph $\widehat{H_\ell}$ consists of a blue path $b, s_1, s_2, w$, a blue path $b = t_0, t_1, t_2, \ldots, t_\ell = w$ (with $\ell \geq 3$ odd), and all bicoloured edges $t_it_j$ with even $i$ and odd $j, j > i + 1$. (Note that this includes the edge $bw$.) These three cycle separable signed graphs are illustrated in Figure 4.2. Note that if the subscript $\ell$ is greater than 0, then it is odd. Moreover, both $H_1$ and $H_3$ have 6 vertices and differ only in the colours of the unicoloured edges forming

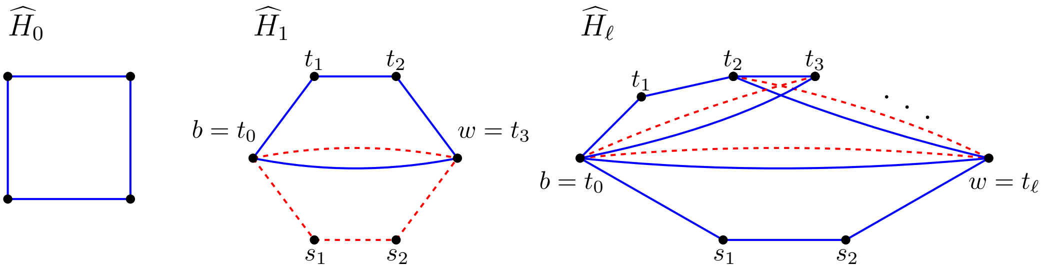


Figure 4.2: The cycle-separable signed graphs $\widehat{H}_0$, $\widehat{H}_1$, and $\widehat{H}_\ell$ with $\ell \geq 3$ odd.

the hamiltonian cycle $C$: $H_1$ has the cycle $C$ unbalanced, and $H_3$ has the cycle $C$ balanced.

For the rest of this section, our goal is to prove the following theorem.

**Theorem 4.5.** *Let $\widehat{H}$ be a cycle-separable signed graph. Then the the list homomorphism problem for $\widehat{H}$ is polynomial-time solvable if $\widehat{H}$ is switching equivalent to $\widehat{H}_0$, or to $\widehat{H}_1$, or to $\widehat{H}_\ell$ for some odd $\ell \geq 3$. Otherwise, the problem is NP-complete.*

### 4.3.1 NP-complete cases

Suppose $\widehat{H}$ is a cycle-separable signed graph for which List-S-Hom($\widehat{H}$) is not NP-complete. As $\widehat{H}$ is irreflexive, it must be bipartite. Further, if $\widehat{H}$ has no bicoloured edges, then it must be a balanced even cycle of length 4, namely it must be $\widehat{H}_0$.

Further suppose that $\widehat{H}$ has at least 6 vertices and let us denote the hamiltonian cycle as $v_0, v_1, \ldots, v_{n-1}, v_0$. Without loss of generality all edges are blue with the possible exception of $v_0v_1$. Consider $\widehat{H} - v_0$. This is switching equivalent to a segmented signed graph with hamiltonian path $v_1, v_2, \ldots, v_{n-1}$ (where $n$ is even), and thus it has the structure described above.

By symmetry, there is a right-leaning segment. Let $v_i$ be the first vertex of the first right-leaning segment. Then $v_{i+1}$ has degree 2 in $\widehat{H} - v_0$ and degree 2 in $\widehat{H}$ unless $v_0v_{i+1}$ is a bicoloured edge. If $v_0v_{i+1}$ is an edge, then $i + 1$ is odd and the forward source $v_i$ sends a bicoloured edge to each $v_o$ for $o$ odd with $i+3 \leq o \leq n-1$. Consequently, $v_0v_{i+1}v_iv_{n-1}v_0$ is an alternating 4-cycle contrary to our assumption that List-S-Hom($\widehat{H}$) is not NP-complete. We conclude $v_{i+1}$ has degree 2 in $\widehat{H}$.

Rename the vertices of the underlying graph $H$ so that $v_0$ is a vertex of degree two. The signed graph $\widehat{H} - v_0$ is path-separable. We are assuming that List-S-Hom($\widehat{H} - v_0$) is not NP-complete, so Theorem 4.3 implies that $\widehat{H} - v_0$ is switching equivalent to a segmented signed graph with spanning path $P = v_1, v_2, \ldots, v_{n-1}$. In particular, we may switch so the spanning path $v_1, v_2, \ldots, v_{n-1}$

of unicoloured edges are all blue, the edge $v_0v_{n-1}$ is blue, and the edge $v_0v_1$ may be red of blue (depending on the sign $C$).

By symmetry, we may assume $v_2$ is adjacent to $v_{n-1}$ (recall $n$ is even), otherwise $\widehat{H}$ contains an induced cycle of length greater than four. Thus we have a 4-cycle $v_0, v_1, v_2, v_{n-1}, v_0$.

Assume first that $v_2, v_3, v_4, v_5, v_2$ is also a 4-cycle. Then we must have that $v_4v_{n-1}$ is also an edge, otherwise $\widehat{H}$ would have 4-cycle pair. If $v_1v_4$ is an edge, then we have an alternating 4-cycle. If $v_6v_{n-1}$ is not an edge, then the path $v_6, v_5, v_4, v_{n-1}, v_0, v_1$ is a case $(b)$ of family $\mathcal{F}$ in Figure 3.1. Similarly, $v_1v_6$ is not an edge and $v_2v_7$ is. By repeating this argument, we conclude that $v_8v_{n-1}, \ldots, v_{n-4}v_{n-1}$ and $v_2v_9, \ldots, v_2v_{n-3}$ must also be edges. Thus using our descriptions of the polynomial path-separable cases, we conclude that $\widehat{H} - v_0 - v_1 = v_2, v_3, \ldots, v_{n-1}$ is just one segment. If $v_{n-1}, v_{n-2}, v_{n-3}, v_{n-4}$ is a 4-cycle, the argument is similar.

If neither $v_2, v_3, v_4, v_5$ nor $v_{n-1}, v_{n-2}, v_{n-3}, v_{n-4}$ is a 4-cycle, then from Theorem 4.3 we conclude that $\widehat{H} - v_0$ is left-right-segmented, with a left-right-leaning segment $S$ not at the end of $P$. This is easy to dismiss, because there would be a $P_5$ (case (b) of family $\mathcal{F}$) involving the segment $S$ and the vertex $v_0$. In conclusion $\widehat{H} - v_0 - v_1$ is just one segment. We label the vertices of $\widehat{H}$ as in Figure 4.2, namely, the segment $v_2, \ldots, v_{n-1}$ is $b = t_0, t_1, \ldots, t_\ell = w$ and the path $v_2, v_1, v_0, v_{n-1}$ is $b, s_1, s_2, w$.

If the cycle $C$ is balanced, then $\widehat{H}$ is switching equivalent to some $\widehat{H_\ell}$ with $\ell \geq 3$. If the cycle $C$ is unbalanced and $n = 6$, then $\widehat{H}$ is switching equivalent to $\widehat{H_1}$. Below we show that both these cases are polynomial-time solvable. Therefore, assume that $\widehat{H}$ has $n > 6$ and its hamiltonian cycle $C$ is unbalanced. We now prove that in this case List-S-Hom($\widehat{H}$) is NP-complete. Without loss of generality, assume the path $b, t_1, \ldots, t_{\ell-1}, w$ is blue and the path $b, s_1, s_2, w$ is red. The vertices of the segment $b, s_1, s_2, w$ are called the *s-vertices* and the vertices of the segment $b, t_1, \ldots, t_{\ell-1}, w$ are called the *t-vertices*.

We will reduce from one of the NP-complete cases of Boolean satisfiability dichotomy theorem of Schaefer [155].

An instance of the problem is a set of Boolean variables $V$ and a set of quadruples $R$ over these variables. The problem asks if there is an assignment of $0, 1$ to the variables so that for every quadruple $(a', b', c', d') \in R$, the Boolean expression $(a' = b' = c' = d') \vee (a' \neq c')$ is satisfied.

Schaeffer [155] proved that a Boolean constraint satisfaction problem is NP-complete except for the well known polynomial cases of 2-SAT, Horn clauses, co-Horn clauses, linear equations modulo two, or when the only satisfying assignments are the all true or the all false assignments. To see that our problem is not expressible as 2-SAT, consider the following three satisfying assignments for $(a', b', c', d')$: $(1, 1, 1, 1), (1, 0, 0, 0), (0, 0, 1, 0)$. It is well known, see e.g. [58] Lemma

4.9, that any problem expressible as 2-SAT has the property that the majority function on three satisfying assignments must also be a satisfying assignment. However, for our three assignments the majority function yields the assignment $(1, 0, 1, 0)$ which is not satisfying. Similarly, our problem is not expressible as Horn clauses (respectively co-Horn clauses) because the minimum (respectively maximum) function on the two satisfying assignments $(0, 1, 1, 0), (1, 1, 0, 0)$ is not a satisfying assignment, cf. Lemma 4.8 in [58]. Finally, our problem is not expressible by linear equations modulo two because the sum modulo two of the three satisfying assignments $(1, 1, 1, 1), (1, 1, 0, 1), (0, 1, 1, 1)$ results in the assignment $(0, 1, 0, 1)$ which is not satisfying, cf. Lemma 4.10 in [58]. Thus our problem is one of the NP-complete cases.

Consider an instance $R$ (over $V$) of our satisfiability problem. We shall now construct a signed graph $\widehat{G}$ with lists such that $\widehat{G}$ admits a list homomorphism to $\widehat{H}$ if and only if there is a satisfying assignment for the set of quadruples $R$.

For each quadruple $(a', b', c', d')$, we construct a copy of the gadget $Q(a', b', c', d')$ (with lists) as in Figure 4.3. Observe that the images of $a', b', c'$ and $d'$ are all fixed. The remaining vertices, which we call *inner vertices*, must all map to the $t$-vertices or must all map to the $s$-vertices.

A variable, say $r$, can appear in multiple quadruples. In this case, there will be a vertex corresponding to $r$ for each quadruple. If $r$ appears multiple times in the first or second coordinate, then we add a vertex $x_r$ to $\widehat{G}$ and a blue edge from $x_r$ to each occurrence of $r$ (in the first two coordinates). The vertex $x_r$ has the list $\{t_1\}$. Similarly, if $r$ appears in the last two coordinates (corresponding to $c'$ and $d'$) of multiple quadruples, then a vertex $y_r$ with list $\{t_{\ell-1}\}$ is added to $\widehat{G}$ together with blue edges joining $y_r$ to each occurrence of $r$ in the last two coordinates. Finally if $r$ occurs in the first or second coordinate in one quadrangle, say as $r'$, and in the third or fourth coordinate of another quadrangle, as $r''$, then a blue path $r', r_1, r_2, \ldots, r_{\ell-1}, r''$ is added to $\widehat{G}$ with $L(r_i) = \{t_i\}$ for each $i = 1, 2, \ldots, \ell - 1$. This path needs only to be added once for the variable $r$. Observe at this point between any two occurrences of $r$ in $\widehat{G}$, there is a blue path whose image under any homomorphism to $\widehat{H}$ is uniquely determined by its lists. Moreover, the image of each such path is a positive walk. Consequently, under any list homomorphism $\widehat{G} \to \widehat{H}$ either no occurrence of $r$ is switched or all occurrences of $r$ are switched.

The repetition of some vertices in lists in the gadget ensures that each longest path can be mapped to a walk of length $k + 1$ in $\widehat{H}$ between $b$ and $w$. If there is some element $r$ contained in the first or second position of two or more quadruples, then we add a new vertex $x_r$ and a blue edge from $x_r$ to each occurrence of $r$. The analogous argument can be used if an element is contained in the third or fourth position of two or more quadruples. Furthermore, if there is some element $r$ in the first or second position of a quadruple and also in the third or fourth position of another quadruple, then we add a blue path $r_1, \ldots, r_{k-2}$ with $L(r_i) = \{t_i\}$ and connect each endpoint with one occurrence by a blue edge. Denote the resulting

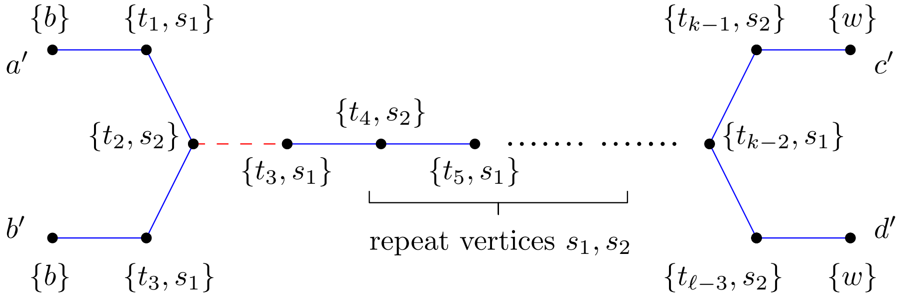


Figure 4.3: A quadruple gadget $Q(a', b', c', d')$.

graph by $\widehat{G}$.

We claim that there exists a satisfying assignment for $R$ if and only if there is a list homomorphism from $\widehat{G}$ to $\widehat{H}$.

Let $f\colon \widehat{G} \to \widehat{H}$ be a homomorphism. We define an assignment $\pi_f\colon V \to \{0, 1\}$ by setting the variable $r$ to 1 if an occurrence of $r$ in $\widehat{G}$ is switched under the homomorphism $f$ and setting $r$ to 0 otherwise. As observed above, under $f$, all occurrences of $r$ are switched or no occurrence is switched. Thus the assignment $\pi_f$ is a well defined.

Furthermore, for a given quadruple gadget, its inner vertices must all choose either the first element described in its list or the second in every possible list homomorphism.

To complete the reduction we show $\pi_f$ is a satisfying truth assignment. Consider a particular copy of the gadget $Q(a', b', c', d')$ and consider the quadruple $(\pi_f(a'), \pi_f(b'), \pi_f(c'), \pi_f(d'))$. Let $P(u, v)$ denote the path from $u$ to $v$ in the copy of $Q$. Initially, $P(a', b')$ and $P(c', d')$ are both positive paths, while

$$P(a', c'), P(a', d'), P(b', c'), P(b', d')$$

are all negative. Switching the end point of a path changes the sign of the path while switching an interior vertex of the path leaves its sign unchanged.

The last thing we need to argue is that certain switching of the endpoints of the quadruple gadget are not possible and some of them are possible. We denote a particular switching as a quadruple $(s_{a'}, s_{b'}, s_{c'}, s_{d'})$ with zeroes and ones with the meaning that one corresponds to not being switched and zero corresponds to being switched. Let us also denote the path between two different occurrence vertices $x, y$ as $P(x, y)$. The crucial observation is that after we fix switchings at endpoints, the signs of the paths $P(a', c')$, $P(a', d')$, $P(b', c')$, $P(b', d')$ (let us call them *main paths*) are invariant upon switching at some inner vertices of the gadget.

The images of $a', b', c'$ and $d'$ under $f$ are uniquely determined by their lists. The remaining vertices, which we call *inner vertices*, must all map to the $t$-vertices or must all map to the $s$-vertices. We consider the two cases.

- *The internal vertices map to the t-vertices.* In this case $P(a', c')$ maps to $b, t_1, t_2, \ldots, t_{\ell-1}, w$. This path is positive in $\widehat{H}$ while $P(a', c')$ is negative in $Q$. Thus exactly one of $a'$ or $c'$ must be switched under $f$. That is, $\pi_f(a') \neq \pi_f(c')$ and $(\pi_f(a'), \pi_f(b'), \pi_f(c'), \pi_f(d'))$ is a satisfying truth assignment.
- *The internal vertices map to the s-vertices.* In this case all of $P(a', c')$, $P(a', d')$, $P(b', c')$, and $P(b', d')$ map to $b, s_1, s_2, s_1, \ldots, s_2, w$. As the four paths in $Q$ and the image (walks) in $\widehat{H}$ are all negative, it follows that either all of $\{a', b', c', d'\}$ are switched or none is switched. Thus $\pi_f(a') = \pi_f(b') = \pi_f(c') = \pi_f(d')$ and again we have a satisfying truth assignment for the quadruple.

Conversely, assume we have a satisfying truth assignment, say $\pi \colon V \to \{0, 1\}$. For each variable $r$, switch all occurrences of $r$ if and only if $\pi(r) = 1$. As observed above, all paths between occurrences of $r$ are (still) positive and admit a list homomorphism to $\widehat{H}$. Consider a particular quadruple $Q(a', b', c', d')$. If $\pi(a') \neq \pi(c')$, then (after switching), the path $P(a', c')$ is positive; hence, we can switch internal vertices in $Q$ to make the path blue. We map it to $b, t_1, \ldots, t_{\ell-1}, w$. We map $P(a', b')$ to $b, t_1, t_2, t_3, b$. Since the edge $bt_3$ is bicoloured, (after possibly switching at the neighbour of $b'$) this mapping is a list homomorphism. A similar analysis works for the path $P(c', d')$. If $\pi(a') = \pi(b') = \pi(c') = \pi(d')$, then all or none of $a', b', c', d'$ are switched. In this case, the internal vertices of $Q$ can be switched so all the edges are red. Hence, $Q$ maps to the path $b, s_1, s_2, w$ which again is the desired list homomorphism.

We divide the analysis of possible switchings into two cases, based on the choice of elements from lists of the quadruple gadget.

- *The first elements of lists are being chosen.* Under this mapping, we have to make sure that $P(a', c')$ is positive. As this path is negative without any switching, we need to switch either at $a'$ and not in $c'$ or vice versa to have this path positive. Switching at $b'$ or $d'$ does not affect this situation. Hence the only admissible quadruples in this case are the ones where the first and the third coordinates differ.
- *The second elements of lists are being chosen.* In this case, all main paths have to map to a negative walk. This is true if we are in case $(0, 0, 0, 0)$ and subsequently in case $(1, 1, 1, 1)$ as in that case, the sign of all main paths remains the same as if we do not switch anywhere. Whenever we have a quadruple with two coordinates having different values, the corresponding path has to be positive and thus we arrive into problems. Hence $(0, 0, 0, 0)$ and $(1, 1, 1, 1)$ are the only possible quadruples for this case.

The argument is concluded by observing that every satisfying assignment corresponds to a list homomorphism with the respective occurrences switched or non-switched, and vice versa.

### 4.3.2 Polynomial cases

Next we show that the list homomorphism problem for $\widehat{H}$ can be solved in polynomial time for all remaining cycle-separable signed graphs.

If $\widehat{H}$ is switching equivalent to $\widehat{H_0}$, then the list homomorphism problem for $\widehat{H}$ is polynomial-time solvable by Theorem 2.7.

Let $\widehat{G}$ together with lists $L$ be an instance of LIST-S-HOM($\widehat{H}$). We may assume $G$ is connected and bipartite. We will call the vertices of parts of bipartition in $\widehat{G}$ black and white as well. First, we try mapping the black vertices of $\widehat{G}$ to the black vertices of $\widehat{H}$. If that fails, we try mapping the white vertices of $\widehat{G}$ to the black vertices of $\widehat{H}$. In the former case we remove all white (respectively black) vertices from the lists of the black (respectively white) vertices in $\widehat{G}$. The latter case is analogous.

First, we preform the arc consistency procedure (cf. [63]) and also the arc consistency procedure for bicoloured edges. If there is a vertex with empty list after this step, then no suitable list homomorphism exists. Otherwise, we define two mappings $f_1$ and $f_2$ as follows.

- $f_1(v) = \min\{t_i : t_i \in L(v)\}$,
- $f_2(v) = \min\{s_i : s_i \in L(v)\}$.

(Observe when $L(v)$ consists of black vertices $f_j(v)$ is the vertex, $t_i$ or $s_i$ for $j = 1$ or $j = 2$, with the smallest index, and conversely for the case $L(v)$ consists of white vertices $f_j(v)$ is the vertex with the largest index.) Let $uv$ be a bicoloured edge of $\widehat{G}$ with $u$ black and $v$ white. Then by arc consistency there is a bicoloured edge between a vertex from $L(u)$ and a vertex from $L(v)$. By the labelling of $\widehat{H}$, it has the form $t_{2i}t_{2j+3}$, where $i \leq j$. By our observation, $f_1(u) = t_{2i'}$ where $i' \leq i$. Similarly, $f_1(v) = t_{2j'+3}$ where $j' \geq j$. This implies $i' \leq j'$ and consequently, $f_1(u)f_2(v)$ is a bicoloured edge. A similar argument applies for $f_2$. Similarly, if $uv$ is a unicoloured edge in $\widehat{G}$ with $u$ black and $v$ white, then there is a (possibly bicoloured) edge $t_{2i}t_{2j+1}$ in $\widehat{H}$ where $t_{2i} \in L(u)$ and $t_{2j+1} \in L(v)$ with $i \leq j - 1$. Again $f_1(u) = t_{2i'}$ with $i' \leq i$ and $f_1(v) = t_{2j'+1}$ with $j' \geq j$. This implies $t_{2i'}t_{2j'+1}$ is an edge of $\widehat{H}$. We conclude that both $f_1$ and $f_2$ are list homomorphisms from $G$ to $H$ (the underlying graphs) with the additional property that vertices adjacent by bicoloured edges in $\widehat{G}$ map to vertices adjacent by bicoloured edges in $\widehat{H}$. We now examine the signs of the unicoloured edges and determine the switchings required to define a list homomorphism from $\widehat{G}$ to $\widehat{H}$. We make the following key observation. Due to the ordering on the vertices if, for example, $f_1(u)f_1(v)$ is a unicoloured edge in $\widehat{H}$ (again $u$ is black and $v$ is white), then under no list homomorphism of $\widehat{G}$ to $\widehat{H}$ (with lists $L$) does $uv$ map to a bicoloured edge. (If such a mapping did exist, then the bicoloured edge would remain as a possible image during the consistency check, and a bicoloured edge would have end points occurring first in the ordering $<$.)

If $b \in L(v)$ for some black vertex $v$, then $f_1(v) = f_2(v) = b$. Analogously, if $w \in L(v)$ for some white vertex $v$, then $f_1(v) = f_2(v) = w$. That is, any vertex that can map to $b$ (respectively $w$) will be mapped to $b$ (respectively $w$). Moreover, when examining the resigning of vertices (below), if there is no resigning that works when $v$ maps to $b$, then there is no homomorphism at all, as $b$ dominates all white vertices (in $\widehat{H}$). Similarly $w$ dominates all black vertices.

Consequently, we can partition the vertices of $\widehat{G}$ into those mapped to $b$ or $w$ under $f_1$ and under $f_2$ and those vertices that can only map to interior vertices of the two segments, i.e. to $t_1, \ldots, t_{\ell-1}$ and $s_1, s_2$. The vertices in the pre-images $f_1^{-1}(b) = f_2^{-1}(b)$ and $f_1^{-1}(w) = f_2^{-1}(w)$ are called *boundary vertices*. Removing the boundary vertices from $\widehat{G}$ leaves a union of components. Consider such a component $K$. The subgraph of $\widehat{G}$ induced by $K$ is called a *region*. For each region, either its vertices all map to $s$-vertices or all to $t$-vertices. (This is similar to the polynomial algorithm of Section 3.3.) We now examine how to test if there is a switching of the boundary vertices of $\widehat{G}$ so that each region maps to $\widehat{H}$.

First suppose that $\widehat{H}$ is switching equivalent to $\widehat{H_\ell}$ with odd $\ell \geq 3$. Let $K$ be some region of $\widehat{G}$. If the lists of vertices of $K$ contain only $s$-vertices or only $t$-vertices, then there is no choice and we will use mappings $f_2$ or $f_1$, respectively. Now suppose that the lists of vertices of $K$ contains both $s$-vertices and $t$-vertices. We claim we can use $f_1$ to map $K$ to $\widehat{H}$. If there is any list homomorphism $\widehat{G} \to \widehat{H}$ under which $K$ maps to $s$-vertices, then there is a switching of $\widehat{G}$ such that $K$, together with its boundary vertices, induces a subgraph having only blue edges. (Recall $b, s_1, s_2, w$ is a blue path.) The mapping $f_1$ restricted to $K$ and its boundary vertices is a homomorphism of $G$ to $H$. As each edge in the segment containing the $t$-vertices is at least blue, $f_1$ is a homomorphism of the induced subgraph to $\widehat{H}$. Thus for any region that has both $s$-vertices and $t$-vertices in its lists after the consistency checks, we may assume if is mapped $\widehat{H}$ under $f_1$. The remaining regions must map to $s$-vertices and we may assume they are mapped using $f_2$. Moreover, by our key observation above, the edges mapping to unicoloured edges under these mappings must map to unicoloured edges under any mapping. In particular, the discovery of a cycle consisting of unicoloured edges in $\mathcal{G}$ whose sign does not agree with the sign of its image under our use of $f_1$ and $f_2$ certifies that $\mathcal{G}$ is a no-instance of the problem.

It now remains to determine the switching of boundary vertices to ensure the signs of all unicoloured edges are positive.

That can be done by considering the subgraph of $\widehat{G}$ induced by edges that map to unicoloured edges of $\widehat{H}$ (under our choices of $f_1$ and $f_2$), identifying the resulting regions (of this unicoloured edge subgraph) between boundary points, and using a system of equations modulo two similar to above. Specifically, for each boundary vertex $u$, a Boolean variable $x_u$ is created. If there is a positive path of unicoloured edges between two boundary vertices $u$ and $v$ in the same region, then the equation $x_u = x_v$ is added to the system. On the other hand, if there is a negative path from $u$ to $v$, then $x_u \neq x_v$ is added to the system. There is a solution to the system if and only if there is a set of switchings on boundary vertices so that the homomorphism from $G$ to $H$ also preserves edge colours giving a homomorphism from $\widehat{G}$ to $\widehat{H}$. This completes the analysis of this case.

Now suppose that $\widehat{H}$ is switching equivalent to $\widehat{H_1}$. The $t$-vertices are the blue path $b, t_1, t_2, w$ and the $s$-vertices are the red path $b, s_1, s_2, w$. By the arc consistency check, all bicoloured edges are mapped to $bw$. Thus the edges of the regions and their edges to boundary vertices are all unicoloured.

Let $K$ be a region. We need to decide whether $K$ will be mapped to the $s$-vertices or the $t$-vertices and to determine an appropriate switching of the boundary vertices. (Under any list homomorphism to $\widehat{H_1}$ the image of $K$ is a single edge. In particularly, $K$ must be balanced. Hence we can switch vertices of $K$ so that it is all blue, and then identify the black vertices and identify the white vertices so that $K$ is now a single blue edge.) In particular, the region has boundary vertices consisting of $u_1, u_2, \ldots$ mapping to $b$ and $v_1, v_2, \ldots$ mapping to $w$. We need to switch the boundary vertices so that the subgraph induced by region and its boundary vertices is a balanced subgraph that maps to the $t$-vertices or an anti-balanced subgraph that maps to the $s$-vertices.

Note that under any suitable homomorphism, the input graph $\widehat{G}$ must be switched so that walks from $v_i$ to $v_{i'}$ in $K$ must be positive and similarly for $u_i$ to $u_{j'}$. To determine the switching we solve (in polynomial time) a system of linear equations modulo two. (This is analogous to our method for irreflexive trees described more fully in Section 3.1.) For each boundary vertex $v_i$, we will introduce a variable $x_i$, and similarly for each $u_j$, we introduce a variable $y_j$.

If there is a positive walk from $v_i$ to $v_{i'}$, we add $x_i = x_{i'}$ to the system (and so on for all pairs from $x_1, x_2, \ldots$). If there is a negative walk, we add $x_i \neq x_{i'}$. Similarly for $u_1, u_2, \ldots$ with the variables $y_j$.

Finally, we introduce equations to determine the signs of walks from $v_i$ to $u_j$. If $K$ can only map to $t$-vertices, then the walks between $v_i$ and $u_j$ must be positive. We code the needed switching with $x_i = y_j$ if there is a positive walk from $v_i$ to $u_j$ in $\widehat{G}$ and $x_i \neq y_j$ if there is a negative walk. Similarly, if $K$ can only map to the $s$-vertices, we must switch to make all walks between $v_i$ and $u_j$ negative. If $K$ has a choice, then we can use a variable $z_k$ that is 0 if $K$ maps to $t$-vertices and 1 if $K$ maps to $s$-vertices. For this situation, we add the equation $x_i = y_i + z_k$ if there is a positive walk from $v_i$ to $u_j$ and $x_i = y_i + z_k + 1$ if there is a negative walk.

## 4.4 Conclusion

It seems difficult to give a full combinatorial classification of the complexity of list homomorphism problems for general signed graphs. For irreflexive signed graphs, which are in a sense the core of the problem, there is a conjectured classification in [117]. Here we have obtained a full dichotomy classification in the special case of separable irreflexive signed graphs. The classification confirms the dichotomy conjecture of [117] for this case, and also confirms that the only polynomial cases enjoy a special min ordering and the only NP-complete cases have chains or invertible pairs, as also conjectured in [117].

# Part II

# Graph coverings

# Chapter 5

# Introduction to graph coverings

This chapter is based on:

- [27] Jan Bok, Jiří Fiala, Petr Hliněný, Nikola Jedličková, and Jan Kratochvíl: *Computational Complexity of Covering Multigraphs with Semi-Edges: Small Cases.* In 46th International Symposium on Mathematical Foundations of Computer Science, MFCS 2021, volume 202 of Leibniz International Proceedings in Informatics (LIPIcs), pages 21:1–21:15, 2021.
- [29] Jan Bok, Jiří Fiala, Nikola Jedličková, Jan Kratochvíl, and Michaela Seifrtová: *Computational Complexity of Covering Disconnected Multigraphs.* In Fundamentals of Computation Theory, FCT 2021, volume 12867 of Lecture Notes in Computer Science, pages 85–89, 2021.
- [28] Jan Bok, Jiří Fiala, Nikola Jedličková, Jan Kratochvíl, and Paweł Rzążewski: *List covering of regular multigraphs.* In Combinatorial Algorithms - 33rd International Workshop, IWOCA 2022, volume 13270 of Lecture Notes in Computer Science, pages 228–242, 2022. `https://doi.org/10.1007/978-3-031-06678-8_17`

## 5.1 Graph coverings and complexity

The notion of a *graph covering* (locally bijective graph homomorphism) is a discretization of coverings between surfaces or topological spaces, a notion well known and deeply studied in classical topology [10, 18, 140, 174]. Graph coverings have found many applications. Primarily as a tool for construction of highly symmetric graphs [16, 59, 87, 95], or for embedding complete graphs in surfaces of higher genus [93, 94, 154].

Graph coverings attracted attention of computer scientists as well. Angluin [7] exploited graph covers when introducing models of local computations, namely by showing that a graph and its cover cannot be distinguished by local computations. Later, Litovsky et al. [134] proved that planar graphs and series-parallel graphs cannot be recognized by local computations, and Courcelle and Metivier [57] showed that in fact no nontrivial minor-closed class of graphs can. In both of these results, graph coverings were used as the main tool, as well as in more recent papers of Chalopin et al. [47, 48]. Here, the authors presented a model for distributed computations and addressed the algorithmic complexity of problems associated with such a model. To this end, they used the existing results on NP-completeness of the covering problem to provide their hardness results. In [49], the authors study a close relation of packing bipartite graphs to a special variant of graph coverings called *pseudo-coverings.*

Another connection to algorithmic theory comes through the notions of the *degree partition* and the *degree refinement matrix* of a graph. These notions were introduced by Corneill [54, 55] in hope of solving the graph isomorphism problem efficiently. It can be easily seen that a graph and all of its covers have the same degree refinement matrix. Motivated by this observation, Angluin and Gardiner [8] proved that any two finite regular graphs of the same valency have a finite common cover, and conjectured the same for every two finite graphs with the same degree refinement matrix, which was proved by Leighton [131].

The stress on finiteness of the common cover is natural. For every matrix, there exists a universal cover, an infinite tree, that covers all graphs with this degree refinement matrix. Trees are planar graphs, and this inspired naturally a question of which graphs allow a finite planar cover. Negami observed that projective planar graphs do (in fact, their double planar covers characterize their projective embedding), and conjectured that these two classes actually coincide [150]. Despite a serious effort of numerous authors, the problem is still open, although the scope for possible failure of Negami's conjecture has been significantly reduced [9, 107, 108].

A natural computational complexity question is how difficult is to decide, given two graphs, if one covers the other one. This question is obviously at least as difficult as the graph isomorphism problem (consider two given graphs on the same number of vertices). Its NP-completeness was proved by Bodlaender [19] (in the case of both graphs being part of the input).

To the best of our knowledge, Abello et al. [1] were the first to ask about the computational complexity of the $H$-Cover problem for a fixed target graph $H$.

| | |
|---|---|
| Problem: | $H$-Cover |
| Input: | A graph $G$. |
| Question: | Does $G$ cover $H$? |

Abello et al. [1] showed that deciding if an input graph covers the dumbbell graph (edge with one loop on both endpoints) is NP-complete. (For future reference, it is important to note that they allowed the input graph to contain loops.) Furthermore, they asked for a complete characterization of the computational complexity, depending on the parameter graphs $H$. Such a line of research was picked by Kratochvíl, Proskurowski, and Telle, who completely characterized the complexity for simple target graphs with at most 6 vertices [126], and then noted that in order to fully characterize the complexity of the $H$-cover problem for simple target graphs, it is sufficient (but also necessary) to classify it for mixed coloured multigraphs with minimum degree at least three [123]. The latter result gives a hope for a more concise description of the characterization, but is also in line with the original motivation of covers from topological graph theory, where loops and multiedges are widely considered.

The complexity of covering 2-vertex multigraphs was fully characterized in [123], the characterization for 3-vertex undirected multigraphs can be found in [127]. The most general NP-hardness result known so far is the hardness of covering simple regular graphs of valency at least three [125, 73]. (Note that in order to map a vertex of $G$ to a vertex of $H$, they must be of the same degree, and thus regular targets $H$ are a natural and interesting special case.)

Let us point out that in all the above results it was assumed that $H$ has no multiple edges.

More recently, Bílka et al. [17] proved that covering several concrete small graphs (including the complete graphs $K_4, K_5$ and $K_6$) remains NP-hard for planar inputs. This shows that planarity does not help in graph covering problems in general, yet the conjecture that the $H$-Cover problem restricted to planar inputs is at least as difficult as for general inputs, provided $H$ itself has a finite planar cover, remains still open. Planar graphs have also been considered by Fiala et al. [77] who showed that for planar input graphs, $H$-RegularCover is in FPT when parameterized by $H$. This is in fact the first and only paper on the complexity of regular covers, i.e., covering projections determined by a regular action of a group of automorphisms on the covering graph.

**Locally constrained homomorphisms.** Graph coverings were also extensively studied under a unifying umbrella of *locally constrained homomorphisms.* In these relaxations, homomorphisms can be either locally injective or locally surjective and not necessarily locally bijective. In other words, we ask for the existence of a homomorphism that is, respectively, surjective or injective in the neighbourhood of each vertex.

The complexity of finding locally constrained homomorphisms was studied by many authors. For locally surjective homomorphisms we know a complete dichotomy [84]. The problem is polynomial-time solvable if the target graph $H$ either (a) has no edge, or (b) has a component that consists of a single vertex with a loop, or (c) is simple and bipartite, with at least one component isomorphic to $K_2$. In all other cases the problem is NP-complete.

The dichotomy for locally injective homomorphisms is still unknown, despite some work [80, 81]. However, we understand the complexity of the *list variant* of the problem [82]: it is polynomial-time solvable if every component of the target graph has at most one cycle, and NP-complete otherwise.

Locally surjective homomorphisms play an important role in social sciences [84, 161], where they study the following problem, called Role Assignment.

| | |
|---|---|
| Problem: | Role Assignment |
| Input: | A connected graph $G$ and an integer $h$. |
| Question: | Does there exist a connected graph $H$ of size $h$ (with possible loops) such that there is a locally surjective homomorphism from $G$ to $H$? |

Locally injective homomorphism have an interesting applied motivation as well. A locally injective homomorphism into the complement of a path of length $k$ corresponds to an $L(2,1)$-labelling of span $k$ [92], an intensively studied notion stemming from the theory of frequency assignment. On the other hand, generalizations include the notion of $H(p,q)$-colouring, a homomorphism into a fixed target graph $H$ with additional rules on the neighbourhoods of the vertices [76, 128].

Recall further that there is also some more work concerning the complexity of locally surjective and injective homomorphisms if $G$ is assumed to come from some special class [14, 17, 50, 77, 152]. We also refer the reader to the survey concerning various aspects of locally constrained homomorphisms [83] and to the references in [141] (for injective variant). For every fixed graph $H$, the existence of a locally injective homomorphism to $H$ is provably at least as hard as the $H$-cover problem. In this sense our hardness results extend the state of the art also for the problem of existence of locally injective homomorphisms.

## 5.2 Graphs with semi-edges

Informally, semi-edges have, compared to the usual edges in graph theory, only one endpoint. The notion of *semi-edges* has been introduced in the modern topological graph theory and it is becoming more and more frequently used (the terminology has not yet stabilized; semi-edges are often called half-edges, edges with free ends, and sometimes fins). Mednykh and Nedela wrote a monograph [145] in which they summarize and survey the ambitions and efforts behind generalizing the notion of graph coverings to the graphs with semi-edges. This generalization, as the authors pinpoint, is not artificial as such graphs emerge "in the situation of taking quotients of simple graphs by groups of automorphisms which are semiregular on vertices and darts (arcs) and which may fix edges". As the authors put it: "A problem arises when one wants to consider quotients of such graphs (graphs embedded to surfaces) by an involution fixing an edge $e$ but transposing the two incident vertices. The edge $e$ is halved and mapped to a semi-edge — an edge with one free end." This direction of research proved to be very fruitful and provided many applications and generalizations to various parts of algebraic graph theory. For example, Malnič et al. [143] considered semi-edges during their study of abelian covers and as they write "...in order to have a broader range of applications we allow graphs to have semi-edges." To highlight a few other contributions, the reader is invited to consult [149, 144], the surveys [129] (containing a very nice introductory section on applications of coverings) and (aforementioned) [145]. Finally, for more recent results, we point to the series of papers [77, 78, 79]. It is also worth noting that the concept of graphs with semi-edges was introduced independently and naturally in mathematical physics by Getzler and Karpanov [89].

In the view of the theory of local computations, semi-edges and their covers prove very natural, too, and it is even surprising that they have not been considered before in the context. If a computer network is constructed as a cover of a small template, the preimages of normal edges in the covering projection are matchings completely connecting nodes of two types (the end-vertices of the covered edge). Preimages of loops are disjoint cycles with nodes of the same type. And preimages of semi-edges are matchings on vertices of the same type. The role of semi-edges was spotted by Woodhouse et al. [160, 164] who have generalized the fundamental theorem of Leighton [131] on finite common covers of graphs with the same degree refinement matrix to graphs with semi-edges.

## 5.3 Formal definitions

In this section we formally define what we call *graphs*. A graph has a set of vertices and a set of edges (also referred to as links). As it is standard in topological graph theory, we automatically allow multiple edges and loops. Every ordinary edge is connecting two vertices, every loop is incident with only one vertex. On top of these, we also allow *semi-edges*. Each semi-edge is also incident with only one vertex. The difference between loops and semi-edges is that a loop contributes two to the degree of its vertex, while a semi-edge only one.

A very elegant description of ordinary edges, loops and semi-edges through the concept of *darts* is used in more algebraic-based papers on covers. The following formal definition is a reformulation of the one given in [145].

**Definition 5.1.** *A* graph *is a triple* $(D, V, \Lambda)$*, where* $D$ *is a set of* darts*, and* $V$ *and* $\Lambda$ *are each a partition of* $D$ *into disjoint sets. Moreover, all sets in* $\Lambda$ *have size one or two.*

With this definition, the *vertices* of a graph $(D, V, \Lambda)$ are the sets of $V$ (note that empty sets correspond to isolated vertices, and since we are interested in covers of connected graphs by connected ones, we assume that all sets of $V$ are non-empty). The sets of $\Lambda$ are referred to as *links*, and they are of three types: *loops* (2-element sets with both darts from the same set of $V$), *(ordinary) edges* (2-element sets intersecting two different sets of $V$), and *semi-edges* (1-element sets). After this explanation it should be clear that this definition is equivalent

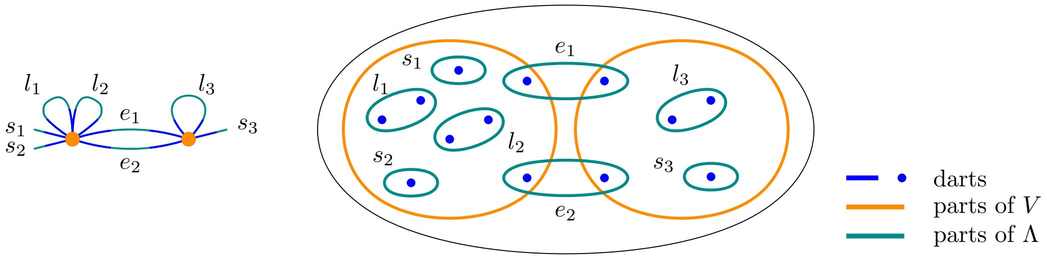


Figure 5.1: An example of a graph presented in a usual graph-theoretical way (left) and using the dart-based Definition 5.1 (right).

to a definition of multigraphs which is standard in the graph theory community.[1]

**Definition 5.2.** *A* graph *is an ordered triple* $(V, \Lambda, \iota)$, *for* $\Lambda = E \cup L \cup S$, *where* $\iota$ *is the* incidence mapping $\iota : \Lambda \longrightarrow V \cup \binom{V}{2}$ *such that* $\iota(e) \in V$ *for all* $e \in L \cup S$ *and* $\iota(e) \in \binom{V}{2}$ *for all* $s \in E$.

For a further comparison of Definitions 5.1 and 5.2, see Figure 5.1. Since we consider multiple edges of the same type incident with the same vertex (or with the same pair of vertices), the edges are given by their names and the incidence mapping $\iota$ expresses which vertex (or vertices) 'belong' to a particular edge. The degree of a vertex is then defined as follows.

**Definition 5.3.** *For a graph* $G = (V, \Lambda = E \cup L \cup S, \iota)$, *the degree of a vertex* $u \in V$ *is defined as*

$$deg_G(u) = p_S(u) + p_E(u) + 2p_L(u),$$

*where* $p_S(u)$ $(p_L(u))$ *is the number of semi-edges* $e \in S$ *(of loops* $e \in L$*) such that* $\iota(e) = u$, *and* $p_E(u)$ *is the number of ordinary edges* $e \in E$ *such that* $u \in \iota(e)$.

We call a graph $G$ *simple* if $p_S(u) = p_L(u) = 0$ for every vertex $u \in V(G)$ (the graph has no loops or semi-edges) and $\iota(e) \neq \iota(e')$ for every two distinct $e, e' \in E$ (the graph has no multiple (ordinary) edges). We call $G$ *semi-simple* if $p_S(u) \leq 1$ and $p_L(u) = 0$ for every vertex $u \in V(G)$ and $\iota(e) \neq \iota(e')$ for every two distinct $e, e' \in E$.

Note that in the language of Definition 5.1, the degree of a vertex $v \in V$ is simply $|v|$. We say that a graph is *regular* if all its vertices have the same degree and we say it is *$k$-regular* if all its vertices have the same degree $k$.

[1] More formally, to see that both approaches are equivalent we show how the dart representation of a graph can be converted into the incidence one, and vice-versa. First, given $G = (D, V, \Lambda)$, we define $\iota(e) = \{v : e \cap v \neq \emptyset\}$. For the reverse transformation, given $G = (V, E \cup L \cup S, \iota)$, we define the set of darts as $D = \{(\iota(e), e) : e \in S \cup L\} \cup \{(v, e) : v \in \iota(e), e \in E\}$, and then the partition $V$ is given by the equivalence relation $\sim_V$: $(v_1, e_1) \sim_V (v_2, e_2)$ if $v_1 = v_2$, and the partition $\Lambda$ by $\sim_\Lambda$: $(v_1, e_1) \sim_\Lambda (v_2, e_2)$ if $e_1 = e_2$.

In this language, the main object of our study, a *graph cover* (or equivalently a *covering projection*), is defined as follows.

**Definition 5.4.** *We say that a graph $G = (D_G, V_G, \Lambda_G)$* covers *a connected graph $H = (D_H, V_H, \Lambda_H)$ (denoted as $G \longrightarrow H$) if there exists a map $f\colon D_G \to D_H$ such that:*

- *The map $f$ is surjective.*
- *For every $u \in V_G$, there is a $u' \in V_H$ such that the restriction of $f$ onto $u$ is a bijection between $u$ and $u'$.*
- *For every $e \in \Lambda_G$, there is an $e' \in \Lambda_H$ such that $f(e) = e'$.*

*The map $f$ is called* graph cover *(or* covering projection*).*

Compare the compactness of this definition with Proposition 5.5, which is the definition of (multi)graph covering in the standard language of Definition 5.2.

The fact that a loop contributes 2 to the degree of its vertex may seem not obvious at first sight, but becomes natural when graphs are considered embedded to surfaces, and is absolutely obvious when we look at the definition of a covering projection (for the sake of exactness, the definition is somewhat technical).

**Proposition 5.5.** *A graph $G$ covers a graph $H$ if and only if $G$ allows a pair of mappings $f_V : V(G) \longrightarrow V(H)$ and $f_\Lambda : \Lambda(G) \longrightarrow \Lambda(H)$ such that*

1. *$f_\Lambda(e) \in L(H)$ for every $e \in L(G)$ and $f_\Lambda(e) \in S(H)$ for every $e \in S(G)$,*
2. *$\iota(f_\Lambda(e)) = f_V(\iota(e))$ for every $e \in L(G) \cup S(G)$,*
3. *for every link $e \in \Lambda(G)$ such that $f_\Lambda(e) \in S(H) \cup L(H)$ and $\iota(e) = \{u, v\}$, we have $\iota(f_\Lambda(e)) = f_V(u) = f_V(v)$,*
4. *for every link $e \in \Lambda(G)$ such that $f_\Lambda(e) \in E(H)$ and $\iota(e) = \{u, v\}$ (note that it must be $f_V(u) \neq f_V(v)$), we have $\iota(f_\Lambda(e)) = \{f_V(u), f_V(v)\}$,*
5. *for every loop $e \in L(H)$, $f^{-1}(e)$ is a disjoint union of loops and cycles spanning all vertices $u \in V(G)$ such that $f_V(u) = \iota(e)$,*
6. *for every semi-edge $e \in S(H)$, $f^{-1}(e)$ is a disjoint union of edges and semi-edges spanning all vertices $u \in V(G)$ such that $f_V(u) = \iota(e)$, and*
7. *for every edge $e \in E(H)$, $f^{-1}(e)$ is a disjoint union of edges (i.e., a matching) spanning all vertices $u \in V(G)$ such that $f_V(u) \in \iota(e)$.*

See an example of a covering projection in Figure 5.2. Conditions 1–4 express the fact that $f_V$ and $f_E$ commute with $\iota$, i.e., that $f$ is a homomorphism from $G$ to $H$. Conditions 5–7 express that this homomorphism is locally bijective:

- for every ordinary edge $e$ incident with $f_V(u)$ in $H$, there is exactly one ordinary edge of $G$ which is incident with $u$ and mapped to $e$ by $f_E$;

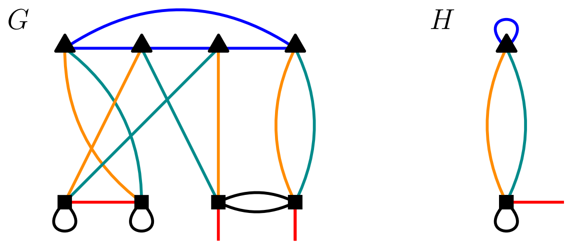


Figure 5.2: An example of a covering. The vertex mapping of the covering from $G$ to $H$ is determined by the shape of the vertices, the edge mapping by the colours of the edges.

- for every semi-edge $e$ incident to $f_V(u)$ in $H$, there is exactly one semi-edge, or exactly one ordinary edge (but not both) in $G$ incident with $u$ and mapped to $e$ by $f_E$;
- and for every loop $e$ incident with $f_V(u)$ in $H$, there is exactly one loop or exactly two ordinary edges (but not both) of $G$ which are incident with $u$ and mapped to $e$ by $f_E$.

Even though the aforementioned definitions of graphs and graph covers through darts are neat and compact, we shall often work with the standard definition of graphs and the equivalent description of graph covers given by Proposition 5.5. The reason is that they are better suited for describing our reductions and understanding the illustrative figures.

It is clear that a covering projection (more precisely, its vertex mapping) preserves degrees. One may ask when (or if) a degree preserving vertex mapping can be extended to a covering projection. An obvious necessary condition is described by the following definition.

**Definition 5.6.** *A vertex mapping $f_V : V(G) \longrightarrow V(H)$ between graphs $G$ and $H$ is called* degree-obedient *if*

1. *for any two distinct vertices $u, v \in V(H)$ and any vertex $x \in f_V^{-1}(u)$, the number of ordinary edges $e$ of $H$ such that $\iota(e) = \{u, v\}$ equals the number of ordinary edges of $G$ with one end-vertex $x$ and the other one in $f_V^{-1}(v)$, and*
2. *for every vertex $u \in V(H)$ and any vertex $x \in f_V^{-1}(u)$, the value $p_{S(H)}(u) + 2p_{L(H)}(u)$ equals $p_{S(G)}(x) + 2p_{L(G)}(x) + r$, where $r$ is the number of edges of $G$ with one end-vertex $x$ and the other one from $f_V^{-1}(u) \setminus \{x\}$,*
3. *for every vertex $u \in V(H)$ and any vertex $x \in f_V^{-1}(u)$, $p_{S(G)}(x) \le p_{S(H)}(u)$.*

We conclude the introductory chapter with shorthand definition for one- and two-vertex graphs, a quick reminder of Cartesian product of graphs, and with an explanation of the term *garbage collection.*

**One-vertex and two-vertex graphs.** It shall be useful for our purposes to specifically denote one-vertex and two-vertex graphs. Let us denote by $F(b, c)$ the one-vertex graph with $b$ semi-edges and $c$ loops and by $W(k, m, \ell, p, q)$ the two-vertex graph with $k$ semi-edges and $m$ loops at one vertex, $p$ loops and $q$ semi-edges at the other one, and $\ell > 0$ multiple edges connecting the two vertices (these edges are referred to as *bars*). In other words, $W(k, m, \ell, p, q)$ is obtained from the disjoint union of $F(k, m)$ and $F(q, p)$ by connecting their vertices by $\ell$ parallel edges. Note that the graph in Figure 5.1 is in fact $W(2, 2, 2, 1, 1)$ and, as an additional example, see the graph $H$ from Figure 5.2 which is isomorphic to both $W(1, 1, 2, 1, 0)$ and $W(0, 1, 2, 1, 1)$.

**Product.** Let us recall that the *product* $G \times H$ of graphs $G$ and $H$ is defined as the graph with the vertex set being the Cartesian product $V(G) \times V(H)$ and with vertices $(u, v)$ and $(u', v')$ being adjacent in $G \times H$ if and only if $u$ is adjacent to $u'$, and $v$ is adjacent to $v'$.

**Garbage collection.** We use the term *garbage collection* throughout this part of the thesis. Loosely speaking, it denotes gadgets or parts of constructions inserted into reductions to ensure that the low-degree vertices of edge or vertex gadgets have the “right degree”.

# Chapter 6

# Complexity of covering small multigraphs with semi-edges

This chapter is based on:

- [27] Jan Bok, Jiří Fiala, Petr Hliněný, Nikola Jedličková, and Jan Kratochvíl: *Computational Complexity of Covering Multigraphs with Semi-Edges: Small Cases.* In 46th International Symposium on Mathematical Foundations of Computer Science, MFCS 2021, volume 202 of Leibniz International Proceedings in Informatics (LIPIcs), pages 21:1–21:15, 2021.

We initiate the study of computational complexity of graph coverings, also know as locally bijective graph homomorphisms, for *graphs with semi-edges.*

In 1991, Abello et al. [1] asked for a classification of the computational complexity of deciding if an input graph covers a fixed target graph, in the ordinary setting of graphs without semi-edges. Although many general results are known, the full classification is still open. In spite of that, we propose to study the more general case of covering graphs composed of edges (including multiedges and loops) and semi-edges.

We show that the presence of semi-edges makes the covering problem considerably harder; e.g., it is no longer sufficient to specify the vertex mapping induced by the covering, but one necessarily has to deal with the edge mapping as well. We show some solvable cases and, in particular, completely characterize the complexity of the already very nontrivial problem of covering one- and two-vertex (multi)graphs with semi-edges. Our NP-hardness results are proven for simple input graphs, and in the case of regular two-vertex target graphs, even for bipartite ones.

We remark that our new characterization results also strengthen previously known results for covering graphs without semi-edges, and they in turn apply to an infinite class of simple target graphs with at most two vertices of degree more than two. Some of the results are moreover proven in a more general setting (for example, finding $k$-tuples of pairwise disjoint perfect matchings in regular graphs, or finding equitable partitions of regular bipartite graphs).

## 6.1 Overview of our results

In Section 6.2 we show major differences between covering graphs with and without semi-edges. We further provide an evidence that the situation is somewhat clearer if the source graph is bipartite. In Theorem 6.4 we prove that if the source graph is bipartite and has no semi-edges, then every degree-obedient vertex mapping can be extended to a covering, while if semi-edges are allowed in the bipartite source graph, it can be at least decided in polynomial time if a degree-obedient mapping is extendable to a covering.

In order to present our results in the strongest possible form, we aim at proving the hardness results for restricted classes of input graphs, while the polynomial ones for the most general inputs. In particular, we only allow simple graphs as inputs when we prove NP-hardness, and on the other hand, we allow loops, multiple edges as well as semi-edges when we present polynomial-time algorithms.

The first NP-hardness result is proven in Theorem 6.5, namely that covering semi-simple regular graphs of valency at least 3 is NP-hard even for simple bipartite input graphs. In Sections 6.3 and 6.4 we give a complete classification of the computational complexity of covering graphs with one and two vertices. This extends the main result of [123] to graphs with semi-edges. Moreover, we strengthen the hardness results of [123] considerably by showing that all NP-hard cases of covering regular two-vertex graphs (even those without semi-edges) remain NP-hard for simple *bipartite* input graphs. Note that through the reduction from [124], our results on the complexity of covering one- or two-vertex graphs provide characterization results on infinitely many simple graphs which contain at most two vertices of degrees greater than 2.

All considered computational problems are clearly in the class NP, and thus we only concentrate on the NP-hardness proofs in the NP-completeness results. We restrict our attention to connected target graphs, in which case it suffices to consider only connected input graphs. In this case every cover is a $k$-fold cover for some $k$, which means that the preimage of every vertex has the same size. The treatment of the case in which the target is a disconnected graph is provided in Chapter 7.

## 6.2 The impact of semi-edges

First, we discuss the necessity of specifying the edge mapping in a covering projection — a major difference between covering graphs with and without semi-edges. In other words, we discuss when a degree mapping can always be extended to a covering, and when this question can be decided efficiently. The following proposition follows straightforwardly from the definitions of Chapter 5.

**Proposition 6.1.** *For every graph covering projection between two graphs, the vertex mapping induced by this projection is degree-obedient.*

**Proposition 6.2.** *If $H$ has no semi-edges, then for any graph $G$, any degree-obedient mapping from the vertex set of $G$ onto the vertex set of $H$ can be extended to a graph covering projection of $G$ to $H$.*

*Proof.* For simple graphs $G$, this is proved already in [123]. If multiple edges and loops are allowed, we use a similar approach. The key point is that Petersen's theorem [153] about 2-factorization of regular graphs of even valence is true for multigraphs without semi-edges as well, and the same holds true for König-Hall theorem [136] on 1-factorization of regular bipartite multigraphs. □

As we will see soon, the presence of semi-edges changes the situation a lot. Even for simple graphs, degree-obedient vertex mappings to a graph with semi-edges may not extend to a graph covering projection, and the possibility of such an extension may even be NP-complete.

**Observation 6.3.** *Let $F(3,0)$ be the graph with one vertex and three semi-edges pending on this vertex. Then a simple graph covers $F(3,0)$ if and only if it is 3-regular and 3-edge-colourable. Testing 3-edge-colourability is well known to be NP-hard for simple graphs.*

However, if the input graph is bipartite, the situation gets much easier.

**Theorem 6.4.** *If a graph $G$ is bipartite, then for any graph $H$, it can be decided in polynomial time whether a degree-obedient mapping from the vertex set of $G$ onto the vertex set of $H$ can be extended to a graph covering projection of $G$ to $H$. In particular, if $G$ has no semi-edges and is bipartite, then every degree-obedient mapping from the vertex set of $G$ onto the vertex set of $H$ can be extended to a graph covering projection of $G$ to $H$.*

*Proof.* Let $G$ be a bipartite graph and let $f_V : V(G) \longrightarrow V(H)$ be a degree-obedient mapping from the vertex set of $G$ to a vertex set of $H$. We seek an edge mapping $f_E : E(G) \longrightarrow E(H)$ such that $(f_V, f_E)$ is a covering projection of $G$ to $H$. For every edge or semi-edge $s$ of $G$, its image under $f_E$ is restricted to be chosen from edges with corresponding end-vertices: if $s$ is a semi-edge on vertex

$u$, $f_E(s)$ must be a semi-edge on $f_V(u)$, and if $s$ is an edge with end-vertices $u$ and $v$ (a loop, when $u = v$), $f_E(s)$ must be an edge with end-vertices $f_V(u)$ and $f_V(v)$ (a loop or a semi-edge, if $f_V(u) = f_V(v)$ ).

Consider two distinct vertices $x, y \in V(H)$, and let them be connected by $k$ edges $e_1, e_2, \dots, e_k$ in $H$. The bipartite subgraph $\widetilde{G_{x,y}}$ of $G$ with classes of bipartition $f_V^{-1}(x)$ and $f_V^{-1}(y)$ and edges of $G$ with end-points in different classes is $k$-regular. By König-Hall theorem, it is $k$-edge colourable. If $\varphi : E(\widetilde{G_{x,y}}) \longrightarrow \{1, 2, \dots, k\}$ is such a colouring, then $f_E : E(\widetilde{G_{x,y}}) \longrightarrow \{e_1, e_2, \dots, e_k\}$ defined by $f_E(h) = e_{\varphi(h)}$ is a covering projection onto the set of parallel edges between $x$ and $y$ in $H$.

The situation is more complex for loops and semi-edges of $H$. Consider a vertex $x \in V(H)$ and the subgraph $\widetilde{G_x}$ of $G$ induced by $f_V^{-1}(x)$. If $x$ has $b$ semi-edges and $c$ loops in $H$, $\widetilde{G_x}$ is $(b + 2c)$-regular. Let $s(u)$ be the number of semi-edges of $G$ incident with $u$, and set $g(u) = b - s(u)$. In a covering projection, for every $u \in f_V^{-1}(x)$, exactly $g(u)$ of edges incident with $u$ must map onto semi-edges of $H$ incident with $x$. Hence a covering projection on the edges of $\widetilde{G_x}$ exists only if $\widetilde{G_x}$ has a $g$-factor for the above defined function $g$. This can be decided in polynomial time (e.g., by network flow algorithms, since $\widetilde{G_x}$ is a bipartite graph, but even for general graphs the existence of a $g$-factor can be reduced to the maximum matching problem). If such a $g$-factor exists, it is $b$-edge-colourable (here and only here we use the assumption that $G$ is bipartite), and such an edge-colouring defines a mapping $f_E$ from the edges of the $g$-factor onto the semi-edges of $H$ incident with $x$. For every vertex $u \in f_V^{-1}(x)$, $g(u)$ edges of $G$ incident with $u$ are mapped onto $g(u)$ distinct semi-edges incident with $x$ in $H$, and $b - g(u) = s(u)$ semi-edges remain available as images of the $s(u)$ semi-edges incident with $u$ in $G$. What remains is to define $f_E$ for the so far unmapped edges of $\widetilde{G_x}$. But these form a $2c$-regular graph which covers $c$ loops on $x$ in $H$ (a consequence of Petersen's theorem, or König-Hall theorem since $G$ is bipartite and hence the edges of a $2c$-regular bipartite graph can be partitioned into $2c$ perfect matchings and these matchings can be paired into $c$ disjoint spanning cycles, each covering one loop).

If $\widetilde{G_x}$ has no semi-edges, then it is bipartite $(b+2c)$-regular and as such it always has a $b$-factor. Hence for a bipartite graph without semi-edges, a degree-obedient vertex mapping can always be extended to a graph covering projection. □

Now we prove that covering semi-simple regular graphs is always NP-complete (this is the case when every vertex of the target graph is incident with at most one semi-edge, and the graph has no multiple edges nor loops). See Figure 6.1 for examples of semi-simple graphs $H$ defining such hard cases.

**Theorem 6.5.** *Let $H$ be a semi-simple $k$-regular graph, with $k \geq 3$. Then the $H$-*Cover *problem is NP-complete even for simple bipartite input graphs.*

*Proof.* Consider $H' = H \times K_2$. This graph is simple, $k$-regular and bipartite,

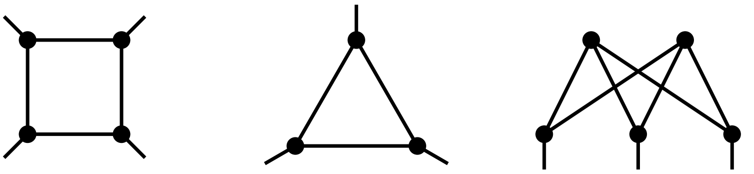


Figure 6.1: Examples of small semi-simple graphs which define NP-complete covering problems.

hence the $H'$-Cover problem is NP-complete by [125]. Given an input $k$-regular graph $G$, it is easy to see that $G$ covers $H'$ if and only it is bipartite and covers $H$. Since bipartiteness can be checked in polynomial time, the claim follows. □

## 6.3 One-vertex target graphs

We start the section by proving a slightly more general hardness result, which may be of interest on its own. In particular, it implies that for every $d \geq 3$, it is NP-complete to decide if a simple $d$-regular graph contains an even 2-factor, i.e., a spanning 2-regular subgraph whose every cycle has even length.

**Theorem 6.6.** *For every $k \geq 2$ and every $d \geq k+1$, it is NP-complete to decide if a simple $d$-regular graph contains $k$ pairwise disjoint perfect matchings.*

*Proof.* The complement of the union of $k$ pairwise disjoint perfect matchings in a $(k+1)$-regular graph is a perfect matching as well, and thus a $(k+1)$-regular graph contains $k$ pairwise disjoint perfect matchings if and only if it is $(k+1)$-edge colourable. Hence for $d = k+1$, the claim follows from the NP-completeness of $d$-edge colourability of $d$-regular graphs which has been proven by Leven and Galil [132].

Let $d \geq k+2$. We prove the claim by a reduction from $(k+1)$-edge colourability of $(k+1)$-regular graphs (using [132] again). Fix a graph $H$ with one vertex, say $x$, of degree $d-2$ and all other vertices having degrees $d$, and such that $H$ contains $d-2$ pairwise disjoint perfect matchings (such a graph can be easily constructed, see the end of the proof). Given a $(k+1)$-regular graph $G$ whose $(k+1)$-edge colourability is questioned, we construct a graph $G'$ as follows: The graph $G'$ contains two disjoint copies $G_1, G_2$ of $G$ such that the two clones of each vertex $u$ of $G$ in $G_1$ and $G_2$ are connected together by $d-k-1$ paths of lengths 2. Moreover, the middle vertices in each of those paths play the role of the vertex $x$ in a copy of $H$ (each copy of $H$ is private to its path). See Figure 6.2. Formally,

$$V(G') = V(G_1) \cup V(G_2) \cup \bigcup_{u \in V(G)} \bigcup_{i=1}^{d-k-1} V(H_{u,i}), \text{ and}$$

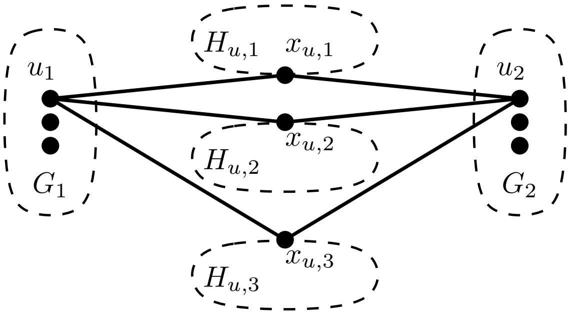


Figure 6.2: An illustration for the construction of the graph $G'$ in the proof of Theorem 6.6.

$$E(G') = E(G_1) \cup E(G_2) \cup \bigcup_{u \in V(G)} \bigcup_{i=1}^{d-k-1} (E(H_{u,i}) \cup \{u_1 x_{u,i}, u_2 x_{u,i}\},$$

where

$$V(G_j) = \{u_j : u \in V(G)\} \text{ and } E(G_j) = \{u_j v_j : uv \in E(G)\}$$

for $j = 1, 2$, and

$$V(H_{u,i}) = \{y_{u,i} : y \in V(H)\} \text{ and } E(H_{u,i}) = \{y_{u,i} z_{u,i} : yz \in E(H)\}$$

for $u \in V(G)$ and $i = 1, 2, \ldots, d-k-1$.

We claim that $G'$ has $k$ pairwise disjoint perfect matchings if and only if $\chi'(G) = k+1$ (where $\chi'(G)$ denotes the *chromatic index of graph $G$*, i.e. the smallest number of colours needed to colour the edges). In one direction, if $G$ is $k$-edge colourable, then for each $j = 1, 2$, the graph $G_j$ has $k$ pairwise disjoint perfect matchings, say $M_h^j, h = 1, 2, \ldots, k$. By the assumption on $H$, each $H_{u,i}$ has $k \le d-2$ pairwise disjoint matchings, say $M_h^{u,i}, h = 1, 2, \ldots, k$, for all $u \in V(G)$ and $i = 1, 2, \ldots, d-k-1$. Then

$$M_h = M_h^1 \cup M_h^2 \cup \bigcup_{u \in V(G)} \bigcup_{i=1}^{d-k-1} M_h^{u,i},$$

for $h = 1, 2, \ldots, k$, are $k$ pairwise disjoint perfect matchings in $G'$.

For the opposite implication, note that no perfect matching of $G'$ contains any of the edges $u_j x_{u,i}$, $u \in V(G), i = 1, 2, \ldots, d-k-1, j = 1, 2$, because each $H_{u,i}$ has an even number of vertices and each $x_{u,i}$ is an articulation in $G'$. So, for every perfect matching $M$ in $G'$, $M \cap E(G_1)$ is a perfect matching in $G$. Thus if $M_h, h = 1, 2, \ldots, k$ are pairwise disjoint perfect matchings in $G'$, then $\{uv \in E(G) : u_1 v_1 \in M_h\}, h = 1, 2, \ldots, k$ are $k$ pairwise disjoint perfect matchings in $G$, and hence $\chi'(G) = k+1$.

To complete the proof, let us show an explicit construction of the auxiliary graph $H$. Fix an odd number $t \ge d+1$. It is well known that the complete graph $K_{t+1}$

is $t$-edge colourable, i.e., its edge set $E(K_{t+1})$ can be partitioned into $t$ perfect matchings, say $M_1, M_2, \ldots, M_t$. Choose vertices $x, y, z$ so that $xy \in M_1, xz \in M_2$, and assume without loss of generality that $yz \in M_t$. Define the graph $H$ as follows:

$$V(H) = V(K_{t+1}),$$

$$E(H) = (\bigcup_{i=1}^{d} M_i \setminus \{xy, xz\}) \cup \{yz\}.$$

Then $\deg_H x = d - 2$ and $\deg_H u = d$ for all $u \in V(H) \setminus \{x\}$. Moreover, $H$ has $d - 2$ pairwise disjoint perfect matchings $M_3, M_4, \ldots, M_d$. □

Now we are ready to prove a dichotomy theorem on the complexity of covering one-vertex graphs. Let us remind that $F(b, c)$ is the one-vertex graph with $b$ semi-edges and $c$ loops.

Before proving the next theorem, let us pinpoint that matchings here can consist of both edges and semi-edges. The generalisation is straightforward, putting semi-edges into matchings can be done by a simple preprocessing, and the notion of perfectness naturally applies to such “generalised matchings”.

**Theorem 6.7.** *The $F(b, c)$-*Cover *problem is polynomial-time solvable if $b \leq 1$, or $b = 2$ and $c = 0$, and it is NP-complete otherwise, even for simple graphs.*

*Proof.* In every case, the input graph $G$ has to be $(b+2c)$-regular, since otherwise it cannot cover $F(b, c)$. This condition can be checked in polynomial time. Next observe that a $(b + 2c)$-regular graph $G$ covers $F(b, c)$ if and only if it contains $b$ pairwise disjoint perfect matchings whose removal leaves us with a $2c$-regular graph without semi-edges. Indeed, these matchings are the preimages of the $b$ semi-edges in a covering projection. The remaining $2c$-regular graph without semi-edges can be always partitioned into $c$ pairwise disjoint 2-factors by the well known Petersen's theorem [136, 153], and each of the 2-factors will cover one of the $c$ loops of $F(b, c)$. Note that a possible presence of loops in the input graph does not cause any problems.

The polynomially solvable cases then follow easily:

- If $b = 0$, the checking is trivial.
- If $b = 1$, the existence of a perfect matching can be checked in polynomial time, for instance by a simple adaptation of Edmonds' blossom algorithm [60]. The only thing we have to be careful about is that any possible semi-edges in the input graph have to be added to the perfect matching. This can be done by a simple preprocessing.
- If $b = 2$ and $c = 0$, $G$ itself has to be a 2-regular graph and hence it contains two disjoint perfect matchings if and only if it contains at least one perfect matching, i.e., when all connected components of $G$ are even.

The NP-complete cases follow from Theorem 6.6 by setting $k = b$ and $d = b + 2c$. □

## 6.4 Two-vertex target graphs

Recall that $W(k, m, \ell, p, q)$ is the two-vertex graph with $k$ semi-edges and $m$ loops at one vertex, $p$ loops and $q$ semi-edges at the other one, and $\ell > 0$ multiple edges connecting the two vertices.

We can now state the full dichotomy for two-vertex targets.

**Theorem 6.8.** *The $W(k, m, \ell, p, q)$-*Cover *problem is solvable in polynomial time in the following cases:*

1. *$k + 2m \neq 2p + q$ and ($k \leq 1$ or $k = 2$ and $m = 0$) and ($q \leq 1$ or $q = 2$ and $p = 0$),*
2. *$k + 2m = 2p + q$ and $\ell = 1$ and $k = q \leq 1$ and $m = p = 0$,*
3. *$k + 2m = 2p + q$ and $\ell > 1$ and $k = m = p = q = 0$,*

*and it is NP-complete otherwise.*

Note that Case 1 applies to non-regular target graph $W$, while Cases 2 and 3 apply to regular graphs $W$, i.e., they cover all cases when $k+2m+\ell = 2p+q+\ell$.

We will refer to the vertex with $k$ semi-edges as *blue* and the vertex with $q$ semi-edges as *red.* In a covering projection $f = (f_V, f_E)$ from a graph $G$ onto $W(k, m, \ell, p, q)$, we view the restricted vertex mapping $f_V$ as a colouring of $V(G)$. We call a vertex $u \in V(G)$ blue (red) if $f_V$ maps $u$ onto the blue (red, respectively) vertex of $W(k, m, \ell, p, q)$.

We divide the proof of Theorem 6.8 into a sequence of claims, proved in the following subsections. This allows us to state several hardness results in a stronger form.

### 6.4.1 Polynomial parts of Theorem 6.8

We follow the case-distinction from the statement of Theorem 6.8:

1. If $k+2m \neq 2p+q$, then the two vertex degrees of $W(k, m, \ell, p, q)$ are different, and the vertex restricted mapping is uniquely defined for any possible graph covering projection from the input graph $G$ to $W(k, m, \ell, p, q)$. For this colouring of $G$, if it exists, we check if it is degree-obedient. If not, then $G$ does not cover $W(k, m, \ell, p, q)$. If yes, we check using Theorem 6.6 whether the blue subgraph of $G$ covers $F(k, m)$ and whether the red subgraph of $G$ covers $F(q, p)$. If any one of them does not, then $G$ does not cover $W(k, m, \ell, p, q)$. If both of them do, then $G$ covers $W(k, m, \ell, p, q)$,

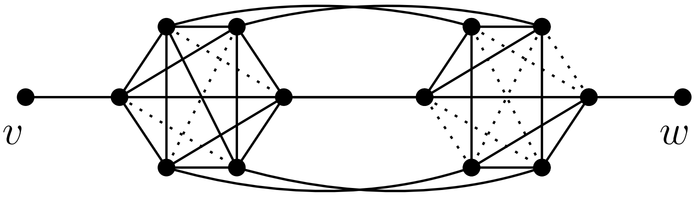


Figure 6.3: A gadget $G_{3,4}$ from Proposition 6.9.

since the "remaining" subgraph of $G$ formed by edges with one end-vertex red and the other one blue is $\ell$-regular and bipartite, thus covering the $\ell$ parallel edges of $W(k,m,\ell,p,q)$ (Proposition 6.2).

2. In Case 2, the input graph $G$ covers $W(1,0,1,0,1)$ only if $G$ is 2-regular. If this holds, then $G$ is a disjoint union of cycles and paths with one semi-edge attached to each of its endpoints. In both cases, it is easy to see that such components cover $W(1,0,1,0,1)$ if and only if their length is divisible by 4. For the subcase of $k = q = 0$, see the next point.
3. The input graph $G$ covers $W(0,0,\ell,0,0)$ only if it is a bipartite $\ell$-regular graph without semi-edges, but in that case it does cover $W(0,0,\ell,0,0)$, as follows from Proposition 6.2.

### 6.4.2 NP-hardness for non-regular target graphs

**Proposition 6.9.** *Let the parameters $k, m, p, q$ be such that $k + 2m \neq 2p + q$, and (($k \geq 3$ or $k = 2$ and $m \geq 1$), or ($q \geq 3$ or $q = 2$ and $p \geq 1$)). Then the $W(k,m,\ell,p,q)$-*Cover *problem is NP-complete.*

*Proof.* The parameters imply that at least one of the problems $F(k,m)$-Cover and $F(q,p)$-Cover is NP-complete by Section 6.3. Without loss of generality assume that this is the case of $F(q,p)$-Cover.

Let $a = k+2m$ and $b = 2p+q$ and let $c$ be the smallest even number greater than both $a$ and $b$. We shall construct a gadget which will be used in our reduction. We shall start with the construction for $\ell = 1$.

We take two disjoint copies of $K_c$ and denote the vertices in the cliques as $x_1, \ldots, x_c$ and $y_1, \ldots, y_c$, respectively. Remove $(c-b-1)$ edge-disjoint perfect matchings, corresponding to $(c-b-1)$ colour classes in some fixed $(c-1)$-edge-colouring of $K_c$, from the first copy of $K_c$, and remove $(c-a-1)$ edge-disjoint perfect matchings, corresponding to $(c-a-1)$ colour classes in some fixed $(c-1)$-edge-colouring of $K_c$, from the second one. Add two new vertices $v, w$ and connect them by edges $vx_1$ and $wy_1$. Furthermore, add edges $x_iy_i$ for all $2 \leq i \leq c$.

We denote the resulting graph by $G_{a,b}$. See Figure 6.3 for an example.

If $\ell > 1$, take $\ell$ disjoint copies of $G_{a,b}$ and denote their $v$-vertices as $v_1, \ldots, v_\ell$ and

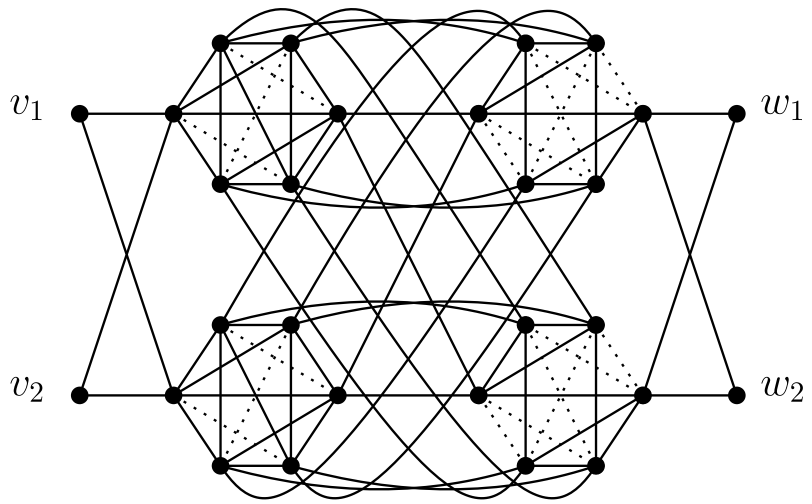


Figure 6.4: A gadget $G_{3,2,4}$ from Proposition 6.9.

their $w$-vertices as $w_1, \ldots, w_\ell$. Furthermore, denote the corresponding vertices in the $j$-th copy $(1 \leq j \leq \ell)$ of $G_{a,b}$ as $x_{j,1}, \ldots, x_{j,c}$ and $y_{j,1}, \ldots, y_{j,c}$.

Insert edges between vertices $v_1, \ldots, v_\ell$ and $x_{1,1}, \ldots, x_{\ell,1}$ so that they induce a complete bipartite graph with one part being $v_1, \ldots, v_\ell$ and the other part being $x_{1,1}, \ldots, x_{\ell,1}$. The analogous construction will be done for $w_1, \ldots, w_\ell$ and $y_{1,1}, \ldots, y_{\ell,1}$. Moreover, for each $i \in \{2, \ldots, c\}$, insert edges between $x_{1,i}, \ldots, x_{\ell,i}$ and $y_{1,i}, \ldots, y_{\ell,i}$ so that they induce a complete bipartite graph with one part being $x_{1,i}, \ldots, x_{\ell,i}$ and the other part being $y_{1,i}, \ldots, y_{\ell,i}$. Denote the resulting graph as $G_{a,\ell,b}$ (for $\ell = 1$, we set $G_{a,1,b} = G_{a,b}$). See Figure 6.4 for an example.

We will reduce from the problem $F(q,p)$-Cover, which is NP-complete for these parameters by the results of the preceding section. Let $G$ be an instance of $F(q,p)$-Cover with $n$ vertices. Without loss of generality we may assume that $n$ is even. We shall construct a new graph $G'$ in the following way. Take $\ell$ copies of the graph $G$ and denote their vertices as $t_{j,1}, \ldots, t_{j,n}$ in the $j$-th copy, respectively. Take $\ell$ copies of a graph with $n$ vertices that covers $F(k,m)$ (any $a$-regular bipartite graph on $n$ vertices will do) and denote their vertices as $u_{j,1}, \ldots, u_{j,n}$ in the $j$-th copy, respectively. For each $h$, $1 \leq h \leq n$, take a new extra copy of $G_{a,\ell,b}$, denote their $v$ and $w$ vertices as $v_{h,1}, \ldots, v_{h,\ell}, w_{h,1}, \ldots, w_{h,\ell}$ in the $h$-th copy, respectively, and identify $v_{h,j}$ with $u_{j,h}$ and $w_{h,j}$ with $t_{j,h}$ for each $1 \leq j \leq \ell$ and $1 \leq h \leq n$. Note that the constructed graph $G'$ is linear in the size of $G$. We claim that $G'$ covers $W(k,m,\ell,p,q)$ if and only if $G$ covers $F(q,p)$.

For the 'only if' direction, suppose that $G'$ covers $W(k,m,\ell,p,q)$. First of all, because of the different degrees of the vertices of $W(k,m,\ell,p,q)$, we have a clear information about the vertex mapping part of the covering projection. In particular, the $v$ and $y$ vertices of the copies of $G_{a,\ell,b}$ are mapped onto the vertex of degree $a+\ell$ in $W(k,m,\ell,p,q)$, while the $x$ and $w$ ones are mapped onto the vertex of degree $b+\ell$. Hence the edges of each copy of $G$ must map onto the loops and half-edges incident with the vertex of degree $b+\ell$ in $W(k,m,\ell,p,q)$, and hence $G$ covers $F(q,p)$.

For the other direction, the covering projection from $G'$ onto $W(k,m,\ell,p,q)$ is constructed as follows. Map the $v$ and $y$ vertices of the copies of $G_{a,\ell,b}$ onto the vertex of degree $a+\ell$ in $W(k,m,\ell,p,q)$, and the $x$ and $w$ ones onto the vertex of degree $b+\ell$. This is a degree obedient vertex mapping of $V(G')$ onto the vertices of $W(k,m,\ell,p,q)$. The edges of $G'$ with one end-vertex of degree $a+\ell$ and the other one of degree $b+\ell$ induce a bipartite $\ell$-regular graph, and therefore can be mapped to the $\ell$ bars of $W(k,m,\ell,p,q)$ in a locally bijective way. If we delete these edges, $G'$ falls apart into several components of connectivity. The components induced by the $x$ vertices from copies of $G_{a,\ell,b}$ are $a$-regular $a$-edge colourable subgraphs of $G_{a,\ell,b}$ and hence their edges cover $F(k,m)$. The components induced by the $y$ vertices from copies of $G_{a,\ell,b}$ are $b$-regular $b$-edge colourable subgraphs of $G_{a,\ell,b}$ and hence their edges cover $F(q,p)$. The components induced by the $v$ vertices induce copies of the $a$-regular $a$-edge colourable graph chosen in the construction of $G'$, and hence they cover $F(k,m)$. Last but not least, the components induced by the $w$ vertices are isomorphic to $G$, whose edges cover $F(q,p)$ by the hypothesis of the 'if' direction of the proof. Putting all these edge mappings together we obtain a covering projection from $G'$ onto $W(k,m,\ell,p,q)$, which concludes the proof. □

### 6.4.3 NP-hardness for connected regular target graphs

The aim of this subsection is to conclude the proof of Theorem 6.8 by showing the NP-hardness for the case of $\ell \geq 1$ and $k+2m=2p+q$. We will actually prove a result which is more general in two directions. First, we formulate the result in the language of colourings of vertices, and then we prove the hardness for bipartite inputs. This might seem surprising, as we have seen in Section 6.2 that bipartite graphs can make things easier. Moreover, this strengthening in fact allows us to prove the result in a unified, and hence simpler, way.

Note that the following definition of a relaxation of usual proper 2-colouring resembles the so-called *defective 2-colouring* (see survey of Wood [162]). However, the definitions are not equivalent and we are not aware of any deeper connection at the moment.

**Definition 6.10.** *A* $(b,c)$-colouring *of a graph is a 2-colouring of its vertices such that every vertex has b neighbours of its own colour and c neighbours of the other colour. (Note that $(b,c)$-colouring is proper if and only if $b=0$.)*

**Observation 6.11.** *For any parameters $k,m,\ell,p,q$ such that $k+2m=2p+q$, a bipartite graph $G$ with no semi-edges covers $W(k,m,\ell,p,q)$ if and only if it allows a $(k+2m,\ell)$-colouring.*

*Proof.* On one hand, any graph covering projection from $G$ to $W(k,m,\ell,p,q)$ induces a $(k+2m,\ell)$-colouring of $G$, provided $k+2m=2p+q$. On the other hand, a $(k+2m,\ell)$-colouring of $G$ is a degree-obedient vertex mapping from $G$ to $W(k,m,\ell,p,q)$, again provided that $k+2m=2p+q$. If $G$ is bipartite and has no semi-edges, then this mapping can be extended to a graph covering projection by Theorem 6.4. □

In view of the previous observation, we will be proving the NP-hardness results for the following problem:

| | |
|---|---|
| PROBLEM: | $(a,b)$-COLOURING |
| INPUT: | A graph $G$. |
| QUESTION: | Does $G$ allow an $(a,b)$-colouring? |

**Theorem 6.12.** *For every pair of positive integers $b,c$ such that $b+c\geq 3$, the $(b,c)$-*COLOURING *problem is NP-complete even for simple bipartite graphs.*

Theorem 6.12 and Observation 6.11 imply the following proposition, which concludes the proof of Theorem 6.8.

**Proposition 6.13.** *The $W(k,m,\ell,p,q)$-*COVER *problem is NP-complete for simple bipartite input graphs for all parameter sets such that $k+2m=2p+q\geq 1$, $\ell\geq 1$, and $k+2m+\ell\geq 3$.*

The rest of this subsection is devoted to the proof of Theorem 6.12.

**Observation 6.14.** *A bipartite graph $G$ allows a $(b,c)$-colouring if and only if it allows a $(c,b)$-colouring.*

*Proof.* Let $A$ and $B$ be the classes of bipartition of $V(G)$ and assume that $G$ has a $(b,c)$-colouring using red and blue colours. By swapping these colours on the set $B$ we obtain a $(c,b)$-colouring. □

**Corollary 6.15.** *The problems $(b,c)$-*COLOURING *and $(c,b)$-*COLOURING *are polynomially equivalent on bipartite graphs.*

**Proposition 6.16.** *For every $b\geq 2$, the problem $(b,1)$-*COLOURING *is NP-complete even for simple bipartite graphs on input.*

We will develop the proof as a series of claims. We first consider $(2,1)$-colouring of cubic bipartite graphs. Through our arguments the classes of bipartition will be indicated in figures by vertex shapes — squares and triangles, while for the $(2,1)$-colouring we use red and blue colours.

Observe first that whenever a graph $G$ contains a $C_4$ as an induced subgraph then in any $(2,1)$-colouring of $G$ it is impossible to colour exactly three vertices of the $C_4$ by the same colour. The reason is that in such a case the remaining vertex

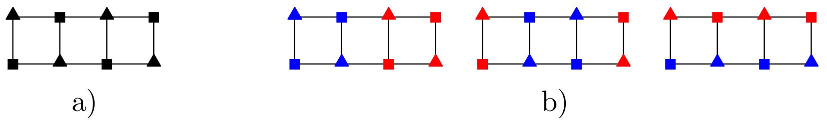


Figure 6.5: Partial $(2,1)$-colourings of an 8-vertex auxiliary subgraph.

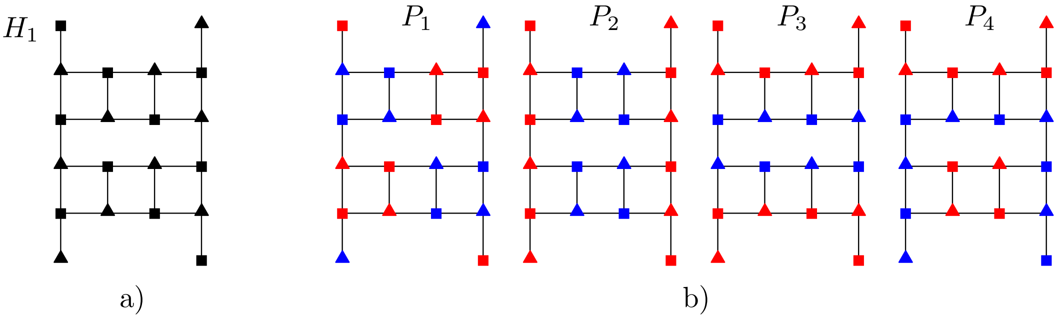


Figure 6.6: A 20-vertex auxiliary graph $H_1$ and its possible partial $(2,1)$-colourings.

would be adjacent to two vertices of the opposite colour, which is not allowed. By the same argument we deduce that if both colours are used on the $C_4$ then vertices of the same colour are adjacent.

The following two observations are immediate.

**Observation 6.17.** *Whenever a graph $G$ contains as a subgraph the graph on 8 vertices depicted in Figure 6.5 a) then in any in any $(2,1)$-colouring of $G$ the colour classes match on the subgraph one of the three patterns depicted in red and blue colours in Figure 6.5 b).*

**Observation 6.18.** *Whenever a graph $G$ contains as a subgraph the graph $H_1$ on 20 vertices depicted in Figure 6.6 a) then in any in any $(2,1)$-colouring of $G$ the colour classes match on $H_1$ one of the four patterns depicted in red and blue colours in Figure 6.6 b).*

**Lemma 6.19.** *Let a graph $G$ contains as a subgraph the graph $H_2$ depicted in Figure 6.7 a). Then in any $(2,1)$-colouring of $G$ the vertices $u_1$ and $u_2$ have the same colour and their neighbours $w_1$, $w_2$ the opposite colour.*

*Proof.* The graph $H_2$ contains three induced copies of $H_1$. If the pattern $P_1$ of Figure 6.6 b) was used on some copy, then the same pattern must be used on all three copies. Consequently, the vertex $w_1$ has two neighbours of the opposite colour as indicated in Figure 6.7 b), which is not allowed. This excludes the pattern $P_1$ from our reasoning.

If the pattern $P_4$ was used on the middle copy of $H_1$, then the vertices $v_1$ and $v_2$

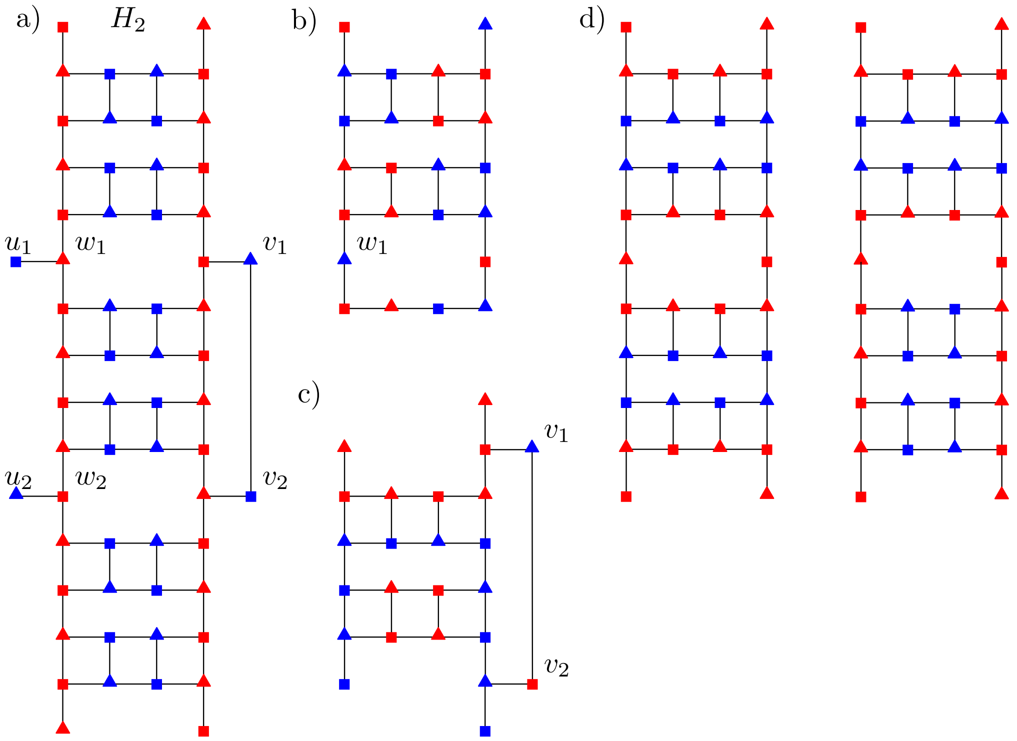


Figure 6.7: Forcing the same colour on $u_1$ and $u_2$.

have two neighbours of the opposite colour as indicated in Figure 6.7 c), which is also not allowed.

Therefore the middle copy of $H_1$ uses either pattern $P_2$ or $P_3$ and the claim follows. Note that both patterns might be used on the same $H_2$ see, Figure 6.7 a) and d) □

**Lemma 6.20.** *The problem* (2, 1)-COLOURING *is NP-complete even for simple bipartite graphs.*

*Proof.* We reduce from the well known NP-complete problem NOT-ALL-EQUAL 3-SAT [88], which given a formula $\phi$ in CNF without negation, consisting of clauses $C_1, \ldots, C_m$, where each $C_j$ is a disjunction of exactly three distinct literals, asks whether $\phi$ has a truth assignment such that each clause contains a negatively valued literal.

For given $\phi$ we build a bipartite cubic graph $G$ that allows a $(2,1)$-colouring if and only if $\phi$ allows required assignment. The graph has several functional blocks: variable gadgets, clause gadgets enforcing the valid truth assignment already for a partial $(2,1)$-colouring and also garbage collection allowing to extend the partial colouring to the entire cubic graph. By partial $(2,1)$-colouring we mean a restriction of a $(2,1)$-colouring to a subgraph, i.e. a vertex 2-colouring where every vertex has at most two neighbours of its own colour and at most one neighbour

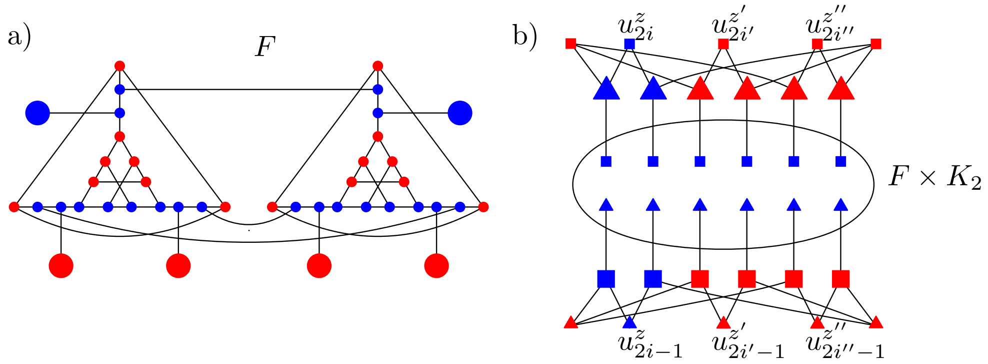


Figure 6.8: Garbage collection and the overall construction for Theorem 6.20. Clause gadgets are in the corners of the part b).

of the other colour.

For a variable $z$ that in $\phi$ has $k$ occurrences, we build a variable gadget consisting of a cyclic chain of $2k$ graphs $H_1$ linked together with further vertices $u^z_i$ and $v^z_i$ so each three consecutive copies of $H_1$ induce the graph $H_2$ of Figure 6.7 a). In this gadget the colour of $u^z_1, \ldots, u^z_{2k}$ represent the truth assignment of $z$.

The clause gadget for a clause $C_j$ is a claw $K_{1,3}$. When a variable $z$ occurs in a clause $C_j$ we add an edge between an $u^z_{2i}$ and a unique leaf of the clause gadget $K_{1,3}$ so that each clause gadget is linked to a distinct $u^z_{2i}$.

Observe that any partial $(2,1)$-colouring of the so far formed graph corresponds to the valid truth assignments and vice-versa: leaves of each clause gadget $K_{1,3}$ are not monochromatic, while the edges added between the vertex and clause gadget have both end of the same colour as each $u^z_{2i}$ has already a neighbour $v^z_{2i}$ of the other colour.

It remains to extend the graph to a cubic graph so that the partial $(2,1)$-colouring is preserved within a "full" $(2,1)$-colouring. We first add further copies of clause gadgets and link them to the vertex gadgets by the same process so that each $u^z_{2i}$ is linked to exactly two clause gadgets and then repeat the same process twice for vertices $u^z_{2i-1}$ with odd valued indices. Now the only vertices that do not have degree three are the former leaves of clause gadgets, where each is now of degree two.

For this purpose we involve an auxiliary graph $F$ and one of its partial $(2,1)$-colourings depicted in Figure 6.8 a). For each clause $C_j$ we take a copy of the bipartite graph $F \times K_2$ and merge its 12 vertices of degree one with the twelve vertices of degree two stemming from the four copies of the clause gadgets as shown in Figure 6.8 a). The merged vertices are indicated by big symbols.

This step completes the construction of the desired simple cubic bipartite graph $G$ that allows a $(2,1)$-colouring if and only if $\phi$ allows not all equal truth assignment. The way how such truth assignment can be derived from a $(2,1)$-colouring has been already discussed. In the opposite way, the truth assignment yields a colouring of the vertex gadgets, say blue colour would represent truly evaluated variables, while red negative ones. Then the colouring can be completed to clause gadgets and auxiliary graphs $F \times K_2$ by using patterns depicted in Figure 6.8. In the last step we involve the standard lift of a colouring to a product, namely that the same colour is used on the two copies of a vertex in the $F \times K_2$ as the original vertex has in $F$. □

*Proof of Proposition 6.16.* For $b \geq 3$ we reduce the $(2,1)$-COLOURING to $(b,1)$-COLOURING. Let $G$ be a bipartite cubic graph whose $(2,1)$-colouring has to be decided.

First we construct an auxiliary graph $F$ consisting of two disjoint unions of $K_{b,b}$ with classes of bipartition $A_1, B_1, A_2$ and $B_2$ that are joined together by two perfect matchings, one between sets $A_1$ and $A_2$ and the other between $B_1$ and $B_2$. Finally, we add two vertices $u$ and $v$, make $u$ adjacent to some $u' \in A_1$ and $v$ adjacent to some $v' \in B_1$ and remove the edge $(u', v')$.

We claim that in any partial $(b,1)$-colouring of $F$ the vertices $u, v, u'$ and $v'$ receive the same colour. Observe first that the complete bipartite graph $K_{b,b}$ on $A_2$ and $B_2$ is monochromatic as otherwise one vertex would have at least two neighbours of the opposite colour. Now each vertex of $A_2$ and $B_2$ has $a$ neighbours of the same colour, say red, so the sets $A_1$ and $B_1$ are blue. The vertex $u'$ now has a single red neighbour and $b-1$ blue neighbours so $u$ is blue as well. Analogously for $v$ and $v'$.

We take two copies $G_1$ and $G_2$ of the graph $G$ and for each $w \in V_G$ we insert $b-2$ copies $F_1^w, \dots, F_{b-2}^w$ of the graph $F$, where we identify $w_1$ with $u_1^w, \dots, u_{b-2}^w$ and also $w_2$ with $v_1^w, \dots, v_{b-2}^w$. By this process we get a bipartite $(b+1)$-regular graph $H$.

The fact that graph $H$ allows an $(b,1)$-colouring if and only if $G$ allows an $(2,1)$-colouring follows from the fact that the all $b-2$ neighbours of any $w_1$ outside $G_1$, i.e. inside the copies of $F$, have the same colour as $w_1$. □

**Proposition 6.21.** *For every $c \geq 2$ and $b \geq c+2$, the $(b,c)$-*COLOURING *problem is NP-complete even for simple bipartite graphs.*

*Proof.* We will prove $(1,c)$-COLOURING $\propto$ $(b,c)$-COLOURING for simple bipartite inputs. Given a simple bipartite $(1+c)$-regular graph $G$ as input of $(1,c)$-COLOURING, construct a graph $G'$ by taking two disjoint copies $G_1, G_2$ of $G$ and connecting them by "bridges" as follows. Let $H$ be a graph with two pendant vertices $x, t$ of degree 1 and all other vertices of degree $b+c$. Let $y$ be the neighbour of $x$ and $s$ the neighbour of $t$ in $H$. The vertices of degree $b+c$ in $H$ will be called its *inner vertices*. Let the companion vertices of $G_1$ and $G_2$ that are copies of a

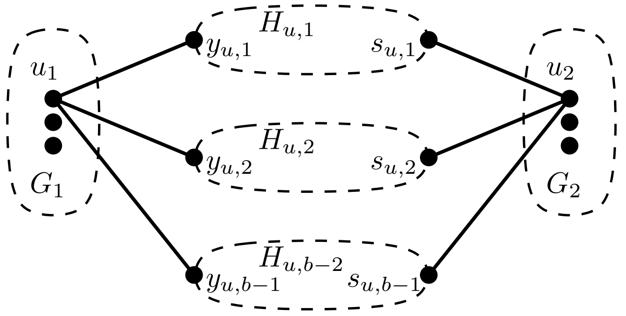


Figure 6.9: An illustration to the construction of graph $G'$ from Proposition 6.21.

vertex $u$ of $G$ be denoted by $u_1$ and $u_2$, respectively. For every vertex $u \in V(G)$, take $b-1$ copies $H_{u,i}, i = 1, 2, \ldots, b-1$ of $H$, with vertices of $H_{u,i}$ denoted by $z_{u,i}$, for $z \in V(H)$. For every $u \in V(G)$, identify the vertices $x_{u,i}, i = 1, 2, \ldots, b-1$ with the vertex $u_1$ and identify the vertices $t_{u,i}, i = 1, 2, \ldots, b-1$ with the vertex $u_2$. See an illustration in Figure 6.9.

**Lemma 6.22.** *Suppose that the number of inner vertices of $H$ is divisible by 4. Let $\varphi : V(H) \longrightarrow \{red, blue\}$ be a red-blue colouring of the vertices of $H$ such that every inner vertex has exactly $b$ neighbours of its own colour (and hence $c$ neighbours of the other colour). Then either $\varphi(x) = \varphi(y) = \varphi(s) = \varphi(t)$, or $\varphi(x) = \varphi(s) \neq \varphi(y) = \varphi(t)$.*

*Proof.* Let $\alpha$ be the number of inner vertices that are coloured red, and let $\beta$ be the number of inner vertices that are coloured blue. Every red inner vertex has $c$ blue neighbours, and so $H$ has $\alpha c$ red-blue edges, with at most two of them being the pendant ones. Similarly, $H$ has $\beta c$ red-blue edges, with at most two of them being the pendant ones. Hence

$$\alpha c - \epsilon_r = \beta c - \epsilon_b$$

for some $\epsilon_r, \epsilon_b \in \{0, 1, 2\}$ (even with some restriction, e.g., $\epsilon_r, \epsilon_b$ cannot be both equal to 2, but that is not important). Therefore,

$$|(\alpha - \beta)c| \leq 2.$$

If $c > 2$, this immediately implies $\alpha = \beta$. If $c = 2$, we might get $|\alpha - \beta| = 1$, but then $\alpha$ and $\beta$ would be of different parities, contradicting the assumption of $\alpha + \beta$ being even. We conclude that $\alpha = \beta$, and this quantity is even.

Suppose $x$ and $y$ have the same colour, say the red one. Then both $s$ and $t$ must be red as well, because

- $\varphi(s) = red, \varphi(t) = blue$ would yield $\alpha c - 1 = \beta c$, which is impossible,
- $\varphi(s) = blue, \varphi(t) = red$ would yield $\alpha c = \beta c - 1$, which is impossible, and

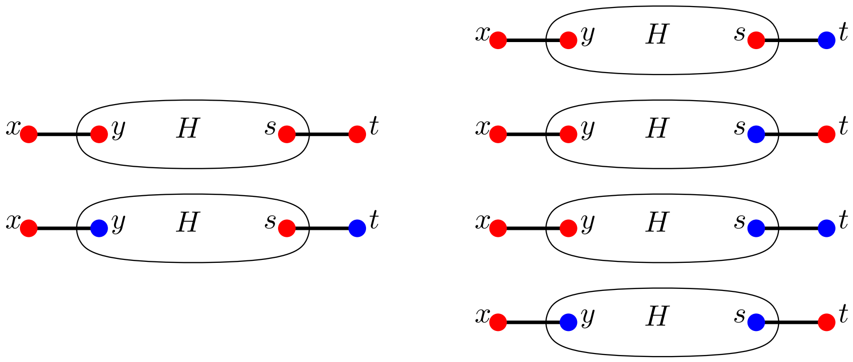


Figure 6.10: Feasible (in the left) and infeasible (in the right) red-blue colourings of a bridge graph $H$.

- $\varphi(s) = \varphi(t) = blue$ would imply that the red subgraph of $H$ has an odd number of vertices of odd degree (either 1, if $b$ is even, or $\alpha + 1$ if $b$ is odd), which is impossible by the well known Handshaking lemma.

Let $x$ and $y$ have different colours, say $x$ is red and $y$ is blue. Then $s$ and $t$ cannot have the same colour by an argument symmetric to the one above. We cannot have $s$ blue and $t$ red, since $\alpha c = \beta c - 2$ in such a case, which is not possible since $\alpha + \beta$ is divisible by 4. Hence $s$ must be red and $y$ blue. This concludes the proof of Lemma 6.22. (See Figure 6.10 for an illustration of feasible and infeasible colourings of $H$.) □

**Lemma 6.23.** *For every $c \geq 2$ and $b \geq c+2$, there exists a bridge graph $H$ whose number of inner vertices is divisible by 4 and which allows a $(b, c)$-colouring such that all four vertices $x, y, s, t$ have the same colour.*

*Proof.* Take two disjoint copies of the complete bipartite graph $K_{b,b}$. Let the classes of bipartition in one of them be $A$ and $B$, and the classes of bipartition in the other one $C$ and $D$. Pick an edge $ys$ such that $y \in A$ and $s \in B$. Add $c$ disjoint perfect matchings between $A$ and $C$, and other $c$ disjoint perfect matching between $B$ and $D$. In this way we have obtained a $(b + c)$-regular bipartite graph with $4b$ vertices. Delete the edge $ys$ and add pendant edges $xy$ and $st$ with new extra vertices $x, t$ of degree 1 to obtain the desired bridge graph $H$. Indeed, colouring $A \cup B \cup \{x, t\}$ red and $C \cup D$ blue yields a $(b, c)$-colouring such that the 4 vertices $x, y, s, t$ get the same colour. See an illustrative example in Figure 6.11. □

Let us return to the proof of Proposition 6.21. Given a simple bipartite graph $G$, we construct $G'$ as described using the bridge graph $H$ from Lemma 6.23. This $G'$ is simple, and since $H$ was created from a bipartite graph, $G'$ is bipartite as well. The proof of the proposition now follows from the following lemma.

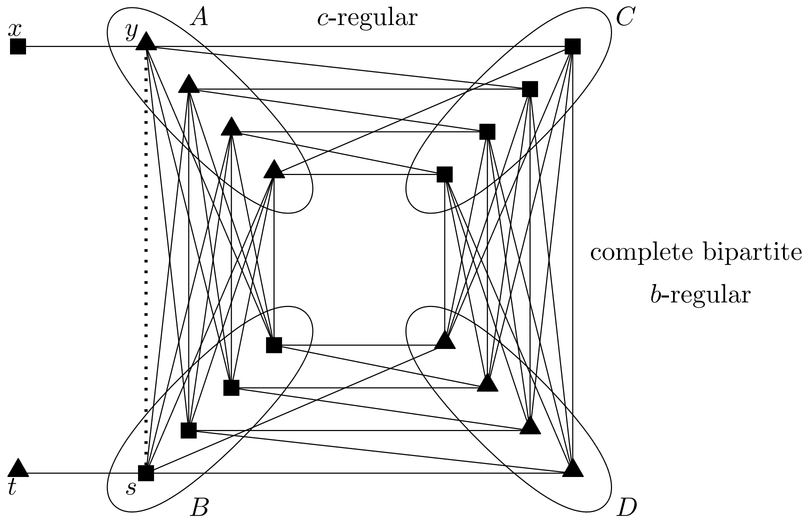


Figure 6.11: An example of the bridge graph $H$ for $b = 4$ and $c = 2$.

**Lemma 6.24.** *The graph $G'$ allows a $(b, c)$-colouring if and only if $G$ allows a $(1, c)$-colouring.*

*Proof.* Suppose $G'$ allows a $(b, c)$-colouring, say $\varphi$. Consider a vertex $u \in V(G)$. Lemma 6.22 implies that either

- $\varphi(u_1) = \varphi(y_{u,i}) = \varphi(s_{u,i}) = \varphi(u_2)$ for all $i = 1, 2, \ldots, b - 1$, or
- $\varphi(u_1) = \varphi(s_{u,i}) \neq \varphi(y_{u,i}) = \varphi(u_2)$ for all $i = 1, 2, \ldots, b - 1$.

But the latter would mean that $u_1$ has $b-1 > c$ neighbours of the opposite colour, which is too many. Hence every vertex $u_1$ has $b - 1$ neighbours of its own colour in the bridge graphs, and therefore the restriction of $\varphi$ to $G_1$ is a $(1, c)$-colouring of $G_1$ (which is isomorphic to $G$).

On the other hand, if $G$ allows a $(1, c)$-colouring, use the same colouring on $G_1$ and $G_2$ and colour the bridges so that for every $u \in V(G)$, both $u_1$ and $u_2$ have all their $b - 1$ neighbours in the bridge graphs coloured with their own colour. This is possible by Lemma 6.23, and this gives a $(b, c)$-colouring of $G'$. □

□

**Proposition 6.25.** *For every $c \geq 2$, the $(c + 1, c)$-COLOURING problem is NP-complete even for simple bipartite graphs.*

*Proof.* We will prove $(1, c)$-COLOURING $\propto$ $(c + 1, c)$-COLOURING for simple bipartite inputs. Given a simple bipartite $(1 + c)$-regular graph $G$ as input of $(1, c)$-COLOURING, construct a graph $G'$ by taking two disjoint copies $G_1, G_2$ of $G$ and connecting them by "bridges", similarly as in the proof of Proposition 6.21.

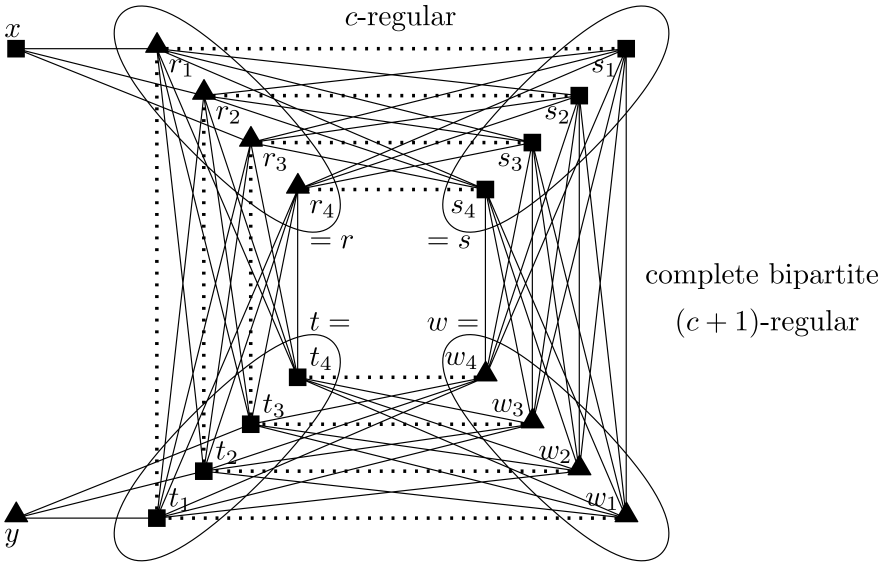


Figure 6.12: An example of the bridge graph $H$ for $c = 3$.

But this time we will describe the bridge graph $H$ explicitly from the very beginning of the proof. It has $4(c+1)$ "inner" vertices of degree $2c+1$ and two "connector" vertices of degree $c$. The inner part of $H$ is created from two copies of the complete bipartite graph $K_{c+1,c+1}$ whose classes of bipartition are connected by cocktail-party graphs (i.e., complete bipartite graphs minus a perfect matching), and in one of the copies $c$ independent edges are deleted and replaced by edges leading to the connector vertices. The graph is illustrated in Figure 6.12, but since we will heavily rely on its structure in the proof of its properties, we also describe it formally:

$$\begin{aligned} V(H) =& \{x,y\} \cup \bigcup_{i=1}^{c+1} \{r_i, s_i, t_i, w_i\}, \\ E(H) =& \bigcup_{i=1}^{c} \{xr_i, yt_i\} \cup \left( \bigcup_{i,j=1}^{c+1} \{r_i t_j\} \setminus \bigcup_{i=1}^{c} \{r_i r_i\} \right) \\ & \cup \bigcup_{i,j=1}^{c+1} \{s_i w_j\} \cup \left( \bigcup_{i,j=1}^{c+1} \{r_i s_j, t_i w_j\} \setminus \bigcup_{i=1}^{c+1} \{r_i s_i, t_i w_i\} \right) \end{aligned} \tag{6.1}$$

where for the sake of brevity, but also to stress their special roles, we write $r = r_{c+1}, s = s_{c+1}, t = t_{c+1}$ and $w = w_{c+1}$.

In the construction of $G'$, for every $u \in V(G)$, let the companion vertices in $G_1$ and $G_2$ which are copies of $u$ be again denoted by $u_1$ and $u_2$, respectively. We take a copy $H_u$ of $H$ and unify its connector vertices with $u_1$ and $u_2$. See an illustrative example in Figure 6.13. Note finally, that $G'$ is a bipartite graph, since $H$ is bipartite and the distance of $x$ and $y$ in $H$ is odd.

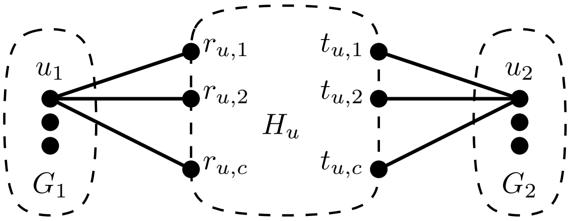


Figure 6.13: An illustration to the construction of $G'$ from Proposition 6.25.

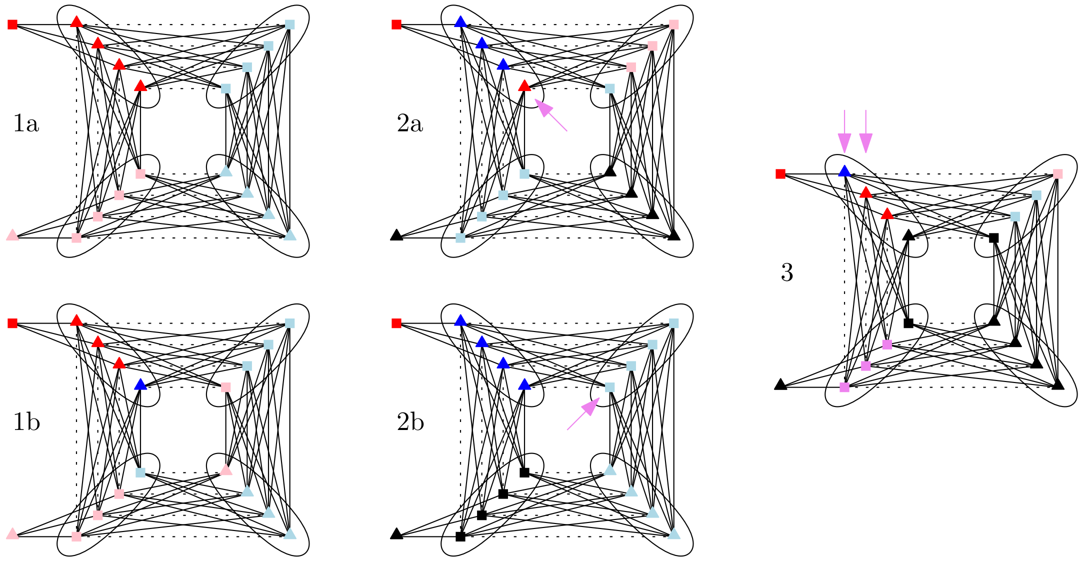


Figure 6.14: An illustration to the case analysis of Lemma 6.26.

**Lemma 6.26.** *Let $\varphi : V(H) \longrightarrow \{red, blue\}$ be a red-blue colouring of the vertices of $H$ such that every inner vertex has exactly $c+1$ neighbours of its own colour (and hence $c$ neighbours of the other colour). Then $\varphi(x) = \varphi(r_i) = \varphi(t_i) = \varphi(y)$ for all $i = 1, 2, \ldots, c$.*

*Proof.* Suppose $\varphi(x) = red$. We will prove the result by a case analysis. In the illustrative Figure 6.14, the assumptions of the cases are marked with dark red and blue, the colourings that are derived from them by light red and blue, and the vertices that cause contradictions are stressed by arrows.

*Case 1: $\varphi(r_i) = red$ for all $i = 1, 2, \ldots, c$.*

*Subcase 1a*: $\varphi(r) = red$
In this case any two vertices $s_i, s_j$ will end up with the same number of red neighbours, regardless of the colours on $w_1, \ldots, w_{c+1}$. Therefore all $s_i$'s must have the same colour. Every vertex $w_i$ then already has $c+1$ neighbours of this colour among $s_i$'s, and thus all vertices $w_i$ have the same colour as the $s_i$'s. If this colour were red, every $s_i$ would have $2c+1$ red neighbours and no blue ones.

Hence $\varphi(s_i) = \varphi(w_i) = blue$ for all $i = 1, 2, \ldots, c+1$. Then each $w_i$ has already $c+1$ neighbours of its own colour, and so all the other neighbours (i.e., the vertices $t_i$, $i = 1, 2, \ldots, c+1$) are red. Now $t_1$ has only $c$ red neighbours among the $r_i$'s, and therefore $y$ must be red as well.

*Subcase 1b:* $\varphi(a) = blue$
In this case, each $s_i$, $i = 1, 2, \ldots, c$ will end up seeing less red neighbours than $s$, regardless of the colours on $w_i$'s ($s$ has a red neighbour $r_i$, while $r_i$ is not a neighbour of $s_i$, and the private neighbour $r$ of $s_i$ is blue). Hence $s$ must be red and all $s_i$, $i = 1, 2, \ldots, c$ are blue. To supply the $s_i$'s with correct numbers of red neighbours, exactly one of the $w_i$'s must be red, all others are blue. The red one has just one red neighbour among $s_i$'s, and hence at least $c$ of the $t_i$'s are red. The blue vertices among $w_i$'s have $c$ blue neighbours among $s_i$'s, and so at least one of the $t_i$'s is blue. It follows that $\varphi(w_i) \neq \varphi(t_i)$ for all $i = 1, 2, \ldots, c+1$. Since every $r_i$, $i = 1, 2, \ldots, c$ has two red neighbours $x$ and $s$, it should have only (and exactly) $c-2$ red neighbours among $t_i$'s, and hence $\varphi(t_i) = \varphi(r_i) = red$ for $i = 1, 2, \ldots, c$. Then $\varphi(t) = blue$. Since $t_1$ has so far $c$ red neighbours ($c-1$ among $r_i$'s and one among $w_i$'s), $y$ must be red.

*Case 2:* $\varphi(r_i) = blue$ *for all* $i = 1, 2, \ldots, c$.

*Subcase 2a:* $\varphi(r) = red$
Any two $s_i, s_j$, $i, j = 1, 2, \ldots, c$ will end up with the same number of red neighbours (regardless the colouring of the $w_i$'s), and hence all $s_i$, $i = 1, 2, \ldots, c$ have the same colour. Since $r$ is not a neighbour of $s$, $s$ will end up with less red neighbours than $s_1$. Therefore, $\varphi(s_i) = red$ for $i = 1, 2, \ldots, c$, and $\varphi(s) = blue$. Since $x$ is red, every $r_i$, $i = 1, 2, \ldots, c$ must have $c$ blue neighbours among the $t_i$'s, and because $c \geq 2$, it follows that all $t_i$'s (including $t = t_{c+1}$) are blue. But then the red vertex $r$ has too many ($c+1$) blue neighbours, a contradiction.

*Subcase 2b:* $\varphi(a) = blue$
Any two $s_i$ vertices will end up with the same number of red neighbours, and hence all $s_i$'s (including $s$) have the same colour, and this colour must be blue, since a blue vertex $r_1$ would have $c+1$ red neighbours otherwise. Now every $w_i$ has already $c+1$ blue neighbours (the $s_i$'s), and thus all $w_i$'s are blue. But this causes a contradiction, since now each $s_i$ has all $2c+1$ neighbours blue.

*Case 3: At least one of the* $r_i$*'s for* $i = 1, 2, \ldots, c$ *is red and at least one of them is blue.*

Consider $i$ and $j$ such that $\varphi(r_i) = \varphi(r_j)$. Regardless the colouring of $w_i$'s, the vertices $s_i$ and $s_j$ will end up with the same number of red neighbours, and hence $\varphi(s_i) = \varphi(s_j)$. If, on the other hand, $\varphi(r_i) \neq \varphi(r_j)$, say $\varphi(r_i) = red$ and $\varphi(r_j) = blue$, then $s_i$ will end up with less red neighbours than $s_j$, and hence $\varphi(s_i) = blue$ and $\varphi(s_j) = red$. We conclude that for every $i = 1, 2, \ldots, c+1$, $r_i$ and $s_i$ get different colours.

Now consider two vertices $t_i, t_j$, $i, j = 1, 2, \ldots, c$. If $\varphi(r_i) = \varphi(r_j)$, then $r_i$ and

$r_j$ have the same number of red neighbours among $\{x\} \cup \{s_1, s_2, \dots, s_{c+1}\} \cup (\{t_1, t_2, \dots, t_{c+1}\} \setminus \{t_i, t_j\})$. In order to end up with the same number of red neighbours in total, it must be $\varphi(t_i) = \varphi(t_j)$. If $r_i$ and $r_j$ got different colours, say $\varphi(r_i) = red$ and $\varphi(r_j) = blue$, then among $\{x\} \cup \{s_1, s_2, \dots, s_{c+1}\} \cup (\{t_1, t_2, \dots, t_{c+1}\} \setminus \{t_i, t_j\})$, $r_i$ has one more red neighbours than $r_j$. But the same difference should apply to the total number of red neighbours of $r_i$ and $r_j$, and hence $\varphi(t_i) = \varphi(t_j)$. We conclude that all vertices $t_j, j = 1, 2, \dots, c$ have the same colour. Since the graph $H$ is symmetric, this is either Case 1 or Case 2 from the standpoint of the $t_i$'s. These cases have already been treated and either they lead to a contradiction, or they require that all vertices $r_i$, $i = 1, 2, \dots, \ell$ get the same colour. Which contradicts the assumption of Case 3. □

To conclude the proof of Proposition 6.25 it only remains to prove the following lemma.

**Lemma 6.27.** *The graph $G'$ allows a $(c+1, c)$-colouring if and only if $G$ allows a $(1, c)$-colouring.*

*Proof.* Suppose $\varphi$ is a $(c+1, c)$-colouring of $G'$. It follows from Lemma 6.26 that every vertex $u_1 \in V(G_1)$ has $c$ neighbours of its own colour in the corresponding bridge $H_u$, and thus the restriction of $\varphi$ to $G_1$ is a $(1, c)$-colouring of $G_1$ (which is isomorphic to $G$).

If $G$ allows a $(1, c)$-colouring, use it on both $G_1$ and $G_2$ and colour the bridges so that for every $u \in V(G)$, the $r_i$ and $t_i$ vertices of $H_u$ get the same colour as $u$ and the vertices $s_i$ and $w_i$ of $H_u$ get the opposite colour. This is a $(c+1, c)$-colouring of $G'$. □

With this, the proof of Proposition 6.25 concludes. □

**Proposition 6.28.** *For every $b \geq 2$, the $(b, b)$-COLOURING problem is NP-complete even for simple bipartite graphs.*

*Proof.* We will reduce from the following problem.

| | |
|---|---|
| PROBLEM: | $(k$-IN-$2k)$-SAT$_q$ |
| INPUT: | A formula $\phi$ with clauses $C_1, \dots, C_m$ in CNF without negations, each $C_i$ is a disjunction of exactly $2k$ distinct literals and every variable occurs exactly $q$ times. |
| QUESTION: | Does there exist a satisfying assignment of $\phi$ such that exactly $k$ literals are true in each $C_i$? |

The problem $(k$-IN-$2k)$-SAT$_q$ was proved to be NP-complete by Kratochvíl [121] for every $k \geq 2, q \geq 3$.

Let $\phi$ be an instance of ($b$-IN-$2b$)-SAT$_q$, $b \geq 2$, with each variable occurring $q = b + 1$ times. Let $C_1, \ldots, C_m$ be the clauses of $\phi$.

Our clause gadget is a complete bipartite graph $K_{b,2b}$. The vertices in the bigger part correspond to variables. More formally, for every variable $x$ occurring in a clause $C_i$, the clause gadget has a vertex $y_x^i$ in its bigger part. To make sure that each variable has an even number of occurrences, we will duplicate each clause gadget and we will refer to these copies as the *left* and *right* ones, with their $y$ vertices being denoted by $y_x^{i,l}$ and $y_x^{i,r}$, respectively.

For each variable $x$, we will construct a variable gadget $V_b$ in the following way. Take complete bipartite graph $K_{2b+1,2b+1}$ and denote its vertices in one part as $u_1, \ldots, u_{2b+1}$ and in the other part as $v_1, \ldots, v_{2b+1}$. Remove the edges $u_iv_i$ for each $1 \leq i \leq 2b+1$ and the edges $u_iv_{i+b}$ for each $2 \leq i \leq b+1$. Take two copies $K_1, K_2$ of the resulting graph and add a new vertex connected to $v_{b+2}, \ldots, v_{2b+1}$ in $K_1$ and $u_2, \ldots, u_{b+1}$ in $K_2$.

Add a new vertex connected to $u_2, \ldots, u_{b+1}$ in $K_1$ (this vertex will be called the *left* vertex) and add a new vertex connected to $v_{b+2}, \ldots, v_{2b+1}$ in $K_2$ (called the *right* one). Take $b+1$ disjoint copies of this graph and add $2b+2$ new vertices $x_1, \ldots, x_{2b+2}$ which shall correspond to the occurrences of the variable $x$. We shall call $x_1, \ldots, x_{b+1}$ the *left occurrences* of $x$ and $x_{b+2}, \ldots, x_{2b+2}$ the *right occurrences* of $x$.

Now we shall insert edges between the left occurrences of $x$ and the left vertices so that they induce a $b$-regular bipartite graph with one part being $x_1, \ldots, x_{b+1}$ and the second one being the left vertices. An analogous construction will be done with $x_{b+2}, \ldots, x_{2b+2}$ and the right vertices. See Figure 6.15 for an example.

To complete the construction, in the left copy of each clause gadget, we identify each vertex of the part of the size $2b$ with the respective left occurrences of the variable $x$ and in the right copy of each clause gadget, we identify each vertex of the part of the size $2b$ with the respective right occurrences of the variable $x$. Formally, if $C_i$ is the $j$-th clause containing the variable $x$, we identify $y_x^{i,l}$ with $x_j$ and $y_x^{i,r}$ with $x_{b+1+j}$. The resulting graph shall be called $G$.

We claim that the formula $\phi$ is satisfiable if and only if $G$ has a $(b, b)$-colouring.

First suppose that $\phi$ is satisfiable and take some satisfying assignment $\pi$. We will construct a suitable colouring in the following way. For a variable $x$, if $\pi(x) =$ true, then colour $x_1, \ldots, x_{2b+2}$ by blue colour and otherwise, colour all $x_1, \ldots, x_{2b+2}$ by red colour. colour all vertices in the smaller parts of the left copies of clause gadgets by red colour and all vertices in the smaller parts of the right copies of clause gadgets by blue colour. In the variable gadgets, vertices of one class of bipartition will be coloured the same regardless the value of the corresponding variable while the colouring of the the other class of bipartition will depend on its value. The left vertices (connecting $x_1, \ldots, x_{b+1}$ to $K_1$) will be all coloured blue, the right vertices (connecting $x_{b+2}, \ldots, x_{2b+2}$ to $K_2$) will

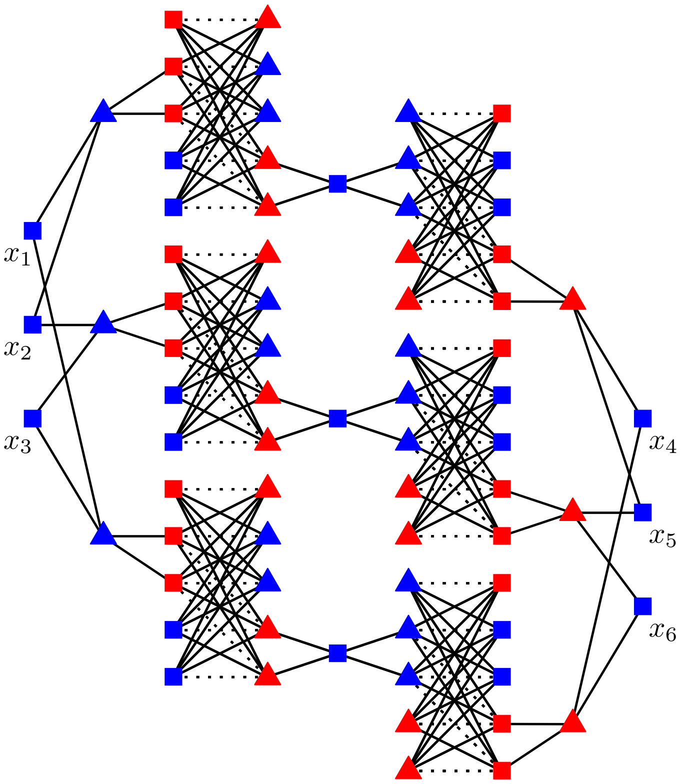


Figure 6.15: A variable gadget $V_2$ for variable $x$ with a $(b,b)$-colouring corresponding to valuation $\pi(x) = \text{true}$.

be all coloured red. The $v_i$'s of $K_1$'s will always be coloured so that $v_1$ and $v_{b+2}, \ldots, v_{2b+1}$ are red and $v_2, \ldots, v_{b+1}$ are blue, the $u_i$'s of $K_2$'s will always be coloured so that $u_1, \ldots, u_{b+1}$ are blue and $u_{b+2}, \ldots, u_{2b+1}$ are red. In the other class of bipartition, if $\pi(x) = \text{true}$, then on top of all the occurrences $x_1, \ldots, x_{2b+2}$, also all the "middle" vertices connecting $K_1$'s to $K_2$'s, the vertices $u_{b+2}, \ldots, u_{2b+1}$ in $K_1$'s and the vertices $v_2, \ldots, v_{b+1}$ in $K_2$'s will be coloured blue, while the vertices $u_1, \ldots, u_{b+1}$ of $K_1$'s and the vertices $v_1, v_{b+2}, \ldots, v_{2b+1}$ in $K_2$'s will be coloured red. If $\pi(x) = \text{false}$, the colours of the vertices in this class of bipartition will be swapped. See an example in Figure 6.15 for a variable evaluated to true. Since in every clause, there are exactly $b$ variables set to true, all vertices in the smaller parts of clause gadgets have exactly $b$ red and exactly $b$ blue neighbours. It can be shown by a detailed case analysis that the same holds for all vertices, and so this is a $(b,b)$-colouring of $G$.

Suppose that $G$ has a $(b,b)$-colouring, and fix one such colouring. For a variable $x$, we set $x$ to be true if all $x_1, \ldots, x_{2b+2}$ are coloured by blue colour and we set it to be false if all $x_1, \ldots, x_{2b+2}$ are coloured by red colour. We need to prove that such assignment always exists and that it is a satisfying assignment.

First we prove that in every $(b,b)$-colouring either all of $x_1, \ldots, x_{2b+2}$ are coloured blue or all of $x_1, \ldots, x_{2b+2}$ are coloured red. Recall the subgraph $K_1$ of a variable gadget with vertices $u_1, \dots, u_{2b+1}$ in one part and $v_1, \dots, v_{2b+1}$ in the other part.

We claim that in every $(b, b)$-colouring of $V_b$ restricted to some copy of $K_1$ and its two adjacent vertices, the vertices $u_2, \dots, u_{b+1}$ are either all red or all blue. Suppose for a contradiction that in some $(b, b)$-colouring there exist indices $i, j \in \{2, \dots, b+1\}$ such that $u_i$ is coloured by red and $u_j$ is coloured by blue. Since $v_1$ is adjacent to all $u_2, \dots, u_{2b+1}$, exactly $b$ of them are coloured red and exactly $b$ of them are coloured blue. Since $v_i$ is not adjacent to $u_i$, we need to colour $u_1$ by red. However, since $v_j$ is not adjacent to $u_j$, we have to colour $u_1$ by blue, a contradiction.

Suppose without loss of generality that all $u_2, \dots, u_{b+1}$ are blue. As argued above, all $u_{b+2}, \dots, u_{2b+1}$ are then red. All of them are neighbours of $v_2$, and hence $u_1$ is blue. Let $w$ be the vertex outside of $K_1$ adjacent to $v_{b+2}, \dots, v_{2b+1}$ in $K_1$. Since $v_{2b+1}$ has only $b-1$ red neighbours in $K_1$, $w$ must be red. Similar arguments apply to $K_2$. Thus, $u_2, \dots, u_{b+1}$ in $K_1$ and $v_{b+1}, \dots, v_{2b+1}$ in $K_2$ always have the same colour. Then all $b$ occurrences of the variable adjacent to the left vertex of $K_1$ and all $b$ occurrences adjacent to the the right vertex of $K_2$ get the same colour. Since $b \geq 2$, it follows from the construction between the occurrences and variable gadgets that all occurrences of the variable have the same colour.

It remains to be proven that this is a satisfying assignment. Since the vertices of the smaller parts of clause gadgets have degree $2b$, exactly $b$ vertices of the bigger part of each clause are coloured by red and exactly $b$ vertices of the bigger part of each clause are coloured by blue. Thus, exactly $b$ variables in each clause are set to be true. This concludes the proof. □

## 6.5 Conclusion

The main goal of this chapter was to initiate the study of the computational complexity of covering graphs with semi-edges. We have exhibited a new level of difficulty that semi-edges bring to coverings by showing a connection to edge-colourings. We have presented a complete classification of the computational complexity of covering graphs with at most two vertices, which is already a quite nontrivial task. In the case of one-vertex target graphs, the problem becomes polynomial-time solvable if the input graph is bipartite, while in the case of two-vertex target graphs, bipartiteness of the input graphs does not help. This provides a strengthening of known results on covering two-vertex graphs without semi-edges.

# Chapter 7

# Complexity of covering disconnected multigraphs

This chapter is based on:

- [29] Jan Bok, Jiří Fiala, Nikola Jedličková, Jan Kratochvíl, and Michaela Seifrtová: *Computational Complexity of Covering Disconnected Multigraphs.* In Fundamentals of Computation Theory, FCT 2021, volume 12867 of Lecture Notes in Computer Science, pages 85–89, 2021.

The notion of graph covers is a discretization of covering spaces introduced and deeply studied in topology. In discrete mathematics and theoretical computer science, they have attained a lot of attention from both the structural and complexity perspectives. Nonetheless, disconnected graphs were usually omitted from the considerations with the explanation that it is sufficient to understand coverings of the connected components of the target graph by components of the source one. However, different (but equivalent) versions of the definition of covers of connected graphs generalize to nonequivalent definitions of disconnected graphs. The aim of this chapter is to summarize this issue and to compare three different approaches to covers of disconnected graphs: 1) locally bijective homomorphisms, 2) globally surjective locally bijective homomorphisms (which we call *surjective covers*), and 3) locally bijective homomorphisms which cover every vertex the same number of times (which we call *equitable covers*). The standpoint of our comparison is the complexity of deciding if an input graph covers a fixed target graph. We show that both surjective and equitable covers satisfy what certainly is a natural and welcome property: covering a disconnected graph is polynomial time decidable if such it is for every connected component of the graph, and it is NP-complete if it is NP-complete for at least one of its components. Despite of this, we argue that the third variant, equitable covers, is the most suitable one when considering covers of coloured (multi)graphs. We conclude the chapter by a complete characterization of the complexity of covering 2-vertex coloured multigraphs with semi-edges.

## 7.1 Motivation and overview of our results

In all the literature devoted to the computational aspects of graph covers, only covers of connected graphs have been considered so far. The authors of [122] justify this by claiming in Fact 2.b that "For a disconnected graph $H$, the $H$-Cover problem is polynomially solvable (NP-complete) if and only if the $H_i$-cover problem is polynomially solvable (NP-complete) for every (for some) connected component $H_i$ of $H$." Though this seems to be a plausible and desirable property, a closer look shows that the validity of this statement depends on the exact definition of covers for disconnected graphs. Namely in the case of multigraphs with semi-edges when the existence of a covering projection does not follow from the existence of a degree-obedient vertex mapping anymore.

The purpose of this chapter is to have a closer look at covers of disconnected graphs in three points of view: the definition, complexity results, and the role of disconnected subgraphs in coloured multigraphs. In Section 7.3 we first discuss what are the possible definitions of covers of disconnected graphs — locally bijective homomorphisms are a natural generalization from the algebraic graph theory standpoint, globally surjective locally bijective homomorphisms (surjective covers) seem to have been understood by the topological graph theory community as the generalization from the standpoint of topological motivation, and a novel and more restrictive definition of equitable covers, in which every vertex of the target graph is required to be covered by the same number of vertices of the source one. The goal of this chapter is to convince the reader that the most appropriate definition is the last one. In Section 7.4 we inspect the three possible definitions under the microscope of computational complexity.

The main result is that the above mentioned Fact 2.b is true for surjective covers, and remains true also for the newly proposed definition of equitable covers of disconnected graphs. The NP-hardness part of the statement is proven for instances when the input graphs are required to be simple. Lastly, in Section 7.5 we review the concept of covers of coloured graphs and show that in this context the notion of equitable covers is indeed the most natural one. We justify our approach by providing a characterization of polynomial/NP-complete instances of the $H$-Cover problem for coloured mixed multigraphs with semi-edges.

We note that, again, we present the results in the utmost generality and strength. We consider (multi)graphs with semi-edges as the inputs for polynomial algorithms, and we prove the NP-completeness results for simple input graphs.

## 7.2 Additional preliminaries

We now introduce a couple of slightly technical definitions in order to be rigorous.

The *multiedge* between $u$ and $v$ is an inclusion-wise maximal subset of links that connect $u$ and $v$, i.e. $\{e \in E : e \cap v \neq \emptyset \wedge v \cap e \neq \emptyset\}$ and the cardinality of this set is the *multiplicity* of the (multi)edge $uv$. In the same way we define the multiplicity of a loop or of a semi-edge.

In case of simple graphs we use also the traditional notation for an edge as $e = uv$ and we write $G = (V, E)$.

A graph $H = (D', V', \Lambda')$ is a *subgraph* of a graph $G = (D, V, \Lambda)$ if their sets of darts satisfy $D' \subseteq D$ and their partitions fulfil $V' = V|_{D'}$ and $\Lambda' = \Lambda|_{D'}$.

A *path* in graph $G$ is a sequence of distinct darts such that consecutive darts either constitute an edge or a vertex of degree 2. A path is *closed* if the first pair as well as the last pair constitute edges. In such a case we say that it connects the vertex containing the first dart to the vertex of the last one. If the first pair and the last pair are vertices then the path is *open*. In all other cases (including a sequence of length 1) the path is *half-way*.

By a *component* of a graph we mean an inclusion-wise maximal induced subgraph such that every two of its vertices are connected by a subgraph isomorphic to a path. We say that a graph is *connected* if it has only a single component.

## 7.3 What is a cover of a disconnected graph?

Throughout this section and the rest of the chapter we assume that we are given two (possibly disconnected) graphs $G$ and $H$, and we are interested in determining whether $G$ covers $H$. In particular, in this section, we discuss what it means that $G$ covers $H$. We assume that $G$ has $p$ components of connectivity, $G_1, G_2, \ldots, G_p$, and $H$ has $q$ components, $H_1, H_2, \ldots, H_q$. It is reasonable to request that a covering projection maps each component of $G$ onto some component of $H$, and this restricted mapping must be a covering. The questions we are raising are:

1. Should the covering projection be globally surjective, i.e., must the preimage of every vertex of $H$ be non-empty?
2. Should the preimages of the vertices of $H$ be of the same size?

Clearly, the "yes" answer to the latter question implies the "yes" answer to the former one. Note also that both these questions are the first ones at hand when

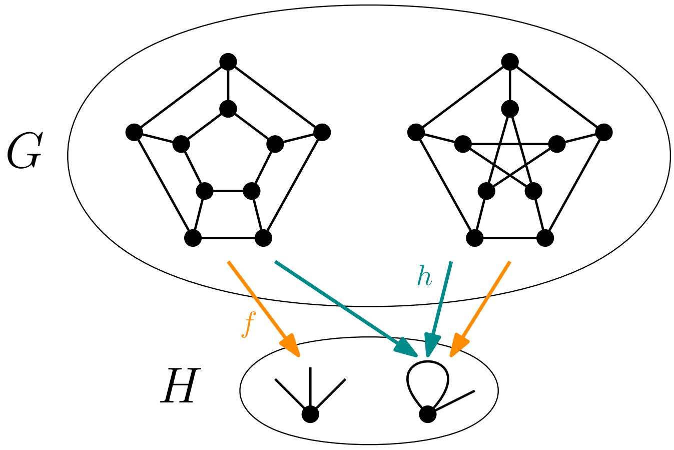


Figure 7.1: An example of differences in definitions of covers of disconnected graphs. The mapping $h\colon G \to H$ is a locally bijective homomorphism, but not a surjective cover, while $f\colon G \to H$ is a locally bijective homomorphism, surjective cover, and also $k$-fold cover.

trying to generalize graph covers to disconnected graphs since the answer is positive in the case of connected graphs (and it is customary to call a projection that covers every vertex $k$ times a *$k$-fold cover*). In the following definition, we identify the variants of the definition of covers of disconnected graphs depending on the outcome of these questions.

**Definition 7.1.** *Let $G$ and $H$ be graphs and let us have a mapping $f\colon G \longrightarrow H$.*

- *We say that $f$ is a* locally bijective homomorphism *of $G$ to $H$ if for each component $G_i$ of $G$, the restricted mapping $F|_{G_i} : G_i \longrightarrow H$ is a covering projection of $G_i$ onto some component of $H$. We write $G \longrightarrow_{lb} H$ if such a mapping exists.*
- *We say that $f$ is a* surjective covering projection *of $G$ to $H$ if for each component $G_i$ of $G$, the restricted mapping $F|_{G_i} : G_i \longrightarrow H$ is a covering projection of $G_i$ onto some component of $H$, and $f$ is surjective. We write $G \longrightarrow_{sur} H$ if such a mapping exists.*
- *We say that $f$ is an* equitable covering projection *of $G$ to $H$ if for each component $G_i$ of $G$, the restricted mapping $F|_{G_i} : G_i \longrightarrow H$ is a covering projection of $G_i$ onto some component of $H$, and for every two vertices $u, v \in V(H)$, $|f^{-1}(u)| = |f^{-1}(v)|$. We write $G \longrightarrow_{equit} H$ if such a mapping exists.*

See Figure 7.1 for an illustration of differences between the three variants.

A useful tool both for describing and discussing the variants, as well as for algorithmic considerations, is introduced in the following definition.

**Definition 7.2.** *Given graphs $G$ and $H$ with components of connectivity $G_1, G_2, \ldots, G_p$, and $H_1, H_2, \ldots, H_q$, respectively, the* covering pattern *of the pair $G, H$ is the weighted bipartite graph* $\mathrm{Cov}(G,H) = (\{g_1, g_2, \ldots, g_p, h_1, h_2, \ldots, h_q\}, \{g_ih_j : G_i \longrightarrow H_j\})$ *with edge weights* $r_{ij} = r(g_ih_j) = \frac{|V(G_i)|}{|V(H_j)|}$.

The following observation follows directly from the definitions, but will be useful in the computational complexity considerations.

**Observation 7.3.** *Let $G$ and $H$ be graphs. Then the following holds.*

- *We have $G \longrightarrow_{lb} H$ if and only if the degree of every vertex $g_i, i = 1, 2, \ldots, p$ in* $\mathrm{Cov}(G,H)$ *is greater than zero.*
- *We have $G \longrightarrow_{sur} H$ if and only if the degree of every vertex $g_i, i = 1, 2, \ldots, p$ in* $\mathrm{Cov}(G,H)$ *is greater than zero and* $\mathrm{Cov}(G,H)$ *has a matching of size $q$.*
- *We have $G \longrightarrow_{equit} H$ if and only if* $\mathrm{Cov}(G,H)$ *has a spanning subgraph* $\mathrm{Map}(G,H)$ *such that every vertex $g_i, i = 1, 2, \ldots, p$ has degree 1 in* $\mathrm{Map}(G,H)$ *and for every vertex $h_j$ of* $\mathrm{Cov}(G,H)$,

$$\sum_{i: g_ih_j \in E(\mathrm{Map}(G,H))} r_{ij} = k, \text{ where } k = \frac{|V(G)|}{|V(H)|}.$$

## 7.4 Complexity results

It would be a desirable situation if $H$-Cover is polynomial-time solvable whenever $H_i$-Cover is polynomial-time solvable for every component $H_i$ of $H$, while $H$-Cover is NP-complete whenever $H_i$-Cover is NP-complete for some component $H_i$ of $H$. This is the point of view under which we will inspect the three possible definitions of covers of disconnected graphs that we have introduced in the previous section.

To strengthen the results, we again allow arbitrary input graphs (i.e., with multiple edges, loops and/or semi-edges) when considering polynomial-time algorithms, while we restrict the inputs to simple graphs when we aim at NP-hardness results. In some cases we are able to prove results also from the fixed-parameter tractability standpoint. In those cases we consider both the source and the target graphs to be a part of the input, and the parameter is typically the maximum size of a component of the target one.

As the first step, we provide the following lemma. Note that though we are mostly interested in the time complexity of deciding $G \longrightarrow H$ for a fixed graph $H$ and input graph $G$, this lemma assumes both the source and the target graphs to be part of the input. The size of the input is measured by the number of edges.

**Lemma 7.4.** *Let $\varphi(A, B)$ be the best running time of an algorithm deciding if $A \longrightarrow B$ for connected graphs $A$ and $B$, and let $\varphi(n, B)$ be the worst case of $\varphi(A, B)$ over all connected graphs of size $n$. Then for given input graphs $G$ and $H$ with components of connectivity $G_1, G_2, \dots, G_p$, and $H_1, H_2, \dots, H_q$, respectively, the covering pattern $\mathrm{Cov}(G, H)$ can be constructed in time*

$$O(pq \cdot \max_{j=1}^{q} \varphi(n, H_j)) = O(n^2 \cdot \max_{j=1}^{q} \varphi(n, H_j)),$$

*where $n$ is the input size, i.e., the sum of the numbers of edges in $G$ and $H$.* □

**Corollary 7.5.** *Constructing the covering pattern of input graphs $G$ and $H$ is in the complexity class XP when parameterized by the maximum size of a component of the target graph $H$, provided the $H_j$-*Cover *problem is polynomial-time solvable for every component $H_j$ of $H$.*

*Proof.* Suppose the size of every component of $H$ is bounded by $M$. Let $\mathcal{P}_M$ be the class of all connected graphs $B$ of size at most $M$ such that $B$-Cover is decidable in polynomial time. By the assumption, every component $H_j$ of $H$ belongs to $\mathcal{P}_M$. The class $\mathcal{P}_M$ is finite (its size depends on $M$), and so there are well defined positive integers $K_M, t_M$ such that $\varphi(n, B) \leq K_M \cdot n^{t_M}$ for every $B \in \mathcal{P}_M$. Hence $\varphi(n, H_j) \leq K_M \cdot n^{t_M}$ for every $j = 1, 2, \dots, q$, and $\mathrm{Cov}(G, H)$ can be constructed in time $O(n^2 \cdot n^{t_M})$ by the preceding lemma. □

**Corollary 7.6.** *The covering pattern of input graphs $G$ and $H$ can be constructed in polynomial time provided all components of $H$ have bounded size and the $H_j$-*Cover *problem is solvable in polynomial time for every component $H_j$ of $H$.* □

In the following subsections, we discuss and compare the computational complexity of deciding the existence of locally bijective homomorphisms, surjective covers, and equitable covers. The corresponding decision problems are denoted by LBHom, SurjectiveCover, and EquitableCover. If the target graph is fixed to be $H$, we write $H$-LBHom (and analogously for the other variants).

### 7.4.1 Locally bijective homomorphisms

Though the notion of locally bijective homomorphisms is seemingly the most straightforward generalization of the fact that in a graph covering projection to a connected graph "the closed neighbourhood of every vertex of the source graph is mapped bijectively to the closed neighbourhood of its image". We show in this subsection that it does not behave as we would like to see it from the computational complexity perspective. Proposition 7.10 shows that there are infinitely many graphs $H$ with only two components each such that $H$-LBHom is polynomial-time solvable, while $H_i$-LBHom is NP-complete for one component $H_i$ of $H$. The polynomial part of the desired properties is, however, fulfilled, even in some cases when both graphs are part of the input:

**Theorem 7.7.** *If $H_i$-*Cover *is polynomial-time solvable for every component $H_i$ of $H$, then*

*i) the $H$-*LBHom *problem is polynomial-time solvable,*

*ii) the* LBHom *problem is in XP when parameterized by the maximum size of a component of the target graph $H$,*

*iii) the* LBHom *problem is solvable in polynomial time, provided the components of $H$ have bounded size.*

*Proof.* i) If $H$ is fixed, it has by itself bounded size, and thus the covering pattern $\mathrm{Cov}(G, H)$ can be constructed in polynomial time by Corollary 7.6. As noted in Observation 7.3, $G \longrightarrow_{lb} H$ if and only if $\deg_{\mathrm{Cov}(G,H)} g_i \geq 1$ for all $i = 1, 2, \ldots, p$, which certainly can be checked in polynomial time, once $\mathrm{Cov}(G, H)$ has been constructed.

ii) Deciding if $G$ allows a locally bijective homomorphism into $H$ is not harder than constructing the covering pattern $\mathrm{Cov}(G, H)$, and this task is in XP when parameterized by the maximum size of a component of the target graph $H$, as shown in Corollary 7.5.

iii) Follows straightforwardly from ii). □

However, it is not true that $H$-LBHom is NP-complete whenever $H_j$-Cover is NP-complete for some component $H_j$ of $H$. Infinitely many examples can be constructed by means of the following proposition. These examples provide another argument for our opinion that the notion of locally bijective homomorphism is not the right generalization of graph covering to disconnected graphs.

**Proposition 7.8.** *Let $H_1 \longrightarrow H_2$ for components $H_1, H_2$ of $H = H_1 + H_2$, and suppose that $H_2$-*Cover *is polynomial-time solvable. Then $H$-*LBHom *is polynomial-time solvable regardless the complexity of $H_1$-*Cover*.*

*Proof.* Under the assumption $H_1 \longrightarrow H_2$, any input graph $G$ allows a locally bijective homomorphism to $H$ if and only if each of its components covers $H_2$. On one hand, if each component of $G$ allows a locally bijective homomorphism to $H_2$, the union of these mappings is a locally bijective homomorphism of $G$ into $H$. On the other hand, if $G$ allows a locally bijective homomorphism into $H$, each component of $G$ covers $H_1$ or $H_2$. However, every component that covers $H_1$ also covers $H_2$. □

There are many examples of pairs of connected graphs $H_1, H_2$ such that $H_1$ covers $H_2$, $H_1$-Cover is NP-complete and $H_2$-Cover is solvable in polynomial time. In a certain sense it is more interesting that a similar phenomenon as in Proposition 7.8 may occur even when $H_1$ and $H_2$ are incomparable by covering.

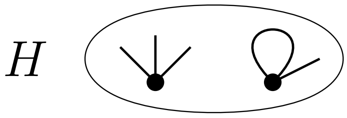


Figure 7.2: The graph $H = H_1 + H_2$, where $H_1 = F(3,0)$ and $H_2 = F(1,1)$.

*Example.* For graphs $H_1 = F(3,0)$ and $H_2 = F(1,1)$, neither $H_1$ covers $H_2$ nor $H_2$ covers $H_1$, yet for their disjoint union $H = H_1 + H_2$, $H$-LBHom is polynomial-time solvable for simple input graphs, while $H_1$-Cover is NP-complete for simple input graphs. (The graph $H$ is depicted in Figure 7.2.)

*Proof.* A connected simple graph covers $F(3,0)$ if and only if it is cubic and is 3-edge-colourable, in which case it also covers $F(1,1)$ (edges of any two colours form a disjoint union of cycles, which itself covers the loop of $F(1,1)$). Hence a simple graph allows a locally bijective homomorphism to $F(3,0)+F(1,1)$ if and only if each of its components covers $F(1,1)$, which can be decided in polynomial time (a connected graph covers $F(1,1)$ if and only if it is cubic and contains a perfect matching). □

This example is a concrete instance of a more general pattern. Graph covering is a transitive relation among connected graphs. Thus when $A \longrightarrow B$ for graphs $A$ and $B$, every graph $G$ that covers $A$ also covers $B$. Surprisingly, the conclusion may hold true also in cases when $A$ does not cover $B$, if we consider only simple graphs $G$. To describe this phenomenon, we introduce the following definition, which will prove useful in several reductions later on.

**Definition 7.9.** *Given connected graphs $A, B$, we say that $A$* is stronger than *$B$, and write $A \triangleright B$, if every simple graph that covers $A$ also covers $B$.*

The smallest nontrivial example of such a pair of graphs are two one-vertex graphs: $F(2,0)$ with a pair of semi-edges and $F(0,1)$, one vertex with a loop. While $F(0,1)$ is covered by any cycle, only cycles of even length cover $F(2,0)$. So $F(2,0) \triangleright F(0,1)$.

Observe that whenever $A$ is simple, then $(A \triangleright B)$ if and only if $(A \longrightarrow B)$.[1]

In the following proposition, the operator $+$ represents the operation of disjoint union of two graphs.

[1] One might also notice that $\triangleright$ defines a quasi-order on connected graphs. Many pairs of graphs are left incomparable with respect to this relation, even those covering a common target graph. On the other hand, the equivalence classes of pair-wise comparable graphs may be nontrivial, and the graphs within one class might have different numbers of vertices. For example, $W(0,0,2,0,0)$ and $F(2,0)$ form an equivalence class of $\triangleright$, as for both of these graphs, the class of simple graphs covering them is exactly the class of even cycles.

**Proposition 7.10.** *Let $H = H_1 + H_2$ for connected graphs $H_1$ and $H_2$ such that $H_1 \triangleright H_2$. Then $H$-*LBHom *for simple input graphs is polynomially reducible to $H_2$-*LBHom *for simple input graphs. In particular, if $H_2$-*LBHom *is polynomial-time decidable, then so is $H$-*LBHom *as well.*

*Proof.* Every component of the input graph, which is assumed to be simple, allows a locally bijective homomorphism into $H$ if and only if it covers $H_2$, by the assumption $H_1 \triangleright H_2$. Hence for a simple graph $G$, we have $G \longrightarrow_{lb} H$ if and only if $G \longrightarrow_{lb} H_2$. □

### 7.4.2 Surjective covers

The notion of surjective covers is favoured by topologists since it captures the fact that every vertex (point) of the target graph (space) is covered. The advantage of this notion for us is that it behaves nicely from the point of view of computational complexity. This is captured by the following theorem.

**Theorem 7.11.** *If $H_i$-*Cover *is polynomial-time solvable for every component $H_i$ of $H$, then*

*(i) the $H$-*SurjectiveCover *problem is polynomial-time solvable,*

*(ii) the* SurjectiveCover *problem is in XP when parameterized by the maximum size of a component of the target graph $H$, and*

*(iii) the* SurjectiveCover *problem is solvable in polynomial time if the components of $H$ have bounded size.*

*Proof.*

(i) If $H$ is fixed, the covering pattern $\mathrm{Cov}(G, H)$ can be constructed in polynomial time by Corollary 7.6. As noted in Observation 7.3, $G \longrightarrow_{sur} H$ if and only if $\deg_{\mathrm{Cov}(G,H)} g_i \geq 1$ for all $i = 1, 2, \ldots, p$ and $\mathrm{Cov}(G, H)$ has a matching of size $q$, which can be checked in polynomial time, once $\mathrm{Cov}(G, H)$ has been constructed (e.g., by network flow algorithms).

(ii) Again we construct the covering pattern $\mathrm{Cov}(G, H)$, which task is in XP when parameterized by the maximum size of a component of the target graph $H$, as shown in Corollary 7.5. Checking the degrees of $\mathrm{Cov}(G, H)$ as well as checking if $\mathrm{Cov}(G, H)$ has a matching of size $q$ can be done in time polynomial in $p + q$ and hence also in the size of the input.

(iii) Follows straightforwardly from (ii). □

For surjective covers, the NP-hardness of the problem of deciding if there is a covering of one component of $H$ propagates to NP-hardness of deciding if there is a surjective covering of entire $H$, even when our attention is restricted to simple input graphs.

**Theorem 7.12.** *The $H$-*SurjectiveCover *problem is NP-complete for simple input graphs if $H_i$-*Cover *is NP-complete for simple input graphs for at least one component $H_i$ of $H$.*

*Proof.* Without loss of generality suppose that $H_1$-Cover is NP-complete for simple input graphs. Let $G_1$ be a simple graph for which $G_1 \longrightarrow H_1$ is to be tested. We show that there exists a polynomial-time reduction from $H_1$-Cover to $H$-SurjectiveCover. For every $j = 2, \dots, q$, fix a simple connected graph $G_j$ that covers $H_j$ such that $G_j \longrightarrow H_1$ if and only if $H_j \triangleright H_1$ (in other words, $G_j$ is a witness which does not cover $H_1$ when $H_j$ is not stronger than $H_1$). Note that the size of each $G_j$, $j = 2, \dots, q$, is a constant which does not depend on the size of the input graph $G_1$. Note also, that since $H$ is a fixed graph, we do not check algorithmically whether $H_j \triangleright H_1$ when picking $G_j$. We are only proving the existence of a reduction, and for this we may assume the relation $H_j \triangleright H_1$ to be given by a table.

Let $G$ be the disjoint union of $G_j$, $j = 1, \dots, q$. We claim that $G \longrightarrow_{sur} H$ if and only if $G_1 \longrightarrow H_1$. The "if" part is clear. We map $G_j$ onto $H_j$ for every $j = 1, 2, \dots, q$ by the covering projections that are assumed to exist. Their union is a surjective covering projection of $G$ to $H$.

For the "only if" direction, suppose that $f \colon V(G) \longrightarrow V(H)$ is a surjective covering projection. Since $f$ must be globally surjective and $G$ and $H$ have the same number of components, namely $q$, different components of $G$ are mapped onto different components of $H$ by $f$. Define $\widetilde{f} \in Sym(q)$ by setting $\widetilde{f}(i) = j$ if and only if $f$ maps $G_i$ onto $H_j$. Then $\widetilde{f}$ is a permutation of $\{1, 2, \dots, q\}$. Consider the cycle containing 1. Let it be $(i_1 = 1, i_2, i_3, \dots, i_t)$, which means that $G_{i_j} \longrightarrow H_{i_{j+1}}$ for $j = 1, 2, \dots, t-1$, and $G_{i_t} \longrightarrow H_{i_1}$. By reverse induction on $j$, from $j = t$ down to $j = 2$, we prove that $H_j \triangleright H_1$. Indeed, for $j = t$, $G_{i_t} \longrightarrow H_1$ means that $H_{i_t}$ is stronger than $H_1$, since we would have set $G_{i_t}$ as a witness that does not cover $H_1$ if it were not. For the inductive step, assume that $H_{i_{j+1}} \triangleright H_1$ and consider $G_{i_j}$. Now $G_{i_j}$ covers $H_{i_{j+1}}$ since $\widetilde{f}(i_j) = i_{j+1}$. Because $G_{i_j}$ is a simple graph and $H_{i_{j+1}}$ is stronger than $H_1$, this implies that $G_{i_j} \longrightarrow H_1$. But then $H_{i_j}$ must itself be stronger than $H_1$, otherwise we would have set $G_{i_j}$ as a witness that does not cover $H_1$. We conclude that $H_{i_2} \triangleright H_1$, and hence $G_1 \longrightarrow H_1$ follows from the fact that the simple graph $G_1$ covers $H_{i_2}$. □

### 7.4.3 Equitable covers

As already announced, we wish to argue that equitable covers are the right generalisation of covers to disconnected graphs. Not only that they capture the crucial properties of covers, but they also behave nicely from the point of view of computational complexity.

**Theorem 7.13.** *The $H$-*EQUITABLECOVER *problem is polynomial-time solvable if $H_i$-*COVER *is polynomial-time solvable for every component $H_i$ of $H$.*

*Proof.* First construct the covering pattern $\text{Cov}(G, H)$. Since $H$ is a fixed graph, this can be done in time polynomial in the size of the input, i.e., $G$, as it follows from Corollary 7.6.

Using dynamic programming, fill in a table $M(s, k_1, k_2, \ldots, k_q)$, $s = 0, 1, \ldots, p$, $k_j = 0, 1, \ldots, k = \frac{|V(G)|}{|V(H)|}$ for $j = 1, 2, \ldots, q$, with values true and false. Its meaning is that $M(s, k_1, k_2, \ldots, k_q) =$ true if and only if $G_1 \cup G_2 \cup \ldots \cup G_s$ allows a locally bijective homomorphism $f$ to $H$ such that for every $j$ and every $u \in V(H_j)$, $|f^{-1}(u)| = k_j$. The table is initialized by setting

$$M(0, k_1, \ldots, k_q) = \begin{cases} \text{true}, & \text{if } k_1 = k_2 = \ldots = k_q = 0 \\ \text{false}, & \text{otherwise.} \end{cases}$$

In the inductive step assume that all values for some $s$ are filled in correctly, and move on to $s + 1$. For every edge $g_{s+1}h_j$ of $\text{Cov}(G, H)$ and every $q$-tuple $k_1, k_2, \ldots, k_q$ such that $M(s, k_1, k_2, \ldots, k_q) =$ true, set $M(s + 1, k_1, k_2, \ldots, k_j + r_{s+1,j}, \ldots, k_q) =$ true, provided $k_j + r_{s+1,j} \leq k$. Clearly, the loop invariant is fulfilled, and hence $G$ is a $k$-fold (equitable) cover of $H$ if and only if $M(p, k, k, \ldots, k)$ is set to true.

The table $M$ has $(p+1) \cdot (k+1)^q = O(n^{q+1})$ entries and the inductive step changes $O((k + 1)^q \cdot q$ values. So processing the table can be concluded in $O((k + 1)^q(1 + pq) = O(n^{q+1})$ steps. □

We are currently unaware of how to avoid $q$ in the exponent if both $G$ and $H$ are part of the input. We provide a simpler result.

**Proposition 7.14.** *The* EQUITABLECOVER *problem is in XP when parameterized by the number $q$ of components of $H$ plus the maximum size of a component of the target graph $H$, provided $H_i$-*COVER *is polynomial-time solvable for every component $H_i$ of $H$.*

*Proof.* The algorithm as described in Theorem 7.13 is in XP when parameterized by the number $q$ of components of $H$ (needed for processing the table $M$) plus the maximum size of a component of $H$ (needed for computing the covering pattern). □

**Problem 7.15.** *Is the* EQUITABLECOVER *problem in XP when parameterized by the maximum size of a component of the target graph $H$, provided each $H_i$-*COVER *is polynomial-time solvable for every component $H_i$ of $H$?*

The NP-hardness theorem holds true as well:

**Theorem 7.16.** *The $H$-*EquitableCover *problem is NP-complete for simple input graphs if $H_i$-*Cover *is NP-complete for simple input graphs for at least one component $H_i$ of $H$.*

*Proof.* We proceed in a similar way as in the proof of Theorem 7.12. Suppose without loss of generality that $H_1$-Cover is NP-complete. We show that there exists a polynomial-time reduction from $H_1$-Cover to $H$-EquitableCover. For every $j = 2, \ldots, q$, fix a simple connected graph $G_j$ that covers $H_j$ such that $G_j \longrightarrow H_1$ if and only if $H_j \rhd H_1$ (in other words, $G_j$ is the witness which does not cover $H_1$ when $H_j$ is not stronger than $H_1$). For every $j = 2, \ldots, q$, we have integers $k_j = \frac{|V(G_j)|}{|V(H_j)|}$ which are constants independent of $G_1$.

Now suppose we are given a simple graph $G_1$ whose covering of $H_1$ is to be tested. Compute $k = \frac{|V(G_1)|}{|V(H_1)|}$, which can be done in time polynomial in the size of $G_1$. This $k$ should be an integer, since otherwise we conclude right away that $G_1$ does not cover $H_1$. Set $K$ to be the least common multiple of $k, k_2, \ldots, k_q$. Define $G$ to be the disjoint union of $\frac{K}{k}$ copies of $G_1$ with $\frac{K}{k_j}$ copies of $G_j$ for all $j = 2, \ldots, q$. (Note that the number of connected components of $G$ is $p = \frac{K}{k} + \sum_{j=2}^{q} \frac{K}{k_j}$.) We claim that $G_1$ covers $H_1$ if and only if $G$ equitably covers $H$, and in that case $G$ is a $K$-fold cover of $H$.

The "only if" part is clear. We map each copy of $G_j$ onto $H_j$ for every $j = 1, 2, \ldots, q$ by the covering projections that are assumed to exist. Their union is a surjective covering projection of $G$ to $H$. To show that this is an equitable covering projection, we do just a little bit of counting. Since $G_j$ is a $k_j$-fold cover of $H_j$ (here and in the sequel, we write $k_1 = k$) and we have $\frac{K}{k_j}$ copies of $G_j$ in $G$, the preimage of each vertex of $H$ in this mapping has size $K$.

For the "if" part, assume that $f \colon G \longrightarrow H$ is a $K$-fold covering projection. Every connected component of $G$ must map onto one connected component of $H$, but it may happen that different copies of the same $G_j$ map onto different components of $H$. Still, we can again find a sequence of indices $i_1 = 1, i_2, \ldots, i_t$ such that for every $j = 1, 2, \ldots, t-1$, some copy of $G_{i_j}$ is mapped to $H_{i_{j+1}}$ by $f$, and some copy of $G_{i_t}$ is mapped onto $H_1$. If some copy of $G_1$ is mapped onto $H_1$, then $t = 1$ and $G_1$ covers $H_1$. Suppose this is not the case. Let $S \subseteq \{1, 2, \ldots, q\}$ and a set $\mathcal{S}$ of components of $G$ be inclusion-wise minimal such that:

*a)* all copies of $G_1$ are in $\mathcal{S}$,

*b)* if a component from $\mathcal{S}$ is mapped onto $H_j$ by $f$, then $j \in S$, and

*c)* if $j \in S$, then all copies of $G_j$ are in $\mathcal{S}$.

The sets $S$ and $\mathcal{S}$ are uniquely defined by application of the rules a), b), and c). It follows that if $1 \in S$, then a sequence $i_1 = 1, i_2, \ldots, i_t$ exists. If $1 \notin S$, then $f$ restricted to $\mathcal{S}$ is a surjective cover of the disjoint union of $H_j, j \in S$. But it cannot be a $K$-fold cover, because the union of the components in $\mathcal{S}$ has $\frac{K}{k}|V(G_1)| + \sum_{j \in S} \frac{K}{k_j}|V(G_j)| = K \cdot |V(H_1)| + K \sum_{j \in S} |V(H_j)|$ vertices, while $\bigcup_{j \in S} H_j$ has $\sum_{j \in S} |V(H_j)|$ vertices. This concludes the proof. □

## 7.5 Covering coloured two-vertex graphs

In this section we introduce the last generalization and consider coverings of graphs which come with links and vertices equipped with additional information, which we simply refer to as a colour. The requirement is that the covering projection respects the colours, both on the vertices and on the links. This generalization is not purposeless as it may seem. It is shown in Kratochvíl et al. [123] that to fully characterize the complexity of $H$-Cover for simple graphs $H$, it is necessary and suffices to understand the complexity of $H$-Cover for coloured mixed multigraphs of valency greater than 2. The requirement on the minimum degree of $H$ gives hope that the characterization can be more easily described. We will first describe the concept of covers of coloured graphs with semi-edges in detail in Subsection 7.5.1, where we also give our final argument in favour of equitable covers. Then we extend the characterization of the computational complexity of covering coloured 2-vertex graphs without semi-edges presented in [123] to general graphs in Subsection 7.5.2.

### 7.5.1 Covers of coloured graphs

**Definition 7.17.** *We say that a graph $G$ is* coloured*, if it is equipped with a function $c\colon D \cup V \to \mathbb{N}$.*

*A coloured graph covers a coloured graph $H$ if $G$ covers $H$ via a mapping $f$ and this mapping respects the colours, i.e., $c_G = c_H \circ f$ on $D$ and every $u \in V_G$ satisfies $c_G(u) = c_H(f(u))$.*

Note that one may assume without loss of generality that all vertices are of the same colour, since we can add the colour of a vertex as a shade to the colours of its darts. However, for the reductions described below, it is convenient to keep the intermediate step of colouring vertices as well.

The final argument that equitable covers are the most proper generalization to disconnected graphs is given by the following observation. (Note that colour-induced subgraphs of a connected graph may be disconnected.)

**Observation 7.18.** *Let a coloured graph $H$ be connected and let $f : G \longrightarrow H$ be a colour preserving mapping that respects the links (i.e., for every link $e \in \Lambda_G$, there is a link $e' \in \Lambda_H$ such that $f(e) = e'$). Then $f$ is a covering projection if and only if $f : G_{i,j} \longrightarrow H_{i,j}$ is an equitable covering projection for every two (not* *necessarily distinct) colours $i, j$, where $G_{i,j}, H_{i,j}$ denote the subgraphs of $G$ and $H$ induced by the links $e$ such that $c(e) = \{i, j\}$.* □

Kratochvíl et al. [123] proved that the existence of a covering between two (simple) graphs can be reduced to the existence of a covering between two coloured graphs of minimum degree three. Their concept of *coloured directed multigraph* is equivalent to our concept of coloured graphs (without semi-edges), namely:

- The vertex colour encoding the collection of trees (without semi-edges) stemming from a vertex is encoded as the vertex colour in the exactly same way.
- The link colour encoding a subgraph isomorphic to coloured induced path between two vertices of degree at least three is encoded as the pair of colours of the edge or a loop that is used for the replacement of the path.
- When the path colouring is symmetric, we use the same colour twice for the darts of the replaced arc which could be viewed as an undirected edge of the construction of [123].
- On the other hand, when the colouring is not symmetric and the replaced arc hence needed to be directed in [123], we use a pair of distinct colours on the two darts, which naturally represents the direction.

When semi-edges are allowed we must take into account one more possibility. The colour used on the two darts representing a symmetric coloured path of even number of vertices may be used also to represent a half-way path with the identical colour pattern ended by a semi-edge. A formal description follows:

By a *pattern* $P$ we mean a finite sequence of positive integers $(p_1, \ldots, p_k)$. A pattern is symmetric if $p_i = p_{k+1-i}$, and the reverse pattern is $\overline{P} = (p_k, \ldots, p_1)$.

The pattern of a closed path $d_1, \ldots, d_{2k}$ in a coloured graph $G$ is the sequence of colours $c(u_0), c(d_1), c(d_2), c(u_1), c(d_3), c(d_4), c(u_2), \ldots, c(d_{2k}), c(u_k)$, where $\forall i \in \{1, \ldots, k\}$ the vertex $u_i$ contains the dart $d_{2i}$, and in addition $u_0$ contains the dart $d_1$. Analogously we define patterns of open and half-way paths — the sequence of dart colours is augmented by the vertex colours.

Now, a half-way path of pattern $P$ that starts in a vertex of degree 3 and ends by a semi-edge will be replaced by a single dart whose colour is identical to those used for the two darts used for the replacement of closed paths whose pattern is the concatenation $P\overline{P}$, see Figure 7.3.

### 7.5.2 Two-vertex graphs

We say that a coloured graph $G$ is *regular* if all its vertices have the same colour and for every $i \in \mathbb{N}$ all vertices are incident with the same number of darts of colour $i$. Kratochvíl et al. [123] completely characterized the computational

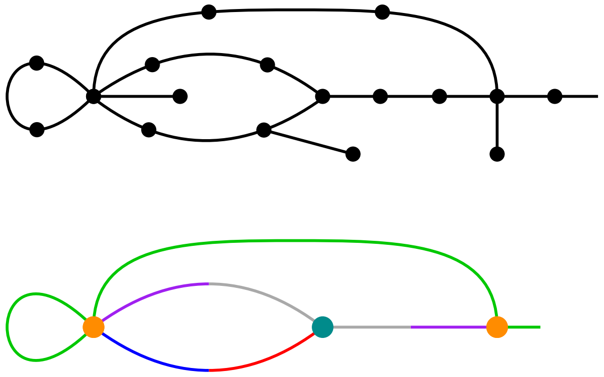


Figure 7.3: Reduction of a graph to a coloured graph of minimum degree 3. Distinct colours represent distinct integers.

complexity of the $H$-Cover problem on coloured graphs on at most two vertices without semi-edges. Their result implies the following:

**Proposition 7.19.** *Let $H$ be a connected coloured graph on at most two vertices without semi-edges. The $H$-*Cover *problem is polynomially solvable if:*

1. *the graph $H$ contains only one vertex, or*
2. *$H$ is not regular, or*
3. (a) *for every colour $i \in \mathbb{N}$, the $H_i$-*EquitableCover *problem is solvable in polynomial time, where $H_i$ is the coloured subgraph of $H$ induced by the links coloured by $i$, and*

   (b) *for every pair of colours $i, j \in \mathbb{N}$, the $H_{i,j}$-*EquitableCover *problem is solvable in polynomial time, where $H_{i,j}$ is the coloured subgraph of $H$ induced by the links $l \in \Lambda$ such that $c(l) = \{i, j\}$.*

*Otherwise, the $H$-*Cover *problem is NP-complete.*

Informally, the NP-completeness persists if and only if $H$ has two vertices which have the same degree in every colour, and the NP-completeness appears on a monochromatic subgraph (either undirected or directed). Such a subgraph must contain both vertices and be connected. We extend this characterization to include semi-edges as well:

**Theorem 7.20.** *Let $H$ be a coloured graph on at most two vertices. The $H$-*EquitableCover *problem is polynomially solvable if:*

1. *The graph $H$ contains only one vertex and for every $i$, $H_i$-*Cover *is solvable in polynomial time, where $H_i$ is the subgraph of $H$ induced by the loops and semi-edges coloured by $i$, or*
2. *$H$ is not regular and for every $i$ and each vertex $u \in V_H$, the $H_i^u$-*Cover

*problem is solvable in polynomial time, where $H_i^u$ is the coloured subgraph of $H$ induced by the loops and semi-edges incident with $u$ coloured by $i$, or*

3. *$H$ is regular graph on two vertices and*

   (a) *for every colour $i \in \mathbb{N}$, the $H_i$-*EquitableCover *problem is solvable in polynomial time, where $H_i$ is the coloured subgraph of $H$ induced by the links coloured by $i$, and*

   (b) *for every pair of colours $i, j \in \mathbb{N}$, the $H_{i,j}$-*EquitableCover *problem is solvable in polynomial time, where $H_{i,j}$ is the subgraph of $H$ induced by the links $l \in \Lambda$ such that $c(l) = \{i, j\}$.*

*Otherwise, the $H$-*EquitableCover *problem is NP-complete.*

Observe that the cases when every colour induces in $H$ a subgraph without semi-edges are covered by Proposition 7.19.

*Proof.* We discuss first the polynomial cases:

1. We accept if and only if all $H_i$-Cover problems accept the input, which is the restriction of $G$ to links coloured by $i$.

   In such admissible case, the overall covering projection $f\colon G \to H$ is the union of all partial covering projections $G_i \to H_i$. Since the derived vertex map $V(G_i) \to V(H_i) = \{u\}$ is uniquely given as the target graph has only one vertex, these partial maps fit together neatly to a bijection between any $w \in V(G)$ and $f(w) \in V(H)$.

2. We proceed analogously to the previous case. The only difference is that $H$ has two vertices $v$ and $w$, which could be distinguished:

   (a) by their vertex colour, and/or

   (b) by the number of incident darts of some colour $i$.

   We perform the same separation on the vertices of $G$ into sets $V_v$ and $V_w$. Namely, $V_v$ contains those vertices of $G$ that have the same colour as $v$ and the same number of incident darts of every colour as $v$ and analogously for $V_w$. In particular we reject the input if $V_v \cup V_w \neq V(G)$.

   We obtain a vertex mapping $V_G \to V_H$ by mapping entire $V_v$ to $v$ and $V_w$ to $w$.

   The edges between $v$ and $w$ can be covered only by edges between $V_v$ and $V_w$.

   When $v$ and $w$ are connected by a multiedge of of colour $i$ and multiplicity $k$ then a covering may exists if and only if edges of colour $i$ between $V_v$ and $V_w$ induce a $k$-regular subgraph. This necessary condition is also sufficient as every $k$-regular bipartite graph can be split into $k$ perfect matchings and these yield the dart mapping of a cover.

If we did not reject the input earlier, we solve the $H_i^v$-Cover and $H_i^w$-Cover problems, where the input for $H_i^v$-Cover is induced by all links coloured by $i$ that have its single end or both ends in $V_v$; analogously for $H_i^w$-Cover.

In an admissible case, the overall covering projection $f\colon G \to H$ is the union of covering projections $G_i \to H_i^v$ and $G_i \to H_i^w$ together with the mapping of edges connecting $V_v$ and $V_w$ and therefore also a feasible solution for $H_i$-LBHom problem.

Observe that in the case when $H$ is connected we may accept $G$ also for $H$-Cover, $H$-SurjectiveCover and $H$-EquitableCover too.

For disconnected $H$, we shall also test whether $V_v$ and $V_w$ are non-empty to accept $G$ for $H$-SurjectiveCover. Analogously, $|V_v| = |V_w|$ is needed for $H$-EquitableCover.

3. Let $V(H) = \{v, w\}$. For every dart $d \in D(G)$, we introduce a Boolean variable $x_d$. Based on the structure of $G$ and $H$ we compose a formula $\varphi$ in CNF with clauses of size two which will allow a satisfying assignment if and only if $G$ covers $H$. In such a satisfying assignment $x_d$ is truly evaluated if for some covering $f\colon G \to H$ the dart $d$ maps within $v$ and $x_d$ is negative if $f(d) \in w$.

   Clearly, all variables corresponding to darts of a vertex must be assigned the same value, which can be guaranteed by a collection of 2-SAT clauses.

   The constitution of $\varphi$ relies on the characterization of polynomially solvable cases of the $H$-Cover problem for symmetric graphs with two vertices given by Kratochvíl et al. [123] and Chapter 6. For the sake of coherence, we provide here a brief summary of this approach:

   - When $H_i$ is disconnected then for every link $(d, e) \in \Lambda(G_i)$ we add two 2-SAT clauses to express $x_d = x_e$.

     Moreover, when $H_i$ contains a semi-edge, we test every component of $G_i$ whether it allows a 1-factor that contains all semi-edges (or in other words whether the subgraph of $G_i$ restricted to the vertices without semi-edges has a perfect matching). Since this is a necessary condition for the existence of a covering, in a negative case we falsify $\varphi$, e.g. by inserting the clause $(x \wedge \neg x)$ for some variable $x$.

     The satisfying truth assignment of $\varphi$ viewed as a vertex map $V(G_i) \to \{v, w\}$ can be converted to a covering projection $f\colon D(G_i) \to D(H_i)$ in two steps. First we separate those components of $G_i$ whose vertices shall be mapped onto $v$ from those that are mapped on $w$ and treat each component $C$ of $G_i$ separately and independently of the other ones. Then, if $H_i$ contains a semi-edge, we map the 1-factor of $C$ on the appropriate semi-edges of $H_i$. What remains is a $2k$-regular subgraph of $C$ that covers $k$ loops by Petersen's theorem [123].

- When $H_i$ is a multiedge of multiplicity $k$, then for every edge $(d, e) \in E(G_i)$, we add two 2-SAT clauses to express $x_d \neq x_e$.

  The existence of a satisfying truth assignment of $\varphi$ implies that $G_i$ is $k$-regular bipartite and therefore can be factorized into a collection of disjoint perfect matchings by Hall's theorem. We map edges of each such matching to the same edge of $H_i$ while using distinct edges of $H_i$ for distinct matchings [123].

- When $H_i$ is a 2-regular graph consisting of a single edge and two semi-edges, then each $G_i$ covering $H_i$ needs to be 2-regular, where each component of $G_i$ is either a cycle of $4k$ vertices or an open path on an even number of vertices.

  For every subgraph of $G_i$ isomorphic to a half-open path on five darts (such path consist of two edges and a single semi-edge) we add two 2-SAT clauses to express $x_d \neq x_e$, where $d$ and $e$ are the two outer darts of such path. Moreover, when a dart $d$ belongs to a semi-edge in $d$, we add two 2-SAT clauses to express $x_d \neq x_e$ for the dart where $e$ is the other end of the half-open path on three darts starting in $d$ in $G_i$.

  The last condition guarantees that semi-edges of $G_i$ are mapped onto semi-edges of $H_i$, while the previous guarantees that every vertex with two neighbours in $G_i$ has one neighbour mapped on the same target in $H_i$ and the other neighbour onto the distinct vertex of $H_i$.

- When $H_{i,j}$ is disconnected we express $x_d = x_e$ for every link $(d, e) \in \Lambda(G_{i,j})$ as above. Now we follow the same approach as for disconnected $H_i$ above. Note only that semi-edges may not appear in $G_{i,j}$.

  In order to factorize each component $C$ of $G_{i,j}$ into $k$ 2-factors where each maps onto a distinct loop incident with the target vertex, we temporarily split each vertex of $C$ into two vertices of degree $k$: one incident with darts of block $B_i$ and the other with $B_j$. Such an auxiliary graph is bipartite and could be factorized into $k$ 1-factors. Edges of each 1-factor of the auxiliary graph correspond to the desired 2-factor of $C$ [123].

- Finally, when $H_{i,j}$ is connected we express $x_d \neq x_e$ for every link $(d, e) \in \Lambda(G_{i,j})$ and proceed analogously as above. However, this case is simpler in the sense that each component of $C$ is bipartite. We perform the auxiliary vertex splitting described in the previous case to obtain two collections of $k$ 1-factors which in $C$ are mapped onto the two multiedges of multiplicity $k$ obtained by splitting of $H_{i,j}$ [123].

For a disconnected $H$, our Theorem 7.13 guarantees that once we are able to decide the existence of a covering projection for both components of $H$ by Case 1 above, we may resolve the $H$-EquitableCover problem as well.

Now we proceed to the NP-complete cases.

1. Let $H_i$ be the subgraph of $H$ for which the $H_i$-Cover is NP-complete. By $H'$ we denote the complement of $H_i$ in $H$. Let $G_i$ be the graph for which the covering to $H_i$ is questioned. We create a graph $G$ from $G_i$ and $|V(G_i)|$ copies of $H'$ so that distinct vertices of $G_i$ are merged with the vertex $u$ of distinct copies of $H'$ and every vertex of $G_i$ participates in one of these joints.

   When $G_i$ covers $H_i$, then it is easy to extend the covering to each copy of $H'$ by the identity mapping on $H'$. On the other hand the restriction of a covering $G \to H$ to the subgraph $G_i$ is a covering to $H_i$.

   When we aim for a simple instance for the $H$-Cover problem, we use any simple graph $H''$ that allows a $k$-fold cover of $H'$. Then we involve $k$ copies of $G_i$ instead of the single one and perform merges so that the $k$ vertices mapped onto $u$ in $H''$ are merged with the $k$ copies of the same vertex in distinct copies of $G_i$.

2. This case follows by the same argument as the previous one, however, we use $H_i^u$ instead of $H_i$.

3. For this case, we need the concept of a ("categorical") graph *product*: For a coloured graph $G$, the product $G \times 2$ has as the dart set the Cartesian product $D(G) \times \{1,2\}$. To simplify our expressions we use $d_1$ for $(d,1)$ and $u_1$ for $u \times \{1\}$ when the use of indices cannot be misinterpreted in another way. The two darts $d_1, d_2$ have the same colour as $d$. Every vertex $u \in V(G)$ yields two vertices $u_1$ and $u_2$ of the same colour as $u$. Every semi-edge $s \in S$ yields two semi-edges $s_1$ and $s_2$, while every loop or edge $\{d, d'\} \in L \cup O$ yields two links $\{d_1, d'_2\}$ and $\{d'_1, d_2\}$.

   Note that when we use the natural projection, i.e. map both $d_1, d_2$ onto $d$ for every $d \in D$, we get a covering $G \times 2 \to G$.

   (a) As above, let $H_i$ be any subgraph of $H$ for which $H_i$-EquitableCover is NP-complete, $H'$ be the complement of $H_i$ in $H$ and $G_i$ be the graph for which the covering to $H_i$ is questioned.

   - For connected $H_i$ we take two copies of $G_i$, one copy of $G_i \times 2$ and $|V(G_i)|$ copies of $H' \times 2$. If $u$ is a vertex of $G_i$, let $u'$ and $u''$ be the two vertices corresponding to $u$ in the two copies of $G_i$, while $u'''_1$ and $u'''_2$ match $u$ in $G_i \times 2$. Analogously let vertices $v_1, v_2, w_1, w_2$

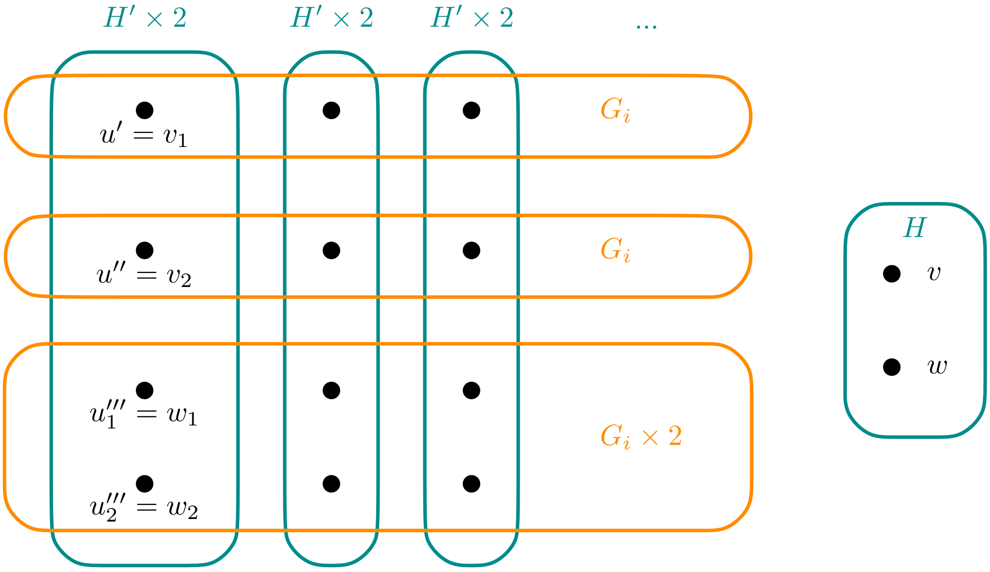


Figure 7.4: Construction of $G$ in Theorem 7.20.

match $v$ and $w$ of $H'$ in $H' \times 2$. We form $G$ by choosing for each $u \in V(G_i)$ a unique copy of $H' \times 2$ and merging the pairs $(u', v_1)$, $(u'', v_2)$, $(u_1''', w_1)$ and $(u_2''', w_2)$, where $v_1, v_2, w_1, w_2$ are taken from the chosen copy of $H' \times 2$, see Figure 7.4

Observe that besides the natural projection, the graph $H' \times 2$ allows also its "swap", $\{v_1, v_2\} \to w, \{w_1, w_2\} \to v$ as a covering projection.

Analogously, if $G_i$ covers $H_i$ then $G_i \times 2$ covers $H_i$ also via a projection and its swap. Thus any covering $G_i \to H_i$ can be extended to a covering of $G \to H$ from the union of the two copies of this covering on the two copies of $G_i$ and its swap on $G_i \times 2$.

As above, the restriction of a covering $G \to H$ to $G_i$ yields a map $G_i \to H_i$. As $H_i$ is connected, this is a $k$-fold covering for some $k$. (Disconnected $H_i$ would yield only a locally bijective homomorphism $G_i \longrightarrow_{lb} H_i$.)

Analogously, to get a simple instance we may involve a simple cover of $H' \times 2$ to connect an appropriate number of copies of $G_i$ and $G_i \times 2$.

- For disconnected $H_i$ we first recall that the $H_i$-EquitableCover is polynomially solvable if each component of $H_i$ is incident with at most one semi-edge or the degree of $H_i$ is two.

  Let $H_i^+$ be the component of $H_i$ with the maximum number of semi-edges, i.e. at least two, and $H_i^-$ its complement in $H_i$.

We reduce the NP-complete problem $H_i^+$-Cover (Note that $H_i^-$-Cover could be polynomially solvable.)

Let us have a given instance $G_i$ of $H_i^+$-Cover. Without loss of generality we may assume that $G_i$ is connected as otherwise we treat each component separately. Since $H_i^+$ contains semi-edges, $G_i$ must have an even number of vertices as otherwise it is a clear no instance. We use $G_i$ together with $|V(G_i)|$ copies of $H_i^-$ and $\frac{1}{2}|V(G_i)|$ copies of $H' \times 2$. We partition vertices of this copy of $G_i$ into pairs arbitrarily and each such pair we match with $v_1, v_2$ of a copy of $H' \times 2$, while $w_1, w_2$ we match with distinct vertices of the copies of $H_i^-$. This concludes the construction of $G$.

Observe that a copy of $H_i^-$ may cover $H_i^+$ if and only if they are isomorphic. Hence we may assume that every copy of $H_i^-$ maps in an equitable covering $G \to H$ on the subgraph $H_i^-$. Then the copy of $G_i$ maps onto $H_i^+$.

For simple instances, we shall involve graph that allows a covering to $H_i^-$ but not to $H_i^+$. For example, the simplest case when $H_i^+$ consists of just of three semi-edges, we would have to involve a *snark* — a cubic graph that is not 3-edge colourable.

(b) When $H_{i,j}$-EquitableCover is NP-complete for some subgraph $H_{i,j}$ then we perform the same construction of $G$ from two copies of the instance $G_{i,j}$, a copy of $G_{i,j} \times 2$ and $|V(G_i)|$ copies of $H' \times 2$ that are merged together in the same way as in the previous case. The arguments are then identical.

We covered all the possible cases and thus the proof concludes. □

## 7.6 Conclusion

The main goal of this chapter was to point out that the generalization of the notion of graph covers of connected graphs to disconnected ones is not obvious. We have presented three variants, depending on whether the projection should or need not be globally surjective and if all vertices should be covered the same number of times. We argue that the most restrictive variant, which we call equitable covers, is the most appropriate one, namely from the point of view of covers of coloured graphs.

We have compared the computational complexity aspects of these variants. We show that two of them, surjective and equitable covers, possess the naturally desired property that $H$-Cover is polynomially solvable if covering each component of $H$ is polynomially solvable and NP-complete if covering at least one component of $H$ is NP-complete. However, we identified some open questions from the view of fixed-parameter tractability.

In the last section, we reviewed the extension of graph covers to covers of coloured graphs, recalling that non-coverable patterns can encode colours in simple graphs and discussing this issue in detail for the case when semi-edges are allowed. With this new feature, we conclude the complete characterization of the computational complexity of covering two-vertex coloured graphs, initiated (and proved for graphs without semi-edges) 24 years ago in [123].

# Chapter 8

# List covering of regular multigraphs

This chapter is based on:

- [28] Jan Bok, Jiří Fiala, Nikola Jedličková, Jan Kratochvíl, and Paweł Rzążewski: *List covering of regular multigraphs.* In Combinatorial Algorithms - 33rd International Workshop, IWOCA 2022, volume 13270 of Lecture Notes in Computer Science, pages 228–242, 2022. `https://doi.org/10.1007/978-3-031-06678-8_17`

It has been known that for every fixed simple regular graph $H$ of valency greater than 2, deciding if an input graph covers $H$ is NP-complete. In recent years, topological graph theory has developed into heavily relying on multiple edges, loops, and semi-edges, but only partial results on the complexity of covering multigraphs with semi-edges are known so far. In this chapter we consider the list version of the problem, called List-$H$-Cover, where the vertices and edges of the input graph come with lists of admissible targets. Our main result reads that the List-$H$-Cover problem is NP-complete for every regular multigraph $H$ of valency greater than 2 which contains at least one *semi-simple vertex* (i.e., a vertex which is incident with no loops, with no multiple edges and with at most one semi-edge). Using this result we show the NP-complete/polynomial time dichotomy for the computational complexity of List-$H$-Cover of cubic multigraphs.

The complexity study of $H$-Cover for graphs $H$ that allow semi-edges has been initiated in 2021 in [27, 29] (Chapters 6 and 7, respectively). We continue this line of research. In particular, our far-reaching goal is to prove the following conjecture.

**Conjecture 8.1** (Strong Dichotomy Conjecture)**.** *For every $H$, the $H$-*Cover *problem is either polynomial-time solvable for general graphs, or NP-complete for simple graphs.*

We believe that the Strong Dichotomy Conjecture holds true also for List-$H$-Cover.

## 8.1 Summary of our results

The goal of this chapter is to push further the understanding of the complexity of $H$-Cover for regular graphs. Recall that the problem is known to be NP-complete for every fixed $k$-regular *simple* graph $H$ of valency $k \geq 3$ [125]. Though it was known already from [123] that in order to fully understand the complexity of covering general simple graphs, it is necessary (and sufficient) to prove a complete characterization for coloured mixed multigraphs, the result of [125] was formulated and proved only for simple graphs. In this chapter we revisit the method developed in [125] and we conclude that though it does not seem to work for multigraphs in general, it is possible to modify it and — under certain assumptions — prove hardness for the *list* variant of the problem, List-$H$-Cover, where the vertices and edges of the instance graph are given lists of admissible targets. Our main result is the following theorem.

**Theorem 8.2.** *Let $k \geq 3$ and let $H$ be a $k$-regular graph. If $H$ contains a semi-simple vertex, then* List-$H$-Cover *is NP-complete for simple input graphs.*

The second goal of this chapter is to show how Theorem 8.2 could be used to prove the Strong Dichotomy Conjecture for cubic graphs. (Recall that for the closely related locally injective homomorphism problem, introducing lists was helpful in obtaining the full complexity dichotomy [82].) We fully characterize the computational complexity of List-$H$-Cover for all cubic graphs in Theorem 8.12.

## 8.2 Additional definitions

We dedicate this section to a couple of definitions not provided by the introductory chapter.

**The problem.** In the List-$H$-Cover problem, the input graph $G$ is given with lists $\mathcal{L} = \{L_u, L_e : u \in V(G), e \in E(G)\}$, such that $L_u \subseteq V(H)$ for every $u \in V(G)$ and $L_e \subseteq E(H)$ for every $e \in E(G)$. A covering projection $f : G \longrightarrow H$ *respects* the lists of $\mathcal{L}$ if $f(u) \in L_u$ for every $u \in V(G)$ and $f(e) \in L_e$ for every $e \in E(G)$.

**Simplicity.** It will be useful to distinguish various levels of simplicity. We say that:

- a vertex is *semi-simple* if it belongs to no loops, no multiple edges and at most one semi-edge,
- a graph is *semi-simple* if each of its vertices is semi-simple,
- a vertex is *simple* if it is semi-simple and is incident with no semi-edges,
- a graph is *simple* if each of its vertices is simple,
- a graph is *bipartite* if it has no loops, no semi-edges and no odd cycles.

**Partial covering.** The mapping $f : V(G) \cup E(G) \longrightarrow V(H) \cup E(H)$ is a *partial covering projection* when the preimages are not required to be spanning subgraphs, but all other properties are fulfilled. In other words, the vertex- and edge-mappings are both surjective and the incidences are retained, the preimage of a ordinary edge connecting vertices say $u$ and $v$ is a matching consisting of edges each connecting a vertex from $f^{-1}(u)$ to a vertex from $f^{-1}(v)$, the preimage of a semi-edge incident with vertex say $u$ is a disjoint union of semi-edges and ordinary edges all incident only with vertices from $f^{-1}(u)$, and the preimage of a loop incident with a vertex say $u$ is a disjoint union of cycles (including loops) and paths whose all edges are incident only with vertices from $f^{-1}(u)$.

**Incident edges.** Given a graph $G$ and a vertex $u \in V(G)$, the set of edges of $G$ incident with $u$ will be denoted by $E_G(u)$.

## 8.3 Proof of Theorem 8.2

In the first two subsections we will prove the theorem for the case when $H$ is bipartite (and hence does not contain loops) and has no semi-edges. By the celebrated König-Hall theorem, such a graph is $k$-edge-colourable.

### 8.3.1 Multicovers

The following operation will be an important tool used in our hardness proof.

**Definition 8.3** (Coloured product)**.**

1. *Let $M_1, M_2, \ldots, M_m$ be $m$ perfect matchings, possibly on different vertex sets. Their product is the graph*

$$\prod_{i=1}^{m} M_i = (\prod_{i=1}^{m} V(M_i), \{uv : u_i v_i \in M_i \text{ for each } i = 1, 2, \ldots, m\})$$

   *where it is assumed that the notation of the vertices of the product is such that $u = (u_1, u_2, \ldots, u_m)$ with $u_i \in V(M_i)$ for all $i = 1, 2, \ldots, m$. Then $\prod_{i=1}^{m} M_i$ is a perfect matching as well.*

2. *Let $G_1, G_2, \ldots, G_m$ be $k$-regular $k$-edge-colourable graphs without loops or semi-edges. For each $i = 1, \ldots, m$, let $\phi_i : E(G_i) \longrightarrow \{1, 2, \ldots, k\}$ be a proper edge-colouring of $G_i$. The* coloured product *of $G_i$'s is the graph $\prod_{i=1}^{m} G_i$ with vertex set being $\prod_{i=1}^{m} V(G_i)$ and edge set being the union of $E(\prod_{i=1}^{m} M_i^j)$, $j = 1, 2, \ldots, k$, where for each $i$ and $j$, $M_i^j = \phi_i^{-1}(j)$ is the perfect matching in $G_i$ formed by edges coloured by colour $j$ in the colouring $\phi_i$. If we define $\phi : E(\prod_{i=1}^{m} G_i) \longrightarrow \{1, 2, \ldots, k\}$ by setting $\phi(e) = j$ if and only if $e \in E(\prod_{i=1}^{m} M_i^j)$, we see that $\phi$ is a proper $k$-edge-colouring of $\prod_{i=1}^{m} G_i$.*
3. *Let $G_1, G_2, \ldots, G_m$ and $\phi_i : E(G_i) \longrightarrow \{1, 2, \ldots, k\}$ be as in 2) above. For each $i = 1, 2, \ldots, m$, define the projection $\pi_i$ from the coloured product $\prod_{i=1}^{m} G_i$ to its i-th coordinate by setting $\pi_i(u) = u_i$ and $\pi_i(e) = e_i$ for such an edge $e_i \in E(G_i)$ that satisfies $\phi_i(e_i) = \phi(e)$ and whose end-vertices are $u_i$ and $v_i$, provided the end-vertices of $e$ are $u$ and $v$.*

An example of coloured product is in Figure 8.1. The next lemma follows immediately from Definition 8.3.

**Lemma 8.4.** *Let $G_1, G_2, \ldots, G_m$ be $k$-regular graphs without loops or semi-edges, and let for each $i = 1, \ldots, m$, $\phi_i : E(G_i) \longrightarrow \{1, 2, \ldots, k\}$ be a proper edge-colouring of $G_i$. Then each projection $\pi_i$, $i = 1, 2, \ldots, m$, is a covering projection from $\prod_{i=1}^{m} G_i$ onto $G_i$.*

Now let us show the main construction used to build the gadgets for our hardness reduction.

**Proposition 8.5.** *Let $H$ be a connected $k$-regular $k$-edge-colourable graph with no loops or semi-edges. Let $x, y$ be two adjacent vertices of $H$. Then there exists a connected simple $k$-regular $k$-edge-colourable graph $G$ and $u \in V(G)$, such that*

*(a) for any bijection from $E_G(u)$ onto $E_H(x)$, there exists a covering projection from $G$ to $H$ which extends this bijection and maps $u$ to $x$, and*
*(b) for any bijection from $E_G(u)$ onto $E_H(y)$ there exists a covering projection from $G$ to $H$ which extends this bijection and maps $u$ to $y$.*

*Proof.* In a way similar to the proof in [125] we first construct a coloured product of many many copies of $H$ that covers $H$ in many ways, in particular, that satisfies (a) and (b). To do so, we take $k!$ copies of $H$ with edge colourings obtained by all permutations of colours, and in their coloured product consider the vertex $a$ whose all projections are $x$. The edges incident with $a$ are projected onto $E_H(x)$ in all possible ways from this coloured product. Then we take the same product and consider its vertex $b = (y, y, \ldots, y)$. The edges incident with $b$ are projected in all possible ways onto $E_H(y)$. Finally, in the product of these two graphs, the vertex $(a, b)$ plays the role of $u$. The difference to the approach in [125] is that it is not sufficient to require that all bijections of the vertex neighbourhoods of $u$ and

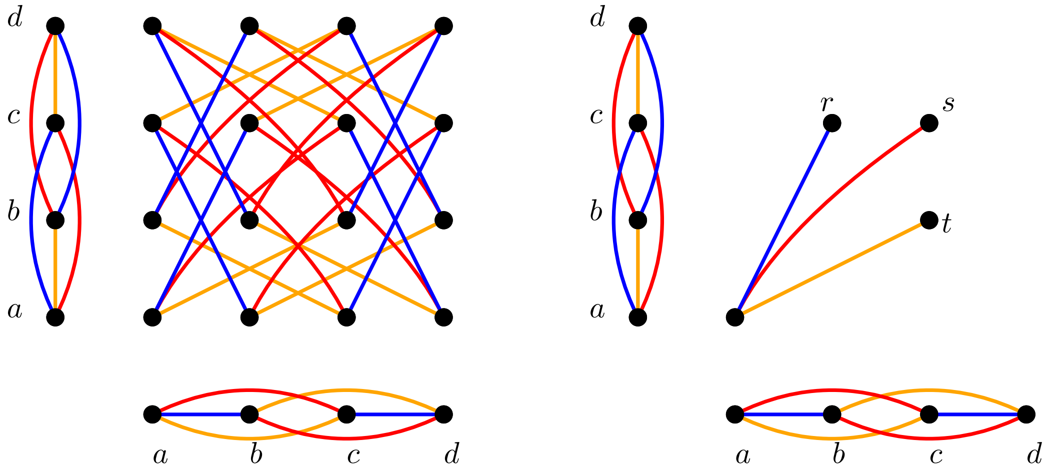


Figure 8.1: An example of the coloured product is on the left. Projections of vertex $u$ and its neighbourhood are visualized on the right; $u, r, s, t$ map in this order onto $a, b, c, c$ in the vertical projection, and onto $a, c, c, b$ in the horizontal one.

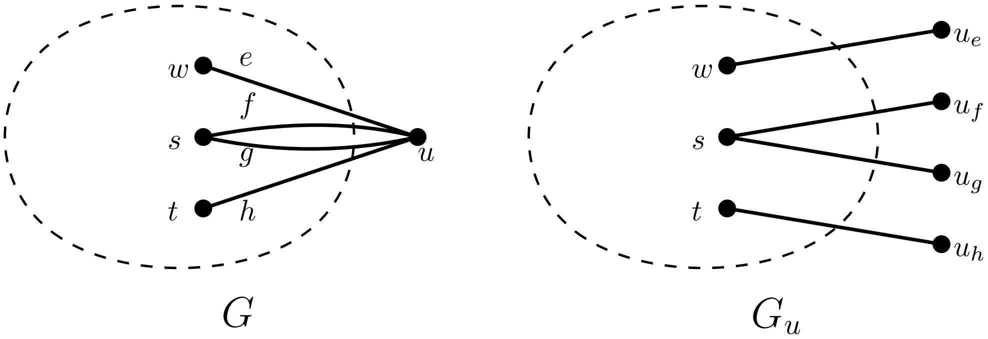


Figure 8.2: An illustration to the construction of $G_u$.

$x$ ($y$, respectively) can be extended to covering projections, but we must aim at extending bijections of the sets of incident edges. The product that we construct may still contain multiple edges. If this is the case, we further take the product with a simple graph, say $K_{k,k}$. This product is already a simple graph and still possesses all the requested covering projections. It may still be disconnected, though, and we denote by $G$ the component that contains the vitally important vertex $u$ in such a case. □

The main building block of our reduction is the graph $G_u$ obtained from $G$ by splitting vertex $u$ into $k$ pendant vertices of degree 1 (see Figure 8.2). For each edge $e$ of $G$ incident with $u$, we formally keep this edge with the same name in $G_u$, denote its pendant vertex of degree 1 by $u_e$ and denote by $w_e$ the other endpoint of $e$. (Thus, with this slight abuse of notation, $E_G(u) = \bigcup_{e \in E_G(u)} E_{G_u}(u_e)$.) Then we have the following proposition.

**Proposition 8.6.** *The graph $G_u$ constructed from the multicover $G$ of $H$ as above satisfies the following:*

*(a) for every bijection $\sigma_x\colon E_G(u) \to E_H(x)$, there exists a partial covering projection of $G_u$ onto $H$ that extends $\sigma_x$ and maps each $u_e, e \in E_G(u)$ to $x$;*
*(b) for every bijection $\sigma_y\colon E_G(u) \to E_H(y)$, there exists a partial covering projection of $G_u$ onto $H$ that extends $\sigma_y$*
*(c) in every partial covering projection from $G_u$ onto $H$, the pendant vertices $u_e, e \in E_G(u)$ are mapped onto the same vertex of $H$;*
*(d) in every partial covering projection from $G_u$ onto $H$, the pendant edges are mapped onto different edges (incident with the image of the pendant vertices).*

*Proof.* Items (a) and (b) follow directly from Proposition 8.5. To prove (c) and (d), we first show that in any partial covering projection $f$ from $G_u$ onto $H$, the pendant edges are mapped onto edges of all colours.

Denote by $E^*$ the pendant edges of $G_u$, and set $V^* = V(G) \setminus \{u\} = V(G_u) \setminus \bigcup_{e\in E_G(u)}\{u_e\}$. Observe first that since $H$ has a perfect matching, the number of its vertices is even, and since $G$ covers $H$, so is the number of vertices of $G$. Hence $|V^*| \equiv 1 \mod 2$.

Suppose $f : G_u \longrightarrow H$ is a partial covering projection. Fix a proper $k$-colouring $\phi$ of the edges of $H$ and define a $k$-colouring $\widetilde{\phi}$ of the edges of $G_u$ by setting $\widetilde{\phi}(e) = \phi(f(e))$. Since $f$ is a partial covering projection, $\widetilde{\phi}$ is a proper edge-colouring of $G_u$. If some colour is missing on all of the edges from $E^*$, edges of this colour would form a perfect matching in $G_u[V^*]$ and $|V^*|$ would be even, a contradiction. Hence every colour appears on exactly one edge of $E^*$.

For every $a \in V(H)$, denote by $h_a$ the number of vertices of $V^*$ that are mapped onto $a$ by $f$. Consider an edge $e'$ connecting vertices $a$ and $b$ of $H$. If $e' = f(e)$ for some edge $e \in E^*$, we have $f(u_e) = a$ and $f(w_e) = b$, or vice versa. Every vertex in $f^{-1}(a) \cap V^*$ is adjacent to exactly one vertex in $f^{-1}(b) \cap V^*$ via an edge of colour $\phi(e')$, whilst every vertex of $f^{-1}(b)\cap V^*$ except for $w_e$ is adjacent to exactly one vertex in $f^{-1}(a) \cap V^*$ via an edge of the same colour. Hence $h_b = h_a + 1$. Orient the edge $e'$ from $a$ to $b$ in such a case. If, on the other hand, the edge $e'$ is not the image of any edge from $E^*$, the edges of colour $\phi(e')$ form a matching between the vertices of $f^{-1}(a) \cap V^*$ and the vertices of $f^{-1}(b) \cap V^*$, and $h_a = h_b$. Leave the edge $e'$ undirected in such a case.

After processing all edges of $H$ in this way, we have constructed a mixed graph $\overrightarrow{H}$ which has exactly one edge of each colour directed. From the meaning of the orientations and non-orientations of edges of $\overrightarrow{H}$ for the values of $h_a, a \in V(H)$, it follows that the vertex set of $H$ falls into levels, say $L_r, L_{r+1}, \ldots, L_s$ such that undirected edges live inside the levels, while directed edges connect vertices of consecutive levels and are directed from $L_i$ to $L_{i+1}$ for suitable $i$. Every two consecutive levels are connected by at least one directed edge in this way. Since

$H$ is connected, every vertex $a \in L_i$ will satisfy $h_a = i$ (the indices $r$ and $s$ are chosen so that $r$ is the smallest value of $h_a$ and $s$ is the largest one). Directed edges connecting two consecutive levels form a cut in $H$, and since edges of each colour form a perfect matching, the parities of the numbers of edges of each colour in this cut are the same. Since the cut contains at least one edge, but at most one edge of each colour (exactly one edge of each colour is directed), it follows that the cut contains all $k$ directed edges and that $H$ has only two levels, i.e., $s = r + 1$. Thus $H$ has $|L_r|$ vertices $a$ with $h_a = r$ and $|L_{r+1}| = |V(H)| - |L_r|$ vertices $a$ with $h_a = r + 1$. It follows that

$$|V^*| = r|L_r| + (r+1)(|V(H)| - |L_r|) = (r+1)|V(H)| - |L_r|.$$

We know that $G$ covers $H$, and so $|V(G)| = \ell|V(H)|$ for some $\ell$. Thus $|V^*| = \ell|V(H)| - 1$. Thus we obtain

$$(r+1)|V(H)| - |L_r| = \ell|V(H)| - 1,$$

which implies

$$(r + 1 - \ell)|V(H)| = |L_r| - 1,$$

and since $1 \leq |L_r| \leq |V(H)| - 1$, the only possible way for $|L_r| - 1$ to be divisible by $|V(H)|$ is $|L_r| = 1$. But this implies that all directed edges of $\overrightarrow{H}$ start in the same vertex, say $z$, and from the construction of $\overrightarrow{H}$ it follows that $f(u_e) = z$ for all $e \in E^*$. This proves (c).

Now (d) follows from the two observations above. The $k$ pendant edges of $E^*$ have mutually distinct colours in $\widetilde{\phi}$, and thus they must be mapped to distinct edges of $E_H(z)$ by $f$. □

### 8.3.2 Reduction from hypergraph colouring

The reduction is exactly the same as in [125], but the proof for the case when multiple edges are allowed needs some extra analysis. Hence we need to describe here the reduction in full detail. We reduce from $k$-edge-colourability of $(k-1)$-uniform $k$-regular hypergraphs. In the wording of the incidence graph of the hypergraph, suppose we are given a simple bi-regular bipartite graph $K = (A \cup B, E)$ such that all vertices in $A$ (which represent the edges of the hypergraph) have degree $k - 1$ and all vertices in $B$ (which represent the vertices of the hypergraph) have degree $k$. The question is, if the vertices of $A$ can be coloured by $k$ colours so that the neighbourhood of each vertex from $B$ is rainbow coloured (i.e., each vertex from $B$ sees all $k$ colours on its neighbours, each colour exactly once). This problem is NP-complete for every fixed $k \geq 3$ [125].

Given such a graph $K$, we build an input graph $G_K$ by local replacements. Recall that we are working with a $k$-edge-colourable $k$-regular graph $H$ with a simple vertex $x$, and with respect to this vertex (and one of its neighbours $y$) we are

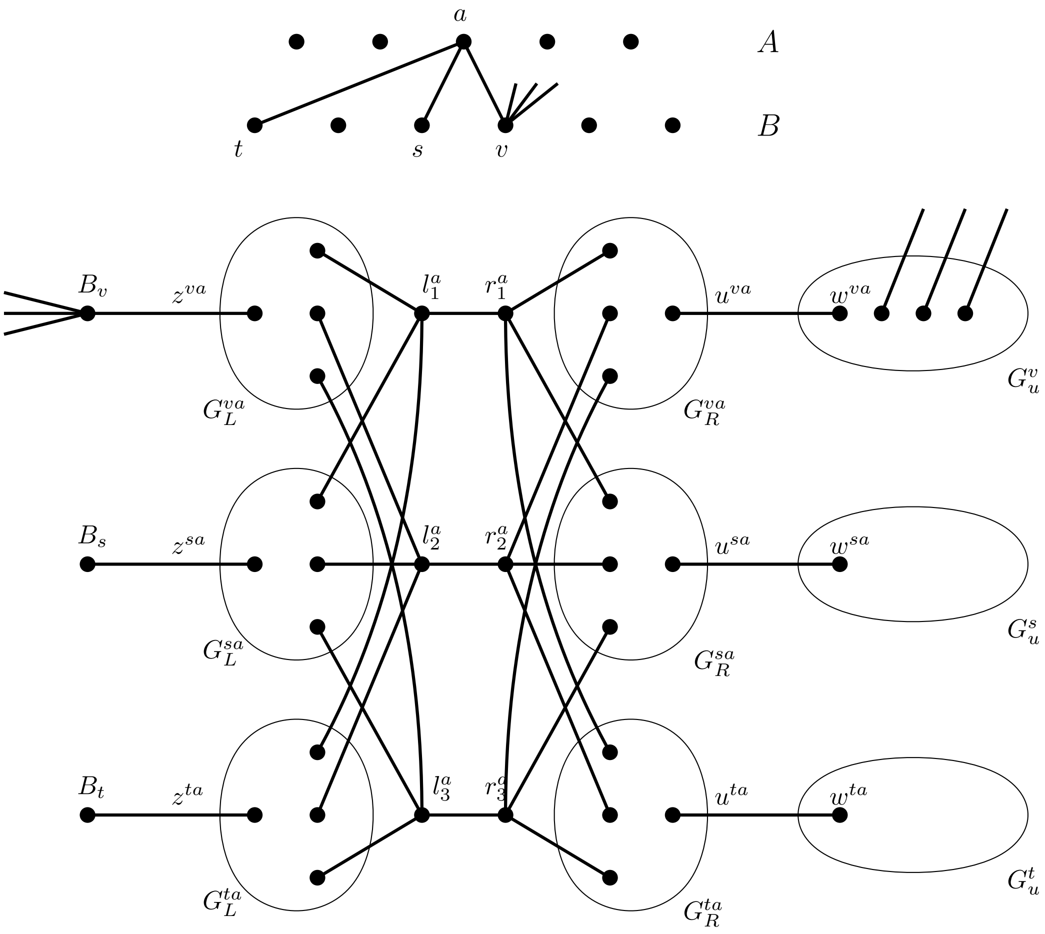


Figure 8.3: An illustration to the construction of $G_K$ for $k = 4$.

guaranteed the existence of a graph $G_u$ which satisfies the properties stated in Proposition 8.6. This $G_u$ will be a key building block in our construction.

First, every vertex $v \in B$ will be replaced by a copy of the so-called vertex gadget, which is a disjoint union of a copy $G_u^v$ of $G_u$ and a single vertex $B_v$. For every neighbour $a \in A$ of $v$, one of the pendant vertices of $G_u^v$ will be denoted by $u^{va}$, and its neighbour within $G_u^v$ will be denoted by $w^{va}$.

The hyperedge gadgets used to replace the vertices of $A$ are more complicated. This gadget consists of $2(k-1)$ copies of $G_u$ linked together in the following way. Let $a \in A$. We take $2(k-1)$ vertices $\ell_i^a, r_i^a, i = 1, 2, \ldots, k-1$, and for every neighbour $v$ of $a$, we take two copies $G_L^{va}$ and $G_R^{va}$ of $G_u$. The pendant vertices of $G_L^{va}$ will be unified with $B_v$ and $\ell_1^a, \ell_2^a, \ldots, \ell_{k-1}^a$, while the pendant vertices of $G_R^{va}$ will be unified with $w^{va}$ and $r_1^a, r_2^a, \ldots, r_{k-1}^a$. The neighbour of $B_v$ in $G_L^{va}$ will be denoted by $z^{va}$. Lastly, the matching $\ell_i^a r_i^a, i = 1, 2, \ldots, k-1$ is added. This completes the construction of $G_K$. See Figure 8.3 for an illustrative example.

The resulting graph $G_K$ is $k$-regular. To make it an instance of the LIST-$H$-

Cover problem, we prescribe that the vertices $B_v, v \in B$ and $\ell_i^a, a \in A, i = 1, 2, \ldots, k-1$ are all mapped onto $x$ (this means, that for these vertices, their lists of admissible target vertices are one-element and all the same, while for the remaining vertices, their lists are full, as well as for all the edges).

The fact that $x$ is simple implies the following observation: For every partial covering projection from $G_u$ to $H$ which maps all the pendant vertices onto $x$, their neighbours in $G_u$ are mapped onto distinct vertices of $H$ (this immediately follows from property (d) of Proposition 8.6). Similarly, if any vertex of $G_K$ is mapped onto $x$ by a covering projection, then its neighbours are mapped onto distinct vertices of $H$ (the neighbourhood $N_H(x)$ of $x$ in $H$). We will exploit these observations in the following argumentation.

Suppose $f : G_K \longrightarrow H$ is a covering projection such that all vertices $B_v, v \in B$ and $\ell_i^a, a \in A, i = 1, 2, \ldots, k-1$ are mapped onto $x$. Consider an $a \in A$, and let $f(r_1^a) = y$, whence $y \in V(H)$ is a neighbour of $x$ in $H$. Property (c) applied to any $G_R^{va}$, for $v$ being a neighbour of $a$ in $K$, implies that $f(r_i^a) = f(w^{va}) = y$ for all $i = 2, \ldots, k-1$. Since each $\ell_i^a$ has a neighbour $r_i^a$ mapped onto $y$, none of their neighbours in $G_L^{va}$ is mapped onto $y$. Property (d) then implies that $f(z^{va}) = y$. Define a colouring $\phi$ of $A$ by colours $N_H(x)$ as $\phi(a) = f(r_1^a)$. Consider a vertex $v \in B$. The neighbours of $B_v$ in $G_K$ are $z^{va}, a \in N_K(v)$. Since $f(B_v) = x$ and $x$ is simple, the vertices $z^{va}, a \in N_K(v)$ are mapped onto different neighbours of $x$ by $f$, and hence the colours $\phi(a), a \in N_K(v)$ are all distinct. Thus $\phi$ is a $k$-colouring of $A$ of the required property.

Suppose for the opposite direction that $A$ allows a $k$-colouring $\phi$ such that each vertex $v \in B$ sees all $k$ colours on its neighbours, and identify the colours with the names of the neighbours of $x$ in $H$. Furthermore, set $f$ in the following way:

$$f(B_v) = \ell_i^a = x \text{ for all } v \in B, a \in A, i = 1, 2, \ldots, k-1 \text{ (as required by the lists),}$$
$$f(u^{va}) = x \text{ for all } v \in B \text{ and } a \in N_K(v), \text{and}$$
$$f(r_i^a) = f(w^{va}) = f(z^{va}) = \phi(a) \text{ for all } a \in A \text{ and } v \in N_K(a).$$

Finally, define $f$ on the edges incident to $\ell_i^a$ ($r_i^a$, respectively) so that for every $i$, these edges are mapped onto different edges incident to $x$ (to $\phi(a)$, respectively), and, on the other hand, for every $v \in N_K(a)$, the pendant edges of $G_L^{va}$ (of $G_R^{va}$, respectively) are mapped onto distinct edges incident to $x$ (to $\phi(a)$, respectively).

The properties (a) and (b) of Proposition 8.6 imply that this mapping can be extended to partial covering projections within each copy of $G_u$ used in the construction of $G_K$. To see that they altogether provide a covering projection from $G_K$ to $H$, note that for each $v \in B$, the edges incident with the vertex $B_v$ are mapped onto different edges because their other endpoints are $\phi(a), a \in N_K(v)$, and hence all different by the assumption on the colouring $\phi$, and also each copy $G_u^v$ has its pendant edges mapped onto different edges incident to $x$, since the pendant vertices $u^{va}, a \in N_K(v)$ are all mapped onto $x$ and their neighbours $w^{va}$

in $G_u^v$ are mapped onto distinct vertices $\phi(a), a \in N_K(v)$. This concludes the proof of the case of bipartite $H$.

### 8.3.3 The non-bipartite case

Suppose the graph $H$ is not $k$-edge-colourable (this includes the case when $H$ contains loops and/or semi-edges). Consider $H' = H \times K_2$. This $H'$ may still contain multiple edges (the product of a multiple ordinary edge with $K_2$ is again a multiple edge, but also the product of a loop with $K_2$ is a double ordinary edge, and the product of a multiple semi-edge with $K_2$ results in a multiple ordinary edge as well), but it is bipartite (and thus has neither semi-edges nor loops) and therefore is $k$-edge-colourable. In the product with $K_2$, every semi-simple vertex of $H$ results in two simple vertices of $H'$. Hence, by the result of the preceding subsection, List-$H'$-Cover is NP-complete.

It is proved in [83] that for simple graphs, $G$ covers $H \times K_2$ if and only if $G$ is bipartite and covers $H$. This proof readily extends to graphs that allow loops, semi-edges and multiple edges. The proof for the list version of the problem may get more complicated in general. However, the list version that we have proven NP-complete in the preceding subsection is very special: the lists of all edges are full, and so are the lists of all the vertices except for those which are prescribed to be mapped onto the same simple vertex, say $x'$. If we take such an instance of List-$H'$-Cover, this $x'$ is a copy of a semi-simple vertex $x \in V(H)$, and all vertices of the input graph $G$ that are prescribed to be mapped onto $x'$ are from the same class of its bipartition. We just prescribe them to be mapped onto $x$ as an instance of List-$H$-Cover. It is easy to see that this mapping can be extended to a covering projection to $H$ if and only if $G$ allows a covering projection to $H'$ in which all these prescribed vertices are mapped onto $x'$. This concludes the proof of Theorem 8.2.

## 8.4 Sausages and rings

In this section we consider two special classes of cubic graphs. These graphs play a special role in the classification in Theorem 8.12. The *$k$-ring* (where $k \geq 2$) is the cubic graph obtained from the cycle of length $2k$ by doubling every second edge. We call a *$k$-sausage* every cubic graph that is obtained from a path on $k$ vertices by doubling every other edge and adding loops or semi-edges to the end-vertices of the path to make the graph 3-regular. Note that while for every $k$, the $k$-ring is defined uniquely, there are several types of $k$-sausages, as depicted in Figure 8.4.

**Proposition 8.7.** *For every $k \geq 2$, let $S_k$ be a $k$-sausage. Then $S_k \times K_2$ is isomorphic to the $k$-ring.*

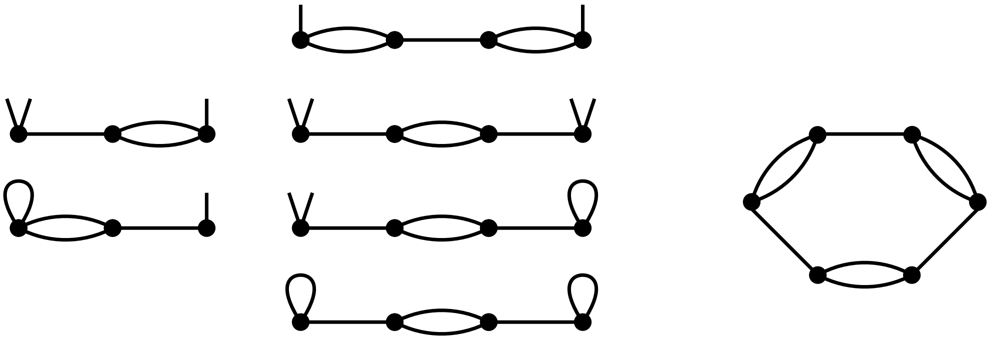


Figure 8.4: The two non-isomorphic 3-sausages (left), the four non-isomorphic 4-sausages (middle), and the 3-ring (right).

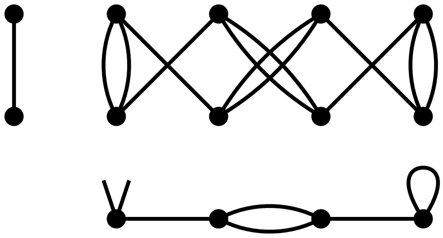


Figure 8.5: The product of a 4-sausage with $K_2$ is isomorphic to the 4-ring.

*Proof.* The product $H \times K_2$ is a bipartite graph with no loops or semi-edges, in which every ordinary edge in $H$ gives rise to a pair of ordinary edges of the same multiplicity, a loop, or a pair of semi-edges incident to the same vertex of $H$ gives rise to a double ordinary edge, and a single semi-edge in $H$ gives rise to a simple ordinary edge in $H \times K_2$. Thus $S_k \times K_2$ has a cyclic structure and the number of double edges is equal to the number of vertices of $S_k$, see Figure 8.5. □

In the following three theorems we show that List-$k$-ring-Cover problem is NP-complete for simple graphs on the input for every $k \geq 3$. (In other words, for all rings, except for 2-ring, for which the problem is already proven to be NP-complete even without lists by Theorem 8.2.)

**Theorem 8.8.** *The $k$-ring-Cover problem is NP-complete for simple input graphs for every $k = 2^\alpha(2\beta + 3)$ such that $\alpha$ and $\beta$ are non-negative integers.*

*Proof.* We reduce from $C_{(2\beta+3)}$-Hom for $\beta$ being a non-negative integer which is known to be NP-complete by the dichotomy theorem of Hell and Nešetřil [102]. We call $k$-ring occasionally $H$. Furthermore, the vertices of $H$, i.e. $k$-ring, will have its vertices consecutively denoted by $1, 1', \ldots, k, k'$ with precisely edges $jj'$ being double for $j \in \{1, \ldots, k\}$.

The reader is advised to consult Figure 8.6 to better understand the gadgets and the reduction.

Let us describe the vertex gadget. Suppose we have a vertex $v$ of degree $\deg(v)$. Then let us take two disjoint copies of $C_l$ where $l = 2k \times \deg(v)$. Let us denote them $C_1^V$ and $C_2^V$ and their vertices $v_{1,1}, \ldots, v_{1,l}$. The construction of the gadget proceeds in the following way:

- Add edges $v_{1,2i-1}v_{2,2i}$ for every $i \in \{1, \ldots, l/2\}$.
- Add edges $v_{2,2i-1}v_{1,2i}$ for every $i \in \{1, \ldots, l/2\}$.
- Delete edge $v_{2,2kj-1}v_{2,2kj}$ for every $j \in \{1, \ldots, \deg(v)\}$ and to each of the endpoints of the deleted edge, add a pendant vertex.

We call these pendant vertices leafs of the vertex gadget and we speak about *pairs of leafs* when we refer to the two pendant vertices created after the deletion of the same edge. Further, since the gadget is bipartite, we can say that leafs are either black or white depending on the part of the bipartition they belong to. Observe that every pair of leafs in the above sense has one black and one white vertex.

Before we describe the edge gadget, let us introduce *enforcing gadget* which is simply the same as the vertex gadget for vertex of degree 1, i.e. it is created as was described in the preceding with $l = 2k$.

In the following, we number the vertices of every cycle consecutively starting with 1. We can thus speak about e.g. the $i$-th even vertex. Also, we automatically take the indices of cycles modulo $2k$.

For the actual edge gadget, take $k$ disjoint copies of cycles $C_{2k}$. Let us denote these cycles $C_1^E, \ldots, C_k^E$. We now insert enforcing gadgets in between the cycles as follows. For all $j$ being odd and $j < k$ and for all even vertices of $C_j^E$, we identify one of the leafs of the enforcing gadget with the $i$-th even vertex of $C_j^E$, and we identify the other leaf of the enforcing gadget with the $i$-th even vertex of $C_{j+1}^E$.

The similar connection is done in case of $j$ even and $j < k$, except that the $i$-th odd vertex of $C_j^E$ is connected by a copy of enforcing gadget to the $i$-th odd vertex of $C_{j+1}^E$.

Now, except for $C_1^E$ and $C_k^E$, all vertices are of degree 3. For every vertex of degree 2 in $C_1^E$ except for the first and the $(1 + 2^{\alpha+1})$-th vertex, let us say the $i$-th one, we place the enforcing gadget between the $i$-th vertex of $C_1^E$ and the $(i + k)$-th vertex of $C_k^E$ again by identifying each of the leafs with one of the mentioned vertices. This completes the construction of the edge gadget. The only vertices of degree 2 are now the first and the $(1 + 2^{\alpha+1})$-th vertex in $C_1^E$ and the $(1 + k)$-th and the $(1 + 2^{\alpha+1} + k)$-th vertex in $C_k^E$.

Let us have an instance $G$ of $C_{(2\beta+3)}$-Hom. We shall construct a new graph $G'$. For each vertex in $G$, we take a copy of the vertex gadget of corresponding size and insert it into $G'$. For each edge $uv$ in $G$, we obtain a new copy of the edge gadget. We connect it with the vertex gadget corresponding to $u$ as follows. We take one of the so-far unused pair of leafs coming from the vertex gadget corresponding to $u$ and identify the black leaf of the pair with the first vertex in $C_1^E$ of the edge gadget and the white leaf of the pair with the $(1+k)$-th vertex of $C_k^E$. For the vertex gadget of $v$, we again take one of the so-far unused pair of leafs coming from the vertex gadget of $v$ and identify the black leaf of the pair with the $(1+2^{\alpha+1})$-th vertex in $C_1^E$ of the edge gadget and the white leaf of the pair with the $(1+2^{\alpha+1}+k)$-th vertex of $C_k^E$.

We shall now describe possible images of vertex and edge gadget under a covering projection to $k$-ring.

We claim that under every covering projection to $H$, all black leafs of a given vertex gadget will be mapped to the same vertex of $k$-ring and white vertices to its prime version (or vice versa). The crucial observation is that $v_{1,1}, v_{2,2}, v_{2,1}, v_{1,2}$ form a 4-cycle in vertex gadget. By a simple analysis then, $v_{1,1}$ and $v_{2,1}$ must be mapped to some $\ell$ of $H$ and $v_{1,2}$ and $v_{2,2}$ to $\ell'$ (or vice versa, but let us further assume the first possibility). Furthermore this enforces the images of vertices $v_{1,3}, v_{2,4}, v_{2,3}, v_{1,4}$ as well (and they form again a 4-cycle). A repeated use of this propagation ensures that images of $v_{2,2k-1}v_{2,2k}$ are $(\ell-1)$ and $(\ell-1)'$, respectively (and possibly modulo $2k$, which will be assumed from now on). By the construction, the pendant vertices have then images $(\ell-1)'$ and $(\ell-1)$. The argument then can be repeated further and further, until we arrive on the conclusion that all black leafs have inevitably the same image $(\ell-1)'$ and the white leafs $(\ell-1)$.

Specially, for the enforcing gadget, we get that its leafs must be mapped to the different endpoints of a specific double edge in $k$-ring. In other words, whenever we have a vertex which is being identified with one of the leafs of the enforcing gadget, then given its image $i$ under a covering projection to the $k$-ring, the other vertex identified with the other leaf of the enforcing gadget has to be mapped to $i'$ in $k$-ring or vice versa.

We claim that under every covering projection, edge gadget will be mapped to $k$-ring in the following way. Without loss of generality, the first vertex of $C_1^E$, let us call it $a$, will be mapped to 1. Clearly, as the edge gadget is connected here to a vertex gadget through $a$, one of the neighbours of $a$ in the edge gadget has to be mapped to $1'$ and the other to $k'$, or vice versa. In both cases these neighbours are connected through enforcing gadget to $C_2^E$ and thus this enforces not only the images of vertices at distance 2 from $a$ on $C_1^E$ but also the images of the vertices at distance two from $a$ on $C_2^E$. Proceeding inductively, we arrive on the conclusion that the vertices of $C_1^E$ are either consecutively $1, 1', 2, 2', \dots, k, k'$ or in the counter-clockwise fashion $1, k', k, \dots, 1'$. Let us, again without loss of generality, describe what follows in the first situation. The other one is in fact just a mirrored situation and the argumentation is thus almost the same. In the first situation, the images of $C_2^E$ are shifted clockwise by one, so the images of vertices of $C_2^E$ are consecutively $k', 1, 1', \dots, k$. This propagates the shifting further to $C_3^E$ and inductively up to $C_k^E$, also thank to the enforcing gadgets between the layers.

Up to now, everything was enforced, all vertices of the edge gadget are mapped. However, it remains to check two things.

1. Whether all the vertices identified with leafs of vertex gadgets are mapped in the right way.
2. Whether the edges between $C_1^E$ and $C_k^E$ and their endpoints do have the right images.

Regarding (1), we assumed $a$ is mapped to 1. Thus the other, white leaf from the pair containing $a$ has to be mapped to $1'$. This is indeed all right considering the shifting of images between the cycles. The same can be argued for the pair of leafs corresponding to the second vertex gadget attached. The argumentation here is based on the fact that the black leaf of the other vertex gadget is identified with vertex at distance $2^{\alpha+1}$ on $C_1^E$.

Case (2) depends on analogous argument, again based on shifting of the images of $C_1^E$ versus the images of $C_k^E$.

Finally, let us observe that $G'$ is again bipartite and thus it remains valid to speak about black and white leafs of vertex gadgets (or about vertices corresponding to such leafs). Furthermore, we can say that $G'$ has black and white vertices.

Suppose that $G'$ covers the $H$. Clearly, as was described and without loss of generality, all vertices corresponding originally to the black leafs of all vertex gadgets in $G$ must be mapped to the black vertices of $H$ and white leafs to the white vertices of $H$. We can further assume without loss of generality that black vertices of vertex gadgets map to vertices $1+(j-1)(2^{\alpha+1})$ where $j \in \{1, \dots, 2\beta-3\}$ of $H$.

Edge gadgets enforce that whenever there is an edge in $G$, then the images of the respective black leafs of vertex gadgets connected to it are exactly at distance $2^{\alpha+1}$ from each other. This implies that there is a homomorphism of $G$ to $C_{(2\beta+3)}$ and we can say that the image of $v$ is the $j$-th vertex of $C_{(2\beta+3)}$ if black leafs of vertex gadget of $v$ map to $1 + (j-1)(2^{\alpha+1})$.

For the other direction, suppose there exists a homomorphism of $G$ to $C_{(2\beta+3)}$. If a vertex $v$ of $G$ is mapped to the $j$-th vertex of $C_{(2\beta+3)}$, we map the vertices corresponding to the black leafs of the vertex gadget of $v$ to the vertex $1 + (j-1)(2^{\alpha+1})$ of $H$. We already know that this ensures that there is only one possible mapping of vertices of the vertex gadget of $v$ to $H$. The same can be done and said for all the other vertex gadgets of $G'$. From the analysis of possible mappings edge gadgets, we know than we can complete the mapping now so that the result is a covering projection of $G'$ to $H$.

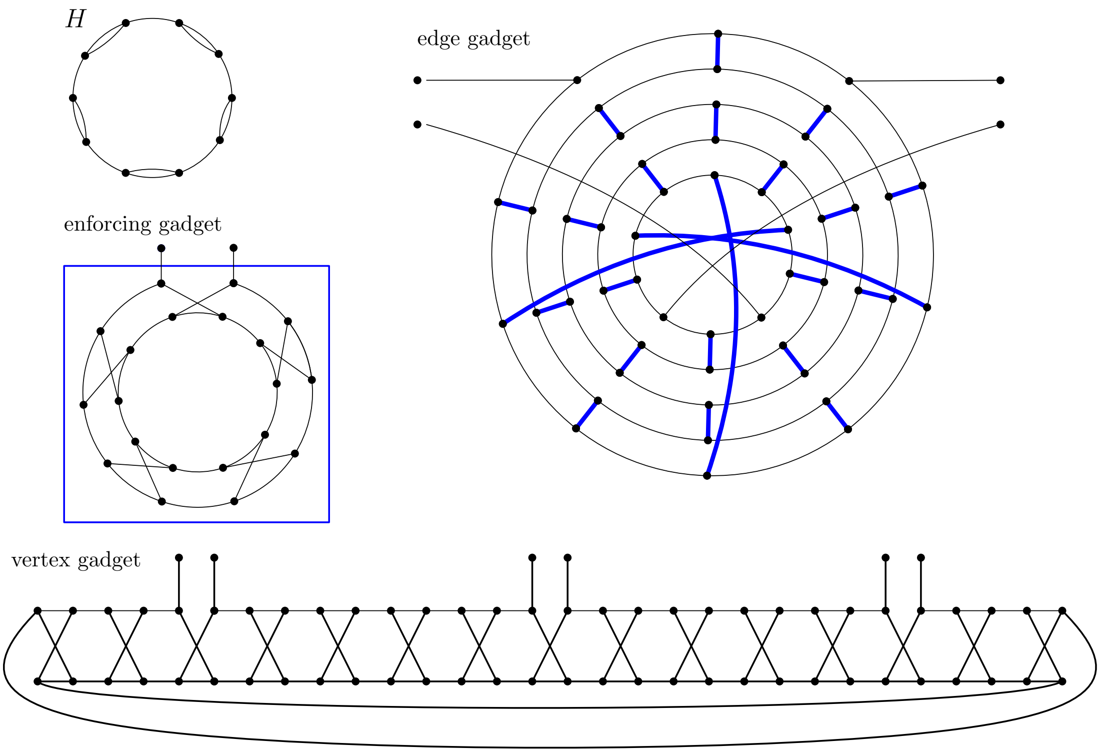


Figure 8.6: An example of the construction for $H$ being 5-ring.

We showed that there is a homomorphism of $G$ to $C_{(2\beta+3)}$ if and only if $G'$ covers $H$. This completes the reduction and thus the proof of the theorem. □

**Theorem 8.9.** *The* List-$k$-ring-Cover *problem is NP-complete for simple input graphs for every $k = 2^{\alpha}$ such that $\alpha \geq 3$ is an integer.*

*Proof.* By a seminal result of Feder et al. [64], List-$H'$-Hom is NP-complete if $H'$ is a bi-arc graph (Definition 2.1). By Corollary 3.1. therein, it follows that all cycles of size at least five are bi-arc graphs. Thus, we shall reduce from List-$C_k$-Hom.

The construction of $(G', L')$ for a given $(G, L)$ being an instance of List-$C_k$-Hom is almost the same as in Theorem 8.8. We shall only describe differences here. Suppose that $C_k$ has its vertices named consecutively $1, 2, \ldots, k$. Again as before, the $k$-ring will have its vertices denoted consecutively by $1, 1', \ldots, k, k'$ with precisely edges $jj'$ being double for $j \in \{1, \ldots, k\}$.

- Suppose we have a vertex $v$ in $G$. Then for every $\ell \in L(v)$, we add $\ell$ to the lists of all black leafs of its vertex gadget and $\ell'$ to all lists of all white leafs. All of the other vertices in $G'$ will have full lists, i.e. all vertices of $k$-ring. Also the edges will have full lists.
- Vertex gadgets will be connected to the respective edge gadget in a slightly

different way. For an edge $uv$ of $G$, we shall identify the black leaf of a pair of a vertex gadget for $u$ with first vertex of $C_1^E$ and the white leaf of the same pair with the $(1+k)$-th vertex of $C_k^E$ of the edge gadget for $uv$. We shall identify the black leaf of a pair of a vertex gadget for $v$ with third vertex of $C_1^E$ and the white leaf of the same pair with the $(3+k)$-th vertex of $C_k^E$ of the edge gadget for $uv$.

Clearly, under any possible list covering projection, all black leafs of a given vertex gadget choose simultaneously one vertex $\ell$ from their lists and subsequently $\ell'$ for all its white leafs (or vice versa).

Now the same analysis can be done as in Theorem 8.8 to show that there exists a list homomorphism of $(G, L)$ to $C_k$ if and only if there exists a list covering projection of $(G', L')$ to $k$-ring. This completes the proof. □

**Theorem 8.10.** *The 4-Ring-Cover problem is NP-complete for simple input graphs.*

*Proof.* The case of 4-rings needs a special ad hoc construction. However, the ideas are very similar to the previous ones. In order to not repeat ourselves, we shall describe a sketch of the reduction with the help of Figure 8.7 (referred to as figure in the rest of the proof). This time, we shall reduce from the problem of 4-Colouring.

Suppose we have a graph $G$ as an instance of 4-Colouring. We will construct $G'$ as an instance of 4-Ring-Cover in the following way. First, we shall insert into $G'$ vertex gadgets, the same ones as in the previous reductions on rings as one can see in the figure.

We introduce two new gadgets that will form our edge gadgets here: *1-gadget* and *0-1-gadget*. We shall also use the enforcing gadget we introduced in Theorem 8.8. The figure shows examples of explicit constructions of such gadgets. For simplicity, let us denote the vertices outside the respective boxes of gadgets (which are technically in the gadgets) as *terminal vertices* or *terminals*.

Let us now discuss the effect of the newly introduced gadgets on its terminal vertices.

- 1-gadget: If the left terminal vertices have $i$ and $i'$ as images, the right terminal vertices will have $i+1$ and $(i+1)'$ as its images (where numbers are taken modulo 4).
- 0-1 gadget: This gadget has a similar effect. If the left terminals have $i$ and $i'$, the right terminals will have either $i$ and $i'$, or $i+1$ and $(i+1)'$ as its images.

We omit the case analysis showing that these gadgets and their terminals indeed admit only the aforementioned images. Observe that enforcing gadgets are vital

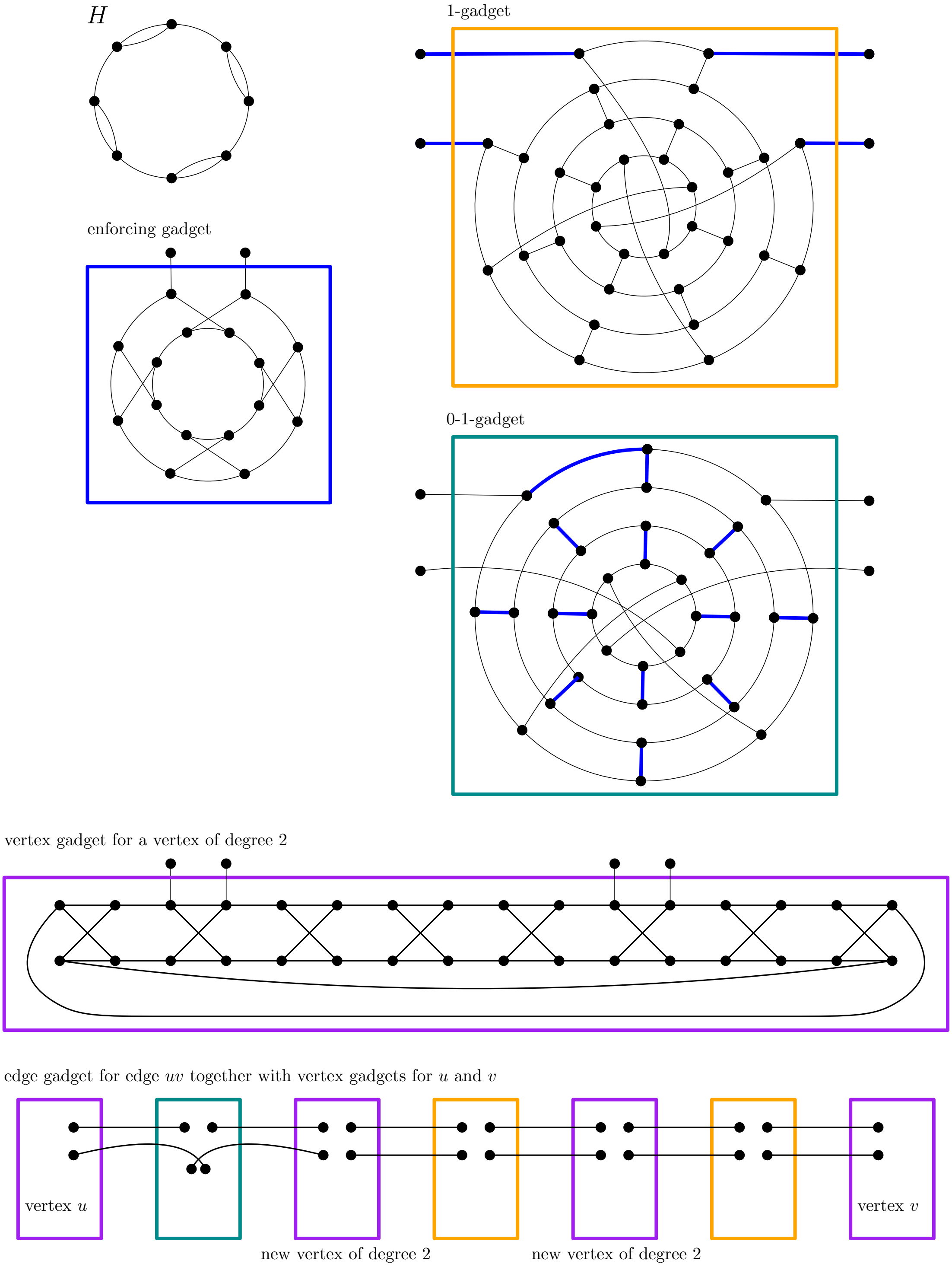

$H$
1-gadget
enforcing gadget
0-1-gadget
vertex gadget for a vertex of degree 2
edge gadget for edge $uv$ together with vertex gadgets for $u$ and $v$
vertex $u$
vertex $v$
new vertex of degree 2
new vertex of degree 2

Figure 8.7: An example of the construction for $H$ being 4-ring.

here and they also significantly simplify that analysis.

Put all together into the edge gadget (depicted in the lower part of the figure), the effect is the following:

1. Vertex gadget on the left has, without loss of generality, 1 and 1' on its terminal vertices.
2. Next, 1-gadget assures that its other terminal vertices will get 2 and 2'.
3. The next gadget is a vertex gadget. It copies the value 2 and 2'.
4. Now 0-1 gadget ensures that the other terminal vertices are either 2 and 2', or 3 and 3'.
5. This is repeated once again, resulting in terminal vertices of the final vertex gadget for the vertex $v$ not having 1 and 1' as images.

It is important to note that the edge gadget is symmetric in the sense that we could in the beginning set the image of the vertex gadget for $v$ and the rest would propagate in the similar manner, so that the terminal vertices of $u$ do not have the same "colour".

The whole construction is again bipartite and the colours of $G$ are encoded as the pairs $i, i'$ of vertices of 4-ring, where $i \in \{1, 2, 3, 4\}$. From the construction of the vertex gadget, we may assume that the white terminal vertices of vertex gadgets have the non-primed vertices of 4-ring as its images.

From the preceding discussion, it is clear that if $G'$ covers 4-ring, then $G$ is 4-colourable.

On the other hand, suppose that $G$ is 4-colourable. Then set the terminal vertices of vertex gadgets of $G'$ according to the colouring. We already know from the previous reductions that vertex gadget can be labelled so that the definition of covering projection is not violated. Also, it is now clear that the vertices of "edge gadgets" of $G'$ can be independently on each other labelled so that the $G'$ in the end covers 4-ring. This completes the sketch of the reduction. □

The following observation shows that hardness for $k$-rings implies the hardness for $k$-sausages.

**Corollary 8.11.** *For every $k \geq 2$ and every $k$-sausage $S_k$, $k$-RING-COVER $\propto$ $S_k$-COVER and LIST-$k$-RING-COVER $\propto$ LIST-$S_k$-COVER.*

*Proof.* A graph $G$ covers $H \times K_2$ if and only if it is bipartite and covers $H$. Since bipartiteness can be tested in polynomial time, testing if $G$ covers the $k$-ring polynomially reduces to testing if $G$ covers $S_k$.

With the list version, one has to be a bit more explicit. Let $S_k = (V, E)$, $V(K_2) = \{b, w\}$ and let the $k$-ring be denoted by $R_k$, with $V(R_k) = \{(u, \alpha) : u \in V, \alpha \in$

$\{b, w\}\}$. After checking that the input graph $G$ of the List-$R_k$-Cover problem is bipartite, let $V(G) = A \cup B$ be the bipartition of $G$. In a feasible covering projection, either the vertices of $A$ are mapped onto the vertices of $V \times \{w\}$ and the vertices of $B$ onto $V \times \{b\}$, or vice-versa. We try these two possibilities separately. For trying the former one, reduce first the lists to $L'_u = L_u \cap (V \times \{w\})$ for $u \in A$ and $L'_u = L_u \cap (V \times \{b\})$ for $u \in B$, and adjust the lists for edges accordingly. Then regard $G$ as an instance of List-$S_k$-Cover with the lists $\widetilde{L}_u = \{x : (x, w) \in L'_u \text{ or } (x, b) \in L'_u\}$, and lists for edges being adjusted accordingly. It is not difficult to see that $G$ allows a covering projection onto $R_k$ that respects $\mathcal{L}'$ if and only if it allows a covering projection onto $S_k$ that respects $\widetilde{\mathcal{L}}$. Check the latter possibility in a similar way and conclude that $G, \mathcal{L}$ is an feasible instance of List-$R_k$-Cover if and only if at least one of the $G, \widetilde{\mathcal{L}}$ instances is feasible for the corresponding List-$S_k$-Cover problem. □

## 8.5 Strong Dichotomy for cubic graphs

In this section we prove the Strong Dichotomy Conjecture for cubic graphs.

**Theorem 8.12.** *Let $H$ be a connected cubic graph. Then* List-$H$-Cover *is polynomial-time solvable for general graphs when $H$ has only one vertex and at most one semi-edge, and it is NP-complete even for simple input graphs otherwise.*

*Proof.* The proof is divided into several cases, depending on the structure of $H$.

*Case 1: We have* $|V(H)| = 1$. We distinguish two subcases.

*Case 1A - The graph $H$ has one semi-edge and one loop.* The preimage of the semi-edge should be a disjoint union of the semi-edges of the input graph $G$ and of a perfect matching on the vertices not incident to a semi-edge. Then the remaining edges of $G$ form a spanning collection of cycles (including loops) which form the preimage of the loop. The existence of a spanning subgraph of $G$ that is a preimage of the semi-edge can be tested in polynomial time.

If lists are present as part of the input, the situation gets a little more tricky. We start with a preprocessing phase. We check the below conditions:

(a) $G$ has a vertex or an edge with an empty list.

(b) $G$ has a vertex incident to two or more semi-edges,

(c) $G$ has a semi-edge whose list does not contain the semi-edge of $H$,

(d) $G$ has a vertex incident to a semi-edge and an edge, whose list does not contain the loop of $H$,

(e) $G$ has a vertex incident to two ordinary edges, whose lists do not contain the loop of $H$,

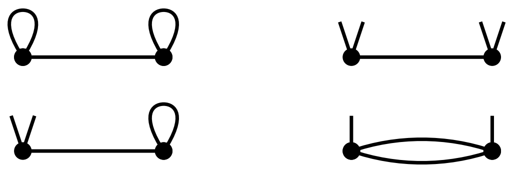


Figure 8.8: The non-bipartite 2-vertex graphs.

(f) $G$ has a loop whose list does not contain the loop of $H$.

It is clear that if any of the above conditions is satisfied, then $(G, \mathcal{L})$ is a no-instance. Thus we reject and quit.

Now we shall construct an auxiliary graph $G'$. We start our construction with $G$ and perform the following steps.

1. If some vertex $v$ is incident to a semi-edge, then delete $v$ with all its edges.
2. If some edge $e$ does not have the semi-edge of $H$ in its list, remove $e$ from the graph.
3. If some edge $e$ does not have the loop of $H$ in the list, leave $e$, but remove all edges incident to $e$.

Let $G'$ be the graph after the exhaustive application of steps 1, 2, and 3. It is straightforward to verify that steps 1 and 2 ensure that the union of a perfect matching in $G'$ and the semi-edges removed in step 1. can be a preimage of the semi-edge of $H$. Furthermore, by step 3 we ensure that if some edge has to be mapped to the semi-edge, then it will be so.

We can verify in polynomial time if $G'$ has a perfect matching. If not, we reject and quit. So let $M$ be a perfect matching in $G'$, and let $M'$ be the union of $M$ and the set of semi-edges removed in step 1. Observe that the graph $G - M$ is 2-regular, in other words a disjoint union of cycles (including loops). Furthermore, every edge of $G - M$ has the loop of $H$ in its list, this is guaranteed by step 3 and the preprocessing phase. Thus in this case we report a yes-instance.

*Case 1B: The graph $H$ has three semi-edges.* In this case already $H$-Cover is NP-complete, as it is equivalent to 3-edge-colourability of cubic graphs.

*Case 2: We have $|V(H)| = 2$.* If $H$ has neither loops nor semi-edges, then $H$ is a bipartite graph formed by a triple edge between two vertices. Only bipartite graphs can cover a bipartite one. Hence a covering projection corresponds to a 3-edge-colouring of the input graph. Thus $H$-Cover is polynomial-time solvable (every cubic bipartite graph is 3-edge-colourable), but List-$H$-Cover is NP-complete, because List 3-colouring is NP-complete for line graphs of cubic bipartite graphs [74]. If $H$ has a loop or a semi-edge, then it is one of the four graphs in Figure 8.8, and for each of these, already the $H$-Cover problem is NP-complete by the results in Chapter 6.

*Case 3:* $|V(H)| \geq 3$. Here we split into several subcases.

- *Case 3A: The graph $H$ is acyclic.* If we shave all semi-edges off from $H$, we get a tree with at least three vertices. At least one of them has degree greater than 1, and such vertex is semi-simple in $H$. Thus List-$H$-Cover is NP-complete by Theorem 8.2, as we remarked earlier.

- *Case 3B: The graph $H$ has a cycle of length greater than 2 which does not span all of its vertices.* Then $H$ has a vertex outside of this cycle, and thus $H$ has a semi-simple vertex and List-$H$-Cover is NP-complete.

- *Case 3C: The graph $H$ has a cycle of length greater than 2 with a diagonal.* Then again $H$ has a semi-simple vertex and List-$H$-Cover is NP-complete.

- *Case 3D: The graph $H$ has a cycle of length greater than 2, but none of the previous cases apply.* Then $H$ is the $k$-ring for some $k \geq 2$. If $k = 2$, 2-ring-Cover is NP-complete by Proposition 6.13. If $k \neq 2^\alpha$ for some $\alpha \geq 2$, $k$-ring-Cover is NP-complete by Theorem 8.8. In the case of $k = 2^\alpha$ with $\alpha \geq 3$, the List-$k$-ring-Cover problem is NP-complete by Theorem 8.9. The case of $k = 4$ is handled by Theorem 8.10.

- *Case 3E - $H$ has a cycle, but all cycles are of length one or two.* If, in addition, $H$ has no semi-simple vertex, then $H$ is a $k$-sausage for some $k \geq 2$. The NP-completeness of List-$H$-Cover follows from Case 3D via Proposition 8.11. □

## 8.6 Concluding remarks

We have studied the complexity of the List-$H$-Cover problem in the setting of graphs with multiple edges, loops, and semi-edges for regular target graphs. We have shown in Theorem 8.2 a general hardness result under the assumption that the target graph contains at least one semi-simple vertex. It is worthwhile to note that in fact we have proved the NP-hardness for the more specific $H$-Precovering Extension problem, when all the lists are either one-element, or full. Actually, we proved hardness for the even more specific Vertex $H$-Precovering Extension version, when only vertices may come with prescribed covering projections, but all edges have full lists.

On the contrary, the nature of the NP-hard cases that appear in the characterization of the complexity of List-$H$-Cover of cubic graphs given by Theorem 8.12 is more varied. Some of them are NP-hard already for $H$-Cover, some of them are NP-hard for $H$-Precovering Extension, but apart from the Vertex $H$-Precovering Extension version in applications of Theorem 8.2, this time we also utilize the Edge $H$-Precovering Extension version for the case of the bipartite 2-vertex graph formed by a triple edge between two vertices. Finally, for the cases of sausages and rings of length power of two, nontrivial lists are required to make our proof technique work.

# Part III

# Acyclic colouring

# Chapter 9

# Acyclic colouring and its complexity for $H$-free graphs

This chapter is based on:

- [32] Jan Bok, Nikola Jedličková, Barnaby Martin, Daniël Paulusma, and Siani Smith: *Acyclic, Star and Injective Colouring: A Complexity Picture for H-Free Graphs.* In 28th Annual European Symposium on Algorithms, ESA 2020, volume 173 of Leibniz International Proceedings in Informatics (LIPIcs), pages 173:22:1–22:22, 2020.
- [31] Jan Bok, Nikola Jedličková, Barnaby Martin, Pascal Ochem, Daniël Paulusma, and Siani Smith: *Acyclic, Star and Injective Colouring: A Complexity Picture for H-Free Graphs.* Submitted, 2021. `https://arxiv.org/abs/2008.09415`

**Note on the organisation of this chapter.** The introductory section follows the papers on which the chapter is based and discusses not only acyclic but also star and injective colouring and the results we have obtained in [32] and then in [31]. However, in Section 9.2, we shall focus further just on acyclic colouring as this was the part of the papers on which the author of this thesis worked the most.

## 9.1 Introduction

We study the complexity of three classical colouring problems. We do this by focusing on *hereditary* graph classes, i.e., classes closed under vertex deletion, or equivalently, classes characterized by a (possibly infinite) set $\mathcal{F}$ of forbidden induced subgraphs. As evidenced by numerous complexity studies in the literature, even the case where $|\mathcal{F}| = 1$ captures a rich family of graph classes suitably interesting to develop general methodology. Hence, we usually first set $\mathcal{F} = \{H\}$ and consider the class of *$H$-free* graphs, i.e., graphs that do not contain $H$ as an induced subgraph. We then investigate how the complexity of a problem restricted to $H$-free graphs depends on the choice of $H$ and try to obtain a *complexity dichotomy.*

To give a well known and relevant example, the Colouring problem is to decide, given a graph $G$ and integer $k \geq 1$, if $G$ has a *$k$-colouring*, i.e., a mapping $c\colon V(G) \to \{1, \ldots, k\}$ such that $c(u) \neq c(v)$ for every two adjacent vertices $u$ and $v$. Král' et al. [120] proved that Colouring on $H$-free graphs is polynomial-time solvable if $H$ is an induced subgraph of $P_4$ or $P_1 + P_3$ and NP-complete otherwise. Here, $P_n$ denotes the $n$-vertex path and $G_1 + G_2 = (V(G_1) \cup V(G_2), E(G_1) \cup E(G_2))$ the disjoint union of two vertex-disjoint graphs $G_1$ and $G_2$. If $k$ is fixed (not part of the input), then we obtain the $k$-Colouring problem. No complexity dichotomy is known for $k$-Colouring if $k \geq 3$. In particular, the complexities of 3-Colouring for $P_t$-free graphs for $t \geq 8$ and $k$-Colouring for $sP_3$-free graphs for $s \geq 2$ and $k \geq 4$ are still open. Here, we write $sG$ for the disjoint union of $s$ copies of $G$. We refer to the survey of Golovach et al. [90] for further details and to [52, 118] for updated summaries.

For a colouring $c$ of a graph $G$, a *colour class* consists of all vertices of $G$ that are mapped by $c$ to a specific colour $i$. We consider the following special graph colourings. A colouring of a graph $G$ is *acyclic* if the union of any two colour classes induces a forest. The $(r+1)$-vertex *star* $K_{1,r}$ is the graph with vertices $u, v_1, \ldots, v_r$ and edges $uv_i$ for every $i \in \{1, \ldots, r\}$. An acyclic colouring is a *star colouring* if the union of any two colour classes induces a *star forest*, that is, a forest in which each connected component is a star. A star colouring is *injective* (or an *$L(1,1)$-labelling* or a *distance-2 colouring*) if the union of any two colour classes induces an $sP_1 + tP_2$ for some integers $s \geq 0$ and $t \geq 0$. An alternative definition is to say that all the neighbours of every vertex of $G$ are uniquely coloured. See an example in Figure 9.1 These definitions lead to the following three decision problems:

Problem: Acyclic Colouring
Input: A graph $G$ and an integer $k \geq 1$.
Question: Does $G$ have an acyclic $k$-colouring?

Problem: Star Colouring
Input: A graph $G$ and an integer $k \geq 1$.
Question: Does $G$ have a star $k$-colouring?

Problem: Injective Colouring
Input: A graph $G$ and an integer $k \geq 1$.
Question: Does $G$ have an injective $k$-colouring?

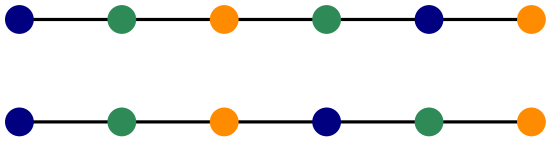


Figure 9.1: The upper path is 3-coloured and the colouring satisfies the property of being a star colouring but not an injective colouring. The 3-colouring of the lower path is an injective colouring and hence also a star colouring.

If $k$ is fixed, we write Acyclic $k$-Colouring, Star $k$-Colouring and Injective $k$-Colouring, respectively.

All three problems have been extensively studied. We note that in the literature on the Injective Colouring problem it is often assumed that two adjacent vertices may be coloured alike by an injective colouring (see, for example, [98, 106, 113]). However, here, we do **not** allow this; as reflected in their definitions we only consider colourings that are proper. This enables us to compare the results for the three different kinds of colourings with each other.

So far, systematic studies mainly focused on structural characterizations, exact values, lower and upper bounds on the minimum number of colours in an acyclic colouring or star colouring (i.e., the *acyclic* and *star chromatic number*); see, e.g., [4, 34, 70, 71, 72, 115, 116, 163, 170, 172], to name just a few papers, whereas injective colourings (and the *injective chromatic number*) were mainly considered in the context of the distance constrained labelling framework (see the survey [46] and Section 9.3 therein). The problems have also been studied from a complexity perspective, but apart from a study on Acyclic Colouring for graphs of bounded maximum degree [146], known results are scattered and relatively sparse. We perform a *systematic* and *comparative* complexity study by focusing on the following research question both for $k$ part of the input and for fixed $k$:

*What are the computational complexities of* Acyclic Colouring, Star Colouring *and* Injective Colouring *for $H$-free graphs?*

### 9.1.1 Known results

Before discussing our results and techniques, we first briefly discuss some known results.

Coleman and Cai [53] proved that for every $k \geq 3$, Acyclic $k$-Colouring is NP-complete for bipartite graphs. Afterwards, a number of hardness results appeared for other hereditary graph classes. Alon and Zaks [5] showed that Acyclic 3-Colouring is NP-complete for line graphs of maximum degree 4. Kostochka [119] proved that Acyclic 3-Colouring is NP-complete for planar graphs. This result was improved to planar bipartite graphs of maximum degree 4 by Ochem [151]. Mondal et al. [146] proved that Acyclic 4-Colouring is NP-complete for graphs of maximum degree 5 and for planar graphs of maximum degree 7. Ochem [151] showed that Acyclic 4-Colouring is NP-complete for planar bipartite graphs of maximum degree 8. We refer to the paper of Angelini and Frati [6] for a further discussion on acyclic colourable planar graphs.

Albertson et al. [2] and Lei et al. [130] proved that Star 3-Colouring is NP-complete for planar bipartite graphs and line graphs, respectively. Shalu and Antony [158] showed that Star Colouring is NP-complete for co-bipartite graphs. Bodlaender et al. [21], Sen and Huson [157] and Lloyd and Ramanathan [135] proved that Injective Colouring is NP-complete for split graphs, unit disk graphs and planar graphs, respectively. Mahdian [142] proved that for every $k \geq 4$, Injective $k$-Colouring is NP-complete for line graphs, whereas Injective 4-Colouring is also known to be NP-complete for cubic graphs (see [46]). Observe that Injective 3-Colouring is trivial for general graphs.

On the positive side, Lyons [139] proved that Acyclic Colouring and Star Colouring are polynomial-time solvable for $P_4$-free graphs; in particular, he showed that every acyclic colouring of a $P_4$-free graph is, in fact, a star colouring. We note that Injective Colouring is trivial for $P_4$-free graphs, as every injective colouring must assign each vertex of a connected $P_4$-free graph a unique colour. Afterwards, the results of Lyons have been extended to $P_4$-tidy graphs and $(q, q-4)$-graphs by Linhares-Sales et al. [133].

Cheng et al. [51] complemented the aforementioned result of Alon and Zaks [5] by proving that Acyclic Colouring is polynomial-time solvable for claw-free graphs of maximum degree at most 3. Calamoneri [46] observed that Injective Colouring is polynomial-time solvable for comparability and co-comparability graphs. Zhou et al. [171] proved that Injective Colouring is polynomial-time solvable for graphs of bounded treewidth (which is best possible due to the W[1]-hardness result of Fiala et al. [75]).

Finally, we refer to [36] for a complexity study of Acyclic Colouring, Star Colouring and Injective Colouring, for graphs of bounded diameter.

### 9.1.2 Our complexity results and methodology

The *girth* of a graph $G$ is the length of a shortest cycle of $G$ (if $G$ is a forest, then its girth is $\infty$). To answer our research question we focus on two important graph classes, namely the classes of graphs of high girth and line graphs of multigraphs, which are interesting classes on their own. If a problem is NP-complete for both classes, then it is NP-complete for $H$-free graphs whenever $H$ has a cycle or a claw. It then remains to analyze the case when $H$ is a *linear forest*, i.e., a disjoint union of paths; see [33, 42, 86, 120] for examples of this approach, which we discuss in detail below.

The construction of graph families of high girth and large chromatic number is well studied in graph theory (see, e.g. [62]). To prove their complexity dichotomy for Colouring on $H$-free graphs, Král' et al. [120] first showed that for every integer $g \geq 3$, 3-Colouring is NP-complete for the class of graphs of girth at least $g$. This approach can be readily extended to any integer $k \geq 3$ [61, 137]. The basic idea is to replace edges in a graph by graphs of high girth and large chromatic number, such that the resulting graph has sufficiently high girth and is $k$-colourable if and only if the original graph is so (see also [91, 110]).

By a more intricate use of the above technique we are able to prove that for every $g \geq 3$ and every $k \geq 3$, Acyclic $k$-Colouring is NP-complete for the class of 2-degenerate bipartite graphs of girth at least $g$. This implies that Acyclic $k$-Colouring is NP-complete for $H$-free graphs whenever $H$ has a cycle. We are also able to prove that for every $g \geq 3$, Star 3-Colouring remains NP-complete even for planar graphs of girth at least $g$ and maximum degree 3. This implies that Star 3-Colouring is NP-complete for $H$-free graphs whenever $H$ has a cycle. We prove the latter result for every $k \geq 4$ by combining known results, just as we use known results to prove that Injective $k$-Colouring ($k \geq 4$) is NP-complete for $H$-free graphs if $H$ has a cycle.

A classical result of Holyer [109] is that 3-Colouring is NP-complete for line graphs (and Leven and Galil [132] proved the same for $k \geq 4$). As line graphs are claw-free, Král' et al. [120] used Holyer's result to show that 3-Colouring is NP-complete for $H$-free graphs whenever $H$ has an induced claw. For Acyclic $k$-Colouring, we can use Alon and Zaks' result [5] for $k = 3$, which we extend to work for $k \geq 4$. For Star $k$-Colouring we extend the recent result of Lei et al. [130] from $k = 3$ to $k \geq 3$ (in both our results we consider line graphs of multigraphs; these graphs are claw-free and hence suffice for our study on $H$-free graphs). For Injective $k$-Colouring ($k \geq 4$) we can use the aforementioned result on line graphs of Mahdian [142].

The above hardness results leave us to consider the case where $H$ is a linear forest. In Section 9.2.1 we will use a result of Atminas et al. [11] to prove a general result from which it follows that for fixed $k$, all three problems are polynomial-time solvable for $H$-free graphs if $H$ is a linear forest. Hence, we have full complexity dichotomies for the three problems when $k$ is fixed. However, these positive results do not extend to the case where $k$ is part of the input. That is, for each of the three problems, we prove NP-completeness for graphs that are $P_r$-free for some small value of $r$ or have a small independence number, i.e., that are $sP_1$-free for some small integer $s$.

Our complexity results for $H$-free graphs are summarized in the following three theorems; see Table 9.1 for a comparison. For two graphs $F$ and $G$, we write $F \subseteq_i G$ or $G \supseteq_i F$ to denote that $F$ is an *induced* subgraph of $G$. The rest of this chapter will be devoted to the proof of Theroem 9.1.

| | polynomial time | NP-complete |
|---|---|---|
| Colouring [120] | $H \subseteq_i P_4$ or $P_1 + P_3$ | else |
| Acyclic Colouring | $H \subseteq_i P_4$ | else except for open cases $H$ being linear forests and $H \not\supseteq 6P_1$ |
| Star Colouring | $H \subseteq_i P_4$ | else except for 1 open case: $H = 2P_2$ |
| Injective Colouring | $H \subsetneq_i 2P_1 + P_4$ | else except for 1 open case: $H = 2P_1 + P_4$ |
| $k$-Colouring (see [52, 90, 118]) | depends on $k$ | infinitely many open cases for all $k \geq 3$ |
| Acyclic $k$-Colouring ($k \geq 3$) | $H$ is a linear forest | else |
| Star $k$-Colouring ($k \geq 3$) | $H$ is a linear forest | else |
| Injective $k$-Colouring ($k \geq 4$) | $H$ is a linear forest | else |

Table 9.1: The state-of-the-art for the three problems in this chapter and the original Colouring problem; both when $k$ is fixed and part of the input.

**Theorem 9.1.** *Let $H$ be a graph. For the class of $H$-free graphs, it holds that:*

*(i)* Acyclic Colouring *is polynomial-time solvable if $H \subseteq_i P_4$ and NP-complete if $H$ is not a linear forest or $H \supseteq_i 6P_1$;*

*(ii) for every $k \geq 3$,* Acyclic $k$-Colouring *is polynomial-time solvable if $H$ is a linear forest and NP-complete otherwise.*

**Theorem 9.2.** *[31] Let $H$ be a graph. For the class of $H$-free graphs, it holds that:*

*(i)* Star Colouring *is polynomial-time solvable if $H \subseteq_i P_4$ and NP-complete if $H \not\subseteq_i P_4$ and $H \neq 2P_2$;*

*(ii) for every $k \geq 3$,* Star $k$-Colouring *is polynomial-time solvable if $H$ is a linear forest and NP-complete otherwise.*

**Theorem 9.3.** *[31] Let $H$ be a graph. For the class of $H$-free graphs, it holds that:*

*(i)* Injective Colouring *is polynomial-time solvable if $H \subsetneq_i 2P_1 + P_4$ and NP-complete if $H \not\subseteq_i 2P_1 + P_4$;*

*(ii) for every $k \geq 4$,* Injective $k$-Colouring *is polynomial-time solvable if*

*$H$ is a linear forest and NP-complete otherwise.*

## 9.2 Acyclic colouring

### 9.2.1 A general polynomial result

A *biclique* or *complete bipartite graph* is a bipartite graph on vertex set $S \cup T$, such that $S$ and $T$ are independent sets and there is an edge between every vertex of $S$ and every vertex of $T$; if $|S| = s$ and $|T| = t$, we denote this graph by $K_{s,t}$, and if $s = t$, the biclique is *balanced* and of *order $s$*. We say that a colouring $c$ of a graph $G$ satisfies the *balanced biclique condition* (BB-condition) if $c$ uses at least $k+1$ colours to colour $G$, where $k$ is the order of a largest biclique that is contained in $G$ as a (not necessarily induced) subgraph.

Let $\pi$ be some colouring property; e.g., $\pi$ could mean being acyclic, star or injective. Then $\pi$ *can be expressed in MSO$_2$* (monadic second-order logic with edge sets) if for every $k \geq 1$, the graph property of having a $k$-colouring with property $\pi$ can be expressed in MSO$_2$ (where $k$ is fixed). The general problem Colouring($\pi$) is to decide, on a graph $G$ and integer $k \geq 1$, if $G$ has a $k$-colouring with property $\pi$. If $k$ is fixed, we write $k$-Colouring($\pi$). We now prove the following result.

**Theorem 9.4.** *Let $H$ be a linear forest, and let $\pi$ be a colouring property that can be expressed in MSO$_2$, such that every colouring with property $\pi$ satisfies the BB-condition. Then, for every integer $k \geq 1$, $k$-*Colouring*($\pi$) is linear-time solvable for $H$-free graphs.*

*Proof.* Atminas, Lozin, and Razgon [11] proved that that for every pair of integers $\ell$ and $k$, there exists a constant $b(\ell, k)$ such that every graph of treewidth at least $b(\ell, k)$ contains an induced $P_\ell$ or a (not necessarily induced) biclique $K_{k,k}$. Let $G$ be an $H$-free graph, and let $\ell$ be the smallest integer such that $H \subseteq_i P_\ell$; observe that $\ell$ is a constant. Hence, we can use Bodlaender's algorithm [20] to test in linear time if $G$ has treewidth at most $b(\ell, k) - 1$.

First suppose that the treewidth of $G$ is at most $b(\ell, k) - 1$. As $\pi$ can be expressed in MSO$_2$, the result of Courcelle [56] allows us to test in linear time whether $G$ has a $k$-colouring with property $\pi$. Now suppose that the treewidth of $G$ is at least $b(\ell, k)$. As $G$ is $H$-free, $G$ is $P_\ell$-free. Then, by the result of Atminas, Lozin and Razgon [11], we find that $G$ contains $K_{k,k}$ as a subgraph. As $\pi$ satisfies the BB-condition, $G$ has no $k$-colouring with property $\pi$. □

We now apply Theorem 9.4 to obtain the polynomial cases for fixed $k$ in Theorem 9.1.

**Corollary 9.5.** *Let $H$ be a linear forest. For every $k \geq 1$,* Acyclic $k$-Colouring *(and* Star $k$-Colouring *and* Injective $k$-Colouring*) are polynomial-time solvable for $H$-free graphs.*

*Proof.* All three kinds of colourings use at least $s$ colours to colour $K_{s,s}$ (as the vertices from one bipartition class of $K_{s,s}$ must receive unique colours). Hence, every acyclic and star and injective colouring of every graph satisfies the BB-condition. Moreover, it is readily seen that the colouring properties of being acyclic, star or injective can all be expressed in $\mathrm{MSO}_2$. Hence, we may apply Theorem 9.4. □

### 9.2.2 NP-hardness results and dichotomies

In this subsection, we prove Theorem 9.1. For a graph $G$ and a colouring $c$, the pair $(G, c)$ has a *bichromatic* cycle $C$ if $C$ is a cycle of $G$ with $|c(V(C)| = 2$, that is, the vertices of $C$ are coloured by two alternating colours (so $C$ is even). The notion of a *bichromatic path* is defined in a similar manner.

**Lemma 9.6.** *For every $k \geq 3$ and every $g \geq 3$,* Acyclic $k$-Colouring *is NP-complete for 2-degenerate bipartite graphs of girth at least $g$.*

*Proof.* We reduce from Acyclic $k$-Colouring, which is known to be NP-complete for bipartite graphs for every $k \geq 3$ [53]. Recall that the *arboricity* of a graph is the minimum number of forests needed to partition its edge set. By counting the edges, a graph with arboricity at most $t$ is $(2t-1)$-degenerate and thus $2t$-colourable. We start by taking a graph $F$ that has no $2k(k-1)$-colouring and that is of girth at least $g$. By a seminal result of Erdős [62], such a graph $F$ exists (and its size is constant, as it only depends on $g$ and $k$ which are fixed integers). Notice that $F$ does not admit a vertex-partition into $k$ subgraphs with arboricity at most $k-1$, since otherwise $F$ would be $2k(k-1)$-colourable.

Now we consider the graph $S$ obtained by subdividing every edge of $F$ exactly once. The graph $S$ is 2-degenerate and bipartite with the *old* vertices from $F$ in one part and the *new* vertices of degree 2 in the other part. Moreover, $S$ has girth at least $g$, as $F$ has girth at least $g$.

We claim that $S$ has no acyclic $k$-colouring. For contradiction, assume that $S$ has an acyclic $k$-colouring. Assign the colour of every old vertex to the corresponding vertex of $F$ and assign the colour of every new vertex to the corresponding edge of $F$. For every colour $i$, we consider the subgraph $F_i$ of $F$ induced by the vertices coloured $i$. For every $j \neq i$, the subgraph of $S$ induced by the colours $i$ and $j$ is a forest. This implies that the subgraph of $F_i$ induces by the edges coloured $j$ is a forest. So the arboricity of $F_i$ is at most $k-1$, that is, the number of colours distinct from $i$. By previous discussion, the chromatic number of $F_i$ is at most

$2(k-1)$, so that $F$ is $2k(k-1)$-colourable. This contradiction shows that $S$ has no acyclic $k$-colouring.

We repeatedly remove new vertices from $S$ until we obtain a graph $S'$ that is acyclically $k$-colourable. Note that $S'$ has girth at least $g$ and is 2-degenerate, as $S$ has girth at least $g$ and is 2-degenerate. Let $x_2$ be the last vertex that we removed and let $x_1$ and $x_3$ be the neighbours of $x_2$ in $S$. By construction, $S'$ is acyclically $k$-colourable and every acyclic $k$-colouring $c$ of $S'$ is such that:

- $c(x_1) = c(x_3)$, since otherwise setting $c(x_2) \notin \{c(x_1), c(x_3)\}$ would extend $c$ to an acyclic $k$-colouring of the larger graph, which is not possible by construction. Without loss of generality, $c(x_1) = c(x_3) = 1$.
- For every colour $i \neq 1$, $S'$ contains a bichromatic path coloured 1 and $i$ between $x_1$ and $x_3$, since otherwise setting $c(x_2) = i$ would extend $c$ to an acyclic $k$-colouring of the larger graph again.

We are ready to describe the reduction. Let $G$ be a bipartite instance of Acyclic $k$-Colouring. We construct an equivalent instance $G'$ with girth at least $g$ as follows. For every vertex $z$ of $G$, we fix an arbitrary order on the neighbours of $z$. We replace $z$ of $G$ by $d$ vertices $z_1, z_2, \ldots, z_d$, where $d$ is the degree of $z$. Then for $1 \leq i \leq d-1$, we take a copy of $S'$ and we identify the vertex $x_1$ of $S'$ with $z_i$ and the vertex $x_3$ of $S'$ with $z_{i+1}$. Now for every edge $uv$ of $G$, say $v$ is the $i^{th}$ neighbour of $u$ and $u$ is the $j^{th}$ neighbour of $v$, we add the edge $u_iv_j$ in $G'$. See also Figure 9.2.

Given an acyclic $k$-colouring of $G$, we assign the colour of $z$ to $z_1, \ldots, z_d$ and extend the colouring to the copies of $F'$, which gives an acyclic colouring of $G'$. Given an acyclic $k$-colouring of $G'$, the copies of $F'$ force the same colour on $z_1, \ldots, z_d$ and we assign this common colour to $z$, which gives an acyclic $k$-colouring of $G$.

Finally, notice that since $G$ and $S'$ are bipartite, $G'$ is bipartite. As $S'$ is 2-degenerate and has girth at least $g$, we find that $G'$ is 2-degenerate and has girth at least $g$. □

The *line graph* of a graph $G$ has vertex set $E(G)$ and an edge between two vertices $e$ and $f$ if and only if $e$ and $f$ share an end-vertex of $G$. We now modify the construction of [5] for line graphs from $k = 3$ to $k \geq 3$.

**Lemma 9.7.** *For every $k \geq 3$,* Acyclic $k$-Colouring *is NP-complete for line graphs of multigraphs.*

*Proof.* For an integer $k \geq 1$, a *$k$-edge colouring* of a graph $G = (V, E)$ is a mapping $c \colon E \to \{1, \ldots, k\}$ such that $c(e) \neq c(f)$ whenever the edges $e$ and $f$ share an end-vertex. A *colour class* consists of all edges of $G$ that are mapped by $c$ to a specific colour $i$. For a fixed integer $k \geq 1$, the Acyclic $k$-Edge Colouring

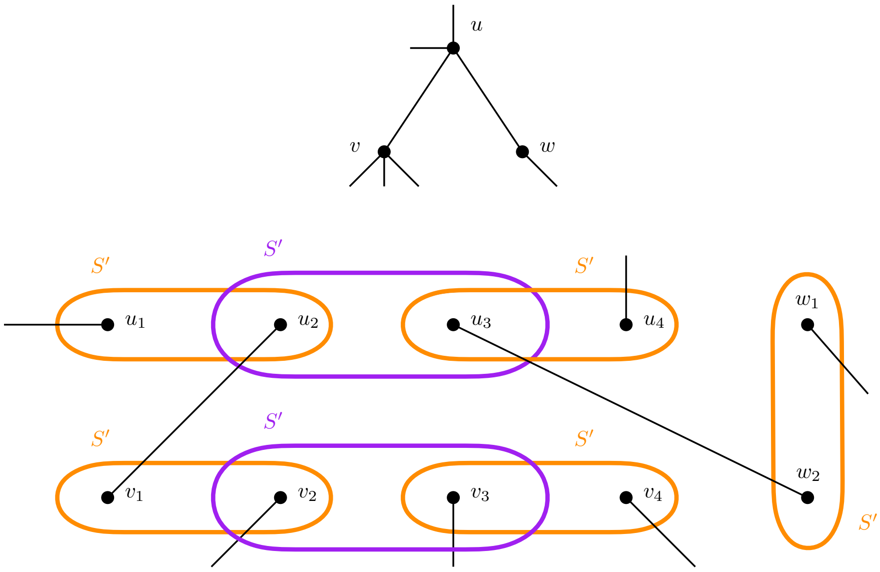


Figure 9.2: An example of part of a graph $G$ (top) and the corresponding part in $G'$ (bottom) from Lemma 9.6. In the part of $G'$ corresponding to vertex $u$, vertex $u_1$ is identified with $x_1$ of the left copy of $S'$; vertex $u_2$ with $x_3$ of the left copy of $S'$ and $x_1$ of the middle copy of $S'$; vertex $u_3$ with $x_3$ of the middle copy of $S'$ and $x_1$ of the right copy of $S'$; and $u_4$ with $x_3$ of the right copy of $S'$.

problem is to decide if a given graph has an acyclic $k$-edge colouring. Alon and Zaks proved that ACYCLIC 3-EDGE COLOURING is NP-complete. We note that a graph has an acyclic $k$-edge colouring if and only if its line graph has an acyclic $k$-colouring. Hence, it remains to generalize the construction of Alon and Zaks [5] from $k = 3$ to $k \geq 3$. Our main tool is the gadget graph $F_k$, depicted in Figure 9.3, about which we prove the following two claims.

(i) *The edges of $F_k$ can be coloured acyclically using $k$ colours, with no bichromatic path between $v_1$ and $v_{14}$.*

(ii) *Every acyclic $k$-edge colouring of $F_k$ using $k$ colours assigns $e_1$ and $e_2$ the same colour.*

We first prove (ii). We assume, without loss of generality, that $v_1v_2$ is coloured by 1, $v_2v_4$ by 2 and the edges between $v_2$ and $v_3$ by colours $3, \ldots, k$. The edge $v_3v_5$ has to be coloured by 1, otherwise we have a bichromatic cycle on $v_2v_3v_5v_4$. This necessarily implies that

- the edges between $v_4$ and $v_5$ are coloured by $3, \ldots, k$,
- the edge $v_5v_7$ is coloured by 2,
- the edge $v_4v_6$ is coloured by 1,
- the edges between $v_6$ and $v_7$ are coloured by $3, \ldots, k$, and

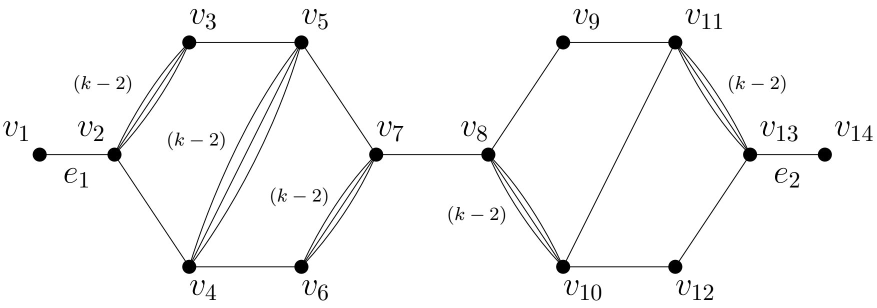


Figure 9.3: The gadget multigraph $F_k$. The labels on edges are multiplicities.

- the edge $v_7v_8$ is coloured by 1.

Now assume that the edge $v_8v_9$ is coloured by $a \in \{2, \dots, k\}$ and the edges between $v_8$ and $v_{10}$ by colours from the set $A$, where $A = \{2, \dots, k\} \setminus a$. The edge $v_{10}v_{11}$ is either coloured $a$ or 1. However, if it is coloured 1, $v_9v_{11}$ is assigned a colour $b \in A$ and necessarily we have either a bichromatic cycle on $v_8v_9v_{11}v_{13}v_{12}v_{10}$, coloured by $b$ and $a$, or a bichromatic cycle on $v_{10}v_{11}v_{13}v_{12}$, coloured by $a$ and 1. Thus $v_{10}v_{11}$ is coloured by $a$. To prevent a bichromatic cycle on $v_8v_9v_{11}v_{10}$, the edge $v_9v_{11}$ is assigned colour 1. The rest of the colouring is now determined as $v_{10}v_{12}$ has to be coloured by 1, the edges between $v_{11}$ and $v_{13}$ by $A$, $v_{12}v_{13}$ by $a$, and $v_{13}v_{14}$ by 1. We then have a $k$-colouring with no bichromatic cycles of size at least 3 in $F_k$ for every possible choice of $a$. This proves that $v_1v_2$ and $v_{13}v_{14}$ are coloured alike under every acyclic $k$-edge colouring.

We prove (i) by choosing $a$ different from 2. Then there is no bichromatic path between $v_1$ and $v_{14}$.

We now reduce from $k$-Edge-Colouring to Acyclic $k$-Edge Colouring as follows. Given an instance $G$ of $k$-edge Colouring we construct an instance $G'$ of Acyclic $k$-Edge Colouring by replacing each edge $uv$ in $G$ by a copy of $F_k$ where $u$ is identified with $v_1$ and $v$ is identified with $v_{14}$.

If $G'$ has an acyclic $k$-edge colouring $c'$ then we obtain a $k$-edge colouring $c$ of $G$ by setting $c(uv) = c'(e_1)$ where $e_1$ belongs to the gadget $F_k$ corresponding to the edge $uv$. If $G$ has a $k$-edge colouring $c$ then we obtain an acyclic $k$-edge colouring $c'$ of $G'$ by setting $c'(e_1) = c(uv)$ where $e_1$ belongs to the gadget corresponding to the edge $uv$. The remainder of each gadget $F_k$ can then be coloured as described above. □

We now focus on linear forests by proving that for $6P_1$-free graphs, Acyclic Colouring is NP-complete. This will allow us to prove Theorem 9.1. We first need to introduce some terminology and prove a lemma on Colouring. A $k$-colouring of $G$ can be seen as a partition of $V(G)$ into $k$ independent sets. Hence, a ($k$-)colouring of $G$ corresponds to a *($k$-)clique-covering* of $\overline{G}$, which is a partition of $V(\overline{G}) = V(G)$ into $k$ cliques. The *clique covering number* $\overline{\chi}(G)$ of $G$ is the smallest number of cliques in a clique-covering of $G$. Note that $\overline{\chi}(G) = \chi(\overline{G})$.

**Lemma 9.8.** Colouring *is NP-complete for graphs with* $\overline{\chi} \leq 3$.

*Proof.* The List Colouring problem takes as input a graph $G$ and a *list assignment* $L$ that assigns each vertex $u \in V(G)$ a list $L(u) \subseteq \{1, 2, \ldots\}$. The question is whether $G$ admits a colouring $c$ with $c(u) \in L(u)$ for every $u \in V(G)$. Jansen [111] proved that List Colouring is NP-complete for co-bipartite graphs. This is the problem we reduce from.

Let $G$ be a graph with a list assignment $L$ and assume that $V(G)$ can be split into two (not necessarily disjoint) cliques $K$ and $K'$. We set $A_1 = K$ and $A_2 = K \setminus K'$. As both $A_1$ and $A_2$ are cliques, we have that $\overline{\chi}(G) \leq 2$. We may assume without loss of generality that the union of all the lists $L(u)$ is $\{1, \ldots, k\}$ for some integer $k$. We now extend $G$ by adding a clique $A_3$ of $k$ new vertices $v_1, \ldots, v_k$ and by adding an edge between a vertex $x_\ell$ and a vertex $u \in V(G)$ if and only if $\ell \notin L(u)$. This yields a new graph $G'$ with $\overline{\chi}(G') \leq 3$. It is readily seen that $G$ has a colouring $c$ with $c(u) \in L(u)$ for every $u \in V(G)$ if and only if $G'$ has a $k$-colouring. □

We now use Lemma 9.8 to prove the hardness result for $6P_1$-free graphs.

**Lemma 9.9.** Acyclic Colouring *is NP-complete for* $6P_1$*-free graphs.*

*Proof.* We reduce from Colouring. Let $(G, k)$ be an instance of this problem. By Lemma 9.8 we may assume that $V(G)$ can be partitioned into three cliques $A_1$, $A_2$, and $A_3$. Let $E^* = \{e_1, \ldots, e_q\}$ be the set of edges in $G$ whose end-vertices belong to different cliques of $\{A_1, A_2, A_3\}$.

Let $K = kq + k$. We construct an instance $G'$ of Acyclic Colouring from $G$ such that $G$ is $k$-colourable if and only if $G'$ has an acyclic $K$-colouring. Our graph $G'$ contains 5 cliques: $A_1$, $A_2$, $A_3$, and two cliques $D$ and $S$ on $kq$ vertices. Every vertex of $D$ is adjacent to every vertex in $A_1$, $A_2$, and $A_3$. The vertices of $S$ are called $s_i^j$ with $1 \leq i \leq k$ and $1 \leq j \leq q$. Every vertex $s_i^j$ in $S$ is adjacent to the two endpoints of the original edge $e_j$ in $E^*$.

First suppose that $G$ has a $k$-colouring. We colour $A_1$, $A_2$, and $A_3$ in $G'$ according to the $k$-colouring of $G$. We use $kq$ other colours for the cliques $D$ and $S$. Let us show that this proper $K$-colouring of $G'$ is acyclic. The coloring is already acyclic on $G' \setminus S$ since this induced subgraph of $G$ is chordal. So a potential bichromatic cycle of $G'$ must contain a vertex $s_i^j$ of $S$. However, $s_i^j$ has $kq - 1$ neighbours in the clique $S$ and two neighbours in distinct cliques among $A_1$, $A_2$, $A_3$, that are the endpoints of the original edge $e_j$ in $E^*$. By the properties of our colouring, all the neighbours of $s_i^j$ have distinct colours. This shows that $s_i^j$ is not contained in a bichromatic cycle.

Now suppose that $G'$ has an acyclic $K$-colouring. Let $d_1, \dots, d_{kq}$ be the colours that appear on the vertices of $D$. Then at most $k$ other colours $c_1, \dots, c_k$ are used to properly colour the cliques $A_1$, $A_2$, $A_3$. For contradiction, suppose that there exist two vertices $x$ and $y$ with the same colour, say $c_1$, that are the endpoints of an original edge in $E^*$, say $e_1$. Notice that $x$ and $y$ are both adjacent to all the vertices of $D$ and to the $k$ vertices $s_i^1$. None of the $k$ vertices $s_i^1$ can get colour $c_1$. Moreover, one of these $k$ vertices, say $s_1^1$, must get a colour not in $c_2, \dots, c_k$. We assume without loss of generality that $s_1^1$ is coloured $d_1$. Then $G'$ contains a bichromatic 4-cycle coloured with colours $c_1$ and $d_1$ consisting of the vertices $x$, $s_1^1$, $y$, and the vertex coloured $d_1$ in $D$. This contradiction shows that the colours $c_1, \dots, c_k$ on $A_1$, $A_2$, and $A_3$ give a $k$-colouring of $G$. □

We combine the above results with a result of Lyons [139] to prove Theorem 9.1.

**Theorem 9.1 (restated).** *Let $H$ be a graph. For the class of $H$-free graphs, it holds that:*

*(i)* Acyclic Colouring *is polynomial-time solvable if $H \subseteq_i P_4$ and NP-complete if $H$ is not a linear forest or $H \supseteq_i 6P_1$;*

*(ii) for every $k \geq 3$,* Acyclic $k$-Colouring *is polynomial-time solvable if $H$ is a linear forest and NP-complete otherwise.*

*Proof.* We first prove (ii). First suppose that $H$ contains an induced cycle $C_p$. Then we use Lemma 9.6. Now assume $H$ has no cycle so $H$ is a forest. If $H$ has a vertex of degree at least 3, then $H$ has an induced $K_{1,3}$. As every line graph of a multigraph is $K_{1,3}$-free, we can use Lemma 9.7. Otherwise $H$ is a linear forest and we use Corollary 9.5.

We now prove (i). Due to (ii), we may assume that $H$ is a linear forest. If $H \subseteq_i P_4$, then we use the result of Lyons [139] that states that Acyclic Colouring is polynomial-time solvable for $P_4$-free graphs. On the other hand, if $H \supseteq_i 6P_1$, we use Lemma 9.9. This concludes the proof. □

### 9.2.3 Randomised reduction for co-bipartite graphs

In our next result, $k$ is part of the input. Recall that a graph is *co-bipartite* if it is the complement of a bipartite graph. As bipartite graphs are $C_3$-free, co-bipartite graphs are $3P_1$-free. We use a result by Alon et al. [3]. However, the problem from which we reduce is proven there to be NP-hard not under deterministic but under randomised reduction. (This means that even in the case P is not equal to NP, the problem can be polynomial. The polynomial algorithm for the problem is ruled out if BPP is not equal to P.) To provide NP-hardness proof under deterministic reduction is an open problem. Its resolution would reduce the number of open cases for Acyclic Colouring to just one — $2P_2$.

**Lemma 9.10.** Acyclic Colouring *is NP-hard (under randomised reduction) for co-bipartite graphs.*

*Proof.* Alon et al. [3, Theorem 3.5] proved that deciding if a balanced bipartite graph on $2n$ vertices has a connected matching of size $n$ is NP-hard under randomised reduction. A matching is called *connected* if no two edges of the matching induce $2K_2$ in the given graph. We shall reduce from this problem to prove our theorem.

To this end, we claim that a balanced bipartite graph $G$ with parts $A$ and $B$ such that $|A| = |B| = n$ has a connected matching of size $n$ if and only if its complement has an acyclic colouring with $n$ colours.

Suppose that there is an acyclic colouring $c$ of $\overline{G}$ with $n$ colours. Clearly, such colouring uses $n$ colours on $A$ and $n$ colours on $B$. Vertices coloured with the same colour do not have an edge between them in $\overline{G}$ and thus are connected by an edge in $G$. Let us take the set of edges formed by each of the $n$ colour classes. By the property of colouring, this is a matching in $G$ and it is of size $n$. To see that it is also connected, suppose for a contradiction that there are two edges of the matching, say $a_1b_1$ and $a_2b_2$, forming an induced $2K_2$ in $G$. Without loss of generality, $c(a_1) = c(b_1) = 1$ and $c(a_2) = c(b_2) = 2$. Now the induced $2K_2$ in $G$ corresponds to a 4-cycle in $\overline{G}$ coloured with two colours, a contradiction with $c$ being an acyclic colouring.

In the opposite direction, let us have a connected matching of size $n$ in $G$. Colour the $n$ vertices in $A$ by $1, \ldots, n$. Let us colour the vertices of $B$ with respect to the connected matching so that each vertex of $B$ gets the colour of the vertex in $A$ it is matched to. Indeed, this is a colouring of $\overline{G}$ by $n$ colours. It remains to prove that it is acyclic. Any cycle in $G$ having more than five vertices has by the definition of our colouring at least three colours. Therefore, a possible bichromatic cycle in $\overline{G}$ must be of size 4. The only possibility for such 4-cycle is that it is formed by two pairs of vertices, each one forming an edge of the connected matching in $G$. However, this implies that these two matching edges induce $2K_2$ in $G$, a contradiction with the connectedness of the original matching. This finishes the proof our claim. □

## 9.3 Conclusion and open problems

The complexity study led to an almost complete complexity classifications (Theorem 9.1). We further identified a number of open questions for future research.

In Lemma 9.6 we prove that for every $k \geq 3$ and every $g \geq 3$, Acyclic $k$-Colouring is NP-complete for graphs of girth at least $g$. It would be nice to prove an analogous result for the injective colouring. We recall that Injective 3-Colouring is polynomial-time solvable for general graphs. Moreover, for every

$k \geq 4$, Injective $k$-Colouring is NP-complete for bipartite graphs and thus for graphs of girth at least 4. Hence, we pose the following open problem.

**Problem 9.11.** *For every $g \geq 5$, determine the complexity of* Injective Colouring *and* Injective $k$-Colouring *($k \geq 4$) for graphs of girth at least $g$.*

This problem has eluded us and remains open and is, we believe, challenging. We have made progress for the corresponding high-girth problem for Star 3-Colouring in [31]. However, we leave the high-girth problem for Star $k$-Colouring open for $k \geq 4$, as follows. We believe it represents an interesting technical challenge. At the moment, we only know that for $k \geq 4$, Star $k$-Colouring is NP-complete for bipartite graphs [2] and thus for graphs of girth at least 4.

**Problem 9.12.** *For every $g \geq 5$, determine the complexity of* Star $k$-Colouring *($k \geq 4$) for graphs of girth at least $g$.*

Naturally we also aim to settle the remaining open cases for our three problems in Table 9.1. In particular, the cases left for the problem Acyclic Colouring. Especially troublesome case is the case of $H = 2P_2$ which is open for both acyclic and star colouring.

**Problem 9.13.** *Determine the complexity of* Acyclic Colouring *and* Star Colouring *for $2P_2$-free graphs.*

Recall that Injective Colouring and Colouring are NP-complete for $2P_2$-free graphs. However, none of the hardness constructions for these problems carry over to Acyclic Colouring and Star Colouring. In this context, the next open problem from Lyons [138] for a subclass of $2P_2$-free graphs is also interesting. A graph $G = (V, E)$ is *split* if $V = I \cup K$, where $I$ is an independent set, $K$ is a clique and $I \cap K = \emptyset$. The class of split graphs coincides with the class of $(2P_2, C_4, C_5)$-free graphs [99] and thus Acyclic Colouring is equivalent to Colouring for split graphs, and hence it is polynomial-time solvable.

# Part IV

# Conclusion

# Chapter 10

# General conclusion and related work

In this final chapter, we will not try again to summarise the work done in the thesis but rather show further open problems we have not discussed and draw attention to interesting recent developments regarding the work presented in this thesis. Mainly, we shall focus on two things: (1) papers that cited our work and (2) the follow-up papers co-authored by the author of this thesis which are not included in this thesis.

## Part I

For a better orientation, we shall shortly speak about the genesis of our results and how the timeline went. Initially, paper [23] contained dichotomies on reflexive and irreflexive signed trees. Immediately after that, work began to provide the full dichotomy for signed trees. We succeeded in [24] (Chapter 3). The conclusion section of [23] also suggested that a possible interesting case to classify could be path- and cycle-separable graphs. This later transpired into the results of [22] (Chapter 4).

The papers on trees and separable graphs attracted attention of Kim and Siggers, who attempted to provide a dichotomy for the special case of reflexive signed graphs: weakly balanced reflexive signed graphs. They conjectured a possible dichotomy (Conjecture 4.1) and proved a general result on weakly balanced reflexive graphs saying that if there is a special min ordering of such graphs, it implies the existence of the so-called *semi-conservative weak near-unanimity polymorphism* for the switching graph. The existence of such a polymorphism implies a polynomial-time algorithm by the dichotomy for CSPs.

Only very recently, we were able to prove the following result for weakly balanced irreflexive graphs. (Recall that the result is stated for bipartite graphs only. Non-bipartite irreflexive signed graphs are not relevant since the list homomorphism problem for them is trivially NP-complete by [102]; the fact that we observed in the beginning of Section 4.2.)

**Theorem 10.1.** *[25] A weakly balanced bipartite signed graph $\widehat{H}$ has a special min ordering if and only if it has no chain and no invertible pair. If $\widehat{H}$ has a special min ordering, then the list homomorphism problem for $\widehat{H}$ can be solved in polynomial time. Otherwise, $\widehat{H}$ has a chain or an invertible pair, and the list homomorphism problem for $\widehat{H}$ is NP-complete.*

Furthermore, for the reflexive case, there is a similar result in [25] (with the polynomial part provided by the result above from [117]).

**Theorem 10.2.** *[25, 117] A weakly balanced reflexive signed graph $\widehat{H}$ has a special min ordering if and only if it has no chain and no invertible pair. If $\widehat{H}$ has a special min ordering, then the list homomorphism problem for $\widehat{H}$ can be solved in polynomial time. Otherwise, $\widehat{H}$ has a chain or an invertible pair, and the list homomorphism problem for $\widehat{H}$ is NP-complete.*

Observe that this, in some sense, generalises the results we obtained in Chapter 4. However, the added value of the chapter on separable graphs is that its results provide a concrete and constructive combinatorial characterisation of polynomial and NP-complete cases for path- and cycle-separable signed graphs. We also construct explicit min orderings for the polynomial cases. Also, some of the cycle-separable graphs are not necessarily weakly balanced. Therefore, this work goes beyond the case of weak balancedness.

Of course, the far-reaching goal is still the general dichotomy. For now, the next steps we are taking are (1) dichotomy for mixed weakly balanced signed graphs and (2) more general classes of signed graphs with unicoloured edges forming exactly a spanning subgraph, for example, *tree-separable graphs.*

# Part II

Our recent work continues in the spirit of [124]. In [30], we present a complete characterisation of the computational complexity of covering coloured mixed (multi)graphs with semi-edges for the case that every equivalence class in the degree partition of the target graph has at most two vertices. We prove that a strong polynomial/NP-complete dichotomy holds: for each fixed target graph $H$ of that type, the $H$-Cover problem is either polynomial-time solvable for arbitrary inputs or NP-complete even for simple graphs on input.

Furthermore, Bulteau et al. [45] recently also revived the study of the complexity of graph coverings and provided a thorough study of the problem of finding a locally constrained graph homomorphism (either locally bijective, surjective, or injective) from the point of view of parameterized complexity. To this end, they introduced a unifying ILP (integer linear programming) model and applied it also to Role Assignment. Seeing these ideas applied to (multi)graphs with semi-edges could be interesting.

# Part III

The main open problems left are summarised in Section 9.3.

The problem of complexity of acyclic (and star and injective) colouring for $H$-free

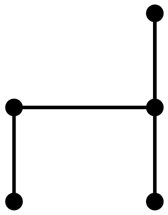


Figure 10.1: The chair graph.

graphs gained already some attention since the publishing of [32] — namely in [15, 35, 37, 159]. Let us further focus on the follow-up progress on acyclic colouring. In [37], Brause et al. show that ACYCLIC-3-COLOURING is polynomial-time solvable for *chair-free graphs* of bounded diameter. (The chair graph is depicted in Figure 10.1.) On the other hand, in [35], Brause et al. show that for graph of diameter at most $d$, ACYCLIC-3-COLOURING is polynomial-time solvable if $d \leq 3$ but NP-complete if $d \geq 8$. Determining the situation of $d$ being at most 4, 5, 6, or 7 is thus another open problem in the area.

# Bibliography

[1] Abello, J., Fellows, M. R., and Stillwell, J. C. On the complexity and combinatorics of covering finite complexes. *Australian Journal of Combinatorics 4* (1991), 103–112.

[2] Albertson, M. O., Chappell, G. G., Kierstead, H. A., Kündgen, A., and Ramamurthi, R. Coloring with no 2-colored $P_4$'s. *Electronic Journal of Combinatorics 11* (2004).

[3] Alon, N., Cohen, J. D., Griffiths, T. L., Manurangsi, P., Reichman, D., Shinkar, I., Wagner, T., and Yu, A. Multitasking capacity: Hardness results and improved constructions. *SIAM Journal on Discrete Mathematics 34*, 1 (2020), 885–903.

[4] Alon, N., McDiarmid, C., and Reed, B. A. Acyclic coloring of graphs. *Random Structures and Algorithms 2* (1991), 277–288.

[5] Alon, N., and Zaks, A. Algorithmic aspects of acyclic edge colorings. *Algorithmica 32* (2002), 611–614.

[6] Angelini, P., and Frati, F. Acyclically 3-colorable planar graphs. *Journal of Combinatorial Optimization 24* (2012), 116–130.

[7] Angluin, D. Local and global properties in networks of processors. *Proceedings of the 12th ACM Symposium on Theory of Computing* (1980), 82–93.

[8] Angluin, D., and Gardiner, A. Finite common coverings of pairs of regular graphs. *Journal of Combinatorial Theory, Series B 30* (1981), 184–187.

[9] Archdeacon, D. Two graphs without planar covers. *Journal of Graph Theory 41*, 4 (2002), 318–326.

[10] Armstrong, M. A. Lifting group actions to covering spaces. *Discrete Groups and Geometry* (1992), 10–15.

[11] Atminas, A., Lozin, V. V., and Razgon, I. Linear time algorithm for computing a small biclique in graphs without long induced paths. *Proceedings of SWAT 2012, LNCS 7357* (2012), 142–152.

[12] Bandelt, H.-J., Farber, M., and Hell, P. Absolute reflexive retracts and absolute bipartite retracts. *Discrete Applied Mathematics 44*, 1-3 (1993), 9–20.

[13] Bang-Jensen, J., Hell, P., and MacGillivray, G. The complexity of colouring by semicomplete digraphs. *SIAM Journal on Discrete Mathematics 1*, 3 (1988), 281–298.

[14] Bard, S., Bellitto, T., Duffy, C., MacGillivray, G., and Yang, F. Complexity of locally-injective homomorphisms to tournaments. *Discrete Mathematics & Theoretical Computer Science 20*, 2 (2018).

[15] Berthe, G., Martin, B., Paulusma, D., and Smith, S. The complexity of l(p, q)-edge-labelling. In *WALCOM: Algorithms and Computation - 16th International Conference and Workshops, WALCOM 2022, Jember, Indonesia, March 24-26, 2022, Proceedings* (2022), P. Mutzel, M. S. Rahman, and Slamin, Eds., vol. 13174 of *Lecture Notes in Computer Science*, Springer, pp. 175–186.

[16] Biggs, N. *Algebraic Graph Theory.* Cambridge University Press, 1974.

[17] Bílka, O., Jirásek, J., Klavík, P., Tancer, M., and Volec, J. On the complexity of planar covering of small graphs. In *Graph-Theoretic Concepts in Computer Science* (2011), P. Kolman and J. Kratochvíl, Eds., vol. 6986 of *Lecture Notes in Computer Science*, Springer, pp. 83–94.

[18] Birman, J. S., and Hilden, H. M. Lifting and projecting homeomorphisms. *Archiv der Mathematik 23*, 1 (1972), 428–434.

[19] Bodlaender, H. L. The classification of coverings of processor networks. *Journal of Parallel Distributed Computing 6* (1989), 166–182.

[20] Bodlaender, H. L. A linear-time algorithm for finding tree-decompositions of small treewidth. *SIAM Journal on Computing 25* (1996), 1305–1317.

[21] Bodlaender, H. L., Kloks, T., Tan, R. B., and van Leeuwen, J. Approximations for lambda-colorings of graphs. *Computer Journal 47* (2004), 193–204.

[22] Bok, J., Brewster, R. C., Feder, T., Hell, P., and Jedličková, N. List homomorphisms to separable signed graphs. In *Algorithms and Discrete Applied Mathematics - 8th International Conference, CALDAM 2022, Puducherry, India, February 10-12, 2022, Proceedings* (2022), N. Balachandran and R. Inkulu, Eds., vol. 13179 of *Lecture Notes in Computer Science*, Springer, pp. 22–35.

[23] Bok, J., Brewster, R. C., Feder, T., Hell, P., and Jedličková, N. List homomorphism problems for signed graphs. In *45th International Symposium on Mathematical Foundations of Computer Science (MFCS 2020)* (Dagstuhl, Germany, 2020), J. Esparza and D. Kráľ,

Eds., vol. 170 of *Leibniz International Proceedings in Informatics (LIPIcs)*, Schloss Dagstuhl–Leibniz-Zentrum für Informatik, pp. 20:1–20:14.

[24] Bok, J., Brewster, R. C., Feder, T., Hell, P., and Jedličková, N. List homomorphism problems for signed graphs, 2021. `https://arxiv.org/abs/2005.05547`.

[25] Bok, J., Brewster, R. C., Hell, P., Jedličková, N., and Rafiey, A. Min orderings and list homomorphism dichotomies for signed and unsigned graphs. *CoRR abs/2206.01068* (2022). `https://arxiv.org/abs/2206.01068`.

[26] Bok, J., Brewster, R. C., Hell, P., and Jedličková, N. List homomorphisms of signed graphs. In *Bordeaux Graph Workshop* (2019), pp. 81–84.

[27] Bok, J., Fiala, J., Hliněný, P., Jedličková, N., and Kratochvíl, J. Computational complexity of covering multigraphs with semi-edges: Small cases. In *46th International Symposium on Mathematical Foundations of Computer Science, MFCS 2021, August 23-27, 2021, Tallinn, Estonia* (2021), F. Bonchi and S. J. Puglisi, Eds., vol. 202 of *LIPIcs*, Schloss Dagstuhl - Leibniz-Zentrum für Informatik, pp. 21:1–21:15.

[28] Bok, J., Fiala, J., Jedličková, N., Kratochvíl, J., and Rzążewski, P. List covering of regular multigraphs. In *Combinatorial Algorithms - 33rd International Workshop, IWOCA 2022, Trier, Germany, June 7-9, 2022, Proceedings* (2022), C. Bazgan and H. Fernau, Eds., vol. 13270 of *Lecture Notes in Computer Science*, Springer, pp. 228–242.

[29] Bok, J., Fiala, J., Jedličková, N., Kratochvíl, J., and Seifrtová, M. Computational complexity of covering disconnected multigraphs. In *Fundamentals of Computation Theory* (2021), E. Bampis and A. Pagourtzis, Eds., Springer, pp. 85–99.

[30] Bok, J., Jedličková, N., Kratochvíl, J., , and Seifertová, M. Computational complexity of covering colored mixed multigraphs with small classes in degree partition. Submitted, 2022.

[31] Bok, J., Jedličková, N., Martin, B., Ochem, P., Paulusma, D., and Smith, S. Acyclic, star and injective colouring: A complexity picture for $H$-free graphs. *CoRR abs/2008.09415* (2021). `https://arxiv.org/abs/2008.09415`.

[32] Bok, J., Jedličková, N., Martin, B., Paulusma, D., and Smith, S. Acyclic, star and injective colouring: A complexity picture for $H$-free graphs. In *28th Annual European Symposium on Algorithms, ESA 2020, September 7-9, 2020, Pisa, Italy (Virtual Conference)* (2020), F. Grandoni,

G. Herman, and P. Sanders, Eds., vol. 173 of *LIPIcs*, Schloss Dagstuhl - Leibniz-Zentrum für Informatik, pp. 22:1–22:22.

[33] Bonamy, M., Dabrowski, K. K., Feghali, C., Johnson, M., and Paulusma, D. Independent feedback vertex set for $P_5$-free graphs. *Algorithmica 81* (2019), 1342–1369.

[34] Borodin, O. V. On acyclic colorings of planar graphs. *Discrete Mathematics 25* (1979), 211–236.

[35] Brause, C., Golovach, P. A., Martin, B., Ochem, P., Paulusma, D., and Smith, S. Acyclic, star, and injective colouring: Bounding the diameter. *Electronic Journal of Combinatorics 29*, 2 (2022).

[36] Brause, C., Golovach, P. A., Martin, B., Paulusma, D., and Smith, S. Acyclic, star, and injective colouring: Bounding the diameter. In *Graph-Theoretic Concepts in Computer Science - 47th International Workshop, WG 2021, Warsaw, Poland, June 23-25, 2021, Revised Selected Papers* (2021), L. Kowalik, M. Pilipczuk, and P. Rzążewski, Eds., vol. 12911 of *Lecture Notes in Computer Science*, Springer, pp. 336–348.

[37] Brause, C., Golovach, P. A., Martin, B., Paulusma, D., and Smith, S. Partitioning h-free graphs of bounded diameter. In *32nd International Symposium on Algorithms and Computation, ISAAC 2021, December 6-8, 2021, Fukuoka, Japan* (2021), H. Ahn and K. Sadakane, Eds., vol. 212 of *LIPIcs*, Schloss Dagstuhl - Leibniz-Zentrum für Informatik, pp. 21:1–21:14.

[38] Brewster, R. C. *Vertex colourings of edge-coloured graphs.* ProQuest LLC, Ann Arbor, MI, 1993. Thesis (Ph.D.)–Simon Fraser University (Canada).

[39] Brewster, R. C., Foucaud, F., Hell, P., and Naserasr, R. The complexity of signed graph and edge-coloured graph homomorphisms. *Discrete Mathematics 340*, 2 (2017), 223–235.

[40] Brewster, R. C., and Graves, T. Edge-switching homomorphisms of edge-coloured graphs. *Discrete Mathematics 309*, 18 (2009), 5540–5546.

[41] Brewster, R. C., and Siggers, M. A complexity dichotomy for signed $H$-colouring. *Discrete Mathematics 341*, 10 (2018), 2768–2773.

[42] Broersma, H., Golovach, P. A., Paulusma, D., and Song, J. Updating the complexity status of coloring graphs without a fixed induced linear forest. *Theoretical Computer Science 414* (2012), 9–19.

[43] Bulatov, A. A. Complexity of conservative constraint satisfaction problems. *ACM Transactions on Computational Logic (TOCL) 12*, 4 (2011), 1–66.

[44] Bulatov, A. A. A dichotomy theorem for nonuniform CSPs. In *58th Annual IEEE Symposium on Foundations of Computer Science—FOCS 2017*. IEEE Computer Soc., Los Alamitos, CA, 2017, pp. 319–330.

[45] Bulteau, L., Dabrowski, K. K., Köhler, N., Ordyniak, S., and Paulusma, D. An algorithmic framework for locally constrained homomorphisms. *CoRR abs/2201.11731* (2022). `https://arxiv.org/abs/2201.11731`.

[46] Calamoneri, T. The $L(h,k)$-labelling problem: An updated survey and annotated bibliography. *Computer Journal 54* (2011), 1344–1371.

[47] Chalopin, J., Métivier, Y., and Zielonka, W. Local computations in graphs: the case of cellular edge local computations. *Fundamenta Informaticae 74*, 1 (2006), 85–114.

[48] Chalopin, J., and Paulusma, D. Graph labelings derived from models in distributed computing: A complete complexity classification. *Networks 58*, 3 (2011), 207–231.

[49] Chalopin, J., and Paulusma, D. Packing bipartite graphs with covers of complete bipartite graphs. *Discrete Applied Mathematics 168* (2014), 40–50.

[50] Chaplick, S., Fiala, J., van 't Hof, P., Paulusma, D., and Tesař, M. Locally constrained homomorphisms on graphs of bounded treewidth and bounded degree. *Theoretical Computer Science 590* (2015), 86–95.

[51] Cheng, C. T., McDermid, E., and Suzuki, I. Planarization and acyclic colorings of subcubic claw-free graphs. *Proc. of WG 2011, LNCS 6986* (2011), 107–118.

[52] Chudnovsky, M., Huang, S., Spirkl, S., and Zhong, M. List 3-coloring graphs with no induced $P_6 + rP_3$. *Algorithmica 83*, 1 (2021), 216–251.

[53] Coleman, T. F., and Cai, J.-Y. The cyclic coloring problem and estimation of sparse Hessian matrices. *SIAM Journal on Algebraic Discrete Methods 7* (1986), 221–235.

[54] Corneil, D. G. *Graph Isomorphism.* PhD thesis, University of Toronto, 1968.

[55] Corneil, D. G., and Gotlieb, C. C. An efficient algorithm for graph isomorphism. *Journal of the Association for Computing Machinery 17* (1970), 51–64.

[56] Courcelle, B. The monadic second-order logic of graphs. I. Recognizable sets of finite graphs. *Information and Computation 85* (1990), 12–75.

[57] Courcelle, B., and Métivier, Y. Coverings and minors: Applications to local computations in graphs. *European Journal of Combinatorics 15* (1994), 127–138.

[58] Creignou, N., Khanna, S., and M., S. Complexity classifications of boolean constraint satisfaction problems. In *SIAM Monographs on Discrete Mathematics and Applications* (2001).

[59] Djoković, D. Ž. Automorphisms of graphs and coverings. *Journal of Combinatorial Theory B 16* (1974), 243–247.

[60] Edmonds, J. Paths, trees, and flowers. *Canadian Journal of Mathematics 17* (1965), 449–467.

[61] Emden-Weinert, T., Hougardy, S., and Kreuter, B. Uniquely colourable graphs and the hardness of colouring graphs of large girth. *Combinatorics, Probability and Computing 7* (1998), 375–386.

[62] Erdős, P. Graph theory and probability. *Canadian Journal of Mathematics 11* (1959), 34–38.

[63] Feder, T., and Hell, P. List homomorphisms to reflexive graphs. *Journal of Combinatorial Theory, Series B 72*, 2 (1998), 236–250.

[64] Feder, T., Hell, P., and Huang, J. List homomorphisms and circular arc graphs. *Combinatorica 19*, 4 (1999), 487–505.

[65] Feder, T., Hell, P., and Huang, J. Bi-arc graphs and the complexity of list homomorphisms. *Journal of Graph Theory 42*, 1 (2003), 61–80.

[66] Feder, T., Hell, P., and Huang, J. The structure of bi-arc trees. *Discrete Mathematics 307* (2007), 393–401.

[67] Feder, T., Hell, P., Huang, J., and Rafiey, A. Interval graphs, adjusted interval digraphs, and reflexive list homomorphisms. *Discrete Applied Mathematics 160*, 6 (2012), 697–707.

[68] Feder, T., Hell, P., Johnsson, P., Krokhin, A., and Nordh, G. Retractions to pseudoforests. *SIAM Journal on Discrete Mathematics 24*, 1 (2010), 101–112.

[69] Feder, T., and Vardi, M. Y. The computational structure of monotone monadic SNP and constraint satisfaction: a study through Datalog and group theory. In *STOC* (1993), pp. 612–622.

[70] Fertin, G., Godard, E., and Raspaud, A. Minimum feedback vertex set and acyclic coloring. *Information Processing Letters 84* (2002), 131–139.

[71] Fertin, G., and Raspaud, A. Acyclic coloring of graphs of maximum degree five: Nine colors are enough. *Information Processing Letters 105* (2008), 65–72.

[72] Fertin, G., Raspaud, A., and Reed, B. A. Star coloring of graphs. *Journal of Graph Theory 47*, 3 (2004), 163–182.

[73] Fiala, J. *Locally injective homomorphisms.* PhD thesis, Charles University, Prague, 2000.

[74] Fiala, J. NP completeness of the edge precoloring extension problem on bipartite graphs. *Journal of Graph Theory 43*, 2 (2003), 156–160.

[75] Fiala, J., Golovach, P. A., and Kratochvíl, J. Parameterized complexity of coloring problems: Treewidth versus vertex cover. *Theoretical Computer Science 412* (2011), 2513–2523.

[76] Fiala, J., Heggernes, P., Kristiansen, P., and Telle, J. A. Generalized $H$-coloring and $H$-covering of trees. *Nordic Journal of Computing 10*, 3 (2003), 206–224.

[77] Fiala, J., Klavík, P., Kratochvíl, J., and Nedela, R. Algorithmic aspects of regular graph covers with applications to planar graphs. In *ICALP (1)* (2014), J. Esparza, P. Fraigniaud, T. Husfeldt, and E. Koutsoupias, Eds., vol. 8572 of *Lecture Notes in Computer Science*, Springer, pp. 489–501.

[78] Fiala, J., Klavík, P., Kratochvíl, J., and Nedela, R. Algorithmic aspects of regular graph covers. *CoRR abs/1609.03013* (2016). `http://arxiv.org/abs/1609.03013`.

[79] Fiala, J., Klavík, P., Kratochvíl, J., and Nedela, R. 3-connected reduction for regular graph covers. *European Journal of Combinatorics 73* (2018), 170–210.

[80] Fiala, J., and Kratochvíl, J. Complexity of partial covers of graphs. In *ISAAC* (2001), P. Eades and T. Takaoka, Eds., vol. 2223 of *Lecture Notes in Computer Science*, Springer, pp. 537–549.

[81] Fiala, J., and Kratochvíl, J. Partial covers of graphs. *Discussiones Mathematicae Graph Theory 22* (2002), 89–99.

[82] Fiala, J., and Kratochvíl, J. Locally injective graph homomorphism: Lists guarantee dichotomy. In *WG* (2006), F. V. Fomin, Ed., vol. 4271 of *Lecture Notes in Computer Science*, Springer, pp. 15–26.

[83] Fiala, J., and Kratochvíl, J. Locally constrained graph homomorphisms — structure, complexity, and applications. *Computer Science Review 2*, 2 (2008), 97–111.

[84] Fiala, J., and Paulusma, D. A complete complexity classification of the role assignment problem. *Theoretical Computer Science 1*, 349 (2005), 67–81.

[85] Foucaud, F., and Naserasr, R. The complexity of homomorphisms of signed graphs and signed constraint satisfaction. In *LATIN 2014: Theoretical Informatics - 11th Latin American Symposium, Montevideo, Uruguay, March 31 - April 4, 2014. Proceedings* (2014), A. Pardo and A. Viola, Eds., vol. 8392 of *Lecture Notes in Computer Science*, Springer, pp. 526–537.

[86] Galby, E., Lima, P. T., Paulusma, D., and Ries, B. Classifying $k$-edge colouring for $H$-free graphs. *Information Processing Letters 146* (2019), 39–43.

[87] Gardiner, A. Antipodal covering graphs. *Journal of Combinatorial Theory, Series B 16* (1974), 255–273.

[88] Garey, M. R., and Johnson, D. S. *Computers and Intractability.* W. H. Freeman and Co., New York, 1979.

[89] Getzler, E., and Kapranov, M. M. Modular operads. *Compositio Mathematica 110*, 1 (1998), 65–125.

[90] Golovach, P. A., Johnson, M., Paulusma, D., and Song, J. A survey on the computational complexity of colouring graphs with forbidden subgraphs. *Journal of Graph Theory 84* (2017), 331–363.

[91] Golovach, P. A., Paulusma, D., and Song, J. Coloring graphs without short cycles and long induced paths. *Discrete Applied Mathematics 167* (2014), 107–120.

[92] Griggs, J. R., and Yeh, R. K. Labelling graphs with a condition at distance 2. *SIAM Journal on Discrete Mathematics 5*, 4 (1992), 586–595.

[93] Gross, J. L. Voltage graphs. *Discrete Mathematics 9*, 3 (1974), 239–246.

[94] Gross, J. L., and Alpert, S. R. The topological theory of current graphs. *Journal of Combinatorial Theory, Series B 17*, 3 (1974), 218–233.

[95] Gross, J. L., and Tucker, T. W. Generating all graph coverings by permutation voltage assignments. *Discrete Mathematics 18* (1977), 273–283.

[96] Guenin, B. A characterization of weakly bipartite graphs. *Journal of Combinatorial Theory, Series B 83*, 1 (2001), 112–168.

[97] Guenin, B. Packing odd circuit covers: A conjecture. Manuscript, 2005.

[98] Hahn, G., Kratochvíl, J., Širáň, J., and Sotteau, D. On the injective chromatic number of graphs. *Discrete Mathematics 256* (2002), 179–192.

[99] Hammer, P. L., and Földes, S. Split graphs. *Congressus Numerantium 19* (1977), 311–315.

[100] Harary, F. On the notion of balance of a signed graph. *Michigan Mathematical Journal 2* (1953/54), 143–146 (1955).

[101] Harary, F., and Kabell, J. A. A simple algorithm to detect balance in signed graphs. *Mathematical Social Sciences 1*, 1 (1980/81), 131–136.

[102] Hell, P., and Nešetřil, J. On the complexity of $H$-coloring. *Journal of Combinatorial Theory. Series B 48*, 1 (1990), 92–110.

[103] Hell, P., and Nešetřil, J. *Graphs and homomorphisms*, vol. 28 of *Oxford Lecture Series in Mathematics and its Applications.* Oxford University Press, Oxford, 2004.

[104] Hell, P., and Nešetřil, J. In praise of homomorphisms. *Computer Science Review 40* (2021), 100352.

[105] Hell, P., and Rafiey, A. The dichotomy of list homomorphisms for digraphs. In *Proceedings of the Twenty-Second Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2011, San Francisco, California, USA, January 23-25, 2011* (2011), D. Randall, Ed., SIAM, pp. 1703–1713.

[106] Hell, P., Raspaud, A., and Stacho, J. On injective colourings of chordal graphs. In *LATIN 2008: Theoretical Informatics, 8th Latin American Symposium, Búzios, Brazil, April 7-11, 2008, Proceedings* (2008), E. S. Laber, C. F. Bornstein, L. T. Nogueira, and L. Faria, Eds., vol. 4957 of *Lecture Notes in Computer Science*, Springer, pp. 520–530.

[107] Hliněný, P. $K_{4,4} - e$ has no finite planar cover. *Journal of Graph Theory 21*, 1 (1998), 51–60.

[108] Hliněný, P., and Thomas, R. On possible counterexamples to Negami's planar cover conjecture. *Journal of Graph Theory 46*, 3 (2004), 183–206.

[109] Holyer, I. The NP-completeness of edge-coloring. *SIAM Journal on Computing 10* (1981), 718–720.

[110] Huang, S., Johnson, M., and Paulusma, D. Narrowing the complexity gap for colouring $(C_s, P_t)$-free graphs. *Computer Journal 58* (2015), 3074–3088.

[111] Jansen, K. Complexity results for the optimum cost chromatic partition problem. *Universität Trier, Mathematik/Infomatik, Forschungsbericht* (1996).

[112] Jeavons, P. On the algebraic structure of combinatorial problems. *Theoretical Computer Science 200*, 1-2 (1998), 185–204.

[113] Jin, J., Xu, B., and Zhang, X. On the complexity of injective colorings and its generalizations. *Theoretical Computer Science 491* (2013), 119–126.

[114] Kaiser, T., Lukoťka, R., and Rollová, E. Nowhere-zero flows in signed graphs: a survey. In *Selected topics in graph theory and its applications*, vol. 14 of *Lect. Notes Semin. Interdiscip. Mat.* Semin. Interdiscip. Mat. (S.I.M.), Potenza, 2017, pp. 85–104.

[115] Kang, R. J., and Müller, T. Frugal, acyclic and star colourings of graphs. *Discrete Applied Mathematics 159* (2011), 1806–1814.

[116] Karthick, T. Star coloring of certain graph classes. *Graphs and Combinatorics 34* (2018), 109–128.

[117] Kim, H., and Siggers, M. Towards a dichotomy for the switch list homomorphism problem for signed graphs. *CoRR abs/2104.07764* (2021). https://arxiv.org/abs/2104.07764.

[118] Klimošová, T., Malík, J., Masařík, T., Novotná, J., Paulusma, D., and Slívová, V. Colouring ($\mathrm{P_r} + \mathrm{P_s}$)-free graphs. *Algorithmica 82*, 7 (2020), 1833–1858.

[119] Kostochka, A. V. *Upper bounds of chromatic functions of graphs.* PhD thesis, University of Novosibirsk, 1978.

[120] Král, D., Kratochvíl, J., Tuza, Z., and Woeginger, G. J. Complexity of coloring graphs without forbidden induced subgraphs. In *Graph-Theoretic Concepts in Computer Science, 27th International Workshop, WG 2001, Boltenhagen, Germany, June 14-16, 2001, Proceedings* (2001), A. Brandstädt and V. B. Le, Eds., vol. 2204 of *Lecture Notes in Computer Science*, Springer, pp. 254–262.

[121] Kratochvíl, J. Complexity of hypergraph coloring and Seidel's switching. In *Graph-Theoretic Concepts in Computer Science, 29th International Workshop, WG 2003, Elspeet, The Netherlands, June 19-21, 2003, Revised Papers* (2003), H. L. Bodlaender, Ed., vol. 2880 of *Lecture Notes in Computer Science*, Springer, pp. 297–308.

[122] Kratochvíl, J., Proskurowski, A., and Telle, J. A. Complexity of graph covering problems. In *WG* (1994), E. W. Mayr, G. Schmidt, and G. Tinhofer, Eds., vol. 903 of *Lecture Notes in Computer Science*, Springer, pp. 93–105.

[123] Kratochvíl, J., Proskurowski, A., and Telle, J. A. Covering directed multigraphs I. Colored directed multigraphs. In *WG* (1997), R. H. Möhring, Ed., vol. 1335 of *Lecture Notes in Computer Science*, Springer, pp. 242–257.

[124] Kratochvíl, J., Proskurowski, A., and Telle, J. A. Covering directed multigraphs II. When 2-SAT helps. Tech. Rep. 1997–354, KAM-DIMATIA Preprint Series, 1997.

[125] Kratochvíl, J., Proskurowski, A., and Telle, J. A. Covering regular graphs. *Journal of Combinatorial Theory, Series B 71*, 1 (1997), 1–16.

[126] Kratochvíl, J., Proskurowski, A., and Telle, J. A. Complexity of graph covering problems. *Nordic Journal of Computing 5* (1998), 173–195.

[127] Kratochvíl, J., Telle, J. A., and Tesař, M. Computational complexity of covering three-vertex multigraphs. *Theoretical Computer Science 609* (2016), 104–117.

[128] Kristiansen, P., and Telle, J. A. Generalized $H$-coloring of graphs. In *ISAAC* (2000), D. T. Lee and S.-H. Teng, Eds., vol. 1969 of *Lecture Notes in Computer Science*, Springer, pp. 456–466.

[129] Kwak, J. H., and Nedela, R. Graphs and their coverings. *Lecture Notes Series 17* (2007). https://www.savbb.sk/~nedela/graphcov.pdf.

[130] Lei, H., Shi, Y., and Song, Z.-X. Star chromatic index of subcubic multigraphs. *Journal of Graph Theory 88* (2018), 566–576.

[131] Leighton, F. T. Finite common coverings of graphs. *Journal of Combinatorial Theory, Series B 33* (1982), 231–238.

[132] Leven, D., and Galil, Z. NP-completeness of finding the chromatic index of regular graphs. *Journal of Algorithms 4* (1983), 35–44.

[133] Linhares-Sales, C., Maia, A. K., Martins, N. A., and Sampaio, R. M. Restricted coloring problems on graphs with few $P_4$'s. *Annals of Operations Research 217* (2014), 385–397.

[134] Litovsky, I., Métivier, Y., and Zielonka, W. The power and the limitations of local computations on graphs. In *WG* (1992), E. W. Mayr, Ed., vol. 657 of *Lecture Notes in Computer Science*, Springer, pp. 333–345.

[135] Lloyd, E. L., and Ramanathan, S. On the complexity of distance-2 coloring. *Proc. ICCI 1992* (1992), 71–74.

[136] Lovász, L., and Plummer, M. D. *Matching Theory.* Akadémiai Kiadó, Budapest, 1986.

[137] Lozin, V. V., and Kamiński, M. Coloring edges and vertices of graphs without short or long cycles. *Contributions to Discrete Mathematics 2*, 1 (2007).

[138] Lyons, A. Restricted coloring problems and forbidden induced subgraphs. *Preprint ANL/MCS-P1611-0409, Mathematics and Computer Science Division, Argonne National Laboratory* (2009). Manuscript.

[139] Lyons, A. Acyclic and star colorings of cographs. *Discrete Applied Mathematics 159* (2011), 1842–1850.

[140] Macbeath, A. M. On a theorem of Hurwitz. *Glasgow Mathematical Journal 5*, 2 (1961), 90–96.

[141] MacGillivray, G., and Swarts, J. The complexity of locally injective homomorphisms. *Discrete Mathematics 310*, 20 (2010), 2685–2696. Graph Theory — Dedicated to Carsten Thomassen on his 60th Birthday.

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

[143] Malnič, A., Marušič, D., and Potočnik, P. Elementary abelian covers of graphs. *Journal of Algebraic Combinatorics 20*, 1 (2004), 71–97.

[144] Malnič, A., Nedela, R., and Škoviera, M. Lifting graph automorphisms by voltage assignments. *European Journal of Combinatorics 21*, 7 (2000), 927–947.

[145] Mednykh, A. D., and Nedela, R. *Harmonic Morphisms of Graphs: Part I: Graph Coverings*, 1st ed. Vydavatelstvo Univerzity Mateja Bela v Banskej Bystrici, 2015.

[146] Mondal, D., Nishat, R. I., Rahman, M. S., and Whitesides, S. Acyclic coloring with few division vertices. *Journal of Discrete Algorithms 23* (2013), 42–53.

[147] Naserasr, R., Rollová, E., and Sopena, É. Homomorphisms of signed graphs. *Journal of Graph Theory 79*, 3 (2015), 178–212.

[148] Naserasr, R., Sopena, É., and Zaslavsky, T. Homomorphisms of signed graphs: An update. *European Journal of Combinatorics 91* (2021), 103222.

[149] Nedela, R., and Škoviera, M. Regular embeddings of canonical double coverings of graphs. *Journal of Combinatorial Theory, Series B 67*, 2 (1996), 249–277.

[150] Negami, S. Graphs which have no planar covering. *Bulletin of the Institute of Mathematics, Academia Sinica 16*, 4 (1988), 377–384.

[151] Ochem, P. *Graph Coloring and Combinatorics on Words.* PhD thesis, Université de Bordeaux, 2005.

[152] Okrasa, K., and Rzążewski, P. Subexponential algorithms for variants of the homomorphism problem in string graphs. *Journal of Computer and System Sciences 109* (2020), 126–144.

[153] Petersen, J. Die Theorie der regulären graphs. *Acta Mathematica 15* (1891), 193–220.

[154] Ringel, G. *Map color theorem*, vol. 209. Springer, Berlin, 1974.

[155] Schaefer, T. J. The complexity of satisfiability problems. In *Proceedings of the tenth annual ACM symposium on Theory of computing* (1978), pp. 216–226.

[156] Schrijver, A. A short proof of guenin's characterization of weakly bipartite graphs. *Journal of Combinatorial Theory, Series B 85*, 2 (2002), 255–260.

[157] Sen, A., and Huson, M. L. A new model for scheduling packet radio networks. *Wireless Networks 3* (1997), 71–82.

[158] Shalu, M. A., and Antony, C. Complexity of restricted variant of star colouring. In *Algorithms and Discrete Applied Mathematics - 6th International Conference, CALDAM 2020, Hyderabad, India, February 13-15, 2020, Proceedings* (2020), M. Changat and S. Das, Eds., vol. 12016 of *Lecture Notes in Computer Science*, Springer, pp. 3–14.

[159] Shalu, M. A., and Antony, C. Star colouring of bounded degree graphs and regular graphs. *Discrete Mathematics 345*, 6 (2022), 112850.

[160] Shepherd, S., Gardam, G., and Woodhouse, D. J. Two generalisations of Leighton's theorem. *CoRR abs/1908.00830* (2019). `https://arxiv.org/abs/1908.00830`.

[161] White, D. R., and Reitz, K. P. Graph and semigroup homomorphisms on networks of relations. *Social Networks 5*, 2 (1983), 193–234.

[162] Wood, D. Defective and clustered graph colouring. *The Electronic Journal of Combinatorics 1000* (2018), 23–13.

[163] Wood, D. R. Acyclic, star and oriented colourings of graph subdivisions. *Discrete Mathematics and Theoretical Computer Science 7* (2005), 37–50.

[164] Woodhouse, D. J. Revisiting Leighton's theorem with the Haar measure. *Mathematical Proceedings of the Cambridge Philosophical Society 170*, 3 (2021), 615–623.

[165] Zaslavsky, T. Characterizations of signed graphs. *Journal of Graph Theory 5*, 4 (1981), 401–406.

[166] Zaslavsky, T. Signed graph coloring. *Discrete Mathematics 39*, 2 (1982), 215–228.

[167] Zaslavsky, T. Signed graphs. *Discrete Applied Mathematics 4*, 1 (1982), 47–74.

[168] Zaslavsky, T. Is there a matroid theory of signed graph embedding? *Ars Combinatoria 45* (1997), 129–141.

[169] Zaslavsky, T. A mathematical bibliography of signed and gain graphs and allied areas. *Electronic Journal of Combinatorics 5* (1998), Dynamic Surveys 8, 124. Manuscript prepared with Marge Pratt.

[170] Zhang, X., and Bylka, S. Disjoint triangles of a cubic line graph. *Graphs and Combinatorics 20* (2004), 275–280.

[171] Zhou, X., Kanari, Y., and Nishizeki, T. Generalized vertex-coloring of partial $k$-trees. *IEICE Transactions on Fundamentals of Electronics, Communication and Computer Sciences E83-A* (2000), 671–678.

[172] Zhu, E., Li, Z., Shao, Z., and Xu, J. Acyclically 4-colorable triangulations. *Information Processing Letters 116* (2016), 401–408.

[173] Zhuk, D. A proof of CSP dichotomy conjecture. In *58th Annual IEEE Symposium on Foundations of Computer Science—FOCS 2017*. IEEE Computer Soc., Los Alamitos, CA, 2017, pp. 331–342.

[174] Zieschang, H. Lifting and projecting homeomorphisms. *Archiv der Mathematik 24*, 1 (1973), 416–421.

# List of figures

# List of publications

1. Jan Bok, Richard C. Brewster, Nikola Jedličková, Pavol Hell, and Arash Rafiey: *Min orderings and list homomorphism dichotomies for signed and unsigned graphs.* Accepted to LATIN 2022, 2022. `https://arxiv.org/abs/2206.01068`

2. Jan Bok, Nikola Jedličková, Jan Kratochvíl, and Michaela Seifrtová: *Computational Complexity of Covering Colored Mixed Multigraphs with Small Classes in Degree Partition.* Submitted, 2022.

3. Jan Bok, Jiří Fiala, Nikola Jedličková, Jan Kratochvíl, and Paweł Rzążewski: *List covering of regular multigraphs.* In Combinatorial Algorithms - 33rd International Workshop, IWOCA 2022, volume 13270 of Lecture Notes in Computer Science, pages 228–242, 2022. `https://doi.org/10.1007/978-3-031-06678-8_17`

4. Jan Bok, Richard C. Brewster, Tomás Feder, Pavol Hell, and Nikola Jedličková: *List homomorphisms to separable signed graphs.* In Algorithms and Discrete Applied Mathematics - 8th International Conference, CALDAM 2022, volume 13179 of Lecture Notes in Computer Science, pages 22–35, 2022. `https://doi.org/10.1007/978-3-030-95018-7_3`

5. Jan Bok, and Martin Černý: *1-convex extensions of incomplete cooperative games and the average value.* Submitted, 2022. `https://arxiv.org/abs/2107.04679`

6. Jan Bok, Martin Černý, David Hartman, and Milan Hladík: *Positivity and convexity in incomplete cooperative games.* Submitted, 2022. `https://arxiv.org/abs/2010.08578`

7. Jan Bok, Jiří Fiala, Nikola Jedličková, Jan Kratochvíl, and Michaela Seifrtová: *Computational Complexity of Covering Disconnected Multigraphs.* In Fundamentals of Computation Theory, FCT 2021, volume 12867 of Lecture Notes in Computer Science, pages 85–89, 2021.

8. Jan Bok, Jiří Fiala, Petr Hliněný, Nikola Jedličková, and Jan Kratochvíl: *Computational Complexity of Covering Multigraphs with Semi-Edges: Small Cases.* In 46th International Symposium on Mathematical Foundations of Computer Science, MFCS 2021, volume 202 of Leibniz International Proceedings in Informatics (LIPIcs), pages 21:1–21:15, 2021.

9. Jan Bok: *Cooperative Interval Games and Selections Revisited.* In Proceedings of the 16th International Symposium on Operational Research in Slovenia, SOR'21, pages 663–669, 2021.

10. Jan Bok, Richard C. Brewster, Tomás Feder, Pavol Hell, and Nikola Jedličková: *List Homomorphism Problems for Signed Graphs.* Submitted, 2021. `https://arxiv.org/abs/2005.05547`

11. Jan Bok, and Nikola Jedličková: *Edge-sum distinguishing labeling.* Commentationes Mathematicae Universitatis Carolinae 62(2):135–149, 2021.

12. Jan Bok, Nikola Jedličková, Barnaby Martin, Pascal Ochem, Daniël Paulusma, and Siani Smith: *Acyclic, Star and Injective Colouring: A Complexity Picture for H-Free Graphs.* Submitted, 2021. `https://arxiv.org/abs/2008.09415`

13. Jan Bok, Nikola Jedličková, Barnaby Martin, Daniël Paulusma, and Siani Smith: *Injective Colouring for H-Free Graphs.* In Computer Science – Theory and Applications, CSR 2021, volume 12730 of Lecture Notes in Computer Science, pages 18–30, 2021. `https://doi.org/10.1007/978-3-030-79416-3_2`

14. Jan Bok, Nikola Jedličková, and Jana Maxová: *A Relaxed Version of Šoltés's Problem and Cactus Graphs.* Bulletin of the Malaysian Mathematical Sciences Society, 44:3733—3745, 2021. `https://doi.org/10.1007/s40840-021-01144-5`

15. Jan Bok, Richard C. Brewster, Tomás Feder, Nikola Jedličková, and Pavol Hell: *List Homomorphism Problems for Signed Graphs.* In 45th International Symposium on Mathematical Foundations of Computer Science, MFCS 2020, volume 170 of Leibniz International Proceedings in Informatics (LIPIcs), pages 170:20:1–20:14, 2020.

16. Jan Bok, Nikola Jedličková, Barnaby Martin, Daniël Paulusma, and Siani Smith: *Acyclic, Star and Injective Colouring: A Complexity Picture for H-Free Graphs.* In 28th Annual European Symposium on Algorithms, ESA 2020, volume 173 of Leibniz International Proceedings in Informatics (LIPIcs), pages 173:22:1–22:22, 2020.

17. Jan Bok, Nikola Jedličková, and Jana Maxová: *On relaxed Šoltés's problem.* Acta Mathematica Universitatis Comenianae 88(3):475–480, 2019.

18. Jan Bok, and Jana Maxová: *Characterizing subclasses of cover-incomparability graphs by forbidden subposets.* Order 36(2):349–358, 2019.

19. Jan Bok, Boris Furtula, Nikola Jedličková, and Riste Škrekovski: *On Extremal Graphs of Weighted Szeged Index.* MATCH 82(1):93–109, 2019.

20. Jan Bok, and Jaroslav Nešetřil: *Graph-indexed random walks on pseudotrees.* Electronic Notes in Discrete Mathematics 68:263–268, 2018.

21. Jan Bok, and Milan Hladík: *Selection-Based Approach to Cooperative Interval Games.* In Operations Research and Enterprise Systems, ICORES

2015, volume 577 of Communications in Computer and Information Science, pages 40-53, 2015.