WARSAW UNIVERSITY OF TECHNOLOGY

DISCIPLINE OF MATHEMATICS
FIELD OF NATURAL SCIENCES

Ph.D. Thesis

Karolina Okrasa, M.Sc.

Graph Homomorphisms: From Structure to Algorithms

Supervisor/Supervisors
prof. Zbigniew Lonc

Additional supervisor
Paweł Rzążewski, Ph. D.

WARSAW 2023

Acknowledgements

I sincerely thank Paweł, the greatest advisor one can have, for all his time, effort and enthusiasm; I would not even think about starting this journey without being sure he is always there to tame not only all my wildest ideas, but also that dramatic side of me that nobody else is aware of.

I would also like to express my deep gratitude to Marcin and Michał, who taught me a lot, and certainly shaped me (each one in his own way) into the researcher I am today.

To Zbigniew, my supervisor, for his trust, patience, and calmness.

To Łukasz and Jana for being the nicest company one could have during this extraordinary adventure, and to all my other colleagues from MiNI and MIMUW for making it inspiring and entertaining.

Finally, to my parents and my sister, for their support and understanding, and to Filip, whose love and care for me and the $S_{t,t,t}$-free graphs kept me motivated to complete this journey.

Abstract

A homomorphism from a graph G to a graph H is an edge-preserving mapping from the vertex set of G to the vertex set of H. Graph homomorphisms are a wide and natural generalization of graph colorings; each proper k-coloring of a graph G is a homomorphism from G to the complete graph K_k and vice versa. For a fixed graph H, the graph homomorphism problem $\textsc{Hom}(H)$ takes a graph G as an instance and asks whether there exists a homomorphism from G to H.

The leading objective of this dissertation is to study how the complexity bounds of the graph homomorphism problem (and its list generalization) depend on the structural properties of the instances. In general, the existence of algorithms solving NP-complete cases of $\textsc{Hom}(H)$ significantly faster than brute force is unlikely under standard complexity assumptions. However, it is still possible to find algorithms working in time polynomial, or at least subexponential, in the size of the input, if we put additional assumptions on the class of instances. Many such phenomena were observed in the case of graph colorings. In this work we examine to which extent they can be generalized to graph homomoprhisms.

We are interested in two kinds of restrictions. First, we investigate the classes of instances that can be obtained by bounding a structural parameter of a graph, i.e., treewidth or clique-width. The homomorphism problems can be solved in polynomial time in classes of bounded treewidth/clique-width, and here we aim to understand the optimal dependence on the parameter. Second, we consider classes of graphs that can be obtained by excluding a fixed graph as an induced subgraph, and show for which such classes the algorithms solving $\textsc{Hom}(H)$ (and its list variant) significantly faster than the standard brute-force approach exists. We complement these results with lower bounds, based on standard complexity theory hypotheses.

We emphasize that while our primary interest comes from the perspective of computational complexity, the presented algorithmic results are usually a consequence of combinatorial properties of graphs and homomorphisms, that require a deep understanding of their structure.

Keywords: graph homomorphisms, list homomorphisms, hereditary graph classes, clique-width, treewidth, Exponential Time Hypothesis, Strong Exponential Time Hypothesis

Streszczenie

Homomorfizmem z grafu G w graf H nazywamy funkcję ze zbioru wierzchołków G w zbiór wierzchołków H zachowującą krawędzie. Homomorfizmy w naturalny sposób uogólniają kolorowania grafów; poprawne k-kolorowanie grafu G możemy interpretować jako homomorfizm z G w graf pełny K_k i na odwrót. Dla ustalonego grafu H problem homomorfizmu grafów $\textsc{Hom}(H)$ przyjmuje jako instancję graf G i odpowiada na pytanie, czy istnieje homomorfizm z G w H.

W niniejszej rozprawie badamy, jak złożoność problemu homomorfizmu grafów (i jego listowej wersji) zależy od struktury instancji. W ogólności, przy standardowych założeniach teorii złożoności, nie istnieją algorytmy rozwiązujące NP-zupełne przypadki problemu $\textsc{Hom}(H)$ istotnie szybciej niż metodą brute-force. Nie wyklucza to jednak możliwości rozwiązania problemu w czasie wielomianowym lub podwykładniczym względem rozmiaru instancji, jeśli nałożymy pewne ograniczenia na klasę tych instancji. Istnienie tego typu algorytmów dla problemu kolorowania zostało przebadane z użyciem wielu metod; w poniższej pracy analizujemy, które z nich można uogólnić na problem homomorfizmu.

Interesują nas dwa rodzaje ograniczeń. W pierwszej kolejności przyjrzymy się klasom instancji, które można otrzymać poprzez ograniczenie pewnego strukturalnego parametru grafu – w naszej pracy będzie to szerokość drzewowa oraz szerokość klikowa. Wiadomo, że jeśli ograniczymy szerokość drzewową lub klikową instancji, wówczas problem homomorfizmu można rozwiązać w czasie wielomianowym. W niniejszej pracy analizujemy zaś dokładnie, jak wygląda zależność funkcji złożoności od parametru. W drugiej części pracy rozważamy klasy instancji, które można zdefiniować poprzez zabranianie ustalonego grafu jako podgrafu indukowanego. Pokazujemy, dla jakich klas, które można otrzymać w ten sposób, istnieją algorytmy rozwiązujące $\textsc{Hom}(H)$ (oraz wariant listowy) istotnie szybciej niż metoda brute-force. Uzupełniamy te wyniki, pokazując dolne ograniczenia przy założeniu standardowych hipotez z teorii złożoności.

Chociaż główna motywacja niniejszej rozprawy wywodzi się z teorii złożoności, warto podkreślić, że przedstawione twierdzenia są wynikiem analizy kombinatorycznych własności grafów oraz ich homomorfizmów, oraz głębokiego zrozumienia ich struktury.

Słowa kluczowe: homomorfizmy grafów, listowe homomorfizmy, dziedziczne klasy grafów, szerokość klikowa, szerokość drzewowa, Hipoteza o czasie wykładniczym, Silna hipoteza o czasie wykładniczym

Contents

Chapter 1

Introduction

Out of the many notions in mathematics that attempt to formalize the intuitive meaning of *similarity* between two objects of the same type, one of the most fundamental and general is the concept of a homomorphism [3, 5, 22, 33, 66, 72]. Homomorphisms allow us to capture how the structure of one object is preserved in the other, though, unlike isomorphisms, they do not require the objects to be essentially "the same". The notion of a homomorphism can be considered on all algebraic structures—groups, rings, lattices or vector spaces (where they are rather known as linear mappings)—as well as other objects, like relational structures.

This dissertation focuses on the study of graph homomorphisms, that can be seen as particular cases of homomorphisms of relational structures, namely, the structures that are finite and equipped with one binary symmetric relation. Formally, for two graphs $G = (V(G), E(G))$ and $H = (V(H), E(H))$, a homomorphism from G to H is a function $f : V(G) \to V(H)$ such that for every edge uv of G it holds that $f(u)f(v)$ is an edge of H (see Figure 1.1 for an example). If f is a homomorphism from G to H, we denote this fact by $f : G \to H$. We call H the *target* graph. For a fixed target H, the homomorphism problem, denoted by $\textsc{Hom}(H)$, asks whether there exists a homomorphism from a given graph G to H.

1.1 A generalization of graph colorings

Many classic graph problems can be described as variants of the graph homomorphism problem [11, 41, 45, 58, 66]. One of the most prominent special cases of this phenomenon

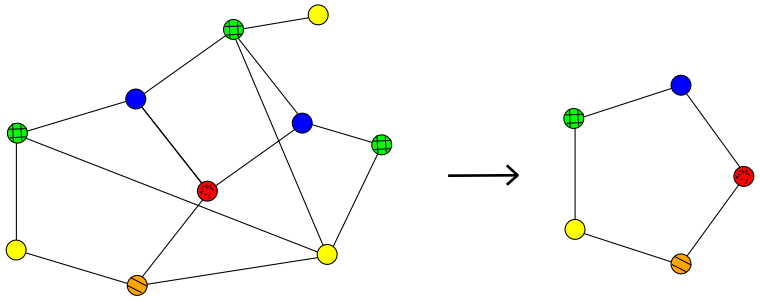


Figure 1.1: An example of a homomorphism from a graph G (left) to a graph H (right) that is a cycle on 5 vertices. Here, vertices of H are depicted using different colors, and the colors of vertices of G illustrate the mapping.

is graph coloring: for a positive integer k, a *k-coloring* of a graph G is a mapping $c : V(G) \to \{1, \ldots, k\}$ such that for every edge uv of G it holds that $c(u) \neq c(v)$. The computational problem of deciding whether G can be colored with k colors is denoted by k-Coloring, and is polynomial-time solvable when $k \leqslant 2$ and NP-complete otherwise [80]. As mentioned, k-Coloring can be seen as a special case of the Hom(H) problem: namely, each k-coloring of a graph G is a homomorphism from G to the complete graph K_k and vice versa.

The symmetric and elegant structure of the objects of interest, the ease of stating questions and conjectures, sometimes surprising results—there are various reasons of the popularity of the graph coloring problem that make it, undoubtfully, one of the best studied graph problems, from the perspective of structural graph theory [1, 19, 59, 60, 78, 111] (that includes the famous Four Color Theorem or Strong Perfect Graph Theorem) or, what is important from our point of view, computational complexity. As a consequence of this interest, the complexity of the k-Coloring problem is deeply understood besides the classic P vs NP-complete dichotomy. The standard brute force approach has the complexity $k^n \cdot n^{\mathcal{O}(1)}$ (here n always stands for the number of vertices of the instance graph), but with more involved techniques it is possible to solve the problem in time $2^n \cdot n^{\mathcal{O}(1)}$ [6]. Moreover, there are numerous algorithms solving the problem effectively on restricted classes of the input instances [30, 53, 57, 94].

The major motivation for the results presented in this dissertation originate from the fact that it seems natural to follow the lines of research for the k-Coloring problem to study the more general Hom(H) problem, and examine to which extent the variety of tools developed to work on coloring problems can be applied. While our primary interest comes from the perspective of the computational problem Hom(H), the algorithmic theorems are

usually a consequence of various combinatorial properties of graphs and homomorphisms between them, and require a deep understanding of their structure. Usually, the crucial obstacle for a straightforward generalization of the aforementioned tools developed for k-Coloring is that the structure of homomorphisms lacks certain symmetries of graph colorings. Intuitively speaking, we need to capture the arbitrary structure of a target graph H, while in the case of colorings the target is always a clique. This is the reason why some of the combinatorial techniques become impractical in the more general setting of homomorphisms, especially when designing lower bounds for the running times of algorithms. This also explains why a plenty of tools used for this kind of proofs comes from algebraic graph theory [13,68,76]. Indeed, some useful properties of homomorphisms, difficult to express by purely combinatorial statements, may be captured using the more universal language of algebra and then exploited in the algorithmic context.

The algebraic approach has proven to be successful in the study on another well-known problem, related to graph homomorphisms, namely, the *Constraint Satisfaction Problem (CSP)* [14, 42, 76, 77]. The CSP problem asks about the existence of a homomorphism between two given relational structures, and is general enough to include many well-known combinatorial problems, such as propositional satisfiability, scheduling problems, systems of linear equations or, of course, graph homomorphisms [4, 91, 108]. From our point of view, it seems also interesting to understand how the methods used to work on graph problems relate to the more generic tools developed for other relational structures.

While the question about the existence of a homomorphism between two graphs seems very natural, it is not the only graph homomorphism-related problem one can consider. For a fixed graph H, a generalization of the Hom(H) problem, and another special case of CSP (known as a *conservative* CSP [12]) that also attracts our attention, is the so-called *list homomorphism* problem LHom(H). An instance of LHom(H) consists of a graph G and a *list* function $L : V(G) \to 2^{V(H)}$. We ask whether there exists a homomorphism $h : G \to H$ such that for every vertex v of G it holds that $h(v) \in L(v)$.

Clearly, list homomorphisms also capture list colorings: the List-k-Coloring is a generalization of the k-Coloring, where every vertex of an instance G is additionally equipped with a subset of $\{1, \ldots, k\}$, interpreted as the list of available colors, and we are only interested in existence of solutions that assign to every vertex a color from its list. While the concept of list colorings itself draws considerable attention amongst

researchers (it was already studied by Vizing [113], and Erdős, Rubin and Taylor [34] in 1970s), we point out that having lists appears to be extremely useful from the point of view of designing algorithms. Indeed, lists allow us to express that in a potential solution some vertex is *not* mapped to a certain value which, in turn, nicely combines with branching techniques. Therefore, numerous algorithms for graph coloring can be obtained as corollaries of algorithms solving the more general list version of the problem (some examples are included below).

1.2 Computing homomorphisms: state of the art

The P vs. NP-complete dichotomy for $\textsc{Hom}(H)$ was proven in 1990 by Hell and Nešetřil [67]. They have shown that the problem is polynomial-time solvable if H contains a vertex with a loop or is bipartite, and is NP-complete for all other graphs H. The complexity dichotomy for the $\textsc{LHom}(H)$ problem was proven in three steps [37–39]. First, Feder and Hell [37] considered the special case of *reflexive* target graphs, i.e., graphs such that each vertex has a loop. Then, together with Huang [38] they have shown the complexity dichotomy for irreflexive graphs (i.e., with no loops). Finally, in the paper [39], also by Feder, Hell and Huang, the results obtained in [37, 38] were generalized: the authors defined the class of *bi-arc-graphs*, and showed that if H is a bi-arc graph, then $\textsc{LHom}(H)$ can be solved in polynomial time, and otherwise the problem is NP-complete.

In this thesis, our objective is to understand the complexity of the $\textsc{Hom}(H)$ and $\textsc{LHom}(H)$ problems on a more fine-grained level than just the P vs. NP-complete classification. Observe that while the dichotomy theorem of Hell and Nešetřil provides a basic characterization of the complexity of $\textsc{Hom}(H)$, still, it does not say much on how quickly one can actually solve these problems. A straightforward brute-force algorithm that enumerates all the possibilities allows us to compute whether a solution exists in time $|V(H)|^n \cdot n^{\mathcal{O}(1)}$. An immediate natural question is whether this running time can be significantly improved—and there exist cases for which it can be answered affirmatively. For example, if our target graphs H are cliques, i.e., we are back in the coloring problem, the already mentioned work of Björklund et al. [6] implies that for every k the $\textsc{Hom}(K_k)$ problem can be solved in time $2^n \cdot n^{\mathcal{O}(1)}$; in particular, the exponential factor does not depend on k. There are also several algorithms for the $\textsc{Hom}(H)$ problem for general

target graphs H that achieve running times of the form $\lambda(H)^n \cdot n^{\mathcal{O}(1)}$, where $\lambda(H)$ is some structural parameter of H [43, 107, 114].

What we need to have in mind here is that the usual $\mathsf{P} \neq \mathsf{NP}$ assumption is not strong enough to obtain tight bounds for the running times of algorithms, in particular, to exclude the existence of an algorithm that solves $\textsc{Hom}(H)$ in time $\lambda(H)^{o(n)}$ for some parameter $\lambda(H)$). There are two stronger hypotheses that are commonly used in this type of lower bounds, namely the Exponential Time Hypothesis (ETH) and the Strong Exponential Time Hypothesis (SETH). The most commonly used version of the first one is actually a corollary that follows from the original conjecture and the so-called Sparsification Lemma [26, 73, 74].[1]

Exponential Time Hypothesis [73]. There is no algorithm that solves 3-Sat with n variables and m clauses in subexponential time, i.e., in time $2^{o(n+m)}$.

Strong Exponential Time Hypothesis [74]. For every $\varepsilon > 0$ there is k such that the Satk problem with n variables cannot be solved in time $2^{(1-\varepsilon)n} \cdot n^{\mathcal{O}(1)}$.

Now, by the result of Cygan et al. [25] we know that the running time given by the brute-force approach for $\textsc{Hom}(H)$ is essentially tight assuming the Exponential Time Hypothesis, as long as one considers only the dependence on the number of vertices of the instance and target graphs. In particular, either there is no universal constant c such that $\textsc{Hom}(H)$ can be solved in time $c^n \cdot n^{\mathcal{O}(1)}$ or the ETH fails. Still, it is often possible to circumvent the general lower bound and obtain significantly better time complexity if we restrict the instances of our problem in some way.

In this work we focus on the complexity of the homomorphism problems precisely from a perspective of restricting the class of the input instances. We focus on these target graphs H, for which the corresponding homomorphism problems are NP-complete. We aim to understand how the structural properties of the instances from a given class may impact the running times of the algorithms that solve $\textsc{Hom}(H)$ and $\textsc{LHom}(H)$. Clearly, it may happen that even for a restricted class of instances certain algorithms work only for some specific target graphs H. As already mentioned, we give a particular attention to many phenomena that were observed in the k-Coloring problem, and examine to which

[1] For completeness we include the formal definitions of the 3-Sat and CNF-Sat problems, as well as the other variants of Sat considered in the thesis, in the appendix.

extent they generalize to the $\textsc{Hom}(H)$ and $\textsc{LHom}(H)$ problems. The thesis collects plenty of tools that help to answer this kind of questions from various perspectives. Some of them come from the literature, others were developed specifically to attack particular problems discussed in the thesis—and usually turned out to be useful also in a more general setting.

Below we describe two ways of restricting the input instances that we consider.

1.2.1 Homomorphisms meet treewidth

One of the most natural ways to study the boundary between tractable and hard instances of graph problems is measuring the size of an instance graph not only by its number of vertices, but also by some other structural parameter of the instance. The parameters that are particularly interesting in the context of algorithmic applications are these defined by certain structural decompositions of a graph. Intuitively, having a "good" decomposition of a graph allows us to solve the problem using dynamic programming routines [7, 9, 24, 26, 35, 48, 75, 106]

Probably the best known and well-studied structural parameter of a graph is the *treewidth* (which we formally define in Chapter 2), discovered independently by several authors with different motivations [2, 61, 105]. Informally speaking, treewidth measures how similar the graph is to a tree. For many classic NP-complete problems, polynomial-time algorithms for graphs with bounded treewidth can be obtained by adapting the dynamic programming algorithms that solve problems for trees to work on *tree decompositions* of input instances. The common reason for that is captured by the powerful meta-theorem of Courcelle [23] which asserts that each problem expressible in Monadic Second Order Logic (MSO_2) can be solved in time $f(t) \cdot n$ on n-vertex graphs with treewidth t where f is some function, depending on the MSO_2 formula that describes the particular problem.

One may, however, ask about the optimal dependence on the treewidth for the particular problems, i.e., how can the function f look like. For the starting point of our investigations, namely the k-Coloring problem, a standard dynamic programming routine on a tree decomposition of the instance graph gives us the complexity $k^t \cdot n^{\mathcal{O}(1)}$, if a tree decomposition of G of width t is given. This was proven to be essentially optimal by Lokshtanov, Marx, and Saurabh.

Theorem 1.2.1 (Lokshtanov, Marx, Saurabh [87]). *For any $k \geqslant 3$, there is no algorithm that solves k-Coloring on n-vertex instances of treewidth t in time $(k-\varepsilon)^t \cdot n^{\mathcal{O}(1)}$,*

unless the SETH fails.

The standard dynamic programming for the $\textsc{Hom}(H)$ problem, if a tree decomposition of G of width t is given, has time complexity $|V(H)|^t \cdot n^{\mathcal{O}(1)}$, and the question whether this bound is tight was already studied by the author of this thesis and Rzążewski [100]. It turned out that for a large family of target graphs H the complexity $|V(H)|^t \cdot n^{\mathcal{O}(1)}$ is essentially optimal, however, in some cases, faster algorithms can be found. The authors derived tight bounds, conditioned on SETH and two conjectures from algebraic graph theory from early 2000s. As these conjectures remain wide open, we know no graph, for which the aforementioned bounds do not apply.

1.2.2 Excluding induced subgraphs

Naturally, there are other ways to restrict the input instances than bounding some structural parameter of an instance graph. Observe that the classes of graphs of bounded treewidth are *hereditary*, i.e., closed under deleting vertices. Being hereditary turns out to be a very useful property in algorithmic design, since it works well with many standard techniques, like branching or divide & conquer. Therefore, if for some computational problem we want to restrict our input instances to some class $\mathcal{G}$, choosing $\mathcal{G}$ to be hereditary appears to be reasonable.

There is another perspective to look at hereditary graph classes: by excluding induced subgraphs. For graphs F and G, we say that G is *F-free* if G does not contain F as induced subgraph. For a family $\mathcal{F}$ of graphs, we say a graph G is *$\mathcal{F}$-free* if G is F-free for every $F \in \mathcal{F}$. Now, we observe that a class $\mathcal{G}$ of graphs is hereditary if and only if there exists a (not necessarily finite) family $\mathcal{F}$ of graphs such that every $G \in \mathcal{G}$ is $\mathcal{F}$-free.

The study of graph coloring in hereditary graph classes turns out to be a very prominent topic amongst researchers [50, 53, 55, 57, 69]. Since a general classification of all such classes with regards to the complexity of graph coloring (or other classic graph problems) may be too complex, here we focus on classifying these graph classes that exclude one connected graph F.

Emden-Weinert et al. [32] proved that for every $k \geqslant 3$, the k-Coloring problem is NP-complete even for graphs of girth at least g, for any fixed g. This implies that if F contains a cycle, then k-Coloring remains NP-complete in F-free graphs, as every graph of girth at least $|V(F)| + 1$ is also F-free. Moreover, k-Coloring is NP-complete

$k\backslash t$	$\leqslant 5$	6	7	$\geqslant 8$
3	P [89]	P [104]	P [10]	Quasi-P [103]
4	P [69]	P [21]	NP-complete [71]	NP-complete [71]
$\geqslant 5$	P [69]	NP-complete [71]	NP-complete [71]	NP-complete [71]

Table 1.1: The current state of art of the complexity of k-Coloring of P_t-free graphs.

in line graphs [70, 86]. Line graphs are in particular $K_{1,3}$-free, so they form a subclass of F-free graphs when F is tree containing a vertex of degree at least 3. (We note that these NP-completeness results are also ruling out the existence of subexponential-time algorithms under ETH.) Thus, the only remaining cases are these in which F is a path. We denote by P_t a path on on t vertices. Here, complete dichotomies are known in almost all cases (see table 1.1): in particular, for every $k \geqslant 4$ there exists t such that:

- the k-Coloring problem can be solved in polynomial time in P_t-free graphs,
- for each $t' > t$ the k-Coloring problem is NP-complete in $P_{t'}$-free graphs and cannot be solved in subexponential time, unless the ETH fails.

Since for $k \leqslant 2$ the problem is polynomial-time solvable in general, the only remaining case is 3-Coloring. Here polynomial-time algorithms are known for $t \leqslant 8$, but we do not know whether the problem is NP-complete for any $t \in \mathbb{N}$. Actually, the belief that the problem is polynomial-time solvable for any constant t is supported by the existence of quasi-polynomial-time algorithms that work for every t by Pilipczuk et al. [103]. Note that if a problem for which a quasi-polyomial-time algorithm is known turns out to be NP-complete, this yields quasi-polyomial-time algorithms for all problems in NP. In particular, such result would disprove the ETH and the SETH.

It is also worth mentioning that an analogous classification is also established for the list version of the coloring problem. In most of the cases the complexities of the two problems coincide, with only one exception: 4-Coloring of P_6-free graphs is polynomial-time solvable [21], while the List-4-Coloring problem in this class is NP-complete and admits no subexponential-time algorithm, assuming the ETH [54].

The study on the complexity of the Hom(H) and LHom(H) problems in hereditary graphs classes for non-complete targets received slightly less attention. Feder, Hell and Huang [40] considered the question on whether the LHom(H) problem becomes easier if we additionally assume that the maximum degree of a vertex the instance is bounded. They have shown that for every non-bi-arc graph H the LHom(H) problem remains

NP-complete in graphs of maximum degree at most 3. One has to have in mind that since LHom(H) is a generalization of Hom(H), the hardness results for LHom(H) do not imply hardness for Hom(H). We note that the result of Feder et al. [40] cannot be generalized to Hom(H), as for example k-Coloring is polynomial-time solvable in graphs of maximum degree k. The study on the non-list variant of the question was initialized by Gallucio et al. [46] who showed that if H is an odd cycle of at least 5 vertices, then the problem is NP-complete even in subcubic graphs. Later, Siggers [109] has proved that for every non-bipartite loopless graph H there exists a constant $b(H)$ such that the Hom(H) problem remains NP-complete in graph of maximum degree at most $b(H)$. Observe that these results imply, in particular, that Hom(H) (LHom(H), respectively) remains NP-complete in F-free graph for every F that contains a vertex of degree at least $b(H)+1$ (4, respectively).

The study on the complexity of the LHom(H) problem for target graphs that are cycles was continued by Chudnovsky et al. [16]. They have showed that the LHom(C_k) problem can be solved in polynomial time in P_9-free graphs for $k \in \{5,7\} \cup [9,\infty)$, and that if F is not a subgraph of a subdivided $K_{1,3}$, then (even a certain restiction of) the LHom(C_k) problem for $k \geqslant 5$ remains NP-complete in F-free graphs. The latter was then generalized by Piecyk and Rzążewski [101, 102] who showed that if F has a connected component which is not a subgraph of a subdivided $K_{1,3}$ and H is non-bi-arc, then the LHom(H) problem remains NP-complete and cannot be solved in subexponential time, unless the ETH fails. Earlier, Groenland et al. [56] showed that for every t, the LHom(H) problem (and thus also the Hom(H) problem) can be solved in subexponential time $2^{\mathcal{O}(\sqrt{n \log n})}$ in P_t-free graphs if H does not contain two vertices with two common neighbors.

If we consider the pairs (F,H) for which the existence of subexponential-time algorithms for LHom(H) in F-free graphs is ruled out under the ETH, still not much is known about the non-list version of the problem. Clearly, there exist pairs (F,H) such that Hom(H) in F-free graphs can be solved in polynomial time, while under the ETH, there is no subexponential-time algorithm for LHom(H) in the same class: this is precisely what happens in the case of 4-Coloring P_6-free graphs. Another example of such a phenomenon follows from the work of Feder and Hell [36], who studied the Hom(H) problem in line graphs. They proved that Hom(W_5) is polynomial-time solvable in line

graphs (here the graph W_5 is the *5-wheel*, i.e., the graph obtained from the cycle C_5 by adding a universal vertex), while from the discussion on List-3-Coloring it follows that LHom(W_5) in lines graph is NP-omplete and cannot be solved in subexponential time in line graphs, unless the ETH fails.

1.3 Overview of the results

The contents of the thesis include (but are not limited to) the material from the following works.

[47] R. Ganian, T. Hamm, V. Korchemna, K. Okrasa, K. Simonov. The Fine-Grained Complexity of Graph Homomorphism Parameterized by Clique-Width, *49th International Colloquium on Automata, Languages, and Programming, ICALP 2022*, volume 229 of *LIPIcs*, pages 66:1−66:20.

[97] K. Okrasa, M. Piecyk, P. Rzążewski. Full Complexity Classification of the List Homomorphism Problem for Bounded-Treewidth Graphs. *28th Annual European Symposium on Algorithms, ESA 2020*, volume 173 of LIPIcs, pages 74:1−74:24.

[99] K. Okrasa, P. Rzążewski. Complexity of the List Homomorphism Problem in Hereditary Graph Classes. *88th Intentional Symposium on Theoretical Aspects of Computer Science, STACS 2021*, volume 187 of LIPIcs, pages 54:1−54:17.

Some of the results and examples from the previous thesis of the author is also included.

[100] K. Okrasa, P. Rzążewski. Fine-Grained Complexity of the Graph Homomorphism Problem for Bounded-Treewidth Graphs, *SIAM J. Comput.*, 50(2):487−508, 2021.

Below we summarize the three problems that are our major focus and the main contributions of the dissertation.

As outlined in the previous section, first, we are going to focus on the complexity of the graph homomorphism problems with respect to certain graph width parameters. Recall the already mentioned work of the author of this thesis and Rzążewski [100] that contains nearly-complete SETH-based lower bound result for the Hom(H) problem parameterized by treewidth. This result covers, in particular, all target graphs that are *projective cores*. While to solve the problem in full generality we may restrict our considerations only to

cores (see Chapter 3 for the discussion on this matter), it is not clear whether the same can be said about projective graphs. However, it can be shown that asymptotically almost all graphs are projective cores [65, 88], thus even though the statement of Theorem 1.3.1 below might seem quite specific, it actually covers a large class of targets. We call *non-trivial* these target graphs H for which $\textsc{Hom}(H)$ is NP-complete.

Theorem 1.3.1 (Okrasa, Rzążewski [100]). *Let H be a fixed non-trivial projective core, and let $\mathrm{tw}(G)$ be the treewidth of an n-vertex instance graph G.*

(a) Assuming a tree decomposition of G of optimal width and size polynomial in n is given, the $\textsc{Hom}(H)$ problem can be solved in time $|V(H)|^{\mathrm{tw}(G)} \cdot n^{\mathcal{O}(1)}$.

(b) There is no algorithm solving the $\textsc{Hom}(H)$ problem in time $(|V(H)| - \varepsilon)^{\mathrm{tw}(G)} \cdot n^{\mathcal{O}(1)}$, for any $\varepsilon > 0$, unless the SETH fails.

It is also worth noting that the authors showed that this result can be lifted to all targets graphs H under long-standing conjectures on the properties of projective cores. For all the formal definitions and a detailed discussion on projective graphs and conjectures we refer the reader to Chapter 3.

We follow this direction in two ways. First, we investigate the complexity bounds for $\textsc{Hom}(H)$ for another well-known parameter called *clique-width*. Second, we study the complexity parameterized by treewidth, but of the more general $\textsc{LHom}(H)$ problem.

$\textsc{Hom}(H)$ parameterized by clique-width. Recall that the $\textsc{Hom}(H)$ problem can be solved in polynomial time in graphs of bounded treewidth. It turns out that this statement can be strengthened, by replacing treewidth by a more general[2] parameter. For example, standard dynamic programming techniques can be used to obtain a $(2^{|V(H)|})^{\mathrm{cw}} \cdot n^{\mathcal{O}(1)}$ time algorithm for the problem, where cw stands for *clique-width* [24]: a well-studied graph parameter that is bounded not only on all graph classes of bounded treewidth, but also on well-structured dense classes such as complete graphs (the formal definition of clique-width is also postponed to Chapter 2).

The k-Coloring problem parameterized by clique-width was studied by Lampis [82] who showed that for $k \geqslant 3$, the k-Coloring problem can be solved in time $(2^{|V(H)|} -$

[2]Parameter λ_A is more general than parameter λ_B if there are graph classes of bounded λ_A and unbounded λ_B but the opposite is not true.

$2)^{\mathrm{cw}} \cdot n^{\mathcal{O}(1)}$ and this is tight under the SETH [82]. Observe that from his work it follows that the aforementioned straightforward $(2^{|V(H)|})^{\mathrm{cw}} \cdot n^{\mathcal{O}(1)}$ upper bound is not optimal even if we consider target graphs H that are cliques.

In our work we generalize this approach to a wider class of target graphs, in particular, to the already mentioned *projective cores*. We introduce a parameter $s(H)$, called the *signature number* of H, that, informally speaking, characterizes the number of non-trivial neighborhood classes of vertex subsets in H. We describe a dynamic programming routine on a cw-expression (which can be seen as an analogue of a tree decomposition, but for clique-width) of an n-vertex instance graph that solves the $\textsc{Hom}(H)$ problem in time $s(H)^{\mathrm{cw}} \cdot n^{\mathcal{O}(1)}$. Then we show a matching lower bound for a wide class of target graphs.

Theorem 1.3.2. *Let H be a fixed non-trivial projective core, and let $\mathrm{cw}(G)$ be the clique-width of an n-vertex instance graph G.*

(a) Assuming a cw*-expression of G of optimal width and size polynomial in n is given, the $\textsc{Hom}(H)$ problem can be solved in time $s(H)^{\mathrm{cw}(G)} \cdot n^{\mathcal{O}(1)}$.*

(b) There is no algorithm solving the $\textsc{Hom}(H)$ problem in time $(s(H) - \varepsilon)^{\mathrm{cw}(G)} \cdot n^{\mathcal{O}(1)}$, for any $\varepsilon > 0$, unless the SETH fails.

In particular, using Theorem 1.3.2 we obtain lower bounds for all non-bipartite graphs H with no loops, conditioned on the same conjectures as the discussed work of the author of this dissertation and Rzążewski for treewidth [100].

$\textsc{LHom}(H)$ parameterized by treewidth. Let us point out that if the input n-vertex graph G is given with a tree decomposition of width $\mathrm{tw}(G)$, then the straightforward dynamic programming that solves $\textsc{Hom}(H)$ problem in time $|V(H)|^{\mathrm{tw}(G)} \cdot n^{\mathcal{O}(1)}$ can easily be adapted for the list version of the problem. Moreover, since $\textsc{Hom}(H)$ is a special case of $\textsc{LHom}(H)$, the lower bounds obtained in [100] hold, in particular, for the $\textsc{LHom}(H)$ problem. This is in particular, the case for List-k-Coloring for $k \geqslant 3$, the list analogue of the k-Coloring problem.

However, one needs to have in mind that there is a family of graphs H, in particular bipartite or with loops, that are not captured by results from [100]. We point out that although algebraic methods, based on graph products, are powerful enough to be used

with almost any graph H that defines an NP-complete case of the HOM(H) problem, they do not capture the unique behavior of bipartite target graphs.

The study of the LHOM(H) problem parameterized by the treewidth of the input graph was initiated by Egri, Marx, and Rzążewski [31] who provided the tight complexity bounds in the case when H is a reflexive graph (i.e., every vertex has a loop). They have shown how we can exploit certain decompositions of target graphs to solve LHOM(H) problem, and introduced the parameter $i^*(H)$ that, roughly speaking, captures the size of the largest *incomparable set* of vertices of a target graph—a set whose each two elements' neighborhoods are incomparable with respect to the inclusion relation.

In our work we generalize the notion of decomposition, and the definition of $i^*(H)$ to all relevant target graphs and show the following dichotomy.

Theorem 1.3.3. *Let H be a non-bi-arc graph (with possible loops) and let* $\mathrm{tw}(G)$ *be the treewidth of an n-vertex instance (G, L).*

(a) Assuming a tree decomposition of G of optimal width and size polynomial in n is given, even if H is given as an input, the LHOM(H) problem can be solved in time $i^*(H)^{\mathrm{tw}(G)} \cdot (n \cdot |V(H)|)^{\mathcal{O}(1)}$.

(b) Even if H is fixed, there is no algorithm solving the LHOM(H) problem in time $(i^*(H) - \varepsilon)^{\mathrm{tw}(G)} \cdot n^{\mathcal{O}(1)}$ for any $\varepsilon > 0$, unless the SETH fails.

Hom(H) and LHom(H) in F-free graphs. As the last topic considered in the thesis we still study the homomorphism problems, but from a slightly different perspective. Denote by $S_{t,t,t}$ the graph obtained by subdividing each edge of $K_{1,3}$ exactly $t-1$ times. Recall that it was shown by Rzążewski and Piecyk [101, 102] that if F is a connected subgraph of $S_{t,t,t}$, then there is no algorithm solving the LHOM(H) problem in subexponential time, unless the ETH fails. Here, we investigate these remaining connected forbidden graphs for which we can hope for the existence of subexponential-time algorithms. We identify these graphs H for which the problem can be solved efficiently in P_t-free graphs, and show that for the every of the remaining graphs H, there exist t such that the existence of subexponential algorithms solving LHOM(H) in P_t-free graphs would violate the ETH.

The dichotomy for P_t-free instances is based on the notion of a *predacious graph* (see Chapter 7 for the precise definition). Roughly speaking, a graph is predacious if it can

be decomposed (with respect to the decompositions that appeared as a tool to prove Theorem 1.3.3) in a way that at least one part is non-bi-arc and contains two sets A, B of incomparable vertices, such that $|A|, |B| \geqslant 2$, and A and B are complete to each other.

Theorem 1.3.4. *Let H be a fixed graph.*

a) If H is not predacious, then for every $t \in \mathbb{N}$ the LHOM(H) problem can be solved in time $n^{\mathcal{O}(\log^2 n)}$ in n-vertex P_t-free graphs.

b) If H is predacious, then there exists $t \in \mathbb{N}$ such that LHOM(H) is NP-complete and cannot be solved in time $2^{o(n)}$ in n-vertex P_t-free graphs, unless the ETH fails.

We note that Theorem 1.3.4 generalizes the mentioned results on (LIST-)k-COLORING. Indeed, each complete graph K_k is undecomposable, and while K_3 does not contain two sets of incomparable vertices of size at least 2 that are complete to each other, if $k \geqslant 4$, any disjoint two-element subsets of $V(K_k)$ have this property.

Next, we investigate the complexity of LHOM(H) in $S_{t,t,t}$-free classes of graphs, and show a similar dichotomy as in Theorem 1.3.4, under an additional assumption that H is undecomposable (again, with respect to the decompositions from the proof of Theorem 1.3.3). In this case we define a class of *safe* graphs, and show the following.

Theorem 1.3.5. *Let H be a fixed undecomposable graph.*

a) If H is safe, then, for every $t \in \mathbb{N}$, the LHOM(H) problem can be solved in time $2^{\mathcal{O}(\sqrt{n}\log^2 n)}$ in n-vertex $S_{t,t,t}$-free graphs.

b) Otherwise, there exists $t \in \mathbb{N}$ such that LHOM(H) is NP-complete and cannot be solved in time $2^{o(n)}$ in n-vertex $S_{t,t,t}$-free graphs, unless the ETH fails.

We observe that the graph $S_{t,t,t}$ in the statement of Theorem 1.3.5 can be replaced by *any* subdivision F of $K_{1,3}$. This follows from the fact that each F-free graph is in particular $S_{t,t,t}$-free for $t = |V(F)|$.

Then we show to which extent the above results can be used to determine the complexity of the HOM(H) problem in graph classes defined by forbidding induced subgraphs. All the results of Chapter 7 serve also as an illustration of how the tools developed in the previous chapters to prove Theorem 1.3.2 and Theorem 1.3.3 can be applied to obtain other homomorphism-related theorems.

Last, let us stress out that while the three papers included in the dissertation, together with [100], seem to be the most important contribution of the author in the topic of graph homomorphism, several other works related to the area could be considered here. We briefly survey these in the last chapter of the thesis.

1.4 Organization of the thesis

Chapter 2 contains basic graph-theoretic notation and short introduction to graph parameters that are investigated in this thesis. Chapter 3 contains the necessary definitions for algebraic tools, including (direct) graph products, projective graphs and constructible sets. We show plenty of connections between these concepts, and describe how graph products can be used to construct gadgets that can be, in turn, used in various kinds of reductions. Then, in Chapter 4 we discuss the complexity of $\textsc{Hom}(H)$ when parameterized by clique-width, and in particular, prove Theorem 1.3.2.

In Chapter 5 we focus on combinatorial techniques that are especially useful when working with the list version of the homomorphism problem. We introduce certain decompositions of the target graphs, and present series of structural lemmas that later allow us, in Chapter 6, to prove Theorem 1.3.3 We provide a general algorithm that can be used to derive a series of results on list homomorphisms. We also construct a general gadget that can express any relation on incomparable sets. We conclude this chapter by demonstrating how both, the algorithm and the gadget construction can be used to provide the dichotomy stated in Theorem 1.3.3.

In Chapter 7 we introduce the auxiliary tools to work with P_t-free and $S_{t,t,t}$-graphs, and prove Theorem 1.3.4 and Theorem 1.3.5. Furthermore, we demonstrate how the lower bounds that follow from Theorem 1.3.4 and Theorem 1.3.5 can be used to derive analogous results for the $\textsc{Hom}(H)$ problem. Last, in Chapter 8 we discuss the other results of the author related to the topic of the dissertation.

Chapter 2

Preliminaries

We denote by $\mathbb{R}$ the set of real numbers, by $\mathbb{R}_+$ the set of positive real numbers, by $\mathbb{N}$ the set of positive integers. For an integer $n \in \mathbb{N}$ we denote by $[n]$ the set of integers $\{1, \ldots, n\}$. For a set X and integer k, by 2^X we denote the family of all subsets of X and by $\binom{X}{k}$ (resp. $\binom{X}{\leqslant k}$) we denote the family of all subsets of X with exactly (resp. at most) k elements.

2.1 Graph notations

For a graph G, we denote by $V(G)$ and $E(G)$, respectively, the set of vertices and the set of edges of G. A vertex v with loop is called *reflexive*, otherwise v is *irreflexive*. A set of vertices is called *(ir)reflexive* if its every element is (ir)reflexive. Similarly, a graph G is called *(ir)reflexive* if $V(G)$ is (ir)reflexive. Unless explicitly stated otherwise, we assume that graphs are finite, undirected, irreflexive and with no multiple edges.

By $N_G(v)$ we denote the set of neighbors of v and by $N_G[v]$ we denote the set $N_G(v) \cup \{v\}$. Note that if v is reflexive, then $v \in N_G(v)$, so $N_G(v) = N_G[v]$. For a set $U \subseteq V(G)$, by $N_G(U)$ we denote the set $\bigcup_{u \in U} N_G(u) \setminus U$, and by $N_G[U]$ the set $N_G(U) \cup U$. For a vertex $v \in V(G)$ the degree $\deg_G v$ of G is the number of neighbors of v. We omit the subscript and write $N(v)$, $N[v]$, $N(U)$, $N[U]$, $\deg v$ respectively, if G is clear from the context. Also, if $U = \{u_1, \ldots, u_k\}$, we omit internal brackets and write $N(u_1, \ldots, u_k)$ and $N[u_1, \ldots, u_k]$. By $\Delta(G)$ we denote $\max_{v \in V(G)} \deg v$.

For a set $U \subseteq V(G)$ (resp. $U \subseteq E(G)$) we denote by $G[U]$ the subgraph of G induced by U, i.e., $(U, \{uv \in E(G) \mid u, v \in U\})$ (resp. $(\bigcup_{uv \in U} \{u, v\}, U)$). By $G - U$ we denote the

graph $G[V(G) \setminus U]$.

We say that two vertices u, v of G are *incomparable* if $N(u) \not\subseteq N(v)$ and $N(v) \not\subseteq N(u)$. We say that a set S of vertices is *incomparable* if its elements are pairwise incomparable.

Let $k \in \mathbb{N}$. A *clique* K_k is the complete graph on k vertices. A *path* P_k is a graph such that $V(P_k) = \{w_1, \ldots, w_k\}$, and $w_i w_j \in E(P_k)$ for some $i, j \in [k]$ if and only if $i \in \{j-1, j+1\}$. If $k \geqslant 3$ by C_k we denote a *cycle*, i.e., $V(C_k) = \{w_1, \ldots, w_k\}$, and $w_i w_j \in E(C_k)$ for some $i, j \in [k+1]$ if and only if $i \in \{j-1, j+1\}$ modulo k.

A *walk* W in G of *length* d is a sequence $(w_1, \ldots, w_{d+1})$ of vertices of G, such that for every $i \in [d]$ vertices w_i and w_{i+1} are adjacent. In order to simplify the notation, whenever is does not lead to confusion we will identify sequences of the vertices with sets of these vertices. The *length* of a path P_k is the length of the walk that uses all vertices of P_k precisely once, i.e., $k-1$. The length of a cycle C_k is k. We call a cycle of odd (resp. even) length an *odd cycle* (resp. *even cycle*). A *bipartite* graph is a graph that does not contain odd cycles. For $p, q \in \mathbb{N}$, by $K_{p,q}$ we denote the *complete bipartite graph*, i.e. graph consisting of two sets of vertices of cardinality, respectively, p and q, and all possible edges between them. The *odd girth* $\mathrm{og}(G)$ of a non-bipartite graph G is the length of the smallest odd cycle in G. By $\omega(G)$ we denote the size of a largest clique contained in G. A *matching* in G is a subset $M \subseteq E(G)$ such that each vertex is incident to at most one vertex in M.

A subset U of vertices of a graph G is *independent* if they are pairwise non-adjacent in G. For two subsets U_1 and U_2 of $V(G)$ by $\mathrm{dist}(U_1, U_2)$ (resp. $\mathrm{dist}_W(U_1, U_2)$ for some $W \subseteq V(G)$) we denote the length of a shortest path between an element of U_1 and an element of U_2 (resp. using only vertices of W), note that $\mathrm{dist}(U_1, U_2) = 0$ if and only in $U_1 \cap U_2 \neq \emptyset$. We say that two disjoint subsets U_1, U_2 of $V(G)$ are *non-adjacent* if there is no edge with one vertex in U_1 and the other in U_2. Analogously, we say that U_1, U_2 are *complete* to each other if for every $u_1 \in U_1$ and $u_2 \in U_2$ we have $u_1 u_2 \in E(G)$.

If a graph G admits a k-coloring, we say that G is k-colorable. The *chromatic number of* G, denoted by $\chi(G)$, is the smallest integer k such that G is k-colorable.

2.2 Graph parameters

Tree decompositions and treewidth. A *tree decomposition* of a graph G is a pair $\left(\mathcal{T}, \{X_a\}_{a \in V(\mathcal{T})}\right)$, in which $\mathcal{T}$ is a tree, whose vertices are called *nodes* and $\{X_a\}_{a \in V(\mathcal{T})}$ is the family of subsets (called *bags*) of $V(G)$, such that

1. every $v \in V(G)$ belongs to at least one bag X_a,
2. for every $uv \in E(G)$ there is at least one bag X_a such that $u, v \in X_a$,
3. for every $v \in V(G)$ the set $\mathcal{T}_v := \{a \in V(\mathcal{T}) \mid v \in X_a\}$ induces a connected subgraph of $\mathcal{T}$.

The *width* of a tree decomposition $\left(\mathcal{T}, \{X_a\}_{a \in V(\mathcal{T})}\right)$ is the number $\max_{a \in V(\mathcal{T})} |X_a| - 1$. The minimum possible width of a tree decomposition of G is called the *treewidth* of G and denoted by $\mathrm{tw}(G)$.

t-expressions and clique-width. For a positive integer t, let a *t-graph* be a graph whose vertices are labeled by the elements of $[t]$. For convenience, we consider a graph to be a t-graph with all vertices labeled by 1. Two t-graphs are isomorphic if their underlying unlabeled graphs are isomorphic. We call the t-graph consisting of exactly one vertex v, labeled by $i \in [t]$, an *initial* t-graph and denote it by $i(v)$.

We consider the following three operations on t-graphs.

(1) Disjoint union: for two t-graphs G_1 and G_2 by $G_1 \oplus G_2$ we denote their disjoint union;

(2) Relabeling: for a t-graph G and distinct $i, j \in [t]$, by $\rho_{i \to j}(G)$ we denote the t-graph obtained by changing all labels on vertices labeled i to j;

(3) Edge insertion: for a t-graph G and distinct $i, j \in [t]$, by $\eta_{i,j}(G)$ we denote the t-graph obtained by adding an edge from each vertex labeled by i to each vertex labeled by j.

For a t-graph G a *t-expression σ defining G* is a rooted binary tree (whose vertices we call *nodes*) defined as follows. Here, we associate with every node τ of σ a maximal subtree of σ rooted at τ. Each τ corresponds to a graph G_τ isomorphic to some subgraph of G. We call G_τ the *evaluation* of τ. Each non-root leaf node τ of σ corresponds to an unique $v \in V(G)$ and $G_\tau = i(v)$ for some $i \in [t]$. Each other node τ of σ is labeled with one of the symbols $\oplus, \rho_{i \to j}$ or $\eta_{i,j}$ (where $i, j \in [t]$) called the *root operation* of τ, so that:

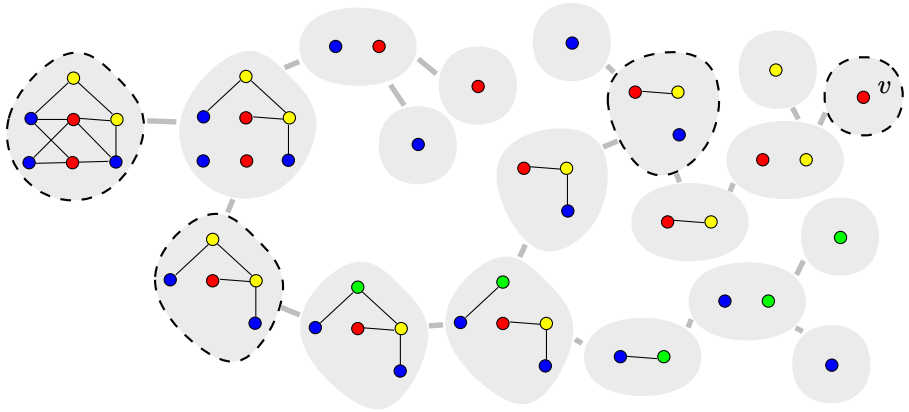


Figure 2.1: An 4-expression defining graph G (in the left-most node). Different labels are represented by different colors. The nodes marked with dashed lines are nodes with root operation being, respectively (from left to right): edge insertion, relabeling, disjoint union, and an initial node.

1. Each node labeled $\oplus$ has precisely two children τ_1 and τ_2, and $G_\tau = G_{\tau_1} \oplus G_{\tau_2}$.

2. Each node labeled $\rho_{i \to j}$, for some distinct $i, j \in [t]$ has an unique child τ', and $G_\tau = \rho_{i \to j}(G_{\tau'})$.

3. Each node labeled $\eta_{i,j}$, for some distinct $i, j \in [t]$ has an unique child τ', and $G_\tau = \eta_{i,j}(G_{\tau'})$.

In other words, each node τ corresponds to a t-graph G' that can be constructed by applying the root operation of τ to the t-graphs corresponding to the children of τ (see Figure 2.1). By $V_\tau^i \subseteq V(G_\tau)$ we denote the set of vertices with label i in G_τ.

If σ is a t-expression, we call t the *width* of σ. For a graph G, the *clique-width* of G is the smallest k such that there exists a t-expression σ defining a k-graph isomorphic to G.

Chapter 3

Graph homomorphisms: toolbox

In this chapter we present a variety of notions regarding homomorphisms in the non-list setting. In particular, we use the tools introduced below to prove Theorem 1.3.2, which is demonstrated in Chapter 4. Further applications of the contents of the present chapter are discussed in the last section of Chapter 7.

3.1 Basic concepts

For two graphs G and G, if there exists a homomorphism $f : G \to H$, we denote it by $G \to H$. If there is no such homomorphism, we write $G \not\to H$. If $G \to H$ and $H \to G$, then we say that G and H are *homomorphically equivalent*. If $G \not\to H$ and $H \not\to G$, we say that G and H are *incomparable*. An *endomorphism* of G is any homomorphism $h : G \to G$. An *automorphism* of G is a bijective endomorphism. If $X \subseteq Y \subseteq V(G)$, and $f : X \to V(H)$, then a homomorphism $g : G[Y] \to H$ such that $g|_X \equiv f$ is an *extension* of f (to Y).

A graph G is a *core* if $G \not\to H$ for every proper subgraph H of G. Equivalently, we can say that G is a core if and only if every endomorphism of G is an automorphism. A graph G is *ramified* if for every distinct $u, v \in V(G)$ we have $N(u) \not\subseteq N(v)$. Note that if $N(u) \subseteq N(v)$, then there always exists a homomorphism $h : G \to G$ that is not an automorphism: one could define $h : G \to G$ that is an identity on all vertices of G but u and set $h(u) = v$. This implies the following observation.

Observation 3.1.1. *Every core is ramified.* □

If H is a subgraph of G such that $G \to H$ and H is a core, we say that H is a core of G.

Notice that if H is a subgraph of G, then $H \to G$, so every graph is homomorphically equivalent to its core. Moreover, if H is a core of G, then H is always an induced subgraph of G, because every endomorphism $f : G \to H$ restricted to H must be an automorphism. It can be shown that every graph has a unique core (up to an isomorphism), see for example [65]. Therefore, when considering the $\textsc{Hom}(H)$ problem, usually it is enough to focus only on these graphs H that are cores.

We observe that if H contains a reflexive vertex v, then, for any graph G, there always exists a homomorphism from G to H. Indeed, in a such case, the function that maps every vertex of G to the vertex v is always a homomorphism. Thus, if H contains a vertex with a loop, its core is always the graph that consists of one vertex with a loop, that we denote by K_1°. On the other hand, since loop is also an edge, note that if $v \in V(G)$ has a loop and $f : G \to H$, then $f(v)$ must have a loop.

We say that a graph H is *trivial* if its core is isomorphic to K_1, K_1°, or K_2. It is straightforward to verify that these three graphs are the only cores with fewer than 3 vertices. The graphs whose cores are trivial are simple to describe.

Observation 3.1.2. *Let H' be a graph, whose core H is trivial.*

1. *$H \simeq K_1$ if and only if $\chi(H') = 1$, i.e., H' has no edges,*
2. *$H \simeq K_2$ if and only if $\chi(H') = 2$, i.e., H' is bipartite and has at least one edge,*
3. *$H \simeq K_1^\circ$ if and only if H' is not irreflexive.*

In particular, every graph that has a loop is trivial. □

Observation 3.1.2 together with the dichotomy theorem of Hell and Nešetřil [67] imply that H is trivial if and only if $\textsc{Hom}(H)$ is polynomial-time solvable. Since most of the time we are interested in NP-complete cases of $\textsc{Hom}(H)$, we usually focus on non-trivial target graphs. Note that if H is non-trivial and an instance G of $\textsc{Hom}(H)$ has a loop, then G is clearly a no-instance—thus we always implicitly assume that an instance of $\textsc{Hom}(H)$ is irreflexive.

The following conditions are necessary for G to have a homomorphism into H.

Observation 3.1.3 ([66]). *Assume that $G \to H$ and G and H are irreflexive. Then $\omega(G) \leqslant \omega(H)$, $\chi(G) \leqslant \chi(H)$, and $\mathrm{og}(G) \geqslant \mathrm{og}(H)$.*

Let $m \in \mathbb{N}$. For graphs $H_1, \dots H_m$ we denote by $H_1 + \dots + H_m$ the disconnected graph with connected components $H_1, \dots, H_m$. Observe that if f is a homomorphism from $G = G_1 + \dots + G_\ell$ to $H = H_1 + \dots + H_m$, then it maps every connected component of G to some connected component of H. Also note that a graph does not have to be connected to be a core, in particular the following characterization follows directly from the definition of a core.

Observation 3.1.4. *A disconnected graph H is a core if and only if its connected components are pairwise incomparable cores.* □

An example of a pair of incomparable cores is shown in Figure 3.1: it is the *Grötzsch graph*, denoted by G_G, and the complete graph K_3. It can be verified that $\mathrm{og}(G_G) > \mathrm{og}(K_3)$ and $\chi(G_G) > \chi(K_3)$, so by Observation 3.1.3, G_G and K_3 are incomparable. Therefore, by Observation 3.1.4, the graph $G_G + K_3$ is a core.

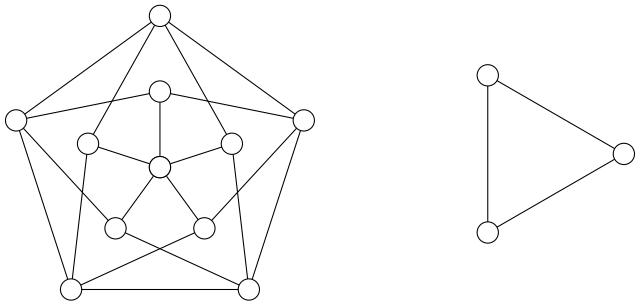


Figure 3.1: An example of incomparable cores, the *Grötzsch graph* (left) and K_3 (right).

3.2 Projectivity

For graphs H_1, H_2, we define their *direct product*, denoted by $H_1 \times H_2$ to be the graph:

$$V(H_1 \times H_2) := \{(x, y) \mid x \in V(H_1) \text{ and } y \in V(H_2)\} \text{ and}$$
$$E(H_1 \times H_2) := \{(x_1, y_1)(x_2, y_2) \mid x_1x_2 \in E(H_1) \text{ and } y_1y_2 \in E(H_2)\}.$$

If $H = H_1 \times H_2$, then the tuple (H_1, H_2) is a *factorization* of H, and H_1 and H_2 are its *factors*. Clearly, the binary operation $\times$ is commutative, and also associative (up to isomorphism), thus we can extend the definition of a direct product for more than two factors:

$$H_1 \times \dots \times H_{m-1} \times H_m := (H_1 \times \dots \times H_{m-1}) \times H_m.$$

In this and the following chapters we are going to consider products of graphs that are products themselves. Formally, the vertices of such graphs are tuples of tuples. If it does not lead to confusion, for $\bar{x} := (x_1, \ldots, x_{k_1})$ and $\bar{y} := (y_1, \ldots, y_{k_2})$, we will treat tuples $(\bar{x}, \bar{y})$, $(x_1, \ldots, x_{k_1}, y_1, \ldots, y_{k_2})$, $(\bar{x}, y_1, \ldots, y_{k_2})$, and $(x_1, \ldots, x_{k_1}, \bar{y})$ as equivalent. This notation is generalized to more factors in a natural way. Moreover, for any graph H and for an integer m, we denote by H^m the graph

$$H^m := \overbrace{H \times \ldots \times H}^{m}.$$

As an example, instead of writing $((x_1, x_2), y_1) \in V((H_1 \times H_1) \times H_2)$, we write $(x_1, x_2, y_1) \in V(H_1^2 \times H_2)$.

Note that if $H_1 \times H_2$ has at least one edge, then $H_1 \times H_2 \simeq H_1$ if and only if $H_2 \simeq K_1^\circ$. We say that a graph H on at least two vertices is *prime* if the fact that (H_1, H_2) is a factorization of H implies that either $H_1 \simeq K_1^\circ$ or $H_2 \simeq K_1^\circ$. A graph that is not prime, is *decomposable*. A factorization where each factor is prime and not isomorphic to K_1°, is called a *prime factorization*.

Recall that a graph is trivial if and only if its core has fewer than 3 vertices, and that by Observation 3.1.2, any bipartite graph is trivial. It turns out that connected non-trivial graphs have unique prime factorizations (see also Theorem 8.17 in [63]).

Theorem 3.2.1 (McKenzie [93]). *Any connected non-bipartite graph with more than one vertex has a unique prime factorization into prime factors (with possible loops), up to the permutation of factors.*

Let $\lambda = (H_1, \ldots, H_m)$ be a (not necessarily prime) factorization of a graph H and let $i \in [m]$. The function $\pi_i^\lambda : V(H) \to V(H_i)$ such that for every $(x_1, \ldots, x_m) \in V(H)$ it holds that $\pi_i^\lambda(x_1, \ldots, x_m) = x_i$ is a *projection on the i-th coordinate*. If a factorization λ is clear from the context, we omit the superscript and write π_i. The definition of the direct product implies that every projection π_i^λ is a homomorphism from H to H_i.

Below we summarize some basic properties of direct products.

Observation 3.2.2. *Let $H = H_1 \times \ldots \times H_m$ be a graph.*

1. *If $H_i = K_1$ for some i, then H consists of isolated verticess.*
2. *If some H_i is bipartite then H is bipartite.*

3. *If* $H_1 = \ldots = H_m$*, (i.e.,* $H = H_1^m$*) then* $\{(x, \ldots, x) \mid x \in V(H_1)\}$ *induces a copy of* H_1 *in* H*. In particular, if* $m \geqslant 2$ *in a such case, then* H *is never a core.*

4. *If* $H_1, H_2, \ldots, H_m$ *are connected, then* H *is connected if and only if at most one* H_i *is bipartite.*

5. *For every* G *it holds that* $G \to H$ *if and only if* $G \to H_i$ *for all* $i \in [m]$*.*

Proof. Items (1), (2), (3) are straightforward to observe. Item (4) follows from a result of Weichsel [115], see also [63, Corollary 5.10]. To prove (5), consider a homomorphism $f : G \to H$ and let $\lambda = (H_1, \ldots, H_m)$. Clearly, $H \to H_i$ for every $i \in [m]$ because each projection $\pi_i^\lambda : H \to H_i$ is a homomorphism. So $\pi_i^\lambda \circ f$ is a homomorphism from G to H_i. On the other hand, if we have some $f_i : G \to H_i$ for every $i \in [m]$, then we can define a homomorphism $f : G \to H$ by setting $f(x) := (f_1(x), \ldots, f_m(x))$ for every $v \in V(G)$. □

The following observation is a consequence of Observation 3.2.2 and the definition of direct product.

Observation 3.2.3. *Let* H *be a non-trivial connected core, and let* $(H_1, \ldots, H_m)$ *be its factorization, such that* $H_i \not\simeq K_1^\circ$ *for all* $i \in [m]$*. Then for* $i \in [m]$ *the graph* H_i *is a non-trivial connected core, incomparable with* H_j *for* $j \in [m] \setminus \{i\}$*.*

Proof. By the fact that direct product is commutative, it is enough to prove that H_1 a non-trivial connected core, incomparable with H_2. To see that H_1 is non-trivial, assume otherwise. By our assumption, $H_1 \not\simeq K_1^\circ$. If $H_1 \simeq K_1$ (resp. $H_1 \simeq K_2$), then by Observation 3.2.2 (1) (resp. Observation 3.2.2 (2)), the core of H is K_1 (resp. K_2). As H is non-trivial, we reach a contradiction.

The fact that H_1 is connected follows from the definition of direct product. Now suppose that H_1 is not a core and let H_1' be the core of H_1, so a proper induced subgraph of H_1. Define $H' = H_1' \times H_2 \times \ldots \times H_m$. Since H_1' is a proper induced subgraph of H_1, graph H' is a proper induced subgraph of H. Let $f : H_1 \to H_1'$. Define a homomorphism $f' : H \to H'$ as follows: $f'(x_1, \ldots, x_m) := (f(x_1), x_2, \ldots, x_m)$. Observe that since f is a homomorphism, so is f'. Therefore H admits a homomorphism into its proper induced subgraph, so H is not a core, a contradiction.

Finally, assume that H_1 and H_2 are not incomparable. Without loss of generality, there exists $g : H_2 \to H_1$. Then, the set $\{(g(x_2), x_2, \ldots, x_m) \in V(H) \mid x_2 \in V(H_2)\}$

induces a subgraph isomorphic to $H_2 \times H_3 \times \ldots \times H_m$ in H. Since we always have $H \to H_2 \times H_3 \times \ldots \times H_m$, and H is a core, it follows that $H \simeq H_2 \times H_3 \times \ldots \times H_m$. But this implies that $H_1 \simeq K_1^\circ$, a contradiction. □

A homomorphism $f : H^m \to H$ is *idempotent*, if for every $x \in V(H)$ it holds that $f(x, x, \ldots, x) = x$. Observe that each projection from H^m to H is idempotent. One of the main characters in our study of the HOM(H) problem is the class of *projective graphs* (also called *essentially unary*), considered e.g. in [77, 85, 96, 110]. Let $m \geqslant 2$. We say that a graph H is *m-projective* if every idempotent homomorphism from H^m to H is a projection. We say that a graph H is *projective*, if it is m-projective for every $m \geqslant 2$. Intuitively, a graph is projective, if all homomorphisms of the form $f : H^m \to H$ for $m \geqslant 2$ depend on a single argument of f. This is formalized as follows.

Observation 3.2.4. *If H is a projective core and $f : H^m \to H$ is a homomorphism, then $f \equiv g \circ \pi_i$ for some $i \in [m]$ and some automorphism g of H.*

Proof. Define $g : V(H) \to V(H)$ by $g(x) := f(x, \ldots, x)$. The function g is an endomorphism of H and H is a core, so g is in fact an automorphism of H. Observe that $g^{-1} \circ f$ is an idempotent homomorphism, so it is equal to π_i for some $i \in [m]$, because H is projective. From this we get that $f \equiv g \circ \pi_i$. □

It is known that projective graphs are always connected [85]. Many graphs are known to be projective: it includes e.g., Kneser graphs (so in particular complete graphs), and non-bipartite, connected, ramified graphs which do not contain C_4 as a subgraph (so in particular odd cycles) [83]. Recall also that asymptotically almost all graphs are projective [88].

We note that if H is non-trivial and projective, then it must be prime. Indeed, assume that (H_1, H_2) is a factorization of some non-trivial H, and that $H \not\simeq K_1^\circ$, $H_2 \not\simeq K_1^\circ$. Consider a homomorphism $f : (H_1 \times H_2)^2 \to H_1 \times H_2$, defined as $f((x, y), (x', y')) = (x, y')$. Note that it is idempotent, but not a projection, so H is not projective.

On the other hand, we do not know whether there are any non-projective non-trivial connected cores that are prime. This problem was studied in a slightly more general setting by Larose and Tardif [85, Problem 2], and it remains wide open. We restate it as a conjecture, only for restricted case that H is a core, which is sufficient for our purpose.

Conjecture 1. *Let H be a connected non-trivial core. Then H is projective if and only if it is prime.*

If it comes to core graphs, it appears that even for connected graphs the properties of being projective and being a core are independent. In particular, the graph in Figure 3.2 is not a core, but is projective. Indeed, it is not a core, since it can be mapped to a triangle, but as already mentioned, Larose [83] proved that all non-bipartite, connected, ramified graphs which do not contain C_4 as a (non-necessarily induced) subgraph, are projective. On the other hand, there are non-projective cores, an example is $G_G \times K_3$, see Figure 3.1.[1]

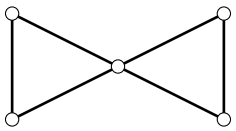


Figure 3.2: An example of a projective graph which is not a core.

3.2.1 Constructible sets

Let H be a graph, let $t \geqslant 1$, and let $\mathcal{S} \subseteq V(H)^t$. A triple $(F, \varphi, \overline{x})$, such that F is a graph, $\varphi : V' \to V(H)$ is a mapping with domain $V' \subseteq V(F)$, and $\overline{x} = (x_1, \ldots, x_t) \in V(F)^t$, is a *construction of* $\mathcal{S}$ if

$$\{(h(x_1), h(x_2), \ldots, h(x_t)) \mid h : F \to H \text{ is an extension of } \varphi\} = \mathcal{S}.$$

See Figure 3.3 for an example.

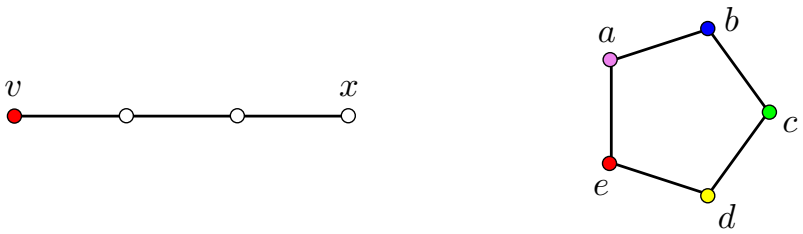


Figure 3.3: An example of a construction $(F, \varphi, (x))$ (left) of a subset $\mathcal{S} = \{a, b, c, d\}$ of vertices of H (right). Here, F is a path on four vertices, $V' = \{v\}$, and φ maps v to e.

[1] Since we were not able to find a simple argument for proving that $G_G \times K_3$ is a core, this fact was verified by exhaustive computer search.

We say that $\mathcal{S} \subseteq V(H)^t$ is *constructible* if there exists a construction of $\mathcal{S}$. In particular, if $(F, \varphi, \overline{x})$ is a construction of $\mathcal{S}$ such that φ is of empty domain, we omit φ, write $(F, \overline{x})$, and call $(F, \overline{x})$ an *$\mathcal{S}$-gadget*.

The relation between projectivity of a graph H and the ability to construct the subsets of $V(H)$ was estabilished by Larose and Tardif.

Theorem 3.2.5 (Larose, Tardif [85]). *A graph H on at least three vertices is projective if and only if every set $\mathcal{S} \subseteq V(H)$ is constructible.*

We observe that Theorem 3.2.5 can be restated as follows.

Corollary 3.2.6. *A graph H on at least three vertices is projective if and only if for every $t \in \mathbb{N}$ and every $\mathcal{S} \subseteq V(H)^t$ there exists a construction of $\mathcal{S}$.*

Proof. The backward implication follows from Theorem 3.2.5 for $t = 1$, so we need to show the forward implication. Let $p = |\mathcal{S}|$, and let $\mathcal{S} = \{(s_1^1, \ldots, s_t^1), \ldots, (s_1^p, \ldots, s_t^p)\}$. Define $F = H^p$, $V' = \{(a, \ldots, a) \mid a \in V(H)\} \subseteq V(F)$, and $\varphi : V' \to V(H)$ to be the function that maps every $(a, \ldots, a) \in V(F)$ to a. Moreover, define $\overline{x} = (x_1, \ldots, x_t) \subseteq V(F)^t$ so that $x_i = (s_i^1, s_i^2, \ldots, s_i^p)$ for every $i \in [t]$. We claim that $(F, \varphi, \overline{x})$ is a construction of $\mathcal{S}$.

Let $(s_1^j, \ldots, s_t^j) \in \mathcal{S}$ for some $j \in [p]$. We define $f : F \to H$ to be the j-th projection π_j. Clearly, π_j is an idempotent homomorphism and thus an extension of φ. Since

$$(f(x_1), \ldots, f(x_t)) = (s_1^j, \ldots, s_t^j),$$

we have that $\{(h(x_1), \ldots, h(x_t)) \mid h : F \to H \text{ is an extension of } \varphi\} \supseteq \mathcal{S}$.

On the other hand, assume that $h : F \to H$ is an extension of φ. Then, by definition of φ, h must be idempotent. Hence, as H is projective, $f \equiv \pi_j$ for some $j \in [p]$, i.e., f is a projection on the j-th coordinate. Thus, $(h(x_1), h(x_2), \ldots, h(x_t)) = (s_1^j, \ldots, s_t^j) \in \mathcal{S}$, which means that $\{(h(x_1), \ldots, h(x_t)) \mid h : F \to H \text{ is an extension of } \varphi\} \subseteq \mathcal{S}$. This concludes the proof. □

To provide constructions that work not only for projective graphs, we are going to have a closer look on how constructible sets behave on decomposable graphs. For that we generalize notions of idempotency and projectivity that we introduced in the previous section. Let H be a graph with factorization (H_1, H_2), such that H_1 is non-trivial, and H_2 is either non-trivial, or isomorphic to K_1°. A homomorphism $f : H_1^\ell \times H_2 \to H_1$

such that for every $x \in V(H_1), y \in V(H_2)$, it holds that $f(x, x, \ldots, x, y) = x$ is called *H_1-idempotent*. The graph H is said to be *H_1-projective*, if for every $\ell \geqslant 2$ and every H_1-idempotent homomorphism $f : H_1^\ell \times H_2 \to H_1$ there exists $i \in [\ell]$ such that $f \equiv \pi_i$. We note that if $H_2 = K_1^\circ$, then H is H_1-projective if and only if it is projective.

Further generalizations of the projectivity property were studied by Larose [83, 84], Nešetřil and Siggers [95]. In [84] *strongly projective* graphs were defined: a graph H_1 on at least three vertices is *strongly projective* if for every connected graph W on at least two vertices, and every $\ell \geqslant 2$ the only H_1-idempotent homomorphisms $f : H_1^\ell \times W \to H_1$ are projections $\{\pi_i^\lambda\}_{i \in [\ell]}$, where $\lambda = (H_1, \ldots, H_1, W)$ is a factorization of $H_1^\ell \times W$. Note that if H_1 is strongly projective, then, in particular, every graph H that is of the form $H_1 \times W$, for some graph W, is H_1-projective.

Interestingly, Larose [83, 84] proved that members of all known families of projective graphs are in fact strongly projective. He also asked whether the same holds for all projective graphs. We state as a conjecture a weaker form of his question, that is sufficient in our setting.

Conjecture 2. *Let H_1 be a projective graph. Then every core H with factorization (H_1, W), for some graph W, is H_1-projective.*

The next lemma is a generalization of Corollary 3.2.6 and establishes the relation between H_1-projective graphs and constructions of certain relations. It plays a crucial role in the proof of lower bounds in Chapter 4. Let $t \in \mathbb{N}$, and let H be a graph with factorization $\lambda = (H_1, W)$. For a relation $\mathcal{S} \subseteq V(H_1)^t$, and $\overline{y} = (y_1, \ldots, y_t) \in V(W)^t$, we denote by $R[\mathcal{S}, \overline{y}] \subseteq V(H)^t$ a t-ary relation such that:

(R1) for every $(s_1, \ldots, s_t) \in \mathcal{S}$ we have $((s_1, y_1), \ldots, (s_t, y_t)) \in R[\mathcal{S}, \overline{y}]$,

(R2) for every $((s_1, y_1'), \ldots, (s_t, y_t')) \in R[\mathcal{S}, \overline{y}]$ we have $(s_1, \ldots, s_t) \in \mathcal{S}$.

We emphasize that $R[\mathcal{S}, \overline{y}]$ does not have to be unique.

Lemma 3.2.7. *Let $t \in \mathbb{N}$, and let H be a graph with factorization $\lambda = (H_1, W)$, such that H_1 is non-trivial, and W is either non-trivial or isomorphic to K_1°. Assume that H is H_1-projective. Let $\mathcal{S} \subseteq V(H_1)^t$, and let $\overline{y} = (y_1, \ldots, y_t) \in V(W)^t$. Then there exists a construction $(F, \varphi, \overline{x})$ of $R[\mathcal{S}, \overline{y}]$. Moreover, $|F| \leqslant |V(H)|^t$.*

Proof. Let $p = |\mathcal{S}|$, and let $\mathcal{S} = \{(s_1^1, \ldots, s_t^1), \ldots, (s_1^p, \ldots, s_t^p)\}$. We define $F = H_1^p \times W$, $V' = \{(a, \ldots, a, b) \mid a \in V(H_1), b \in V(W)\} \subseteq V(F)$, and $\varphi : V' \to V(H_1 \times W)$ to be the function that maps every $(a, \ldots, a, b)$ to (a, b). Moreover, define $\overline{x} = (x_1, \ldots, x_t) \subseteq V(F)^t$ so that for every $i \in [t]$ we have

$$x_i = (s_i^1, s_i^2, \ldots, s_i^p, y_i).$$

Let α be a factorization of H_1^p that is a p-tuple $(H_1, \ldots, H_1)$. Let

$$R[\mathcal{S}, \overline{y}] = \{(h(x_1), \ldots, h(x_t)) \mid h : H_1^p \times W \to H_1 \times W \text{ is an extension of } \varphi\}.$$

We claim that $R[\mathcal{S}, \overline{y}]$ satisfies (R1) and (R2).

Let $(s_1^j, \ldots, s_t^j) \in \mathcal{S}$ for some $j \in [p]$. We define $f : F \to H_1 \times W$:

$$f(v_1, \ldots, v_t, y') = (\pi_j^\alpha(v_1, \ldots, v_t), y').$$

As $(f(x_1), \ldots, f(x_t)) = ((s_1^j, y_1), \ldots, (s_t^j, y_t))$,$f$ is a homomorphism, and an extension of φ, we have that $R[\mathcal{S}, \overline{y}]$ satisfies (R1).

For (R2), let $h : F \to H_1 \times W$ be an extension of φ. By the definition of φ, a homomorphism $h_1 : H_1^p \times W \to H_1$, defined as $h_1 \equiv \pi_1^\lambda \circ h$ must be H_1-idempotent. Hence, as H is H_1-projective, $h_1 \equiv \pi_i^\alpha$ for some $j \in [p]$, i.e., h_1 is the projection on the j-th coordinate. Thus, $(h(x_1), h(x_2), \ldots, h(x_t)) = ((s_1^j, y_1'), \ldots, (s_t^j, y_t'))$ for some $(s_1^j, \ldots, s_t^j) \in \mathcal{S}$ and $y_1', \ldots, y_t' \in V(W)$, which means that $R[\mathcal{S}, \overline{y}]$ satisfies (R2). Since $|F| \leqslant |V(H)|^t$, this concludes the proof. □

Here we stated and proved Lemma 3.2.7 in a very general form. We note however that an analogous method was already used by Okrasa and Rzążewski [100] to determine tight complexity bounds for the Hom(H) problem parameterized by treewidth of the instance graph.

Lemma 3.2.8 ([100]). *Let H be a non-trivial core with factorization $\lambda = (H_1, W)$, such that H_1 is non-trivial, and W is either non-trivial or isomorphic to K_1°. Assume that H is H_1-projective. Let $\mathcal{S} \subseteq V(H)^2$ be the inequality relation, i.e., $\mathcal{S} = \{(u, v) \in V(H)^2 \mid u \neq v\}$, and let $y \in V(W)$. Then there exists an $R[\mathcal{S}, (y, y)]$-gadget.*

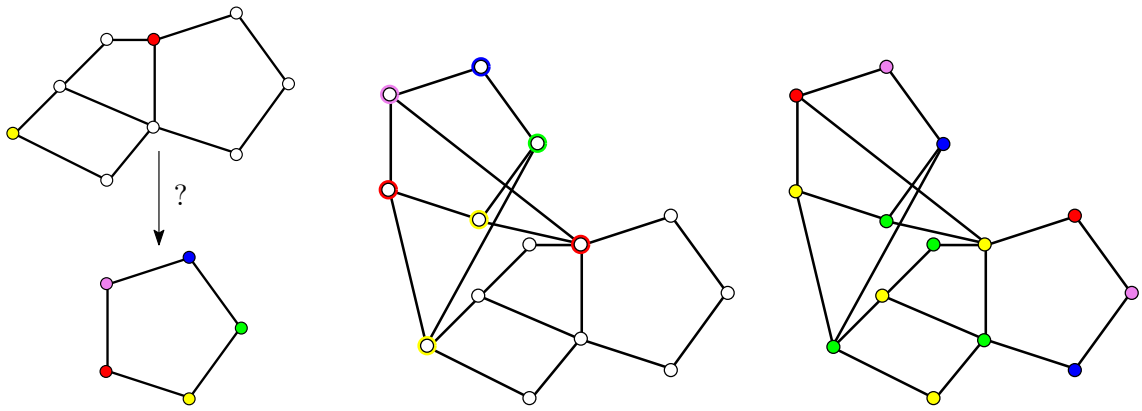


Figure 3.4: An instance (G, φ) of ExtHom(H) (left). The graph G' (middle) where the colors of vertices of V' and $\widehat{H}$ illustrate which vertices are forced to be mapped to the same vertex of H. A homomorphism f from G to H (right) that does not extend φ. However, it is straightforward to verify that there exists an automorphism of H that can be composed with f to obtain an extension of φ.

For the last result of this section, we first define a generalization of the Hom(H) problem, in which an instance can have some vertices already mapped to the vertices of H. For a fixed H, ExtHom(H) takes as an instance a pair (G, φ), where G is a graph and $\varphi : V' \to V(H)$ is a mapping from some $V' \subseteq V(G)$. We ask whether there exists a homomorphism that is an extension of φ to $V(G)$. Note that ExtHom(H) is a special case of LHom(H), in which every list is either equal to $V(H)$ or contains one element.

We observe that the Hom(H) problem is precisely the ExtHom(H) problem with $V' = \emptyset$. Moreover, note that for $t \geqslant 1$ and a set $\mathcal{S} \subseteq V(H)^t$, while an $\mathcal{S}$-gadget is, in particular, an instance of the Hom(H) problem, a construction of $\mathcal{S}$ is an instance of ExtHom(H).

It turns out that if H is a core, we can reduce ExtHom(H) to Hom(H) as follows.

Theorem 3.2.9. *Let H be a fixed core. Given an instance (G', φ) of ExtHom(H), we can construct an equivalent instance G of Hom(H) such that G' is an induced subgraph of G, and $|V(G)| = |V(G')| + |V(H)|$.*

Proof. Let $V' \subseteq V(G')$ be the domain of φ. We construct G as follows: first, we take G' and a copy $\widehat{H}$ of H. For brevity, we slightly abuse the notation and assume that H and $\widehat{H}$ have the same set of vertices; formally in such a case we should use an isomorphism between H and $\widehat{H}$. Next, we add an edge vu for every $v \in V'$ and $u \in V(\widehat{H})$ if $\varphi(v)u \in E(H)$. That concludes the construction, see Figure 3.4.

We first prove that if there exists an extension $h : G' \to H$ of φ, then h can be further

extended to G, by setting $h(v) = v$ for every $v \in V(\widehat{H})$. Indeed, to see that such h is a homomorphism from G to H consider $uv \in E(G)$. If $u, v \in V(G')$, then $h(u)h(v) \in E(H)$, by the assumption that h is a homomorphism. If $u, v \in V(\widehat{H})$, then $uv \in E(H)$, thus, $h(u)h(v) = uv \in E(H)$. Finally, assume that $v \in V(G'), u \in V(\widehat{H})$. Note that, by the definition of G, this can happen only if $v \in V'$ and u is adjacent to $\varphi(v)$ in H. Hence, $h(u)h(v) = u\varphi(v) \in E(H)$.

For the reverse direction, assume that there exists a homomorphism $f : G \to H$. We show that there exists an extension $h : G' \to H$ of φ. Let $\sigma : \widehat{H} \to H$ be the restriction of f to $\widehat{H}$. Since H is a core, σ is an isomorphism. We claim that the function $g \equiv \sigma^{-1} \circ f : V(G) \to V(H)$ is an extension of φ to G (so in particular, is an extension of φ to G'). Observe that by the definition of g, if $v \in V(\widehat{H})$, then $g(v) = v$.

Clearly, g is a composition of homomorphisms, so also a homomorphism. Therefore, it remains to show that for every $v \in V'$ we have $\varphi(v) = g(v)$. Consider a vertex $v \in V'$. Since H is a core, by Observation 3.1.1, H is ramified. Thus, as $N_G(\varphi(v)) \cap V(\widehat{H}) \subseteq N_G(v)$, we have that $f(v) = f(\varphi(v))$. It follows that

$$g(v) = \sigma^{-1}(f(v)) = \sigma^{-1}(f(\varphi(v))) = g(\varphi(v)).$$

However, recall that for every $u \in V(\widehat{H})$ we have that $g(u) = u$, so since $\varphi(v) \in V(\widehat{H})$, in particular, $g(v) = \varphi(v)$.

Finally, the facts that G' is an induced subgraph of G, and that $|V(G)| = |V(G')| + |V(H)|$ follow from the construction. □

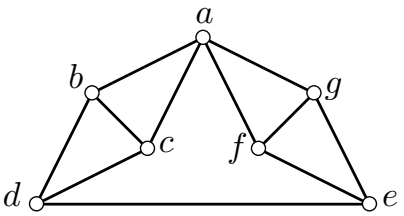
3.3 Signature sets

Let H be a graph. For a non-empty set $T \subseteq V(H)$ we say that $S_H(T)$ is the *signature set of* T if $S_H(T) = \bigcap_{t \in T} N(t)$ (see Figure 3.5). We omit the subscript and write $S(T)$ if it does not lead to confusion. We say that a non-empty set $Q \subseteq V(H)$ is a *signature set*, if there exists T such that $Q = S(T)$. We denote by $\mathcal{S}(H)$ the set of all signature sets of H.

We observe some basic properties of signature sets.

Observation 3.3.1. *Let $T \subseteq V(H)$ be non-empty.*

1. *If $a \in T$, $b \in S(T)$, then $ab \in E(H)$. In particular, for every $a \in T$ we have*



$S(\{b,c\}) = \{a,d\}$,
$S(\{b,c,g\}) = S(\{c,f\}) = \{a\}$,
$S(\{b,e\}) = S(\{c,e\}) = \{d\}$.

Figure 3.5: An example of graph H and signature sets of some of the subsets of $V(H)$. It can be verified that $s(H) = 18$.

$S(T) \subseteq N_H(a)$.

2. *If $T' \subseteq V(H)$ is non-empty, then $S(T \cup T') = S(T) \cap S(T')$. In particular, if $T \subseteq T'$, then $S(T) \supseteq S(T')$.*

Proof. The first statement follows directly from the definition of a signature set. For the second one, observe that $S(T \cup T') = \bigcap_{t \in T \cup T'} N(t) = \bigcap_{t \in T} N(t) \cap \bigcap_{t \in T'} N(t) = S(T) \cap S(T')$. □

We also note that the operation of taking a signature set is reversible on $\mathcal{S}(H)$.

Observation 3.3.2. *If T is a non-empty proper subset of $V(H)$, then $T \subseteq S(S(T))$. Moreover, if $T \in \mathcal{S}(H)$, then the reverse also holds, and $S(S(T)) = T$.*

Proof. By the definition of a signature set, for each two vertices $u, v \in V(H)$ such that $u \in T$ and $v \in S(T)$, we have $uv \in E(H)$, so $T \subseteq S(S(T))$. For the converse direction observe that if $T \in \mathcal{S}(H)$, there exists a non-empty subset A of $V(H)$ such that $T = S(A)$. Pick any $u \in S(S(T))$. Clearly, for every $v \in S(T)$ we have $uv \in E(H)$. Since $A \subseteq S(S(A)) = S(T)$, in particular, for every $v \in A$ we have $uv \in E(H)$. Hence by definition $u \in S(A) = T$. □

We observe that a signature set of a graph that is a direct product is precisely the Cartesian product of signature sets of its factors.

Observation 3.3.3. *Let $H = H_1 \times H_2$. Then $\mathcal{S}(H) = \mathcal{S}(H_1) \times \mathcal{S}(H_2)$.*

Proof. We prove that $\mathcal{S}(H)$ is of the form

$$\mathcal{S}(H) = \{S_{H_1}(T_1) \times S_{H_2}(T_2) \mid \emptyset \neq T_i \subseteq V(H_i) \text{ and } S_{H_i}(T_i) \neq \emptyset \text{ for } i = 1, 2\}.$$

Let T_1 and T_2 be some non-empty subsets of, respectively, $V(H_1)$ and $V(H_2)$. Clearly,

$$S_{H_1}(T_1) \times S_{H_2}(T_2) = \left(\bigcap_{t\in T_1} N_{H_1}(t)\right) \times \left(\bigcap_{t'\in T_2} N_{H_2}(t')\right) = \bigcap_{(t,t')\in T_1\times T_2} N_{H_1\times H_2}(t,t') \\ = S_H(T_1 \times T_2). \tag{3.1}$$

Therefore, if $S_{H_1}(T_1)$ and $S_{H_2}(T_2)$ are non-empty, we get that $S_{H_1}(T_1) \times S_{H_2}(T_2) \in \mathcal{S}(H)$.

To see that $\mathcal{S}(H) \subseteq \mathcal{S}(H_1) \times \mathcal{S}(H_2)$, we show that for every non-empty $T \subseteq V(H)$ the set $S_H(T) \in \mathcal{S}(H)$ is of the form $S_{H_1}(T_1) \times S_{H_2}(T_2)$ for some $T_1 \subseteq V(H_1), T_2 \subseteq V(H_2)$. Define T_1 and T_2 to be minimal sets such that $T \subseteq T_1 \times T_2$. Hence, by (3.1), $S_{H_1}(T_1) \times S_{H_2}(T_2) = S_H(T_1 \times T_2) \subseteq S_H(T)$. On the other hand, for every $(s, s') \in S_H(T)$ we have $s \in \bigcap_{t\in T_1} N_{H_1}(t)$ and $s' \in \bigcap_{t'\in T_2} N_{H_2}(t')$, so the equality follows. □

Let the *signature number* of H, denoted $s(H)$, be defined as $|\mathcal{S}(H)|$. Observe that if H is non-trivial, so in particular irreflexive, for every non-empty $T \subseteq V(H)$ we have that $S(T) \cap T = \emptyset$. From that it is easy to see that $V(H)$ never belongs to $\mathcal{S}(H)$. Since, by definition, $\emptyset \notin \mathcal{S}(H)$, we get the following bounds for $s(H)$.

Observation 3.3.4. *Let H be an irreflexive graph. Then $s(H) \leqslant 2^{|V(H)|} - 2$.* □

Notice that since $2^{|V(H)|} - 2$ is the number of all proper non-empty subsets of $V(H)$, the equality in Observation 3.3.4 holds if and only if H is a clique. We note that if H is a core graph, we can also bound the minimum cardinality of $\mathcal{S}(H)$.

Observation 3.3.5. *Let H be a non-trivial core. Then $s(H) \geqslant 2|V(H)|$, and if H does not contain C_4 as a subgraph, then the equality holds.*

Proof. Recall that a core is always ramified and thus $N(v) \not\subseteq N(u)$ for every distinct $u, v \in V(H)$. It is straightforward to verify that a non-trivial core does not contain vertices of degree smaller than two. Thus, for each $v \in V(H)$ we have $|N(v)| \geqslant 2$, and clearly $N(v) \in \mathcal{S}(H)$, as $S(\{v\}) = N(v)$. Hence different vertices give rise to $|V(H)|$ signature sets, denote the family of these sets by $\mathcal{S}_1$. On the other hand, we have that $S(N(v)) = \{v\}$, as if $u \in S(N(v))$, then $N(v) \subseteq N(u)$ and thus $u = v$. Therefore, different neighborhoods give rise to another $|V(H)|$ signature sets, denote them by $\mathcal{S}_2$. Observe that $\mathcal{S}_1 \cap \mathcal{S}_2 = \emptyset$, as elements of $\mathcal{S}_2$ are singletons, while sets in $\mathcal{S}_1$ contains at least two elements.

Now, if we additionally assume that H does not contain C_4 as a subgraph, we claim that there is no element $T \in \mathcal{S}(H)$ that is not of the form $\{v\}$ or $N(v)$ for some $v \in V(H)$. Otherwise, there exist $a, b \in T$ and $c, d \in S(T)$, all distinct, and by Observation 3.3.1 1., graph $H[\{a, c, b, d\}]$ contains a C_4. □

We observe that the signature number can never increase by taking an induced subgraph of a graph.

Observation 3.3.6. *Let H' be a graph and let H be an induced subgraph of H'. Then $s(H) \leqslant s(H')$.*

Proof. Given a connected graph Q without loops, consider an equivalence relation $\sim_Q$ on the set of nonempty subsets of $V(Q)$ defined as follows: $V_1 \sim_Q V_2$ if and only if V_1 and V_2 have the same signature sets in Q. Observe that $s(Q)$ is equal to the number of equivalence classes of $\sim_Q$ minus one (as there are subsets V such that $S(V) = \emptyset$). Hence to prove the claim, it suffices to show that whenever two subsets of $V(H)$ belong to different equivalence classes of $\sim_H$, they also belong to different equivalence classes of $\sim_{H'}$. For this, consider any two non-empty subsets V_1 and V_2 of $V(H)$ such that $V_1 \nsim_H V_2$.

Since $S_H(V_1) \neq S_H(V_2)$, by symmetry of V_1 and V_2, we can assume that there exists $v \in S_H(V_1) \setminus S_H(V_2)$. Then for every $t \in V_1$ we have $vt \in E(H) \subseteq E(H')$, i.e., $v \in S_{H'}(V_1)$. On the other hand, there exists $t_0 \in V_2$ such that $vt_0 \notin E(H)$, as otherwise v would belong to $S_H(V_2)$. As H is induced subgraph of H', it means that $vt_0 \notin E(H')$. Hence, v does not belong to $S_{H'}(V_2)$ and thus $V_1 \nsim_{H'} V_2$. □

We note that the signature number of H is the sum of signature numbers over its connected components.

Observation 3.3.7. *Let H be a graph, and let $H_1, \ldots, H_m$ be the connected components of H. Then*

$$\mathcal{S}(H) = \mathcal{S}(H_1) \cup \ldots \cup \mathcal{S}(H_m),$$

and for each element S of $\mathcal{S}(H)$ there exists an unique $i \in [m]$ such that $S \in \mathcal{S}(H_i)$.

Proof. Clearly, if $S \in \mathcal{S}(H_i)$ for some $i \in [m]$, then $S \in \mathcal{S}(H)$. On the other hand, observe that for every set T that is not contained in $V(H_i)$ for some $i \in [m]$, we have that $S(T) = \emptyset$, hence $S(T)$ is not included in $\mathcal{S}(H)$. □

If $Q \in \mathcal{S}(H)$, we call T such that $S(T) = Q$ a *witness* of Q. Clearly, we can have distinct T_1, T_2 such that $S(T_1) = S(T_2)$, however, notice that in such a case there exists $T = T_1 \cup T_2$ such that $S(T) = S(T_1) = S(T_2)$. Hence, there exists a unique maximal (with respect to inclusion) witness of Q, and we denote it by $M(Q)$. In fact, it is not difficult to see that $M(Q) = \{v \in V(H) \mid Q \subseteq N_H(v)\}$.

As the last thing of this section, we observe that each signature set is constructible. Indeed, let $S \in \mathcal{S}(H)$, and let $T \subseteq V(H)$ be a witness of S. Let F be the star with the central vertex c and the set X of $|T|$ leaves, and let $\varphi : X \to T$ be any bijection. Then it is straightforward to verify that (F, φ, X) is a construction of S.

Chapter 4

The homomorphism problem parameterized by clique-width

In this chapter we prove Theorem 1.3.2 and discuss its generalizations.

Theorem 1.3.2. *Let H be a fixed non-trivial projective core, and let* $\mathrm{cw}(G)$ *be the clique-width of an n-vertex instance graph G.*

(a) Assuming a cw*-expression of G of optimal width and size polynomial in n is given, the Hom(H) problem can be solved in time $s(H)^{\mathrm{cw}(G)} \cdot n^{\mathcal{O}(1)}$.*

(b) There is no algorithm solving the Hom(H) problem in time $(s(H) - \varepsilon)^{\mathrm{cw}(G)} \cdot n^{\mathcal{O}(1)}$, for any $\varepsilon > 0$, unless the SETH fails.

The algorithmic and hardness part of the problem are the subject of, respectively, Section 4.1 and Section 4.2.

4.1 The algorithm

As a first step, we present an algorithm that plays a crucial role for the upper bound, i.e., it implies Theorem 1.3.2 (a).

Theorem 4.1.1. *Let H be a fixed graph, and let G be an n-vertex graph. Assuming a t-expression of G of and size polynomial in n is given, the Hom(H) problem can be solved in time $s(H)^t \cdot n^{\mathcal{O}(1)}$.*

Proof of Theorem 4.1.1. Let σ be a k-expression defining G. First, we observe that we can assume that:

(a) for each node $\tau = \rho_{i\to j}(\tau')$ of σ, there is at least one vertex with label i in $G_{\tau'}$,

(b) for each node $\tau = \eta_{i,j}(\tau')$ of σ, there is at least one vertex with label i and at least one vertex with label j in $G_{\tau'}$.

Indeed, if this is not the case, we can delete the node τ from σ and connect the unique child τ' of τ to the parent of τ, if one exists. This operation produces an equivalent t-expression that defines G. Since the number of nodes in σ is polynomial in n, this step can be performed in time polynomial in n.

If G is disconnected, we process every connected component of G independently. If $|V(G)| = 1$, we return that G is a yes-instance, as $K_1 \to H$ for every graph H. From now on, let us assume that G is connected and $|V(G)| > 1$. We describe a dynamic program that proceeds in a leaf-to-root fashion along σ.

For a node τ of σ, we say that i is a *live label* in τ if there is an edge of G which is incident to V_τ^i and does not appear in G_τ. Observe that in this case, by the construction of the t-expression, for every $v \in V_\tau^i$ there exists such an edge incident to v. Denote the set of live labels in τ by L_τ. Since G is connected, $L_\tau \neq \emptyset$ for any non-root node τ of σ.

Our algorithm, for each node τ of σ, computes a set P_τ consisting of functions $p\colon L_\tau \to \mathcal{S}(H)$ where $p \in P_\tau$ if and only if

$$\text{there exists } h_p : G_\tau \to H \text{ s.t. for every } i \in L_\tau \text{ we have } p(i) \subseteq S(h_p(V_\tau^i)). \tag{4.1}$$

In other words, we add p to P_τ if and only if there exists a homomorphism $h_p : G_\tau \to H$ such that the signature set of the image of vertices labeled i contains vertices from $p(i)$. Intuitively, we use $p(i)$ to preemptively store the images of the neighbors of V_τ^i in the final graph G—that is why $p(i)$ is not necessarily the exact signature, but could be any signature that occurs as a subset. We say that $p \in P_\tau$ *describes* the homomorphism h_p in τ or, equivalently, that h_p *witnesses* p in τ.

First, we describe how to compute the sets P_τ, and the intuition standing behind these computations. Then we argue that by computing P_τ we can compute the solution to $\textsc{Hom}(H)$, and show correctness of our computations, i.e., that (4.1) is satisfied. Last, we discuss the complexity of computing sets P_τ.

We distinguish the cases based on the root operation of τ.

Case: $\tau = i(v)$ for some $i \in [t]$. In this case $L_\tau = \{i\}$, since we assume that G is connected and $|V(G)| > 1$. We add to P_τ all functions $p : \{i\} \to \mathcal{S}(H)$.

Case: $\tau = \rho_{i\to j}(\tau')$ for the unique child τ' of τ. If $i \notin L_{\tau'}$ then clearly $j \notin L_\tau$ and we set $P_\tau = P_{\tau'}$. Otherwise, if $i \in L_{\tau'}$ and $j \notin L_{\tau'}$, every $p \in P_\tau$ arises from some $p' \in P_{\tau'}$ by replacing i by j in the domain:

$$P_\tau = \{(p' \setminus (i, S)) \cup (j, S) \mid S = p'(i) \wedge \ p' \in P_{\tau'}\}.$$

Finally, if $i \in L_{\tau'}, j \in L_{\tau'}$, then $L_\tau = L_{\tau'} \setminus \{i\}$ and $P_\tau = \{p'|_{L_\tau} \mid p' \in P_{\tau'} \wedge p'(i) = p'(j)\}$.

Intuitively, we simply rename label i to j, however, since after this step the vertices labeled i and j in τ' are indistinguishable, we first ensure that the image of their potential neighbors is the same.

Case: $\tau = \tau_1 \oplus \tau_2$ for children τ_1 and τ_2 of τ. In this case $L_\tau = L_{\tau_1} \cup L_{\tau_2}$ as $E(G_\tau) = E(G_{\tau_1}) \cup E(G_{\tau_2})$ and

$$P_\tau = \{p = p_1 \cup p_2 \mid p_1 \in P_{\tau_1} \wedge p_2 \in P_{\tau_2} \wedge (\forall \ell \in L_{\tau_1} \cap L_{\tau_2} \colon p_1(\ell) = p_2(\ell))\}$$

Intuitively, we will construct a homomorphism on the disjoint union of two subgraphs by "gluing together" the homomorphisms on the subgraphs. If the subgraphs share any live labels, after this step they will all be treated equally. For this reason we require the images of the neighbors of such labels to be the same in both subgraphs.

Case: $\tau = \eta_{i,j}(\tau')$. In this case $L_\tau = L_{\tau'} \setminus I$ where $I \subseteq \{i, j\}$ is the set of live labels in τ' that are no longer live labels in τ. We set:

$$P_\tau = \{p \mid \exists p' \in P_{\tau'} \colon p'(i) \supseteq S\Big(p'(j)\Big) \ \wedge p|_{L_\tau \setminus \{i,j\}} = p'|_{L_\tau \setminus \{i,j\}} \\ \wedge \ p(i) \subseteq p'(i) \ \wedge \ p(j) \subseteq p'(j)\}.$$

where for $\ell \in I$, we interpret $p(\ell)$ as an empty set.

Intuitively, we can add the edges between vertices in two live labels if and only if we first ensure there are edges between their images. This concludes the description of how to compute P_τ. Before we show that it respects (4.1), we argue that computing all P_τ indeed solves the problem.

For that, note that it is enough to prove that a mapping $p : L_\sigma \to \mathcal{S}(H)$ belongs to P_σ if and only if there exists $h : G \to H$ such that for every $i \in L_\sigma$ we have $p(i) \subseteq S\Big(h\Big(V^i_\sigma\Big)\Big)$. As $L_\sigma = \emptyset$, this is equivalent to saying that the empty mapping p belongs to P_σ if and only if there exists $h : G \to H$. Thus, we can read the answer from P_σ: if contains the empty mapping, i.e., $P_\sigma = \{\emptyset\}$, then $G \to H$, otherwise $P_\sigma = \emptyset$ and $G \not\to H$.

Thus it remains to show that $p : L_\sigma \to \mathcal{S}(H) \in P_\sigma$ if and only if (4.1) is satisfied. The proof of each implication is provided in a form of a separate claim.

Claim 4.1.1.1. *If $p \in P_\tau$, then (4.1) is satisfied, i.e., there exists $h_p : G_\tau \to H$ such that for every $i \in L_\tau$ we have $p(i) \subseteq S\Big(h_p\Big(V^i_\tau\Big)\Big)$.*

Proof of Claim. First, we consider the base case of our algorithm.

Case: $\tau = i(v)$ for some $i \in [t]$. Since $p(i) \in \mathcal{S}(H)$, by Observation 3.3.1 2., there exists $u \in V(H)$ such that $p(i) \subseteq N_H(u)$. Then p describes the homomorphism $h_p : G_\tau \to H$ defined by $h_p(v) = u$.

Now let us assume that τ is a non-leaf node, and that the claim holds for all its children. There are three possible types of τ.

Case: $\tau = \rho_{i\to j}(\tau')$ for the unique child τ' of τ. Let $p' \in P_{\tau'}$ be a function such that p arises from p' in the construction of P_τ. Consider a witness $h : G_{\tau'} \to H$ of $p' \in P'_\tau$. For every $\ell \in L_\tau \setminus \{j\}$ we have $p(\ell) = p'(\ell) \subseteq S\Big(h\Big(V^\ell_{\tau'}\Big)\Big) = S\Big(h\Big(V^\ell_\tau\Big)\Big)$, where the first equality follows from the definition of p, the containment is by the inductive assumption, and the last equality is because $V^\ell_\tau = V^\ell_{\tau'}$. If now $i \notin L_{\tau'}$, then $j \notin L_\tau$, and thus p describes h in τ.

Hence assume that $i \in L_{\tau'}$ and note that in a such case $j \in L_\tau$. Now $p(j) = p'(i) \subseteq S\Big(h\Big(V^i_{\tau'}\Big)\Big)$, by definition of p and the inductive assumption, and if additionally $j \in L_{\tau'}$, then $p(j) = p'(j) \subseteq S\Big(h\Big(V^j_{\tau'}\Big)\Big)$. By Observation 3.3.1, if $j \in L_{\tau'}$, we have

$$p(j) \subseteq S\left(h\left(V^i_{\tau'}\right)\right) \cap S\Big(h\Big(V^j_{\tau'}\Big)\Big) = S\Big(h\left(V^i_{\tau'}\right) \cup h\Big(V^j_{\tau'}\Big)\Big) = S\Big(h\Big(V^i_{\tau'} \cup V^j_{\tau'}\Big)\Big) = S\Big(h\Big(V^j_\tau\Big)\Big).$$

Hence p describes h in τ. On the other hand, if $j \notin L_{\tau'}$, then $V^i_{\tau'} = V^j_\tau$, and we have $p(j) \subseteq S\Big(h\Big(V^i_{\tau'}\Big)\Big) = S\Big(h\Big(V^j_\tau\Big)\Big)$. Again, p describes h in τ.

Case: $\tau = \tau_1 \oplus \tau_2$ for children τ_1 and τ_2 of τ. Let $p_1 \in P_{\tau_1}$, $p_2 \in P_{\tau_2}$ be functions such that $p = p_1 \cup p_2$. Let h_i be a witness of p_i in τ_i, for $i = 1, 2$. As the domains of h_1

and h_2 are disjoint, we can define $h = h_1 \cup h_2$. Since there are no edges between $V(G_{\tau_1})$ and $V(G_{\tau_2})$ in G_τ, h is a homomorphism from G_τ to H.

Observe that for all $\ell \in L_{\tau_1} \setminus L_{\tau_2}$, we have $p(\ell) = p_1(\ell) \subseteq S\Big(h_1\Big(V_\tau^\ell\Big)\Big) = S\Big(h\Big(V_\tau^\ell\Big)\Big)$, by inductive assumption, similarly for $\ell \in L_{\tau_2} \setminus L_{\tau_2}$. For $\ell \in L_{\tau_1} \cap L_{\tau_2}$, for every $i \in \{1, 2\}$ by inductive assumption we have $p(\ell) = p_i(\ell) \subseteq S\Big(h_i\Big(V_{\tau_i}^\ell\Big)\Big)$. Therefore, again by Observation 3.3.1:

$$p(\ell) \subseteq S\Big(h_1\Big(V_{\tau_1}^\ell\Big)\Big) \cap S\Big(h_2\Big(V_{\tau_2}^\ell\Big)\Big) = S\Big(h_1\Big(V_{\tau_1}^\ell\Big) \cup h_2\Big(V_{\tau_2}^\ell\Big)\Big) = S\Big(h\Big(V_\tau^\ell\Big)\Big).$$

Hence h is a witness of p in τ.

Case: $\tau = \eta_{i,j}(\tau')$ for the unique child τ' of τ. Let $p' \in P_{\tau'}$ be a function such that p arises from p' in the construction of P_τ. Consider a witness $h\colon G_{\tau'} \to H$ of $p' \in P'_\tau$. First, we note that h is a homomorphism from G_τ to H. For that, it is enough to show that h preserves edges between V_τ^i and V_τ^j. Recall that $p'(i) \supseteq S(p'(j))$ by the definition of P_τ, and the inductive assumption gives us that $p'(i) \subseteq S\Big(h\Big(V_{\tau'}^i\Big)\Big)$ and $p'(j) \subseteq S\Big(h\Big(V_{\tau'}^j\Big)\Big)$. Along with Observations 3.3.1 and 3.3.2 this results in

$$S\Big(h\Big(V_{\tau'}^i\Big)\Big) \supseteq p'(i) \supseteq S\Big(p'(j)\Big) \supseteq S\Big(S\Big(h\Big(V_{\tau'}^j\Big)\Big)\Big) \supseteq h\Big(V_{\tau'}^j\Big).$$

Hence $h\Big(V_\tau^i\Big) \times h\Big(V_\tau^j\Big) \subseteq E(H)$, so h is a homomorphism from G_τ to H. By construction, for every $\ell \in L_\tau$ it holds that $p(\ell) \subseteq p'(\ell) \subseteq S\Big(h\Big(V_\tau^\ell\Big)\Big)$. Thus h witnesses p in τ, and that concludes the proof of claim. ⌟

Now we show the converse implication.

Claim 4.1.1.2. *Let $p : L_\tau \to \mathcal{S}(H)$. Assume that (4.1) is satisfied, i.e., there exists $h : G_\tau \to H$ such that for every $i \in L_\tau$ we have $p(i) \subseteq S\Big(h\Big(V_\tau^i\Big)\Big)$. Then $p \in P_\tau$.*

Proof of Claim. Again, we first consider the base case of our algorithm.

Case: $\tau = i(v)$ for some $i \in [t]$. In this case clearly $p \in P_\tau$, as $L_\tau = \{i\}$ and P_τ contains all possible functions from $\{i\}$ to $\mathcal{S}(H)$.

Now let us assume that τ is a non-leaf node, and that the claim holds for all its children. There are three possible types of τ.

Case: $\tau = \rho_{i \to j}(\tau')$ for the unique child τ' of τ. First, we observe that if $i \notin L_{\tau'}$, then clearly $i \notin L_\tau$. Moreover, in this case, $j \notin L_\tau$. Indeed, otherwise, by the definition

of a live label, this means that G contains edges incident to V_τ^j that do not appear in G_τ. By construction of a clique-width expression, such an edge exists for every $v \in V_\tau^j$, in particular for every $v \in V_{\tau'}^i \subseteq V_\tau^j$, contradicting the fact that $i \notin L_{\tau'}$. Hence $j \notin L_\tau$, in particular $j \notin L_{\tau'}$. Therefore, both i and j are not live neither in τ nor in τ', so $L_{\tau'} = L_\tau$. By the inductive assumption, $p \in P'_\tau$. Recall that we set $P_\tau = P_{\tau'}$ in this case.

If now $i \in L_{\tau'}$, then $j \in L_\tau$. Define $p' : L_{\tau'} \to \mathcal{S}(H)$ to be such that for every $\ell \in L_{\tau'} \setminus \{i\} \subseteq L_\tau$ it holds that $p'(\ell) = p(\ell)$, and $p'(i) = p(j)$. By our assumption on h, for every $\ell \in L_{\tau'} \setminus \{i, j\}$ we have that $p'(\ell) = p(\ell) \subseteq S\Big(h\Big(V_\tau^\ell\Big)\Big) = S\Big(h\Big(V_{\tau'}^\ell\Big)\Big)$. Moreover,

$$p'(i) = p(j) \subseteq S\Big(h\Big(V_\tau^j\Big)\Big) = S\Big(h\Big(V_{\tau'}^j \cup V_{\tau'}^i\Big)\Big) = S\Big(h\Big(V_{\tau'}^j\Big) \cup h\Big(V_{\tau'}^i\Big)\Big) = S\Big(h(V_{\tau'}^j)\Big) \cap S\Big(h\Big(V_{\tau'}^i\Big)\Big),$$

where the last equality is by Observation 3.3.1. This in particular means that if $j \in L_{\tau'}$, by the definition of p' we have that $p'(j) = p(j) = p'(i) \subseteq S\Big(h\Big(V_{\tau'}^j\Big)\Big) \cap S\Big(h\Big(V_{\tau'}^i\Big)\Big)$. Hence, $p'(i) \subseteq S\Big(h\Big(V_{\tau'}^i\Big)\Big)$ and, if $j \in L_{\tau'}$, then $p'(j) \subseteq S\Big(h\Big(V_{\tau'}^j\Big)\Big)$. Therefore, by the inductive assumption we have that $p' \in P_{\tau'}$.

Now it is straightforward to verify that if $j \notin L_{\tau'}$ then $p = (p' \setminus (i, p'(i))) \cup (j, p'(i)) \in P_\tau$. Otherwise, i.e., if $j \in L_{\tau'}$, we conclude that $p = p'|_{L_\tau} \in P_\tau$.

Case: $\tau = \tau_1 \oplus \tau_2$ for children τ_1 and τ_2 of τ. For every $i \in \{1, 2\}$ let $p_i : L_{\tau_i} \to \mathcal{S}(H)$ be the restriction of p to L_{τ_i}, i.e. $p_i \equiv p|_{L_{\tau_i}}$. Clearly, for each $\ell \in L_{\tau_1} \cap L_{\tau_2}$ we have $p_1(\ell) = p_2(\ell)$. Moreover, for every $i \in \{1, 2\}$ let $h_i : G_{\tau_i} \to H$ be the restriction of h to $V(G_{\tau_i})$.

We observe that for every $i \in \{1, 2\}$ and for every $\ell \in L_{\tau_i}$ we have $V_{\tau_i}^\ell \subseteq V_\tau^\ell$, so $S\Big(h\Big(V_\tau^\ell\Big)\Big) \subseteq S\Big(h\Big(V_{\tau_i}^\ell\Big)\Big)$ by Observation 3.3.1. In particular,

$$p_i(\ell) = p(\ell) \subseteq S\Big(h\Big(V_\tau^\ell\Big)\Big) \subseteq S\Big(h\Big(V_{\tau_i}^\ell\Big)\Big) = S\Big(h_i\Big(V_{\tau_i}^\ell\Big)\Big).$$

Hence, by the inductive assumption, for every $i \in \{1, 2\}$ we have $p_i \in P_{\tau_i}$. This, combined with the definition of P_τ, implies that $p \in P_\tau$.

Case: $\tau = \eta_{i,j}(\tau')$ for the unique child τ' of τ. Recall that the first item of our preprocessing stage guarantees that $i, j \in L_{\tau'}$. We define $p' : L_{\tau'} \to \mathcal{S}(H)$ as follows.

$$p'(\ell) = \begin{cases} S\Big(h\Big(V_{\tau'}^\ell\Big)\Big) & \text{if } \ell \in \{i, j\}, \\ p(\ell) & \text{otherwise.} \end{cases}$$

By the assumption on h, for every $\ell \in L_{\tau'}$ we have that $p'(\ell) \subseteq S\Big(h\Big(V^{\ell}_{\tau'}\Big)\Big)$. By the inductive assumption, $p' \in P_{\tau'}$.

We observe that since h is a homomorphism, for every $a \in h\Big(V^i_{\tau'}\Big) = h\Big(V^i_{\tau}\Big)$ and every $b \in h\Big(V^j_{\tau'}\Big) = h\Big(V^j_{\tau}\Big)$ we have that $ab \in E(H)$. Therefore, $h\Big(V^j_{\tau'}\Big) \subseteq S\Big(h\Big(V^i_{\tau'}\Big)\Big)$, so by Observation 3.3.1 $S\Big(S\Big(h\Big(V^i_{\tau'}\Big)\Big)\Big) \subseteq S\Big(h\Big(V^j_{\tau'}\Big)\Big)$ and furthermore $S\Big(S\Big(h\Big(V^j_{\tau'}\Big)\Big)\Big) \subseteq S\Big(S\Big(S\Big(h\Big(V^i_{\tau'}\Big)\Big)\Big)\Big)$. By Observation 3.3.2, as $S\Big(h\Big(V^i_{\tau'}\Big)\Big) \in \mathcal{S}(H)$, we have that

$$S\Big(S\Big(S\Big(h\Big(V^i_{\tau'}\Big)\Big)\Big) = S\Big(h\Big(V^i_{\tau'}\Big)\Big).$$

Thus, $S\Big(S\Big(h\Big(V^j_{\tau'}\Big)\Big)\Big) \subseteq S\Big(h(V^i_{\tau'})\Big)$, or equivalently $S\Big(p'(j)\Big) \subseteq p'(i)$. For $\ell \in \{i, j\}$ we have $p(\ell) \subseteq S\Big(h\Big(V^{\ell}_{\tau'}\Big)\Big) = p'(\ell)$, and for every $\ell \in L_\tau \setminus \{i, j\}$ we have $p(\ell) = p'(\ell)$. Therefore, $p \in P_\tau$ by the definition of P_τ. This concludes the proof of claim. ⌟

It remains to discuss the complexity of our algorithm. We note that $|P_\tau| \leqslant s(H)^t$ for each node τ of σ. Thus, in each node, the computation requires time $c \cdot t \cdot s(H)^2 \cdot s(H)^t$ for some constant $c > 0$. As the preprocessing stage takes time polynomial in $|V(G)|$, i.e., bounded by $c' \cdot |V(G)|^d$ for some constants $c', d > 0$, this yields the complexity of

$$c'' \cdot |V(G)|^d \cdot t \cdot s(H)^2 \cdot s(H)^t = c'' \cdot |V(G)|^{d'} \cdot s(H)^t = \mathcal{O}^*\Big(s(H)^t\Big),$$

where $c'' = \max\{c, c'\} \cdot s(H)^2$ and $d' = d + 1$. □

4.1.1 Consequences of Theorem 4.1.1

While Theorem 4.1.1 serves as an upper bound that will match our target SETH-based lower bounds for $\textsc{Hom}(H)$ for, intuitively, the "most difficult" choices of H, in many cases one can in fact improve the algorithm's runtime by exploiting well-known properties graph homomorphisms.

As a simple example showcasing this, consider the *6-wheel* graph W_6 (see Figure 4.1). Since W_6 is 3-colorable, it holds that $W_6 \to K_3$, and since K_3 is a core and an induced subgraph of W_6, it is the core of W_6. We recall that if H is the core of H', then for every graph G it holds that $G \to H$ if and only if $G \to H'$. Hence, if G is an instance of $\textsc{Hom}(W_6)$ given with a t-expression of size polynomial in $|V(G)|$ is given, we can solve it by using Theorem 4.1.1 for $H = K_3$ to decide whether $G \to W_6$ in total running time $\mathcal{O}^*(s(K_3)^t)$. As $s(K_3) < s(W_6)$ (as shown in Figure 4.1), this yields a better running time

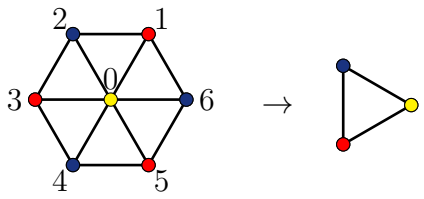


Figure 4.1: The graphs W_6 (left) and K_3 (right). Colors on the vertices of W_6 indicate the homomorphism $h : W_6 \to K_3$. It can be verified that $s(W_6) = 19$, but, since K_3 is a clique, we have $s(K_3) = 6$. Thus, $s(K_3) < s(W_6)$.

bound than the direct use of Theorem 4.1.1. While this example shows that the signature number can decrease by taking an induced subgraph, recall from Observation 3.3.6 that it can never increase.

Thus, at this point, we may ask whether the procedure of computing the unique core H of the fixed connected target graph H', and then applying Theorem 4.1.1 for each of the connected components of H could yield a tight upper bound for $\textsc{Hom}(H')$. Unfortunately, the situation is more complicated than that. Recall that by Theorem 3.2.1, each non-trivial, connected target graph H has a unique prime factorization $(H_1, \ldots, H_m)$. If we treat H as a fixed graph, it can be found in constant time. Now, let G be an instance graph of $\textsc{Hom}(H)$ and assume that the clique-width expression σ of G of width t is given. If we run Theorem 4.1.1 separately for every factor of H, by Observation 3.2.2 (5), the $\textsc{Hom}(H)$ problem can be solved in time $\max_{i\in[m]} s(H_i)^{\mathrm{cw}(G)} \cdot n^{\mathcal{O}(1)}$.

On the other hand, recall that by Observation 3.3.3, the notion of signature sets we introduced in the previous section behaves multiplicatively with respect to taking direct product of graphs. In particular, it follows from Observation 3.3.3 that if H is a graph with factorization $H_1 \times \ldots \times H_m$, then $s(H) = s(H_1) \cdot \ldots \cdot s(H_m)$. Recall that if K_1 and K_2 never appear as a Therefore, if H is non-trivial and not prime, there exist at least two factors H_i, H_j such that $s(H_i) > 1$, $s(H_j) > 1$ (recall Observation 3.2.3 and Observation 3.3.5), applying Theorem 4.1.1 separately for each factor of H yields a better running time than applying it for H. Formally, we obtain the following.

Corollary 4.1.2. *Let H be a fixed connected graph and let H' be its unique core. Let $(H_1, \ldots, H_m)$ be a factorization of H'. Let G be an instance graph of $\textsc{Hom}(H)$. Assuming that the clique-width expression of G of width t is given, the $\textsc{Hom}(H)$ problem can be solved in time $\max_{i\in[m]} s(H_i)^t \cdot n^{\mathcal{O}(1)}$.*

4.2 Lower bounds

This subsection is devoted to the proof of the hardness counterpart of Theorem 1.3.2 (b) and a discussion of its possible generalizations. To prove it, we again focus on a slightly different (in fact, more general) theorem. Let H be a non-trivial core, and let $(H_1, \dots, H_m)$ be a prime factorization of H such that $s(H_1) = \max_{j \in [m]} s(H_j)$ (if this holds for more than one factor, we chose one of them arbitrarily). Let $W = H_2 \times \ldots \times H_m$ if $m \geqslant 2$, and $W = K_1^\circ$ otherwise. Define the *canonical factorization* γ of H to be (H_1, W).

We claim that the following theorem implies Theorem 1.3.2 (b).

Theorem 4.2.1. *Let H be a fixed connected non-trivial core, and let $\gamma = (H_1, W)$ be its canonical factorization. Assume that H is H_1-projective. There is no algorithm solving an instance G of ExtHom(H) in time $(s(H_1) - \varepsilon)^{\mathrm{cw}(G)} \cdot n^{\mathcal{O}(1)}$ for any $\varepsilon > 0$, unless the SETH fails.*

Clearly, ExtHom(H) problem is more general than Hom(H). However, Theorem 4.2.1 together with Theorem 3.2.9 actually imply Theorem 1.3.2 (b), as shown below.

Theorem 4.2.1 → Theorem 1.3.2 (b): Assume Theorem 1.3.2 (b) does not hold, i.e., there exists an algorithm A that solves every instance G of Hom(H) in time $(s(H_1) - \varepsilon)^{\mathrm{cw}(G)} \cdot n^{\mathcal{O}(1)}$.

Let (G', φ) be an instance of ExtHom(H). We use Theorem 3.2.9 to transform (G', φ) into an equivalent instance G of Hom(H), such that $|V(G)| = |V(G')| + |V(H)|$ and G' is an induced subgraph of G.

Note that $\mathrm{cw}(G) \leqslant \mathrm{cw}(G') + |V(H)|$. Indeed, that we can obtain G by adding exactly $|V(H)|$ vertices to G'. This means we can modify any $\mathrm{cw}(G)$-expression σ for G' to obtain a $\mathrm{cw}(G)$-expression of G as follows. Each added vertex is introduced with a designated label that is distinct from all labels used in σ. Then each node of σ that introduces a vertex of G' can be replaced by a tree that introduces the vertex and inserts all required edges incident to the vertices of $V(G) \setminus V(G')$.

Now we can use use A to decide whether $G \to H$ in time

$$((s(H_1) - \varepsilon)^{\mathrm{cw}(G)}\, n^{\mathcal{O}(1)} = (s(H_1) - \varepsilon)^{\mathrm{cw}(G')} \cdot (s(H_1) - \varepsilon)^{|V(H)|} n^{\mathcal{O}(1)}.$$

Since H is a fixed graph, $(s(H_1) - \varepsilon)^{|V(H)|}$ is a constant. As $n > 1$, we have $(s(H_1) -$

$\varepsilon)^{\mathrm{cw}(G)}n^{\mathcal{O}(1)} = (s(H_1) - \varepsilon)^{\mathrm{cw}(G')}n^{\mathcal{O}(1)}$. Since $G \to H$ if and only if (G', φ) is a yes-instance of ExtHom(H), we get a contradiction with Theorem 4.2.1. □

Our reduction generalizes the construction used by Lampis [82] to reduce a certain variant of CSP, denoted by q-CSP(B), to k-Coloring. Let $q, B \geqslant 2$ be integers. The q-CSP(B) problem is defined as follows. An instance of q-CSP(B) consists of a set X of variables and a set C of q-constraints. A q-constraint $c \in C$ is a tuple of at most $q' \leqslant q$ elements from X and a set $P(c)$ of tuples of at most q' elements from $[B]$. The q-CSP(B) problem asks whether there exists an assignment $\gamma : X \to [B]$, such that each constraint is satisfied, i.e., if $c = ((x_1, \ldots, x_q), P(c)) \in C$, then $(\gamma(x_1), \ldots, \gamma(x_q)) \in P(c)$.

The following theorem of Lampis is the starting point of both reductions.

Theorem 4.2.2 ([82]). *For every $B \geqslant 2, \varepsilon > 0$ there exists q such that n-variable q-CSP(B) cannot be solved in time $(B - \varepsilon)^n n^{\mathcal{O}(1)}$, assuming the SETH.*

On a high level, in the proof of Lampis, the possible variable assignments are encoded by mapping specified subsets of vertices of an instance to arbitrary non-trivial subsets of colors. The straightforward generalization of this approach to our setting would be to map to non-trivial subsets of $V(H)$. However, the structure of H allows only certain configurations of subsets as images for the specified vertices in a solution for Hom(H)—which is precisely where the signature sets come into play.

Focusing on the ExtHom(H) problem allows us to slightly simplify the reasoning, as we can use the fact that some vertices of the defined instance are forced to be mapped to some specific vertices of H. This nicely combines with Lemma 3.2.7 that produces constructions of relations that we use to simulate the constraint satisfaction problem instance.

To prove Theorem 4.2.1, we introduce two types of constructions that are used in the proof. For fixed vertices $a, b \in V(H_1)$, let $\mathcal{S}_{a\to b} = \{(a', b') \mid a' \neq a, b' \in V(H_1)\} \cup \{(a, b)\}$. By Lemma 3.2.7, for any $\overline{y} = (y_1, y_2) \in V(W)^2$ there exists a construction of $R[\mathcal{S}_{a\to b}, \overline{y}]$. Intuitively, we use $R[\mathcal{S}_{a\to b}, \overline{y}]$ to simulate implication $a \Rightarrow b$, since in every homomorphism $h : F \to H$ that extends φ, if $\pi_1^\gamma \circ h(p) = a$, then $\pi_1^\gamma \circ h(q) = b$. Hence we call this type of construction an *implication construction*.

The second type of relation we need to express corresponds to disjunction of arity t,

for any fixed $t \in \mathbb{N}$. Let $a, b \in V(H_1)$. We define the relation

$$\mathrm{OR}_t(a,b) = \{(r_1, \ldots, r_t) \subseteq \{a,b\}^t \mid r_i = a \text{ for some } i \in [t]\}.$$

By Lemma 3.2.7, for any $y \in V(W)$ there exists a construction of $\mathcal{R}[\mathrm{OR}_t(a,b), \overline{y}_t]$, where $\overline{y}_t$ is the t-tuple $(y, y, \ldots, y)$. We call this construction an *or-construction* of arity t.

We have all the tools to perform the final reduction.

Proof of Theorem 4.2.1. Fix $\varepsilon > 0$ and set $B = s(H_1)$. As H is H_1-projective, H is non-trivial, and in particular, $B \geqslant 3$ by Observation 3.3.5. Fix two distinct vertices $a, b \in V(H_1)$. Since H is a non-trivial core, W must have at least one edge yy' (it may happen that $y = y'$). From now on a, b, y and y' are fixed. Let $q \in \mathbb{N}$ be such that q-CSP(B) on n variables cannot be solved in time $(B-\varepsilon)^n n^{\mathcal{O}(1)}$ assuming the SETH given by Theorem 4.2.2.

Let $\phi = (X, C)$ be an instance of q-CSP(B), where $X = \{x_1, \ldots, x_n\}$ is the set of variables and $C = \{c_0, \ldots, c_{m-1}\}$ is the set of constraints. For every $j \in \{0, \ldots, m-1\}$ denote by X_j the set of variables that appear in the constraint c_j. Let $P(c_j) = \{f_1^j, \ldots, f_{p_j}^j\}$. Note that each element of $P(c_j)$ naturally corresponds to a mapping from X_j to $[B]$. Clearly, for every $j \in \{0, \ldots, m-1\}$ we have $|p_j| \leqslant B^q$. Let $L = m(n|H_1|+1)$, and let $\lambda : [B] \to \mathcal{S}(H_1)$ be some fixed bijection.

We construct an instance (G_ϕ, L) of ExtHom(H). For each $j \in \{0, \ldots, L-1\}$, let $j' = j \mod m$. For every $j \in \{0, \ldots, L-1\}$ let $R_j = (r_1^j, \ldots, r_{p_{j'}}^j)$ be the tuple of newly introduced vertices, where each vertex r_k^j corresponds to the assignment $f_k^{j'} : X_{j'} \to [B]$ (in an arbitrarily fixed order). We introduce the or-construction (F_j, φ_j, R_j) of $R[\mathrm{OR}_{p_{j'}}(a,b), \overline{y}_{p_{j'}}]$.

For each $x_i \in X_{j'}$, and for each $f_k^{j'} \in P(c_{j'})$ we do the following:

1. Let $w = f_k^{j'}(x_i) \in [B]$. Construct an independent set $V_i^{j,k}$ of $|\lambda(w)|$ vertices and an independent set $U_i^{j,k}$ of $|S(\lambda(w))|$ vertices.

2. For each $d \in \lambda(w)$ select a distinct vertex $z_d \in V_i^{j,k}$ and add an implication construction $(F, \varphi, (r_k^j, z_d))$ of $R[\mathcal{S}_{a \to d}, (y, y)]$. For each $d \in S(\lambda(w))$ select a distinct vertex $z_d \in U_i^{j,k}$ and add an implication construction $(F, \varphi, (r_k^j, z_d))$ of $R[\mathcal{S}_{a \to d}, (y, y')]$.

3. Connect all vertices of $U_i^{j,k}$ with all vertices of previously constructed sets $V_i^{\ell,k'}$ for $\ell < j$ and $k' \in [p_\ell]$ (see Figure 4.2).

The partial mapping φ is the union of all the partial mappings that are introduced by the constructions. This concludes the definition of the instance (G_ϕ, φ) of ExtHom(H).

Claim 4.2.2.1. *If ϕ is a yes-instance of q-CSP(B), then there is a homomorphism $h : G_\phi \to H$ that extends φ.*

Proof of Claim. If ϕ is a yes-instance of q-CSP-B, then there exists an assignment $\gamma : X \to [B]$ satisfying each constraint. We define $h : G_\phi \to H$ as follows.

Fix $j \in \{0, \ldots, L-1\}$, and consider the or-construction (F_j, φ_j, R_j) of $R[\mathrm{OR}_{p_{j'}}(a, b), \overline{y}_{p_{j'}}]$. Recall that the set $P(c_{j'})$ consists of all assignments of variables in $X_{j'}$ that satisfy the constraint $c_{j'}$. Therefore, there exists an assignment $f_k^{j'} \in P(c_{j'})$ such that $\lambda|_{X_{j'}} \equiv f_k^{j'}$. Consider the vertex r_k^j that corresponds to that assignment. By Lemma 3.2.7 (R1), we know that there exists a homomorphism $h : F_j \to H_1 \times W$ that extends φ, such that

(i) $\pi_1^\gamma \circ h(r_k^j) = a$,

(ii) for every $r_{k'}^j \in R_j, k' \neq k$ we have that $\pi_1^\gamma \circ h(r_{k'}^j) = b$.

Let $x_i \in X_{j'}$ and let $w = f_k^{j'}(x_i) \in [B]$. Since for each $d \in \lambda(w)$ there exists a vertex $z_d \in V_i^{j,k}$ such that there is an implication construction $(F, \varphi, (r_k^j, z_d))$ of $R[R_{a\to d}, (y, y)]$, the condition (i) implies that $\pi_1^\gamma \circ h(V_i^{j,k}) = \lambda(w)$. We assign the vertices of H to the vertices of $V_i^{j,k}$ in a way that $h(V_i^{j,k}) = \lambda(w) \times \{y\}$.

Also, since for each $d \in S(\lambda(w))$ there exists a vertex $z_d \in U_i^{j,k}$ such that there is an implication construction $(F, \varphi, (r_k^j, z_d))$ of $\mathcal{R}(R_{a\to d}, (y, y'))$, the condition (i) implies that $h_1(U_i^{j,k}) = S(\lambda(w))$. We assign the vertices of H to the vertices of $U_i^{j,k}$ in a way that $h(U_i^{j,k}) = S(\lambda(w)) \times \{y'\}$.

Because of (ii), the implication constructions $(F, \varphi, (r_{k'}^j, z))$ for $z \in V_i^{j,k'} \cup U_i^{j,k'}$ do not put any constraints on the images of the sets $V_i^{j,k'}$ and $U_i^{j,k'}$. Therefore, for each $v \in V_i^{j,k'}$ we set $h(v)$ to be arbitrarily chosen vertex from $\lambda(w) \times \{y\}$. Similarly, for each $u \in U_i^{j,k'}$ we set $h(u)$ to be arbitrarily chosen vertex from $S(\lambda(w)) \times \{y'\}$. Since $(b, z) \in R_{a\to d}$ for any $z \in V(H_1)$ and $d \in \lambda(w) \cup S(\lambda(w))$, Lemma 3.2.7 (R1) asserts that we can always extend this to a homomorphism from F to H.

It remains to argue that the edges between the sets $V_i^{j_1,k_1}$ and $U_i^{j_2,k_2}$ are mapped to edges of H, for any $j_1 < j_2$ and k_1, k_2. Indeed, observe that since σ is an extension of some $f_{k_1}^{j'_1} \in P(c_{j'_1})$ and $f_{k_2}^{j'_2} \in P(c_{j'_1})$, we must have $f_{k_1}^{j'_1}(x_i) = f_{k_2}^{j'_2}(x_i) = w$. Hence, h

maps every $v \in V_i^{j_1,k_1}$ to some element of $\lambda(w) \times \{y\}$, and every $u \in U_i^{j_2,k_2}$ to some element of $S(\lambda(w)) \times \{y'\}$. By Observation 3.3.1, and since $yy' \in E(W)$, we get that $h(v)h(u) \in E(H)$. That concludes the proof of the claim. ⌟

Claim 4.2.2.2. *If there exists a homomorphism $h : G_\phi \to H$ that extends φ, then ϕ is a yes-instance of q-CSP-B.*

Proof of Claim. We will define the assignment $\sigma : X \to [B]$ that makes every constraint from C satisfied. Let $h_1 = \pi_1^\gamma \circ h : G_\phi \to H_1$.

Fix $j \in \{0, \ldots, L-1\}$, and consider the or-construction (F_j, φ_j, R_j) of $R[\mathrm{OR}_{p_{j'}}(a,b), \bar{y}_{p_{j'}}]$. By the definition of $\mathrm{OR}_{p_{j'}}(a,b)$, there exists $k = k_j \in [p_{j'}]$ such that $h_1(r_k^j) = a$. Observe that the implication gadgets $(F, \varphi, (r_k^j, z))$ for $z \in V_i^{j,k} \cup U_i^{j,k}$ assert that $h_1(V_i^{j,k}) \in \mathcal{S}(H_1)$ and that $S_{H_1}(h_1(V_i^{j,k})) = h_1(U_i^{j,k})$. Then, by Observation 3.3.2, we have

$$S_{H_1}(h_1(U_i^{j,k})) = S_{H_1}(S_{H_1}(h_1(V_i^{j,k})) = h_1(V_i^{j,k}).$$

Denote $h_1(U_i^{j,k})$ by T_i^j, then $h_1(V_i^{j,k}) = S_{H_1}(T_i^j)$. Recall that r_k^j corresponds to an assignment $f_k^{j'} \in P(c_{j'})$. Let $w_i^j = f_k^{j'}(x_i)$ be the *candidate assignment at index j for $x_i \in X_{j'}$*, recall that by the step 2. of the definition of G_ϕ we have $w_i^j = \lambda^{-1}(S_{H_1}(T_i^j))$.

Let $i \in [n]$ be fixed and let $j_1, j_2 \in [L], j_1 < j_2$ be such that $x_i \in X_{j_1'} \cap X_{j_2'}$. Observe that $T_i^{j_1} \supseteq T_i^{j_2}$. Indeed, denote $k_1 = k_{j_1}$ and $k_2 = k_{j_2}$ and observe that we have

(i) $h_1(V_i^{j_1,k_1}) = S_{H_1}(T_i^{j_1})$ and $h_1(U_i^{j_1,k_1}) = T_i^{j_1}$,

(ii) $h_1(V_i^{j_2,k_2}) = S_{H_1}(T_i^{j_2})$ and $h_1(U_i^{j_2,k_2}) = T_i^{j_2}$.

Recall that each vertex from $U_i^{j_2,k_2}$ is adjacent to each vertex from $V_i^{j_1,k_1}$. Since h_1 is a homomorphism, the same holds for their images: each vertex from $T_i^{j_2}$ is adjacent to each vertex from $S_{H_1}(T_i^{j_1})$. Then $S_{H_1}(T_i^{j_2}) \supseteq S_{H_1}(T_i^{j_1})$, so

$$T_i^{j_1} = S_{H_1}(S_{H_1}(T_i^{j_1})) \supseteq S_{H_1}(S_{H_1}(T_i^{j_2})) \supseteq T_i^{j_2}.$$

We say that an index $j_1 \in \{0, \ldots, L-1\}$ is *problematic* for i if there is $j_2 > j_1$ such that $x_i \in X_{j_1'} \cap X_{j_2'}$ and $T_i^{j_1} \neq T_i^{j_2}$. Since for each variable we have at most $|H_1|$ problematic indices, there are at most $|V(H_1)| \cdot n$ problematic indices for all variables. Since $L = m(|V(H_i)| \cdot n + 1)$, by pigeonhole principle we get that there exists a set

$J \subseteq \{0, \ldots, L-1\}$ of m consecutive indices such that none of them is problematic for any i. For every $i \in [n]$, we fix some $j \in J$ such that $x_i \in X_{j'}$ and set $\lambda(x_i) = w_i^j$ (observe that here the choice of j does not matter).

We claim that λ is an assignment that satisfies every constraint from ϕ. Indeed, for any $j' \in [m]$ there exists $j \in J$ such that $j' = j \mod m$. For every $i \in X_{j'}$, we have $\sigma(x_i) = w_i^j = f_{k_j}^{j'}(x_i)$, so λ satisfies $c_{j'}$. This concludes the proof of claim. ⌟

Finally, it remains to adapt the arguments of Lampis [82] to establish the desired clique-width bound.

Claim 4.2.2.3. *G_ϕ can be constructed in time polynomial in n, and we have $\mathrm{cw}(G_\phi) \leqslant n + f(\varepsilon, |V(H)|)$ for some function f.*

Proof of Claim. By Lemma 3.2.7 (R3), every implication construction has at most $|V(H)|^{|V(H)|^2}$ vertices. By the same property, each or-construction of arity $p_{j'}$ has $|V(H)|^{p_{j'}}$ vertices.

For fixed H and $\varepsilon > 0$, by Observation 3.3.4 we have that $B = s(H_i) \leqslant 2^{|V(H_i)|} - 2$ and q is a constant that only depends on B, ε (that is, on $|V(H_i)|$, ε). Each constraint of the q-CSP(B) instance has at most B^q satisfying assignments. In particular, the number of vertices in each or-construction is upper-bounded by $|V(H)|^{B^q}$. Therefore, we see that the whole construction can be performed in polynomial time, if H is fixed and ε is a constant. In the clique-width expression for G_ϕ we use the following labels:

1. n *main* labels, representing the variables of ϕ.
2. A single *done* label. Its informal meaning is that a vertex that receives this label will not be connected to anything else not yet introduced in the graph.
3. $|V(H)|^{B^q}$ *or-construction* labels.
4. $qB^q \cdot |V(H)|^{|V(H)|^2}$ *variable-constraint incidence work* labels.

Thus, $f(\varepsilon, |V(H)|) = 1 + |V(H)|^{B^q} + qB^q \cdot |V(H)|^{|V(H)|^2}$.

To give a clique-width expression we will describe how to build the graph, following essentially the steps given in the description of the construction by maintaining the following invariant: before starting iteration j, all vertices of the set $W_i^j = \bigcup_{j'<j} \bigcup_{k \in [p_{j'}]} V_i^{j',k}$ have label i, and all other vertices introduces so far have the *done* label. This invariant

is vacuously satisfied at the beginning, since the graph is empty. Suppose that for some $j \in \{0, \ldots, L-1\}$ the invariant is true. We use the $|V(H)|^{B^q}$ or-construction labels to introduce the vertices of the $p_{j'}$-or-gadget (F_j, φ_j, R_j), giving each vertex a distinct label. We use join operations to construct the internal edges of the or-gadget. Then, for each variable x_i that appears in the current constraint we do the following: we use $qB^q \cdot |V(H)|^{|V(H)|^2}$ of the variable-constraint incidence work labels to introduce for all $k \in [p_{j'}]$ the vertices of $V_i^{j,k}$ and $U_i^{j,k}$ as well as the implication gadgets connecting these to r_k^j . Again we use a distinct label for each vertex, but the number of vertices (including internal vertices of the implication gadgets) is at most $qB^q \cdot |V(H)|^{|V(H)|^2}$, so we have sufficiently many labels to use distinct labels for each of the q variables of the constraint. We use join operations to add the edges inside all implication gadgets. Then we use join operations to connect $U_i^{j,k}$ to all vertices W_i^j. This is possible, since the invariant states that all the vertices of W_i^j have the same label i. We then rename all the vertices of $U_i^{j,k}$ for all k to the done label, and do the same also for internal vertices of all implication gadgets. We proceed to the next variable of the same constraint and handle it using $qB^q \cdot |V(H)|^{|V(H)|^2}$ labels. Once we have handled all variables of the current constraint, we rename all vertices of each $V_i^{j,k}$ to label i for all k. We then relabel all vertices of the $p_{j'}$-or-gadget (F_j, φ_j, R_j) gadget to the done label and increase j by 1. Now it is straightforward to verify that we have maintained the invariant and constructed all edges induced by the vertices introduced in steps up to j, so repeating this process constructs the graph. ⌟

Together the claims imply Theorem 4.2.1 in the following way: For an arbitrary instance ϕ of q-CSP(B), our construction produces an instance $I = (G_\phi, \varphi)$ of Ex-tHom(H), and the instances are equivalent by Claim 4.2.2.1 and Claim 4.2.2.2. Assume that we can solve $I = (G_\phi, \varphi)$ in time $(s(H_1) - \varepsilon)^{\mathrm{cw}(G_\phi)} \cdot |V(G_\phi)|^{\mathcal{O}(1)}$ for $\varepsilon > 0$. We can use our construction to transform ϕ into (G_ϕ, φ) in time polynomial in n. Thus, it would be possible to solve ϕ in time

$$\begin{aligned}\Big(s(H_1) - \varepsilon\Big)^{\mathrm{cw}(G_\phi)} \cdot |V(G_\phi)|^{\mathcal{O}(1)} + n^{\mathcal{O}(1)} &\leqslant \Big(s(H_1) - \varepsilon\Big)^{n + f(\varepsilon, |V(H)|)} \cdot n^{\mathcal{O}(1)} \\ &\leqslant \Big(s(H_i) - \varepsilon\Big)^{f(\varepsilon, |V(H)|)} \cdot \Big((s(H_i) - \varepsilon\Big)^n \cdot n^{\mathcal{O}(1)}.\end{aligned}$$

As $\Big(s(H_1) - \varepsilon\Big)^{f(\varepsilon, |V(H)|)}$ is a constant that depends only on $B = s(H_1)$, ε and H, this proce-

dure would have complexity $(B-\varepsilon)^n n^{\mathcal{O}(1)}$. By our choice of q according to Theorem 4.2.2, this contradicts the SETH. □

Now, Theorem 4.2.1 gives us the following corollary, that together with Corollary 4.1.2 give us tight bounds for the complexity of $\textsc{Hom}(H)$ when parameterized by clique-width, for all connected target graphs.

Corollary 4.2.3. *Assume that Conjecture 1 and Conjecture 2 hold. Let H be a fixed connected graph and let H' be its core. Let $(H_1, \ldots, H_m)$ be the prime factorization of H. Let* cw *be the clique-width of an n-vertex instance graph G. There is no algorithm solving the $\textsc{Hom}(H)$ problem in time*

$$\max_{i\in[m]}(s(H_i)-\varepsilon)^{\mathrm{cw}} \cdot n^{\mathcal{O}(1)},$$

unless the SETH fails.

Proof. Assume the corollary does not hold, i.e., there exists an algorithm $\mathcal{A}$ that solves $\textsc{Hom}(H)$ in time $\max_{i\in[m]}(s(H_i)-\varepsilon)^{\mathrm{cw}} \cdot n^{\mathcal{O}(1)}$.

Let (G', φ) be an instance of $\textsc{ExtHom}(H)$. Using Theorem 3.2.9, we transform (G', φ) into an equivalent instance G of $\textsc{Hom}(H)$, such that $|V(G)| = |V(G')| + |V(H)|$ and G' is an induced subgraph of G. Recall from the proof that Theorem 4.2.1 implies Theorem 1.3.2 (b) that $\mathrm{cw}(G) \leqslant \mathrm{cw}(G') + |V(H)|$.

Without loss of generality assume that (H_1, W), for $W = H_2 \times \ldots \times H_m$, is the canonical factorization of H. In particular $s(H_1) = \max_{j\in[m]} s(H_j)$. Now, by the definition of a prime factorization, the graph H_1 is prime. Thus, assuming Conjecture 1, H_1 is projective, and assuming Conjecture 2, the graph H is H_1-projective. Thus can use $\mathcal{A}$ to decide whether $G \to H$, and, equivalently, whether φ can be extended to a homomorphism from G to H, in time

$$\begin{aligned}\max_{i\in[m]}(s(H_i)-\varepsilon)^{\mathrm{cw}} \cdot n^{\mathcal{O}(1)} = (s(H_1)-\varepsilon)^{\mathrm{cw}} \cdot n^{\mathcal{O}(1)} &= s(H_1)-\varepsilon)^{\mathrm{cw}(G')+|V(H)|} \cdot n^{\mathcal{O}(1)} \\ &= s(H_1)-\varepsilon)^{\mathrm{cw}(G')} \cdot n^{\mathcal{O}(1)},\end{aligned}$$

and this contradicts Theorem 4.2.1. □

As a last remark, let us mention that the restriction to connected target graphs in Corollary 4.1.2 and Corollary 4.2.3 can be avoided: on the algorithmic side, it is enough

to branch over all connected component of H (separately for each connected component of G), while for the lower bound it can be shown that focusing on connected targets solves the problem in general, for example by repeating the construction in [100, Lemma 3.4].

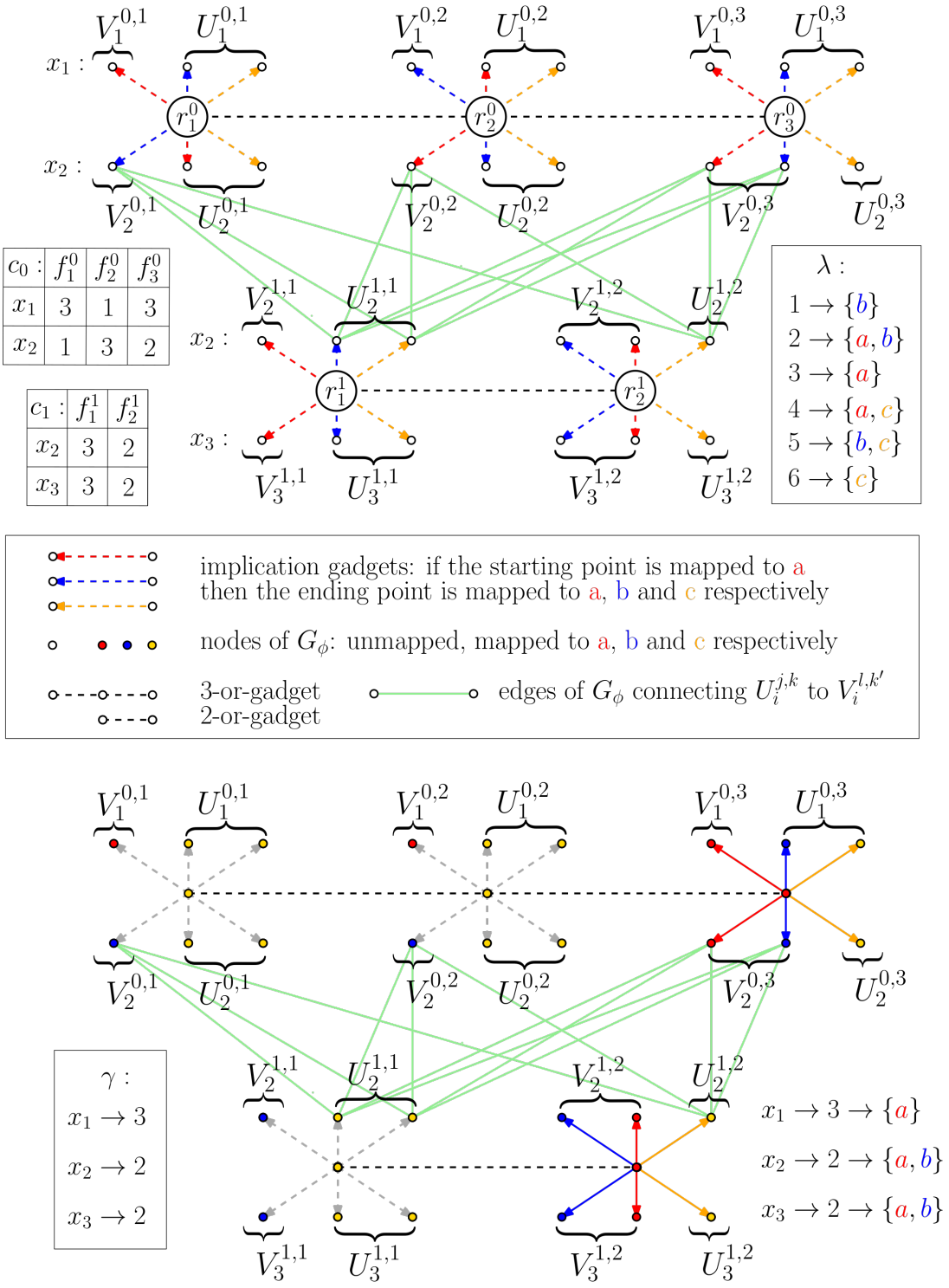


Figure 4.2: Construction of the instance of ExtHom(K_3) for $B = 6$, variable set $X = \{x_1, x_2, x_3\}$ and constraint set $C = \{c_0, c_1\}$. The satisfying assignments f_k^j and the bijection λ are as depicted. The vertices of K_3 are a, b, c. Top: fragment of G_ϕ restricted to $j = 0, 1$. Bottom: the homomorphism from G_ϕ to H corresponding to the satisfying assignment γ. Author: Viktoriia Korchemna.

Chapter 5

List homomorphisms: toolbox

In this chapter we define the basic notions and tools that we use to study the list version of the homomorphism problem. For graphs G, H and a function $L : V(G) \to 2^{V(H)}$, we say that a homomorphism $h : G \to H$ *respects* L, if for every $v \in V(G)$ we have $h(v) \in L(v)$. In that case we write $h : (G, L) \to H$ or simply $(G, L) \to H$, and if no such homomorphism exists, we write $(G, L) \not\to H$.

Since in this and the following sections we consider list homomorphisms, we emphasize that in contrary to the non-list version, this notion does not trivialize if the target graph H has loops. Hence, unless explicitly stated otherwise, we assume that all target graphs H in the following sections may contain loops. Note, however, that we may still assume that instance graphs G are irreflexive, as it is enough to remove all irreflexive vertices from the list of a reflexive vertex $v \in V(G)$ and then remove its loop to obtain an equivalent instance.

For a graph $G = (V, E)$, by G^* we denote the *associated bipartite graph* $G \times K_2$. Formally, vertices of G^* are pairs (v, u) such that $v \in V(G)$ and $u \in \{u', u''\} = V(K_2)$. For brevity, if $v \in V(G)$, we denote (v, u') by v' and (v, u'') by v''. The vertices v' and v'' are called *twins*.

Theorem 5.0.1 ([62], Theorem 5.9). *Let G be a connected graph. Then G is bipartite if and only if G^* is disconnected. Moreover, if G is bipartite, then G^* consists of two isomorphic copies of G.*

For graphs G and H and function $L : V(G) \to 2^{V(H)}$ we define the *associated list function* $L^* : V(G^*) \to 2^{V(H^*)}$ as follows. For $v \in V(G)$, we set $L^*(v') = \{a' \mid a \in L(v)\}$

and $L^*(v'') = \{a'' \mid a \in L(v)\}$. Note that in the associated lists, the vertices appearing in $L^*(v')$ are precisely the twins of the vertices appearing in the list of $L^*(v'')$.

A homomorphism $h\colon (G^*, L^*) \to H^*$ is *clean* if it maps twins to twins, i.e., $f(v') = a'$ if and only if $f(v'') = a''$. The following simple observation was the crucial step of the proof of the complexity dichotomy for list homomorphisms, shown by Feder, Hell, and Huang [39]. We state it using slightly different language, which is more suitable for our purpose.

Proposition 5.0.2 (Feder, Hell, Huang [39]). *There is a homomorphism $h : (G, L) \to H$ if and only if there is a clean homomoprhism $(G^*, L^*) \to H^*$.*

Let us point out that the restriction to clean homomorphisms is necessary for the equivalence. Indeed, consider for example graphs $G = K_3$ and $H = C_6$, and the list function $L : V(G) \to 2^{V(K_3)}$ that assigns to each vertex $v \in V(G)$ the set $V(K_3)$. Clearly $G \not\to H$, so $(G, L) \not\to H$. However, we have $G^* \simeq C_6$ and $H^* \simeq 2C_6$, so $(G^*, L^*) \to H^*$.

5.1 Bi-arc graphs and their characterizations

As explained that Feder, Hell, and Huang [39] proved that the LHom(H) problem is polynomial-time solvable if H is a so-called *bi-arc* graph, and is NP-complete otherwise. In their work bi-arc graphs are defined in terms of a certain geometric representations, however, equivalent definitions in the language of the associated bipartite graphs H^* are also provided there. In this section we introduce these definitions, and discuss some elementary properties of the related objects.

Let C be a circle and let p and q be distinct, fixed points on C. A *bi-arc* is a pair (N, S) of arcs such that $p \in N \setminus S$ and $q \in S \setminus N$. A graph H is bi-arc if there exists a family $\{(N_x, S_x) \mid x \in V(H)\}$ of bi-arcs such that, for any (not necessarily distinct) $x, y \in V(H)$ we have that (i) if $xy \in E(H)$, then $N_x \cap S_y = N_y \cap S_x = \emptyset$, and (ii) if $xy \in E(H)$, then $N_x \cap S_y \neq \emptyset$ and $N_y \cap S_x \neq \emptyset$.

A *circular-arc* graph is a graph that can be represented as the intersection graph of a family of arcs of a circle. An equivalent definition of bi-arc graphs, by Feder et al. [39], states that an arbitrary graph H (with loops allowed) is a bi-arc graph if and only if H^* is the complement of a circular-arc graph.

For the other characterization, we first need to consider certain properties of walks. Let $\mathcal{P} = (p_1, \ldots, p_\ell)$ be a walk. By $\overline{\mathcal{P}}$ we denote the walk $\mathcal{P}$ reversed, i.e., $\overline{\mathcal{P}} = (p_\ell, \ldots, p_1)$. If vertices a and b are, respectively, the first and the last vertex of a walk $\mathcal{P}$, we say that $\mathcal{P}$ is an a-b-walk and denote it by $\mathcal{P} : a \to b$.

Definition 5.1.1 (Avoiding). For walks $\mathcal{P} = (p_1, \ldots, p_\ell)$ and $\mathcal{Q} = (q_1, \ldots, q_\ell)$ of equal length we say $\mathcal{P}$ *avoids* $\mathcal{Q}$ if $p_1 \neq q_1$ and for every $i \in [\ell - 1]$ it holds that $p_i q_{i+1} \notin E(H)$.

We note that the condition $p_1 \neq q_1$ does not follow from the second condition only if $\ell = 1$.

The following observations are immediate and summarize the basic properties of avoiding walks.

Observation 5.1.2. *Assume that $\mathcal{P} = (p_1, p_2 \ldots, p_\ell)$ avoids $\mathcal{Q} = (q_1, q_2, \ldots, q_\ell)$. Then*

1. *$\overline{\mathcal{Q}}$ avoids $\overline{\mathcal{P}}$,*
2. *if $\ell \geqslant 2$, then $q_2 \in N(q_1) \setminus N(p_1)$ and $p_{\ell-1} \in N(p_\ell) \setminus N(q_\ell)$. In particular, $N(q_1) \not\subseteq N(p_1)$ and $N(p_\ell) \not\subseteq N(q_\ell)$.* □

For walks $\mathcal{P} = (p_1, \ldots, p_\ell)$ and $\mathcal{Q} = (q_1, \ldots, q_k)$ such that $p_\ell = q_1$, we define

$$\mathcal{P} \circ \mathcal{Q} := (p_1, \ldots, p_\ell, q_2, \ldots, q_k).$$

In particular, $(p_1, \ldots, p_\ell) \circ (p_\ell) = (p_1, \ldots, p_\ell)$. We observe the following.

Observation 5.1.3. *If $\mathcal{P} : x \to y$ avoids $\mathcal{Q} : p \to q$ and $\mathcal{P}' : y \to z$ avoids $\mathcal{Q}' : q \to r$, then $\mathcal{P} \circ \mathcal{P}'$ avoids $\mathcal{Q} \circ \mathcal{Q}'$.* □

Recall that if it does not lead to confusion, we identify a sequence of vertices $\mathcal{P}$ with the set of vertices that appear in $\mathcal{P}$, so in particular we say that walks $\mathcal{P}$ and $\mathcal{Q}$ are non-adjacent. Clearly, if two walks of the same length are non-adjacent, they avoid each other.

For a walk $\mathcal{P} = (p_1, p_2 \ldots, p_\ell)$ and $1 \leqslant i \leqslant j \leqslant \ell$, we define the *$p_i$-$p_j$-subwalk* of $\mathcal{P}$, denoted by $\mathcal{P}[p_i, p_j]$, as $\mathcal{P}[p_i, p_j] := (p_i, p_{i+1}, \ldots, p_{j-1}, p_j)$. (Note that the vertices p_i and p_j might appear on $\mathcal{P}$ more than once, then $\mathcal{P}[p_i, p_j]$ can be chosen arbitrarily.)

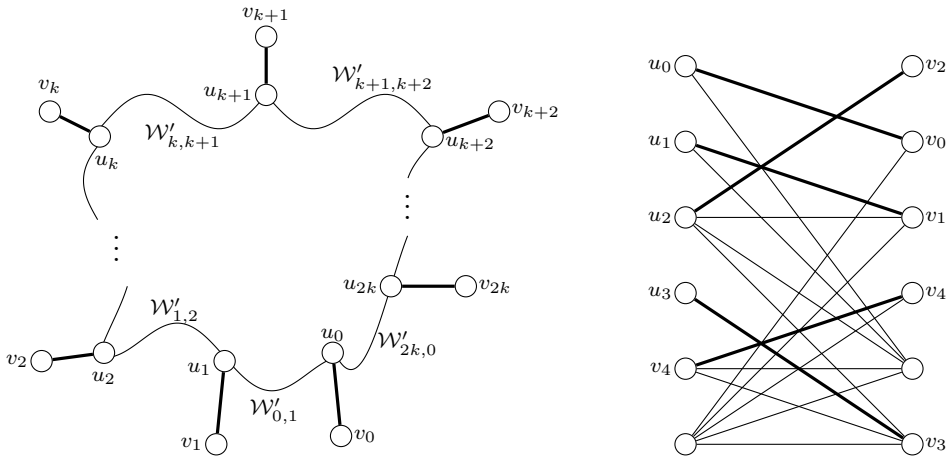


Figure 5.1: A general scheme of a special edge asteroid (left) and an example (right).

We define a structure called a *special edge asteroid*, see also Figure 5.1.[1]

Definition 5.1.4 (Special edge asteroid). Let H be a bipartite graph, with bipartition classes X and Y, let $k \geqslant 1$, and let $A_X = (u_0, \ldots, u_{2k})$ and $A_Y = (v_0, \ldots, v_{2k})$ be sequences of pairwise distinct vertices, respectively, from X and Y. We say that (A_X, A_Y) is a *special edge asteroid* if, for each $i \in \{0, \ldots, 2k\}$, we have $u_i v_i \in E(H)$, and there exists a u_i-u_{i+1}-path $\mathcal{W}_{i,i+1}$ (indices are computed modulo $2k+1$), such that

(a) sets $\{u_i, v_i\}$ and $\{v_{i+k}, v_{i+k+1}\} \cup \mathcal{W}_{i+k,i+k+1}$ are non-adjacent, and

(b) sets $\{u_0, v_0\}$ and $\{v_1, \ldots, v_{2k}\} \cup \bigcup_{i=1}^{2k-1} \mathcal{W}_{i,i+1}$ are non-adjacent.

Now, yet another characterization of bi-arc graphs by Feder et al. [38, 39] says that a (not necessarily bipartite) graph H is a bi-arc graph if and only if H^* does not contain an induced cycle on at least 6 vertices or a special edge asteroid.

In a bipartite graph H with bipartition classes X and Y, fix a special edge asteroid (A_X, A_Y), where $A_X = (u_0, \ldots, u_{2k})$ and $A_Y = (v_0, \ldots, v_{2k})$. We observe that there might be many ways to choose paths $\mathcal{W}_{i,i+1}$. It is convenient for us to use paths that satisfy certain minimality conditions. We say that a family $\{\mathcal{W}_{i,i+1}\}_{i=0}^{2k}$ of paths is *canonical with respect to* (A_X, A_Y), if for each $i \in \{0, \ldots, 2k\}$, the path $\mathcal{W}_{i,i+1}$ is a shortest path containing vertices $u_i, v_i, u_{i+1}, v_{i+1}$, such that conditions (a) and (b) from Definition 5.1.4

[1] Let us point out that the definition of a special edge asteroid from [38] is slightly different than ours: there it is a set of edges $u_0v_0, u_1v_2, \ldots, u_{2k}v_{2k}$, for which there exist paths satisfying properties (a) and (b) from Definition 5.1.4. However, for our purpose it it much more convenient to think about special edge asteroids as sequences of vertices.

are satisfied. Note that the existence of a canonical family of paths follows directly from the fact that (A_X, A_Y) is a special edge asteroid. Furthermore, since H is bipartite, each $\mathcal{W}_{i,i+1}$ induces either a path or a cycle C_4.

For every $i \in \{0, \dots 2k\}$ denote by $\mathcal{W}^u_{i,i+1}$ (resp. $\mathcal{W}^v_{i,i+1}$) the u_i-u_{i+1}-path (resp. v_i-v_{i+1}-path) that is contained in $\mathcal{W}_{i,i+1}$. Clearly, $\mathcal{W}^u_{i,i+1}$ and $\mathcal{W}^v_{i,i+1}$ are induced paths. Furthermore, the family $\{\mathcal{W}^u_{i,i+1}\}_{i=0}^{2k}$ of paths satisfies the conditions of Definition 5.1.4. Similarly, the family $\{\mathcal{W}^v_{i,i+1}\}_{i=0}^{2k}$ of paths satisfies conditions (a) and (b) of Definition 5.1.4, with the roles of u_i and v_i swapped. Hence, a special edge asteroid is symmetric with respect to the bipartition of H:

Lemma 5.1.5. *Let (A_X, A_Y) be a special edge asteroid, and let $\{\mathcal{W}_{i,i+1}\}_{i=0}^{2k}$ be its canonical family of paths. Then (A_Y, A_X) is a special edge asteroid and $\{\mathcal{W}_{i,i+1}\}_{i=0}^{2k}$ is also the family of canonical paths for (A_Y, A_X).* □

From now on, we always assume implicitly that special edge asteroids (A_X, A_Y) and (A_Y, A_X) come with a fixed family $\{\mathcal{W}_{i,i+1}\}_{i=0}^{2k}$ of canonical paths. Note that such a family can easily be found in time polynomial in $|V(H)|$ (for a given asteroid).

For a special edge asteroid (A_X, A_Y) we define the *asteroidal subgraph* to be the subgraph $\mathbb{O}$ of H' induced by the set $\bigcup_{i=0}^{2k} \mathcal{W}_{i,i+1}$. In particular $A_X \cup A_Y \subseteq V(\mathbb{O})$. Observe that if $\mathbb{O}$ is the asteroidal subgraph for (A_X, A_Y), it is also the asteroidal subgraph for (A_Y, A_X). Therefore, whenever we find in H an asteroidal subgraph, we can freely choose the appropriate special edge asteroid, usually we do it implicitly. In a natural way we treat paths that belong to asteroidal subgraph as walks. In particular, for each $i \in \{0, \dots 2k\}$ we define $\mathcal{W}_{i+1,i} := \overline{\mathcal{W}_{i,i+1}}$, $\mathcal{W}^u_{i+1,i} := \overline{\mathcal{W}^u_{i,i+1}}$, and $\mathcal{W}^v_{i+1,i} := \overline{\mathcal{W}^v_{i,i+1}}$.

An induced subgraph $\mathbb{O}$ of H' is an *obstruction* if it is isomorphic to C_6 or C_8, or it is an asteroidal subgraph for some special edge asteroid. Observe that each obstruction is connected, and each induced cycle on at least 10 vertices contains a special edge asteroid

Summarizing, we can state the characterizations of bi-arc graphs as follows.

Theorem 5.1.6 (Feder, Hell, and Huang [38, 39]). *Let H be a graph. The following conditions are equivalent:*

- *H is a bi-arc graph,*
- *H^* is the complement of a circular-arc graph,*

- *H^* does not contain an obstruction as an induced subgraph.*

We note that Theorem 5.1.6 together with Theorem 5.0.1 imply that a bipartite graph H is a bi-arc graph if and only if H is the complement of a circular arc graph. Indeed, if H is bipartite, then by Theorem 5.0.1, H^* consists of two copies of H. Since every obstruction is a connected subgraph, H is the complement of circular-arc graph if and only if H^* is.

Let us mention that recognizing circular-arc graphs (and so their complements) can be done in time polynomial in $|V(H)|$ [92]. In the positive case we can also find a circular-arc representation of the input graph. However, up to the best of our knowledge, there is no known algorithm that provides also a certificate of the negative case. In particular, we are not aware of any algorithm finding a special edge asteroid in polynomial time.

Obstructions. In this part we discuss some basic properties of obstructions. Here, we denote the consecutive vertices of the cycle C_k by $w_1, w_2, \ldots, w_k$ (with $w_1w_k \in E(C_k)$). For special edge asteroids, we use the notation from Definition 5.1.4.

Definition 5.1.7 ($C(\mathbb{O})$). For an obstruction $\mathbb{O}$, we define

$$C(\mathbb{O}) := \begin{cases} \{(w_1, w_5), (w_2, w_4)\} & \text{if } \mathbb{O} \text{ is isomorphic to } C_6, \\ \{(w_1, w_5), (w_2, w_6)\} & \text{if } \mathbb{O} \text{ is isomorphic to } C_8, \\ \{(u_0, u_1), (v_0, v_1)\} & \text{if } \mathbb{O} \text{ is an asteroidal subgraph.} \end{cases}$$

For a pair $(\alpha, \beta) \in C(\mathbb{O})$ we define (α', β') to be the other element of $C(\mathbb{O})$. Due to the symmetry of cycles and Lemma 5.1.5, we are able to assume symmetry between the elements of $C(\mathbb{O})$, i.e., the properties of (α, β) that we require are also satisfied for (α', β'). Thus in proofs it is usually sufficient to consider one pair in $C(\mathbb{O})$, as the proof for the other one is analogous.

In the following two observations we notice the existence of some useful walks in $\mathbb{O}$.

Observation 5.1.8. *Let $\mathbb{O}$ be an obstruction in H and let $C(\mathbb{O}) = \{(\alpha, \beta), (\alpha', \beta')\}$. Then $\mathbb{O} \setminus N[\alpha, \alpha']$ is connected. In particular, for every pair of vertices x, y of $\mathbb{O}$, such that $x, y \notin N[\alpha, \alpha']$, there exists an x-y-walk $\mathcal{W}[x, y]$, using only vertices of $\mathbb{O}$, which is non-adjacent to $\{\alpha, \alpha'\}$.* □

The notation used in Observation 5.1.8 is justified by interpreting these walks in the case

when $\mathbb{O}$ is an asteroidal subgraph as subwalks of

$$\mathcal{W} := \mathcal{W}^u_{0,1}[p, u_1] \circ (u_1, v_1) \circ \mathcal{W}^v_{1,2} \circ (v_2, u_2) \circ \mathcal{W}^u_{2,3} \circ \ldots \circ \mathcal{W}^v_{2k-1,2k} \circ (v_{2k}, u_{2k}) \circ \mathcal{W}^u_{2k,0}[u_{2k}, q],$$

where p and q are, respectively, the first vertex of $\mathcal{W}^u_{0,1}$, which is not in $N[u_0, v_0]$, and the last vertex of $\mathcal{W}^u_{2k,0}$, which is not in $N[u_0, v_0]$.

Observation 5.1.9. *Let H be a bipartite graph with an obstruction $\mathbb{O}$, let $(\alpha, \beta) \in C(\mathbb{O})$.*

1. *There exist walks $\mathcal{X}, \mathcal{X}' : \alpha \to \beta$ and $\mathcal{Y}, \mathcal{Y}' : \beta \to \alpha$, such that $\mathcal{X}$ avoids $\mathcal{Y}$ and $\mathcal{Y}'$ avoids $\mathcal{X}'$.*

2. *Let $\mathbb{O}$ be an asteroidal subgraph, let $\gamma \in \{u_{k+1}, v_{k+1}\}$ be in the same bipartition class as α, β, and let $c \in \{\alpha, \beta, \gamma\}$. Then for any a, b, such that $\{a, b, c\} = \{\alpha, \beta, \gamma\}$, there exist walks $\mathcal{X}_c : \alpha \to a$ and $\mathcal{Y}_c : \alpha \to b$, and $\mathcal{Z}_c : \beta \to c$, such that $\mathcal{X}_c, \mathcal{Y}_c$ avoid $\mathcal{Z}_c$ and $\mathcal{Z}_c$ avoids $\mathcal{X}_c, \mathcal{Y}_c$.*

All walks use only vertices of $\mathbb{O}$.

Proof. Let us start with proving (a) in the case if $\mathbb{O}$ is either isomorphic to C_6 or to C_8; recall that consecutive vertices of such a cycle are denoted by $w_1, w_2, \ldots$. We can also assume that $\alpha = w_1$ and $\beta = w_5$, as the other case is symmetric.

If $\mathbb{O} \simeq C_6$, we set

$$\begin{aligned} \mathcal{X} &:= (w_1, w_6, w_5, w_4, w_5) \quad \mathcal{X}' := (w_1, w_2, w_3, w_4, w_5), \\ \mathcal{Y} &:= (w_5, w_4, w_3, w_2, w_1) \quad \mathcal{Y}' := (w_5, w_6, w_1, w_2, w_1). \end{aligned}$$

If $\mathbb{O} \simeq C_8$, we set

$$\begin{aligned} \mathcal{X} = \mathcal{X}' &:= (w_1, w_2, w_3, w_4, w_5), \\ \mathcal{Y} = \mathcal{Y}' &:= (w_5, w_6, w_7, w_8, w_1). \end{aligned}$$

It is straightforward to verify that these walks satisfy the statement (a).

So from now on let us assume that $\mathbb{O}$ is an asteroidal subgraph, and, by symmetry, that $\alpha = u_0$ and $\beta = u_1$. Then $\gamma = u_{k+1}$. Before we proceed to the proof of (a), let us

define three auxiliary walks $\mathcal{A} : u_0 \to u_0$, $\mathcal{B} : u_0 \to u_{k+1}$, and $\mathcal{C} : u_1 \to u_1$, as follows.

$$\begin{aligned}
\mathcal{A} &:= \mathcal{U}_0 \circ \mathcal{U}_0 \circ \mathcal{U}_0 \circ \mathcal{U}_0 \circ \ldots \circ \mathcal{U}_0 \circ \mathcal{U}_0, \\
\mathcal{B} &:= \mathcal{U}_0 \circ \mathcal{W}^u_{0,2k} \circ \mathcal{U}_{2k} \circ \mathcal{W}^u_{2k,2k-1} \circ \ldots \circ \mathcal{U}_{k+2} \circ \mathcal{W}^u_{k+2,k+1}, \\
\mathcal{C} &:= \mathcal{W}[u_1, u_k] \circ \mathcal{U}_k \circ \mathcal{W}^u_{k,k-1} \circ \mathcal{U}_{k-1} \circ \ldots \circ \mathcal{W}^u_{2,1} \circ \mathcal{U}_1,
\end{aligned}$$

where by $\mathcal{U}_i$ we mean a walk $(u_i, v_i, u_i, \ldots, v_i, u_i)$ of the appropriate length (so that the parts of $\mathcal{A}$, $\mathcal{B}$ and $\mathcal{C}$ are of the same length), and $\mathcal{W}[u_1, u_k]$ is obtained by Observation 5.1.8. Note that here we abuse the notation slightly, as the length of each $\mathcal{U}_i$ might be different. By Definition 5.1.4, we have that $\mathcal{U}_i$ and $\mathcal{W}^u_{i+k,i+k+1}$ are non-adjacent, and so are sets $\{u_0, v_0\}$ and $\{v_1, \ldots, v_{2k}\} \cup \bigcup_{i=1}^{2k-1} \mathcal{W}^u_{i,i+1}$. So it is straightforward to observe that $\mathcal{A}, \mathcal{B}$ avoid $\mathcal{C}$ and $\mathcal{C}$ avoids $\mathcal{A}, \mathcal{B}$.

In order to prove (a), we define

$$\begin{aligned}
\mathcal{X} = \mathcal{X}' &:= \mathcal{B} \circ \mathcal{U}_{k+1} \circ \mathcal{W}[u_{k+1}, u_1], \\
\mathcal{Y} = \mathcal{Y}' &:= \mathcal{C} \circ \mathcal{W}^u_{1,0} \circ \mathcal{U}_0,
\end{aligned}$$

where $\mathcal{W}[u_{k+1}, u_1]$ is given by Observation 5.1.8, and $\mathcal{U}_i$ is defined as previously. Observe that these walks avoid each other because $\mathcal{B}$ and $\mathcal{C}$ avoid each other, the walk $\mathcal{W}^u_{1,0}$ is non-adjacent to $\{u_{k+1}, v_{k+1}\}$, and $\mathcal{W}[u_{k+1}, u_1]$ is non-adjacent to $\{u_0, v_0\}$ by Observation 5.1.8.

Now let us show (b). Consider three cases. If $c = u_0$, then by symmetry assume that $u_1 = a, u_{k+1} = b$, and define

$$\begin{aligned}
\mathcal{X}_c &:= \mathcal{X} \circ \mathcal{U}_1, \\
\mathcal{Y}_c &:= \mathcal{X} \circ \mathcal{W}[u_1, u_{k+1}], \\
\mathcal{Z}_c &:= \mathcal{Y} \circ \mathcal{U}_0,
\end{aligned}$$

where walks $\mathcal{X}, \mathcal{Y}$ are given by statement (a) and $\mathcal{W}[u_1, u_{k+1}]$ is given by Observation 5.1.8. Note that in case of asteroidal subgraph we have $\mathcal{X} = \mathcal{X}'$ and $\mathcal{Y} = \mathcal{Y}'$, and therefore $\mathcal{X}, \mathcal{Y}$ avoid each other.

If $c = u_1$, then we assume that $u_0 = a, u_{k+1} = b$ and we can observe that our auxiliary

walks $\mathcal{A}, \mathcal{B}, \mathcal{C}$ already satisfy the statement of the lemma. Indeed, is it sufficient to set

$$\mathcal{X}_c := \mathcal{A}, \quad \mathcal{Y}_c := \mathcal{B}, \quad \mathcal{Z}_c := \mathcal{C}.$$

Finally, for $c = u_{k+1}$ and $u_0 = a, u_1 = b$, we can define

$$\begin{aligned} \mathcal{X}_c &:= \mathcal{U}_0 &&\circ \mathcal{U}_0, \\ \mathcal{Y}_c &:= \mathcal{U}_0 &&\circ \mathcal{W}^u_{0,1}, \\ \mathcal{Z}_c &:= \mathcal{W}[u_1, u_{k+1}] &&\circ \mathcal{U}_{k+1}, \end{aligned}$$

where $\mathcal{W}[u_1, u_{k+1}]$ is given by Observation 5.1.8. This completes the proof. □

5.2 Decompositions

Recall (Observation 3.2.2) that if a graph H is of the form $H_1 \times \ldots \times H_m$, we can determine whether $G \to H$ by checking whether $G \to H_i$ for every $i \in [m]$. The proof of Corollary 4.2.3 exploits this fact: to solve $\textsc{Hom}(H)$ for every non-trivial target graph H, it is enough to focus only on these target graphs that are prime (or even projective, assuming Conjecture 1). Thus, in some algorithmic applications, instead of solving $\textsc{Hom}(H)$, we can solve $\textsc{Hom}(H')$ for every H' that is a prime factor of H.

While the factorization approach cannot be easily generalized to the list version of the problem, in this section we introduce certain notions of decompositions of a graph H that, intuitively, play a similar role for $\textsc{LHom}(H)$. For every H we define a family $\text{Factors}(H)$ that can be obtained by recursively decomposing H into some "simpler" graphs. Then, in Chapter 6, we show that, for certain computational applications, is enough to focus on target graphs that cannot be further decomposed.

Assume first that H is a bipartite graph, in particular, it has no loops. We fix its bipartition classes X and Y.

Definition 5.2.1 (Bipartite decomposition). For a connected bipartite graph H, a partition of $V(H)$ into an ordered triple of sets (D, N, R) is a *bipartite decomposition* if the following conditions are satisfied.

(B1) N is non-empty and separates D and R,

(B2) $|D \cap X| \geqslant 2$ or $|D \cap Y| \geqslant 2$,

(B3) N induces a biclique in H,

(B4) $D \cap X$ is complete to $N \cap Y$, and $D \cap Y$ is complete to $N \cap X$.

If a connected bipartite graph H admits a bipartite decomposition, then it is *decomposable*, otherwise it is *undecomposable*. If H is disconnected, H is decomposable if any of its connected components is decomposable. If (D, N, R) is then a decomposition of some connected component H' of H, then $(D, N, R \cup (V(H) \setminus V(H')))$ is a decomposition of H.

Let $\delta = (D, N, R)$ be a bipartite decomposition of H. We define graphs H_1 and H_2 as follows: the graph H_1 is the subgraph of H induced by the set D. The graph H_2 is obtained in the following way. For $Z \in \{X, Y\}$, if $D \cap Z$ is non-empty, then we contract it to a vertex d_Z. If there is at least one edge between the sets $D \cap X$ and $D \cap Y$, we add the edge $d_X d_Y$. We call the pair (H_1, H_2) the *factorization of H with respect to δ*, and define its *factors* to be H_1 and H_2. We denote this by $\text{Factors}(H, \delta) = \{H_1, H_2\}$.

Now we show that when we decompose a graph that is not the complement of a circular-arc graph, then at least one of its factors is also not the complement of a circular-arc graph. For the proof, we will require yet another equivalent characterization of bipartite graphs H, whose complement is a circular-arc graph. Let X, Y be the bipartition classes of H and consider a circle C with two specified, distinct points p and q. A *co-circular-arc representation* of H is mapping of $V(H)$ to the closed arcs of the circle C, such that (i) each arc corresponding to a vertex from X contains p but not q, (ii) each arc corresponding to a vertex from Y contains q but not p, and (iii) the vertices are adjacent if and only if their corresponding arcs are disjoint. It is known that the complement of a bipartite graph H is a circular-arc graph if and only if H admits a co-circular-arc representation [64, 112].

Lemma 5.2.2. *Let H be a bipartite graph that admits a decomposition (D, N, R) with factors H_1, H_2. If both H_1 and H_2 are complements of circular-arc graphs, so is H.*

Proof. We show that we can construct a co-circular-arc representation of H from the representations of H_1 and H_2. Fix some representations of H_1 and H_2 on a circle C with two specified, distinct points p and q. For every $v \in V(H_1) \cup V(H_2)$ denote by $\widetilde{v}$ the arc corresponding to v.

Consider the representation of H_2. Clearly, at least one of the vertices $\widetilde{d_X}$ and $\widetilde{d_Y}$ exists, so without loss of generality assume that $\widetilde{d_X} \in V(H_2)$ and $p \in \widetilde{d_X}$. Observe that

(i) since N is a biclique, there must be two (open) arcs, a'_E and a'_W, that are disjoint with every $\widetilde{v}$ such that $v \in N$. We denote by a'_N and a'_S the two (closed) arcs such that $\{a'_N, a'_S, a'_E, a'_W\}$ is the partition of the circle. Clearly, (i) every $\widetilde{v}$ such that $v \in N$ is contained in a'_N or a'_S. On the other hand, (ii) since N is a separator, every $\widetilde{u}$ that contains p (resp. q) such that $u \in R$ must intersect $\widetilde{d_Y}$ (resp. $\widetilde{d_X}$). Moreover, (iii) since there are all edges between one bipartition class of N and d_X, and the other bipartition class of N and d_Y, the arc $\widetilde{d_X}$ (resp. $\widetilde{d_Y}$) cannot intersect any $\widetilde{v}$ that contains q (resp. p) such that $v \in N$.

First, consider the case in which $\widetilde{d_X}$ and $\widetilde{d_Y}$ intersect, i.e., the graph H_1 has no edges. In such situation, we take the representation of H_2, remove $\widetilde{d_X}$ and $\widetilde{d_Y}$, and represent each vertex v of D by a copy of $\widetilde{d_X}$ or $\widetilde{d_Y}$, depending on to which bipartition class of H_1 vertex v belongs. By (ii), every $\widetilde{w}$ such that $w \in D$ intersects every $\widetilde{u}$ such that $u \in R$. By (iii) every $\widetilde{w}$ that contains p and such that $w \in D$ cannot intersect any $\widetilde{v}$ that contains q such that $v \in N$. Analogously, every $\widetilde{w}$ that contains q such that $w \in D$ cannot intersect any $\widetilde{v}$ that contains p such that $v \in N$.

Therefore, we can assume that $\widetilde{d_X}$ and $\widetilde{d_Y}$ are disjoint. Now consider a representation of H_1. Note that in such a representation there exists a (closed) arc, a''_N (resp. a''_S), such that $p \in a''_N$ (resp. $q \in a''_S$) and a''_N is contained in all arcs that contain p (resp. a''_S is contained in arcs containing q). We denote by a''_W and a''_E the two (open) arcs such that $\{a''_N, a''_S, a''_E, a''_W\}$ is a partition of the circle.

We claim that we can obtain a representation of H from the representations of H_1 and H_2 by identifying the arcs $a'_N \cup \widetilde{d_X}, a'_S \cup \widetilde{d_Y}, a'_W \setminus (\widetilde{d_X} \cup \widetilde{d_Y}), a'_E \setminus (\widetilde{d_X} \cup \widetilde{d_Y})$, respectively, with $a''_N, a''_S, a''_W, a''_E$ (see Figure 5.2). Note that the intersections between the arcs corresponding to the vertices of $N \cup R$ did not change, so we only need to argue that each $\widetilde{w}$ that contains p (resp. q) such that $w \in D$ (a) intersects every $\widetilde{u}$ that contains q (resp. p) and $u \in R$, and (b) is disjoint with each $\widetilde{v}$ that contains q (resp. p) and $v \in N$.

Consider some $u \in R$ and $w \in D$ such that $q \in \widetilde{u}, p \in \widetilde{w}$. To see (a), note that $\widetilde{w}$ contains $a''_N \cup \widetilde{d_X}$ and by (ii) $\widetilde{u}$ is intersecting $\widetilde{d_X}$, so $\widetilde{w}$ must intersect $\widetilde{u}$. For (b) note that by (i) $\widetilde{u}$ is contained in $a''_S \cup d_Y$, and by (iii) $\widetilde{w}$ must be contained in the remaining part of the cycle, they are not intersecting, and this concludes the proof. □

Next, we generalize the notion of a decomposition for general target graphs. We define the following three types of decompositions of a graph H (see Figure 5.3). Note that

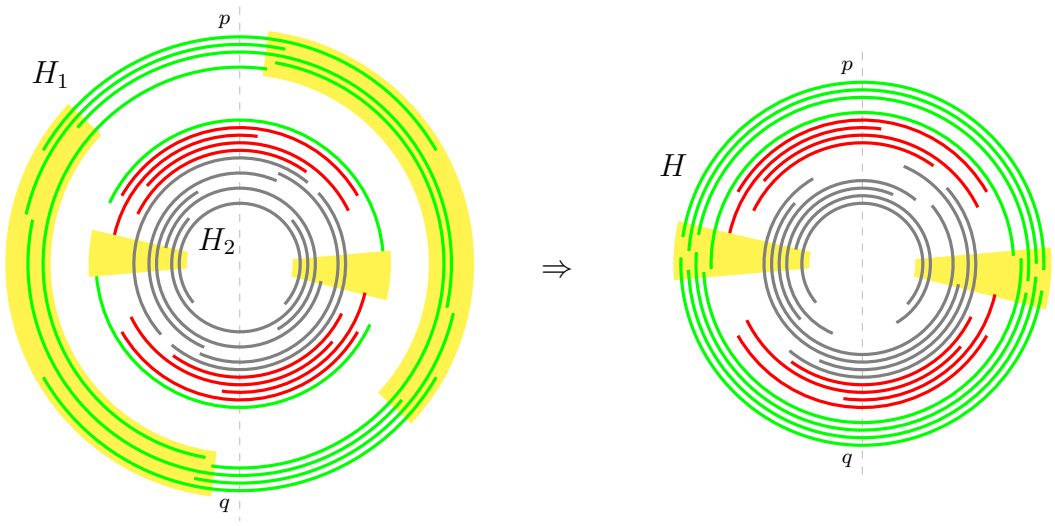


Figure 5.2: Assume that H is bipartite and has a decomposition (D, N, R) with factors H_1, H_2, which are both complements of circular-arc graphs. In the representation of H_2 (left, interior), arcs corresponding to sets $\{d_X, d_Y\}$, N, and R are indicated respectively by green, red, and gray color. Arcs a''_W and a''_E in the representation of H_1, $a'_W - (d_X \cup d_Y)$ and $a'_E - (d_X \cup d_Y)$ in the representation of H_2 (left) and the result of identifying them (right) are denoted by yellow. Intuitively, we can obtain a representation of H by "squeezing" the non-trivial part of the representation of H_1 in the yellow area.

unless stated explicitly, we do not insist that any of the defined sets is non-empty. The high-level idea is to define decompositions of H so that they will correspond to bipartite decompositions of H^*.

Definition 5.2.3 (F-decomposition). For a connected graph H, a partition of $V(H)$ into an ordered triple of sets (F, K, Z) is an *F-decomposition* if the following conditions are satisfied (see Figure 5.3, left).

(F1) K is non-empty and separates F and Z,

(F2) $|F| \geqslant 2$,

(F3) K induces a reflexive clique,

(F4) F is complete to K.

Definition 5.2.4 (BP-decomposition). For a connected graph H, a partition of $V(H)$ into an ordered five-tuple of sets (B, P, M, K, Z) is a *BP-decomposition* if the following conditions are satisfied (see Figure 5.3, middle).

(BP1) $M \cup K$ is non-empty, and there are no edges between $(P \cup B)$ and Z,

(BP2) $|P| \geqslant 2$ or $|B| \geqslant 2$,

(BP3) $K \cup P$ induces a reflexive clique and B is an independent set,

(BP4) M is complete to $P \cup K$ and B is complete to K,

(BP5) B is non-adjacent to M.

Definition 5.2.5 (BB-decomposition). For a connected graph H, a partition of $V(H)$ into an ordered six-tuple of sets $(B_1, B_2, K, M_1, M_2, Z)$ is a *BB-decomposition* if the following conditions are satisfied (see Figure 5.3, right).

(BB1) $K \cup M_1 \cup M_2$ is non-empty, and there are no edges between $(B_1 \cup B_2)$ and Z,

(BB2) $|B_1| \geqslant 2$ or $|B_2| \geqslant 2$,

(BB3) K induces a reflexive clique and each of B_1, B_2 is an independent set,

(BB4) K is complete to $M_1 \cup M_2 \cup B_1 \cup B_2$, and M_2 is complete to $M_1 \cup B_1$, and M_1 is complete to B_2,

(BB5) B_1 is non-adjacent to M_1 and B_2 is non-adjacent to M_2.

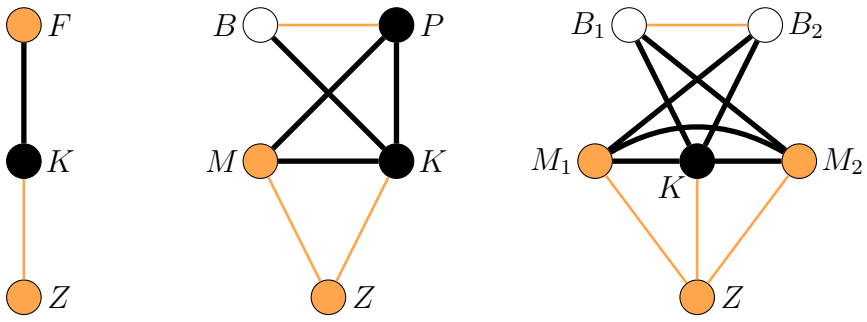


Figure 5.3: A schematic view of an F-decomposition (left), a BP-decomposition (middle), and a BB-decomposition of H (right). Disks correspond to sets of vertices: white ones depict independent sets, black ones depict reflexive cliques, and orange ones depict arbitrary subgraphs. Similarly, thick black lines indicate that all possible edges between two sets exist, and thin orange lines depict edges that might exist, but do not have to. The lack of a line means that there are no edges between two sets.

Observe that a graph H can have more than one type of decomposition: for example, if $(B, P, \emptyset, K, Z)$ is a BP-decomposition of H, then $(B \cup P, K, Z)$ is an F-decomposition of H. We say that a connected graph H is *decomposable* if it admits a Γ-decomposition, for some

$\Gamma \in \{F, BP, BB\}$, and *undecomposable* otherwise. Note that a bipartite decomposition is a special case of a BB-decomposition (with $K = \emptyset$ and M_1, M_2 being independent sets). A disconnected graph H is decomposable if and only if there is a connected component H' of H that is decomposable, and to obtain a decomposition of H from a decomposition δ of H', we simply add all the vertices of $V(H) \setminus V(H')$ to the set Z (regardless of the type of δ).

For each kind of decomposition, we define graphs H_1, H_2, as follows:

for an F-decomposition: $H_1 = H[F]$ and H_2 is obtained from H by contracting F to a vertex f. It has a loop if and only if F is not an independent set.

for a BP-decomposition: $H_1 = H[B \cup P]$ and H_2 is obtained from H by contracting P and B respectively (if they are non-empty), to vertices p and b, such that p has a loop and b does not. Also, $pb \in E(H_2)$ if and only if there is any edge between P and B in H.

for a BB-decomposition: $H_1 = H[B_1 \cup B_2]$ and H_2 is obtained from H by contracting B_1 and B_2 respectively (if they are non-empty), to vertices b_1 and b_2 (without loops). Also, $b_1b_2 \in E(H_2)$ if and only if there is any edge between B_1 and B_2 in H.

For a connected graph H, and its Γ-decomposition δ, where $\Gamma \in \{F, BP, BB\}$, we define the *factorization of H with respect to δ* to be the pair (H_1, H_2) and define $\text{Factors}(H, \delta) = \{H_1, H_2\}$. Note that if a graph admits a bipartite decomposition, which is a special case of BB-decomposition, these notions coincide.

Note that the construction of H_2 implies that if H is connected, so is H_2. Also, in a BP-decomposition, a BB-decomposition, and an F-decomposition when F is an independent set or contains a vertex with a loop, the elements of $\text{Factors}(H, \delta)$ are always induced subgraphs of H. Indeed, we can equivalently obtain H_2 by removing certain vertices from H. In the case of an F-decomposition when F contains at least one edge and has only vertices without loops, H_2 is not an induced subgraph of H. Then we can equivalently define H_2 as the graph obtained by removing from F all but two vertices that are adjacent to each other, and then replacing them with a vertex with a loop.

Let us prove that the three types of decompositions defined above correspond to bipartite decompositions of the associated bipartite graph H^*. Here, for any $W \subseteq V(H)$, we define two subsets of $V(H^*)$ as follows: $W' := \{x' \mid x \in W\}$ and $W'' := \{x'' \mid x \in W\}$. Note that a subset $W \subseteq V(H)$ induces a reflexive clique in H if and only if $W' \cup W''$ induces a biclique in H^*.

Lemma 5.2.6. *Let H be a connected graph that admits a Γ-decomposition δ, for $\Gamma \in \{F, BP, BB\}$ with factorization (H_1, H_2). Then H^* is decomposable. Moreover, if H is a non-bi-arc graph, then at least one of H_1, H_2 is a non-bi-arc graph.*

Proof. If H is bipartite, then, by Theorem 5.0.1, H^* consists of two isomorphic copies of H. Since a bipartite decomposition is a special case of the BB-decomposition (with $K = \emptyset$ and M_1, M_2 being independent sets), the theorem holds for bipartite graphs, as H^* contains a decomposable component.

Thus we assume that H is non-bipartite, and consider three cases, depending on the type of a decomposition of H.

If H has an F-decomposition $\delta = (F, K, Z)$, define D, N, R as (see Figure 5.4 a)):

$$D := F' \cup F'', \quad N := K' \cup K'', \quad R := Z' \cup Z''.$$

The fact that (D, N, R) is a bipartite decomposition of H^* follows directly from the definition of a F-decomposition (recall Definition 5.2.3): each property in Definition 5.2.1 follows from the corresponding property in Definition 5.2.3.

Now let (H_1, H_2) and $((H^*)_1, (H^*)_2)$ be factorizations of, respectively H and H^* with respect to the decompositions considered above. If H is non-bi-arc, then by Theorem 5.1.6, H^* is not the complement of a circular-arc graph. Then Lemma 5.2.2 implies that at least one of $(H^*)_1, (H^*)_2$ is not the complement of a circular arc graph. However, note that $(H^*)_1 = (H_1)^*$ and $(H^*)_2 = (H_2)^*$, hence again by Lemma 5.2.2, at least one of H_1, H_2 is not a bi-arc graph.

If H has a BP-decomposition $\delta = (B, P, M, K, Z)$, then (D, N, R) is as follows (see Figure 5.4 b)):

$$\begin{aligned} D :=& B' \cup P'', \\ N :=& K' \cup M' \cup P' \cup K'', \\ R :=& Z' \cup Z'' \cup M'' \cup B''. \end{aligned}$$

If H has a BB-decomposition $\delta = (B_1, B_2, K, M_1, M_2, Z)$, then we define (D, N, R) as

follows (see Figure 5.4 c)):

$$\begin{aligned} D :=& B'_1 \cup B''_2, \\ N :=& K' \cup M'_1 \cup K'' \cup M''_2, \\ R :=& Z' \cup M'_2 \cup B'_2 \cup Z'' \cup M''_1 \cup B''_1. \end{aligned}$$

Again, it is straightforward to verify that in both cases $\delta_1 = (D, N, R)$ is a bipartite decomposition of H^*.

Let (H_1, H_2) and $((H^*)_1, (H^*)_2)$ be factorizations of, respectively H and H^* with respect to δ and δ_1. Let $(S_1, S_2) = (B, P)$ if δ_1 is a BP-decomposition, and $(S_1, S_2) = (B_1, B_2)$ if δ_1 is a BB-decomposition. Suppose that $S_1 \neq \emptyset$ and $S_2 \neq \emptyset$. Let s_1 and s_2 be, respectively, vertices of H_2, that were obtained by contracting S_1 and S_2. We note that if s'_1 and s''_2 are vertices of $(H^*)_2$ that were obtained by contracting, respectively, S'_1 and S''_2, then we can define a bipartite decomposition $\delta_2 = (S''_1 \cup S'_2, \hat{N}, \hat{R})$ of $(H^*)_2$, where

$$\hat{N} = \begin{cases} K'' \cup M'' \cup \{p''\} \cup K' & \text{if } \delta_1 \text{ is a } BP\text{-decomposition}, \\ K'' \cup M''_1 \cup K' \cup M'_2 & \text{if } \delta_1 \text{ is a } BB\text{-decomposition}. \end{cases}$$

and

$$\hat{R} = \begin{cases} Z'' \cup Z' \cup M' \cup \{b'\} & \text{if } \delta_1 \text{ is a } BP\text{-decomposition}, \\ Z'' \cup M''_2 \cup Z' \cup M'_1 \cup \{b'_1, b''_2\} & \text{if } \delta_1 \text{ is a } BB\text{-decomposition}. \end{cases}$$

Denote by $(((H^*)_2)_1, ((H^*)_2)_2)$ the factorization of $(H^*)_2$ with respect to δ_2.

Observe that $((H^*)_2)_1 = H^*[S''_1 \cup S'_2] = H^*[S'_1 \cup S''_2] = (H^*)_1$, and this is an induced subgraph of $(H_1)^* = (H[S_1 \cup S_2])^* = H^*[S'_1 \cup S''_1 \cup S'_2 \cup S''_2]$. If now s''_1 and s'_2 are vertices of $((H^*)_2)_2$ that were obtained by contracting, respectively, S''_1 and S'_2, then note that $\{s''_1, s'_2\} \cup \hat{N} \cup \hat{R}$ is the set of vertices of $((H^*)_2)_2$ and moreover, $((H^*)_2)_2 = (H_2)^*$.

Hence, by Theorem 5.1.6, if H is a non-bi-arc graph, then H^* is not a complement of circular-arc graph, and Lemma 5.2.2 implies that either $((H^*)_2)_1 = (H^*)_1$ or $((H^*)_2)_2 = (H_2)^*$ is not a complement of circular-arc graph. Since the first one is an induced subgraph of $(H_1)^*$, either $(H_1)^*$ or $(H_2)^*$ is not a complement of circular-arc graph. By Theorem 5.1.6, at least one of H_1, H_2 is a non-bi-arc graph.

The cases when either S_1 or S_2 is empty are analogous, the only difference is that some of vertices of the second factors or each decomposition (e.g., s_1 or s_2) just do not exist. This concludes the proof. □

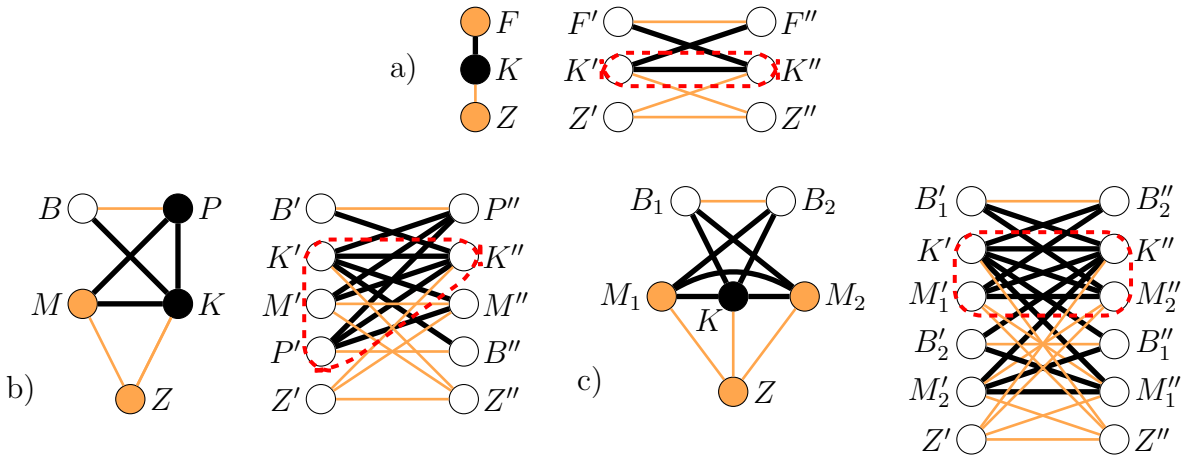


Figure 5.4: Decompositions of a graph H (left) and their corresponding bipartite decompositions (D, N, R) of H^* (right): a) an F-decomposition, b) a BP-decomposition and c) a B-decomposition. Dashed lines mark the set N.

We can prove the converse statement, assuming that H is not a *strong split graph*. A graph H is a strong split graph if its set of vertices can be partitioned into sets B and P such that B is an independent set and P is a reflexive clique. We call such a pair (B, P) a *split partition* of H.

Lemma 5.2.7. *Let H be a connected non-bi-arc graph. If H is not a strong split graph and H is undecomposable, then H^* is undecomposable.*

Proof. Similarly like in the previous proof, if H is bipartite, the statement holds by Theorem 5.0.1, and the fact that a bipartite decomposition is a special case of the B-decomposition. Hence we can assume that H is non-bipartite. In particular this means that H^* is connected.

Suppose for contradiction that (D, N, R) is a bipartite decomposition of H^*. Recall that $N \neq \emptyset$ and one bipartition class of D has at least two elements. We aim to show that H has a Γ-decomposition for $\Gamma \in \{F, BP, BB\}$. We partition the vertices of H into nine sets as follows (see Figure 5.5):

$V_{DD} := \{x \mid x' \in D, x'' \in D\}$, $V_{ND} := \{x \mid x' \in N, x'' \in D\}$, $V_{RD} := \{x \mid x' \in R, x'' \in D\}$,
$V_{DN} := \{x \mid x' \in D, x'' \in N\}$, $V_{NN} := \{x \mid x' \in N, x'' \in N\}$, $V_{RN} := \{x \mid x' \in R, x'' \in N\}$,
$V_{DR} := \{x \mid x' \in D, x'' \in R\}$, $V_{NR} := \{x \mid x' \in N, x'' \in R\}$, $V_{RR} := \{x \mid x' \in R, x'' \in R\}$.

Clearly, from the definition of the bipartite decomposition it follows that some pairs of these sets cannot be both non-empty (e.g. V_{DD} and V_{RN}, because V'_{DD} must be complete to V''_{RN} and in the same time V''_{DD} is non-adjacent to V'_{RN}). Observe that $V_{NR} \cup V_{ND} \cup V_{NN}$ is complete to $V_{ND} \cup V_{RD} \cup V_{DD} \cup V_{DN} \cup V_{RN} \cup V_{NN}$ and $V_{DN} \cup V_{RN} \cup V_{NN}$ is complete to $V_{DN} \cup V_{DR} \cup V_{DD} \cup V_{NR} \cup V_{ND} \cup V_{NN}$. Also, $V_{DR} \cup V_{DN} \cup V_{DD}$ is non-adjacent to $V_{NR} \cup V_{DR} \cup V_{RR}$ and $V_{RD} \cup V_{ND} \cup V_{DD}$ is non-adjacent to $V_{RN} \cup V_{RD} \cup V_{RR}$. In particular, it implies that V_{ND}, V_{DN}, and V_{NN} must be reflexive cliques and V_{DR}, V_{RD} are independent sets. Finally, at least one of the sets $V_{DR} \cup V_{DN} \cup V_{DD}$ and $V_{RD} \cup V_{ND} \cup V_{DD}$ has at least two vertices and at least one of the sets $V_{NR} \cup V_{ND} \cup V_{NN}$ and $V_{DN} \cup V_{RN} \cup V_{NN}$ is non-empty.

Case 1: $V_{DD} \neq \emptyset$**.** It implies that $V_{NR}, V_{RN} = \emptyset$. If $V_{NN} \neq \emptyset$, then it is straightforward to observe that $(V_{DD} \cup V_{DN} \cup V_{ND} \cup V_{DR} \cup V_{RD}, V_{NN}, V_{RR})$ is an F-decomposition of H. Thus assume that $V_{NN} = \emptyset$ and observe it implies that $V_{RR} = \emptyset$, as we assumed that H is connected. Moreover, since $V_{NR} \cup V_{RN} \cup V_{DN} \cup V_{ND} \cup V_{NN} \neq \emptyset$, at least one of V_{DN}, V_{ND} is non-empty. Assume that $V_{DN} \neq \emptyset$, as the other case is symmetric. Recall that this means that $V_{DR} = \emptyset$.

If $V_{ND} \neq \emptyset$, then $V_{RD} = \emptyset$ and $(V_{DD} \cup V_{ND}, V_{DN}, \emptyset)$ is an F-decomposition. So let $V_{ND} = \emptyset$. If $|V_{DD}| \geqslant 2$, then (V_{DD}, V_{DN}, V_{RD}) is an F-decomposition. If $|V_{RD}| \geqslant 2$ or $|V_{DN}| \geqslant 2$, then $(V_{RD}, V_{DN}, V_{DD}, \emptyset, \emptyset)$ is a BP-decomposition. So in the last case we have $|V_{DD}| = |V_{DN}| = 1$ and $|V_{RD}| \leqslant 1$. It is easy to verify that then H is a bi-arc graph (or, equivalently, H^* does not contain an induced cycle on at least 6 vertices or an edge asteroid). This contradicts our assumption on H.

Case 2: $V_{DD} = \emptyset$**.** We consider three subcases: either $V_{DR}, V_{RD} \neq \emptyset$, or $V_{DR} = \emptyset$ and $V_{RD} \neq \emptyset$ (the case $V_{DR} \neq \emptyset$ and $V_{RD} = \emptyset$ is symmetric), or $V_{DR}, V_{RD} = \emptyset$. The first case implies that $V_{DN}, V_{ND} = \emptyset$, so we immediately obtain a BB-decomposition $(V_{DR}, V_{RD}, V_{NN}, V_{NR}, V_{RN}, V_{RR})$.

In the second one we have $V_{ND} = \emptyset$. If $V_{DN} \neq \emptyset$, then $V_{NR} = \emptyset$. If additionally $V_{RN} \cup V_{NN} \neq \emptyset$, then there exists a BP-decomposition $(V_{RD}, V_{DN}, V_{RN}, V_{NN}, V_{RR})$. On the other hand, if $V_{RN} \cup V_{NN} = \emptyset$, then $V_{RR} = \emptyset$ by connectivity of H. Thus the only non-empty sets are V_{DN} and V_{RD}, so H must be a strong split graph, a contradiction. Finally, if $V_{DN} = \emptyset$, then H admits a BB-decomposition $(\emptyset, V_{RD}, V_{NN}, V_{NR}, V_{RN}, V_{RR})$.

So let us assume that $V_{DR}, V_{RD} = \emptyset$. We consider further subcases: either $V_{NR}, V_{RN} \neq \emptyset$, or $V_{NR} = \emptyset$ and $V_{RN} \neq \emptyset$ (the other case, i.e., $V_{NR} \neq \emptyset$ and $V_{RN} = \emptyset$, is symmetric), or $V_{NR}, V_{RN} = \emptyset$. Note that if $V_{NR}, V_{RN} \neq \emptyset$, then $V_{DN}, V_{ND} = \emptyset$, so $D = \emptyset$, which is a contradiction with (D, N, R) being a bipartite decomposition of H^*. If $V_{NR} = \emptyset$ and $V_{RN} \neq \emptyset$, then $V_{ND} = \emptyset$ and $|V_{DN}| \geqslant 2$ and we have a BP-decomposition $(\emptyset, V_{DN}, V_{RN}, V_{NN}, V_{RR})$. Lastly, when $V_{NR}, V_{RN} = \emptyset$, then either $|V_{DN}| \geqslant 2$ or $|V_{ND}| \geqslant 2$. By symmetry, we can assume that the first case holds. Then, either $(V_{DN}, V_{ND} \cup V_{NN}, V_{RR})$ is an F-decomposition, or $V_{ND} \cup V_{NN} = \emptyset$, then, by connectivity of H, $V_{RR} = \emptyset$ and H is just a reflexive clique induced by V_{DN}. However, such a graph is a bi-arc graph, a contradiction. □

We note that the assumption on H not being a split graph is essential, as it may happen that a strong split graph H is undecomposable when H^* admits a decomposition, see Figure 5.6 for an example.

Now, with a notion of a decomposition in hand, we can define a generalized set of factors of H. Intuitively, such a set consists of two kinds of graphs, bi-arc graphs and undecomposable graphs, that are obtained by recursively decomposing H, factors of H, factors of factors of H, etc.

Formally, for a graph H we recursively construct a binary tree T as follows. We start with a single node H, and as long as T contains a leaf node H' that is non-bi-arc and admits a decomposition δ with factorization (H'_1, H'_2), we add H'_1 and H'_2 as the children of H'. Hence, all leaves of T are either bi-arc or undecomposable. We call T a *factorization tree* of H. By $\mathrm{Factors}_T(H)$ we denote the set of leaves of T. If the tree T is clear from the context, we omit the subscript and write $\mathrm{Factors}(H)$. Clearly, if H is undecomposable, then $\mathrm{Factors}(H) = \{H\}$.

As the last proof in this section we show that even if H' is not an induced subgraph of H, the associated bipartite graph H'^* is still an induced subgraph of H^*.

Lemma 5.2.8. *If $H' \in \mathrm{Factors}_T(H)$ for some T, then H'^* is an induced subgraph of H^*.*

Proof. Recall that by the definitions of decompositions, H' was obtained from H by a sequence of two types of operations:

(O1) removing some vertices or, equivalently, taking an induced subgraph (BP-, BB-, and F-decomposition when the set F contains a loop or is independent),

(O2) removing some vertices and then contracting two adjacent vertices a, b, such that $N(a) \setminus \{b\} = N(b) \setminus \{a\}$ and none of a, b has a loop, to a vertex c with a loop; vertices a and b are removed from the graph and the new vertex c becomes adjacent to all vertices in $N(a) \setminus \{b\} = N(b) \setminus \{a\}$ (F-decomposition when F has no loop and contains at least one edge, see Figure 5.7).

Observe that if only the first type of operation was applied, then H' is an induced subgraph of H, which implies that H'^* is an induced subgraph of H^*. Let us analyze the case when some operations of the second type were applied as well.

In this case H' might not be an induced subgraph of H, but each newly created vertex c uniquely corresponds to two adjacent vertices, i.e., a, b. Moreover, when the operation was applied, a and b had the same neighborhoods in the current graph, except of being adjacent to each other. Note that both a and b are vertices of H: they do not have loops, while we only add vertices with loops. Furthermore, when c is created, a and b are removed from the graph, so each vertex is used in the operation of the second type at most once.

Let us consider H'^*. Note that the edge ab of H corresponds to two edges $a'b''$ and $a''b'$ in H^*. Moreover, we can map the vertex c' to a' and c'' to b''. Since $N(a)\setminus\{b\} = N(b)\setminus\{a\}$, when the operation was applied, the associated bipartite graph of the obtained graph is indeed an induced subgraph of H^*, and the mapping mentioned above defines the isomorphism from H'^* to a subgraph of H^*. □

5.3 Incomparable sets

A set $S \subseteq V(H)$ is *sound* if (i) S is contained in one connected component of H, and (ii) if H is bipartite, then S is contained in one bipartition class of H. By $i(H)$ we denote the size of maximum sound incomparable set in H. We observe the following.

Observation 5.3.1. *If H is a bipartite graph, which is not a complement of a circular-arc graph, then $i(H) \geqslant 3$. In particular, each bipartition class of H contains at least 3 vertices.*

Proof. From Theorem 5.1.6 it follows that H contains an obstruction, which is either an induced C_6, an induced C_8, or an asteroidal subgraph. Observe that all vertices from one bipartition class of C_6 or C_8 form an incomparable set of size at least 3. On the other hand, note that in a special edge asteroid $\{u_0v_0, u_1v_1, u_{k+1}v_{k+1}\}$ is an induced matching, so their appropriate endvertices form the desired incomparable set. □

We can now formally define the parameter $i^*(H)$. For a bipartite graph H let

$$i^*(H) := \max\{i(H') \colon H' \text{ is an induced subgraph of } H \text{ that is undecomposable, connected and whose complement is not a circular-arc graph}\}$$

if H is not a complement of a circular arc graph, and $i^*(H) = 1$ otherwise. For a general graph H, we define

$$i^*(H) := i^*(H^*).$$

Recall from Theorem 5.0.1 that if H is bipartite, then H^* consists of two disjoint copies of H, so clearly $i^*(H) = i^*(H^*)$ and i^* is well-defined for bipartite graphs. Moreover, note that Theorem 5.1.6 implies that if H is a bi-arc graph, then $i^*(H) = 1$.

The following observation is straightforward.

Observation 5.3.2. *Let H' be an induced subgraph of H. Then $i^*(H') \leqslant i^*(H)$.* □

Observation 5.3.2 implies that a similar fact can be shown for the factors of H.

Observation 5.3.3. *Let H be a graph and let $H' \in \text{Factors}_T(H)$ for some T. Then $i^*(H') \leqslant i^*(H)$.*

Proof. By the definition of i^* and Observation 5.3.2, $i^*(H') = i^*(H'^*) \leqslant i^*(H^*) = i^*(H)$.□

Recall that if H is a strong split graph, the correspondence between undecomposability of H and H^* does not necessarily hold. Because of that, in some arguments, we need to treat strong split graphs separately. For that, we summarize the important properties these graphs in the following lemma.

Lemma 5.3.4. *Let H be a connected, undecomposable, non-bi-arc, strong split graph with partition (B, P). Then the following holds.*

1. $i^*(H) = i(H^*)$,
2. $|B|, |P| \geqslant 3$,
3. *the bipartite graph H', obtained from H by removing all edges with both endvertices in P (including loops), is connected, undecomposable, and not a complement of circular-arc graph.*

Proof. Let (B, P) be the partition of H. Since we always have $i^*(H^*) \leqslant i(H^*)$ we only need to show that $i^*(H^*) \geqslant i(H^*)$.

Since H is non-bi-arc, $P, B \neq \emptyset$, so in particular H is not bipartite (as it contains a vertex with a loop). By Theorem 5.0.1, H^* is connected. Moreover, by Theorem 5.1.6, H^* is not the complement of a circular-arc graph. Hence, by Observation 5.3.1, each bipartition class of H^* contains at least three vertices, i.e., $|B' \cup P''| = |B'' \cup P'| \geqslant 3$.

We observe that $H^*[B' \cup P'']$ is isomorphic to H', so in particular, H' is an induced subgraph of H^*. We claim that $i^*(H^*) \geqslant i(H') = i(H^*)$.

First, we prove that H' is not the complement of circular-arc graph. Note that H^* always admits a bipartite decomposition $\delta_1 = (B' \cup P'', P', B'')$. Let (H_1, H_2) be the factorization of δ_1, recall that H_1 is isomorphic to H'. Note that H_2 is isomorphic to $H^*[B'' \cup P']$ plus two adjacent vertices $d_{B'}, d_{P''}$ such that $d_{P''}$ is additionally complete to P'. Hence, H_2 can be further decomposed into $\delta_2 = (B'' \cup P', \{d_{P''}\}, \{d_{B'}\})$ with factorization $(\hat{H}_1, \hat{H}_2)$. Again, note that $\hat{H}_1$ and $\hat{H}_2$ are isomorphic, respectively, to H', and a path P_4 on 4 vertices. Clearly, P_4 is the complement of a circular-arc graph, as P_4 is a circular-arc graph and $\overline{P_4} = P_4$. If now H' is also the complement of a circular arc graph, by Lemma 5.2.2 so is H_2, and then so is H^*. However, since H is non-bi-arc, this is a contradiction with Theorem 5.1.6. Hence, H' is not a complement of circular-arc graph. In particular, by Observation 5.3.1, each bipartition class of H', i.e., B or P, contains at least three vertices (that proves the second item).

Now we observe that H' must be connected. Assume otherwise, and note that since H' is bipartite and, by the previous paragraph, not the complement of a circular-arc graph, by Theorem 5.1.6, there is a connected component H'' of H' that contains an obstruction. Again, by Observation 5.3.1, each bipartition class of H'' contains at least three vertices.

Since H is connected, H'' contains some $p \in P$. If $|B - V(H'')| \geqslant 2$ or $|P - V(H'')| \geqslant 2$, we note that $(B - V(H''), P - V(H''), P \cap V(H''), \emptyset, V(H'') \cap B)$ is a BP-decomposition of H. Otherwise, $(B \cap V(H''), P \cap V(H''), P \setminus V(H''), \emptyset, B \setminus V(H''))$ is a BP-decomposition of H. In both cases we reach a contradiction with H being undecomposable.

Summarizing, H' is a connected, induced subgraph of H^* whose complement is not a circular-arc graph.

We prove the equality $i(H') = i(H^*)$. To see that $i(H') \geqslant i(H^*)$ consider a sound incomparable set S in H^*. Without loss of generality, $S \subseteq B' \cup P'$. However, S cannot contain $b' \in B'$ and $p' \in P'$, since $N_{H^*}(b') \subseteq N_{H^*}(p')$. Hence, each incomparable set in H^* that is contained in one bipartition class, corresponds to a sound incomparable set in H'. This means that $i(H') \geqslant i(H^*)$, thus (as the converse inequality follows from Observation 5.3.2) $i(H') = i(H^*)$.

To show that $i^*(H^*) \geqslant i(H')$, it is enough to prove that H' is a connected induced undecomposable subgraph of H^*, and not the complement of a circular-arc graph. It remains to show that H' is undecomposable. Assume otherwise and let (D, N, R) be a bipartite decomposition of H'. We observe that $(B \cap D, P \cap D, (B \cap N) \cup (P \cap R), P \cap N, B \cap R)$ is a BP-decomposition of H, a contradiction. This concludes the proof. $\square$

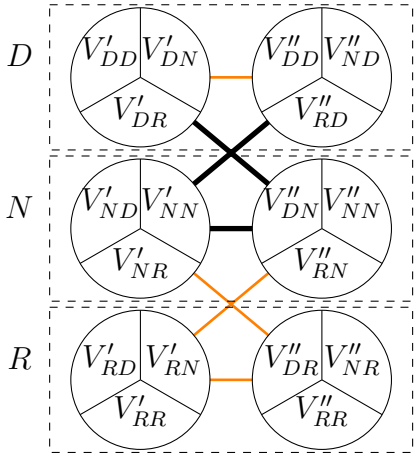


Figure 5.5: Schematic definition of sets in the proof of Lemma 5.2.7. E.g. V_{DN} is the set of those $x \in V(G)$, for which $x' \in D$ and $x'' \in N$. Author: Marta Piecyk.

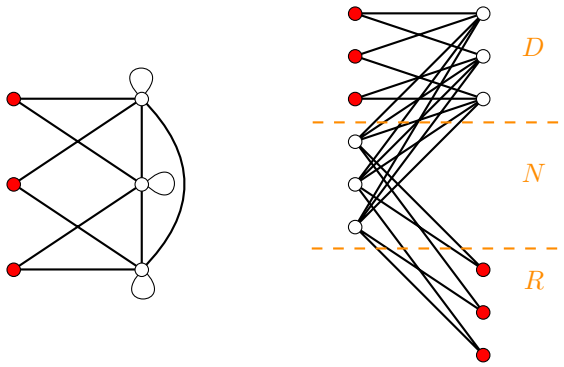


Figure 5.6: A strong split undecomposable graph H (left) and the graph H^* with a decomposition (right).

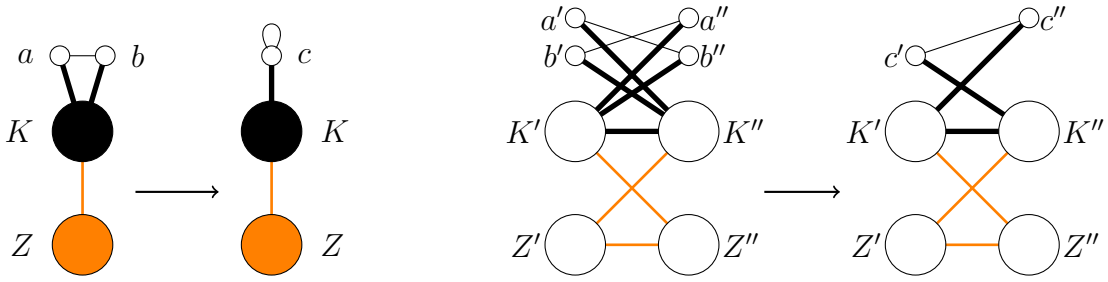


Figure 5.7: Operation (O2) applied to vertices a, b in H (left) and the corresponding operation applied to vertices a', a'', b', b'' in H^* (right). Author: Paweł Rzążewski.

Chapter 6

The list homomorphism problem parameterized by treewidth

In this chapter we provide the crucial tools to analyze the algorithmic aspects of the list homomorphism problem, and show how these tools can be used to derive Theorem 1.3.3.

First, we are going to show that the key to understand the complexity of the list homomorphism problem is the case when the target H is undecomposable (with respect to the decompositions defined previous chapter). Intuitively, we show that if we are able to decide whether $(G, L) \to H$ for any pair (G, L) and any undecomposable graph H, then we are able to solve the problem for every other graph H in roughly the same time.

Next, we show an analogue of Lemma 3.2.7 in the list regime and prove that we are able to simulate certain relations defined on $V(H)$, provided that H is undecomposable.

6.1 Decomposition lemmas

The main technical contribution of this section is the so-called decomposition theorem (Theorem 6.1.6). Intuitively speaking, for a target graph H, it asserts that if we have an algorithm to solve the list homomorphism problem for every H' that is either an induced subgraph or a factor of H, then we can solve the problem for H in roughly the same time. We are going to show that the lemma holds even if we assume that the graph H is not fixed, but given as part of the input. Formally, we consider the LHOM problem, that takes as an instance a triple (G, H, L) such that G and H are graphs, and $L : V(G) \to 2^{V(H)}$, and we ask whether $(G, L) \to H$.

First, we show that we can effectively compute whether a given graph H is undecomposable. For that, we need the following structural lemma.

Lemma 6.1.1. *Let H be a connected, bipartite undecomposable graph with bipartition classes X, Y, let $\{s, v\} \in X$ be incomparable, and let $t \in X$. Then there exist a vertex $q \in X$ and two pairs of walks:*

1. *$\mathcal{P}, \mathcal{P}' : s \to t$ and $\mathcal{Q}, \mathcal{Q}' : v \to q$, or*
2. *$\mathcal{P}, \mathcal{P}' : s \to q$ and $\mathcal{Q}, \mathcal{Q}' : v \to t$,*

such that $\mathcal{P}$ avoids $\mathcal{Q}$ and $\mathcal{Q}'$ avoids $\mathcal{P}'$. Moreover, if t is incomparable with at least one of s, v then $q = v$ in the first case and $q = s$ in the other.

Finally, given a bipartite graph H and $s, v, t \in X$, in time polynomial in $|V(H)|$ we can either find the desired walks or a decomposition of H.

Since the proof of Lemma 6.1.1 is technically involved, we postpone its proof to Section 6.3. For now, we assume that it holds, and show the following.

Theorem 6.1.2. *For a connected, non-bi-arc graph H, in time polynomial in $|V(H)|$ we can either find a decomposition of H, or correctly conclude that H is undecomposable.*

Proof. We consider two cases, depending on whether H is a strong split graph or not. Clearly, strong split graphs can be recognized in time polynomial in $|V(H)|$.

Hence, assume first that H is a strong split graph with partition (B, P). Lemma 5.3.4 asserts that $|B|, |P| \geqslant 3$. Let H' be the graph defined as in Lemma 5.3.4, recall by the same lemma H' is not the complement of a circular-arc graph. Consider the following sufficient conditions for H to be decomposable.

(i) H' is disconnected. Then Lemma 5.3.4 implies that H is decomposable.

(ii) There are two vertices $v_1, v_2 \in B$ such that $N(v_1) = N(v_2)$. Then $(\{v_1, v_2\}, N(v_1, v_2), V(H) \setminus N[v_1, v_2])$ is an F-decomposition of H.

We claim the following.

Claim 6.1.2.1. *Graph H is undecomposable if and only if H' is undecomposable, and (i)-(ii) does not hold.*

Proof of Claim. The forward implication follows already from Lemma 5.3.4 and the above analysis. Hence assume that H' is undecomposable and (i),(ii) do not hold, but H is decomposable.

- If H has a BB-decomposition $(B_1, B_2, K, M_1, M_2, Z)$, then either there is an edge between B_1 and B_2 (and thus in B), a contradiction, or two vertices in $B \cap B_1$ or $B \cap B_2$ that satisfy (ii), also a contradiction.
- If H has a BP-decomposition $(\widehat{B}, \widehat{P}, \widehat{M}, \widehat{K}, \widehat{Z})$, then $(\widehat{B} \cup \widehat{P}, (\widehat{M} \cap B) \cup \widehat{K}, \widehat{Z} \cup (\widehat{M} \cap P))$ is a bipartite decomposition of H', a contradiction.
- If H has an F-decomposition (F, K, Z), and $|P \cap F| \geqslant 2$ or $|B \cap F| \geqslant 2$ then (F, K, Z) is also a bipartite decomposition of H' (with bipartition classes B and P, but note that $K \cap B = \emptyset$). Hence, as $|F| \geqslant 2$, F consists of precisely one loopless vertex b, and one vertex p with loop. Note that in this case we have $Z \subseteq B$, as otherwise $p \in F$ has a neighbor in Z. If $bp \notin E(H)$, then H' is not connected, as p is isolated, thus (i) holds, a contradiction. Otherwise $bp \in E(H)$ and $(K \cup Z \cup \{p\}, \{b\}, \emptyset)$ is a bipartite decomposition of H'.

That concludes the proof of claim. ⌟

Hence, after checking (i)-(ii) and constructing H' (both can be done in polynomial time), it is enough to check whether H' admits a bipartite decomposition.

On the other hand, if H is not a strong split graph, by Lemma 5.2.6 and Lemma 5.2.7, it is enough to check whether H^* admits a bipartite decomposition. Hence, to prove the lemma in both cases, it is sufficient to prove the following statement:

- ⋆ For a bipartite graph H that is not a complement of circular-arc graph, in time polynomial in $|V(H)|$ we can either find a decomposition of H, or correctly conclude that H is undecomposable.

Let us then assume that H is bipartite, with bipartition classes X and Y. Consider the following three-phase algorithm. If any of the calls results in finding a bipartite decomposition of H, we return it and terminate the execution of the algorithm. Otherwise, we return that H is undecomposable.

1. For every triple of vertices s, v, t in one bipartition class, such that s and v are incomparable, apply the algorithm from Lemma 6.1.1, in order to find the walks or a decomposition of H.

2. For every vertex v, let N'_v be the set of vertices with exactly the same neighborhood as v. Verify whether $(V(H) \setminus N'_v, N'_v, \emptyset)$ is a bipartite decomposition of H.

3. For every $x \in X$ and every $y \in Y$, let X_x (resp. Y_y) be the set of vertices of X (resp. Y), whose neighborhood is contained in the neighborhood of x (resp. y). Verify whether one of triples $(X_x \cup Y_y, N(X_x \cup Y_y), V(H) \setminus N[X_x \cup Y_y])$, $(X_x, N(X_x), V(H) \setminus N[X_x])$, or $(Y_y, N(Y_y), V(H) \setminus N[Y_y])$ is a bipartite decomposition of H.

Clearly the algorithm terminates after a polynomial number of steps: in each phase we iterate over a polynomial number of tuples of vertices, Lemma 6.1.1 yields a polynomial-time algorithm for finding a decomposition, and in polynomial time we can verify whether a given triple is a bipartite decomposition of H. Furthermore, it never finds a decomposition of an undecomposable graph. So let us assume that H has some decomposition, say (D, N, R), but our algorithm fails to find any decomposition.

We consider three cases, corresponding to the three stages of the algorithm.

Case 1. First, suppose that for some bipartition class of H, say X, the set $D \cap X$ contains two incomparable vertices s and v and there exists $t \in (N \cup R) \cap X$. Note that the triple s, v, t is considered in the first stage of the algorithm. Since the algorithm fails to report a decomposition of H, calling Lemma 6.1.1 yields the walks given in the statement of the lemma. By the symmetry of s and v, we can assume that we obtain walks $\mathcal{P}' \colon s \to t$ and $\mathcal{Q}' \colon v \to q$, where q is some vertex in X, and $\mathcal{P}'$ avoids $\mathcal{Q}'$. Let us enumerate $\mathcal{P}' = (p_1, p_2, \ldots, p_\ell)$ and $\mathcal{Q}' = (q_1, q_2, \ldots, q_\ell)$. Observe that for odd i we have $p_i, q_i \in X$, and for even i we have $p_i, q_i \in Y$. Let i be the smallest index such that $\{p_1, p_2, \ldots, p_i\} \subseteq D$ and $p_{i+1} \in N$; it is well defined, as $p_1 = s \in D$, $p_\ell = t \in N \cup R$, and N separates D and R. If $q_i \in D$, then $q_i p_{i+1} \in E(H)$, which contradicts $\mathcal{Q}'$ avoiding $\mathcal{P}'$. So let $j < i$ be the minimum index, such that $\{q_1, q_2 \ldots, q_j\} \subseteq D$ and $q_{j+1} \in N$; again, it is well defined, as $q_1 = v \in D$ and $q_i \in N \cup R$. Now observe that $q_{j+1} \in N$ and $j + 2 \leqslant i + 1$, so $p_{j+2} \in D \cup N$. This implies that $q_{j+1} p_{j+2} \in E(H)$, a contradiction. Summing up, if the first case applies, then the algorithm finds a decomposition of H.

Case 2. Now suppose that for some bipartition class of H, say X, it holds that

$(N \cup R) \cap X = \emptyset$. This means that $R \cap Y = \emptyset$, as H is connected. Let v be any vertex in $N \cap Y$, it exists, as $N \neq \emptyset$. Observe that $N(v) = D \cap X = X$. By Observation 5.3.1, $|D \cap X| \geqslant 3$ Let $N'_v \subseteq Y$ be the set of vertices with exactly the same neighborhood as v. It is straightforward to verify that $(V(H) \setminus N'_v, N'_v, \emptyset)$ is a decomposition of H, and it is found in the second stage of the algorithm. (We point out that this decomposition is not necessarily equal to (D, N, R), as some vertices of N'_v might be in D.)

Case 3. Finally, suppose that none of the cases above applies. In particular, every pair of vertices in $D \cap X$ (resp. $D \cap Y$) is comparable, as otherwise we are in Case 1. If $D \cap X \neq \emptyset$, we define x to be a vertex of $D \cap X$ whose neighborhood is maximal, i.e., for each $x' \in D \cap X$ it holds that $N(x') \subseteq N(x)$. Otherwise x is an arbitrary vertex from X, note that it exists since we are not in Case 2 and thus $(N \cup R) \cap X \neq \emptyset$. Similarly, let $y \in D \cap Y$ be a vertex with maximal neighborhood among all vertices from $D \cap Y$, or an arbitrary vertex from Y, if $D \cap Y = \emptyset$. Again, it is straightforward to verify that one of the triples considered for x, y in the third stage is a decomposition of H. □

The remaining part of this section consists of a series of technical lemmas, which, combined together, give us the mentioned Decomposition Lemma. Let us intuitively describe these steps. First, in Lemma 6.1.4 we show that it is enough to focus on the instances that are *reduced*, i.e., obtained from a general instance by applying some simple preprocessing steps. Second, in Lemma 6.1.5, we prove that if (H_1, H_2) is a factorization of H with respect to some decomposition δ, and we are able to solve LHOM for the instances (G', H', L') such that $H \in \{H_1, H_2\}$, then we can compute the answer also for the instances where $H' = H$. Finally, Theorem 6.1.6 states that if we can solve LHOM for the reduced instances (G', H', L') such that $H' \in \mathrm{Factors}_T(H)$ for some T, then we are able to solve also the case when $H' = H$.

We formally introduce the notion of a reduced instance.

Definition 6.1.3. An instance (G, H, L) of LHOM is *reduced* if:

(1) for each $v \in V(G)$, the set $L(v)$ is sound and incomparable, and

(2) for each $v \in V(G)$ the set $L(v)$ has at least two elements, and

(3) for each $a \in V(H)$ there exists $v \in V(G)$ such that $a \in L(v)$, and

(4) the graph H is connected.

For an instance (G, H, L) let us define its *measure* as

$$||(G, H, L)|| := \sum_{v \in V(G)} (|L(v)| - 1).$$

We stress out that formally the measure of an instance (G, H, L) does not depend on the graph H.

Observe that we always have $||(G, H, L)|| \leqslant |V(G)| \cdot |V(H)|$. On the other hand, if the instance is reduced, every $v \in V(G)$ contributes to $||(G, H, L)||$ by at least one, so $||(G, H, L)|| \geqslant |V(G)|$. Moreover, since each vertex of H appears on some list, $||(G, H, L)|| + |V(G)| = \sum_{v \in V(G)}(|L(v)| - 1) + |V(G)| \geqslant |V(H)|$, so $||(G, H, L)|| \geqslant |V(H)|/2$. Combining the two inequalities, we get that if (G, H, L) is reduced, then $||(G, H, L)|| \geqslant \frac{1}{3}(|V(G)| + |V(H)|)$.

For two instances $I = (G, H, L)$ and $I' = (G', H', L')$ of LHOM, we say that I' is a *subinstance* of I if G' is an induced subgraph of G and (here we use notation from Definition 5.2.3, Definition 5.2.4 and Definition 5.2.5):

- for every $v \in V(G')$ we have $L'(v) \subseteq L(v)$, or
- (this case applies only if there is a Γ-decomposition of H with factorization (H_1, H_2)) for every $v \in V(G')$ we have either $L'(v) \subseteq L(v)$ or
  - if $\Gamma \in \{F, BP\}$: if $L(v) \cap S \neq \emptyset$ for $S \in \{F, B, P\}$, then $L'(v) \subseteq L(v) \setminus S \cup \{s\}$ for the corresponding $s \in \{f, b, p\}$ (here $s \in V(H_2)$ is obtained by contracting the set S to a single vertex, as in the definition of H_2),
  - if $\Gamma = BB$: if $L(v) \cap (B_1 \cup B_2) \neq \emptyset$, then $L'(v) \subseteq L(v) \setminus (B_1 \cup B_2) \cup B'$, where $B' = \{b_1\}$ (resp. $B' = \{b_2\}$) if $L(v) \cap B_2 = \emptyset$ (resp. $L(v) \cap B_1 = \emptyset$) and $B' = \{b_1, b_2\}$ if $L(v) \cap B_1, L(v) \cap B_2 \neq \emptyset$.

The following lemma shows that we can restrict our attention to reduced instances.

Lemma 6.1.4. *Let $I = (G, H, L)$ be an instance of LHOM. Suppose we have an algorithm $\mathcal{A}$ that solves every reduced subinstance of I. Then there exists a (possibly empty) family $\mathbb{I}$ of reduced subinstances of I such that $\sum_{I' \in \mathbb{I}} ||I'|| \leqslant ||I||$ and that we can solve I by calling the algorithm $\mathcal{A}$ on instances from $\mathbb{I}$. All the additional computations are polynomial in $|V(G)| \cdot |V(H)|$.*

Proof. First note that if I is reduced, then we can solve it just by calling $\mathcal{A}$ on I. Furthermore, if for some $v \in V(G)$ we have $L(v) = \emptyset$, then we are dealing with a trivial no-instance. So from now on assume that I is not reduced and each list is non-empty.

We will prove the statement by the induction on $\mu(I) := |V(G)| + |V(H)| + ||I|| = \sum_{v \in V(G)} |L(v)| + |V(H)|$.

As a base case, consider $\mu(I) \leqslant |V(G)| + |V(H)|$. Since every list is non-empty, we have that $\mu(I) = |V(G)| + |V(H)|$ and each list has exactly one element. Then in time polynomial in $|V(G)| \cdot |V(H)|$ we can solve I by checking whether the unique function from $V(G)$ to $V(H)$ that respects lists L is a homomorphism.

Now suppose that the lemma holds for all instances I' which are either reduced or satisfy $\mu(I') < \mu(I)$. There are five possible reasons why I is not reduced, as indicated by cases (a)–(e) below. In cases (a)–(c) we will construct an equivalent subistance I' with $\mu(I') < \mu(I)$, which can be solved by the inductive assumption. Cases (d) and (e) are slightly more complicated.

(a) If for some $v \in V(G)$ the list $L(v)$ is not incomparable, i.e., contains two vertices a, b such that $N_H(a) \subseteq N_H(b)$, we consider $I' := (G, H, L')$, where L' is obtained from L by removing a from $L(v)$. To see that (G, H, L) and (G, H, L') are equivalent, note that in any homomorphism $f\colon (G, L') \to H$ with $f(v) = a$ we can always recolor v to the color b. Finally, observe that $\mu(I') = \mu(I) - 1$.

(b) If for some $v \in V(G)$ and $a \in V(H)$ we have $L(v) = \{a\}$, we consider $I' := (G - v, H, L')$, where L' is obtained from L by removing all non-neighbors of a from the lists of neighbors of v. Clearly, $\mu(I') \leqslant \mu(I) - 1$.

(c) If some $a \in V(H)$ does not appear in any list L, we consider $I' := (G, H - a, L)$. Clearly, $\mu(I') \leqslant \mu(I) - 1$.

(d) If H is disconnected, denote by $G_1, \ldots, G_p$ and $H^{(1)}, \ldots, H^{(q)}$, respectively, the connected components of G and of H. For $j \in [q]$, by L^j denote the lists L restricted to the vertices of $H^{(j)}$. For every $i \in [p]$ we define the set of instances $\mathbb{I}_i = \{(G_i, H^{(j)}, L_j) \mid j \in [q]\}$. As we always have $|V(H^{(j)})| < |V(H)|$, the instances in $\mathbb{I}_i$ can be solved by the inductive assumption. Clearly, (G, H, L) is a yes-instance if and only if every (G_i, H, L) is a yes-instance, and we observe that (G_i, H, L) is a yes-instance if and only if $\mathbb{I}_i$ contains a yes-instance. Hence, it is enough to solve every

instance that belongs to $\mathbb{I} = \bigcup_{i\in[p]} \mathbb{I}_i$. It is clear to observe that their total measure is at most $||I||$.

(e) If for some $v \in V(G)$ the list $L(v)$ is not sound, then either case (d) holds, or H is bipartite and $L(v)$ contains vertices from different bipartition class of H. Thus assume the latter holds. Denote these classes by X and Y. Clearly, if G is not bipartite, (G, H, L) is a no-instance. Otherwise, denote by $G_1, \ldots, G_p$ the connected components of G. Fix $i \in [p]$ let X_i, Y_i be the bipartition classes of G_i. Define $\mathbb{I}_i = \{(G_i, H, L_1), (G_i, H, L_2)\}$, where for every $u \in X_i$ we have $L_1(u) = L(u) \cap X$ and $L_2(u) = L(u) \cap Y$, and for every $u \in Y_i$ we have $L_1(u) = L(u) \cap Y$ and $L_2(u) = L(u) \cap X$. The instances in $\mathbb{I}_i$ can be solved by the inductive assumption. Clearly, (G, H, L) is a yes-instance if and only if every (G_i, H, L) is a yes-instance, and we observe that (G_i, H, L) is a yes-instance if and only if $\mathbb{I}_i$ contains a yes-instance. □

For a family $\mathcal{F}$ of graphs, by $\mathrm{Ind}(\mathcal{F})$ we denote the class of graphs that consists of all connected induced subgraphs of elements of $\mathcal{F}$.

The following lemma describes the crucial step of the proof of Theorem 6.1.6.

Lemma 6.1.5 (Decomposition Lemma). *Let $I = (G, H, L)$ be a reduced instance of LHom. Suppose that H has a Γ-decomposition δ for $\Gamma \in \{F, BB, BP\}$ with factorization (H_1, H_2). Suppose that we have an algorithm $\mathcal{A}$ that solves every reduced subinstance (G', H', L') of I such that $H' \in \mathrm{Ind}(\{H_1, H_2\})$. Then we can solve (G, H, L) by calling the algorithm $\mathcal{A}$ on a family $\mathbb{I}$ of such reduced subinstances of I such that $\sum_{I'\in\mathbb{I}} ||I'|| \leqslant ||I||$. All the additional computations are polynomial in $||I||$.*

Proof. The idea of the proof is simple: we aim to construct an instance (G, H_2, L_2) of LHom, that is a subinstance of (G, H, L) equivalent to (G, H, L), and then solve it, using the algorithm $\mathcal{A}$, by Lemma 6.1.4. First, we consider the subgraph of G induced by the vertices whose lists intersect $V(H_1)$. For each connected component of this subgraph we compute whether it can be mapped to H_1. Then, we define L_2 in a way that reflects the answer: if a component can be mapped to H_1, we replace vertices from $V(H_1)$ on its lists L by $V(H_2) \setminus V(H)$, otherwise we delete vertices of $V(H_1)$ from the lists. The actual proof is technically involved because the arguments on correctness and running time of a such procedure differ depending on the type of decomposition; in particular, if $\Gamma \in \{BP, BB\}$, there is more than one way how vertices of H_1 interact with the remaining vertices of H.

For a homomorphism $h : (G, L) \to H$ we define a *H_1-component for h* to be a maximal connected induced subgraph G' of G such that $h(V(G')) \subseteq V(H_1)$. Note that if $\Gamma = BB$, then each H_1-component must be bipartite.

We start with partitioning the vertex set of G according to the function L. The partition depends on the type of decomposition. We use the notation from Definition 5.2.3, Definition 5.2.4, and Definition 5.2.5.

If $\Gamma = F$, then we partition $V(G)$ into X_1 and X_2, so that:

- for every $x_1 \in X_1$, we have that $L(x_1) \subseteq F \cup Z$ and $L(x_1) \cap F \neq \emptyset$,
- for every $x_2 \in X_2$, we have that $L(x_2) \subseteq K \cup Z$.

Observe that this is indeed a partition, as every list is an incomparable set and thus $L(x) \cap F \neq \emptyset$ implies $L(x) \cap K = \emptyset$.

If $\Gamma = BP$, we partition the vertices of G into three sets X_1, X_2, X_3, so that

- for every $x_1 \in X_1$, we have $L(x_1) \subseteq B \cup Z$ and $L(x_1) \cap B \neq \emptyset$,
- for every $x_2 \in X_2$, we have $L(x_2) \subseteq M \cup P \cup Z$ and $L(x_2) \cap P \neq \emptyset$,
- for every $x_3 \in X_3$, we have $L(x_3) \subseteq M \cup K \cup Z$.

We claim that X_1, X_2, X_3 form a partition of $V(G)$. Since every list is an incomparable set, if there is a vertex from B in some $L(x)$, then no vertex from $P \cup K \cup M$ can be in $L(x)$. Similarly, if there is a vertex from P in $L(x)$, then no vertex from K can be in $L(x)$.

If $\Gamma = BB$, we partition the vertex set of G into four sets X_1, X_2, X_3, X_4 so that:

- for every $x_1 \in X_1$, we have $L(x_1) \subseteq B_1 \cup B_2 \cup Z$, $L(x_1) \cap B_1 \neq \emptyset$ and $L(x_1) \cap B_2 \neq \emptyset$,
- for every $x_2 \in X_2$, we have $L(x_2) \subseteq B_1 \cup M_2 \cup Z$ and $L(x_2) \cap B_1 \neq \emptyset$,
- for every $x_3 \in X_3$, we have $L(x_3) \subseteq B_2 \cup M_1 \cup Z$ and $L(x_3) \cap B_2 \neq \emptyset$,
- for every $x_4 \in X_4$, we have $L(x_4) \subseteq M_1 \cup M_2 \cup K \cup Z$.

Again, if a vertex from B_1 is in some list $L(x)$, then no vertex of $M_1 \cup K$ can be in $L(x)$, and if a vertex from B_2 is in $L(x)$, then no vertex from $M_2 \cup K$ can be in $L(x)$.

Forcing edges and family $\mathbb{C}$. Now let us define a special type of edges in G, called the *forcing edges*. The set of forcing edges will be denoted by $\widetilde{E}$ and its definition again depends on the type of the decomposition.

For $\Gamma = F$, the set $\widetilde{E}$ is the set with both endvertices in X_1. For $\Gamma = BP$, we define $\widetilde{E}$ to be the set of edges that are either contained in X_1 or are between X_1 and X_2. If $\Gamma = BB$, we define $\widetilde{E}$ to contain edges inside X_1, between X_1 and $X_2 \cup X_3$, and between X_2 and X_3.

The following claim will be crucial in our algorithm.

Claim 6.1.5.1. *Let $xy \in \widetilde{E}$ and let $h : (G, L) \to H$. The following holds:*

(1) if $\Gamma \in \{F, BP\}$, then $h(x) \in V(H_1)$ if and only if $h(y) \in V(H_1)$,

(2) if $\Gamma = BB$ and:

- *(a) $x, y \in X_1$, then $h(x) \in B_1$ if and only if $h(y) \in B_2$,*
- *(b) $x \in X_1$, $y \in X_2$, then $h(x) \in B_2$ if and only if $h(y) \in B_1$,*
- *(c) $x \in X_1$, $y \in X_3$, then $h(x) \in B_1$ if and only if $h(y) \in B_2$,*
- *(d) $x \in X_2$, $y \in X_3$, then $h(x) \in B_1$ if and only if $h(y) \in B_2$.*

Proof of Claim. If $\Gamma = F$, the claim is immediate, as K is a separator in H. For $\Gamma = BP$, the claim follows from the fact that there are no edges between B and $B \cup Z$, and no edges between P and Z.

Finally, if $\Gamma = BB$, there are three possible cases. If $x, y \in X_1$, we have $L(x), L(y) \subseteq B_1 \cup B_2 \cup Z$, so indeed the claim follows. If $x \in X_1$, $y \in X_2$ (and, by symmetry, if $x \in X_1$, $y \in X_3$) we have $L(x) \subseteq B_1 \cup B_2 \cup Z$ and $L(y) \subseteq B_1 \cup M_2 \cup Z$ and the claim follows by the fact that B_2 is non-adjacent to $M_2 \cup Z$ and B_1 is non-adjacent to $B_1 \cup Z$. Finally, if $x \in X_2$, $y \in X_3$, we have $L(x) \subseteq B_1 \cup M_2 \cup Z$ and $L(y) \subseteq B_2 \cup M_1 \cup Z$ and the claim follows from the fact that B_1 is non-adjacent to $M_1 \cup Z$ and B_2 is non-adjacent to $M_2 \cup Z$. This completes the proof of the claim. ⌟

Now we define a family $\mathbb{C}$ that will consists of pairs $(C, \mathcal{P})$, where C is an induced subgraph of G, and $\mathcal{P}$ is an ordered partition of $V(C)$ into at most two sets. We say that $(C, \mathcal{P}) \in \mathbb{C}$ *contains* a vertex $x \in V(G)$ or x *belongs* to $(C, \mathcal{P})$ if $x \in V(C)$.

If $\Gamma = F$, then we add to $\mathbb{C}$ each pair $(G[V(C)], (V(C)))$ such that C is a connected component of $(G[X_1], \widetilde{E})$. Similarly, if $\Gamma = BP$, we add to $\mathbb{C}$ each pair $(G[V(C)], (V(C)))$ such that C is a connected component of $(G[X_1 \cup X_2], \widetilde{E})$. Clearly, each vertex of G belongs to at most one element of $\mathbb{C}$.

If $\Gamma = BB$, the elements of $\mathbb{C}$ will be generated by exhaustive application of rules in Claim 6.1.5.1. For every $x \in \bigcup_{i\in[3]} X_i$ and every $j \in \{1,2\}$ we attempt to define a graph $C_{x,j}$ and a partition $\mathcal{P} = (V_1, V_2)$ of $V(C_{x,j})$. We initialize $V(C_{x,j}) := \{x\}$ and $V_j := \{x\}$. Then, for each $xy \in \widetilde{E}$, such that $x \in V(C)$ and $y \notin V(C)$, we proceed as follows.

(a) If $x, y \in X_1$ and $x \in V_1$, then add y to V_2.
If $x, y \in X_1$ and $x \in V_2$, then add y to V_1.

(b) If $x \in X_1, y \in X_2$ and $x \in V_2$, then add y to V_1.
If $x \in X_2, y \in X_1$ and $x \in V_1$, then add y to V_2.

(c) If $x \in X_1, y \in X_3$ and $x \in V_1$, then add y to V_2.
If $x \in X_3, y \in X_1$ and $x \in V_2$, then add y to V_1.

(d) If $x \in X_2, y \in X_3$ and $x \in V_1$, then add y to V_2.
If $x \in X_3, y \in X_2$ and $x \in V_2$, then add y to V_1.

If at any point of the construction we introduce to V_1 (V_2, resp.) a pair of adjacent vertices, we discard the current run. Otherwise, if none of the above rules can be applied anymore, we obtain $C_{x,j} = G[V_1 \cup V_2]$, and add $(C_{x,j}, (V_1, V_2))$ to $\mathbb{C}$. We stress out that $\mathbb{C}$ is a set, not a multiset. Because of that, since the above construction is exhaustive, each vertex $y \in V(G)$ belongs to at most two elements $(C, (V_1, V_2))$ of $\mathbb{C}$: once as a member of V_1 and once as a member of V_2.

Recall that by Claim 6.1.5.1, if $\Gamma \in \{F, BP\}$, for every list homomorphism $h : (G, L) \to H$, either all vertices from C are mapped to $V(H_1)$, or no vertex from C is mapped to $V(H_1)$.

The following observation encapsulates the main properties of $\mathbb{C}$.

Claim 6.1.5.2. *For every $(C, \mathcal{P}) \in \mathbb{C}$, the graph C is an induced subgraph of G.*

For $\Gamma \in \{F, BP\}$:

(1) Each vertex $x \in V(G)$ belongs to at most one element of $\mathbb{C}$.

(2) For each $(C, (V(C))) \in \mathbb{C}$ and $xy \in \widetilde{E}$, we have $x \in V(C)$ if and only if $y \in V(C)$.

(3) For each $h : (G, L) \to H$, and every H_1-component C for h we have $(C, (V(C))) \in \mathbb{C}$.

For $\Gamma = BB$:

(1) Each vertex $x \in V(G)$ belongs to at most two elements $(C, (V_1, V_2))$ of $\mathbb{C}$, once as an element of V_1 and once as an element of V_2.

(2) For every $(C, (V_1, V_2)) \in \mathbb{C}$ and $xy \in \widetilde{E}$ such that $x \in X_1 \cup X_2$, $y \in X_1 \cup X_3$, we have $x \in V_1$ if and only if $y \in V_2$.

(3) For every $h : (G, L) \to H$, and for every BB-component C for h we have $(C, (V_1, V_2)) \in \mathbb{C}$, where $V_j = \{v \in V(C) \mid h(v) \in B_j\}$ for $j \in \{1, 2\}$.

We define m_Γ as $m_F = m_{BP} = 1$, and $m_{BB} = 2$. Intuitively, m_Γ is the maximum possible number of elements of $\mathbb{C}$ that contain a fixed vertex x of $V(G)$, or, equivalently the number of sets in $\mathcal{P}$ for each $(C, \mathcal{P}) \in \mathbb{C}$.

Now we define a set $\mathbb{I}$ of subinstances of I that will be used to construct the lists L_2.

Claim 6.1.5.3. *In time polynomial in $||I||$ we can construct a family $\widehat{\mathbb{I}} = \{I_\mathcal{C} \ : \ \mathcal{C} \in \mathbb{C}\}$ of subinstances of I, such that:*

1. *for every $\mathcal{C} = (C, \mathcal{P}) \in \mathbb{C}$, the instance $I_\mathcal{C}$ is $(C, H_1, L_\mathcal{C})$, where*
   - *(a) if $\Gamma \in \{F, BP\}$, then for every $x \in V(C)$ we have $L_\mathcal{C}(x) = V(H_1) \cap L(x)$,*
   - *(b) if $\Gamma = BB$ and $\mathcal{P} = (V_1, V_2)$, then for $j \in \{1, 2\}$ and every $x \in V_j$ we have $L_\mathcal{C}(x) = B_j \cap V(H_1) \cap L(x)$;*
2. *it holds that*

$$\sum_{I' \in \widehat{\mathbb{I}}} ||I'|| \leqslant \sum_{x \in V(G)} (|L(x) \cap V(H_1)| - m_\Gamma).$$

Proof of Claim. First, consider $\Gamma \in \{F, BP\}$. We construct $\widehat{\mathbb{I}}$ as follows. For each $\mathcal{C} = (C, (V(C))) \in \mathbb{C}$ and $x \in V(C)$ we define $L_\mathcal{C}(x) = L(x) \cap V(H_1)$. By Claim 6.1.5.2, each vertex $x \in V(G)$ belongs to at most one instance in $\widehat{\mathbb{I}}$. Hence

$$\sum_{(C, H_1, L_\mathcal{C}) \in \widehat{\mathbb{I}}} \ \sum_{x \in V(C)} (|L_\mathcal{C}(x)| - 1) \leqslant \sum_{x \in V(G)} (|L(x) \cap V(H_1)| - 1).$$

If $\Gamma = BB$, for each $\mathcal{C} = (C, (V_1, V_2)) \in \mathbb{C}$ we define

$$L_\mathcal{C}(x) = \begin{cases} L(x) \cap B_1 \text{ if } x \in V_1, \\ L(x) \cap B_2 \text{ if } x \in V_2. \end{cases}$$

Then, the family $\widehat{\mathbb{I}}$ consists of all instances $(C, H_1, L_{\mathcal{C}})$ constructed in this way for $\mathcal{C} = (C, (V_1, V_2)) \in \mathbb{C}$.

Observe that by Claim 6.1.5.2 each vertex $x \in V(G)$ belongs to at most two instances $(C, L_{\mathcal{C}}), (C', L_{\mathcal{C}'}) \in \widehat{\mathbb{I}}$, and $L_{\mathcal{C}}(x) \subseteq B_1, L_{\mathcal{C}'}(x) \subseteq B_2$. Hence in particular, $L_{\mathcal{C}}(x) \cap L_{\mathcal{C}'}(x) = \emptyset$, so we have

$$\sum_{(C,H_1,L_{\mathcal{C}})\in\widehat{\mathbb{I}}} \sum_{x\in V(C)} (|L_{\mathcal{C}}(x)| - 1) \leqslant \sum_{x\in V(G)} (|L(x) \cap V(H_1)| - 2).$$

This completes the proof of claim. ⌟

For the next step we define $\mathbb{I} \subseteq \widehat{\mathbb{I}}$ to be the set of yes-instances in $\widehat{\mathbb{I}}$. Let us discuss how to obtain $\mathbb{I}$ from $\widehat{\mathbb{I}}$. Consider an instance $I' = (C, H_1, L_{\mathcal{C}}) \in \widehat{\mathbb{I}}$. Note that every subinstance of I' is also a subinstance of I. Thus the algorithm $\mathcal{A}$ satisfies the assumption of Lemma 6.1.4. By this lemma, I' can be solved by running the algorithm $\mathcal{A}$ on a family $\mathbb{I}'$ of reduced subinstances of I' such that $\sum_{J\in\mathbb{I}'} ||J|| \leqslant ||I'||$. Summing up, we can obtain $\mathbb{I}$ by calling $\mathcal{A}$ on reduced subinstances of I of total measure

$$\sum_{I'\in\widehat{\mathbb{I}}} ||I'|| \leqslant \sum_{x\in V(G)} (|L(x) \cap V(H_1)| - m_\Gamma), \tag{6.1}$$

where the bound follows from Claim 6.1.5.3.

Finally, for each $I' = (C, H_1, L_{\mathcal{C}}) \in \mathbb{I}$, we fix a homomorphism $h_{I'} : (C, L_{\mathcal{C}}) \to H_1$; it exists by the definition of $\mathbb{I}$.[1]

The last step is the construction of a single instance (G, H_2, L_2). The definition of the lists L_2 depends on the set $\mathbb{I}$. To introduce them, let us define one more notion.

Suppose a subset $S \subseteq V(H_1)$ was collapsed into a vertex s of H_2, i.e.,

$$\begin{aligned}
&\text{if } \Gamma = F \quad \text{and } S = F, \text{ then } s = f,\\
&\text{if } \Gamma = BP \text{ and } S = B, \text{ then } s = b,\\
&\text{if } \Gamma = BP \text{ and } S = P, \text{ then } s = p,\\
&\text{if } \Gamma = BB \text{ and } S = B_1, \text{ then } s = b_1,\\
&\text{if } \Gamma = BB \text{ and } S = B_2, \text{ then } s = b_2.
\end{aligned}$$

[1] An astute reader might notice that we present our algorithms for the decision version of the problem and formally we do not compute $h_{I'}$ while solving I'. Even though the algorithms can be easily adapted to find such a solution, we do not actually need to compute it: $h_{I'}$ is only used to argue the correctness.

We say that a vertex $x \in V(G)$ is an *s-vertex* if there is $I_x = (C, H_1, L_C) \in \mathbb{I}$, such that $x \in V(C)$ and $L_C(x) \cap S \neq \emptyset$. For every $x \in V(G)$ we obtain the list $L_2(x)$ from $L(x)$ by removing all vertices of $V(H_1)$ and adding a vertex s if x is an s-vertex. Note that in case of the BB-decomposition it might happen that a vertex is simultaneously a B_1- and a B_2-vertex, in such a case we add both b_1 and b_2 to $L_2(x)$.

Claim 6.1.5.4. *Let $xy \in \widetilde{E}$, such that x is an s_1-vertex and y is an s_2-vertex, for some $S_1, S_2 \subseteq V(H_1)$. If $s_1 s_2 \in E(H_2)$, then $h_{I_x}(x) h_{I_y}(y) \in E(H)$.*

Proof of Claim. We aim to show that $I_x = I_y$, which is a sufficient condition for the statement to hold since h_{I_x} is a homomorphism. Let $I_x = I_{(C_x, \mathcal{P}_x)} = (C_x, H_1, L_x), I_y = I_{(C_y, \mathcal{P}_y)} = (C_y, H_1, L_y)$. If $\Gamma \in \{F, BP\}$, then the fact that $(C_x, \mathcal{P}_x) = (C_y, \mathcal{P}_y)$ is a direct consequence of Claim 6.1.5.2, item (2). Hence, consider $\Gamma = BB$. Since $s_1 s_2 \in E(H_2)$, without loss of generality we may assume that $s_1 = b_1$ and $s_2 = b_2$. Let $\mathcal{P}_x = (V_1^x, V_2^x)$, $\mathcal{P}_y = (V_1^y, V_2^y)$.

By the definition of a b_1-vertex, $L_x(x) \cap B_1 \neq \emptyset$. Similarly, $L_y(y) \cap B_2 \neq \emptyset$. Hence, by the construction of $\mathbb{I}$ and Claim 6.1.5.3, we have that $x \in V_x^1$ and $y \in V_y^2$. Moreover, $L_x(x) \subseteq L(x)$ and $L_y(y) \subseteq L(y)$, and therefore $x \in X_1 \cup X_2$, and $y \in X_1 \cup X_3$. Hence, by Claim 6.1.5.2 (2), $y \in V_x^2$, and thus, by Claim 6.1.5.2 (1), $(C_x, (V_x^1, V_x^2)) = (C_y, (V_y^1, V_y^2))$. This concludes the proof. ⌟

The last crucial observation is as following.

Claim 6.1.5.5. *There exists a list homomorphism $h : (G, L) \to H$ if and only if there exists a list homomorphism $h' : (G, L_2) \to H_2$.*

Now, by the claim above and the assumption that there exists $\mathcal{A}$, it is sufficient to call Lemma 6.1.4 in order to solve the instance (G, H_2, L_2). Before we proceed to the proof of Claim 6.1.5.5, let us assume that it holds and show that it already implies our lemma.

Running time and the total size of $\widehat{\mathbb{I}}$. By Claim 6.1.5.3, the family $\widehat{\mathbb{I}}$ can be constructed in time polynomial in $||I||$. It remains to analyze the total sum of the measures of the instances we need to solve in order to solve I. Specifically, we will show that

$$\sum_{(C, H_1, L_C) \in \widehat{\mathbb{I}}} ||(C, H_1, L_C)|| + ||(G, H_2, L_2)|| \leqslant \sum_{x \in V(G)} (|L(x) \cap V(H_1)| - 1). \tag{6.2}$$

By Lemma 6.1.4, the total measure of instances considered while solving (G, H_2, L_2) is at most $||(G, H_2, L_2)||$. For every $x \in V(G)$, if $\Gamma = BB$, then $L_2(x) \subseteq (L(x) \setminus V(H_1)) \cup \{b_1, b_2\}$, and otherwise $L_2(x) \subseteq (L(x) \setminus V(H_1)) \cup \{s\}$, where $s \in \{b, p, f\}$. Consequently,

$$||(G, H_2, L_2)|| = \sum_{x \in V(G)} (|L_2(x)| - 1) \leqslant \sum_{x \in V(G)} (|L(x) \setminus V(H_1)| + m_\Gamma - 1).$$

Combining this with (6.1), the left side of (6.2) is bounded by

$$\sum_{x \in V(G)} (|L(x) \cap V(H_1)| + |L(x) \setminus V(H_1)| - 1) \leqslant \sum_{x \in V(G)} (|L(x)| - 1).$$

Since we call Lemma 6.1.4 precisely once for each instance in $\widehat{\mathbb{I}} \cup \{(G, H_2, L_2)\}$, this implies that the total size of instances for which we called the algorithm $\mathcal{A}$ is bounded by $||I||$.

This concludes the proof of the lemma, assuming Claim 6.1.5.5.

Proof of Claim 6.1.5.5. We prove the claim for $\Gamma \in \{BP, BB\}$, as they are more complicated ones. The proof for $\Gamma = F$ is analogous to the one for $\Gamma = BP$.

Case 1: $\Gamma = BP$**.** First, suppose there exists a list homomorphism $h : (G, L) \to H$. We define $h' : (G, L) \to H_2$ as follows:

$$h'(x) := \begin{cases} b & \text{if } h(x) \in B, \\ p & \text{if } h(x) \in P, \\ h(x) & \text{otherwise.} \end{cases}$$

Let us prove that h' is a list homomorphism from (G, L_2) to H_2. By definition of BP-decomposition and H_2, it is straightforward to verify that h' is a homomorphism. Let us prove that is respects lists L_2. Consider a vertex $x \in V(G)$. If $h(x) \notin B \cup P$, then clearly $h'(x) \in L_2(x)$, as no vertices from $V(H) \setminus (B \cup P)$ were removed from $L(x)$ to obtain $L_2(x)$. So let us assume that $h(x) \in B \cup P$. Then $x \in V(C)$ for some BP-component C for h. By Claim 6.1.5.2 (3) we know that $\mathcal{C} = (C, (V(C))) \in \mathbb{C}$. Hence, by Claim 6.1.5.3, there exists a yes-instance $I_\mathcal{C} = (C, H_1, L_\mathcal{C}) \in \widehat{\mathbb{I}}$ such that $L_\mathcal{C}(x) = L(x) \cap V(H_1)$. Thus, $(C, H_1, L_\mathcal{C}) \in \mathbb{I}$. Note that $L(x)$ cannot intersect both B and P. Hence, if $h(x) \in B$ (resp., $h(x) \in P$), then $h_{I_\mathcal{C}}(x) \in B$ (respectively, $h_{I_\mathcal{C}}(x) \in P$), implying that x is a b-vertex (resp., a p-vertex). Then $b \in L_2(x)$ (resp., $p \in L_2(x)$), This proves that h' respects L_2.

For the other direction, suppose that there exists a list homomorphism $h' : (G, L_2) \to$

H_2. We define $h : (G, L) \to H$ as follows. If $h'(x) \notin \{b, p\}$, then we set $h(x) := h'(x)$. If $h'(x) = b$ (or $h'(x) = p$, resp.), then x is a b-vertex and $x \in X_1$ and (resp., x is a p-vertex, and $x \in X_2$), in particular there exists an instance $I_x = (C_x, L_x) \in \mathbb{I}$ such that $x \in V(C)$ and $L_x(x) \subseteq B$ (resp., $L_x(x) \subseteq P$). We set $h(x) := h_{I_x}(x)$. By Claim 6.1.5.3 we have $L_x(x) \subseteq L(x) \cap B$ (resp., $L_i(x) \subseteq L(x) \cap P$), hence it remains to check if h is a homomorphism from G to H.

Let $xy \in E(G)$. Since h' is a homomorphism, if $h(x), h(y) \notin B \cup P$, then $h(x)h(y) \in E(H)$. If $h(x), h(y) \in B \cup P$, then either $h(x), h(y) \in P$, or $xy \in \widetilde{E}$. In the first case $h(x)h(y) \in E(H)$ since P is a reflexive clique, in the second case, by Claim 6.1.5.4, $h(x)h(y) = h_{I_x}(x)h_{I_y}(y) \in E(H)$.

Finally, assume that, say, $h(x) \in B \cup P$ and $h(y) \notin B \cup P$. Suppose first that $h(x) \in B$ and thus $h'(x) = b$. As h' is a homomorphism, we have that $h(y) = h'(y) \in N(b) \setminus \{p\} \subseteq K$, and thus $h(y)$ must be adjacent to $h(x)$. If $h(x) \in P$, we have $h'(x) = p$. Therefore, since h' is a homomorphism, $h(y) = h'(y) \in K \cup M$. Hence, $h(y)$ is adjacent to $h(x)$, which concludes the proof in this case.

Case 2: $\Gamma = BB$. First, suppose there exists a list homomorphism $h : (G, L) \to H$. We define $h' : (G, L_2) \to H_2$ as follows:

$$h'(x) := \begin{cases} b_1 & \text{if } h(x) \in B_1, \\ b_2 & \text{if } h(x) \in B_2, \\ h(x) & \text{otherwise.} \end{cases}$$

It is straightforward to observe that h' preserves edges. Let us verify that it respects lists L_2. For the sake of contradiction, suppose there is a vertex x of G such that $h'(x) \notin L_2(x)$. As h is a homomorphism respecting lists L and we removed only vertices of $B_1 \cup B_2$ to obtain L_2 from L, this can only happen if $h'(x) = b_j$ for $j \in \{1, 2\}$ and $b_j \notin L_2(x)$—by symmetry, assume that $j = 1$. Hence, there exists a BB-component C for h such that $x \in V(C)$. Let V_1, V_2 be the sets of vertices of C that are mapped by h, respectively, to B_1 and B_2. In particular, $x \in V_1$. By Claim 6.1.5.2 (3), $\mathcal{C} = (C, (V_1, V_2)) \in \mathbb{C}$. Hence, by Claim 6.1.5.3, there exists a yes-instance $I_{\mathcal{C}} = (C, H_1, L_{\mathcal{C}}) \in \widehat{\mathbb{I}}$ such that $L_{\mathcal{C}}(x) = B_1 \cap V(H_1) \cap L(x)$. Hence, $I_{\mathcal{C}} := (C, H_1, L_{\mathcal{C}}) \in \mathbb{I}$. Furthermore, $h_{I_{\mathcal{C}}}(x) \in B_1$, implying that x is a b_1-vertex, and thus $b_1 \in L_2(x)$. We proved that h' respects L_2.

For the opposite direction, suppose there is a list homomorphism $h' : (G, L_2) \to H_2$.

We define $h : (G, L) \to H$ as follows. If $h'(x) \notin \{b_1, b_2\}$, we set $h(x) := h'(x)$. If $h'(x) = b_j$ for $j \in \{1, 2\}$, then x is a b_j-vertex, i.e., there exists an instance $I_x = (C_x, L_x) \in \mathbb{I}$ such that $x \in C_x$ and $h_{I_x}(x) \in B_j$. We set $h(x) = h_{I_x}(x)$. Again, by Claim 6.1.5.3 we have that $L_x(x) \subseteq L(x) \cap B_j$, it remains to verify that h is a homomorphism from G to H.

Let $xy \in E(G)$ and, for the sake of contradiction, suppose that $h(x)h(y) \notin E(H)$. Since h' is a homomorphism, it cannot happen that $h(x), h(y) \notin B_1 \cup B_2$. By symmetry, assume that $h(x) \in B_1$, which implies that $x \in X_1 \cup X_2$ and that $h'(x) = b_1$. Note that we cannot have $h(y) \in B_1$ as this would imply that $h'(y) = b_1$, so $h'(x)h'(y) \notin E(H_2)$, a contradiction with h' being a homomorphism. If $h(y) \in B_2$, then $y \in X_1 \cup X_3$, and $xy \in \widetilde{E}$. Since $h'(x)h'(y) \in E(H_2)$, by Claim 6.1.5.4 we obtain that $h(x)h(y) = h_{I_x}(x)h_{I_y}(y) \in E(H)$. On the other hand, if $h(y) \in M_2 \cup K$, then $h(y) = h'(y) \in M_2 \cup K$ and thus $h(x)h(y)$ is an edge of H. This completes the proof of Claim 6.1.5.5 and the proof of Lemma 6.1.5.□

Let f be a function that takes two graphs G and H as arguments, and returns an element of $\mathbb{R}_+$. We say that f is *factor-monotone* if

- $f(G', H) \leqslant f(G, H)$, if G' is an induced subgraph of G, and
- $f(G, H') \leqslant f(G, H)$, if H' is an induced subgraph of H or there exists a factorization tree T such that $H' \in \text{Factors}_T(H)$.

Now, with Lemma 6.1.4 and Lemma 6.1.5 in hand, we can finally prove our main decomposition theorem. We state the theorem in general, in terms of factor-monotone functions.

Theorem 6.1.6. *Let $I = (G, H, L)$ be an instance of LHOM. Suppose we have an algorithm that solves every reduced instance $\widehat{I} = (\widehat{G}, \widehat{H}, \widehat{L})$ of LHOM, such that*

1. *$\widehat{G}$ is an induced subgraph of G, and*
2. *$\widehat{H}$ is an undecomposable graph in $\text{Ind}(\text{Factors}_T(H))$, for some T,*

in time $f(\widehat{G}, \widehat{H}) \cdot ||\widehat{I}||^{c_1}$ for some factor-monotone function f and some constant $c_1 > 0$. Then we can solve I in time $2f(G, H) \cdot (|V(G)| \cdot |V(H)|)^{c_2}$ for some constant $c_2 > 0$ that depends only on c_1.

Proof. We can assume that $|V(G)|$ is sufficiently large, as otherwise we can solve the problem by brute-force.

We observe that it is enough to prove the following claim.

Claim 6.1.6.1. *Let $I' = (G', H', L')$ be a reduced instance of LHom. Suppose we have an algorithm that solves every reduced instance $\widehat{I}' = (\widehat{G}', \widehat{H}', \widehat{L}')$ of LHom, such that*

1. *$\widehat{G}'$ is an induced subgraph of G', and*
2. *$\widehat{H}'$ is an undecomposable graph in $\mathrm{Ind}(\mathrm{Factors}_T(H'))$, for some T,*

in time $f(\widehat{G}', \widehat{H}') \cdot ||\widehat{I}'||^{c_1}$ for some factor-monotone function f and some constant $c_1 > 0$. Then we can solve I' in time $2f(G', H') \cdot ||I'||^{c_3}$, for some constant $c_3 > 0$.

Indeed, $I = (G, H, L)$ is a non-reduced instance, where G is sufficiently large and we assume that the claim holds, i.e., there exists an algorithm $\mathcal{A}$ that solves every reduced instance $I' = (G', H', L')$ in time $2f(G', H') \cdot ||I'||^{c_3}$, for some constant $c_3 > 0$. By Lemma 6.1.4, we can solve I by running $\mathcal{A}$ on a (possibly empty) family $\mathbb{I}$ of reduced subinstances of I, such that $\sum_{I' \in \mathbb{I}} ||I'|| \leqslant ||I||$, and all the additional computations are done in time $(|V(G)| \cdot |V(H)|)^{c_4}$ for some constant $c_4 > 0$. Note that

$$\sum_{I'=(G',H',L') \in \mathbb{I}} \left(2f(G', H') \cdot ||I'||^{c_3}\right) \leqslant 2f(G, H) \left(\sum_{I'=(G',H',L') \in \mathbb{I}} ||I'||\right)^{c_3} \leqslant 2f(G, H) \cdot ||I||^{c_3},$$

where the first inequality follows from the fact that $f(G', H') \leqslant f(G, H)$, and the second one follows from the convexity of the function $g(x) = x^{c_3}$. As $||I|| \leqslant |V(G)| \cdot |V(H)|$ and $|V(G)|$ is sufficiently large, the running time we obtain is bounded by

$$(|V(G)| \cdot |V(H)|)^{c_4} + 2f(G, H) \cdot ||I||^{c_3} \leqslant 2f(G, H) \cdot (|V(G)| \cdot |V(H)|)^{c_2},$$

for $c_2 = c_4 + c_3 + 1$.

It remains to prove Claim 6.1.6.1. Again, we observe that it is enough to prove the following.

Claim 6.1.6.2. *Let $I' = (G', H', L')$ be a reduced instance of LHom. Suppose we have an algorithm that solves every reduced instance $\widehat{I}' = (\widehat{G}', \widehat{H}, \widehat{L}')$ of LHom, such that*

1. *$\widehat{G}'$ is an induced subgraph of G', and*
2. *$\widehat{H}'$ is an undecomposable graph in $\mathrm{Ind}(\mathrm{Factors}_T(H'))$, for some T,*

in time $f(\widehat{G}', \widehat{H}') \cdot ||\widehat{I}'||^{c_1}$ for some factor-monotone function f and constant $c_1 > 0$. Then we can solve I' in time $f(G', H') \cdot ||I'||^{c_5} \cdot |V(H')|$, for some factor-monotone function f and constant $c_5 > 0$.

Indeed, to obtain Claim 6.1.6.1 from Claim 6.1.6.2 it is enough to recall that since I' is reduced, we have that $|V(H')| \leqslant 2 \cdot ||I'||$. Hence,

$$f(G', H') \cdot ||I'||^{c_5} \cdot |V(H')| \leqslant 2f(G', H') \cdot ||I'||^{c_3},$$

where $c_3 = c_5 + 1$.

Proof of Claim. We prove the claim by the induction on the number of vertices of H'. We assume that G' is connected, as otherwise we can solve the problem separately for each connected component of G'. For the base case, suppose that H' is either (i) a bi-arc graph, or (ii) a connected, undecomposable graph. In particular, if $|V(H')| \leqslant 2$, then H' must be a bi-arc graph, since H'^* has at most four vertices (recall Observation 5.3.1). Note that these cases can be recognized in polynomial time (combining Theorem 5.1.6 and the fact that circular-arc graphs can be recognized in time $|V(H')|^{c_6} \leqslant ||I'||^{c_6}$ for some constant c_6 [92], and by Theorem 6.1.2).

In case (i) we solve the problem in time polynomial in G' and H', which, since I' is reduced, is polynomial in $||I'||$. Hence, assume that (ii) holds, i.e., H' is connected (since I' is reduced), non-bi-arc, undecomposable, and has at least three vertices. By the assumption of the claim, we can solve I' in time $f(G', H') \cdot ||I'||^{c_1} \leqslant f(G', H') \cdot ||I'||^{c_1} \cdot (|V(H')| - 1)$ for some $c_1 > 0$. We define $c_7 = \max\{c_1, c_6\}$. The total time, including recognizing case (ii), is bounded by

$$||I'||^{c_6} + f(G', H') \cdot ||I'||^{c_1} \cdot (|V(H')| - 1) \leqslant f(G', H') \cdot ||I'||^{c_7} \cdot |V(H')|.$$

This completes the proof of the base case.

Now suppose that H' is a connected, non-bi-arc graph that is decomposable, and assume that for every $\widehat{H}'$ such that $|V(\widehat{H}')| < |V(H')|$ we can solve every instance $\widehat{I}' = (\widehat{G}', \widehat{H}', \widehat{L}')$ of LHOM in time $f(\widehat{G}', \widehat{H}') \cdot ||\widehat{I}'||^{c_7} \cdot |V(\widehat{H}')|$. Let δ be a decomposition of H' and let (H_1, H_2) be its factorization. Recall that if H' is decomposable, then for every $H'' \in \mathrm{Ind}(\mathrm{Factors}(H'))$ we have $|V(H'')| < |V(H')|$. Hence, by the induction hypothesis, there is an algorithm $\mathcal{A}$ that solves every instance $\widehat{I}' = (\widehat{G}', \widehat{H}', \widehat{L}')$ of LHOM where $\widehat{G}'$ is an induced subgraph of G' and $\widehat{H}' \in \mathrm{Ind}(\mathrm{Factors}(H'))$, in time

$$f(\widehat{G}', \widehat{H}') \cdot ||\widehat{I}'||^{c_7} \cdot |V(\widehat{H}')| \leqslant f(G', H') \cdot ||\widehat{I}'||^{c_7} \cdot |V(\widehat{H}')|.$$

By Lemma 6.1.5, we can solve the instance I' by calling algorithm $\mathcal{A}$ on a family $\mathbb{I}$ of instances with total measure bounded by $||I'||$, such that all the additional computations can be done in time $||I'||^{c_8}$, for some constant $c_8 > 0$. The total running time (including finding a decomposition of H) is bounded by

$$\begin{aligned}
&||I'||^{c_8} + \sum_{\widehat{I'}=(\widehat{G'},\widehat{H'},\widehat{L'})\in\mathbb{I}} f(\widehat{G'},\widehat{H'}) \cdot ||\widehat{I'}||^{c_7} \cdot |V(\widehat{H'})| \\
\leqslant\ &||I'||^{c_8} + \sum_{\widehat{I'}=(\widehat{G'},\widehat{H'},\widehat{L'})\in\mathbb{I}} f(G',H') \cdot ||\widehat{I'}||^{c_7} \cdot |V(\widehat{H'})| \\
\leqslant\ &||I'||^{c_8} + f(G',H') \sum_{\widehat{I'}\in\mathbb{I}} ||\widehat{I'}||^{c_7} \cdot (|V(H')| - 1) \\
=\ &||I'||^{c_8} + f(G',H') \cdot (|V(H')| - 1) \sum_{\widehat{I'}\in\mathbb{I}} ||\widehat{I'}||^{c_7} \\
\leqslant\ &||I'||^{c_8} + f(G',H') \cdot (|V(H')| - 1) \cdot ||I'||^{c_7} \leqslant f(G',H') \cdot ||I'||^{c_5} \cdot |V(H')|,
\end{aligned}$$

where the last inequality follows by choosing $c_5 = \max\{c_7, c_8\}$. ⌟

This concludes the proof of Theorem 6.1.6. □

To conclude this section, we summarize important properties of Factors(H).

Theorem 6.1.7. *Let H be a graph. In time polynomial in $|V(H)|$ we can construct a family* Factors(H) *of $\mathcal{O}(|V(H)|)$ graphs, such that:*

(1) each $H' \in$ Factors(H) is either bi-arc or undecomposable,

(2) for each $H' \in$ Factors(H), the graph H'^* is an induced subgraph of H^*,

(3) H is a bi-arc graph if and only if every $H' \in$ Factors(H) is a bi-arc graph,

(4) for every instance (G, H, L) of LHom*, the following holds:*

Assume that there exists an algorithm $\mathcal{A}$ that solves every reduced instance $I' = (G', H', L')$ such that G' is an induced subgraph of G and H' is an undecomposable element of Ind(Factors(H)), *in time $f(G', H') \cdot ||I'||^{\mathcal{O}(1)}$ for some factor-monotone function f. Then we can solve I in time $f(G, H) \cdot ||I||^{\mathcal{O}(1)}$.*

Proof of Theorem 6.1.7. Clearly, property (1) follows from the construction of Factors(H) and property (2) follows from Lemma 5.2.8.

To see that property (3) holds, assume that H is non-bi-arc and there exists a factorization (H_1, H_2) of H. It is enough to show that at least one of H_1, H_2 is non-bi-arc. If H is connected, then we are done by Lemma 5.2.6. Otherwise, there exists (i) a connected component H' of H that is decomposable with factorization (H'_1, H'_2) such that $H_1 = H'_1$ and H'_2 is an induced subgraph of H_2, and (ii) a connected component H'' of H that is non-bi-arc. Indeed, (i) follows from the definition of factorization of a disconnected graph. For (ii), if every connected component of H is a bi-arc graph, then by Theorem 5.1.6, H^* is a complement of circular-arc graph, so H is also a bi-arc graph. If H' is a bi-arc graph, then again, by the definition of factorization (H_1, H_2), H_2 contains H'' as an induced subgraph, so H_2 is non-bi-arc. If H' is non-bi-arc, by Lemma 5.2.6 at least one of H'_1, H'_2 is non-bi-arc, so at least one of H_1, H_2 is non-bi-arc. That concludes the proof of (3).

Last, property (4) follows precisely from Theorem 6.1.6. □

6.2 The algorithm

In this section we use Theorem 6.1.6 to prove the following version of Theorem 1.3.3 a). Note that here the input tree decomposition is not assumed to be optimal.

Theorem 1.2' a). *Let $I = (G, H, L)$ be instance of LHOM. We can solve I in time $i^*(H)^t \cdot (n \cdot |V(H)|)^{\mathcal{O}(1)}$, provided that G is an n-vertex graph given with its tree decomposition of width t and of size polynomial in n.*

Before we proceed to the proof of Theorem 1.2' a), we need two more tools. For an instance $I = (G, H, L)$ of LHOM, the *associated instance* is the instance (G^*, H^*, L^*) of LHOM, where G^* and H^* are, respectively, associated bipartite graphs of G and H, and L^* is the associated list function. We start by the following observation.

Observation 6.2.1. *Consider an instance (G, H, L) of LHOM and its associated instance (G^*, H^*, L^*). Suppose that G is given along with a tree decomposition $\mathcal{T}$ of width t.*

(1) For each $v \in V(G)$, $a \in V(H)$, we have $a \in L(v)$ if and only if $a' \in L^*(v')$ if and only if $a'' \in L^*(v'')$. In particular, each list is contained in one bipartition class of H^*.

(2) In time polynomial in $|V(G)|$ we can construct a tree decomposition $\mathcal{T}^*$ of G^* of width at most $2t$ with the property that for each $x \in V(G)$, each bag of $\mathcal{T}^*$ either contains both x', x'' or none of them.

Proof. The first item follows directly from the definition of (G^*, H^*, L^*). To see the second item, consider a tree decomposition $\mathcal{T}$ of G. We construct $\mathcal{T}^*$ by taking the same tree structure as for $\mathcal{T}$, and replacing each vertex v of G in each bag of $\mathcal{T}$ by the vertices v', v'' of G^*. It is straightforward to verify that this way we obtain a tree decomposition $\mathcal{T}^*$ with the desired properties. □

The only step that is missing is describing how to solve the instances (G, H, L) where H is an undecomposable graph. This follows, in particular, from the next lemma.

Lemma 6.2.2. *A reduced instance $I = (G, H, L)$ of LHOM can be solved in time $i(H^*)^t \cdot ||I||^{c_1}$, for some constant $c_1 > 0$, assuming a tree decomposition of G of width at most t and size polynomial in n is given.*

Proof. Let (G^*, H^*, L^*) be the associated instance of I. By Proposition 5.0.2, we know that $(G, L) \to H$ if and only if there is a clean (i.e., mapping twins to twins) homomorphism $(G^*, L^*) \to H^*$. We focus on finding such a clean homomorphism.

First, since I is reduced, each list L is a sound incomparable set. Observation 6.2.1 (1) asserts that every list in L^* is also a sound incomparable set. Thus the size of each list in L^* is at most $i(H^*)$. Moreover, for every $v \in V(G)$, the vertices in $L^*(v')$ are precisely the twins of the vertices in $L^*(v'')$. Finally, by Observation 6.2.1 (2), in time polynomial in $|V(G)|$ we can obtain a tree decomposition $\mathcal{T}^*$ of G^* with width at most $2t$, in which vertices of G^* come in pairs: a bag contains v' if an only if it also contains v''.

Consider the straightforward dynamic programming algorithm for LHOM, using the tree decomposition $\mathcal{T}^*$ of G^*. We observe that since we are looking for a clean homomorphism, we do not need to remember partial solutions in which the twins are not mapped to twins. Thus, even though the size of each bag of $\mathcal{T}^*$ is at most $2t$, the number of partial colorings we need to consider is bounded by $(\max_{v \in V(G^*)} |L^*(v)|)^t \leqslant i(H^*)^t$, and the lemma follows. □

Finally, let us wrap everything up and prove Theorem 1.2' a).

Proof of Theorem 1.2' a). First, we note that if G' is an induced subgraph of G, then we can easily modify the input tree decomposition of G into a tree decomposition of G' with width at most t. Second, if H' is an induced subgraph of H, or belongs to $\mathrm{Ind}(\mathrm{Factors}(H))$, then, by Lemma 5.2.8, $i^*(H') \leqslant i^*(H)$. Hence, $f(G, H) = i^*(H)^t$ is factor-monotone. By

Theorem 6.1.6, it is enough to show that every reduced instance $I' = (G', H', L')$ of LHom, such that G' is an induced subgraph of G and H' is an undecomposable graph in Ind(Factors(H)), in time $i^*(H')^t \cdot ||I'||^{c_1}$ for some constant $c_1 > 0$.

Hence, consider a reduced instance $I' = (G', H', L')$. If H' is a bi-arc graph, we solve the problem in time polynomial in G' and H' [39], which, since I' is reduced, is polynomial in $||I'||$. (An astute reader may notice that in [39] the target graph is assumed to be fixed. However, the algorithm works in polynomial time also if H is assumed to be a part of the input.) Since for bi-arc graphs H' we have $i^*(H') = 1$ by definition, the statement holds.

Thus, assume that H' is connected (since I' is reduced), non-bi-arc, and undecomposable. Recall that by Lemma 6.2.2, we can solve I' in time $i(H'^*)^t \cdot ||I'||^{c_1}$ for some constant $c_1 > 0$. If H' is a strong split graph, then by Lemma 5.3.4 we have that $i^*(H') = i(H'^*)$, and we are done. Otherwise, by Theorem 5.0.1, H'^* is connected (if H' is non-bipartite), or consists of two disjoint copies of H' (if H' is bipartite). Moreover, by Lemma 5.2.7, H'^* is undecomposable. Thus, by the definition of $i^*(H')$, we observe that

$$i(H'^*) \leqslant \max\{i(H'') \colon H'' \text{ is an undecomposable, connected, induced subgraph of } H'^*, \text{ whose complement is not a circular-arc graph}\} = i^*(H'^*) = i^*(H').$$

Hence, the total time is bounded by

$$i(H'^*)^t \cdot ||I'||^{c_1} \leqslant i^*(H')^t \cdot ||I'||^{c_1}.$$

That completes the proof. □

6.3 Building list gadgets

In this section we are going to introduce the tools necessary to prove the hardness counterpart of Theorem 1.3.3, and then prove Theorem 1.3.3 (b). As highlighted in its statement, we are going to show that the lower bound holds even if target graph as fixed, and not a part of an instance. Recall that for a fixed H, the LHom(H) problem takes as an instance a pair (G, L), where G is a graph and $L : V(G) \to 2^{V(H)}$, and asks whether $(G, L) \to H$. Again, we are going to show a slightly stronger theorem: that the claimed lower bound holds even if we assume that the input graph G is bipartite.

Theorem 6.3.1. *Let H be a fixed, connected, non-bi-arc graph. Unless the SETH fails, there is no algorithm that solves the LHom(H) problem on bipartite instances with n vertices and treewidth t in time $(i^*(H) - \varepsilon)^t \cdot n^{\mathcal{O}(1)}$, for any $\varepsilon > 0$.*

For that, we are going to define a list analogue of an $\mathcal{S}$-gadget that we introduced in Section 3.1.2. Let H be a graph, let $t \in \mathbb{N}$ and let $\mathcal{S} \subseteq V(H)^t$. A *list-$\mathcal{S}$-gadget* is a triple $(F, L, \overline{x})$, where F is a graph, $L : V(F) \to 2^{V(H)}$, and $\overline{x} = (x_1, \ldots, x_t) \in V(F)^t$, such that

$$\{(h(x_1), h(x_2), \ldots, h(x_t)) \mid h : (F, L) \to H\} = \mathcal{S}.$$

We call the elements of $\overline{x}$ the *interface vertices* of the gadget.

Observe that since the LHom(H) problem is a generalization of Hom(H) and ExtHom(H) problems, and the notions of $\mathcal{S}$-gadget and $\mathcal{S}$-construction, introduced in Section 3.1.2, can be seen as particular cases of list-$\mathcal{S}$-gadgets. The consequences of this fact are discussed later in Section 7.3.

The key element of the hardness proof is the existence of certain list-$\mathcal{S}$-gadgets.

Theorem 6.3.2. *Let $t \in \mathbb{N}$, and let H be a connected, non-bi-arc, undecomposable graph. Let $\mathcal{S} \subseteq V(H)^t$ be such that the set $\mathcal{S}_i = \{a_i \in V(H) \mid (a_1, \ldots, a_{i-1}, a_i, a_{i+1}, \ldots, a_t) \in \mathcal{S}\}$, for each $i \in [t]$, is a sound incomparable set in $V(H)$. Then there exists a list-$\mathcal{S}$-gadget.*

The general idea of the proof of Theorem 6.3.1 is quite straightforward: we are going to reduce from the k-Coloring problem. Consider an instance G of k-Coloring, where $k = i^*(H)$. Let $H' \in \text{Factors}_T(H)$, for some factorization tree T, contain a sound incomparable set S of size k. We construct a graph G' by replacing each edge uv of G with a copy of the gadget that simulates the inequality relation on S. By the properties of the gadget, we observe that G' has a list homomorphism to H' if and only if G is a yes-instance of k-Coloring. Furthermore, the construction of the gadget depends on H' only, and since H (and thus H') is assumed to be fixed, we conclude that $\text{tw}(G') = \text{tw}(G) + \mathcal{O}(1)$. The formal proof of this reduction is shown in Section 6.4. The remaining part of the current section is dedicated to the proof of Theorem 6.3.2. We also point out that to prove Theorem 6.3.1 we need Theorem 6.3.2 only for case $t = 1$, but the general setting will become useful in Chapter 7.

We claim that to prove Theorem 6.3.2 it is enough to focus on bipartite target graphs. The reason for that is captured by the following statement. Recall that for a set $W \subseteq$

$V(H)$ by W' (W'', resp.) we denote the set $\{a' \in V(H^*) \mid a \in W\} \subseteq V(H^*)$ ($\{a'' \in V(H^*) \mid a \in W\} \subseteq V(H^*)$, resp.) where H^* is the associated bipartite graph of H.

Proposition 6.3.3. *Let H be a graph, let (G, L) be an instance of LHOM(H^*), such that*

- *G is bipartite, with bipartition classes X and Y,*
- *$L(X) \subseteq V(H)'$ and $L(Y) \subseteq V(H)''$.*

Define $\widehat{L} : V(G) \to 2^{V(H)}$ as $\widehat{L}(v) = \{a \mid \{a', a''\} \cap L(v) \neq \emptyset\}$. Then the following hold:

1. *For $f \mid (G, L) \to H^*$, define $f' : V(G) \to V(H)$ by setting $f'(v)$ to be the unique vertex a of H, such that $f(v) \in \{a', a''\}$. Then $f' : (G, \widehat{L}) \to H$.*
2. *For $f' \mid (G, \widehat{L}) \to H$, define $f : V(G) \to V(H^*)$ by setting $f(v) := a'$ if $v \in X$ and $f(v) := a''$ if $v \in Y$, where $a = f'(v)$. Then $f : (G, L) \to H^*$.*

Proof. For 1., it is straightforward to verify that f' is well-defined. To see that f' is a homomorphism, consider an edge xy of G. Since f is a homomorphism, we have $f(x)f(y) \in E(H^*)$. Without loss of generality assume that $f(x) = a'$ and $f(y) = b''$ for some some edge ab of H (possibly $a = b$). But then $f'(x)f'(y) = ab \in E(H)$, so f' is a homomorphism from G to H. It remains to show that f' respects $\widehat{L}$. This follows from the definition of $\widehat{L}$: consider a vertex $v \in V(G)$, and let $a \in V(H)$ be such that $f(v) \in \{a', a''\}$. Note that since $\{a', a''\} \cap L(v) \neq \emptyset$, we have $a \in \widehat{L}(v)$.

Next, we prove 2. Again, note that f is well-defined. To see that f is a homomorphism, consider an edge xy of G, such that $x \in X$ and $y \in Y$. Assume that $f'(x) = a$ and $f'(y) = b$, where ab is an edge of H (possibly $a = b$). Then $f(x)f(y) = a'b'' \in E(H^*)$. To see that f respects the lists L. Consider a vertex $x \in X$ (the case of a vertex in Y is symmetric). Since $f'(x) = a$, we observe that $a \in \widehat{L}(x)$, which means that $a' \in L(x)$. That concludes the proof. □

Intuitively, Proposition 6.3.3 states that we can slightly modify the lists in a list-$\mathcal{S}$-gadget that is constructed for H^* to obtain a gadget constructed for H. Because of that, it turns out that it is enough to focus on bipartite graphs H. In particular, the following lemma is the crucial part of the proof of Theorem 6.3.2.

Lemma 6.3.4. *Let $t \in \mathbb{N}$, and let H be a connected, bipartite, undecomposable graph that is not the complement of a circular-arc graph. Let $\mathcal{S} \subseteq V(H)^t$ be such that for*

every $i \in [t]$ *the set* $\mathcal{S}_i = \{a_i \in V(H) \mid (a_1, \ldots, a_{i-1}, a_i, a_{i+1}, \ldots, a_t) \in \mathcal{S}\}$ *is a sound incomparable set. Then there exists a list-*$\mathcal{S}$*-gadget.*

We show that Lemma 6.3.4 implies Theorem 6.3.2.

Lemma 6.3.4 → Theorem 6.3.2: We distinguish three cases, based on whether H is bipartite, H is a strong split graph or none of these holds.

If H is bipartite, the lemma follows directly. If H is a strong split graph, let (B, P) be its partition, and let H' be the undecomposable and connected graph defined as in Lemma 5.3.4 3.. Recall that H' is not the complement of circular-arc graph. Observe that since for each $b \in B$ and $p \in P$ we have $N_H(b) \subseteq N_H(p)$, every set $\mathcal{S}_i$ is contained either in B or P. Thus, every $\mathcal{S}_i$ is a sound incomparable set in H'. We apply Lemma 6.3.4 to H' and obtain a list-$\mathcal{S}$-gadget.

If H is not bipartite nor a strong split graph, let H^* be an associated bipartite graph of H. By Theorem 5.1.6, the graph H^* is not the complement of a circular-arc graph. Moreover, by Theorem 5.0.1, since H is non-bipartite and connected, H^* is connected, and by Lemma 5.2.7, since H is connected, undecomposable, non-bi-arc, and not a strong split graph, H^* is undecomposable. Since every $\mathcal{S}_i$ is a sound incomparable set in H, every $\mathcal{S}'_i \subseteq V(H)'$ is a sound incomparable set in H^*. Define

$$\mathcal{S}' = \{(s'_1, \ldots, s'_k) \in \mathcal{S}'_1 \times \ldots \times \mathcal{S}'_m \mid (s_1, \ldots, s_k) \in \mathcal{S}\}.$$

We apply Lemma 6.3.4 to H^* and obtain a list-$\mathcal{S}'$-gadget $(F, L, \overline{x})$. We obtain the desired gadget $(F, \widehat{L}, \overline{x})$ by applying Proposition 6.3.3 to the instance (F, L) of $\text{LHOM}(H^*)$. □

Building gadgets from walks. To construct list-$\mathcal{R}$-gadgets as in Lemma 6.3.4, we use the existence of certain avoiding walks in H. For a set $\mathbb{K}$ of walks in H of equal length $\ell \geqslant 1$, let

$$\partial\mathbb{K} = \{(d_1, d_{\ell+1}) \in V(H)^2 \mid \mathcal{K} : d_1 \to d_{\ell+1} \in \mathbb{K}\},$$

and by $\mathbb{K}_i$, for $i \in [\ell + 1]$, the set of all vertices of H that occur as the i-th element of some walk in $\mathbb{K}$.

We construct a list gadget for a relation D such that $\partial\mathbb{K} \subseteq D$, but no pair of the form

$(d_1, d'_{\ell+1})$, where d_1 is the beginning of some $\mathcal{K} \in \mathbb{K}$, $d'_{\ell+1}$ is the end of some $\mathcal{K}' \in \mathbb{K}$, and $\mathcal{K}$ avoids $\mathcal{K}'$, belongs to D. We formalize it as follows.

Lemma 6.3.5. *Let $\mathbb{K} = \{\mathcal{K}_i\}_{i=1}^k$ be a set of walks in a graph H of equal length $\ell \geqslant 1$. Let $\mathbb{A}, \mathbb{B}$ be a partition of $\mathbb{K}$ such that every walk in $\mathbb{A}$ avoids every walk in $\mathbb{B}$. Then there exists a relation $D[\mathbb{A}, \mathbb{B}]$ and a list-$D[\mathbb{A}, \mathbb{B}]$-gadget $(P, L_{\mathbb{A},\mathbb{B}}, (x, y))$ such that:*

(a) $L_{\mathbb{A},\mathbb{B}}(x) = \mathbb{K}_1$, and $L_{\mathbb{A},\mathbb{B}}(y) = \mathbb{K}_{\ell+1}$,

(b) $\partial\mathbb{K} \subseteq D[\mathbb{A}, \mathbb{B}]$,

(c) if $(a, b) \in D[\mathbb{A}, \mathbb{B}]$ and $a \in \mathbb{A}_1$, then $b \notin \mathbb{B}_{\ell+1}$.

Furthermore, if every walk in $\mathbb{B}$ avoids every walk in $\mathbb{A}$, we additionally have

(d) if $(b, a) \in D[\mathbb{A}, \mathbb{B}]$ and $b \in \mathbb{B}_1$, then $a \notin \mathbb{A}_{\ell+1}$.

Proof. For every $i \in [k]$ and $j \in [\ell]$, by d^i_j we denote the j-th vertex of $\mathcal{K}_i$. Let $P = (p_1, \ldots, p_{\ell+1})$ be a path of length ℓ. Define $L_{\mathbb{A},\mathbb{B}} : V(P) \to 2^{V(H)}$ as $L_{\mathbb{A},\mathbb{B}}(p_i) = \mathbb{K}_i$, and let $x = p_1$ and $y = p_{\ell+1}$. Clearly, $(P, L_{\mathbb{A},\mathbb{B}})$ is an instance of LHOM(H), and $(P, L_{\mathbb{A},\mathbb{B}}, (x, y))$ a list-$D[\mathbb{A}, \mathbb{B}]$-gadget for some relation $D[\mathbb{A}, \mathbb{B}]$. We claim that $D[\mathbb{A}, \mathbb{B}]$ satisfies (a)-(c).

The statement (a) follows directly from the definitions of $L_{\mathbb{A},\mathbb{B}}$, x and y. For (b), consider any $i \in [k]$ and $(d^i_1, d^i_{\ell+1}) \in \partial\mathbb{K}$. Define $f_i(p_j) := d^i_j$ for $j \in [\ell + 1]$. Observe that f_i is indeed a list homomorphism $(P, L_{\mathbb{A},\mathbb{B}}) \to H$, since for every edge $p_j p_{j+1}$ of P it holds that $f_i(p_j)$ and $f_i(p_{j+1})$ are consecutive vertices d^i_j and d^i_{j+1} of the walk $\mathcal{K}_i$, and thus they are adjacent in H. Since $f_i(x) = d^i_1$ and $f_i(y) = d^i_{\ell+1}$, the statement (b) follows.

To show (c), suppose there exists a list homomorphism $f : (P, L_{\mathbb{A},\mathbb{B}}) \to H$ such that $f(x) \in \mathbb{A}_1$ and $f(y) \in \mathbb{B}_{\ell+1}$. We note that $\mathbb{A}_1 \cap \mathbb{B}_1 = \emptyset$, as $\ell \geqslant 1$ and each walk from $\mathbb{A}$ avoids each walk from $\mathbb{B}$. Let $i \in [\ell + 1]$ be the minimum integer such that there exists a walk $\mathcal{K}_r \in \mathbb{B}$ with $f(p_i) = d^r_i$. Note that it exists since $f(p_{\ell+1}) \in \mathbb{B}_{\ell+1}$. Moreover, $i \geqslant 2$ since $f(p_1) \in \mathbb{A}_1$ and $\mathbb{A}_1 \cap \mathbb{B}_1 = \emptyset$. By minimality of i, we have $f(p_{i-1}) = d^s_{i-1}$ for some $\mathcal{K}_s \in \mathbb{A}$. Thus there is a walk $\mathcal{K}_s \in \mathbb{A}$ and a walk $\mathcal{K}_r \in \mathbb{B}$, such that $\mathcal{K}_s$ does not avoid $\mathcal{K}_r$, a contradiction. The property (d) can by shown by switching the roles of $\mathbb{A}$ and $\mathbb{B}$.□

Let $\mathcal{S}^1 \subseteq V(H)^t$ and $\mathcal{S}^2 \subseteq V(H)^{t'}$. Let $W_1 = (P_1, L, (x_1, \ldots, x_t))$ and $W_2 = (P_2, L, (y_1, \ldots, y_{t'}))$ be, respectively, a list-$\mathcal{S}^1$-gadget and a list-$\mathcal{S}^2$-gadget, such that $V(P_1) \cap$

$V(P_2) = \emptyset$. A *composition at* (i, i') of W_1 and W_2, where $i \in [t], i' \in [t']$, is a list gadget

$$W = (P, L', (x_1, \ldots, x_t, y_1, \ldots, y_{i'-1}, y_{i'+1}, \ldots, y_{t'})),$$

obtained by

- identifying $x_i \in V(P_1)$ and $y_{i'} \in V(P_2)$,
- setting $L'(v) = L(v)$ for every $v \in V(P) \setminus \{x_i\}$ and $L'(x_i) = L(x_i) \cap L(y_{i'})$.

It is straightforward to verify that W is a list-$\mathcal{S}$-gadget, where $\mathcal{S}$ is defined as

$$\mathcal{S} = \{(s_1, \ldots, s_t, s'_1, \ldots, s'_{i'-1}, s'_{i'+1}, \ldots, s'_{t'}) \mid (s_1, \ldots, s_t) \in \mathcal{S}^1, \\ (s'_1, \ldots, s'_{t'}) \in \mathcal{S}^2 \text{ and } s_i = s'_{i'}\}.$$

Clearly, a composition of two gadgets can be described by identifying interface vertices of these gadgets.

Before we prove Lemma 6.3.4, we first show how to obtain a few particular gadgets that will serve as building blocks for the general construction. Let H be a bipartite graph with an obstruction $\mathbb{O}$, and let $(\alpha, \beta) \in C(\mathbb{O})$. To make the definitions more intuitive, let us assign logic values to vertices α, β in the following way: α will be interpreted as false, and β will be interpreted as true. Define a k-ary relation $\mathrm{OR}_t = \{\alpha, \beta\}^k \setminus \{\alpha^k\}$ and a k-ary relation $\mathrm{NAND}_t = \{\alpha, \beta\}^k \setminus \{\beta^k\}$.

Lemma 6.3.6. *Let H be a bipartite graph with an obstruction $\mathbb{O}$ and let $(\alpha, \beta) \in C(\mathbb{O})$. For every $k \geqslant 2$ there exist a list-NAND_k-gadget and a list-OR_k-gadget.*

Proof. First, notice that for any $k \geqslant 4$, the relation OR_k can be expressed as a composition of the binary relation $\mathrm{NEQ} = \{\alpha\beta, \beta\alpha\}$ and ternary OR_3. Indeed, note that a list-OR_k-gadget with interface vertices $(x_1, x_2, \ldots, x_k)$ can be easily constructed by introducing a list-OR_{k-1}-gadget with interface vertices $(x_1, x_2, \ldots, x_{k-2}, y)$, a list-$\mathrm{OR}_3$-gadget with interface vertices (y', x_{k-1}, x_k), and a list-NEQ-gadget with interface vertices (y, y') (this corresponds to the textbook NP-hardness reduction from Cnf-Sat to 3-Sat [49, Sec. 3.1.1.]).

We also observe that for any $k \geqslant 3$, the relation NAND_k can be expressed using the composition of the OR_k relation and the NAND_2 relation. Indeed, introduce a list-OR_k-gadget with interface vertices $(x_1, x_2, \ldots, x_k)$, and k copies of a list-NAND_2-gadget, each

with interface vertices (y^i, z^i), for $i \in [k]$. We obtain a list-NAND$_k$-gadget with interface vertices $(y^1, \ldots, y^k)$ by identifying x_i with z^i for each $i \in [k]$.

We construct a list-NEQ-gadget as follows. Let $\mathcal{X}, \mathcal{Y}, \mathcal{X}', \mathcal{Y}'$ be the walks given by Observation 5.1.9. Let $(P_1, L_{\{\mathcal{X}\},\{\mathcal{Y}\}}, (x, y))$ and $(P_2, L_{\{\mathcal{Y}'\},\{\mathcal{X}'\}}, (x', y'))$ be, respectively, the gadgets obtained by Lemma 6.3.5. It is straightforward to observe that the graph obtained from P_1 and P_2 by identifying x with x', and y with y', is a NEQ-gadget $(F, L, (x, y))$.

Note that a list-OR$_2$-gadget can be obtained by identifying two of three interface vertices of a list-OR$_3$-gadget. Also, a list-NAND$_2$-gadget can be obtained by taking two list-NEQ-gadgets W_1 and W_2, and a list-OR$_2$-gadget Z, and identifying each interface vertex of Z with one interface vertex of Z_1 and Z_2, respectively.

So, in order to prove the lemma, it remains to build gadget for the ternary relation OR$_3$. By symmetry of $\mathbb{O}$, we can assume that $\alpha = w_1, \beta = w_5$ if $\mathbb{O}$ is either an induced 6-cycle or an induced 8-cycle, or $\alpha = u_0, \beta = u_1$ if $\mathbb{O}$ is an asteroidal subgraph. Set $\gamma = w_3$ in the former case, or $\gamma = u_{k+1}$ in the latter one. The high-level idea is to construct, for every $c \in \{\alpha, \beta, \gamma\}$, a list-$R(c)$-gadget $(P_c, L, (x_c, y_c))$ for $R(c) = \{ab \mid a \in \{\alpha, \beta\}, b \in \{\alpha, \beta, \gamma\}\} \setminus \{(\alpha, c)\}$.

Suppose we have constructed gadgets $(P_c, L, (x_c, y_c))$ for every $c \in \{\alpha, \beta, \gamma\}$. Consider a graph G obtained from P_α, P_β, and P_γ by identifying the vertices y_α, y_β, and y_γ to a single vertex y. The lists of vertices remain unchanged, note that all y-vertices have the same list $\{\alpha, \beta, \gamma\}$, so identifying the vertices does not cause any conflict here. Obviously we have $L(x_\alpha) = L(x_\beta) = L(x_\gamma) = \{\alpha, \beta\}$. Observe it is not possible to map all $x_\alpha, x_\beta, x_\gamma$ to α. However, every triple of colors from $\{\alpha, \beta\}^3 \setminus \{(\alpha, \alpha, \alpha)\}$ might appear on vertices $x_\alpha, x_\beta, x_\gamma$ in some list homomorphism $(G, L) \to H$. Thus, $(G, L, (x_\alpha, x_\beta, x_\gamma))$ satisfies the definition of a list-OR$_3$-gadget. Thus, let us show how to construct $(P_c, L, (x_c, y_c))$ for every $c \in \{\alpha, \beta, \gamma\}$.

If $\mathbb{O}$ is isomorphic to C_6 or C_8, then the construction is straightforward and it is shown on Figure 6.1 (the picture shows list-OR$_3$-gadgets, with y-vertices of P_c's already identified). For the case if $\mathbb{O}$ is an asteroidal subgraph, the construction will be quite similar, but a bit more involved.

First, let us build an auxiliary gadget. Recall the walks $\mathcal{W}[u_1, u_{k+1}]$ and $\mathcal{W}[u_{k+1}, u_1]$

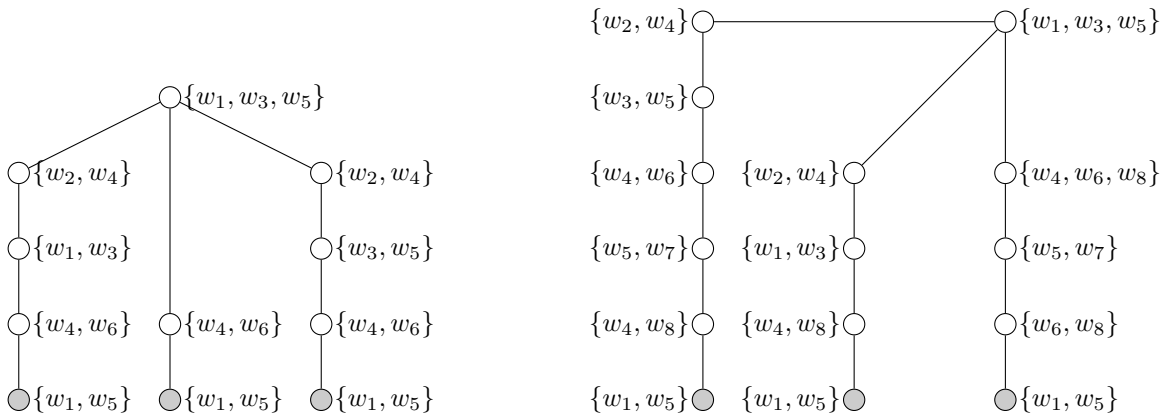


Figure 6.1: An OR_3-gadget for $\mathbb{O} \simeq C_6$ (left) and for $\mathbb{O} \simeq C_8$ (right). Recall that the consecutive vertices of $\mathbb{O}$ are denoted by $(w_1, w_2, \ldots)$. The sets next to vertices indicate lists. Interface vertices are marked gray.

given by Observation 5.1.8, and consider the following walks of equal length:

$$\mathcal{A} = (u_0, v_0, \ldots, u_0), \qquad \mathcal{B}_1 = (u_1, v_1, \ldots, u_1), \qquad \mathcal{B}_2 = \mathcal{W}[u_1, u_{k+1}].$$

Note that $\mathcal{A}$ is non-adjacent to $\mathcal{B}_1, \mathcal{B}_2$. Moreover, define

$$\begin{aligned} \mathcal{A}_1 &= (u_0, v_0, \ldots, u_0) && \circ (u_0, v_0, \ldots, u_0), \\ \mathcal{A}_2 &= \mathcal{W}^u_{1,0} && \circ (u_0, v_0, \ldots, u_0), \\ \mathcal{B} &= (u_{k+1}, v_{k+1}, \ldots, u_{k+1}) && \circ \mathcal{W}[u_{k+1}, u_1], \end{aligned}$$

and note that $\mathcal{A}_1, \mathcal{A}_2$ are non-adjacent to $\mathcal{B}$. Thus, by Lemma 6.3.5 we obtain that a composition $(P, L, (x_1, y_2))$ of $(P_1, L_{\{\mathcal{A}\},\{\mathcal{B}_1,\mathcal{B}_2\}}, (x_1, y_1))$ and $(P_2, L_{\{\mathcal{A}_1,\mathcal{A}_2\},\{\mathcal{B}\}}, (x_2, y_2))$, is a list-$\{u_0u_0, u_1u_0, u_1u_1\}$-gadget.

Now, to construct $(P_c, L, (x_c, y_c))$, suppose we are given $a, b \in V(H)$ such that $\{a, b, c\} = \{u_0, u_1, u_{k+1}\}$. Let $\mathcal{X}_c : u_0 \to a$, $\mathcal{Y}_c : u_0 \to b$, and $\mathcal{Z}_c : u_1 \to c$ be the walks given by Observation 5.1.9 such that $\mathcal{X}_c, \mathcal{Y}_c$ avoid $\mathcal{Z}_c$ and $\mathcal{Z}_c$ avoids $\mathcal{X}_c, \mathcal{Y}_c$. Consider the gadget $(F, L_{\{\mathcal{X}_c,\mathcal{Y}_c\},\{\mathcal{Z}_c\}}, (x_3, y_3))$, obtained by applying Lemma 6.3.5, and note that this is a list-$\{u_0a, u_0b, u_1c\}$-gadget. It is straightforward to observe (see Figure 6.2) that the composition of $(P, L, (x_1, y_2))$ and $(F, L_{\{\mathcal{X}_c,\mathcal{Y}_c\},\{\mathcal{Z}_c\}}, (x_3, y_3))$ is an $R(c)$-gadget $(P_c, L, (x_c, y_c))$. This concludes the construction of the OR_3-gadget and the proof of the lemma. □

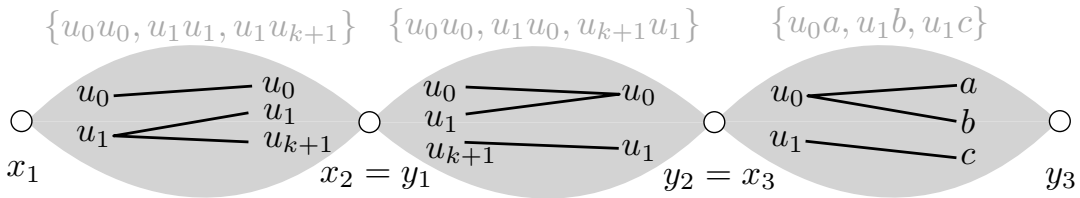


Figure 6.2: The composition of gadgets $(P_1, L_{\{\mathcal{A}\},\{\mathcal{B}_1,\mathcal{B}_2\}}, (x_1, y_1))$,$(P_2, L_{\{\mathcal{A}_1,\mathcal{A}_2\},\{\mathcal{B}\}}, (x_2, y_2))$ and $(F, L_{\{\mathcal{X}_c,\mathcal{Y}_c\},\{\mathcal{Z}_c\}}, (x_3, y_3))$ that together creates a list-$R(c)$-gadget. Thick black lines denote the possible mappings of white vertices to vertices of H.

The last ingredient needed for the proof of Lemma 6.3.4 is the construction of distinguisher gadgets.

Definition 6.3.7 (Distinguisher). Let H be a bipartite graph, let $S \subseteq V(H)$, and let $(\alpha, \beta) \in \mathcal{C}(\mathbb{O})$, such that $\{\alpha, \beta\} \cup S$ is contained in one bipartition class of H. Let $a, b \in S$. Consider a binary relation $\mathcal{D}_{a/b} \subseteq S \times \{\alpha, \beta\}$ such that $(a, \alpha), (b, \beta) \in \mathcal{D}_{a/b}$, $(a, \beta) \notin \mathcal{D}_{a/b}$ and for every $c \in S \setminus \{a, b\}$ we have $\{(c, \alpha), (c, \beta)\} \cap \mathcal{D}_{a/b} \neq \emptyset$. A relation $\mathcal{D}_{a/b}$ is called an (a, b)-*distinguishing* relation. A list-$\mathcal{D}_{a/b}$-gadget is called a *distinguisher* gadget.

Distinguisher gadgets will be constructed using the following lemma, whose proof is postponed to Section 6.3.2.

Lemma 6.3.8. *Let H be a connected, bipartite, undecomposable graph that contains an obstruction $\mathbb{O}$, and let $(\alpha, \beta) \in C(\mathbb{O})$. Let $S \subseteq V(H)$ be an incomparable set of $k \geqslant 2$ vertices, contained in the same bipartition class as α, β. Let a and b be two distinct vertices of S. For each $v \in S$ there exists a walk $\mathcal{K}_v$, of positive length, satisfying the following properties:*

(1) $\mathcal{K}_v$ is a v-α-walk or a v-β-walk,

(2) $\mathcal{K}_a$ is an a-α-walk and $\mathcal{K}_b$ is a b-β-walk,

(3) all walks terminating at α avoid all walks terminating at β. In particular, all walks have equal length.

By Lemma 6.3.5, for any $a, b \in S$, the existence of distinguisher gadgets for some distinguishing relation $\mathcal{D}_{a/b}$ follows directly from Lemma 6.3.8 applied for $\mathbb{A}$ being the set of walks terminating at α, and $\mathbb{B}$ being the set of walks terminating at β.

Corollary 6.3.9. *Let H be a connected, bipartite, undecomposable graph with an obstruction $\mathbb{O}$ and let $(\alpha, \beta) \in C(\mathbb{O})$. Let $S \subseteq V(H)$ be an incomparable set contained in the same bipartition class as α and β. Then for every pair (a, b) of distinct elements of S there exists a list-$\mathcal{D}_{a/b}$-gadget for some distinguishing relation $\mathcal{D}_{a/b}$.* □

Now Lemma 6.3.4 is a consequence of Lemma 6.3.6 and Corollary 6.3.9.

Proof of Lemma 6.3.4. If $t = 1$, then consider a graph F that consists of one vertex v with list $L(v) = \mathcal{S}$. Clearly, $(F, L, \{v\})$ is a list-$\mathcal{S}$-gadget. Thus assume that $t \geqslant 2$. Let X and Y be the bipartition classes of H. Recall (Theorem 5.1.6) that since H is not the complement of a circular-arc graph, then it contains an obstruction $\mathbb{O}$. Let $C(\mathbb{O}) = \{(\alpha, \beta), (\alpha', \beta')\}$, without loss of generality assume that $\alpha, \beta \in X$ and $\alpha', \beta' \in Y$. Let $\mathcal{R} = (\mathcal{S}_1 \times \ldots \times \mathcal{S}_t) \setminus \mathcal{S} = \{R_1, \ldots, R_q\}$, where for each $k \in [q]$ we have $R_k = (r_1^k, \ldots, r_t^k)$. For each $\mathcal{S}_i$ we enumerate $\mathcal{S}_i = \{s_1^i, \ldots, s_{p_i}^i\}$, where $p_i = |\mathcal{S}_i|$.

We construct the desired gadget $(F, L, (x_1, \ldots, x_t))$ in three steps as follows.

Step I. In this step, for every $i \in [t]$ and $j \in [p_i]$ we construct a list-$I(i,j)$-gadget $(I_{i,j}, L, (x_{i,j}, c_{i,j}))$, where $I(i,j) \subseteq \mathcal{S}_i \times \{\alpha, \beta\}$ is a binary relation such that:

- $(s_j^i, \beta) \in I(i,j)$,
- $(s_j^i, \alpha) \notin I(i,j)$, and
- for every $j' \neq j$ we have that $(s_{j'}^i, \alpha) \in I(i,j)$.

We first assume that $\mathcal{S}_i \subseteq X$. For every $j' \in [p_i] - \{j\}$ we call Corollary 6.3.9 to obtain a list-$D_{s_j^i / s_{j'}^i}$-gadget $(D_{j/j'}, L, (x_{j/j'}, y_{j/j'}))$. We identify the vertices $x_{j/j'}$, for all $j' \in [p_i] \setminus \{j\}$, into a single vertex $x_{i,j}$, and introduce a new vertex $c'_{i,j}$. Next, we use Lemma 6.3.6 to construct a list-OR_{p_i}-gadget and identify its p_i interface vertices with distinct elements of $\{c'_{i,j}\} \cup \{y_{j/j'} \mid j \neq j'\}$. This completes the construction of $(I_{i,j}, L, (x_{i,j}, c_{i,j}))$, where $c_{i,j} = c'_{i,j}$.

Indeed, consider a homomorphism $h : (I_{i,j}, L) \to H$. If $h(x_{i,j}) = s_j^i$, then, for every $j' \in [p_i] \setminus \{j\}$ the list-$D_{s_j^i / s_{j'}^i}$-gadget forces $h(y_{j/j'}) = \alpha$, so the list-OR_{p_i}-gadget forces $h(c_{i,j}) = \beta$. Thus, $(s_j^i, \alpha) \notin I(i,j)$. Moreover, by the properties of a distinguisher, there exists a homomorphism that maps $x_{i,j}$ to s_j^i and every $y_{j/j'}$ to α. By the properties of the OR_{p_i} relation, this can be extended to the remaining vertices of the list-OR_{p_i}-gadget so

that $c_{i,j}$ is mapped to β. Thus, $(s^i_j, \beta) \in I(i,j)$. Last, if $h(x_{i,j}) = s^i_{j'}$ for some $j' \neq j$, then we can have $h(y_{j/j'}) = \beta$ and hence also we can have $h(c_{i,j}) = \alpha$. Hence, $(s^i_{j'}, \alpha) \in I(i,j)$. Therefore, the gadget constructed in this case satisfies the desired properties.

If $\mathcal{S}_i \subseteq Y$, we analogously use Corollary 6.3.9 and Lemma 6.3.6 to construct a list-$I'(i,j)$-gadget $(I'_{i,j}, L, (x_{i,j}, c'_{i,j}))$, where $I'(i,j) \subseteq S_i \times \{\alpha', \beta'\}$ is a relation such that:

- $(s^i_j, \beta') \in I'(i,j)$,
- $(s^i_j, \alpha') \notin I'(i,j)$, and
- for every $j' \neq j$ we have that $(s^i_{j'}, \alpha') \in I'(i,j)$.

Then we create $I_{i,j}$ by adding a vertex $c_{i,j}$ with list $\{\alpha, \beta\}$ and making it adjacent to $c'_{i,j}$. Recall that $\alpha\alpha', \beta\beta' \in E(H)$, but $\alpha\beta', \alpha'\beta \notin E(H)$, hence we obtain the desired gadget.

Step II. In this step, for every tuple $\bar{j} = (j_1, \ldots, j_t) \in [p_1] \times \ldots \times [p_t]$, we construct a list-$J(\bar{j})$-gadget $(J_{\bar{j}}, L, (x^{\bar{j}}_1, \ldots, x^{\bar{j}}_t))$, where $J(\bar{j}) \subseteq \mathcal{S}_1 \times \ldots \times \mathcal{S}_t$ is the relation in which:

- $(s^1_{j_1}, \ldots, s^t_{j_t}) \notin J(\bar{j})$,
- for any $(s'_1, \ldots, s'_t) \in \mathcal{S}_1 \times \ldots \times \mathcal{S}_t \setminus \{(s^1_{j_1}, \ldots, s^t_{j_t})\}$, it holds that $(s'_1, \ldots, s'_t) \in J(\bar{j})$.

For every $i \in [t]$, we introduce a list-$I(i, j_i)$-gadget, constructed in previous step, with interface vertices $(x^{\bar{j}}_i, c_{i,j_i})$. Then we call Lemma 6.3.6 to introduce a NAND$_t$-gadget. We identify its interface vertices with vertices c_{i,j_i} (in arbitrary order). This completes the construction of $(J_{\bar{j}}, L, (x^{\bar{j}}_1, \ldots, x^{\bar{j}}_t))$.

Suppose that there is a list homomorphism $h : (J_{\bar{j}}, L) \to H$ such that $h(x_{j_i}) = s^i_{j_i}$ for every $i \in [t]$. By the properties of $I(i,j)$-gadget, this implies that $h(c_{i,j_i}) = \beta$ for every $i \in [m]$. This in turn contradicts the fact that c_{i,j_i} are the interface vertices of a NAND$_m$-gadget, thus $(s^1_{j_1}, \ldots, s^t_{j_t}) \notin J(\bar{j})$. On the other hand, if $(s'_1, \ldots, s'_t) \in \mathcal{S}_1 \times \ldots \times \mathcal{S}_t \setminus \{(s^1_{j_1}, \ldots, s^t_{j_t})\}$, there exists $i \in [t]$ such that $s'_i \neq s^i_{j_i}$. We define $h : (J_{\bar{j}}, L) \to H$ as follows. Let $h : (I_{i,j_i}, L) \to H$ be such that $h(x^{\bar{j}}_{j_i}) = s'_i$ and $h(c_{i,j}) = \alpha$, and for every $i' \neq i$, let $h : (I_{i',j_{i'}}, L) \to H$ be such that $h(x_{j_{i'}}) = s^{i'}_{j_{i'}}$ and $h(c_{i',j}) = \beta$. The existence of such homomorphisms follows from the properties of I-gadgets. Now since at least one interface vertex of the list-NAND$_t$-gadget, i.e., c_{i,j_i} is mapped to α we can extend the homomorphism to the list-NAND$_t$-gadget. Therefore $J(\bar{j})$ satisfies the desired properties.

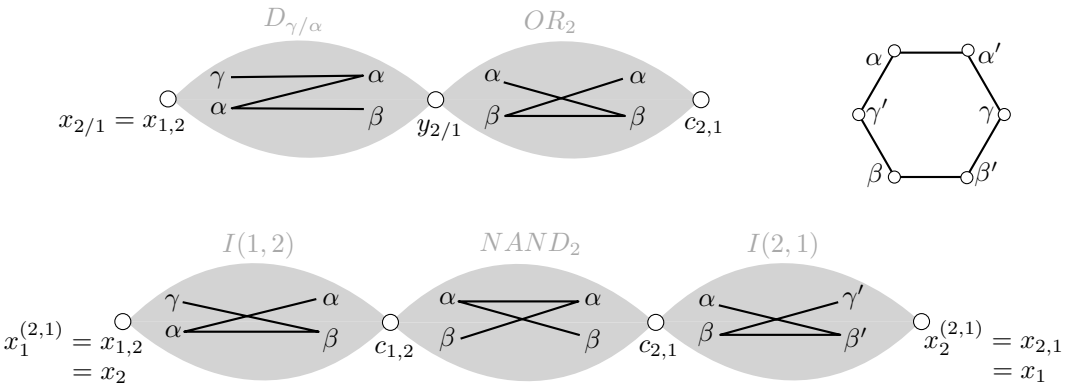


Figure 6.3: An example of gadgets built as in the proof of Lemma 6.3.4. Here, $H = C_6$, (upper right) $X = \{\alpha, \gamma, \beta\}$, $X = \{\alpha', \beta', \gamma'\}$, $\mathcal{S} = \{(\alpha, \gamma'), (\alpha, \beta'), (\gamma, \beta')\}$ and $\mathcal{R} = \{(\gamma, \gamma')\}$. A list-$I(1,2)$-gadget (upper left) and a list-$J(2,1)$-gadget which is the same as list-$\mathcal{S}$-gadget (bottom).

Step III. Finally we construct a list-$\mathcal{S}$-gadget. For every $\overline{j} = (j_1, \ldots, j_t)$ such that $(s^1_{j_1}, \ldots, s^t_{j_t}) \in \mathcal{R}$ we introduce a list-$J(\overline{j})$-gadget with interface vertices $(x^{\overline{j}}_{j_1}, \ldots, x^{\overline{j}}_{j_t})$. For every $i \in [t]$ and every suitable $\overline{j}$ we identify all vertices $x^{\overline{j}}_{j_i}$ into a single vertex x_i. This completes the construction of $(F, L, (x_1, \ldots, x_t))$.

Now observe that if there exists a list homomorphism $h : (F, L) \to H$ such that $(h(x_1), \ldots, h(x_t)) = (s^1_{j_1}, \ldots, s^t_{j_t}) \in \mathcal{R}$, this is a contradiction with the first property of the list-$J(\overline{j})$-gadget for $\overline{j} = (j_1, \ldots, j_t)$. On the other hand, if $(s^1_{j_1}, \ldots, s^t_{j_t}) \notin \mathcal{R}$, function $h(x_i) = s^i_{j_i}$ can be extended to every list-J-gadget we introduced, so there exists a homomorphism $h : (F, L) \to H$ such that $(h(x_1), \ldots, h(x_t)) = (s^1_{j_1}, \ldots, s^t_{j_t})$. This concludes the proof. □

An example of gadgets built as in the proof of Lemma 6.3.4 is depicted in Figure 6.3.

Constructing walks All that remains to complete the proof of Theorem 6.3.2 is to show Lemma 6.3.8. For that, we need a series of technical lemmas.

We first aim to show Lemma 6.1.1, i.e., that for any two incomparable vertices s, v in the same bipartition class X of H, and a third vertex $t \in X$, there exists a vertex $q \in X$ (possibly equal to s or v) such that we can either construct walks from s to t and from v to q, or walks from v to t and from s to q, which satisfy certain avoiding conditions. However, due to some corner cases, the proof is complicated, thus we split it into three steps (Lemma 6.3.10, Lemma 6.3.11, and finally the proof of Lemma 6.1.1).

Then, in the next three lemmas (Lemma 6.3.12, Lemma 6.3.13 and Lemma 6.3.14) we

show how we can use an obstruction in H to construct walks that avoid each other. Last, in Lemma 6.3.15 we show a special case of Lemma 6.3.8, and then prove Lemma 6.3.8 in full generality.

Lemma 6.3.10. *Let H be a connected bipartite graph with bipartition classes X and Y. Let $s, v \in X$ be incomparable vertices and let $s' \in N(s) \setminus N(v)$, $v' \in N(v) \setminus N(s)$. Let $S \subseteq N(s, s', v, v')$ be non-empty. Let U be the set of vertices reachable from $\{s, v\}$ in $H - S$.*

Then either H has a bipartite decomposition (D, N, R) such that $\{s, v, s', v'\} \subseteq D$ and $S \subseteq N$ or there exist $y \in S$ and $x \in U$ (both in the same bipartition class) and two pairs of walks of length $\ell \geqslant 1$ such that either

(1.) $\mathcal{A}, \mathcal{A}' : s \to y$ and $\mathcal{B}, \mathcal{B}' : v \to x$, or

(2.) $\mathcal{A}, \mathcal{A}' : v \to y$ and $\mathcal{B}, \mathcal{B}' : s \to x$,

and $\mathcal{A}$ avoids $\mathcal{B}$, $\mathcal{B}'$ avoids $\mathcal{A}'$. All four walks are entirely contained in $S \cup U$ and for every $i \in [\ell + 1]$ it holds that $\{\mathcal{A}, \mathcal{B}\}_i \not\subseteq S$.
Furthermore one of the following holds:

a) $x \in \{v, v'\}$ in case (1.) and $x \in \{s, s'\}$ in case (2.), or

b) walks $\mathcal{B}, \mathcal{B}'$ are entirely contained in U.

The decomposition of H or the walks $\mathcal{A}, \mathcal{A}', \mathcal{B}, \mathcal{B}'$ can be constructed in time polynomial in $|V(H)|$.

Proof. We split the proof into three cases.

Case 1: There exists $a \in N(s, v)$ and $b \in S \cap X$ such that $ab \notin E(H)$. By symmetry assume that $a \in N(s)$. If $b \in N(v')$, we set $x = s, y = b$, and define $\mathcal{A} = \mathcal{A}' = (v, v', b)$ and $\mathcal{B} = (s, s', s), \mathcal{B}' = (s, a, s)$, obtaining walks as in the statement (2.) a). On the other hand, if $b \notin N(v')$, then necessarily $b \in N(s')$. Then we take $x = v, y = b$, and define $\mathcal{A} = \mathcal{A}' = (s, s', b)$, and $\mathcal{B} = \mathcal{B}' = (v, v', v)$, which satisfy statement (1.) a).

Case 2: There exists $a \in S \cap Y$ and $b \in N(s', v')$ such that $ab \notin E(H)$. By symmetry assume that $b \in N(s')$. If $a \in N(v)$, then we set $x = s', y = a$, and define $\mathcal{A} = (v, a)$, $\mathcal{B} = (s, s')$, $\mathcal{A}' = (v, v', v, a)$, and $\mathcal{B}' = (s, s', b, s')$, obtaining statement (2.) a). On the other hand, if $a \notin N(v)$, then $a \in N(s)$. In this case we take $x = v', y = a$, and define $\mathcal{A} = \mathcal{A}' = (s, a)$ and $\mathcal{B} = \mathcal{B}' = (v, v')$, obtaining statement (1.) a).

Case 3: S is bipartite-complete to $N[s, s', v, v']$. Note that this is happens if and only if neither Case 1 nor Case 2 holds. In particular, $H[S]$ is a biclique.

Let K be the set of all vertices w in U such that there exists $s_w \in S$ belonging to the other bipartition class than w such that $ws_w \notin E(H)$. Since we are in Case 3, we observe that $K \cap N[s, v, s', v'] = \emptyset$.

If $K = \emptyset$, then U is bipartite-complete to S and there are no edges between U and $H - (S \cup U)$, so $\Big(U, S, V(H) \setminus (S \cup U)\Big)$ is a bipartite decomposition of H that satisfies the statement of the lemma. Therefore, we can assume that $K \neq \emptyset$. For contradiction, suppose that the lemma does not hold and H is a counterexample with the minimum number of vertices, i.e., for any graph H' with strictly fewer vertices and any choice of four vertices and a set, satisfying the assumptions of the lemma, the desired decomposition or walks exist.

Let $\widetilde{S}$ be a minimal $\{s, s', v, v'\}$-K-separator in $H[U]$ contained in $N(s, v, s', v')$, and let $\widetilde{U}$ be the set of vertices reachable from $\{s, s', v, v'\}$ in $H[U] - \widetilde{S}$. Note $s, s', v, v' \in U$, which in particular implies that the vertices s, v are incomparable in $H[U]$. Also, $\widetilde{U} \cup \widetilde{S}$ is bipartite-complete to S, and the set $\widetilde{S}$ is contained in $N(s, s', v, v') \cap U$.

Observe that the graph $H[U]$, vertices s, s', v, v', and the sets $\widetilde{U}$ and $\widetilde{S}$ satisfy the assumptions of the lemma (where the roles of U and S are played, respectively, by $\widetilde{U}$ and $\widetilde{S}$). Since $S \neq \emptyset$ and $S \subseteq V(H) - U$, we obtain that $H[U]$ has strictly fewer vertices than H. So, by minimality of H, Lemma 6.3.10 holds for $H[U]$.

Claim 6.3.10.1. *If $H[U]$ has a decomposition $(\widetilde{D}, \widetilde{N}, \widetilde{R})$, where $\{s, s', v, v'\} \subseteq \widetilde{D}$ and $\widetilde{S} \subseteq \widetilde{N}$, then H has a decomposition (D, N, R) such that $\{s, v, s', v'\} \subseteq D$ and $S \subseteq N$.*

Proof of Claim. Recall that $K \cap N[s, v, s', v'] = \emptyset$. By the properties of decomposition, $\widetilde{N} \subseteq N(s, v, s', v')$, so $\widetilde{N}$ is disjoint with K. Observe that $(\widetilde{D} \setminus K, \widetilde{N}, \widetilde{R} \cup K)$ is also a decomposition of $H[U]$ such that $\{s, s', v, v'\} \subseteq \widetilde{D}$ and $\widetilde{S} \subseteq \widetilde{N}$. Thus, we can assume that $\widetilde{D} \cap K = \emptyset$.

Note that $\widetilde{N}$ is bipartite-complete to S, implying that $S \cup \widetilde{N}$ is a biclique. Moreover, since $\widetilde{D} \cap K = \emptyset$, so by definition of K, $\widetilde{D}$ is bipartite-complete to $S \cup \widetilde{N}$, and there are no more vertices in H adjacent to $\widetilde{D}$. Then $(\widetilde{D}, \widetilde{N} \cup S, V(H) \setminus (\widetilde{D} \cup \widetilde{N} \cup S))$ is the decomposition of H, such that $s, s', v, v' \in \widetilde{D}$ and $S \subseteq (\widetilde{N} \cup S)$. ⌟

Thus we can assume that there is no decomposition of $H[U]$ that satisfies the statement of Claim 6.3.10.1, and there exist $\widetilde{x} \in \widetilde{U}, \widetilde{y} \in \widetilde{S}$ and one of the following quadruples of walks:

$$\begin{aligned} &\mathcal{C}, \mathcal{C}' : s \to \widetilde{y} \quad \text{and} \quad \mathcal{D}, \mathcal{D}' : v \to \widetilde{x} \quad \text{(statement (1.))}, \quad \text{or} \\ &\mathcal{C}, \mathcal{C}' : v \to \widetilde{y} \quad \text{and} \quad \mathcal{D}, \mathcal{D}' : s \to \widetilde{x} \quad \text{(statement (2.))}, \end{aligned}$$

where $\mathcal{C}$ avoids $\mathcal{D}$ and $\mathcal{D}'$ avoids $\mathcal{C}'$, all four walks are contained in $\widetilde{U} \cup \widetilde{S}$, andfor every $i \in [||\mathcal{C}||]$ we hafe $\{\mathcal{C}, \mathcal{D}\}_i \not\subseteq \widetilde{S}$. We show that in this case we can obtain the desired walks in H. Let us consider the first situation (statement (1.)), i.e., that we obtained walks $\mathcal{C}, \mathcal{C}' : s \to \widetilde{y}$ and $\mathcal{D}, \mathcal{D}' : v \to \widetilde{x}$ (the other one is symmetric).

For a walk $\mathcal{V}$ by $\mathcal{V}^{\vdash}$ we denote $\mathcal{V}$ with the last vertex removed. By minimality of $\widetilde{S}$ there exists a walk $\mathcal{P} : \widetilde{y} \to w$, where $w \in K$ and no vertex from $\mathcal{P}$ except $\widetilde{y}$ is in $\widetilde{S}$. This means that no vertex from $\mathcal{P}$, except for the first vertex, is adjacent to any vertex from $\widetilde{U}$, so, in particular, w is non-adjacent to $\widetilde{x}$. Observe that since $K \cap \widetilde{S} = \emptyset$ we must have $||\mathcal{P}|| \geqslant 1$. Let $w^{\bullet}$ be the last vertex of $\mathcal{P}^{\vdash}$. Note that we may have $w^{\bullet} = \widetilde{y}$.

Let $\widetilde{x}^{\bullet}$ be the last vertex of $(\mathcal{D}')^{\vdash}$. By Observation 5.1.2 applied to $\mathcal{D}', \mathcal{C}'$ we know that $\widetilde{x}^{\bullet}$ is adjacent to $\widetilde{x}$ and non-adjacent to $\widetilde{y}$. Moreover, since all vertices of $\mathcal{D}'$ are in $\widetilde{U} \cup \widetilde{S}$, we note that $\widetilde{x}^{\bullet}$ is adjacent to all vertices from the appropriate bipartition class of S.

Now we will separately consider two subcases, depending whether the call for $H[U]$ resulted in the statement (1.) a) or (1.) b).

Case 3 a): the call for $H[U]$ resulted in the statement (1.) a). This means that $\widetilde{x} \in \{v, v'\}$.

If $||\mathcal{P}|| = 1$, i.e., $\mathcal{P} = (\widetilde{y}, w)$, then we set $x = \widetilde{y}$, $y = s_w$, and define:

$$\begin{aligned} \mathcal{A} &= \mathcal{D}' \circ (\widetilde{x}, \widetilde{x}^{\bullet}, s_w) \qquad \mathcal{A}' = \mathcal{D} \circ (\widetilde{x}, \widetilde{x}^{\bullet}, s_w) \\ \mathcal{B} &= \mathcal{C}' \circ (\widetilde{y}, w, \widetilde{y}) \qquad \mathcal{B}' = \mathcal{C} \circ (\widetilde{y}, w, \widetilde{y}). \end{aligned} \tag{6.3}$$

Note that $s_w \in S$ is adjacent to $\widetilde{x}^\bullet \in \widetilde{U} \cup \widetilde{S}$, as in this case $\widetilde{y}, \widetilde{x}, s_w$ are in one bipartition class, while $\widetilde{x}^\bullet$ is in the other one, and $\widetilde{U} \cup \widetilde{S}$ is bipartite-complete to S. Observe that since $\mathcal{C}$ and $\mathcal{C}'$ are contained in $H[U]$, the walks satisfy the statement (2.) b).

So let us assume that $||\mathcal{P}|| > 1$, i.e., $w^\bullet \neq \widetilde{y}$. Define $\widetilde{x}'$ in a way that $\{\widetilde{x}, \widetilde{x}'\} = \{v, v'\}$. We define walks

$$\begin{aligned} \mathcal{R}' &= (\widetilde{x}, \widetilde{x}^\bullet, \widetilde{x}) \circ (\widetilde{x}, \widetilde{x}', \widetilde{x}, \ldots, \widetilde{x}'') \\ \mathcal{R} &= (\widetilde{x}, \widetilde{x}', \widetilde{x}) \circ (\widetilde{x}, \widetilde{x}', \widetilde{x}, \ldots, \widetilde{x}''), \end{aligned}$$

such that $||\mathcal{P}|| = ||\mathcal{R}|| = ||\mathcal{R}'||$, where $\widetilde{x}''$ is either $\widetilde{x}$ or $\widetilde{x}'$ (i.e., either v or v'), depending on the parity of $\mathcal{P}$. Note that $\widetilde{x}''$ is in the same bipartition class as w. We observe that $\mathcal{P}$ avoids $\mathcal{R}'$: recall that $\widetilde{y}$ is non-adjacent to $\widetilde{x}^\bullet$, and all other vertices of $\mathcal{P}$ are in $U \setminus (\widetilde{S} \cup \widetilde{U})$, and thus they are non-adjacent to $\{v, v'\} = \{\widetilde{x}, \widetilde{x}'\}$. Moreover, the latter implies that $\mathcal{R}$ avoids $\mathcal{P}$. We set $x = w^\bullet$, $y = s_w$ and:

$$\begin{aligned} \mathcal{A} &= \mathcal{D}' \circ \mathcal{R} \circ (\widetilde{x}'', s_w) & \mathcal{A}' &= \mathcal{D} \circ \mathcal{R}' \circ (\widetilde{x}'', s_w) \\ \mathcal{B} &= \mathcal{C}' \circ \mathcal{P} \circ (w, w^\bullet) & \mathcal{B}' &= \mathcal{C} \circ \mathcal{P} \circ (w, w^\bullet). \end{aligned} \tag{6.4}$$

Recall that $\widetilde{x}'' \in \{v, v'\} \subseteq \widetilde{U}$ and as S is bipartite-complete to $\widetilde{U}$, we have that $\widetilde{x}''$ is adjacent to $s_w \in S$. Moreover, we observe that $\mathcal{A}$ avoids $\mathcal{B}$ and $\mathcal{B}'$ avoids $\mathcal{A}'$, as $w^\bullet$ is in $U \setminus (\widetilde{S} \cup \widetilde{U})$ and thus is non-adjacent to $\widetilde{x}'' \in \widetilde{U}$. It is straightforward to verify that the constructed walks satisfy the statement (2.) b).

Case 3 b): the call for $H[U]$ resulted in the statement (1.) b). This means that $\mathcal{D}, \mathcal{D}'$ are entirely contained in $\widetilde{U}$, in particular $\widetilde{x}, \widetilde{x}^\bullet \in \widetilde{U}$. We define $\mathcal{R} = (\widetilde{x}, \widetilde{x}^\bullet, \widetilde{x}, \ldots, \widetilde{x}'')$ so that $||\mathcal{R}|| = ||\mathcal{P}||$, where $\widetilde{x}''$ is either $\widetilde{x}$ or $\widetilde{x}^\bullet$, depending on the parity of $||\mathcal{P}||$. Note that $\mathcal{P}$ and $\mathcal{R}$ avoid each other, as $\widetilde{y}$ is non-adjacent to $\widetilde{x}^\bullet$, and no vertex from $\mathcal{P}$, except for $\widetilde{y}$, is adjacent to any vertex of $\widetilde{U}$.

We set $x = w^\bullet$, $y = s_w$ and:

$$\begin{aligned} \mathcal{A} &= \mathcal{D}' \circ \mathcal{R} \circ (\widetilde{x}'', s_w), & \mathcal{A}' &= \mathcal{D} \circ \mathcal{R} \circ (\widetilde{x}'', s_w), \\ \mathcal{B} &= \mathcal{C}' \circ \mathcal{P} \circ (w, w^\bullet), & \mathcal{B}' &= \mathcal{C} \circ \mathcal{P} \circ (w, w^\bullet). \end{aligned} \tag{6.5}$$

Similarly to the case of walks constructed in (6.4), we can observe that $\mathcal{A}$ avoids $\mathcal{B}$ and

$\mathcal{B}'$ avoids $\mathcal{A}'$. Note that this works also for the case $w^\bullet = \widetilde{y}$, that is, if $||\mathcal{P}|| = 1$. Then the walks from (6.5) are as follows:

$$\mathcal{A} = \mathcal{D}' \circ (\widetilde{x}, \widetilde{x}^\bullet, s_w), \quad \mathcal{A}' = \mathcal{D} \circ (\widetilde{x}, \widetilde{x}^\bullet, s_w),$$
$$\mathcal{B} = \mathcal{C}' \circ (\widetilde{y}, w, \widetilde{y}), \qquad \mathcal{B}' = \mathcal{C} \circ (\widetilde{y}, w, \widetilde{y}),$$

Observe that in (6.5) we used only vertices from U (except s_w) and thus walks $\mathcal{B}, \mathcal{B}'$ are entirely contained in U. Moreover, for every $i \in [||\mathcal{A}||]$ it holds that $\{\mathcal{A}, \mathcal{B}\}_i \not\subseteq S$. Thus we obtain the statement (2.) b) of the lemma.

Finally, the proof is clearly constructive, so the claim about the polynomial-time algorithm follows easily. □

Using Lemma 6.3.10 we can derive the following statement.

Lemma 6.3.11. *Let H be a connected, undecomposable, bipartite graph with bipartition classes X, Y and let $s, v \in X$ be incomparable vertices. Let $T \subseteq X$ be a non-empty set of vertices that are incomparable with both s, v. Then there exist $t \in T$ and two pairs of walks:*

1. *$\mathcal{A}, \mathcal{A}' : s \to t$ and $\mathcal{B}, \mathcal{B}' : v \to v$, or*

2. *$\mathcal{A}, \mathcal{A}' : s \to s$ and $\mathcal{B}, \mathcal{B}' : v \to t$,*

such that $\mathcal{A}$ avoids $\mathcal{B}$ and $\mathcal{B}'$ avoids $\mathcal{A}'$. Walks $\mathcal{A}, \mathcal{A}', \mathcal{B}, \mathcal{B}'$ can be constructed in time polynomial in $|V(H)|$.

Proof. Since s and v are incomparable, there exist $v' \in N(v) \setminus N(s)$ and $s' \in N(s) \setminus N(v)$.

Case 1: there exists $t \in T$ and either a vertex $t'_v \in N(t) \setminus N(v)$, which is adjacent to s, or a vertex $t'_s \in N(t) \setminus N(s)$, which is adjacent to v. Assume, by symmetry, that t'_v exists. Let $v'_t \in N(v) \setminus N(t)$, it exists since v and t are incomparable. Then we can set: $\mathcal{A} = \mathcal{A}' = (s, t'_v, t), \mathcal{B} = (v, v', v)$ and $\mathcal{B}' = (v, v'_t, v)$. It is straightforward to verify that $\mathcal{A}$ avoids $\mathcal{B}$ and $\mathcal{B}'$ avoids $\mathcal{A}'$. The case that t'_s exists is symmetric.

Case 2: for every $t \in T$, every neighbor of t is either adjacent to both s, v or non-adjacent to both. Observe that this implies that $T \cap N(s', v') = \emptyset$: otherwise, if there is any $t \in T$ adjacent to one of s', v', say s', then s' must be adjacent to both s, v, which contradicts its definition.

Let S be a minimal $\{s, v, s', v'\}$-T-separator contained in $N(s, v, s', v')$ and let U be the set of vertices reachable from $\{s, s', v, v'\}$ in $H - S$. Observe that the graph H, vertices s, s', v, v', and the set S satisfy the assumptions of Lemma 6.3.10. So, since H is undecomposable, by Lemma 6.3.10, there are $y \in S$ and $x \in U$, and walks $\mathcal{C}, \mathcal{C}' : s \to y, \mathcal{D}, \mathcal{D}' : v \to x$ (statement (1.)) or $\mathcal{C}, \mathcal{C}' : v \to y, \mathcal{D}, \mathcal{D}' : s \to x$ (statement (2.)), such that $\mathcal{C}$ avoids $\mathcal{D}$ and $\mathcal{D}'$ avoids $\mathcal{C}'$.

Suppose that calling Lemma 6.3.10 resulted in statement (1.), i.e., the obtained walks are $\mathcal{C}, \mathcal{C}' : s \to y$ and $\mathcal{D}, \mathcal{D}' : v \to x$ (the other case is symmetric). Let $x^\bullet$ be the last vertex on $(\mathcal{D}')^\vdash$, note that it is adjacent to x and, by Observation 5.1.2, non-adjacent to y. By minimality of S there is $t \in T$ and a y-t-path $\mathcal{P}$ of length at least 1, whose every vertex, except for y, is in $V(H) \setminus (S \cup U)$. This implies that y is the only vertex of $\mathcal{P}$ with a neighbor in U. Let us consider two subcases.

Case 2a: calling Lemma 6.3.10 resulted in statement (1.) a). This means that $x \in \{v, v'\}$. Let us define x', such that $\{x, x'\} = \{v, v'\}$. If $||\mathcal{P}|| > 1$, we define $\mathcal{R} = (x, x^\bullet, x, x', x, \ldots, v)$ and $\mathcal{R}' = (x, x', x, x', x, \ldots, v)$, such that $||\mathcal{R}|| = ||\mathcal{R}'|| = ||\mathcal{P}||$. Recall that v and t are in the same bipartition class, so the last vertex of both walks is indeed v. Observe that $\mathcal{P}$ avoids $\mathcal{R}$ and $\mathcal{R}'$ avoids $\mathcal{P}$, as $x^\bullet$ is non-adjacent to y, and every vertex of $\mathcal{P}$, except y, is non-adjacent to every vertex of U, so, in particular, to $\{x, x'\} = \{v, v'\}$. We set:

$$\begin{aligned} \mathcal{A} &= \mathcal{C} \circ \mathcal{P}, \quad \mathcal{A}' = \mathcal{C}' \circ \mathcal{P}, \\ \mathcal{B} &= \mathcal{D} \circ \mathcal{R}, \quad \mathcal{B}' = \mathcal{D}' \circ \mathcal{R}', \end{aligned}$$

which are walks as in the statement (1.). And if $||\mathcal{P}|| = 1$, i.e. $\mathcal{P} = (y, t)$, then $x = v'$ since x and y are in the same bipartition class. We can define $\mathcal{R} = (v', x^\bullet, v', v)$, $\mathcal{R}' = (v', v, v', v)$ and $\mathcal{P}' = (y, t, t', t)$, where t' is a vertex in $N(t) \setminus N(s, v)$ (note that it exists, since t is incomparable with s, v and every vertex in $N(t) \cap N(s, v)$ is adjacent to both s, v by the definition of Case 2). Note that $\mathcal{R}'$ avoids $\mathcal{P}'$ and $\mathcal{P}'$ avoids $\mathcal{R}$. We set:

$$\begin{aligned} \mathcal{A} &= \mathcal{C} \circ \mathcal{P}', \quad \mathcal{A}' = \mathcal{C}' \circ \mathcal{P}', \\ \mathcal{B} &= \mathcal{D} \circ \mathcal{R}, \quad \mathcal{B}' = \mathcal{D}' \circ \mathcal{R}', \end{aligned}$$

and obtain walks as in the statement (1.).

Case 2b: calling Lemma 6.3.10 resulted in statement (1.) b). This means that walks $\mathcal{D}, \mathcal{D}'$ are entirely contained in U. Let t' be a neighbor of t such that $t' \notin S \cap Y$, again it exists in this case (recall that $S \cap Y \subseteq N(s, v)$). We set:

$$\mathcal{A} = \mathcal{C} \circ \mathcal{P}^*, \quad \mathcal{A}' = \mathcal{C}' \circ \mathcal{P}^*,$$
$$\mathcal{B} = \mathcal{D} \circ \mathcal{D}^*, \quad \mathcal{B}' = \mathcal{D}' \circ \mathcal{D}^*,$$

where $\mathcal{P}^* := \mathcal{P} \circ (t, t', \ldots, t)$ and $\mathcal{D}^* := \overline{\mathcal{D}'} \circ (v, v', \ldots, v)$ are defined in a way that $||\mathcal{P}^*|| = ||\mathcal{D}^*|| := \max(||\mathcal{P}||, ||\mathcal{D}'||)$ (basically, these walks play the same role as $\mathcal{P}$ and $\overline{\mathcal{D}'}$, but extra padding added to one of them ensures they have the same length). Observe that $\mathcal{D}^*$ is entirely contained in U and thus it avoids $\mathcal{P}^*$ whose only vertex adjacent to U is the first one. Moreover $\mathcal{P}^*$ avoids $\mathcal{D}^*$, since the first vertex of $\mathcal{P}$ is y and it is non-adjacent to $x^\bullet$, which is the second vertex on $\mathcal{D}^*$. Thus $\mathcal{A}$ avoids $\mathcal{B}$ and $\mathcal{B}'$ avoids $\mathcal{A}'$ and we have obtained statement (1.).

Again, note that walks $\mathcal{A}, \mathcal{A}', \mathcal{B}, \mathcal{B}'$ can be constructed in time polynomial in $|V(H)|$.□

We have all tools to prove Lemma 6.1.1 that we recall here.

Lemma 6.1.1. *Let H be a connected, bipartite undecomposable graph with bipartition classes X, Y, let $\{s, v\} \in X$ be incomparable, and let $t \in X$. Then there exist a vertex $q \in X$ and two pairs of walks:*

1. *$\mathcal{P}, \mathcal{P}' : s \to t$ and $\mathcal{Q}, \mathcal{Q}' : v \to q$, or*
2. *$\mathcal{P}, \mathcal{P}' : s \to q$ and $\mathcal{Q}, \mathcal{Q}' : v \to t$,*

such that $\mathcal{P}$ avoids $\mathcal{Q}$ and $\mathcal{Q}'$ avoids $\mathcal{P}'$. Moreover, if t is incomparable with at least one of s, v then $q = v$ in the first case and $q = s$ in the other.

Finally, given a bipartite graph H and $s, v, t \in X$, in time polynomial in $|V(H)|$ we can either find the desired walks or a decomposition of H.

Proof of Lemma 6.1.1. Observe that if t is incomparable with both s, v, we can obtain the desired walks by applying Lemma 6.3.11 for s, v, and $T = \{t\}$.

Now let us assume that t is incomparable with exactly one of s, v, say v (the other case is symmetric) and let t'_v be a vertex in $N(t) \setminus N(v)$ and let v'_t be a vertex in $N(v) \setminus N(t)$.

Since s and v are incomparable, there are $v'_s \in N(v) \setminus N(s)$ and $s'_v \in N(s) \setminus N(v)$. Since t is comparable with s, we either have $N(t) \subseteq N(s)$ or $N(s) \subseteq N(t)$. If $N(t) \subseteq N(s)$, we observe that t'_v must be adjacent to s, and v'_s must be non-adjacent to t since $v'_s \notin N(s)$. Then we set $q = v$, $\mathcal{P} = \mathcal{P}' = (s, t'_v, t)$, and $\mathcal{Q} = \mathcal{Q}' = (v, v'_s, v)$. On the other hand, if $N(s) \subseteq N(t)$, we observe that s'_v must be adjacent to t and v'_t must be non-adjacent to s. Then we set $q = v$, $\mathcal{P} = \mathcal{P}' = (s, s'_v, t)$, and $\mathcal{Q} = \mathcal{Q}' = (v, v'_t, v)$.

So from now we can assume that t is comparable with both s, v. Observe that it implies that either $N(s, v) \subseteq N(t)$, or $N(t) \subseteq N(s) \cap N(v)$. Note that the other cases are not possible: if we have, say, $N(v) \subseteq N(t)$ and $N(t) \subseteq N(s)$, then $N(v) \subseteq N(s)$, a contradiction. Observe that if there is $s' \in N(s) \setminus N(v)$ and $v' \in N(v) \setminus N(s)$, such that t is adjacent to at least one of s', v', then we can set $S := \{t\} \subseteq N(s, s', v, v')$ and, by the assumption of H, it has no decomposition. Thus we can call Lemma 6.3.10 for vertices s, s', v, v' and the set $S = \{t\}$. Since t is the only vertex of S, we obtain a vertex q, reachable from $\{s, v\}$ in $H - \{t\}$, and walks $\mathcal{P}, \mathcal{P}' : s \to t$ and $\mathcal{Q}, \mathcal{Q}' : v \to q$, or $\mathcal{Q}, \mathcal{Q}' : v \to t$ and $\mathcal{P}, \mathcal{P}' : s \to q$, such that $\mathcal{P}$ avoids $\mathcal{Q}$ and $\mathcal{Q}'$ avoids $\mathcal{P}'$.

So it only remains to consider the case in which t is non-adjacent to every vertex in $\Big(N(s) \setminus N(v)\Big) \cup \Big(N(v) \setminus N(s)\Big)$. Choose arbitrary $s' \in N(s) \setminus N(v)$ and $v' \in N(v) \setminus N(s)$. Observe that by the case we are considering, we have $N(t) \subseteq N(s) \cap N(v)$, which implies that $N(t) \subseteq N(s, s', v, v')$. We call Lemma 6.3.10 for H, vertices s, s', v, v', and the set $S = N(t)$. We obtain vertices $y \in N(t)$ and $x \in U$, where U is the set of vertices reachable from $\{s, v\}$ in $H - S$, and walks $\mathcal{A}, \mathcal{A}' : s \to y, \mathcal{B}, \mathcal{B}' : v \to x$ or $\mathcal{A}, \mathcal{A}' : v \to y, \mathcal{B}, \mathcal{B}' : s \to x$ of length at least one, such that $\mathcal{A}$ avoids $\mathcal{B}$ and $\mathcal{B}'$ avoids $\mathcal{A}'$. Let q be the last vertex on $(\mathcal{B}')^{\vdash}$, note that q is adjacent to x and non-adjacent to y. We also have that x is non-adjacent to t, since all neighbors of t are in S. If we obtain walks $\mathcal{A}, \mathcal{A}' : s \to y, \mathcal{B}, \mathcal{B}' : v \to x$ then we set:

$$\mathcal{P} = \mathcal{A} \circ (y, t), \quad \mathcal{P}' = \mathcal{A}' \circ (y, t),$$
$$\mathcal{Q} = \mathcal{B} \circ (x, q), \quad \mathcal{Q}' = \mathcal{B}' \circ (x, q),$$

and if we obtain walks $\mathcal{A}, \mathcal{A}' : v \to y, \mathcal{B}, \mathcal{B}' : s \to x$, then we set:

$$\mathcal{P} = \mathcal{B}' \circ (x, q), \quad \mathcal{P}' = \mathcal{B} \circ (x, q),$$
$$\mathcal{Q} = \mathcal{A}' \circ (y, t), \quad \mathcal{Q}' = \mathcal{A} \circ (y, t).$$

It is straightforward to verify that in both cases $\mathcal{P}$ avoids $\mathcal{Q}$ and $\mathcal{Q}'$ avoids $\mathcal{P}'$.

Observe that the proof is again constructive, so the claim about a polynomial-time algorithm follows. □

Now, having proven Lemma 6.1.1, we analyze how our walks can reach the obstruction.

Lemma 6.3.12. *Let H be a connected, undecomposable, bipartite graph containing an obstruction $\mathbb{O}$. Let $(\alpha, \beta) \in C(\mathbb{O})$, and let s be a vertex from the same bipartition class as β but incomparable with it. Then there exist walks $\mathcal{A}, \mathcal{A}' : s \to \alpha$ and $\mathcal{B}, \mathcal{B}' : \beta \to \beta$, such that $\mathcal{A}$ avoids $\mathcal{B}$ and $\mathcal{B}'$ avoids $\mathcal{A}'$.*

Proof. We use Lemma 6.1.1 for $s = \alpha$, $v = \beta$, and $t = s$ to obtain walks $\mathcal{P}, \mathcal{P}' : \alpha \to s$ and $\mathcal{Q}, \mathcal{Q}' : \beta \to \beta$ or $\mathcal{P}, \mathcal{P}' : \alpha \to \alpha$ and $\mathcal{Q}, \mathcal{Q}' : \beta \to s$, such that $\mathcal{P}$ avoids $\mathcal{Q}$ and $\mathcal{Q}'$ avoids $\mathcal{P}'$. In the first case, it is enough to take $\mathcal{A} = \overline{\mathcal{P}'}$, $\mathcal{B} = \overline{\mathcal{Q}'}$, $\mathcal{A}' = \overline{\mathcal{P}}$, $\mathcal{B}' = \overline{\mathcal{Q}}$, since, by Observation 5.1.2, $\overline{\mathcal{P}'}$ avoids $\overline{\mathcal{Q}'}$ and $\overline{\mathcal{Q}}$ avoids $\overline{\mathcal{P}}$. In the second case, we use Lemma 6.1.1 again, now for $s = s$, $v = \beta$ and $t = \alpha$ and obtain walks $\mathcal{R}, \mathcal{R}' : s \to \alpha$ and $\mathcal{S}, \mathcal{S}' : \beta \to \beta$ or $\mathcal{R}, \mathcal{R}' : s \to s$ and $\mathcal{S}, \mathcal{S}' : \beta \to \alpha$, such that $\mathcal{R}$ avoids $\mathcal{S}$ and $\mathcal{S}'$ avoids $\mathcal{R}'$. In the first case, we are done, as these are the walks we are looking for. In the second case we set

$$\mathcal{A} = \mathcal{R} \circ \overline{\mathcal{Q}} \circ \mathcal{Y}', \quad \mathcal{A}' = \mathcal{R}' \circ \overline{\mathcal{Q}'} \circ \mathcal{Y},$$
$$\mathcal{B} = \mathcal{S} \circ \overline{\mathcal{P}} \circ \mathcal{X}', \quad \mathcal{B}' = \mathcal{S}' \circ \overline{\mathcal{P}'} \circ \mathcal{X},$$

where $\mathcal{X}, \mathcal{X}' : \alpha \to \beta$ and $\mathcal{Y}, \mathcal{Y}' : \beta \to \alpha$ are obtained by Observation 5.1.9. □

In the next proof some constructions will depend on the bipartition class of a particular vertex. To avoid considering cases separately, for two non-necessarily distinct vertices u and u', such that $u' \in N[u]$, we introduce a walk

$$\mathcal{E}[u, u'] := \begin{cases} (u, u') & \text{if } uu' \in E(H), \\ (u) & \text{if } u = u'. \end{cases}$$

Note that if sets $\{u, u'\}$ and $\{v, v'\}$ are non-adjacent (again, it might be that $u = u'$ or $v = v'$), then the walks $\mathcal{E}[u, u']$ and $\mathcal{E}[v, v']$ are clearly non-adjacent.

Lemma 6.3.13. *Let H be a bipartite graph with an obstruction $\mathbb{O}$. Let $(\alpha, \beta) \in C(\mathbb{O})$ and let $s \in V(H)$ be a vertex that belongs to the same bipartition class as α, β. Assume that $N(s) \setminus N(\beta) \neq \emptyset$ and $\operatorname{dist}(N[s] \setminus N(\beta), \mathbb{O}) \leqslant 1$. Then one of the following exists:*

1. *walks $\mathcal{P} : \alpha \to \alpha, \mathcal{Q} : s \to \alpha$ and $\mathcal{R} : \beta \to \beta$, such that $\mathcal{P}, \mathcal{Q}$ avoid $\mathcal{R}$,*

2. *walks $\mathcal{P} : \alpha \to \alpha, \mathcal{Q} : s \to \beta$ and $\mathcal{R} : \beta \to \beta$, such that $\mathcal{P}$ avoids $\mathcal{Q}, \mathcal{R}$.*

Proof. Let α', β' be the pair of vertices, such that $C(\mathbb{O}) = \{(\alpha, \beta), (\alpha', \beta')\}$.

By Definition 5.1.7 recall that edges $\alpha\alpha'$ and $\beta\beta'$ are independent. First, we consider some simple special cases separately. If $s\beta' \in E(H)$, we obtain the statement (2.) by setting

$$\begin{aligned}\mathcal{P} &= (\alpha, \alpha'\alpha),\\ \mathcal{Q} &= (s, \beta', \beta),\\ \mathcal{R} &= (\beta, \beta', \beta).\end{aligned}$$

So now $s\beta' \notin E(H)$, and if there exists $s' \in N(s) \setminus N(\beta)$ such that $\alpha s' \in E(H)$ (in particular when s is adjacent to α'), then we obtain the statement (1.) by setting

$$\begin{aligned}\mathcal{P} &= (\alpha, \alpha', \alpha),\\ \mathcal{Q} &= (s, s', \alpha),\\ \mathcal{R} &= (\beta, \beta', \beta).\end{aligned}$$

Thus we can assume that the cases above do not apply. Let us fix any $s' \in N(s) \setminus N(\beta)$ for which $\operatorname{dist}(\{s, s'\}, \mathbb{O}) \leqslant 1$. Note that the edges $\alpha\alpha', \beta\beta'$, and ss' are independent (i.e., they form an induced matching).

Case 1: there exists $p \in N[s, s'] \cap V(\mathbb{O})$ such that $p \notin N(\alpha, \alpha')$. Clearly $p \notin \{\alpha, \alpha', \beta, \beta'\}$, as otherwise edges $\alpha\alpha', \beta\beta', ss'$ would not be independent. This, in particular, cannot occur if $\mathbb{O}$ is isomorphic to C_6, see Figure 6.4 (left). Let $\overline{s}$ be an element of $\{s, s'\}$ which is a neighbor of p. Recall that since $p, \beta \notin N[\alpha, \alpha']$, by Observation 5.1.8,

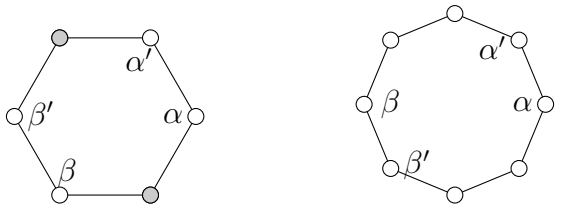


Figure 6.4: The position of elements of $C(\mathbb{O})$ when $\mathbb{O}$ is isomorphic to C_6 (left) of C_8 (right). Gray vertices indicate the possible position of vertices p and r in the Case 2 of the proof of Lemma 6.3.13.

there exists a p-β-walk $\mathcal{W}[p,\beta]$ in $\mathbb{O}$ which is non-adjacent to $\{\alpha,\alpha'\}$. We define:

$$\begin{aligned}\mathcal{P} &:= (\alpha,\alpha',\alpha,\ldots,\alpha),\\ \mathcal{Q} &:= \mathcal{E}[s,\overline{s}]\circ(\overline{s},p)\circ\mathcal{W}[p,\beta],\\ \mathcal{R} &:= (\beta,\beta',\beta,\ldots,\beta),\end{aligned}$$

in a way that $||\mathcal{P}|| = ||\mathcal{R}|| = ||\mathcal{Q}||$. Clearly $\mathcal{P}$ avoids $\mathcal{R}$. Moreover, $\mathcal{P}$ avoids $\mathcal{Q}$ since $s,s',p \notin N[\alpha,\alpha']$ and $\mathcal{W}[p,\beta]$ is non-adjacent to $\{\alpha,\alpha'\}$. We obtain the statement (2.).

Case 2: $N[s,s'] \cap V(\mathbb{O}) \subseteq N(\alpha,\alpha')$**.** It implies that $\mathrm{dist}(\{s,s'\},\mathbb{O}) = 1$, as if $\{s,s'\} \cap V(\mathbb{O}) \neq \emptyset$, then $\{s,s'\}$ and $\{\alpha,\alpha'\}$ would be adjacent. So there exists $p \in N(s,s') \cap V(\mathbb{O}) \cap N(\alpha,\alpha')$. Again, let $\overline{s}$ be an element of $\{s,s'\}$ which is a neighbor of p and let $\{\overline{\alpha},\overline{\alpha}'\} := \{\alpha,\alpha'\}$ such that $\overline{\alpha}p \in E(H)$. Note that if $p \notin N(\beta,\beta')$, we can set

$$\begin{aligned}\mathcal{P} &:= (\alpha,\alpha',\ldots,\alpha),\\ \mathcal{Q} &:= \mathcal{E}[s,\overline{s}]\circ(\overline{s},p,\overline{\alpha})\circ\mathcal{E}[\overline{\alpha},\alpha]\\ \mathcal{R} &:= (\beta,\beta',\ldots,\beta).\end{aligned}$$

in a way that $||\mathcal{P}|| = ||\mathcal{R}|| = ||\mathcal{Q}||$. Then $\mathcal{P},\mathcal{Q}$ avoid $\mathcal{R}$ and we get the statement (2.). Observe that if $\mathbb{O}$ is isomorphic to C_8, then the above case applies; see Figure 6.4 (right).

So we can assume that $p \in N(s,s') \cap N(\alpha,\alpha') \cap N(\beta,\beta')$. Let $\{\overline{\beta},\overline{\beta}'\} := \{\beta,\beta'\}$, such that $\overline{\beta}p \in E(H)$. If $\mathbb{O}$ is isomorphic to C_6, then let r be the other vertex of $\mathbb{O}$ which belongs to $N(\alpha,\alpha') \cap N(\beta,\beta')$ (r is uniquely determined, see Figure 6.4 (left)). As

$pr \notin E(H)$, we define the walks that satisfy the statement (1.):

$$\begin{aligned}\mathcal{P} &:= \mathcal{E}[\alpha, \overline{\alpha}] \circ (\overline{\alpha}, p, \overline{\alpha}, \overline{\alpha}') \circ \mathcal{E}[\overline{\alpha}', \alpha]\\ \mathcal{Q} &:= \mathcal{E}[s, \overline{s}] \circ (\overline{s}, p, \overline{\alpha}, \overline{\alpha}') \circ \mathcal{E}[\overline{\alpha}', \alpha],\\ \mathcal{R} &:= \mathcal{E}[\beta, \overline{\beta}] \circ (\overline{\beta}, \overline{\beta}', r, \overline{\beta}') \circ \mathcal{E}[\overline{\beta}', \beta].\end{aligned}$$

So finally consider the case that $\mathbb{O}$ is an asteroidal subgraph. We use the notation introduced in Definition 5.1.4. Then $p \in \mathcal{W}_{0,i}$ for some $i \in \{1, 2k\}$, as all other vertices of $\mathbb{O}$ are non-adjacent to α. Let $\{\kappa, \kappa'\} = \{u_{k+1}, v_{k+1}\}$ if $i = 1$ or $\{\kappa, \kappa'\} = \{u_k, v_k\}$ if $i = 2k$, where κ' and p are in the same bipartition class. Note that by the definition of a special edge asteroid we know that the walk $\mathcal{W}_{0,i}$, and in particular p, is non-adjacent to $\{\kappa, \kappa'\}$. Recall that the walk $\mathcal{W}[\beta, \kappa]$ given by Observation 5.1.8 is non-adjacent to $\{\alpha, \alpha'\} = \{u_0, v_0\}$. So, since we are in Case 2, we observe that s and s' are non-adjacent to $\{\kappa, \kappa'\}$ and to $\mathcal{W}[\beta, \kappa]$, as otherwise we would have chosen $p \in \{\kappa, \kappa'\} \cup \mathcal{W}[\beta, \kappa]$, such that $p \in N[s, s'] \cap V(\mathbb{O})$ and $p \notin N(\alpha, \alpha')$, ending up in Case 1. We set

$$\begin{aligned}\mathcal{P} &:= (\alpha, \alpha', \ldots, \alpha) \circ \mathcal{E}[\alpha, \overline{\alpha}] \circ (\overline{\alpha}, \overline{\alpha}', \overline{\alpha}) \circ \mathcal{E}[\overline{\alpha}, \alpha] \circ (\alpha, \alpha', \ldots, \alpha),\\ \mathcal{Q} &:= (s, s', \ldots, s) \circ \mathcal{E}[s, \overline{s}] \circ (\overline{s}, p, \overline{\alpha}) \circ \mathcal{E}[\overline{\alpha}, \alpha] \circ (\alpha, \alpha', \ldots, \alpha),\\ \mathcal{R} &:= \mathcal{W}[\beta, \kappa] \circ (\kappa, \kappa', \kappa) \circ \mathcal{W}[\kappa, \beta],\end{aligned}$$

so that the lengths of the subwalks in each aligned column are equal. Clearly, $\mathcal{R}$ is non-adjacent to both $\mathcal{P}$ and $\mathcal{Q}$, so we obtain the statement (1). □

Lemma 6.3.14. *Let H be a connected, undecomposable, bipartite graph with an obstruction $\mathbb{O}$, let $(\alpha, \beta) \in C(\mathbb{O})$ and let $s \in V(H)$ be a vertex that belongs to the same bipartition class as β, but is incomparable with it. Then at least one of the following exists:*

1. *walks $\mathcal{P} : \alpha \to \alpha, \mathcal{Q} : s \to \alpha$ and $\mathcal{R} : \beta \to \beta$, such that $\mathcal{P}, \mathcal{Q}$ avoid $\mathcal{R}$,*
2. *walks $\mathcal{P} : \alpha \to \alpha, \mathcal{Q} : s \to \beta$ and $\mathcal{R} : \beta \to \beta$, such that $\mathcal{P}$ avoids $\mathcal{Q}, \mathcal{R}$.*

Proof. Let (α', β') be the pair of vertices, such that $C(\mathbb{O}) = \{(\alpha, \beta), (\alpha', \beta')\}$. Let X, Y be the bipartition classes of H, such that $\alpha, \beta \in X$ and $\alpha', \beta' \in Y$. If $\operatorname{dist}(N[s] \setminus N(\beta), \mathbb{O}) \leqslant 1$, we are done by Lemma 6.3.13. Thus let us assume that $\operatorname{dist}(N[s] \setminus N(\beta), \mathbb{O}) \geqslant 2$. Fix any $s' \in N(s) \setminus N(\beta)$ and observe that edges $\alpha\alpha', \beta\beta'$ and ss' are independent. Let P be

the minimal $\{s\}$-$\mathbb{O}$-separator contained in $N(\mathbb{O})$. Let R be the set of vertices reachable from s in $H \setminus P$ and let Q be the set of vertices reachable from α, β in $H \setminus P$. Note that $V(\mathbb{O}) \subseteq Q$ and $s' \in R$, as it does not belong to $N(\mathbb{O})$.

Case 1: there exists $p \in P$ such that $p \notin N(\alpha, \beta, \alpha', \beta')$. Denote by $\mathcal{S}_p$ the s-p-walk, such that $\mathcal{S}_p^{\vdash}$ is contained in R. The walk $\mathcal{S}_p$ exists, because P is a minimal s-$\mathbb{O}$-separator. Let $p^\bullet \in R$ be the last vertex of $\mathcal{S}_p^{\vdash}$ (it is possible that $p^\bullet = s$).

If $p \in P \cap X$, then clearly $p \in N(\mathbb{O}) \setminus N(\beta)$, so $\operatorname{dist}(N[p] \setminus N(\beta), \mathbb{O}) \leqslant 1$. Moreover, $N(p) \setminus N(\beta)$ is non-empty, because $p^\bullet \in N(p) \setminus N(\beta)$. So by Lemma 6.3.13 we can obtain walks $\mathcal{T} : \alpha \to \alpha$, $\mathcal{U} : p \to \lambda$ for $\lambda \in \{\alpha, \beta\}$, and $\mathcal{V} : \beta \to \beta$ such that if $\mathcal{U}$ is a p-α-walk, then $\mathcal{T}, \mathcal{U}$ avoid $\mathcal{V}$, and otherwise $\mathcal{T}$ avoids $\mathcal{U}, \mathcal{V}$. Then we define $\mathcal{P} := (\alpha, \alpha', \ldots, \alpha) \circ \mathcal{T}$, $\mathcal{Q} := \mathcal{S}_p \circ \mathcal{U}$, and $\mathcal{R} := (\beta, \beta', \ldots, \beta) \circ \mathcal{V}$ in a way that $|\mathcal{P}| = |\mathcal{Q}| = |\mathcal{R}|$. Clearly, $\mathcal{S}_p$ is non-adjacent to $\{\alpha, \alpha', \beta, \beta'\}$, because $\mathcal{S}_p^{\vdash} \subseteq R$ and $p \notin N(\alpha', \beta')$. Thus walks $\mathcal{P}, \mathcal{Q}, \mathcal{R}$ satisfy statement (1.) or (2.), depending on the statement in the call of Lemma 6.3.13.

Similarly, if $p \in P \cap Y$, we observe that $\operatorname{dist}(N[p] \setminus N(\beta'), \mathbb{O}) \leqslant \operatorname{dist}(p, \overline{\mathbb{O}}) = 1$ and $p^\bullet \in N(p) \setminus N(\beta')$. Thus again we can use Lemma 6.3.13, but now for $(\alpha', \beta') \in C(\mathbb{O})$ instead of (α, β). We obtain walks $\mathcal{T} : \alpha' \to \alpha', \mathcal{U} : p \to \lambda'$ for $\lambda' \in \{\alpha', \beta'\}$ and $\mathcal{V} : \beta' \to \beta'$, such that if $\mathcal{U}$ is a p-α'-walk, then $\mathcal{T}, \mathcal{U}$ avoid $\mathcal{V}$ and otherwise $\mathcal{T}$ avoids $\mathcal{U}, \mathcal{V}$. We define

$$\begin{aligned} \mathcal{P} &= (\alpha, \alpha', \ldots, \alpha') \circ \mathcal{T} \circ (\alpha', \alpha), \\ \mathcal{Q} &= \mathcal{S}_p \circ \mathcal{U} \circ (\lambda', \lambda), \\ \mathcal{R} &= (\beta, \beta', \ldots, \beta') \circ \mathcal{V} \circ (\beta', \beta), \end{aligned}$$

where λ is the vertex in $\{\alpha, \beta\} \cap N(\lambda')$. Analogously like in the subcase when $p \in P \cap X$, we can verify that the statement of the lemma holds.

Case 2: for every $p \in P$ it holds that $p \in N(\alpha, \beta, \alpha', \beta')$. We use Lemma 6.3.10 for $s = \alpha, s' = \alpha', v = \beta, v' = \beta'$ and $S = P$ to obtain vertices $y \in P$ and $x \in Q$ and walks $\mathcal{A}, \mathcal{A}' : \alpha \to y, \mathcal{B}, \mathcal{B}' : \beta \to x$ or $\mathcal{A}, \mathcal{A}' : \beta \to y, \mathcal{B}, \mathcal{B}' : \alpha \to x$, such that $\mathcal{A}$ avoids $\mathcal{B}$, $\mathcal{B}'$ avoids $\mathcal{A}'$. Furthermore, for every $i \in [||\mathcal{A}|| + 1]$ we have $\{\mathcal{A}, \mathcal{B}\}_i \not\subseteq P$ and $\mathcal{A}, \mathcal{A}', \mathcal{B}, \mathcal{B}' \subseteq P \cup Q$.

Observe that we can assume that $\mathcal{A}, \mathcal{A}' : \alpha \to y, \mathcal{B}, \mathcal{B}' : \beta \to x$. Indeed, observe that otherwise we can consider walks $\mathcal{X} \circ \mathcal{A}, \mathcal{X}' \circ \mathcal{A}' : \alpha \to y$ and $\mathcal{Y} \circ \mathcal{B}, \mathcal{Y}' \circ \mathcal{B}' : \beta \to x$, where

$\mathcal{X}, \mathcal{X}', \mathcal{Y}, \mathcal{Y}'$ are given by Observation 5.1.9.

Now let $\mathcal{S}_y$ be the s-y-walk, such that $\mathcal{S}_y^{\vdash}$ is contained in R. Define $\mathcal{A}^* := (\alpha, \alpha', \alpha) \circ \mathcal{A}$, $\mathcal{B}^* := (\beta, \beta', \beta) \circ \mathcal{B}$, and $\mathcal{S}^* := (s, s', s) \circ \mathcal{S}_y$. Observe that $||\mathcal{A}^*|| = ||\mathcal{B}^*||$ and we have $\alpha, \alpha' \in \mathcal{A}^*$, $\beta, \beta' \in \mathcal{B}^*$, and $s, s' \in \mathcal{S}^*$. Denote the consecutive vertices of these walks by $\mathcal{A}^* = (a_1, \ldots, a_\ell)$, $\mathcal{B}^* = (b_1, \ldots, b_\ell)$, and $\mathcal{S}^* = (s_1, \ldots, s_m)$. Note that $a_1 = \alpha, s_1 = s, b_1 = \beta$ and $a_\ell = s_m = y$, $b_\ell = x$, and $\mathcal{A}^*$ avoids $\mathcal{B}^*$.

Observe that there exist $i \in [\ell]$ and $j \in [m-1]$ such that $a_i s_j \in E(H)$ or $b_i s_j \in E(H)$ (for example we have $a_\ell = s_m = y$, so $a_\ell s_{m-1} \in E(H)$). Take minimum such j and for that j take minimum i.

Define $\mathcal{C}_\alpha := (\alpha, \alpha', \ldots, \alpha)$ and $\mathcal{C}_\beta := (\beta, \beta', \ldots, \beta)$, so that $||\mathcal{C}_\alpha|| = ||\mathcal{C}_\beta|| = \max(j - i + 1, 0)$, and $\mathcal{C}_s := (s, s', \ldots, s)$, such that $||\mathcal{C}_s|| := \max(i - 1 - j, 0)$. Note that then three walks $\mathcal{C}_\alpha \circ (a_1, \ldots, a_{i-1})$, $\mathcal{C}_\beta \circ (b_1, \ldots, b_{i-1})$, and $\mathcal{C}_s \circ (s_1, \ldots, s_j)$ have the same length.

We claim that only one of the edges $a_i s_j$ and $b_i s_j$ exists. Indeed, recall that for every $j \in [m-1]$ we have $s_j \in R$. On the other hand, walks $\mathcal{A}^*$ and $\mathcal{B}^*$ are contained in $Q \cup P$. So a_i (or b_i) can only be adjacent to s_j if $a_i \in P$ ($b_i \in P$). However, since for every apropppriate j we have $\{\mathcal{A}, \mathcal{B}\}_j \not\subseteq P$, we know that at most one of a_i, b_i may be in P.

If $a_i s_j \in E(H)$ and $b_i s_j \notin E(H)$, we set

$$\begin{aligned}
\mathcal{P} &= \mathcal{C}_\alpha \circ (a_1, \ldots, a_{i-1}) \circ (a_{i-1}, a_i) \circ (a_i, \ldots, a_\ell) \circ \overline{\mathcal{A}'} \\
\mathcal{Q} &= \mathcal{C}_s \circ (s_1, \ldots, s_j) \circ (s_j, a_i) \circ (a_i, \ldots, a_\ell) \circ \overline{\mathcal{A}'} \\
\mathcal{R} &= \mathcal{C}_\beta \circ (b_1, \ldots, b_{i-1}) \circ (b_{i-1}, b_i) \circ (b_i, \ldots, b_\ell) \circ \overline{\mathcal{B}'}
\end{aligned}$$

Note that $\mathcal{P} = \mathcal{C}_\alpha \circ \mathcal{A}^* \circ \overline{\mathcal{A}'}$ and $\mathcal{R} = \mathcal{C}_\beta \circ \mathcal{B}^* \circ \overline{\mathcal{B}'}$. Observe that since $s, s' \in \mathcal{S}^*$, $\beta, \beta' \in \mathcal{B}$, and by the definition of j and i, the subwalk $\mathcal{C}_s \circ (s_1, \ldots, s_j)$ of $\mathcal{Q}$ is non-adjacent to the subwalk $\mathcal{C}_\beta \circ (b_1, \ldots, b_{i-1})$ of $\mathcal{R}$.

By analogous arguments we can see that if $b_i s_j \in E(H)$ and $a_i s_j \notin E(H)$, then for

$$\begin{aligned}
\mathcal{P} &= \mathcal{C}_\alpha \circ (a_1, \ldots, a_{i-1}) \circ (a_{i-1}, a_i) \circ (a_{i+1}, \ldots, a_\ell) \circ \overline{\mathcal{A}'} \\
\mathcal{Q} &= \mathcal{C}_s \circ (s_1, \ldots, s_j) \circ (s_j, b_i) \circ (b_{i+1}, \ldots, b_\ell) \circ \overline{\mathcal{B}'} \\
\mathcal{R} &= \mathcal{C}_\beta \circ (b_1, \ldots, b_{i-1}) \circ (b_{i-1}, b_i) \circ (b_{i+1}, \ldots, b_\ell) \circ \overline{\mathcal{B}'}
\end{aligned}$$

we have that $\mathcal{P}$ avoids $\mathcal{Q}$ and $\mathcal{R}$. □

Before we prove Lemma 6.3.8 in full generality, we consider a special case. We say that

an incomparable set S is *strongly incomparable* if for every $v \in S$ there exists $v' \in N(v)$ such that for every $u \in S \setminus \{v\}$ it holds that $uv' \notin E(H)$. Observe that then the edges $\{vv'\}_{v \in S}$ are independent.

Lemma 6.3.15. *Lemma 6.3.8 holds if we assume that S is strongly incomparable.*

Proof. Let X, Y be the bipartition classes of H such that $S = \{x_1, \ldots, x_k\} \subseteq X$, where $k \geqslant 2, x_1 = a, x_2 = b$. To simplify the notation, if it does not lead to confusion, we write $\mathcal{K}_i$ instead of $\mathcal{K}_{x_i}$. Let (α', β') be the pair, such that $C(\mathbb{O}) = \{(\alpha, \beta), (\alpha', \beta')\}$.

First, observe that all constructed walks must have even length, as they start and end in the same bipartition class. Thus the only possibility to have walks of length less than two that satisfy conditions (1)-(3) is if $|S| = 2$ and $a = \alpha$ and $b = \beta$. To avoid having walks of length 0, in this case we return walks $\mathcal{K}_a = (\alpha, \alpha', \alpha)$ and $\mathcal{K}_b = (\beta, \beta', \beta)$. So from now on we do not need to worry about the length of the walks.

We prove the lemma by induction on k. Consider the base case that $S = \{a, b\}$. By Lemma 6.1.1 used for $s = a, v = b$ and $t = \beta$, we obtain a vertex q and walks $\mathcal{P}, \mathcal{P}' : a \to \beta, \mathcal{Q}, \mathcal{Q}' : b \to q$ or $\mathcal{P}, \mathcal{P}' : a \to q, \mathcal{Q}, \mathcal{Q}' : b \to \beta$, such that $\mathcal{P}$ avoids $\mathcal{Q}$ and $\mathcal{Q}'$ avoids $\mathcal{P}'$. Note that in both cases q must be incomparable with β (recall Observation 5.1.2), so we can use Lemma 6.3.12 for $s = q$ to obtain $\mathcal{A}, \mathcal{A}' : q \to \alpha$ and $\mathcal{B}, \mathcal{B}' : \beta \to \beta$, such that $\mathcal{A}$ avoids $\mathcal{B}$ and $\mathcal{B}'$ avoids $\mathcal{A}'$.

If $\mathcal{P} : a \to q$ and $\mathcal{Q} : b \to \beta$, then clearly we can define $\mathcal{K}_a = \mathcal{P} \circ \mathcal{A}$ and $\mathcal{K}_b = \mathcal{Q} \circ \mathcal{B}$. On the other hand, if $\mathcal{P} : a \to \beta, \mathcal{Q} : b \to q$, we define $\mathcal{K}_a = \mathcal{P} \circ \mathcal{B}' \circ \mathcal{Y}'$ and $\mathcal{K}_b = \mathcal{Q} \circ \mathcal{A}' \circ \mathcal{X}'$, where $\mathcal{X}', \mathcal{Y}'$ are given by Observation 5.1.9. This proves the base case.

So now assume that $S = \{x_1, \ldots, x_k\}$ for $k \geqslant 3$ and $x_1 = a$, $x_2 = b$, and the lemma holds for $k - 1$. Let $\{\widetilde{\mathcal{K}_i}\}_{i=1}^{k-1}$ be the set of walks given by the inductive call for the set $S \setminus \{x_k\}$, where $\widetilde{\mathcal{K}_i} = (d_1^i, \ldots, d_\ell^i)$. Let $\mathbb{A}$ be the set of walks $\widetilde{\mathcal{K}_i}$ terminating at α and let $\mathbb{B}$ be the set of walks $\widetilde{\mathcal{K}_i}$ terminating at β. Clearly $\widetilde{\mathcal{K}_a} \in \mathbb{A}$ and $\widetilde{\mathcal{K}_b} \in \mathbb{B}$.

As S is strongly incomparable, for every $x_i \in S$ there exists $x_i' \in N(x_i)$, such that the edges in the set $\{x_i x_i'\}_{i \in [k]}$ are independent.

Case 1: There is an edge between $\{x_k, x_k'\}$ and $\bigcup_{i \in [k-1]} \widetilde{\mathcal{K}_i}$. This means that there is some $j \in [\ell]$ and $p \in [k-1]$ such that one of the edges $x_k d_j^p$ or $x_k' d_j^p$ exists (note that both edges cannot exist since H is bipartite). We choose the minimum such j, and for this j, if only possible, we choose p such that $\widetilde{\mathcal{K}_p} \in \mathbb{B}$.

Observe that $j > 1$, because otherwise $x_p x'_k \in E(H)$, a contradiction with the definition of x'_k. Let $\overline{x}_k \in \{x_k, x'_k\}$ be the vertex for which $\overline{x}_k d^p_j \in E(H)$. Then we define

$$\mathcal{K}_i := \begin{cases} (x_k, x'_k, \ldots, \overline{x}_k) \circ (\overline{x}_k, d^p_j, \ldots, d^p_\ell) & \text{if } i = k, \\ \widetilde{\mathcal{K}_i} & \text{if } i \in [k-1], \end{cases}$$

so that they have equal length. Denote the consecutive vertices of $\mathcal{K}_k$ by $d^k_1, \ldots, d^k_\ell$. It is clear that these walks satisfy conditions (1) and (2), so we only need to prove the condition (3). Assume that there exist walks $\mathcal{K}_q : x_q \to \alpha$ and $\mathcal{K}_r : x_r \to \beta$, such that $\mathcal{K}_q$ does not avoid $\mathcal{K}_r$. Note that by the inductive assumption this cannot happen if $q, r \in [k-1]$. Thus either $r = k$ or $q = k$.

Assume that $r = k$, i.e., $\mathcal{K}_k$ is an x_k-β-walk. Note that this means that $\widetilde{\mathcal{K}_p} \in \mathbb{B}$. Let $c \geqslant 2$ be the minimum index for which $d^q_{c-1} d^k_c \in E(H)$. If $c < j$, it means $d^q_{c-1} x_k \in E(H)$ or $d^q_{c-1} x'_k \in E(H)$, which contradicts the minimality of j. If $c \geqslant j$, then $d^k_c = d^p_c$, so d^q_{c-1} is adjacent to d^p_c. Thus the x_q-α-walk $\widetilde{K_q}$ does not avoid the x_p-β-walk $\widetilde{K_p}$, a contradiction.

So assume $q = k$, i.e., $\mathcal{K}_k$ is an x_k-α-walk and thus $\widetilde{\mathcal{K}_p} \in \mathbb{A}$. Let $c \geqslant 2$ be the smallest index for which $d^k_{c-1} d^r_c \in E(H)$. The argument is analogous: if $c < j$, then we have a contradiction with the minimality of j. If $c > j$, then the x_p-α-walk $\widetilde{\mathcal{K}_p}$ does not avoid the x_r-β walk $\widetilde{\mathcal{K}_r}$, a contradiction. Finally, if $c = j$, recall that we would choose r instead of p, as $\widetilde{\mathcal{K}_r} \in \mathbb{B}$ and $\widetilde{\mathcal{K}_p} \notin \mathbb{B}$. This completes the proof of this case.

Case 2: There are no edges between $\{x_k, x'_k\}$ and $\bigcup_{i \in [k-1]} \widetilde{\mathcal{K}_i}$. Then we use Lemma 6.3.14 for α, β and $s = x_k$ to obtain walks $\mathcal{P}, \mathcal{Q}$ and $\mathcal{R}$ and define

$$\mathcal{K}_i := \begin{cases} \widetilde{\mathcal{K}_i} & \circ \mathcal{P} \quad \text{if } \mathcal{K}_i \in \mathbb{A}, \\ \widetilde{\mathcal{K}_i} & \circ \mathcal{R} \quad \text{if } \mathcal{K}_i \in \mathbb{B}, \\ (x_k, x'_k, \ldots, x_k) & \circ \mathcal{Q} \quad \text{if } i = k, \end{cases}$$

so that all walks have the same length $\ell' - 1$. We extend the naming of vertices of walks $\mathcal{K}_i$ by denoting their consecutive vertices by $d^i_1, \ldots, d^i_{\ell'}$, note that this is consistent with previous notation, as for every $i \in [k-1]$ the walk $\widetilde{\mathcal{K}_i}$ is the prefix of $\mathcal{K}_i$.

Again, properties (1) and (2) are straightforward, let us verify the property (3). As $\mathcal{P}$ avoids $\mathcal{R}$ and for every $q \in \mathbb{A}$ and $r \in \mathbb{B}$ the walk $\mathcal{K}_q$ avoids $\mathcal{K}_r$, by Observation 5.1.3 we

know that the property (3) holds for all $q, r \in [k-1]$.

So we only need to consider two cases. First, assume that $\mathcal{Q}$ is an x_k-α-walk (and thus so is $\mathcal{K}_k$), and $\mathcal{P}, \mathcal{Q}$ avoid $\mathcal{R}$. Suppose that there exists $\mathcal{K}_r \in \mathbb{B}$ such that $\mathcal{K}_k$ does not avoid $\mathcal{K}_r$, i.e., there exists $c \geqslant 2$ such that $d^k_{c-1}d^r_c \in E(H)$. Recall that the number of vertices in $\widetilde{\mathcal{K}_r}$ is ℓ. If $c-1 \geqslant \ell$, then $\mathcal{Q}$ does not avoid $\mathcal{R}$, a contradiction. And if $c-1 < \ell$, then $d^k_{c-1} \in \{x_k, x'_k\}$, so $\widetilde{\mathcal{K}_r}$ is adjacent to x_k or x'_k, a contradiction with the assumption of the case.

Now assume that $\mathcal{Q}$ is an x_k-β-walk and there exists $q \in \mathbb{A}$ such that $\mathcal{K}_q$ does not avoid $\mathcal{K}_k$, i.e., there exists $c \geqslant 2$ such that $d^q_{c-1}d^k_c \in E(H)$. The arguments which lead to the contradiction are analogous—if $c-1 \geqslant \ell$, then $\mathcal{P}$ does not avoid $\mathcal{Q}$, and if $c-1 < \ell$, then $\widetilde{\mathcal{K}_q}$ is adjacent to x_k or x'_k. □

Now we are finally ready to prove Lemma 6.3.8.

Proof of Lemma 6.3.8. Let X, Y be the bipartition classes of H such that $S = \{x_1, \ldots, x_k\}$ is contained in X, $k \geqslant 2, x_1 = a, x_2 = b$. Again, to simplify the notation, we sometimes write $\mathcal{K}_i$ instead of $\mathcal{K}_{x_i}$. Let $a' \in N(a) \setminus N(b)$ and $b' \in N(b) \setminus N(a)$. We define $x'_1 := a'$, $x'_2 := b'$, and for every $i \geqslant 3$ such that $x_i \notin N(a')$ we choose x'_i to be any vertex from $N(x_i) \setminus N(a)$; they exist, since S is incomparable.

Let $U = \{x_i, x'_i : x_i \notin N(a')\} \cup \{a, a'\}$, and let $\mathcal{C}$ be the set of all connected components of $H \setminus N(a, a')$. Note that x_i and x'_i are always in the same component in $\mathcal{C}$. Moreover, there is a component $C_a \in \mathcal{C}$, such that $V(C_a) = \{a, a'\}$, and a component $C_b \in \mathcal{C}$, such that $b, b' \in V(C_b)$. For each $C \in \mathcal{C}$ containing at least one vertex from U, we choose one vertex $u \in U \cap X \cap V(C)$ and call it the *representative* of C. The representatives are chosen arbitrarily, except that we choose b as the representative of C_b. Note that necessarily a is the representative of C_a. Let $R \subseteq S$ be the set of all vertices that are representatives of components in $\mathcal{C}$, clearly $a, b \in R$. For every vertex of $C \in \mathcal{C}$, its representative is the representative of C.

We claim that R is strongly incomparable. Indeed, note that for every $x_i \in R$, the vertex $x'_i \in N(x_i)$ is non-adjacent to every $x_j \in R \setminus \{x_i\}$. This is because if x'_i is adjacent to some x_j, then, since x'_i is non-adjacent to a, both x_i, x_j must be in the same component in $\mathcal{C}$, so they cannot both belong to R. So calling Lemma 6.3.15 for the set R and a, b, α, β gives us the family of walks $\{\widetilde{\mathcal{K}_i}\}_{x_i \in R}$.

Recall that the only vertices $x_i \in S$ for which x'_i is not defined yet are in $(S \cap N(a')) \setminus \{a\}$. Let us consider such x_i, clearly it is adjacent to some vertices in $U \cap Y$ (at least a'). If there is some $x_j \in U$, such that:

- x_i is adjacent to x'_j, and
- $\widetilde{\mathcal{K}_r}$ terminates at β, where x_r is the representative of x_j,

then we set $x'_i := x'_j$. Otherwise, we set $x'_i := a'$ (note that $\widetilde{\mathcal{K}_a}$ terminates at α). Now for every x_i we have defined x'_i, and always $x'_i \in U \cap Y$. In particular, no x'_i, except for a', is adjacent to a.

Consider a vertex $x_i \in S$ and let $C \in \mathcal{C}$ be the component containing x'_i. Let x_r be the representative of C, and let $\mathcal{Z}_i$ be an x'_i-x_r-walk, contained in C. We define

$$\mathcal{K}_i := \underbrace{(x_i, x'_i) \circ \mathcal{Z}_i \circ (x_r, x'_r, \ldots, x_r)}_{t \text{ vertices}} \circ \widetilde{\mathcal{K}_r},$$

where t is chosen so that all $\mathcal{K}_i$'s are of equal length. Let us denote by $d^i_1, \ldots, d^i_\ell$ the consecutive vertices of $\mathcal{K}_i$, note that $(d^i_t, d^i_{t+1}, \ldots, d^i_\ell) = \widetilde{\mathcal{K}_r}$ and $d^i_t = x_r$.

It is clear that all walks $\mathcal{K}_i$ terminate at α or β, and in particular $\mathcal{K}_a$ is an a-α-walk and $\mathcal{K}_b$ is a b-β-walk, so the properties (1) and (2) hold. To prove the property (3), suppose that there are $p, q \in [k]$ such that $\mathcal{K}_p : x_p \to \alpha$ does not avoid $\mathcal{K}_q : x_q \to \beta$. So there exists $c \geqslant 2$ such that d^p_{c-1} is adjacent to d^q_c. Let $x_{p'}, x_{q'}$ be, respectively, the representatives of x_p and x_q. Clearly $x'_p, x'_q \in R$. Note that if $c - 1 \geqslant t$, then the $x_{p'}$-α-walk $\widetilde{\mathcal{K}_{p'}}$ does not avoid the $x_{q'}$-β-walk $\widetilde{\mathcal{K}_{q'}}$, a contradiction with the properties of the walks $\{\widetilde{\mathcal{K}_i}\}_{x_i \in R}$ ensured by Lemma 6.3.15.

If $2 \leqslant c-1 < t$, then there exists an $x_{p'}$-$x_{q'}$-path in $H - N(a, a')$, so they are in the same connected component in $\mathcal{C}$. Since each component in $\mathcal{C}$ has exactly one representative, we obtain that $p' = q'$ and thus and $\mathcal{K}_p$ and $\mathcal{K}_q$ both terminate in α or in β.

Finally, consider $c = 2$, which means that $d^p_1 = x_p$ is adjacent to $d^q_2 = x'_q$. There are three possibilities: (i) $x_p, x'_q \notin N(a, a')$, or (ii) $x_p \in N(a')$, or (iii) $x'_q \in N(a)$. In case (i) we observe that x_p and x'_q are in the same connected component in $\mathcal{C}$, which means that $x_{p'} = x_{q'}$, so both walks $\mathcal{K}_p, \mathcal{K}_q$ terminate at the same vertex. In case (ii), recall that when choosing x'_p, we gave preference to vertices whose representative's walk ends at β (see the second condition in definition). Thus we would have chosen $x'_p = x'_q$, a contradiction.

Finally, in case (iii), recall that $x'_q = a'$. But since the representative of a' is a, the walk $\mathcal{K}_q$ terminates at α, a contradiction. This completes the proof of the lemma. □

6.4 Lower bounds

It remains to show how to use Theorem 6.3.2 to derive Theorem 6.3.1. As already described we are going to reduce from the k-Coloring problem. We claim that it is sufficient to show the following.

Theorem 6.4.1. *Let H be a fixed connected bipartite undecomposable graph, whose complement is not a circular-arc graph. Unless the SETH fails, there is no algorithm that solves the LHom(H) problem on instances with n vertices and treewidth t in time $(i(H) - \varepsilon)^t \cdot n^{\mathcal{O}(1)}$, for any $\varepsilon > 0$.*

Let us show that Theorem 6.4.1 implies Theorem 6.3.1.

(Theorem 6.4.1 → Theorem 6.3.1). Assume the SETH and that Theorem 6.4.1 holds, but Theorem 6.3.1 fails. So there is $\varepsilon > 0$, a connected non-bi-arc graph H, and an algorithm A that solves LHom(H) in time $(i^*(H) - \varepsilon)^t \cdot n^{\mathcal{O}(1)}$ for every input (G, L) such that G is an n-vertex bipartite graph. Recall that $i^*(H) = i^*(H^*)$.

By definition of $i'(H^*)$, there exists a connected, bipartite, undecomposable graph H', whose complement is not a circular-arc graph, such that H' is an induced subgraph of H^*, and $i^*(H^*) = i(H')$. Let (G, L) be an instance of LHom(H'), such that G is a graph on n vertices. Note that we can assume that G is a bipartite graph, with bipartition classes X and Y, since all the remaining instances can be answered in time polynomial in n. We can also assume that G is connected, since otherwise we run the algorithm separately for every connected component of G. Moreover, if A and B are different bipartition classes of H', then we can assume that $L(X) \subseteq A$ and $L(Y) \subseteq B$, or $L(X) \subseteq B$ and $L(Y) \subseteq A$ (formally, we can solve two instances with the lists restricted to respective bipartite classes, and return that (G, L) is a yes-instance if and only if one of these is a yes-instance).

Clearly, (G, L) can be seen as an instance of LHom(H^*), where no vertex from $V(H^*) \setminus V(H')$ appears in any list. Note that by our assumptions, (G, L) is precisely as in the assumptions of Proposition 6.3.3 . Consider the instance $(G, \widehat{L})$ of LHom(H), constructed

as in Proposition 6.3.3. The algorithm $\mathcal{A}$ solves this instance in time $(i^*(H)-\varepsilon)^t \cdot n^{\mathcal{O}(1)}$. By Proposition 6.3.3, this is equivalent to solving the instance (G, L) of LHOM(H^*) in time $(i^*(H^*)-\varepsilon)^t \cdot n^{\mathcal{O}(1)}$, and this, in turn, is equivalent to solving the instance (G, L) of LHOM(H') in time $(i(H')-\varepsilon)^t \cdot n^{\mathcal{O}(1)}$, a contradiction. □

It remains to prove Theorem 6.4.1.

Proof of Theorem 6.4.1. Let S be a largest sound incomparable set in H. Let $k = |S|$, i.e., $k = i(H) \geqslant 3$ (by Observation 5.3.1).

Let G be an instance of k-COLORING. Clearly we can assume that G is connected and has at least 3 vertices. We will construct an instance (G', L) of LHOM(H) that has the following properties:

(a) $(G', L) \to H$ if and only if G is k-colorable,

(b) the number of vertices of G' is at most $g(H) \cdot |E(G)|$ for some function g,

(c) the treewidth of G' is at most $g(H) + \mathrm{tw}(G)$,

(d) G' can be constructed in time $|V(G)|^{\mathcal{O}(1)} \cdot g'(H)$ for some function g'.

Observe that this is sufficient to prove the theorem. Indeed, suppose that for some $\varepsilon > 0$ we can solve LHOM(H) in time $(k-\varepsilon)^t \cdot n^{\mathcal{O}(1)}$ on instances of treewidth t. Thus, for an instance G, we construct (G', L) in time $|V(G)|^{\mathcal{O}(1)} \cdot g'(H)$, and apply algorithm for LHOM(H) to (G', L). This gives an algorithm solving the k-COLORING problem on G in time

$$\begin{aligned} |V(G)|^{\mathcal{O}(1)} \cdot g'(H) + (k-\varepsilon)^t \cdot |V(G')|^{\mathcal{O}(1)} &\leqslant (k-\varepsilon)^{t+g(H)} \cdot \left(g'(H) \cdot g(H) \cdot |E(G)|\right)^{\mathcal{O}(1)} \\ &= (k-\varepsilon)^t \cdot |V(G)|^{\mathcal{O}(1)}, \end{aligned}$$

where the last step follows since $|V(H)|$ is a constant. Recall that by Theorem 1.2.1, the existence of such an algorithm for k-COLORING contradicts the SETH.

We start the construction of G' with the vertex set of G. The lists of these vertices are set to S. Let $\mathrm{NEQ}(S) = \{(u, v) \in S^2 \mid u \neq v\}$. For each edge uv of G, we introduce a copy $(F_{uv}, L, (x^{uv}, y^{uv}))$ of a list-NEQ(S)-gadget given by Theorem 6.3.2. We identify x^{uv}, y^{uv} with u and v, respectively. This completes the construction of (G', L). Let us show that it satisfies the properties (a)–(d).

Note that (a) follows directly from Theorem 6.3.2. Indeed, consider an edge uv of G. On one hand, for every list homomorphism $f : (G', L) \to H$ we have that $f(u) \neq f(v)$. On the other hand, mapping u and v to any distinct vertices from S can be extended to a homomorphism of the whole graph F_{uv}.

To show (b), recall that $|V(F_{uv})|$ depends only on H, denote it be $g(H)$. Every original vertex of G belongs to some gadget in G', so G' contains at most $g(H) \cdot |E(G)|$ vertices.

Next, consider a tree decomposition $\mathcal{T} = (T, \{X_t\}_{t \in V(\mathcal{T})})$ of G of width t. We extend $\mathcal{T}$ to a tree decomposition $\mathcal{T}^*$ of G' as follows: for every edge uv in G we choose one bag X_i such that $u, v \in X_i$, and we add a new bag $X_i' = X_i \cup V(F_{uv})$, which becomes the parent of X_i, and a child of the previous parent of X_i. We repeat this for every edge, making sure that for X_i we can only choose the original bags coming from $\mathcal{T}$. Note that it might happen that we will insert several new bags in a row, if the same X_i was chosen for different edges, but this is not a problem. Is it straightforward to observe that $\mathcal{T}^*$ is a tree decomposition of G', and the width of $\mathcal{T}^*$ is at most $g(H) + t$. This proves (c).

Finally, it is straightforward to observe that the construction of G' was performed in time polynomial in G (recall that we treat H as a constant-size graph). □

Chapter 7

Complexity of the homomorphism problems in F-free classes

The last major problem investigated in this dissertation is the study of the graph homomorphism problems in graph classes that are obtained by excluding a fixed graph as an induced subgraph. As already explained in the introduction, we aim to classify for which pairs (F, H) of fixed graphs the $\textsc{Hom}(H)$ and $\textsc{LHom}(H)$ problems can be solved in subexponential time in F-free graphs. In addition, some of the results presented in this chapter serve as a nice illustration of the possible applications of several already introduced tools.

We focus mainly on the $\textsc{LHom}(H)$ problem. Recall that we restrict our considerations to connected forbidden induced subgraphs. As discussed in the introduction, if F is a connected graph that is not a path nor a subdivided $K_{1,3}$ (called also subdivided *claws*), then for every non-bi-arc graph H the $\textsc{LHom}(H)$ problem remains NP-hard in F-free graphs and cannot be solved in subexponential time, unless the ETH fails [101, 102]. Thus, we focus on the remaining cases, i.e, graphs F that are paths and subdivided claws. Recall that by $S_{t,t,t}$ we denote the claw whose every edge is subdivided $t-1$ times.

We complete the classification (up to one extra assumption in the case of $S_{t,t,t}$-free graphs) in the following sense: we identify two families of graphs H, called *predacious* graphs and *safe* graphs, and prove the two dichotomy theorems:

Theorem 1.3.4. *Let H be a fixed graph.*

a) If H is not predacious, then for every $t \in \mathbb{N}$ the $\textsc{LHom}(H)$ problem can be solved in time $n^{\mathcal{O}(\log^2 n)}$ in n-vertex P_t-free graphs.

b) If H is predacious, then there exists $t \in \mathbb{N}$ such that LHOM(H) is NP-complete and cannot be solved in time $2^{o(n)}$ in n-vertex P_t-free graphs, unless the ETH fails.

Theorem 1.3.5. *Let H be a fixed undecomposable graph.*

a) If H is safe, then, for every $t \in \mathbb{N}$, the LHOM(H) problem can be solved in time $2^{\mathcal{O}(\sqrt{n}\log^2 n)}$ in n-vertex $S_{t,t,t}$-free graphs.

b) Otherwise, there exists $t \in \mathbb{N}$ such that LHOM(H) is NP-complete and cannot be solved in time $2^{o(n)}$ in n-vertex $S_{t,t,t}$-free graphs, unless the ETH fails.

Since the precise definitions of predacious and safe graphs are quite technical and in particular the first one is based on the notions of decomposable graphs introduced in Chapter 5, they are postponed to the next sections, where we also give some intuition behind them.

One has to have in mind that since LHOM(H) is a generalization of HOM(H), the hardness results for LHOM(H) do not imply hardness for HOM(H). In the last section of this chapter we provide some lower bounds for the HOM(H) problem in F-free graphs that can be, in particular, obtained by combining various tools and theorems presented in this and previous chapters.

Notation and fixed targets. We emphasize that in the statements of theorems presented in this chapter we assume that H is a fixed graph. In particular, H is a graph of constant size. Recall that in Chapter 6 we considered the LHOM variant of list homomorphism problem, where H is given as a part of the input, so the formal definition of an instance was slightly different. Now, some of the notions we introduced before have become simpler, however, to use them, we have to adjust them to the appropriate setting.

Recall that formally, an instance of LHOM(H) is now a pair (G, L) such that G is a graph and $L : V(G) \to 2^{V(H)}$ is a list function. Clearly, if (G, L) is an instance of LHOM(H), then (G, H, L) is an equivalent instance of LHOM.

Recall that for a graph H, the set $\text{Factors}_T(H)$ consists of the graphs corresponding to the leaves of a factorization tree T (that is defined using the notions of decompositions and factors introduced in Chapter 5). We denote by $\mathcal{H}$ the family of the graphs corresponding to the leaves of all possible factorization trees T. In other words, $\mathcal{H}$ contains all the non-bi-arc undecomposable graphs that can be obtained by decomposing H iteratively,

and also some bi-arc graphs that can be obtained this way. Note that since H is fixed, we can compute $\mathcal{H}$ in constant time. Note also that if H is undecomposable, then $\mathcal{H} = \{H\}$.

For an instance $I = (G, L)$ of LHOM(H), we say that $I' = (G', L')$ is a *subinstance* of (G, L) if G' is an induced subgraph of G and (here we use notation from Definition 5.2.3, Definition 5.2.4 and Definition 5.2.5):

- I' is an instance of LHOM(H), and for every $v \in V(G')$ we have $L'(v) \subseteq L(v)$, or
- (this case applies only if there is a Γ-decomposition of H with factorization (H_1, H_2)) I' is an instance of LHOM(H_1) or LHOM(H_2), and for every $v \in V(G')$ we have either $L'(v) \subseteq L(v)$ or
  - if $\Gamma \in \{F, BP\}$: if $L(v) \cap S \neq \emptyset$ for $S \in \{F, B, P\}$, then $L'(v) \subseteq L(v) \setminus S \cup \{s\}$ for the corresponding $s \in \{f, b, p\}$ (here $s \in V(H_2)$ is obtained by contracting the set S to a single vertex, as in the definition of H_2),
  - if $\Gamma = BB$: if $L(v) \cap (B_1 \cup B_2) \neq \emptyset$, then $L'(v) \subseteq L(v) \setminus (B_1 \cup B_2) \cup B'$, where $B' = \{b_1\}$ (resp. $B' = \{b_2\}$) if $L(v) \cap B_2 = \emptyset$ (resp. $L(v) \cap B_1 = \emptyset$) and $B' = \{b_1, b_2\}$ if $L(v) \cap B_1, L(v) \cap B_2 \neq \emptyset$.

We also extend the relation of being a subinstance so that it is closed under transitivity: if (G'', L'') is a subinstance of (G', L'), and (G', L') is a subinstance of (G, L), we call (G'', L'') a subinstance of (G, L).

Let us observe that we can define the measure $||I||$ of an instance $I = (G, L)$ of LHOM(H) as $||I|| = \sum_{v \in V(G)} (|L(v)| - 1)$, so that it coincides with the measure $||I'||$ of an instance $I' = (G, H, L)$ of LHOM, defined as in Section 5.1. However, we note that since H is fixed, we now have $||I|| = \mathcal{O}(|V(G)|)$.

Summing up, we can restate Theorem 6.1.7 for the case when H is fixed as follows.

Theorem 7.0.1. *Let H be a fixed graph. There is a family* Factors$(H) \subseteq \mathcal{H}$*, such that:*

(1) each $H' \in$ Factors(H) is either bi-arc or undecomposable,

(2) H is a bi-arc graph if and only if every $H' \in$ Factors(H) is a bi-arc graph,

(3) for each $H' \in$ Factors(H), the graph H'^* is an induced subgraph of H^*,

(4) for every instance $I = (G, L)$ of LHOM(H), the following holds:

Assume that for every undecomposable $H' \in \mathcal{H}$ there exists an algorithm $\mathcal{A}_{H'}$ that solves every subinstance (G', L') of (G, L) that is an instance of LHOM(H') in time $f'(G') \cdot |V(G')|^{\mathcal{O}(1)}$ for some function f' that is monotone w.r.t. taking induced subgraphs. Then we can solve I in time $f'(G) \cdot |V(G)|^{\mathcal{O}(1)}$.

Proof. Note that items (1)-(3) follow directly from Theorem 6.1.7. To see that (4) is also a consequence of Theorem 6.1.7, define $f(\widehat{G}, \widehat{H}) = f'(\widehat{G})$ and note that by the assumption on f', f is factor-monotone. Now, by the assumption of (4), we can solve every subinstance $I' = (G', L')$ of (G, L) that is an instance of LHOM(H') in time $f'(G') \cdot |V(G')|^{\mathcal{O}(1)} = f(G', H') \cdot ||I'||^{\mathcal{O}(1)}$. By Theorem 6.1.7 (4) (applied constantly many times), we can then solve $I = (G, L)$ in time $f(G, H) \cdot ||I||^{\mathcal{O}(1)} = f'(G) \cdot |V(G)|^{\mathcal{O}(1)}$. □

The remaining part of the chapter is divided into two sections. Section 7.1 contains the proof of Theorem 1.3.4, and Section 7.2 contains the proof of Theorem 1.3.5. Each of these sections contains two subsections, corresponding to parts a) and b) of respective theorems.

7.1 P_t-free graphs

In this section we focus on Theorem 1.3.4. Before we present the definition of predacious graphs, that are crucial for the statement of the theorem, let us discuss some intuition behind it. Informally speaking, we need to understand what property of target graphs H makes it possible to solve the LHOM(H) problem in P_t-free graphs for a fixed t, i.e., distinguish "easy" non-predacious graphs H from the "hard" predacious ones, so the dichotomy in Theorem 1.3.4 holds.

As already discussed in the introduction, in the classes of graphs defined by forbidding a fixed path P_t the 3-COLORING problem behaves differently than the k-COLORING problem for $k \geqslant 4$. Indeed, for every $k \geqslant 4$ the problem is "hard", in a sense that there exists $t \in \mathbb{N}$ such that k-COLORING in NP-complete and cannot be solved in subexponential time in P_t-free graphs, unless the ETH fails. This is not the case for 3-COLORING, as witnessed by the mentioned quasipolynomial-time algorithm by Pilipczuk et al. [103]. Therefore, if we are interested in the boundary between the "easy" and "hard" target graphs H, the clique K_3 falls under the first category, while all the bigger cliques fall into the other.

Recall that Groenland et al. [56] proved that if H is a graph in which any two distinct vertices share at most one common neighbor, then LHOM(H) (or actually a more general, *weighted* version of the problem) can be solved in subexponential time in P_t-free graphs. See Figure 7.1. While such graphs H turned out to be only a subclass of non-predacious target graphs for our problem, a crucial observation was made there: if no two vertices of H have two common neighbors, then assigning any vertex from $L(v)$ to a vertex v of the instance allows us to propagate the choice to the neighbors of v. It was then used to design a branching algorithm for the weighted homomorphism problem.

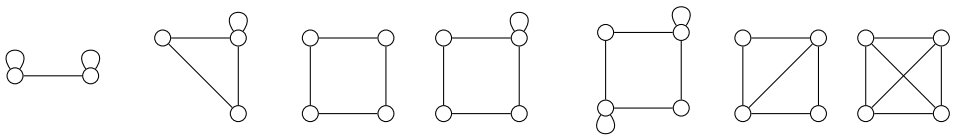


Figure 7.1: No two vertices of H have two common neighbors if and only if H does not contain any of the graphs above as an induced subgraph.

Since in the LHOM(H) problem we can in particular assume that the lists are incomparable sets, to use a similar method, we introduce a more specific structure, called a *predator* (see Figure 7.2).

Definition 7.1.1. A 4-tuple (a_1, a_2, b_1, b_2) of vertices of H is called a *predator* if $\{a_1, a_2\}$ and $\{b_1, b_2\}$ are incomparable sets, complete to each other.

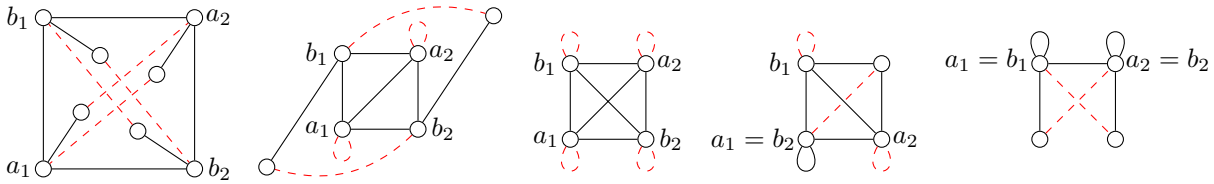


Figure 7.2: Examples of predators (a_1, a_2, b_1, b_2) and their neighbors. Red dashed lines denote the edges that cannot exist. The edges that are not drawn are possible, but not necessary.

It turns out that if H is an undecomposable graph, then the existence of a predator in H is the main obstacle for an efficient algorithm for LHOM(H) in P_t-free graphs. An undecomposable graph H is *predacious* if there a connected non-bi-arc component of H that contains a predator.

The only thing that is missing now is how the decompositions of the target graph come into play. We show that in order to solve LHOM(H), it is sufficient to solve LHOM(H')

for every $H' \in \mathcal{H}$ (this is a consequence of Theorem 7.0.1). Thus, if every graph in $\mathcal{H}$ is non-predacious, we are able to solve LHOM(H) in P_t-free graphs efficiently, thus H is an "easy" target graph.

On the other hand, we find a way to reduce LHOM(H') to LHOM(H) for every $H' \in \mathcal{H}$. Therefore, if any $H' \in \mathcal{H}$ is predacious, then so should be H. We are ready to define the class of predacious graphs.

Definition 7.1.2. A graph H is *predacious* if there exists a graph $H' \in \mathcal{H}$ and a non-bi-arc connected component of H' that contains a predator.

In other words, H is predacious if and only if there exists $H' \in \mathcal{H}$ that is predacious.

The proof of Theorem 1.3.4 (a) builds on the decomposition of target graphs H defined in Chapter 5, and on the quasi-polynomial-time algorithm for 3-COLORING P_t-free graphs by Pilipczuk et al. [103], which is in turn inspired by the algorithm for MAX INDEPENDENT SET by Gartland and Lokshtanov [51]. The main part of our algorithm is a simple branching procedure: for an instance (G, L) of LHOM(H), we choose a vertex $v \in V(G)$ and $a \in L(v)$ and guess whether we map v to a or not. The crucial challenge here is choosing the pair (v, a) appropriately, so that the at least one of the instances obtained in the branches is significantly smaller (with respect to a certain measures) than the original instance. To prove the existence of a such pair we use the fact that H does not contain a predator—note, however, that this is not exactly the definition of a non-predacious graph. Here Theorem 7.0.1 comes into play: we note that it is enough to have an algorithm that solves LHOM(H) when H is undecomposable.

The hardness counterpart of Theorem 1.3.4, consists of three parts. First, we separately consider strong split target graphs H. We show that if H is a strong split non-bi-arc graph, then there always exists $t \in \mathbb{N}$ such that LHOM(H) is hard in P_t-free graphs. Indeed, although we will not use this fact, it can be shown that every non-bi-arc strong split graph is predacious. Second, we proceed to undecomposable bipartite non-bi-arc predacious target graphs H. We exploit the structure of a predator in H, and this, combined with Theorem 6.3.2, turns out to be precisely what is needed to perform an elegant reduction from the 3-SAT problem. Last, we conclude the hardness proof by showing that the two mentioned reductions are enough to construct a lower bound for the general case, using the idea of associated bipartite graphs introduced in Proposition 5.0.2.

7.1.1 The algorithm

We claim that to prove Theorem 1.3.4 a), it is sufficient to show the following.

Theorem 7.1.3. *Let H be a fixed connected, non-bi-arc graph that does not contain a predator. Then for every $t \in \mathbb{N}$, the LHOM(H) problem can be solved in time $n^{\mathcal{O}(\log^2 n)}$ in n-vertex P_t-free graphs.*

Indeed, consider a non-predacious graph H. By definition, no graph $H' \in \mathcal{H}$ has a connected component that is non-bi-arc and contains a predator. Thus we can always solve LHOM(H') in time $|V(G)|^{\mathcal{O}(\log^2 |V(G)|)}$ by Theorem 7.1.3 and the fact that for bi-arc target graphs the list homomorphism problem can be solved in polynomial time. Since for every constant $c \geqslant 1$ the function $|V(G)|^{c \log^2 |V(G)|)}$ is monotone w.r.t. taking induced subgraphs, Theorem 1.3.4 follows from Theorem 7.0.1 (4).

Thus, from now on, we focus on proving Theorem 7.1.3. We need the following simple observation.

Observation 7.1.4. *Let H be a graph which does not contain a predator. For any incomparable sets $X, Y \subseteq V(H)$, each of size at least 2, there exist $a \in X$ and $b \in Y$ such that $ab \notin E(H)$.*

Proof. For contradiction, suppose that there are two incomparable sets X, Y, each of size at least 2, which are complete to each other. Let a_1, a_2 be distinct elements from X, and b_1, b_2 be distinct elements from Y. Then (a_1, a_2, b_1, b_2) is a predator. □

We proceed to the proof of Theorem 7.1.3.

Proof of Theorem 7.1.3. Let (G, L) be an instance of LHOM(H), such that G is P_t-free. We start with a preprocessing phase, in which we exhaustively perform the following steps, in given order.

1. If for some $v \in V(G)$ it holds that $L(v) = \emptyset$, then we terminate and report a no-instance.

2. We enumerate all $S \in \binom{V(G)}{\leqslant t}$, and all possible homomorphisms $(G[S], L) \to H$. If for some $v \in V(G)$, some $a \in L(v)$, and some $S \in \binom{V(G)}{\leqslant t}$ such that $v \in S$ there is no $h : (G[S], L) \to H$ such that $h(v) = a$, we remove a from $L(v)$.

3. If for some $v \in V(G)$ we have $|L(v)| = 1$, we remove v from G. Note that by the previous step the lists of neighbors of v contain only neighbors of the vertex in $L(v)$.

The correctness of the above steps is straightforward. Furthermore, as t and $|V(H)|$ are constant, we can perform the whole preprocessing phase in time polynomial in $|V(G)|$.

Assume that none of the steps 1.-3. can be further applied. We continue calling the current instance (G, L), let n be its number of vertices. The instance satisfies the following properties.

(P1) For every $v \in V(G)$, the set $L(v)$ is incomparable and has at least two elements.

(P2) For every $v \in V(G)$, every $S \in \binom{V(G)}{\leqslant t}$, such that $v \in S$, and every $a \in L(v)$, there exists $h : (G[S], L) \to H$ which maps v to a.

Now let us describe the algorithm. If $n \leqslant 1$, then we report a yes-instance; recall that by property (P1) each list is non-empty. If the instance G is disconnected, we call the algorithm for each connected component independently. We report that (G, L) is a yes-instance if and only if all these calls report yes-instances.

If none of the above cases occurs, we perform branching. We will carefully choose a *branching pair* (v, a), where $v \in V(G)$ and $a \in L(v)$, and branch into two possibilities. In the first one, called the *successful* branch, we call the algorithm recursively with the list $L(v)$ set to $\{a\}$. This corresponds to coloring v with a. Note that in the preprocessing phase of this call we will remove all non-neighbors of a from the lists of neighbors of v, and then remove v from the graph. In the second branch, called the *failure branch*, we call the algorithm with a removed from $L(v)$. This corresponds to choosing not to color v with a. Then, we report a yes-instance if and only if at least one of the branches reports a yes-instance.

Now let us discuss how we select a branching pair. For each $\{u, u'\} \in \binom{V(G)}{2}$ we define the *buckets* $\mathcal{B}'_{u,u'}$ and $\mathcal{B}_{u,u'}$. The elements of $\mathcal{B}'_{u,u'}$ are the induced u-u'-paths in G. Let $\theta(u, u') := |\mathcal{B}'_{u,u'}|$ The elements of $\mathcal{B}_{u,u'}$ are all possible pairs (P, h), where $P \in \mathcal{B}'_{u,u'}$ and h is a list homomorphism from (P, L) to H. We refer to pairs (P, h) as *colored paths*.

Claim 7.4.1 ([103]). *A connected P_t-free graph has a vertex v such that there exists at least $\binom{n}{2}/2t$ sets $\{u, u'\} \in \binom{V(G)}{2}$ such that $N[v]$ intersects at least $\theta(u, u')/2t$ paths in $B'_{u,u'}$.*

Note that since G is P_t-free, the total size of every bucket $B_{u,u'}$ is $\mathcal{O}(n^t)$, and they can be enumerated in polynomial time. Furthermore, by property (P2), we know that $\mathcal{B}_{u,u'}$ is non-empty if and only if u and u' are in the same connected component of G. Even more, if w belongs to an induced u-u'-path P, and $a \in L(w)$, then $\mathcal{B}_{u,u'}$ contains a colored path (P, h), such that $h(w) = a$.

Define

$$\delta := \frac{1}{2^{|V(H)|+1} \cdot t} \qquad \text{and} \qquad \varepsilon := \frac{1}{2^{|V(H)|+1} \cdot |V(H)|^{t-1} \cdot t} = \frac{\delta}{|V(H)|^{t-1}}.$$

Intuitively, the following claim shows that we can always choose a branching pair that allows us to significantly decrease the size of at least one instance in the branches.

Claim 7.4.2. *If G is a connected P_t-free graph, then there is a pair (v, a), where $v \in V(G)$ and $a \in L(v)$, with the following property. There is a set $Q \subseteq \binom{V(G)}{2}$ of size at least $\delta \cdot \binom{n}{2}$, such that for every $\{u, u'\} \in Q$ there is a subset $\mathcal{P}_{u,u'} \subseteq \mathcal{B}_{u,u'}$ of size at least $\varepsilon \cdot |\mathcal{B}_{u,u'}|$, such that for every $(P, h) \in \mathcal{P}_{u,u'}$, there is $w_P \in V(P) \cap N[v]$, such that $h(w_P) \notin N_H(a)$.*

Proof of Claim. By Claim 7.4.1, there is a vertex $v \in V(G)$, such that for at least $\frac{1}{2t}\binom{n}{2}$ pairs $\{u, u'\} \in \binom{V(G)}{2}$ and for at least $\frac{1}{2t}\ \theta(u, u')$ induced u-u'-paths P, the set $N[v]$ intersects $V(P)$.

Recall that the number of distinct elements of $\{L(u) \mid u \in V(G)\}$ is at most $2^{|V(H)|}$. Thus, by the pigeonhole principle there is a list $L' \subseteq V(H)$ and a subset $Q \subseteq \binom{V(H)}{2}$ of size at least $\frac{1}{2^{|V(H)|+1} \cdot t}\binom{n}{2} = \delta \cdot \binom{n}{2}$, such that the following property is satisfied.

($\star$) For every $\{u, u'\} \in Q$ there exists a set $\mathcal{P}_{u,u'}$ of at least $\delta \cdot \theta(u, u')$ induced u-u'-paths, such that for every $P \in \mathcal{P}_{u,u'}$ there exists $w_P \in N[v] \cap V(P)$, such that $L(w_P) = L'$.

By property (P1) we know that each of $L(v)$ and L' is an incomparable set with at least two elements. Thus by Observation 7.1.4 we know that there are $a \in L(v)$ and $b \in L'$, which are non-adjacent in H.

Let us argue that the pair (v, a) satisfies the desired conditions. Fix some $\{u, u'\} \in Q$. As every induced u-u' path has at most $t - 1$ elements, we have that $|\mathcal{B}_{u,u'}| \leqslant |V(H)|^{t-1} \cdot \theta(u, u')$. On the other hand, by property (P2), for every $P \in \mathcal{P}_{u,u'}$ there exists a homomorphism $h : (P, L) \to H$ such that $h(w_P) = b \notin N_H(a)$. So, summing up, we

obtain that the number of such pairs $(P,h) \in \mathcal{B}_{u,u'}$ is at least

$$|\mathcal{P}_{u,u'}| \geqslant \delta \cdot \theta(u,u') \geqslant \frac{\delta}{|V(H)|^{t-1}} \cdot |\mathcal{B}_{u,u'}| = \varepsilon \cdot |\mathcal{B}_{u,u'}|.$$

⌟

Consider the successful branch for the branching pair (v,a) given by Claim 7.4.2. We use the notation from the statement of the claim. For some $\{u,u'\} \in Q$, let (P,h) be a colored path in $\mathcal{P}_{u,u'}$, and let w_P be as in the claim. Consider the preprocessing phase of the current call. If $w_P = v$, then w_P is removed from the graph, so (P,h) will no longer appear in the bucket of $\{u,u'\}$. Similarly, if $w_P \neq v$, then we remove $h(w_P)$ from $L(w_P)$, so (P,h) will not appear in the bucket of $\{u,u'\}$. Thus, informally speaking, when we branch using the pair (v,a), in the successful branch we remove an ε-fraction of elements in a δ-fraction of buckets.

Note that in each recursive call the total size of lists is reduced, so the algorithm terminates. It is also clear that it always returns a correct answer. So let us argue that the complexity is indeed bounded by $n^{\mathcal{O}(\log^2 n)}$. The analysis is essentially the same as the one of the algorithm by Pilipczuk et al. [103]. We present it for the sake of completeness.

Let $\mathcal{T}$ be the recursion tree of the algorithm called for the instance (G,L). The nodes of $\mathcal{T}$ correspond to calls at instances (G',L'), where G' is an induced subgraph of G, and for each $v \in V(G')$ it holds that $L'(v) \subseteq L(v)$. For a node of $\mathcal{T}$, corresponding to a call at instance (G',L'), its *local subtree* consists of all descendant calls where the instance graph has at least $0.99|V(G')|$ vertices. We find a partition Π of nodes of $\mathcal{T}$ into local subtrees in a greedy way. We start with $\Pi = \emptyset$, and while there are still some nodes that are not covered by Π, we include in Π the local subtree of such a node, which is closest to the root.

Clearly each path from the root to a leaf of $\mathcal{T}$ intersects $\mathcal{O}(\log n)$ elements of Π. Consider a local subtree $\mathcal{T}'$, whose root corresponds to the call at an instance (G',L') with n' vertices. We claim that $\mathcal{T}'$ has $(n')^{\mathcal{O}(\log n')} = n^{\mathcal{O}(\log n)}$ leaves. To see that, we now mark some edges of $\mathcal{T}'$. Consider a call in $\mathcal{T}'$ at an instance (G'',L'').

1. If G'' is disconnected, there is at most one child call that belongs to $\mathcal{T}'$: its instance is the component of G'' with at least $0.99|V(G'')|$ vertices. If such a child call exists, we mark the edge to it.

2. If G'' is connected, then we mark the edge to the call in the failure branch (if it belongs to $\mathcal{T}'$).

Every node in $\mathcal{T}'$ has at most one marked edge to a child. Let $\mathcal{T}''$ be obtained from $\mathcal{T}'$ contracting all marked edges. Every node of $\mathcal{T}''$ has $\mathcal{O}(n')$ children and each edge of $\mathcal{T}''$ corresponds to a successful branch in some call in $\mathcal{T}'$. Now it is sufficient to argue that the depth of $\mathcal{T}''$ is $\mathcal{O}(\log n') = \mathcal{O}(\log n)$. Indeed, this implies that the number of leaves in $\mathcal{T}''$ is $n^{\mathcal{O}(\log n)}$, and the number of leaves in $\mathcal{T}'$ is at most twice the number of leaves of $\mathcal{T}''$, thus also $n^{\mathcal{O}(\log n)}$.

For a call at instance (G'', L'') in $\mathcal{T}'$ we define the potential:

$$\mu(G'', L'') := - \sum_{\{u,u'\} \in \binom{V(G'')}{2}} \log_{1-\varepsilon}(1 + |\mathcal{B}_{u,u'}|).$$

At the root call of $\mathcal{T}'$ we have $\mu(G', L') = \mathcal{O}((n')^2 \log n')$. Indeed, the size of each bucket is at most $(n' \cdot |V(H)|)^{t-1} \leqslant (n' \cdot |V(H)|)^t$, and thus

$$\begin{aligned}\mu(G', L') &\leqslant -(n')^2 \cdot \log_{1-\varepsilon}(1 + (|V(H)| \cdot n')^t) = c \cdot (n')^2 \log(1 + (|V(H)| \cdot n')^t) \\ &\leqslant 2c \cdot (n')^2 \log(|V(H)| \cdot n')^t = 2ct \cdot (n')^2 \log(|V(H)| \cdot n') = \mathcal{O}((n')^2 \log n'),\end{aligned}$$

where $c = -\log(1-\varepsilon)$ (here, log is the natural logarithm function). By Claim 7.4.2, every successful branch at a call on an instance (G'', L'') decreases the potential μ by at least

$$\delta \cdot \binom{|V(G'')|}{2} \geqslant \delta \cdot \binom{\lceil 0.99n' \rceil}{2} \geqslant 0.9\delta \cdot \binom{n'}{2}.$$

Since μ is non-negative, it follows that the depth of $\mathcal{T}''$ is bounded by $\mathcal{O}(\log n') = \mathcal{O}(\log n)$. We conclude that $\mathcal{T}'$ has $n^{\mathcal{O}(\log n)}$ leaves.

Now we observe that $\mathcal{T}'$ has also $n^{\mathcal{O}(\log n)}$ vertices, as every leaf-to-root path in $\mathcal{T}'$ has at most n vertices, and there is at most $n^{\mathcal{O}(\log n)}$ such paths. Consider a tree $\mathcal{T}'''$ obtained from $\mathcal{T}$ by contracting each local subtree $\mathcal{T}'$ to one vertex. As each vertex of $\mathcal{T}$ has at most $\mathcal{O}(n)$ children, each internal node of $\mathcal{T}'''$ has at most $n^{\mathcal{O}(\log n)}$ children. As each leaf-to-root path in $\mathcal{T}'''$ is of length at most $\mathcal{O}(\log n)$, the total number of nodes in $\mathcal{T}'''$ is at most $n^{\mathcal{O}(\log^2 n)}$. Since each local subtree has $n^{\mathcal{O}(\log n)}$ vertices, we obtain that the total number of nodes of $\mathcal{T}$ is $n^{\mathcal{O}(\log^2 n)}$. □

7.1.2 Lower bounds

We will prove the hardness part in a few steps. First, we focus on target graphs that are strong split graphs, and then on target graphs that are bipartite and undecomposable. Having this, we will be able to work with the general case, i.e., to obtain the hardness counterpart of Theorem 1.3.4.

Strong split target graphs Recall that graph G is a strong split graph, if its set of vertices can be partitioned into sets B and P such that B is an independent set and P is a reflexive clique. Similarly, G is a *split graph* if its vertex set can be partitioned into B and P such that B is an independent set and P is an irreflexive clique. Equally, split graphs can be defined as graphs that are $\{C_4, C_5, P_5\}$-free.

We start with the following structural observation.

Observation 7.1.5. *Let H be a graph, and let $H' \in \mathcal{H}$ be a connected, non-bi-arc strong split graph. Then H' is an induced subgraph of H.*

Proof. Recall that by definitions of the factors, if H' is not (isomorphic to) an induced subgraph of H, it is because of vertices that have loops, and all their neighbors also have loops (and they appear after we apply the F-decomposition). Thus such vertices belong to the reflexive clique in H' and have no neighbors outside this clique. It is straightforward to verify that if f is such a vertex, then $(B, P \setminus \{f\}, \{f\}, \emptyset, Z)$ is a BP-decomposition of H'. Indeed, as H is non-bi-arc, $|B| \geqslant 2$. However, since $H' \in \mathcal{H}$, H' must be undecomposable, a contradiction. □

The first part of the hardness proof focuses on strong split target graphs. We reduce from the general LHOM(H) problem and show that it remains hard even if we assume that G is a split graph (so, in particular, is P_5-free).

Theorem 7.1.6. *Let H be a fixed connected non-bi-arc strong split graph. Then the LHOM(H) problem is NP-hard and cannot be solved in time $2^{o(n)}$ in n-vertex split graphs, unless the ETH fails.*

Proof. Let P be the set of vertices in H that have loops, and let B be the set of vertices of H without loops. Consider an instance (G, L) of LHOM(H). As usual, we can assume that each list $L(v)$ is an incomparable set. As for every $p \in P$ and $b \in B$ it holds that

$N(b) \subseteq N(p)$, no vertex in G has both a vertex from P and a vertex from B in its list. Since we can assume that every list is non-empty, we can partition the vertex set of $V(G)$ into two sets:

$$X := \{v \in V(G) \mid L(v) \cap P \neq \emptyset\} \quad \text{and} \quad Y := \{v \in V(G) \mid L(v) \cap B \neq \emptyset\}.$$

Furthermore, as B is independent, without loss of generality we can assume that Y is independent, as otherwise (G, L) is a no-instance. Let G' be obtained from G by turning X into an irreflexive clique, i.e., we add all edges with both endvertices in X (except for loops). It is straightforward to verify that $(G, L) \to H'$ if and only if $(G', L) \to H'$. As $V(G')$ is partitioned into a clique X and an independent set Y, the theorem follows. □

General target graphs As a second part of our hardness proof, we focus on bipartite undecomposable target graphs. Then, we wrap up the proof and show Theorem 1.3.4 b).

Theorem 7.1.7. *Let H be a fixed, bipartite, undecomposable, predacious graph. Then there exists $t \in \mathbb{N}$, such that the LHOM(H) problem in NP-hard and cannot be solved in time $2^{o(n)}$ in n-vertex P_t-free graphs, unless the ETH fails.*

Proof. If H is undecomposable, so is its every connected component. Let H' be a non-bi-arc connected component of H that contains a predator (a_1, a_2, b_1, b_2). As H' is bipartite, we note that $\{a_1, a_2\}$ (resp. $\{b_1, b_2\}$) is contained in one bipartition class, thus, is a sound incomparable set. We reduce from 3-SAT. Consider a formula Φ of 3-SAT with variables $x_1, \ldots, x_N$ and clauses $C_1, \ldots, C_M$. Without loss of generality we can assume that each clause has exactly three literals (we can ensure this by duplicating some literal in a shorter clause). We construct an instance (G_Φ, L) of LHOM(H') (and thus of LHOM(H)) as follows. First, we introduce a biclique with partite sets $V := \{v_1, \ldots, v_N\}$ and $U := \{u_1, \ldots, u_{3M}\}$. Vertices in V are in one-to-one correspondence to the variables of Φ, while vertices in U are in one-to-one correspondence to literals in Φ, i.e., the occurrences of the variables in clauses. For a clause C_i, by U_i we denote the three-element subset of vertices of U corresponding to the literals of C_i. For every $j \in [N]$ we set $L(v_j) = \{a_1, a_2\}$ and for every $i \in [3M]$ we set $L(u_i) = \{b_1, b_2\}$.

The intuition behind the construction is that mapping the vertex v_j to a_1 (a_2, resp.) corresponds to making the variable v_j true (false, resp.). Similarly, we will interpret u_j

being mapped to b_1 (b_2, resp.) as setting the corresponding literal true (false, resp.). So we need to ensure that (i) the coloring of vertices in V is correspond to the coloring of vertices in U, according to the above interpretation, and (ii) for each clause C_i, at least one vertex in U_i is mapped to b_1.

To ensure property (i), we will introduce two types of *occurrence gadgets*. We use Theorem 6.3.2 to construct a list-$\{(a_1, b_1), (a_2, b_2)\}$-gadget, called *positive occurrence gadget*, and list-$\{(a_1, b_2), (a_2, b_1)\}$-gadget, called *negative occurrence gadget*. Recall from the definition that a positive occurrence gadget $(F, L, (x_1, x_2))$ has two interface vertices x_1 and x_2 such that

$$\{(h(x_1), h(x_2)) \mid h : (F, L) \to H\} = \{(a_1, b_1), (a_2, b_2)\}.$$

Similarly, a negative occurrence gadget $(F, L, (x_1, x_2))$ has two interface vertices x_1 and x_2 such that

$$\{(h(x_1), h(x_2)) \mid h : (F, L) \to H\} = \{(a_1, b_2), (a_2, b_1)\}.$$

Note that from the properties of every gadget (F, L, X) we introduced it follows that there exists a homomorphism $F \to H'$, thus F is in particular bipartite, with interface wertices in different bipartition classes.

Now consider a vertex $u_i \in U$, which corresponds to an occurrence of a variable x_j, and thus to the vertex v_j. If u_i corresponds to a positive (resp., negative) literal, we introduce a positive (resp., negative) occurrence gadget, and identify v_j and u_i, respectively with its first and second interface vertex.

In the following straightforward claim we summarize that the constructed gadgets can indeed be used to ensure property (i).

Claim 7.1.7.1. *Let $(F, L, (v_j, u_i))$ be a positive (resp., negative) occurrence gadget. There exist homomorphisms $h_1, h_2 : (G, L) \to H$, such that $h_1(v_j) = a_1$ and $h_2(v_j) = a_2$. Moreover, for any homomorphism $(G, L) \to H$, it holds that $h(v_j) = a_1$ if and only if $h(u_i) = b_1$ (resp., $h(u_i) = b_2$).*

To ensure (ii), consider a set $U_i = \{u^1, u^2, u^3\}$, corresponding to the literals of some clause C_i. We observe that in order to satisfy property (ii), we need to construct a list-$\mathrm{OR}_3(b_1, b_2)$-gadget, whose interface vertices are precisely u^1, u^2, and u^3. We introduce a list-$\mathrm{OR}_3(b_1, b_2)$-gadget given by Theorem 6.3.2, and identify its interface vertices with

u^1, u^2, and u^3. Clearly, the gadget again must be bipartite, and its interface vertices belong to the same bipartition class. This completes the construction of (G_Φ, L). The following claim follows directly from the discussion above.

Claim 7.1.7.2. *There exists a list homomorphism $h : (G_\Phi, L) \to H$ if and only if there exists a truth assignment of variables of Φ such that all clauses are satisfied.*

It remains to show that the length of a longest induced path in G_Φ is bounded by a constant (that may depend on $|V(H')|$). Let t' be the maximum of the numbers of vertices in a positive occurrence gadget, a negative occurrence gadget, and an $\mathrm{OR}_3(b_1, b_2)$-gadget, and let $t := 4t' + 4$.

Claim 7.1.7.3. *The graph G_Φ is P_t-free.*

Proof of Claim. For contradiction, suppose that P is an induced path in G_Φ with at least t vertices. A *segment* of P is an inclusion-wise maximal subpath consisting of vertices of one gadget, excluding the vertices of $V \cup U$. Note that each two consecutive segments on P are separated by a vertex from $V \cup U$. As each segment has at most t' vertices, we obtain that P contains (at least) four vertices from $V \cup U$. Clearly, if P contains two vertices from U and two vertices form V, then P cannot be an induced path. Similarly, P cannot contain three vertices from U (resp. V) and a vertex from V (resp. U). Thus we either have $V(P) \cap (U \cup V) \subseteq V$ or $V(P) \cap (U \cup V) \subseteq U$. Note, however, that the first case is impossible, since each vertex of V is in a separate connected component of $G_\Phi - U$. Similarly, there are at most three vertices of U in a single list-$\mathrm{OR}_3(b_1, b_2)$-gadget, and thus each connected component of $G_\Phi - V$ contains at most three vertices of U. This is a contradiction with the fact that P contains at least four vertices of $U \cup V$. ⌟

We note that we do not make any attempt in the above claim to optimize t. As the number of vertices in G_Φ is $\mathcal{O}(N + M)$, the theorem holds. □

It remains to show how to derive Theorem 1.3.4 b).

Proof of Theorem 1.3.4 b). Since H is predacious, there exists $H' \in \mathcal{H}$ that has a non-bi-arc connected component H'' with a predator.

If H'' is a strong split graph, then by Theorem 7.1.6, the $\textsc{LHom}(H'')$ problem cannot be solved in subexponential time in split graphs, that are in particular P_5-free. Since, by

Observation 7.1.5, H'' is an induced subgraph of H, any instance of LHOM(H'') can be seen as an instance of the LHOM(H) problem. Thus, the statement follows.

If H'' is not a strong split graph, suppose, for contradiction, that for every t there is an algorithm $\mathcal{A}_t$, which solves every P_t-free instance of LHOM(H) in subexponential time. By Lemma 5.2.7, H''^* is undecomposable. Thus, H''^* satisfies the assumptions of Theorem 7.1.7. Let t be given by Theorem 7.1.7 for H''^*.

Consider an arbitrary instance (G, L) of LHOM(H''^*), where G is P_t-free. As H''^* is an induced subgraph of H^*, the instance (G, L) can be seen as an instance of LHOM(H^*). Create an instance $(G, \widehat{L})$ of LHOM(H), where $\widehat{L}$ is as in Proposition 6.3.3. Constructing $\widehat{L}$ clearly takes polynomial time, and $(G, L) \to H^*$ if and only if $(G, \widehat{L}) \to H$ by Proposition 6.3.3. We can use $\mathcal{A}_t$ to decide whether $(G, \widehat{L}) \to H$ or, equivalently, whether $(G, L) \to H''^*$, in subexponential time. This contradicts Theorem 7.1.7. □

7.2 $S_{t,t,t}$-free graphs

In this section we focus on Theorem 1.3.5. Before we formally define safe graphs, let us first provide an intuition behind their structure, and the overall description of the result.

The class of $S_{t,t,t}$-free graphs can be seen as a generalization of claw-free graphs (that we equivalently call $S_{1,1,1}$-free or $K_{1,3}$-free graphs). These are, in turn, a generalization of another important class of *line graphs*. Recall that for $k \geqslant 3$ the k-COLORING problem is NP-complete and cannot be solved in subexponential time even in line graphs, unless the ETH fails [70, 86]. This of course implies that if H is a clique on at least three vertices, the HOM(H) problem is NP-complete and cannot be solved in subexponential time even in line graphs, unless the ETH fails.

Recall that in the case of P_t-free graphs it was observed that similar techniques can be used to solve (LIST-)3-COLORING and MAX INDEPENDENT SET (MIS) in quasi-polynomial time. We exploited this similarity in our work, generalizing this approach allowed us to solve LHOM(H) for all non-predacious graphs H. Here, we again aim for the algorithm that solves the list version of the homomorphism problem, however, we already noted that the existence of an algorithm that solves (LIST-)3-COLORING in subexponential time in $S_{t,t,t}$-free graphs is unlikely. Still, algorithms that solve the MIS problem in $S_{t,t,t}$-free graphs in subexponential time exist [17, 18, 90]—and surprisingly,

it turns out that there is again a similarity of behavior between the MIS problem and the LHOM(H) problem for certain target graphs H. Although we still need to develop a framework that allows us to work with homomorphisms instead of independent sets, it is possible to apply techniques used to solve MIS in subexponential time in our algorithm for the LHOM(H) problem in $S_{t,t,t}$-free graphs.

It is worth to mention that while the current techniques developed in the area seem to be insufficient to design a polynomial-time algorithm for the MIS problem in $S_{t,t,t}$-free graph classes, the problem itself brings a lot of attention of researchers—including a recent breakthrough result claimed by Gartland et al. [52] who provided a quasi-polynomial time algorithm for the problem. It is possible that their approach could also be used to improve our algorithm for the LHOM(H) problem, as in the case of P_t-free graphs; we are, however, basing our work on the algorithm of Majewski et al. [90] that has the subexponential time complexity.

There are two main phases of the algorithm of Majewski et al. [90]. One of them is branching, similar to the one in the proof of Theorem 7.1.3, but conceptually simpler; since we do not know how to pick a good branching pair, we choose a vertex of the instance graph that has large degree. Then, if there is no such vertex, we use the notion of an *extended strip decomposition* introduced by Chudnovsky and Seymour [20] (that we define formally in Section 7.2.2). Informally, an instance graph G that can be nicely decomposed this way is "similar" to a line graph. In particular, in such a decomposition one can distinguish *particles* being induced subgraphs of G whose interaction with each other is somehow restricted; an algorithm for MIS can recurse on individual particles, compute the maximum independent sets there, and combine the results into a maximum independent set in G using a maximum matching algorithm on an auxiliary graph.

We need to understand which property of a target graph H allow us to apply a similar strategy to solve the LHOM(H) problem. We call *safe* the graphs H for which there exists a subexponential time algorithm solving LHOM(H) in $S_{t,t,t}$-free graphs for every t. We focus on the case when H is an undecomposable graph.

The branching phase again requires only the absence of a predator in H. Clearly, a safe graph cannot be predacious—for each fixed $t \in \mathbb{N}$, the class of P_t-free graphs is a subclass of $S_{t,t,t}$-free graphs, thus any algorithm that works for the class of $S_{t,t,t}$-free graphs in particular solves the problem for P_t-free graphs.

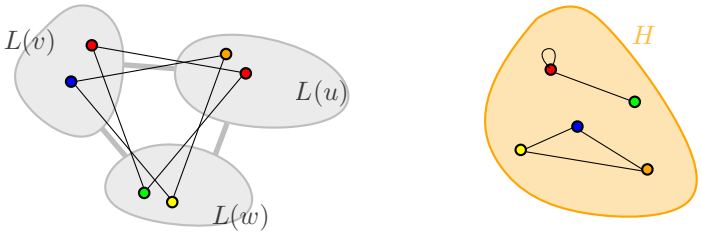


Figure 7.3: An example of a double-triangle that appears on the lists L of an instance (G, L). Here, three vertices v, u, w form a triangle in G, and each of their lists consists of two vertices of H (different vertices of H are represented by different colors). Other edges in H may also exist.

The part where extended strip decompositions come into play is certainly more complicated. First, it turns out that if the graph H does not contain a *non-trivial triangle* then a simple preprocessing of an instance (G, L) is enough to solve it. A *triangle* in a graph H is a triple of vertices (a, b, c) of H such that $ab, bc, ac \in E(H)$. In particular, if $a \in V(H)$ is reflexive, then (a, a, a) is a triangle. A non-trivial triangle is a triangle (a, b, c) such that either $H[\{a, b, c\}]$ is an irreflexive triangle, or two of the vertices a, b, c are equal and incomparable with the third one. We denote by $T(H)$ the set of triangles in H.

If H contains a non-trivial triangle, we decompose the graph G into particles, using the notion of an extended strip decomposition, and proceed by recursion. Then, the main obstacle we face is ensuring that the solution obtained by combining the solutions of the recursive calls for the particles of the decomposition is still a (list) homomorphism. The structure of an extended strip decomposition guarantees that some parts of G in which two different particles "interact" are complete to each other. It turns out that in order to effectively combine the solutions for different particles, we cannot have a triangle in G whose lists contain two, in some sense edge disjoint, triangles of H (see Figure 7.3). Such a property is guaranteed assuming the absence of a *double triangle* in H. A pair $((a_1, a_2, a_3), (b_1, b_2, b_3))$ of elements of $T(H)$ is a *double-triangle* if for every $i \in [3]$ set $\{a_i, b_i\}$ is incomparable. With these notions we can define a *trap* in H (see Figure 7.4).

Definition 7.2.1. Let H be a graph. A triple (τ_1, τ_2, τ_3) of triangles from $T(H)$ is a *trap* if (τ_1, τ_2) is a double-triangle, and τ_3 is a non-trivial triangle.

Now, an undecomposable graph H is *safe* if it does not contain a non-bi-arc connected component that contains a trap.

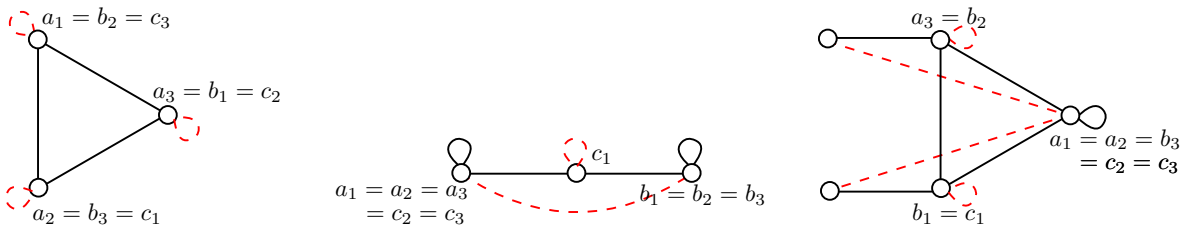


Figure 7.4: Examples of traps $((a_1, a_2, a_3), (b_1, b_2, b_2), (c_1, c_2, c_3))$.

The section is organized as follows: first, in Section 7.2.1 we discuss the preprocessing phase, and provide two important technical observations. In Section 7.2.2 we define extended strip decompositions and prove two technical lemmas that allow us to work with a simplified version of these structures. In Section 7.2.3 we introduce the main structural result, which is the existence of so-called *neutral functions* defined on some subset of the instance graph. Then, in Section 7.2.4, we provide the algorithmic part of Theorem 1.3.5. In Section 7.2.5 we show that if H is undecomposable but not safe, then the LHOM(H) problem is NP-complete in $S_{t,t,t}$-free graphs, and there is no subexponential-time algorithm solving it, assuming the ETH. In other words, safe graphs are precisely the "easy" cases of the LHOM(H) problem in $S_{t,t,t}$-free graphs for undecomposable graphs H, assuming the ETH. We also discuss the possible ways to extend Theorem 1.3.5 to decomposable target graphs.

7.2.1 Consistent instances

Let (G, L) be an instance of LHOM(H). We start by describing simple reduction rules that can be applied to (G, L). For a pair (v, u) of adjacent vertices of G, we define:

$$L(v,u) = \{(a,b) \in L(v) \times L(u) \mid ab \in E(H), \text{ and for every } w \in N(v) \cap N(u) \text{ there is } c \in L(w) \text{ s.t. } abc \in T(H)\}.$$

Note that $(a, b) \in L(v, u)$ if and only if $(b, a) \in L(u, v)$. For a triangle $(v, u, w) \in T(G)$ we define

$$L(v,u,w) = \{(a,b,c) \in V(H)^3 \mid (a,b) \in L(v,u), (b,c) \in L(u,w), (c,a) \in L(w,v)\}.$$

Whenever the ordering of the vertices of a triangle (a, b, c) does not matter, we treat it as an unordered set $\{a, b, c\}$ on at most three vertices and/or write abc instead of (a, b, c).

Since the sets $L(v, u)$ and $L(v, u, w)$ can be computed in polynomial time for all pairs and triples of vertices of G, from now on we always implicitly assume they are given with the instance.

Definition 7.2.2. An instance (G, L) of LHOM(H) is *consistent* if for every $v \in V(G)$:

(a) the set $L(v)$ is incomparable,

(b) $|L(v)| \geqslant 2$,

(c) for every $u \in N(v)$ and for every $a \in L(v)$ there is $b \in L(u)$ such that $(a, b) \in L(v, u)$,

(d) for every $u \in N(v)$, for every $(a, b) \in L(v, u)$ and for every $w \in N(v) \cap N(u)$ there exists $c \in L(w)$ such that $(a, b, c) \in L(v, u, w)$.

From combining Definition 7.2.2 (b) and (d) we get the following.

Observation 7.2.3. *Let H be a fixed, and let (G, L) be a consistent instance of LHOM(H). Then for every $vuw \in T(G)$ we have $|L(v, u, w)| \geqslant 2$.*

We note that an instance (G, L) of LHOM(H) can be efficiently transformed into an equivalent, consistent one.

Lemma 7.2.4. *Let H be a fixed graph and let $I = (G, L)$ be an instance of LHOM(H). Then, in time polynomial in $|V(G)|$ we can either solve I or construct a consistent instance I' of LHOM(H) such that I' is a subinstance of I and I' is a yes-instance if and only if I is a yes-instance.*

Proof. We construct an equivalent instance by performing the following steps exhaustively in the given order (formally after each step we obtain a different instance, but we keep calling it (G, L)). If we remove an element from some $L(v)$, we also remove all corresponding elements from $L(v, u)$ and $L(v, u, w)$ for all appropriate vertices u, w, so that the definitions of $L(v, u)$ and $L(v, u, w)$ are still satisfied. If at any point some set $L(v), L(v, u)$ or $L(v, u, w)$ becomes empty, we return that (G, L) is a no-instance. If at any point we obtain a graph with no edges and non-empty lists, we return that (G, L) is a yes-instance.

1. For every $v \in V(G)$ and every $a, b \in L(v)$, if $N_H(a) \subseteq N_H(b)$, we delete a from $L(v)$.

2. For every $u \in V(G)$ and $a \in L(u)$, if there exists $v \in N_H(u)$ such that no pair of the form (a, b) for some $b \in L(v)$ belongs to $L(u, v)$, we delete a from $L(u)$. If a can be deleted, it means there is no $h : (G, L) \to H$ such that $h(u) = a$.

3. For every pair (u, v) of adjacent vertices of G, and every adjacent $a \in L(u)$ and $b \in L(v)$, we check whether for every $w \in N_H(u) \cap N_H(v)$ there exists $c \in L(w)$ such that $(a, b, c) \in L(u, v, w)$. If not, we delete (a, b) from $L(u, v)$.

4. For every $v \in V(G)$ such that $L(v) = \{a\}$ for some $a \in V(H)$, we remove v from G. This is safe, because if for any neighbor u of v there was $b \in L(u)$ such that $ab \notin E(H)$, then b would have been removed in the second step.

We claim that if none of the above steps can be further applied, then the obtained instance is indeed consistent. Clearly, the first step guarantees that for each $v \in V(G)$ the set $L(v)$ is incomparable, and the last step guarantees that for each $v \in V(G)$ set $L(v)$ has at least two elements. The property (c) follows from the second step, and the third step guarantees that (e) holds.

Recall that G has no loops and observe that since

$$\sum_{v \in V(G)} |L(v)| + \sum_{vu \in E(G)} |L(v, u)| + \sum_{vuw \in T(G)} |L(v, u, w)| \leqslant |V(G)|^3 \cdot |V(H)|^3,$$

and in each step we remove some vertex from some list, the total number of steps above is bounded by $|V(G)|^3 \cdot |V(H)|^3$. Since each step can be performed in time polynomial in $|V(G)|$, the lemma follows. □

7.2.2 Known tools and notions

In this section we introduce a few tools that will be crucial for our algorithm.

Let G be a graph and let $0 < \alpha < 1$. A vertex separator S in G is α-balanced if every connected component of $G - S$ has at most $\alpha \cdot |V(G)|$ vertices. Recall also that by $T(G)$ we denote the set of triangles of a graph G, i.e., triples (a, b, c) such that $ab, bc, ac \in E(G)$.

Definition 7.2.5 (Extended strip decomposition). Let G be a graph. An *extended strip decomposition* (*e.s.d.* in short, see Figure 7.5) (D, η) of G consists of:

- an irreflexive graph D and a function $\eta : V(D) \cup E(D) \cup T(D) \to 2^{V(G)}$,
- for each $xy \in E(D)$, subsets $\eta(xy, x), \eta(xy, y) \subseteq \eta(xy)$,

which satisfy the following properties:

1. $\{\eta(o) \mid o \in V(D) \cup E(D) \cup T(D)\}$ is a partition of $V(G)$,
2. for each $x \in V(D)$ and every distinct $y, z \in N_D(x)$, the set $\eta(xy, x)$ is complete to $\eta(xz, x)$,
3. each $uv \in E(G)$ is contained in one of the sets $\eta(o)$ for $o \in V(D) \cup E(D) \cup T(D)$ or:
   - $u \in \eta(xy, x), v \in \eta(xz, x)$ for some $x \in V(D)$ and $y, z \in N_D(x)$, or
   - $u \in \eta(xy, x), v \in \eta(x)$ for some $xy \in E(D)$, or
   - $u \in \eta(xyz)$ and $v \in \eta(xy, x) \cap \eta(xy, y)$ for some $xyz \in T(D)$.

We will sometimes refer to elements of $V(D) \cup E(D) \cup T(D)$ as *objects* of D. An e.s.d. (D, η) is *rigid* if

(i) for every $xy \in E(D)$ it holds that $\eta(xy, x) \neq \emptyset$, and

(ii) for every $x \in V(D)$ such that x is an isolated vertex it holds that $\eta(x) \neq \emptyset$.

Note that the size of the graph D in a rigid decomposition is polynomial in $|V(G)|$. A rigid e.s.d. (D, η) is *nice* if D has no vertices of degree 1 nor 2.

For an extended strip decomposition (D, η) we introduce two types of special subsets of $V(G)$ called *particles*.[1] For each $x \in V(D)$, a *vertex particle* is the set $A_x := \eta(x)$, and for each $xy \in E(D)$ an *edge particle* is the set

$$A_{xy} := \eta(x) \cup \eta(y) \cup \eta(xy) \cup \bigcup_{xyz \in T(D)} \eta(xyz).$$

Observe that the number of all particles of (D, η) is $\mathcal{O}(|V(D)|^2)$. Thus if (D, η) is rigid, the number of particles is polynomial in $|V(G)|$. For $0 < \alpha < 1$ we say that an extended strip decomposition (D, η) of G is α-*balanced* if every particle contains at most $\alpha \cdot |V(G)|$ vertices of G.

[1] We point out that usually five different types of particles are defined in the literature (see e.g., [18, 52, 90]). For simplicity, we omitted the types that are not relevant for our work.

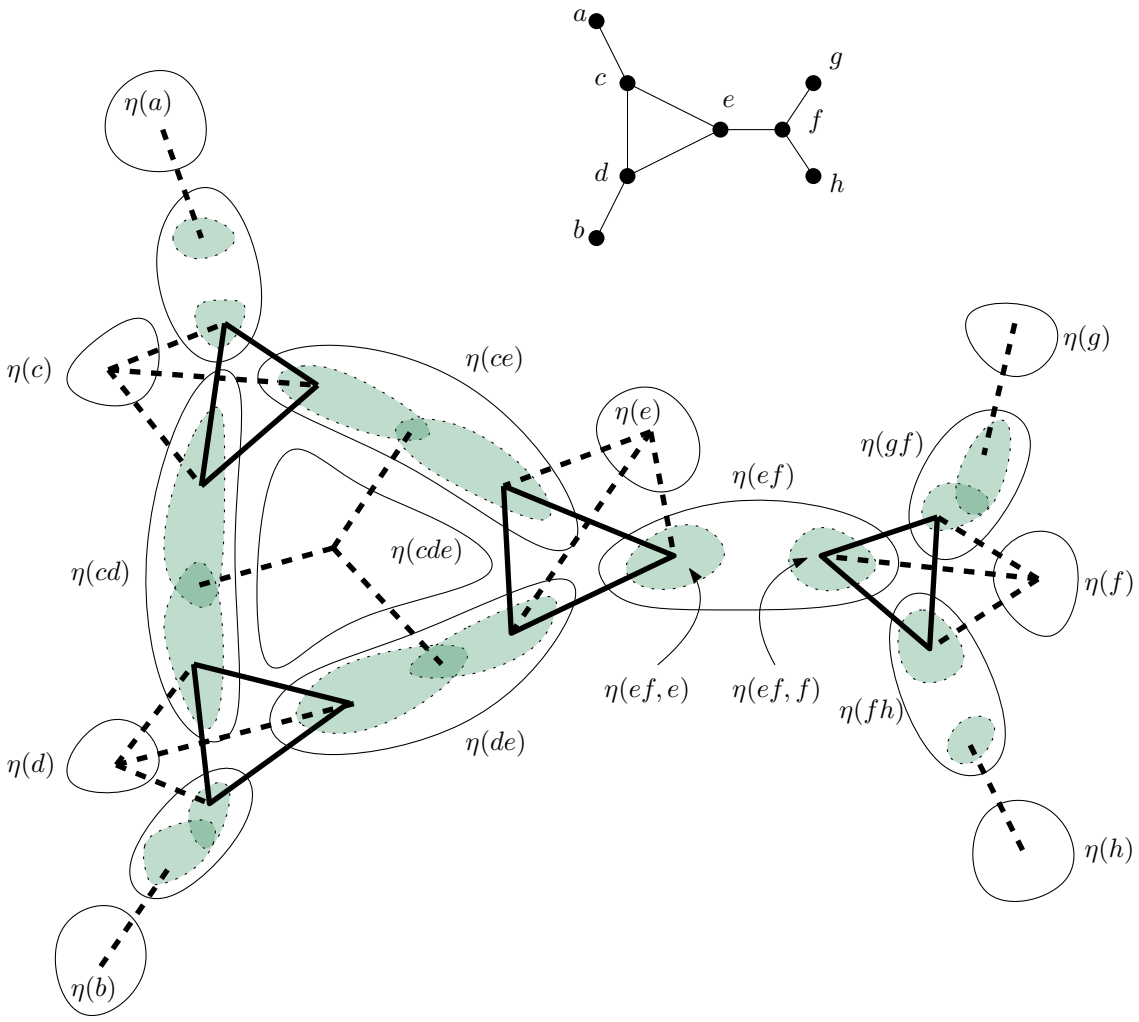


Figure 7.5: A graph D (top), and a schematic view of an e.s.d. (D, η) (bottom). Shapes drawn with narrow black line represent sets $\eta(o)$ for $o \in V(D) \cup E(D) \cup T(D)$, green shapes represent sets $\eta(xy, x)$ and $\eta(xy, y)$ for $xy \in E(D)$. Thick black lines represent edges of G: if two sets are joined by a thick continuous (resp., dashed) line, it means that all (resp., some) of the edges between its elements exist (resp., may exist). Author: Paweł Rzążewski [52].

The following theorem by Majewski et al. [90] says that if we remove a small set of vertices and its neighborhood from an $S_{t,t,t}$-free graph G, we can compute an extended strip decomposition with useful properties for the remaining part of the graph.

Theorem 7.2.6 ([90]). *Let $t \geqslant 1$. Given an n-vertex $S_{t,t,t}$-free graph G, one can in polynomial time output a set $P \subseteq V(G)$ of size $\mathcal{O}(\log n)$, and a $1/2$-balanced rigid extended strip decomposition of $G - N[P]$.*

Let G be a graph and let (D, η) be its extended strip decomposition. For $xy \in E(D)$ by $\partial(xy)$ we denote the set $\eta(xy, x) \cup \eta(xy, y)$. For $x \in V(D)$ we set $\text{potato}(x) = \bigcup_{y \in N_D(x)} \eta(xy, x)$. For a triangle $xyz \in T(D)$ let

$$\text{potato}(xyz) = (\eta(xy, x) \cap \eta(xy, y)) \cup (\eta(xz, x) \cap \eta(xz, z)) \cup (\eta(yz, y) \cap \eta(yz, z)).$$

We note that for every $o \in V(D) \cup T(D)$ we have $N_G(\eta(o)) \subseteq \text{potato}(o)$.

To simplify some descriptions later, we first prove two technical lemmas that allow us to focus on decompositions with some extra properties. The first one claims that for any $W \subseteq V(G)$ we can efficiently modify a rigid e.s.d. of G to a rigid e.s.d of $G - W$. Note that if (D, η) is an extended strip decomposition of G, a decomposition obtained by modifying η by removing the vertices of W from the values of η is also an extended strip decomposition of $G' = G - W$, so the challenge here is modifying it into rigid one.

Lemma 7.2.7. *Let G be an n-vertex graph, and let (D, η) be its rigid extended strip decomposition. Let $W \subseteq V(G)$. Then in time polynomial in n we can find a rigid e.s.d. (D', η') of $G' = G - W$.*

Proof. Let (D, η_0) be an e.s.d. of G' obtained by removing the vertices of W from the values of η. For simplicity, first modify (D, η_0) into an intermediate e.s.d. (D_1, η_1) by (i) getting rid of these components of D that do not give rise to non-empty particles with respect to η_0 and (ii) ensuring that for every remaining $xy \in E(D)$ that belongs to a triangle xyz of D and such that $\eta_0(xy, x) = \emptyset$ we can easily transfer all the vertices of $\eta_0(xyz)$ to some other set. This is important, since in the second step we want to create (D', η') by removing such edges xy.

A component C of D is *η_0-empty* if for every $o \in V(C) \cup E(C) \cup T(C)$ we have $\eta_0(o) = \emptyset$.

Claim 7.2.7.1. *In time polynomial in n we can transform (D, η_0) into an e.s.d. (D_1, η_1) of G' so that*

(A1) there is no η_1-empty component of D_1,

(A2) for every $xyz \in T(D_1)$ such that $\eta_1(xy, x) = \emptyset$ we have $\eta_1(xyz) = \emptyset$.

Proof of Claim. First, we construct an auxiliary graph D_0, by including in D_0 all edges and vertices of D, except these in η_0-empty components of D. As (D, η) is a rigid decomposition of G, the number of particles of (D_0, η_0) is polynomial in n, thus this step can be done in time polynomial in n. Clearly, (D_0, η_0) is an e.s.d. of G' that satisfies (A1).

Now, we iteratively modify (D_0, η_0) into (D_1, η_1) that satisfies (A1) and (A2). As long as there exists $xyz \in T(D_1)$ such that $\eta_0(xy, x) = \emptyset$ and $\eta_0(xyz) \neq \emptyset$, we are going to transfer the elements of the set $\eta_0(xyz)$ either to some vertex particle, or one of the sets $\eta_0(yz)$ or $\eta_0(xz)$. Formally, we do the following.

- If $N_{G'}(\eta_0(xyz)) = \emptyset$, we introduce a new isolated vertex t to D_1 and set $\eta_1(t) = \eta_0(xyz)$ and $\eta_1(xyz) = \emptyset$.
- If $\emptyset \neq N_{G'}(\eta_0(xyz)) \subseteq \eta_0(yz)$, set $\eta_1(yz) = \eta_0(yz) \cup \eta_0(xyz)$ and $\eta_1(xyz) = \emptyset$.
- If $N_{G'}(\eta_0(xyz)) \cap \eta_0(yz) \neq \emptyset$ and $N_{G'}(\eta_0(xyz)) \cap \eta_0(xz) \neq \emptyset$, set $\eta_1(z) = \eta_0(z) \cup \eta_0(xyz)$ and $\eta_1(xyz) = \emptyset$.

If an η_1-empty component was created (note that it is possible in the first case), we remove it from D_1. For all the remaining arguments we set η_1 the same as η_0. Clearly, in every of the cases above (D_1, η_1) is still an e.s.d. of G that satisfies (A1) and if the procedure above cannot be applied anymore, (D_1, η_1) also satisfies (A2). ⌟

Next, we show that in time polynomial in n we can transform (D_1, η_1) into a rigid e.s.d. (D', η') of G'. For that we additionally need to guarantee that for every $xy \in E(D')$ we have that $\eta(xy, x) \neq \emptyset$.

To obtain (D', η') we again consider an iterative procedure. For each edge $xy \in E(D_1)$ where $\eta_1(xy, x) = \emptyset$ we do as follows. By property (A2) of (D_1, η_1), for every $z \in V(D_1)$ such that $xyz \in T(D)$ we have $\eta_1(xyz) = \emptyset$. If $\eta_1(xy) = \emptyset$, we remove xy from the graph. If $\eta_1(xy, y) = \emptyset$ but $\eta_1(xy) \neq \emptyset$, we remove xy, introduce a new isolated vertex t to D, and set $\eta'(t) = \eta_1(xy)$. If $\eta_1(xy, y) \neq \emptyset$, we also remove xy and then introduce a new vertex

t, but now we make t adjacent to y. We set $\eta'(t) = \emptyset, \eta'(yt, y) = \eta'(yt, t) = \eta_1(xy, y)$ and $\eta'(yt) = \eta_1(xy)$. Again, if after any of these steps an η'-empty component was created, we remove it from D'. For all the remaining arguments of η', we set their value equal to their corresponding value of η_1. We note that since $\eta_1(xyz) = \emptyset$, in each case the obtained structure is still an e.s.d. of G'.

We apply the above operation iteratively for every $xy \in E(D_1)$ for which $\eta_1(xy, x) = \emptyset$. Since (D', η') can be found in time polynomial in n, and satisfies the statement of the lemma, this concludes the proof. □

The second lemma asserts that under certain assumptions we can turn a rigid e.s.d. of G into a nice e.s.d. of G without increasing the size of particles.

Lemma 7.2.8. *Let $1/2 \leqslant \alpha < 1$, and let G be an n-vertex graph. Let (D', η') be its α-balanced rigid extended strip decomposition. Assume that there is no α-balanced separator of size at most $6\Delta(G)$ in G. Then in polynomial time we can find an α-balanced nice e.s.d. (D, η) of G.*

Proof. The idea of the construction is to delete vertices of degree 1 from D', and then contract every edge of D' with at least one endpoint of degree 2 (in both cases adjusting the function η appropriately). Formally, we exhaustively apply the following operations.

1. For a vertex $x \in V(D')$ of degree 2 whose neighbors are y and z, we remove x and add an edge yz if it does not exists yet. We include in $\eta(yz)$ every vertex that belongs to $\eta'(xy) \cup \eta'(x) \cup \eta'(xz)$, and, if yz was already in $E(D')$, we also include in $\eta(yz)$ vertices that belong to $\eta'(yz) \cup \eta'(xyz)$. Then we include in $\eta(yz, y)$ (resp., in $\eta(yz, z)$) every vertex that belongs to $\eta'(xy, y)$ (resp. $\eta'(xz, z)$), and, if yz was already in $E(D')$, we also include in $\eta(yz, y)$ (resp. $\eta(yz, z)$) vertices that belong to $\eta'(yz, y)$ (resp. $\eta'(yz, z)$). For all the remaining arguments of η, we set their value equal to their value for η'.

2. For a vertex $x \in V(D')$ of degree 1 whose neighbor is y, we remove x from $V(D')$. We include in $\eta(y)$ every vertex that belongs to $\eta'(y) \cup \eta'(xy) \cup \eta'(x)$. For all the remaining arguments of η, we set their value equal to their value for η'.

It is straightforward to observe that η is still an e.s.d. of G and if none of the above operations can be applied, there is no vertices of degree 1 nor 2 in D'. Let D be the

graph obtained from D' this way. Thus it is enough to show that applying one of the operations above neither impacts the rigidness of the decomposition nor creates a particle that contains more than αn vertices of G. The first property is straightforward by the construction, since the initial decomposition is rigid. We observe that for every $y \in V(D')$ such that $\deg(y) \geqslant 2$ we have that size of potato(y) is at most $2\Delta(G)$. To analyze the size of particles, we consider two cases, depending on the applied operation.

Assume first that we deleted a vertex x of degree 2 from D' (and added the edge yz if it was not there yet). Since the size of every particle of (D', η') was bounded by αn, if now (D, η) is not α-balanced, the edge particle A_{yz} of (D, η) must be larger than $\alpha n \geqslant n/2$. Let $S := \text{potato}_{D'}(y) \cup \text{potato}_{D'}(x) \cup \text{potato}_{D'}(z)$. Note that since the size of each particle of (D', η') is bounded by αn, the size of each connected component of $G - S$ that is contained in A_{yz} is also bounded by αn. Since each other connected component if $G - S$ must be disjoint with A_{yz}, then it must be smaller than $n - \alpha n \leqslant n/2$ and thus S is an α-balanced separator in G of size $6\Delta(G)$, a contradiction.

Now suppose that we applied the second type of operation, i.e., we deleted a vertex x of degree 1 from D. Since the size of every particle of (D', η') was bounded by αn, again if (D, η) is not α-balanced, then either the particle A_y or A_{yz} for some $z \in N_D(y)$ of (D, η) must be larger than αn. Similarly as before, it is then enough to note that $S := \text{potato}_{D'}(y)$ is an α-balanced separator in G of size $2\Delta(G)$.

Since the initial size of the graph D' is polynomial in terms of n, and every operation reduces the number of vertices of D' by one, the whole procedure can be done in time polynomial in n. This concludes the proof of the lemma. □

Maximmum matchings. As already mentioned, our algorithm will reduce an instance of LHom(H) to a certain variant of the Maximum Weight Matching problem. An instance of Maximum Weight Matching* (MWM*) is a tuple $(G, U, \mathfrak{w}, k)$, where G is a graph, U is a subset of $V(G)$, $\mathfrak{w} : E(G) \to \mathbb{N} \cup \{0\}$ is a *weight function*, and $k \in \mathbb{N} \cup \{0\}$. We ask whether there exists a matching M in G such that (i) M covers all vertices from U, and (ii) $\mathfrak{w}(M) \geqslant k$.

By a simple reduction to the Maximum Weight Matching problem we can show that MWM* can be solved in polynomial time. A proof of this fact can be found e.g., in [28].

Lemma 7.2.9 ([28]). *The MWM* problem can be solved in polynomial time.*

7.2.3 Safe graphs and neutral functions

This section is devoted to the main structural result that is crucial for the proof of Theorem 1.3.5 (a). For a graph H, by $R(H)$ we denote the set of reflexive vertices of H. We start with the following lemma.

Lemma 7.2.10. *Let H be a fixed connected non-bi-arc undecomposable graph that does not contain a predator nor a trap. Then the set $R(H)$ induces a reflexive clique.*

Proof. Assume there are two reflexive, non-adjacent vertices $a, b \in V(H)$. Clearly, a and b are incomparable. This means that there is no non-trivial triangle τ in H, as otherwise $((a,a,a),(b,b,b),\tau)$ is a trap in H. Let $\mathcal{P}$ be the set of all induced a-b-paths in H; since H is connected, $\mathcal{P}$ is non-empty.

First, we note that every path $P \in \mathcal{P}$ consists of reflexive vertices. Indeed, otherwise consider $P \in \mathcal{P}$, let $c \in V(P)$ be the first irreflexive vertex on P, and let $c' \in V(P)$ be its predecessor on P (possibly $c' = a$). In particular, c' is reflexive, and c' and c are incomparable. Then (c, c', c') is a non-trivial triangle in H, a contradiction.

Now, we note that every $P \in \mathcal{P}$ has precisely 3 vertices. Otherwise, if P has at least two (adjacent) internal vertices c, d, then (c, d, c, d) is a predator (as the predecessor of c belongs to $N(c) \setminus N(d)$ and the successor of d belongs to $N(d) \setminus N(c)$).

Let $W = \bigcup_{P \in \mathcal{P}} V(P) \setminus \{a, b\}$. Since $\mathcal{P}$ is non-empty and every $P \in \mathcal{P}$ has precisely three vertices, W is non-empty. Moreover, W consists of reflexive vertices, and we note that $W \cup \{a\}$ is a reflexive clique. Otherwise there are vertices $c \in W$ and $a' \in W$ such that $ca' \notin E(H)$, and (a, c, a, c) is a predator (as $b \in N(c) \setminus N(a)$ and $a' \in N(a) \setminus N(c)$). Let A and B be the connected components of $G - W$ that contain, respectively, a and b. Clearly, $A \neq B$ by definition of W.

We claim that A is complete to W. Otherwise, there exists $a' \in A$ and $w \in W$ such that $a'w \notin E(H)$. We choose a' so that $\mathrm{dist}_A(a, a')$ is minimized, clearly $a' \neq a$. Consider a shortest a-a' path P_A in A. We again observe that all vertices of P_A except possibly a' must be reflexive, as otherwise there is a non-trivial triangle in H. Also, since there is no predator in H, P_A has at most three vertices. Let a'' be a predecessor of a' on P_A (possibly $a'' = a$). Now note that (a'', w, w) is a non-trivial triangle in H, since $a' \in N(a'') \setminus N(w)$,

and $b \in N(w) \setminus N(a'')$ as $A \neq B$, a contradiction. Analogously we can show that B is complete to W.

Since $a, b \in A \cup B$ we have $|A \cup B| \geqslant 2$, and remains to observe that

$$(A \cup B, W, V(H) \setminus (A \cup B \cup W))$$

is an F-decomposition of H, a contradiction. □

We need the following definition.

Definition 7.2.11. Let G be a graph and let $X \subseteq V(G)$. We say that a partition $\mathcal{A} = \{A_1, \ldots, A_d\}$ of X is a *mesh partition*, if $d \geqslant 3$ and

- for every $i \in [d]$ we have $A_i \neq \emptyset$,
- for distinct $i, j \in [d]$ sets A_i and A_j are complete to each other.

For a mesh partition $\mathcal{A}$ of X, denote by $G_{\mathcal{A}}[X]$ the graph obtained from $G[X]$ by removing all edges that have both endpoints in one block of $\mathcal{A}$. In other words, $G_{\mathcal{A}}[X]$ is a complete d-partite spanning subgraph of $G[X]$.

Before we proceed to the formal statement and the proof of the main result of this subsection, let us briefly discuss the intuition behind it. Assume H is an undecomposable graph that contains neither a predator not a trap, and let (G, L) be an instance of LHOM(H). We claim that if $X \subseteq V(G)$ admits a mesh partition $\mathcal{A}$, then there exists a "canonical" function $\sigma : X \to V(H)$, called the *neutral function*, such that (i) every homomorphism from $(G[X], L)$ to H differs from σ on at most one block of $\mathcal{A}$, and (ii) for every block A of $\mathcal{A}$ and every $\rho : A \to V(H)$ that respects L, the function that maps every vertex $v \in A$ to $\rho(v)$, and every other vertex v of X to $\sigma(v)$, preserves all edges between different blocks of $\mathcal{A}$, i.e., is a homomorphism from $(G_{\mathcal{A}}[X], L)$ to H.

Having σ, we can exploit the structure of an extended strip decomposition (D, η) of G. Observe that for every $x \in V(D)$ of degree at least 3, the set potato(x) admits a mesh partition. Similarly, so does potato(t) for every $t \in T(D)$. Since two disjoint particles can interact with each other only on such potatoes, and we know that each homomorphisms restricted to a single potato is "almost" equal to σ, we can effectively combine the solutions returned by recursive calls on the particles of an e.s.d. by reducing the problem to a variant of the maximum matching problem.

Lemma 7.2.12. *Let H be a fixed connected non-bi-arc undecomposable graph that does not contain a predator nor a trap. Let (G, L) be a consistent instance of $\text{LHOM}(H)$, and let $X \subseteq V(G)$. Assume that there exists a mesh partition $\mathcal{A} = \{A_1, A_2, A_3\}$ of X. Then:*

- *There exists a unique function $\sigma^{\mathcal{A}} : X \to V(H)$ called the* neutral function *such that*

  1. *for every $v \in X$ it holds that $\sigma^{\mathcal{A}}(v) \in L(v)$,*
  2. *for each $i \in [3]$, and each function $\rho : A_i \to V(H)$ that respects L, the function $h : X \to V(H)$ defined as*

$$h(v) = \begin{cases} \rho(v) & \text{if } v \in A_i, \\ \sigma^{\mathcal{A}}(v) & \text{otherwise,} \end{cases}$$

     is a homomorphism from $G_{\mathcal{A}}[X]$ to H respecting L,
  3. *for each homomorphism $h : (G, L) \to H$ there exists $i \in [3]$ such that $h|_{X \setminus A_i} \equiv \sigma^{\mathcal{A}}|_{X \setminus A_i}$.*

  Moreover, $\sigma^{\mathcal{A}}$ can be found in time polynomial in $|X|$.

- *If there exists a set $Y \subseteq V(G)$ that admits a mesh partition $\mathcal{B} = \{B_1, B_2, B_3\}$, such that $B_1 \subseteq A_1$, $B_2 \subseteq A_2$, and $B_3 \cap (A_1 \cup A_2) = \emptyset$, then for every $v \in B_1 \cup B_2$ we have $\sigma^{\mathcal{A}}(v) = \sigma^{\mathcal{B}}(v)$.*

Proof. We prove Lemma 7.2.12 in a series of steps, each of them in form of a short claim. First, we note that every list contains at most one reflexive vertex.

Claim 7.2.12.1. *Let $v \in V(G)$. Then $L(v)$ contains at most one reflexive vertex. In particular, $L(v)$ contains at least one irreflexive vertex.*

Proof of Claim. If $a, b \in L(v)$ are reflexive, then $ab \in E(H)$ by Lemma 7.2.10. By the consistency of (G, L), a and b are incomparable, and thus (a, b, a, b) is a predator in H, a contradiction. The fact that $L(v)$ contains an irreflexive vertex follows now from the fact that $|L(v)| \geqslant 2$. ⌟

We need the next claim to say that there exists a non-trivial triangle in H.

Claim 7.2.12.2. *The graph H contains a non-trivial triangle.*

Proof of Claim. First, suppose that there are at least two blocks of $\mathcal{A}$, say A_1, A_2, that contain vertices $v \in A_1$, $u \in A_2$ with irreflexive lists, i.e., every $a \in L(v) \cup L(u)$ does not have a loop. Fix some $w \in A_3$ and note that by the definition of a mesh partition, vuw is a triangle in G. Let $a \in L(v)$. Since (G, L) is consistent, there exist $b \in L(u)$ and $c \in L(w)$ such that $(a, b, c) \in L(v, u, w)$, so in particular, $ab, bc, ac \in E(H)$. By the assumption, a and b are irreflexive. If c is also irreflexive, abc is a non-trivial triangle in H and we are done. Thus assume c is reflexive. By Claim 7.2.12.1, there exists irreflexive $c' \in L(w)$. Again, by the consistency of (G, L) we have some irreflexive $a' \in L(v), b' \in L(u)$ (possibly $a' = a$ or $b' = b$) such that $a'b', b'c', c'a' \in E(H)$. Now $a'b'c'$ is an irreflexive (and thus non-trivial) triangle in H.

So now w.l.o.g. assume that all vertices of X with irreflexive lists belong to A_1. Fix some $v \in A_1$, $u \in A_2$ and $w \in A_3$ and let $b \in L(u)$ and $c \in L(w)$ be reflexive. By Lemma 7.2.10, $bc \in E(H)$. Since $|L(u)|, |L(w)| \geqslant 2$, there exist irreflexive $b' \in L(u) \setminus \{b\}$ and $c' \in L(w) \setminus \{c\}$. Clearly, we can assume that $bb', cc' \notin E(H)$ as otherwise either (b', b, b) or (c', c, c) is a non-trivial triangle. Moreover, if $b'c, bc' \in E(H)$ then b and c are incomparable, reflexive vertices and (b, c, b, c) is a predator, a contradiction. Thus without loss of generality we assume that $bc' \notin E(H)$. By consistency there exist $a \in L(v)$ and irreflexive $b'' \in L(u)$ so that $(a, b'', c') \in L(v, u, w)$. Again, $bb'' \notin E(H)$ as otherwise we are done.

If now a is irreflexive, then $ab''c'$ is an irreflexive triangle in H and again we are done. Thus we can assume that a is reflexive, and that there exists irreflexive $a' \in L(v)$. If $aa' \in E(H)$, then (a', a, a) is a non-trivial triangle, so assume otherwise. Again, by consistency, there is $b''' \in L(u)$ and $c'' \in L(w)$ such that $(a', b''', c'') \in L(v, u, w)$. If both are irreflexive, then we have a irreflexive triangle. If $b''' = b$ or $c'' = c$ then (a, b, a, b) or, respectively, (a, c, a, c) is a predator in H (as $b'' \in N(a) \setminus N(b)$ and $a' \in N(b) \setminus N(a)$, or $c' \in N(a) \setminus N(c)$ and $a' \in N(c) \setminus N(a)$), a contradiction. ⌟

Let $\{i, j, \ell\} = [3]$, and let $u \in A_j, w \in A_\ell$. If $a \in L(v)$ for $v \in A_i$ is such that for every $(b, c) \in L(u, w)$ we have $(a, b, c) \in L(v, u, w)$, we say that a is *v-neutral for* (u, w).

Fix a non-trivial triangle τ_3 in H, it exists by the previous claim. We aim to show that for every $v \in A_i$ there exists a unique $a \in L(v)$ such that for every $u \in A_j, w \in A_\ell$ the vertex a is v-neutral for (u, w). This will be done in the next two claims. Then we define $\sigma^{\mathcal{A}}(v) = a$ and show that in this way we obtain a function that satisfies the first

statement of the lemma.

Claim 7.2.12.3. *Let $\{i,j,\ell\} = [3]$, and let $v \in A_i$. For every $u \in A_j, w \in A_\ell$ there exists a v-neutral vertex $a \in L(v)$ for (u,w).*

Proof of Claim. Assume otherwise and let $u \in A_j, w \in A_\ell$ be such that there is no v-neutral vertex for (u,w). Let $a_1 \in L(v)$. Since a_1 is not v-neutral for (u,w), there exist $b_2 \in L(u)$ and $c_2 \in L(w)$ such that $(b_2,c_2) \in L(u,w)$ and $(a_1,b_2,c_2) \notin L(v,u,w)$. This means that either $(a_1,b_2) \notin L(v,u)$ or $(a_1,c_2) \notin L(v,w)$. By symmetry we assume that the first case holds.

Since (G,L) is consistent, by Definition 7.2.2 (c) and (d), there exist $b_1 \in L(u)$ and $c_1 \in L(w)$ such that $\tau_1 := (a_1,b_1,c_1) \in L(v,u,w)$. On the other hand, as $(b_2,c_2) \in L(u,w)$, there exist $a_2 \in L(v)$ such that $\tau_2 := (a_2,b_2,c_2) \in L(v,u,w)$. Clearly, $a_1 \neq a_2$, and $b_1 \neq b_2$, as $(a_1,b_2) \notin L(v,u)$. Observe that a_1 and a_2 (respectively b_1 and b_2) are incomparable by Definition 7.2.2 (a). Now if $c_1 \neq c_2$, then (τ_1,τ_2,τ_3) is a trap in H, a contradiction. Thus, we must have $c_1 = c_2$.

As (G,L) is consistent, $|L(w)| \geqslant 2$ and there exists $c' \neq c_1$ in $L(w)$. Moreover, there exist $a' \in L(v)$ and $b' \in L(u)$ such that $\tau' := (a',b',c') \in L(v,u,w)$. Now observe that we must have $a' = a_2$ and $b' = b_1$, as otherwise either (τ_1,τ',τ_3) or (τ_2,τ',τ_3) is a trap in H ($a' = a_1$ and $b' = b_2$ is impossible since $(a_1,b_2) \notin L(v,u)$). It implies that $(a_2,b_1,c_2) = (a_2,b_1,c_1) \in L(v,u,w)$ (as $(a_2,b_1) = (a',b') \in L(v,u)$, $(a_2,c_2) \in L(v,w)$ and $(b_1,c_1) \in L(u,w)$).

However, by our assumption, a_2 is not a v-neutral vertex for (u,w). So there exist $(b_3,c_3) \in L(u,w)$ such that $(a_2,b_3,c_3) \notin L(v,u,w)$. But since $(b_3,c_3) \in L(u,w)$, there is $a_3 \in L(v)$ such that $\tau_3' = (a_3,b_3,c_3) \in L(v,u,w)$. Clearly, $a_3 \neq a_2$. Now consider (τ_2,τ_3',τ_3), $((a_2,b_1,c_2),\tau_3',\tau_3)$ and $((a_2,b_1,c'),\tau_3',\tau_3)$. As the first one is not a trap, we get $b_2 = b_3$ or $c_2 = c_3$. From the second one we have either $b_1 = b_3$ or $c_2 = c_3$, and from the third one either $b_1 = b_3$ or $c' = c_3$. As $b_1 \neq b_2$ and $c_2 = c_1 \neq c'$, we obtain that $b_1 = b_3$ and $c_2 = c_3$. Thus $(a_2,b_3,c_3) = (a_2,b_1,c_1) \in L(v,u,w)$, a contradiction. ⌟

Now we observe that if $a \in L(v)$ is v-neutral for some (u,w), where $u \in A_j, w \in A_\ell$, then it is v-neutral for every (u',w') such that $u' \in A_j, w' \in A_\ell$.

Claim 7.2.12.4. *Let $i \in [3]$, and let $v \in A_i$. There exists $a \in L(v)$ such that for every j,ℓ such that $\{i,j,\ell\} = [3]$ and every $u \in A_j, w \in A_\ell$ it holds that a is v-neutral for (u,w).*

Proof of Claim. By Claim 7.2.12.3, for every $u \in A_j$ and $w \in A_\ell$ there exists $a \in L(v)$ that is v-neutral for (u, w). We will show that if $u, u' \in A_j$ and $w, w' \in A_\ell$, and $a_1 \in L(v)$ is v-neutral for (u, w) and $a_2 \in L(v)$ is v-neutral for (u', w'), then either a_1 is v-neutral for (u', w') or a_2 is v-neutral for (u, w). Note this is sufficient to obtain the statement of the lemma.

Observe that it is enough to prove this statement assuming that $u = u'$. Indeed, then we can call it once for u and $w \neq w'$, and then for $u' \neq u$ and w'. Therefore, assume for contradiction that we have $a_1, a_2 \in L(v)$ such that there exist $u \in A_j$, $w, w' \in A_\ell$, $b_1, b_2 \in L(u)$, $c_1 \in L(w)$ and $c_2 \in L(w')$, such that $\tau_1 := (a_1, b_1, c_1) \in L(v, u, w), \tau_2 := (a_2, b_2, c_2) \in L(v, u, w')$ but $(a_2, b_1, c_1) \notin L(v, u, w), (a_1, b_2, c_2) \notin L(v, u, w')$. This in particular means that $a_1 \neq a_2$.

First, we show that $b_1 \neq b_2$. Assume otherwise. By Claim 7.2.12.3, there exists a vertex $c_1' \in L(w)$ that is w-neutral for (v, u). In particular, $(a_1, c_1'), (a_2, c_1') \in L(v, w)$. Clearly, $c_1' \neq c_1$, since $(a_2, b_1, c_1') \in L(v, u, w)$. Let $b' \in L(u)$ be a vertex distinct from b_1. By Definition 7.2.2 (d), there are vertices $a' \in L(v)$, and $c' \in L(w)$ such that $(a', b', c') \in L(v, u, w)$. Since neither $(\tau_1, (a', b', c'), \tau_3)$ nor $((a_2, b_1, c_1'), (a', b', c'), \tau_3)$ is a trap in H, we have $a_1 = a'$ or $c_1 = c'$, and $a_2 = a'$ or $c_1' = c'$. Since $a_1 \neq a_2$ and $c_1' \neq c_1$, this is equivalent to saying that we either have $a_1 = a'$ and $c_1' = c'$ or $a_2 = a'$ and $c_1 = c'$. Note however, that the latter case implies that $(a_2, c_1) \in L(v, w)$, and we obtain that (a_1, a_2, c_1, c_1') is a predator in H, a contradiction. Thus, $(a_1, b', c_1') \in L(v, u, w)$. Now there exists a w'-neutral vertex $c_2' \in L(w')$, and since $(a_1, b') \in L(v, u)$, we have $(a_1, b', c_2') \in L(v, u, w')$. Note that since $(a_1, c_2) \notin L(v, w')$, we have $c_2' \neq c_2$. We obtain that $((a_2, b_1, c_2), (a_1, b', c_2'), \tau_3)$ is a trap in H, a contradiction.

Therefore we can assume that $b_1 \neq b_2$. Since $(a_1, b_1) \in L(v, u)$ there exists $c_2' \in L(w')$ such that $(a_1, b_1, c_2') \in L(v, u, w')$. Observe that if $c_2' \neq c_2$ then $((a_1, b_1, c_2'), \tau_2, \tau_3)$ is a trap in H. Thus, $c_2' = c_2$ and hence $(a_1, c_2) \in L(v, w')$ and $(b_1, c_2) \in L(u, w')$. Similarly we can show that since $(a_2, b_2) \in L(v, u)$, we must have that $(a_2, c_1) \in L(v, w)$ and $(b_2, c_1) \in L(u, w)$.

Since (G, L) is consistent, there exist $c_1'' \in L(w)$ distinct from c_1, and $a_1' \in L(v)$, $b_1' \in L(u)$ such that $\tau_1' = (a_1', b_1', c_1'') \in L(v, u, w)$. Observe that we either have $a_1' = a_2$ and $b_1' = b_1$ or $a_1' = a_1$ and $b_1' = b_2$, as otherwise either $((a_1, b_1, c_1), \tau_1', \tau_3)$ or $((a_2, b_2, c_1),_1', \tau_3)$ is a trap in H. However, the first case implies that $(a_2, b_1) \in L(v, u)$, so $(a_2, b_1, c_1) \in$

$L(v,u,w)$, and the second case that $(a_1,b_2) \in L(v,u)$, so $(a_1,b_2,c_2) \in L(v,u,w')$. In both we reach a contradiction. ⌟

We note that the vertex a given by Claim 7.2.12.4 must be unique. Indeed, from the consistency of the instance we know that for every vertex $b \in L(u)$ for some $u \in A_j, j \neq i$ there exists $c \in L(w)$ for every $w \in A_\ell$ and $\ell \in [3] \setminus \{i,j\}$ such that $(b,c) \in L(u,w)$. Thus, if a is v-neutral, a must be in particular adjacent to all vertices in $\bigcup_{u \in X \setminus A_i} L(u)$. If now $L(v)$ contains two vertices a, a' that satisfy Claim 7.2.12.4, then for any two elements b, b' of $L(u)$ for some $u \in A_j \cup A_\ell$ the tuple (a, a', b, b') is a predator, a contradiction.

We are ready to define $\sigma^{\mathcal{A}}$. For every $v \in X$ we set $\sigma^{\mathcal{A}}(v)$ to be the v-neutral vertex $a \in L(v)$, given by Claim 7.2.12.4. The condition 1. follows from the definition of $\sigma^{\mathcal{A}}$, now let us show that the conditions 2. and 3. are satisfied. For that, we first prove yet another property of neutral vertices.

Claim 7.2.12.5. *Let $i \in [3]$, let Let $v \in A_i, u \in A_j$ for distinct $i,j \in [3]$ and let $a \in L(v)$. Then $a\sigma^{\mathcal{A}}(u) \in E(H)$.*

Proof of Claim. Let $w \in A_\ell$, for $\ell \in [3] \setminus \{i,j\}$. By Definition 7.2.2 (c), there exists $c \in L(w)$ such that $(a,c) \in L(v,w)$. As $\sigma^{\mathcal{A}}(u)$ is u-neutral for (v,w), we have that $(a, \sigma^{\mathcal{A}}(u), c) \in L(v,u,w)$, so in particular $(a, \sigma^{\mathcal{A}}(u)) \in L(v,u)$, thus $a\sigma^{\mathcal{A}}(u) \in E(H)$. ⌟

Now, condition 2. is a straightforward consequence of Claim 7.2.12.5. To see 3. consider any homomorphism $h : (G,L) \to H$ and assume that there are distinct $i,j \in [d]$ and $v \in A_i, u \in A_j$ such that $h(v) \neq \sigma^{\mathcal{A}}(v)$ and $h(u) \neq \sigma^{\mathcal{A}}(u)$. As u and v belong to distinct sets of $\mathcal{A}$, we have that $vu \in E(G)$, and therefore $h(v)h(u) \in E(H)$. By Claim 7.2.12.5, $\sigma^{\mathcal{A}}(v)\sigma^{\mathcal{A}}(u), \sigma^{\mathcal{A}}(v)h(u), h(v)\sigma^{\mathcal{A}}(u) \in E(H)$. Thus, $(\sigma^{\mathcal{A}}(v), h(v), \sigma^{\mathcal{A}}(u), h(u))$ is a predator, a contradiction.

To conclude the proof of the first statement of Lemma 7.2.12 we note that since the proof is constructive, $\sigma^{\mathcal{A}}$ can be found in time polynomial in $|X|$.

It remains to prove the second statement of the lemma. Note that by symmetry and by the definitions of $\sigma^{\mathcal{A}}$ and $\sigma^{\mathcal{B}}$, it is enough to show that if $v \in A_1$ and $a \in L(v)$ is v-neutral for some (u,w), where $u \in A_2, w \in A_3$, then it is v-neutral for every (u',w') such that $u' \in A_2, w' \in B_3$.

Claim 7.2.12.6. *Let $v \in A_1$. There exists $a \in L(v)$ such that for every $u, u' \in A_2, w \in A_3$ and $w' \in B_3$ vertex a is v-neutral for (u,w) and for (u',w').*

Proof of Claim. Let $a_1, a_2 \in L(v)$. We again observe that it is enough to take $u = u'$ and prove that if a_1 is neutral for (u, w), and a_2 is v-neutral for (u, w'), then either a_1 is neutral for (u, w') or a_2 is neutral for (u, w). By Definition 7.2.2 (b), there exist distinct $b_1, b_2 \in L(u)$. By Definition 7.2.2 (c) there exist $c_1 \in L(w)$ and $c_2 \in L(w')$ such that $(b_1, c_1) \in L(u, w)$ and $(b_2, c_2) \in L(u, w')$. Since a_1 is v-neutral for (u, w), $a_1b_1, a_1b_2 \in E(H)$. Similarly, as a_2 is v-neutral for (u, w'), we have $a_2b_1, a_2b_2 \in E(H)$. Thus (a_1, a_2, b_1, b_2) is a predator in H, unless $a_1 = a_2$. ⌟

Similarly as before, we note that if there are two vertices $a, a' \in L(v)$ that satisfy Claim 7.2.12.6, then for any $u \in A_2$ and $b, b' \in L(u)$ tuple (a, a', b, b') is a predator. That concludes the proof of Lemma 7.2.12. □

As a corollary we obtain the following.

Corollary 7.2.13. *Let H be a fixed connected non-bi-arc undecomposable graph that does not contain a predator nor a trap. Let (G, L) be a consistent instance of LHOM(H), and let $X \subseteq V(G)$. Let $d \geqslant 3$ and assume that there exists a mesh partition $\mathcal{A} = \{A_1, \ldots, A_d\}$ of X. Then:*

(a) There exists a unique function $\sigma^{\mathcal{A}} : X \to V(H)$ called the neutral function *such that*

1. *for every $v \in X$ it holds that $\sigma^{\mathcal{A}}(v) \in L(v)$,*
2. *for each $i \in [d]$, and each function $\rho : A_i \to V(H)$ that respects L, function $h : X \to V(H)$ defined as*

$$h(v) = \begin{cases} \rho(v) & \text{if } v \in A_i, \\ \sigma^{\mathcal{A}}(v) & \text{otherwise,} \end{cases}$$

   is a homomorphism from $G_{\mathcal{A}}[X]$ to H respecting L,
3. *for each homomorphism $h : (G, L) \to H$ there exists $i \in [d]$ such that $h|_{X \setminus A_i} \equiv \sigma^{\mathcal{A}}|_{X \setminus A_i}$.*

Moreover, $\sigma^{\mathcal{A}}$ can be found in time polynomial in $|X|$.

(b) If there exists a set $Y \subseteq V(G)$ that admits a mesh partition $\mathcal{B} = \{B_1, B_2, B_3\}$, such that $B_1 \subseteq A_1$, $B_2 \subseteq A_2$, and $B_3 \cap (A_1 \cup A_2) = \emptyset$, then then for every $v \in B_1 \cup B_2$ we have $\sigma^{\mathcal{A}}(v) = \sigma^{\mathcal{B}}(v)$.

Proof. The case $d = 3$ follows from the statement of Lemma 7.2.12, so we can assume that $d \geqslant 4$. Let $\mathcal{A}' = \{A_1, A_2, A_3 \cup \ldots \cup A_{d-1}\}$, $\mathcal{B}_1 = \{A_1, A_2, A_d\}$ and $\mathcal{B}_2 = \{A_2, A_3 \cup \ldots \cup A_{d-1}, A_d\}$, and let $\sigma^{\mathcal{A}}, \sigma^{\mathcal{B}_1}, \sigma^{\mathcal{B}_2}$ be given by Lemma 7.2.12. We define

$$\sigma^{\mathcal{A}}(v) = \begin{cases} \sigma^{\mathcal{A}'}(v) & \text{if } v \in \bigcup_{A \in \mathcal{A}'} A, \\ \sigma^{\mathcal{B}_1}(v) & \text{if } v \in A_d. \end{cases}$$

Now by the second item of Lemma 7.2.12 we have

- for every $v \in A_1 \cup A_2$ we have $\sigma^{\mathcal{A}'}(v) = \sigma^{\mathcal{B}_1}(v)$,
- for every $v \in A_2 \cup \ldots \cup A_{d-1}$ we have $\sigma^{\mathcal{A}'}(v) = \sigma^{\mathcal{B}_2}(v)$,
- for every $v \in A_2 \cup A_d$ we have $\sigma^{\mathcal{B}_1}(v) = \sigma^{\mathcal{B}_2}(v)$.

It is straightforward to verify that statements of the lemma follow from the respective conditions given by Lemma 7.2.12 for $\sigma^{\mathcal{A}}, \sigma^{\mathcal{B}_1}$ and $\sigma^{\mathcal{B}_2}$. □

7.2.4 The algorithm

The only remaining tool that we need before we prove Theorem 1.3.5 (a) is how to compute a solution for an instance (G, L) recursively, using an extended strip decomposition (D, η) of G. We say that a subinstance (G', L') of (G, L) is a *local subinstance* if G' is contained in some particle of (D, η).

Lemma 7.2.14. *Let H be a fixed connected non-bi-arc undecomposable graph that does not contain a predator nor a trap, let $t \in \mathbb{N}$, and let (G, L) be a consistent instance of LHOM(H). Let (D, η) be a nice e.s.d. of G. Suppose that we have an algorithm $\mathcal{A}$ that solves LHOM(H) on every local subinstance of (G, L). Then we can solve (G, L) by running $\mathcal{A}$ on a (possibly empty) family $\mathbb{I}$ of at most $n^{\mathcal{O}(1)}$ local subinstances of (G, L). All the additional computations can be done in time polynomial in $|V(G)|$.*

Proof. We prove the lemma by reducing (G, L) to an instance $(D', U, \mathfrak{w}, k)$ of MWM* problem, so that we can use Lemma 7.2.9 to solve it.

Let $V' \subseteq V(D)$ be the set of vertices of positive degree in D. First, we note that for every $x \in V'$ we have that $G[\text{potato}(x)]$ admits a mesh partition $\mathcal{P}_x = \{\eta(xy, x)\}_{y \in N_D(x)}$ (recall that in a nice e.s.d. every vertex in V' has degree at least 3). Thus, for every

$x \in V(D)$, we can call Corollary 7.2.13 to obtain the neutral function $\sigma^{\mathcal{P}_x} : \text{potato}(x) \to V(H)$. For simplicity, we will be writing σ^x instead of $\sigma^{\mathcal{P}_x}$.

Next, we note that for every $t = xyz \in T(D)$ we have that $G[\text{potato}(t)]$ admits a mesh partition $\mathcal{Q}_t = \{\eta(xy, x) \cap \eta(xy, y), \eta(xz, x) \cap \eta(xz, z), \eta(yz, y) \cap \eta(yz, z)\}$. We call Lemma 7.2.12 to obtain the function $\sigma^{\mathcal{Q}_t} : \text{potato}(t) \to V(H)$ as in the statement of the lemma. We denote it by σ^t for simplicity.

Claim 7.2.14.1. *Let $t = xyz \in T(D)$. For every $v \in \eta(xy, x) \cap \eta(xy, y)$ we have that $\sigma^x(v) = \sigma^y(v) = \sigma^t(v)$.*

Proof of Claim. Observe that by symmetry of x and y it is enough to show that $\sigma^x(v) = \sigma^t(v)$. This follows from applying Corollary 7.2.13 to $\mathcal{A} = \{\eta(xz, x)\}_{z \in N(x)}$ and $\mathcal{B} = \{\eta(xy, x) \cap \eta(xy, y), \eta(xz, x) \cap \eta(xz, z), \eta(yz, y) \cap \eta(yz, z)\}$. ⌟

Auxiliary definitions. Let $f : W \to V(H)$ for some $W \subseteq V(G)$ be such that for every $v \in W$ we have $f(v) \in L(v)$. We define a function $L_f : V(G) \to 2^{V(H)}$ as

$$L_f(v) = \begin{cases} L(v) \cap \bigcap_{u \in N_G(v) \cap W} N_H(f(u)) & \text{if } N_G(v) \cap W \neq \emptyset, \\ L(v) & \text{otherwise.} \end{cases}$$

Observe that f is a fixed mapping on W that respects lists L. Intuitively, L_f are obtained by trimming lists L so that for any homomorphism $g : (G - W, L_f) \to H$, if f is also homomorphism, then the disjoint union of f and g (as they have disjoint domains) is a homomorphism from G to H. Formally, we state it as follows.

Claim 7.2.14.2. *Let $f : W \to V(H)$ for some $W \subseteq V(G)$. Let $g : V(G) \setminus W \to V(H)$ be such that for every $v \in V(G) \setminus W$ we have $g(v) \in L_f(v)$. For every adjacent $u \in W, v \in V(G) \setminus W$ it holds that $f(u)g(v) \in E(H)$.*

Let $xy \in E(D)$. We say that a function $f : \partial(xy) \to V(H)$ is *x-neutral* (resp. *y-neutral*) if f is equivalent to σ^x on the set $\eta(xy, x)$ (resp. to σ^y on the set $\eta(xy, y)$). We say that $f : \partial(xy) \to V(H)$ is *neutral* if f is both x- and y-neutral. Observe that there is at most one neutral function for each $xy \in E(D)$: recall that $\partial(xy) = \eta(xy, x) \cup \eta(xy, y)$ and $\sigma^x|_{\eta(xy,x)}$ and $\sigma^y|_{\eta(xy,y)}$ are unique, but for some $v \in \eta(xy, x) \cap \eta(xy, y)$ it may happen that $\sigma^x(v) \neq \sigma^y(v)$ (if xy does not belong to a triangle of D).

For every $xy \in E(D)$ we define up to four homomorphisms as follows.

(S1) If there exists $f : (G[\eta(xy)], L) \to H$ such that $f|_{\partial(xy)}$ is neutral, we fix one such homomorphism and denote it by $f^{\perp}_{xy}$.

(S2) Let $g \equiv \sigma^y|_{\text{potato}(y)\setminus\eta(xy,y)}$. If there exists a homomorphism $f : (G[\eta(xy)\cup\eta(y)], L_g) \to H$ such that $f|_{\eta(xy,x)} \equiv \sigma^x$, we fix one such homomorphism and denote it by f^y_{xy}. By switching the role of x and y we analogously define f^x_{xy} (if it exists).

(S3) Let $g \equiv \sigma^x|_{\text{potato}(x)\setminus\eta(xy)} \cup \sigma^y|_{\text{potato}(y)\setminus\eta(xy)}$. If there exists $f : (G[A^{xy}_{xy}], L_g) \to H$, we fix one such homomorphism and denote it by f^{xy}_{xy}.

Let $\mathbb{I}_{xy}$ be the set of at most four instances of $\text{LHOM}(H)$ considered in the steps (S1)-(S3). Clearly, each element of $\mathbb{I}_{xy}$ is a local subinstance of (G, L). For every $xy \in E(D)$ let $\mathcal{F}_{xy}$ consist of these elements $f^{\perp}_{xy}, f^y_{xy}, f^x_{xy}$ or f^{xy}_{xy}, that were defined above. We observe that the definition of $f^{\perp}_{xy}, f^y_{xy}$ and f^{xy}_{xy} together with the property 3 of neutral functions (stated in Corollary 7.2.13) imply the following.

Claim 7.2.14.3. *Let $h : (G, L) \to H$. Then for every $xy \in E(D)$:*

1. *if $h|_{\partial(xy)}$ is neutral, then $f^{\perp}_{xy} \in \mathcal{F}_{xy}$,*
2. *if $h|_{\partial(xy)}$ is neither x-neutral nor y-neutral, then $f^{xy}_{xy} \in \mathcal{F}_{xy}$,*
3. *if $h|_{\partial(xy)}$ is x-neutral but not y-neutral, then $f^y_{xy} \in \mathcal{F}_{xy}$.*

Recall that for every $x \in V(D)$ we have that $G[\text{potato}(x)]$ admits a mesh partition $\{\eta(xy, x)\}_{y\in N_D(x)}$. Thus, by Corollary 7.2.13.3, for every $h : (G, L) \to H$ there is at most one $y \in N_D(x)$ such that $h|_{\partial(xy)}$ is not x-neutral. Similarly, since for every $xyz \in T(D)$, the graph $G[\text{potato}(xyz)]$ admits a mesh partition $\{\eta(xy, x) \cap \eta(xy, y), \eta(xz, x) \cap \eta(xz, z), \eta(yz, y) \cap \eta(yz, z)\}$, there is at most one element $e \in \{xy, yz, xz\}$ such that $h|_{\partial(e)}$ is not neutral. We obtain the following.

Claim 7.2.14.4. *For every $h : (G, L) \to H$ and for every $x \in V(D)$, there is at most one $y \in N_D(x)$ such that $h|_{\partial(xy)}$ is not x-neutral. Similarly, for every $xyz \in T(D)$ at most one of $h|_{\partial(xy)}$, $h|_{\partial(yz)}$, and $h|_{\partial(xz)}$ is not neutral.*

Next, we define subsets $\mathcal{V}, \mathcal{E}$, and $\mathcal{T}$ of, respectively $V(D), E(D)$, and $T(D)$. Intuitively, they consist of these objects o, for which the neutral function defined on some subset of $N_G[\eta(o)]$ (in case of $\mathcal{V}$ and $\mathcal{T}$) or of $\eta(o)$ (in case of $\mathcal{E}$) cannot be extended to

$\eta(o)$. Formally, let $\mathcal{E}$ be the set of these edges $xy \in E(D)$ for which $f_{xy}^{\perp}$ does not exist, and let

$$\mathcal{V} = \{x \in V(D) \mid (G[\eta(x)], L_{\sigma^x}) \text{ is a no-instance}\},$$
$$\mathcal{T} = \{t \in T(D) \mid (G[\eta(t)], L_{\sigma^t}) \text{ is a no-instance}\}.$$

For every $o \in V(D) \cup T(D) \setminus (\mathcal{V} \cup \mathcal{T})$, let f_o be a fixed homomorphism from $(G[\eta(o)], L_{\sigma^o})$ to H. Last, let $\delta : E(D) \to \mathbb{N}$ be such that $\delta(xy) = |\{z \in V(D) \mid xyz \in \mathcal{T}\}|$.

Constructing an instance of the matching problem. Now we show how to construct the instance $(D', \mathcal{V}, \mathfrak{w}, k)$ of MWM* as follows.

The set of vertices of D' is $V(D) \cup \{v_e \mid e \in E(D)\}$. We define $E' = E(D) \cup \{xv_{xy}, yv_{xy} \mid xy \in E(D)\}$. The graph D' is a (spanning) subgraph of $(V(D'), E')$, obtained by deleting elements $e \in E'$ if one of the following conditions is satisfied.

(R1) If $e = xy \in E(D)$ and $f_{xy}^{xy} \notin \mathcal{F}_{xy}$.

(R2) If $e = xv_{xy} \notin E(D)$ and $f_{xy}^{x} \notin \mathcal{F}_{xy}$.

We extend the definition of δ by setting $\delta(xy) = 0$ for every $xy \in E(D') \setminus E(D)$. For every $e \in E(D')$ we define

$$\mathfrak{s}(e) = \begin{cases} 1 & \text{if } e \in \mathcal{E} \text{ or } e \in \{xv_{xy}, yv_{xy}\} \text{ for some } xy \in \mathcal{E}, \\ 0 & \text{otherwise.} \end{cases}$$

For every $e \in E(D')$ define $\mathfrak{w}(e) = \delta(e) + \mathfrak{s}(e)$, and let $k = |\mathcal{T}| + |\mathcal{E}|$. That concludes the definition of $(D', \mathfrak{w}, \mathcal{V}, k)$. It remains to show that $(D', \mathfrak{w}, \mathcal{V}, k)$ is equivalent to (G, L). This is proven in the following two claims (each covering one of the implications).

Claim 7.2.14.5. *If $(G, L) \to H$ then there exists a matching in D' of weight at least k that covers $\mathcal{V}$.*

Proof of Claim. First, we construct a subset $M \subseteq E'$. We add $e \in E'$ to M if the following conditions are satisfied:

(C1) $e = xy \in E(D)$, and $h|_{\partial(xy)}$ is neither x-neutral nor y-neutral,

(C2) $e = xv_{xy}$ for some $xy \in E(D)$, and $h|_{\partial(xy)}$ is y-neutral but not x-neutral.

We need to prove that (M1) $M \subseteq E(D')$, (M2) M is a matching, (M3) M covers $\mathcal{V}$, and (M4) $\sum_{e \in M} \mathfrak{w}(e) \geqslant k$.

First, we verify that (M1) follows from the definition of $E(D')$ and M. Consider $e \in M$.

1. If $e = xy \in E(D)$, by (C1), $h|_{\partial(xy)}$ is neither x-neutral nor y-neutral. Claim 7.2.14.3 asserts that $f^{xy}_{xy} \in \mathcal{F}_{xy}$, and therefore the edge e is not deleted in step (R1).

2. If $e = xv_{xy}$ for some $xy \in E(D)$, by (C2), $h|_{\partial(xy)}$ is y-neutral but not x-neutral. By Claim 7.2.14.3 $f^{x}_{xy} \in \mathcal{F}_{xy}$, and therefore the edge e is not deleted in step (R2).

To see that (M2) holds, assume that there exists $x \in V(D')$ and distinct $y, z \in N_{D'}(x)$ such that $xy, xz \in M$. If $x \notin V(D)$, then $y, z \in V(D)$ and $x = v_{yz}$. Since $yv_{yz} \in M$, we have that $h|_{\partial(yz)}$ is z-neutral but not y-neutral. On the other hand, $zv_{yz} \in M$ implies that $h|_{\partial(yz)}$ is y-neutral but not z-neutral, a contradiction. Similarly we can show contradiction if $x, y \in V(D)$ and $z = v_{xy}$: since $xy \in M$, $h|_{\partial(xy)}$ is not x-neutral nor y-neutral. However, since $xv_{xy} \in M$, we have that $h|_{\partial(xy)}$ is y-neutral (and not x-neutral).

Hence, the remaining case is that $x \in V(D)$, $y \in \{y', v_{xy'}\}$ for some $y' \in N_D(x)$, and $z \in \{z', v_{xz'}\}$ for some $z' \in N_D(x) \setminus \{y'\}$. The fact that $xy \in M$ implies that $h|_{\partial(xy')}$ is not x-neutral, and $xz \in M$ implies that $h|_{\partial(xz')}$ is not x-neutral. However, recall that by Claim 7.2.14.4 there is at most one $x' \in N_D(x)$ such that $h|_{\partial(xx')}$ is not x-neutral, so we again reach a contradiction. We can conclude that M is a matching.

To observe that (M3) M covers $\mathcal{V}$, consider a vertex $x \in \mathcal{V}$. By the definition of $\mathcal{V}$, $(G[A_x], L_{\sigma^x}) \not\to H$. This and the definition of L_{σ^x} imply that there is no $f : (G[\eta(x) \cup \text{potato}(x)], L) \to H$ such that $f \equiv \sigma^x$ on potato(x). By Lemma 7.2.12.3. this is equivalent to saying that there exists $y \in N_D(x)$ such that the homomorphism $h|_{\partial(xy)}$ is not x-neutral. Hence, by the definition of M, either $xy \in M$ or $xv_{xy} \in M$ and thus x is covered.

It remains to prove (M4), i.e., to show the inequality

$$\sum_{e \in M} \mathfrak{w}(e) = \sum_{e \in M} \delta(e) + \sum_{e \in M} \mathfrak{s}(e) \geqslant |\mathcal{T}| + |\mathcal{E}| = k.$$

Observe that if $\sum_{e \in M} \delta(e) = \sum_{xy \in M} |\{z \in V(D) \mid xyz \in \mathcal{T}\}| < |\mathcal{T}|$, it means that there exists a triangle $xyz \in \mathcal{T}$ such that $\{xy, xz, yx\} \cap M = \emptyset$. However, the definition of M

implies that in a such case $h|_{\partial(xy)}$, $h|_{\partial(xz)}$ and $h|_{\partial(yz)}$ are neutral. This is equivalent to saying that h is equivalent to σ^t on potato(t) (as, by Claim 7.2.14.1, $\sigma^t \equiv \sigma^x \equiv \sigma^y$ on potato(t)). However, the latter is a contradiction with the definition of $\mathcal{T}$.

On the other hand, if $\sum_{e\in M} \mathfrak{s}(e) < |\mathcal{E}|$, then there exists an edge $xy \in \mathcal{E}$ such that $\{xy, xv_{xy}, yv_{xy}\} \cap M = \emptyset$. By the definition of M this means that $h|_{\partial(xy)}$ is neutral, and this contradicts the fact that $xy \in \mathcal{E}$. In summary, we get that $\sum_{e\in M} \delta(e) + \sum_{e\in M} \mathfrak{s}(e) \geqslant |\mathcal{T}| + |\mathcal{E}|$, which proves the desired inequality, and proves that if (G, L) is a yes-instance of LHOM(H), then $(D', \mathfrak{w}, \mathcal{V}, k)$ is a yes-instance of MWM*. ⌟

Claim 7.2.14.6. *If there exists a matching in D' of weight at least k that covers $\mathcal{V}$ then $(G, L) \to H$.*

Proof of Claim. Assume that $(D', \mathfrak{w}, \mathcal{V}, k)$ is a yes-instance of MWM* that admits a solution M. Recall that $|\mathcal{T}| + |\mathcal{E}| = k \leqslant \sum_{e\in M} \mathfrak{w}(e) = \sum_{e\in M} \delta(e) + \sum_{e\in M} \mathfrak{s}(e)$. We start by showing that $\sum_{e\in M} \delta(e) \leqslant |\mathcal{T}|$ and $\sum_{e\in M} \mathfrak{s}(e) \leqslant |\mathcal{E}|$. Assume otherwise and note that then we either have $\sum_{e\in M} \delta(e) \geqslant |\mathcal{T}| + 1$ or $\sum_{e\in M} \mathfrak{s}(e) \geqslant |\mathcal{E}| + 1$. Suppose first that

$$\sum_{e\in M} \delta(e) = \sum_{xy\in E(D)\cap M} |\{z \in V(D) \mid xyz \in \mathcal{T}\}| \geqslant |\mathcal{T}| + 1$$

holds. This means, precisely, that there exists $xyz \in \mathcal{T}$ such that $|\{xy, yz, xz\} \cap M| \geqslant 2$, which is impossible, since xy, xz, yz induce a triangle in D' and M is a matching. Similarly, if

$$\sum_{e\in M} \mathfrak{s}(e) = \sum_{xy\in\mathcal{E}} \sum_{e\in\{xy, xv_{xy}, yv_{xy}\}} |\{e\} \cap M| \geqslant |\mathcal{E}| + 1,$$

then there exist an edge $xy \in \mathcal{E}$ such that $|\{xy, xv_{xy}, yv_{xy}\} \cap M| \geqslant 2$. This again is impossible, since xy, xv_{xy}, yv_{xy} induce a triangle in D' and M is a matching.

Now, since $|\mathcal{T}| + |\mathcal{E}| \leqslant \sum_{e\in M} \delta(e) + \sum_{e\in M} \mathfrak{s}(e)$, from the fact that $\sum_{e\in M} \delta(e) \leqslant |\mathcal{T}|$ and $\sum_{e\in M} \mathfrak{s}(e) \leqslant |\mathcal{E}|$, we can conclude that $\sum_{e\in M} \delta(e) = |\mathcal{T}|$ and $\sum_{e\in M} \mathfrak{s}(e) = |\mathcal{E}|$. This, in turn, implies that for every $t = xyz \in \mathcal{T}$, and for every $e = xy \in \mathcal{E}$, we have, respectively, that $|\{xy, yz, xz\} \cap M| = 1$, and that $|\{xy, xv_{xy}, yv_{xy}\} \cap M| = 1$.

Recall that for $o \in V(D) \cup T(D) \setminus (\mathcal{V} \cup \mathcal{T})$ the function f_o is some fixed homomorphism from $(G[\eta(o)], L_{\sigma^o})$ to H. We will define the function $h : V(G) \to V(H)$ using the

functions from the set

$$\mathcal{F} = \{f_o \mid o \in V(D) \cup T(D) \setminus (\mathcal{V} \cup \mathcal{T})\} \cup \bigcup_{xy \in E(D)} \mathcal{F}_{xy},$$

and then show that h is a homomorphism that respects lists L.

For every $o \in V(D) \cup E(D) \cup T(D)$ and for every $v \in V(G)$ we set $h(v)$ as follows.

(F1) If $v \in \eta(x)$ such that x is not covered by M, then $h(v) = f_x(v)$.

(F2) If $v \in \eta(xyz)$ such that $\{xy, yz, zx\} \cap M = \emptyset$, then $h(v) = f_{xyz}(v)$.

(F3) If $v \in \eta(xy)$ such that $\{xy, xv_{xy}, y_{xy}\} \cap M = \emptyset$, then $h(v) = f^{\perp}_{xy}(v)$.

(F4) If $v \in \eta(xy) \cup \eta(x)$ such that $xv_{xy} \in M$, then $h(v) = f^{x}_{xy}(v)$.

(F5) If $v \in A^{xy}_{xy}$ such that $xy \in M$, then $h(v) = f^{xy}_{xy}(v)$.

We need to prove that (H1) for every $v \in V(G)$ the vertex $h(v)$ is well-defined, in particular that the appropriate function $f \in \mathcal{F}$ exists, (H2) h respects the lists L, and (H3) h is a homomorphism.

We start by proving (H1). Consider first the case that $v \in \eta(x)$ for some $x \in V(D)$. Clearly, since M is a matching in D', x is covered by at most one edge of M. If x is not covered by M, then v is considered precisely once in (F1). We note that $f_x \in \mathcal{F}$ since $x \notin \mathcal{V}$ (as it is not covered by M). If x is covered by an edge $e \in M$, then e is either of type xv_{xy} or xy for some $y \in N_D(x)$. Hence, v is considered precisely once in, respectively, (F4) or (F5), and the fact that f^{x}_{xy} or, respectively f^{xy}_{xy}, belongs to $\mathcal{F}$ follows from the existence of the edge xv_{xy} or xy in D'.

Assume that $v \in \eta(t)$ for some $t = xyz \in T(D)$. Again, since M is a matching, at most one edge of t belongs to M. If none of them belongs to M, then v is considered precisely once in (F2). Note that $f_t \in \mathcal{F}$ as otherwise $t \in \mathcal{T}$, and we already showed that if $t \in \mathcal{T}$ then $|\{xy, yz, xz\} \cap M| = 1$. Otherwise, one of the edges of t, say xy, belongs to M. Then v is considered precisely once in (F5), and the fact that $f^{xy}_{xy} \in \mathcal{F}$ follows from the fact that $xy \in E(D')$.

Last, assume that $v \in \eta(xy)$ for some $xy \in E(D)$. If $\{xy, xv_{xy}, yv_{xy}\} \cap M = \emptyset$, then v is considered in (F3). Since $|\{xy, xv_{xy}, yv_{xy}\} \cap M| = 0$, $xy \notin \mathcal{S}$, and thus $f^{\perp}_{xy} \in \mathcal{F}$. If $xy \in M$, then $xv_{xy}, yv_{xy} \notin M$ and v is considered only in (F5). Again, the fact that

$f^{xy}_{xy} \in \mathcal{F}$ follows from the fact that $xy \in E(D')$. Last, by symmetry we can assume that $xv_{xy} \in M$. Then $xy, yv_{xy} \notin M$, and v is considered once in (F4), and the fact that $f^{x}_{xy} \in \mathcal{F}$ follows from $xv_{xy} \in E(D')$. That concludes the proof of (H1).

Observe that the property (H2) follows from the definitions of elements of $\mathcal{F}$. Indeed, each $f \in \mathcal{F}$ is a homomorphism from some G' to H that respects lists L', where (G', L') is a subinstance of (G, L), and thus for every $v \in V(G)$ we must have $h(v) \in L(v)$.

It remains to show (H3), i.e., that for an edge $uv \in E(G)$ we have $h(u)h(v) \in E(H)$. Note that by the definition of an extended strip decomposition, we can divide the analysis into the following three cases:

(i) there is $o \in V(D) \cup E(D) \cup T(D)$ such that $u, v \in \eta(o)$,

(ii) there are $o \in V(D) \cup T(D)$ and $e \in E(D)$ such that $u \in \eta(o)$ and $v \in \text{potato}(o)$,

(iii) there are $xy, yz \in E(D)$ such that $u \in \eta(xy, y)$, $v \in \eta(yz, y)$.

Before we proceed to the subcases, we note that the definition of h implies that if $v \in \eta(xy, x)$ for some $xy \in E(D)$, and $xy, xv_{xy} \notin M$, then $h|_{\partial(xy)}$ is x-neutral, so in particular $h(v) = \sigma^x(v)$.

Observe that case (i) is straightforward: if $u, v \in \eta(o)$, by definition of h there exists a unique homomorphism $f \in \mathcal{F}$ such that $h(u) = f(u)$ and $h(u) = f(v)$. Since f is a homomorphism, we obtain that $h(u)h(v) \in E(H)$.

If (ii) holds, then $v \in \eta(xy)$ for some $xy \in E(D)$. Note that by the definition of an extended strip decomposition we either have that $v \in \eta(xy, x)$ and $u \in \eta(x)$ for some $xy \in E(D)$ (note that the case $v \in \eta(xy, y)$ and $u \in \eta(y)$ is symmetric), or $v \in \eta(xy, x) \cap \eta(xy, y)$ and $u \in \eta(xyz)$ for some $xyz \in T(D)$.

Consider the first subcase and note that $v \in \text{potato}(x)$. If x is not covered by M, then we have $h(u)h(v) = f_x(u)\sigma^x(v)$. Recall that the definition of f_x, implies that $f_x(u) \in L_{\sigma^x}(u)$. Therefore, by Claim 7.2.14.2 (for $f = \sigma^x$ and $g = f_x$), we have $f_x(u)\sigma^x(v) \in E(H)$. If $xv_e \in M$ then we have $h(u)h(v) = f^x_{xy}(u)f^x_{xy}(v) \in E(H)$ by (F4). If $e \in M$, then $h(u)h(v) = f^e_e(u)f^e_e(v) \in E(H)$ by (F5). Finally, if there exists $y' \in N_D(x) \setminus \{y\}$ such that $\{xy', xv_{xy'}\} \cap M \neq \emptyset$, let $e' = xy'$. Then again $h(v) = \sigma^x(v)$ and $h(u) = f^x_{e'}(u)$ or $h(u) = f^{e'}_{e'}(u)$. Thus $h(u) \in L_g(u)$, where g is defined, respectively, as in (S2) or (S3), and $h(u)\sigma^x(v) = h(u)g(v) \in E(H)$ by Claim 7.2.14.2.

Consider the second subcase, i.e., $v \in \eta(xy,x) \cap \eta(xy,y)$ and $u \in \eta(xyz)$. If $\{xy, yz, xz\} \cap M = \emptyset$ then by (F3) and/or (F4) we have that $h(v) = \sigma^x(v)$, and by Claim 7.2.14.1, $h(v) = \sigma^{xyz}(v)$. Thus by (F2) we have that $h(u)h(v) = f_{xyz}(u)\sigma^{xyz}(v)$. By definition of f_{xyz} we have that $f_{xyz}(u) \in L_{\sigma^{xyz}}(u)$, hence $f_{xyz}(u)\sigma^{xyz}(v) \in E(H)$ by Claim 7.2.14.2. If $e = xy \in M$, then we have $h(u)h(v) = f^{xy}_{xy}(u)f^{xy}_{xy}(v) \in E(H)$ by (F5). Last, assume that $\{yz, xz\} \cap M \neq \emptyset$, by symmetry, $yz \in M$. It implies that $h(v)h(u) = \sigma^x(v)f^{yz}_{yz}(u)$. Recall that $f^{yz}_{yz}(u) \in L_g(u)$, where g is defined as in (S3). In particular, $f^{yz}_{yz}(u)\sigma^{xyz}(v) \in E(H)$. By Claim 7.2.14.1 $\sigma^{xyz}(v) = \sigma^x(v)$, and thus $h(v)h(u) = f^{yz}_{yz}(u)\sigma^x(v) \in E(H)$. That concludes the proof of (ii).

It remains to show (iii), i.e., that if $u \in \eta(xy,y)$, $v \in \eta(yz,y)$, then $h(u)h(v) \in E(H)$. If none of xy, yv_{xy}, yz, yv_{yz} belongs to M, we have that $h(u)h(v) = \sigma^y(u)\sigma^y(v) \in E(H)$. Otherwise, since M is a matching, precisely one of $h(u) = \sigma^y(u), h(v) = \sigma^y(v)$ is true, by symmetry say $h(u) = \sigma^y(u)$. From Lemma 7.2.12.2 it follows that $h(u)\sigma^y(v) \in E(H)$. This gives the desired statement. ⌟

The number of local subinstances. To conclude the proof of the lemma, we note that for every $xy \in E(D)$ we solve at most 4 local subinstances (G', L') from $\mathbb{I}_{xy}$, and for every $o \in V(D) \cup T(D)$ we solve one instance when defining $\mathcal{V}$ and $\mathcal{T}$. As the number of particles in a rigid e.s.d. is polynomial in $|V(G)|$, so is the size of $\mathbb{I}$. By Lemma 7.2.9, the instance $(D', \mathfrak{w}, \mathcal{V}, k)$ can be solved in polynomial time. This concludes the proof. □

We now have all the necessary tools to prove Theorem 1.3.5 (a).

Proof of Theorem 1.3.5 (a). Let (G', L') be an instance of LHOM(H). Clearly, without loss of generality we can assume that G and H are connected, as otherwise we can solve the problem independently for each pair (G', H') of connected components G' of G and H' of H (trimming the lists appropriately). Moreover, we we assume that H is non-bi-arc, as otherwise the problem can be solved in polynomial time.

Our algorithm will be recursive, taking as an input an instance (G', L') of LHOM(H) and a rigid e.s.d. (D', η') of G'. Initially, we set $D' = (\{x\}, \emptyset)$ and $\eta'(x) = V(G')$. Let $n' = |V(G')|$.

We start by applying Lemma 7.2.4 to (G', L'). We either solve (G', L') or obtain an equivalent, consistent subinstance (G, L) of LHOM(H). Let $W = V(G') \setminus V(G)$ and $n = |V(G)|$. We can assume that n is sufficiently large, as otherwise we can now solve

the instance (G, L) in constant time by brute-force. By Lemma 7.2.7, we can obtain from (D', η'), in time polynomial in n', a rigid e.s.d. of G (note that n may be significantly smaller than n'). We keep calling it (D', η').

Now we distinguish three cases.

Case 1: G contains a vertex v of degree at least $\sqrt{n} \cdot 2^{|V(H)|}$. This implies that there exists a list $\widetilde{L} \subseteq V(H)$ assigned to at least $\ell := \sqrt{n}$ neighbors of v. By Observation 7.1.4 there exist $a \in L(v)$ and $b \in \widetilde{L}$ such that $ab \notin E(H)$. We branch on assigning a to v; either we remove a from $L(v)$ or color v with a and remove b from the lists of all neighbors of v. In other words, we define $L_1, L_2 : V(G) \to 2^{V(H)}$ as

$$L_1(u) = \begin{cases} \{a\} & \text{if } u = v, \\ L(u) \setminus \{b\} & \text{if } L(u) = \widetilde{L}, \\ L(u) & \text{otherwise,} \end{cases} \qquad L_2(u) = \begin{cases} L(u) \setminus \{a\} & \text{if } u = v, \\ L(u) & \text{otherwise,} \end{cases}$$

Then we call the algorithm recursively on instances (G, L_1) and (G, L_2) and return that (G, L) is a yes-instance if and only if at least one of (G, L_1) and (G, L_2) is a yes-instance.

If Case 1 does not hold, for every $Y \subseteq V(G)$ of size at most $6\sqrt{n} \cdot 2^{|V(H)|}$ we check whether Y is a 1/2-balanced separator in G.

Case 2: there exists $Y \subseteq V(G)$ of size at most $6\sqrt{n} \cdot 2^{|V(H)|}$ that is a 1/2-balanced separator. We do one step of the divide & conquer approach as follows. Let $V_1, \ldots, V_q$ be the sets of vertices of connected components of $G - Y$, each of size at most $n/2$. Recall that for every $g : Y \to V(H)$ that respects L we can define $L_g : V(G) \setminus Y \to 2^{V(H)}$ as

$$L_g(v) = \begin{cases} L(v) \cap \bigcap_{u \in N_G(v) \cap Y} N_H(g(u)) & \text{if } N_G(v) \cap Y \neq \emptyset, \\ L(v) & \text{otherwise.} \end{cases} \tag{7.1}$$

We call the algorithm recursively on instances $(G[V_i], L_g)$ for every $i \in [q]$, and return that (G, L) is a yes-instance if and only if there exists g such that $(G[V_i \cup Y], L_g)$ and is a yes-instance for every $i \in [q]$.

Case 3: $\Delta(G) \leqslant \sqrt{n} \cdot 2^{|V(H)|}$ and there is no $1/2$-balanced separator of size at most $6\sqrt{n} \cdot 2^{|V(H)|}$ in G. Then we proceed depending on whether (D', η') is a $3/4$-balanced decomposition of G or not.

Subcase 3.1: (D', η') is not a $3/4$-balanced decomposition of G. We start by applying Theorem 7.2.6 to G (recall that G is $S_{t,t,t}$-free as an induced subgraph of G') to obtain, in time polynomial in n, a set P consisting of $\mathcal{O}(\log n)$ vertices of G, and a $1/2$-balanced rigid e.s.d. (D, η) of $G - N[P]$. Let $X = N[P]$. Then, for each homomorphism $\rho : (G[X], L) \to H$ we call the algorithm recursively for the instance $(G - X, L_\rho)$ and (D, η), where L_ρ is defined similarly as L_g in (7.1). We return that (G, L) is a yes-instance if and only if there exists ρ such that $(G - X, L_\rho)$ is a yes-instance.

Subcase 3.2: (D', η') is a $3/4$-balanced decomposition of G. We apply Lemma 7.2.8 to (D', η') to obtain a nice e.s.d. (D, η) of G. Then, we solve (G, L) by Lemma 7.2.14, which requires calling the algorithm recursively on $n^{\mathcal{O}(1)}$ local subinstances (G'', L'') of G.

That concludes the description of the algorithm, its correctness is straightforward to verify in cases 1, 2 and 3.2, and in case 3.2 is a consequence of Lemma 7.2.14.

Running time. It remains to analyze the running time. We measure the running time in terms of $||(G, L)|| = \sum_{v \in V(G)} |L(v)| - 1$. We note that the consistency of (G, L) guarantees that removing $p \geqslant 1$ vertices from G decreases the measure by at least p, as each $L(v)$ has at least two elements.

If the condition in Case 1 holds, the processing of the instance takes time polynomial in n, and then we call the algorithm on two instances of measure bounded by, respectively, $||(G, L)|| - \sqrt{n}$ (as we remove at least $\sqrt{n}$ elements of the lists) and $||(G, L)|| - 1$. We call these subinstances type (1a) and (1b), respectively.

If the condition in Case 1 does not hold, we exhaustively search for a $1/2$-balanced separator of size $6\sqrt{n} \cdot 2^{|V(H)|}$. This can be done in time $n^{\mathcal{O}(\sqrt{n})}$. Then, in Case 2, we call the algorithm on $2^{\mathcal{O}(\sqrt{n})}$ instances, each of measure at most $||(G, L)|| - n/2$. We call these subinstances type (2).

Otherwise, in Case 3, we check whether the given e.s.d. (D', η') is $3/4$-balanced. If no, we find the set X and a new rigid e.s.d. in time polynomial in n (by Theorem 7.2.6). Then, we recurse on at most $2^{\mathcal{O}(\sqrt{n} \log n)}$ instances that correspond to the choices of ρ :

$X \to V(H)$. Each such instance is of measure at most $||(G,L)||$. These are subinstances of type (3).

If (D', η') is 3/4-balanced, we run the polynomial-time algorithm from Lemma 7.2.8 to obtain a nice e.s.d., and solve the instance by calling the algorithm recursively on $n^{\mathcal{O}(1)}$ instances, each of measure at most $||(G,L)|| - n/4$. These are subinstances of type (4).

We estimate the number of calls of the algorithm as follows. Let $\mathcal{T}$ be the recursion tree of the calls of the algorithm for the instance (G,L). The nodes of $\mathcal{T}$ correspond to calls at subinstances (G',L') of (G,L). Similarly as in the proof of Theorem 7.1.3, for a node of $\mathcal{T}$ that corresponds to a call at instance (G',L') we define a *local subtree* that consists of all descendant calls where the measure of the corresponding instance (G'',L'') is at least $0.99||(G',L')||$. We greedily find a partition Π of nodes of $\mathcal{T}$ into local subtrees. Clearly, each path from the root to a leaf of $\mathcal{T}$ intersects $\mathcal{O}(\log n)$ elements of Π.

Consider a local subtree $\mathcal{T}'$, whose root corresponds to a call at an instance (G',L') with n' vertices. We claim that $\mathcal{T}'$ has $2^{\mathcal{O}(\sqrt{n'}\log n')}$ leaves. First, we note that (i) the nodes of $\mathcal{T}'$ that correspond to the subinstances of type (2) and (4) can occur only as the leaves of $\mathcal{T}'$, since each child of such node corresponds to an instance whose measure is at most 3/4 of the parent's measure. We also note that (ii) there is at most one node of type (3) for every leaf-to-root path in $\mathcal{T}'$. Indeed, assume that there exist two nodes τ_1 and τ_2 in $\mathcal{T}'$ corresponding to, respectively, instances (G_1,L_1) and (G_2,L_2) of type (3), such that τ_2 is the highest descendant of τ_1 with this property. Let $n_1' = |V(G_1)|$ and $n_2' = |V(G_2)|$. In τ_1 we compute an e.s.d. (D_1,η_1) whose each particle is of size at most $n_1'/2$. It follows from the description of the algorithm that in the call τ_2 we are given a decomposition (D_1',η_1') that is obtained from (D_1,η_1) by applying Lemma 7.2.7 and Lemma 7.2.8. Since τ_2 is of type (3), (D_1',η_1') is not 3/4-balanced, i.e., there exists a particle of size larger than $3n_2'/4$. Note that the only way in which this could happen is that $n_2' \leqslant 2n_1'/3$, which is a contradiction with the definition of a local subtree $\mathcal{T}'$, as $0.99n_1' \leqslant n_2' \leqslant 2n_1'/3$.

Let $\mathcal{T}''$ be obtained by contracting all edges leading to a node of type (1b). The type of the resulting new node is inherited from the parent of the node of type (1b). Thus, all the internal nodes of $\mathcal{T}''$ are either of type (1a) or (3), and the number of their children in $\mathcal{T}''$ is bounded, respectively, by $\mathcal{O}(n')$ and $(n')^{\mathcal{O}(\sqrt{n'})}$. Moreover, the number of leaves of $\mathcal{T}''$ is at least half of the number of leaves of $\mathcal{T}'$. Therefore, the maximum length of a leaf-to-root path P in $\mathcal{T}''$ is $\mathcal{O}(\sqrt{n'})$, since at most three nodes of P (the root, the leaf

and one node of type (3)) are not of type (1a). We conclude that the number of leaves of $\mathcal{T}''$ (and thus of $\mathcal{T}'$) is also $2^{\mathcal{O}(\sqrt{n'}\log n')}$.

Now we observe that $\mathcal{T}'$ has at most $2^{\mathcal{O}(\sqrt{n'}\log n')}$ nodes. Indeed, note that the only possibility that a node τ of $\mathcal{T}'$ is of degree 2 is when τ is of type (3) (this corresponds to the situation when the set $\mathcal{P}$, given by Theorem 7.2.6, is empty). However, by (ii), we know that the number of such nodes in $\mathcal{T}'$ is at most the number of leaves of $\mathcal{T}'$. As the number of leaves is bounded by $2^{\mathcal{O}(\sqrt{n'}\log n')}$, so is the total number of nodes of $\mathcal{T}'$.

Last, consider a tree $\mathcal{T}'''$ obtained from $\mathcal{T}$ by contracting each local subtree $\mathcal{T}'$ to one vertex. As each vertex of $\mathcal{T}$ has at most $2^{\mathcal{O}(\sqrt{n}\log n)}$ children, each internal node of $\mathcal{T}'''$ has at most $2^{\mathcal{O}(\sqrt{n}\log n)} \cdot 2^{\mathcal{O}(\sqrt{n}\log n)} = 2^{\mathcal{O}(\sqrt{n}\log n)}$ children. As each leaf-to-root path in $\mathcal{T}'''$ is of length at most $\mathcal{O}(\log n)$, the total number of nodes in $\mathcal{T}'''$ is at most $2^{\mathcal{O}(\sqrt{n}\log^2 n)}$. Since each local subtree has $2^{\mathcal{O}(\sqrt{n'}\log n')}$ nodes, we obtain that the total number of nodes of $\mathcal{T}$ is $2^{\mathcal{O}(\sqrt{n}\log^2 n)}$.

This concludes the proof of Theorem 1.3.5 (a). □

7.2.5 Lower bounds and generalizations

Now, we complete the dichotomy stated in Theorem 1.3.5 by showing that if an undecomposable graph H is not safe, then there exists $t \in \mathbb{N}$ such that the LHom(H) problem is NP-complete and there is no subexponential-time algorithm that solves it in $S_{t,t,t}$-free graphs unless the ETH fails.

Proof of Theorem 1.3.5 (b). If H contains a predator, then the statement follows from Theorem 1.3.4 (b), as P_t-free graphs are a subclass of $S_{t,t,t}$-free graphs. Hence, assume that there exists a trap $(\tau_1, \tau_2, \tau_3) = ((a_1, b_1, c_1), (a_2, b_2, c_2), (a_3, b_3, c_3))$ in H.

Auxiliary gadgets. We start by defining a ternary relation $T = \{\tau_1, \tau_2\}$ on vertices of H. By Theorem 6.3.2, there exists a list-T-gadget $(F_T, L, (x_1, x_2, x_3))$. Define the graph F'_T to be the graph F_T with edges x_1x_2, x_1x_3, x_2x_3 added (if they do not already exist) and observe that since $a_1b_1c_1$ and $a_2b_2c_2$ are triangles, $(F'_T, L, (x_1, x_2, x_3))$ is still a list-T-gadget. Next, define a binary relation $O = \{(a_1, c_3), (a_2, a_3), (a_2, b_3)\}$. Let $(F_O, L, (y_1, y_2))$ be the list-O-gadget constructed by Theorem 6.3.2.

We reduce from Pos-1-in-3-Sat (see the appendix for details). In Pos-1-in-3-Sat, for a given set of variables $U = \{u_1, \ldots, u_N\}$ and clauses $\mathcal{C} = \{C_1, \ldots, C_M\}$, such that

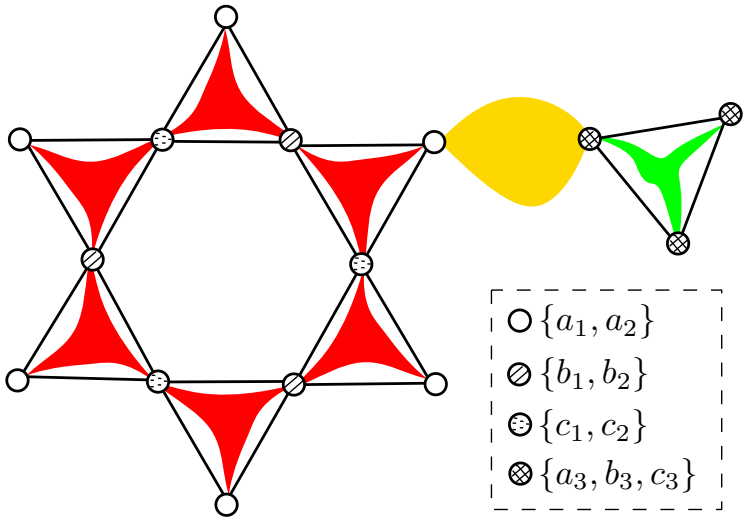


Figure 7.6: An example of a variable gadget, an occurrence gadget and a clause gadget, and how they are connected with each other. The vertices with the same lists are marked by the same vertex pattern. The list-T-gadget $(F_T, L, (x_1, x_2, x_3))$, depicted in red, ensures that its interface vertices are mapped either to $a_1, b_1, c1$ or to a_2, b_2, c_2. The occurrence gadget $(F^O, L, (y_1, y_2))$, depicted in orange, guarantees that its interface vertices are mapped either to a_1, a_3, a_2, b_3 or a_2, c_3. The list-W-gadget $(F^W, L, (z_1, z_2, z_3))$ depicted in green, guarantees that one of its interface vertices is mapped to a_3, one to b_3 and one for c_3 (recall that we may have $b_3 = c_3$. The black edges are extra edges added to the graph G to make it $S_{t,t,t}$-free.

every clause contains at most 3 variables and all of them are non-negated, we ask if there exists a truth assignment $f : U \to \{0, 1\}$ such that for every clause $C \in \mathcal{C}$ we have that precisely one literal is mapped to 1.

We define an instance (G, L) of LHOM(H) as follows. For every $u_i \in U$ that occurs k_i times in clauses in $\mathcal{C}$, consider a k_i-ary relation $A_i = \{(a_1, \ldots, a_1), (a_2, \ldots, a_2)\}$. We construct an list-$A_i$-gadget $(Q_i, L, (x'_1, \ldots, x'_{k_i}))$ (which we call a *variable gadget*) as follows. We introduce k_i copies of $(F'_T, L, (x_1, x_2, x_3))$, denote the j-th copy by $(F'^j_T, L^j, (x^j_1, x^j_2, x^j_3))$. For every even j we identify the vertex x^j_1 with x^{j-i}_1, and the vertex x^j_2 with x^{j+i}_2. Equivalently, for every odd j we identify x^j_2 with x^{j-i}_2, and x^j_1 with x^{j+i}_1 (see Figure 7.6). We define $(x'_1, \ldots, x'_{k_i})$ to be $(x^1_3, \ldots, x^{k_i}_3)$ and note that all these vertices must be mapped either to a_1 or to a_2. This will correspond, respectively, to setting u_i to true or to false.

To create a *clause gadget*, we recall that $\{a_3, b_3, c_3\}$ is a non-trivial triangle, i.e., either $a_3b_3c_3$ is a simple triangle, which implies that all 2-subsets of $\{a_3, b_3, c_3\}$ are incomparable, or $a_3 = b_3$ and $a_3b_3, a_3c_3 \in E(H)$. Define $W = \{(a_3, b_3, c_3), (b_3, c_3, a_3), (c_3, a_3, b_3)\}$. By Theorem 6.3.2, we obtain a list-W-gadget $(F^W, L, (z_1, z_2, z_3))$. Again, modify F^W to obtain $F^{W'}$ by adding edges z_1z_2, z_2z_3, z_3z_1. For every $C_j \in \mathcal{C}$ we introduce W-gadget

$(F^{W'}{}_j, L, (z_1^j, z_2^j, z_3^j))$.

We connect variable gadgets with clause gadgets using the *occurrence gadgets*. If the ℓ-th occurrence of the variable u_i appears in the clause C_j, we introduce the O-gadget $(F^O, L, (y_1, y_2))$ and identify y_1 and y_2, respectively, with $x'_\ell \in V(Q_i)$, and with one of the vertices z_1^j, z_2^j, z_3^j of $V(F^{W'}{}_j)$. That concludes the construction.

It is straightforward to verify that the properties of the gadgets ensure that there exists a truth assignment $f : U \to \{0, 1\}$ that satisfies all the clauses if and only if there exists a homomorphism $h : (G, L) \to H$. Moreover, let t' be the maximum integer such that there exists induced $P_{t',t',t'}$ path in some of the gadgets introduced by Theorem 6.3.2, and let $t = t' + 2$.

We claim that G is $S_{t,t,t}$-free. Assume for contradiction that there G contains an induced $S_{t,t,t}$. Let v_c the its vertex of degree 3. First, note that no interface vertex belongs to more than two gadgets. If v_c is an interface vertex of a variable (resp. clause) gadget Q_i and some other gadget F^O, then, without loss of generality, one of the other interface vertices of Q_i must be a neighbor of v_c in $S_{t,t,t}$. However, in this case Q_i contains an induced path P_{t-1} (that in particular does not use interface vertices of Q_i), a contradiction with the definition of t. On the other hand, if v_c is in a gadget F but is not an interface vertex of it, each of the components of $S_{t,t,t} - v_c$ must contain an interface vertex of F. This is also a contradiction, since either there are only two distinct interface vertices of F (in F is an O-gadget) or they induce a triangle (in all other cases). We obtain that G is $S_{t,t,t}$-free. This concludes the proof. □

Decomposable target graphs. We note that Theorem 1.3.5 (b) works for any target graph H that contains a non-safe graph H' as an induced subgraph—as the instance produced by Theorem 1.3.5 (b) for LHOM(H') is also an instance of LHOM(H). One could try to extend the definition of safe graphs to decomposable graphs that does not contain non-safe undecomposable induced subgraphs, and try to show that for every such graph H there is an algorithm solving LHOM(H) in subexponential time in $S_{t.t.t}$ graphs for every $t \in \mathbb{N}$.

However, the problem seems to be more complicated, and unfortunately, the above definition does not seem to be the correct one. Note that the graph H depicted on Figure 7.7 (left) does not contain a non-safe undecomposable induced subgraph, but still, it can be shown that there is $t \in \mathbb{N}$ such that LHOM(H) is NP-complete in $S_{t,t,t}$-free

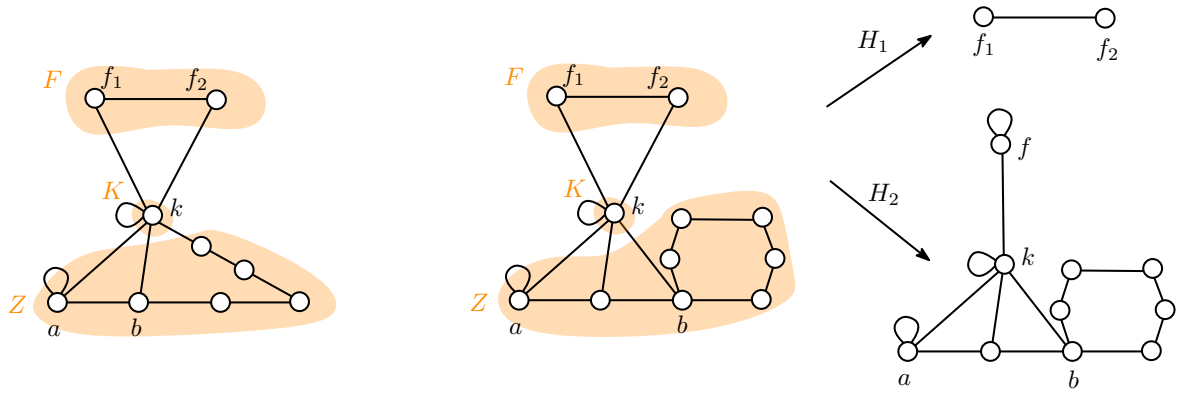


Figure 7.7: An example of a graph H that does not contain a non-safe indecomposable induced subgraph (left). An example of a graph H and its factor H_2, such that H does not contain a trap, but $((f, f, f), (a, a, a), (k, k, d))$ is a trap in H_2 (right).

graphs and there is no subexponential-time algorithm to solve it unless the ETH fails.

On the other hand, the graph H depicted on Figure 7.7 (right) does not even contain a trap, but it can be decomposed in a way that a trap appears in one if its factors. This, in turn, suggests that solving $\text{LHOM}(H')$ for some $H' \in \mathcal{H}$ may actually be harder than solving $\text{LHOM}(H)$, so the idea that it is enough to restrict to undecomposable target graphs, that worked for predacious graphs, may not work here either. Indeed, if H does not have a predator, it cannot appear in H_1 or H_2.

7.3 Possible extensions of the results

Dependence of the target graph and the forbidden subgraph Observe that the forbidden induced subgraph in instances constructed in Theorem 1.3.4 (b) depends on the target graph H. One might ask if this is necessary—perhaps we could improve the construction to make the graph P_t-free, where t does not depend on H, as it was the case in Theorem 7.1.6. The following example shows that in some cases this is impossible.

Proposition 7.3.1. *For every $t \geqslant 1$ there exists a bipartite graph H_t and an integer $t' > t$, such that:*

1. *$\text{LHOM}(H_t)$ can be solved in polynomial time in P_t-free graphs,*
2. *$\text{LHOM}(H_t)$ is NP-complete in P_t-free graphs and cannot be solved in subexponential time, unless the ETH fails.*

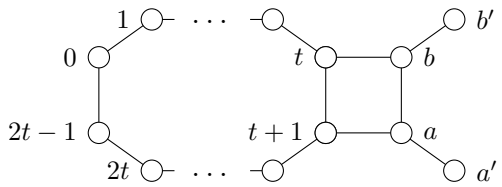


Figure 7.8: The graph H_t from Proposition 7.3.1.

Proof. We can safely assume that $t \geqslant 3$, as P_2-free graphs have no edges and thus LHOM(H) is trivial on these graphs. Let H_t be constructed as follows. We start with an even cycle with consecutive vertices $\{0, 1, \ldots, 2t-1\}$. Then we add vertices a, a', b, b', and edges $(t+1)a$, tb, ab, aa', and bb' (see Figure 7.8). Observe that vertices $0, 1, \ldots, 2t-1$ induce a cycle of length at least 6, so the complement of H_k is not a circular-arc graph (recall Theorem 5.1.6 and the notion of obstruction). On the other hand, $(t, a, t+1, b)$ is a predator. Finally, one can readily verify that H_k is undecomposable, so the second statement follows from Theorem 7.1.7.

Now let us prove the first statement. Let (G, L) be an instance of LHOM(H) such that G is P_t-free. If G is not connected, we can solve the problem separately for each connected component. Thus assume G is connected. The crucial observation is that in any homomorphism $h : G \to H_t$, either $h^{-1}(\{0, 1\}) = \emptyset$, or $h^{-1}(\{t, t+1, a, b, a', b'\}) = \emptyset$. Indeed, suppose that there exists $h : G \to H_t$, and vertices u, v of G, such that $h(u) \in \{0, 1\}$, and $h(v) \in \{t, t+1, a, b, a', b'\}$. Let Q be an induced u-v-path in G, it exists as G is connected. As G is P_t-free, we know that Q has at most $t-1$ vertices. Now observe that the colors of consecutive vertices of Q form an $h(u)$-$h(v)$-walk in H_t. However, a shortest walk in H_t, starting in $\{0, 1\}$ and terminating in $\{t, t+1, a, b, a', b'\}$ has t vertices, a contradiction.

Thus given an instance (G, L) of LHOM(H_t), where G is P_t-free, we can reduce it to solving instances of LHOM($H_t - \{0, 1\}$) and instances of LHOM($H_t - \{t, t+1, a, b, a', b'\}$) independently. One can verify that each of these two target graphs is a complement of a circular-arc graph (as none contains an obstruction), so each of the instances can be solved in polynomial time. □

Note that Proposition 7.3.1 implies an analogous result for $S_{t,t,t}$-free graphs in case when H is predacious.

The Hom(H) problem in F-free graphs. Here we briefly discuss the consequences of the theorems proven in this chapter for the non-list variant of the homomorphism problem. We start by recalling that in case of the Hom(H) problem, we can restrict our considerations to connected non-trivial core graphs H for which the Hom(H) problem is NP-complete, i.e., non-bipartite and irreflexive. This in particular means that H is undecomposable, i.e., H does not admit any of the decompositions introduced in Chapter 5. Indeed, observe that in each case the existence of such decomposition implies that there exist two vertices $u, v \in V(H)$ such that $N(u) \subseteq N(v)$. If H is a core, this is however a contradiction with Observation 3.1.1.

Recall that focusing the LHom(H) problem in F-free graphs where F is a path or a subdivided claws was motivated by the result of Piecyk and Rzążewski [101, 102] that for all other connected graphs F the problem is NP-complete and cannot be solved in subexponential time unless the ETH fails. We point out that a general classification of the Hom(H) problem in these other F-free graphs is still open and seem to be more challenging than the list version (for partial results see [16, 28, 110]), here, however, we are going to focus on our work for P_t-free and $S_{t,t,t}$-free graphs.

P_t-free graphs. We first consider the case when F is an induced path on t vertices. Since we can assume that H is non-trivial, connected, irreflexive and undecomposable, this simplifies the definition of predacious graphs. We say that a 4-tuple (a_1, b_1, a_2, b_2) of vertices of H is an *incomparable* C_4 if (a_1, b_1, a_2, b_2) are consecutive vertices of a (not necessarily induced) cycle C_4 such that $\{a_1, a_2\}$ and $\{b_1, b_2\}$ are incomparable sets. It is straightforward to observe that a non-trivial core H is predacious if H contains an incomparable C_4. We also note that if H is a core, then, since H is ramified, any C_4 is an incomparable C_4. We call a graph *square-free* if it does not contain C_4 as a subgraph.

It follows that we can restate Theorem 1.3.4 (a) as follows.

Theorem 7.3.2. *Let H be a square-free core. Then, for every $t \in \mathbb{N}$, the Hom(H) problem can be solved in time $n^{\mathcal{O}(\log^2 n)}$ in n-vertex P_t-free graphs.*

Consider the graph H_B depicted on Figure 7.9 (left), called the Brinkman graph. It can be verified by exhaustive computer search that $K_3 \times H_B$ is a core. Clearly, both K_3 and H_B are non-predacious, as none of them contains C_4 as a subgraph. By combining Theorem 7.3.2 with Observation 3.2.2 (5) we can solve Hom(H) in time $n^{\mathcal{O}(\log^2 n)}$

in n-vertex P_t-free graphs However, as shown on Figure 7.9 (right), $K_3 \times H_B$ contains a predator.

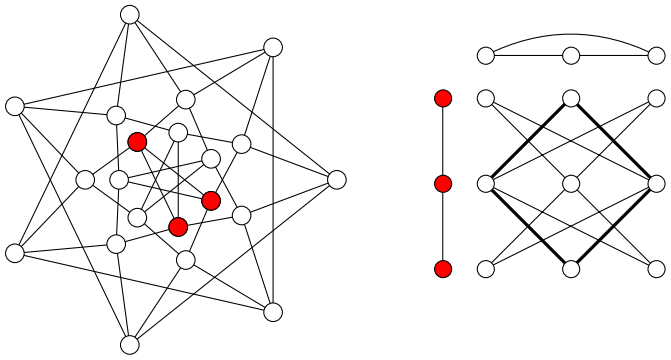


Figure 7.9: The Brinkmann graph H_B (left) with a subgraph P_3 distinguished in red, and illustration that the graph $H_B \times K_3$ contains a C_4 as a subgraph (right).

This observation can be generalized as follows.

Corollary 7.3.3. *Let H be a non-trivial core with factorization $(H_1, \ldots, H_m)$. If for every $i \in [m]$ the graph H_i is square-free, then, for every $t \in \mathbb{N}$, the* Hom*(H) problem can be solved in time $n^{\mathcal{O}(\log^2 n)}$ in n-vertex P_t-free graphs.* □

If it comes to lower bounds the situation becomes more complicated. Although we are not able to deliver a full dichotomy for the Hom(H) problem in P_t-free graphs, we are still able to provide certain lower bounds. First, we note that combining some results that we proved before we can show the lower bound when the target graph is additionally assumed to be projective.

Theorem 7.3.4. *Let H be a non-trivial, projective graph that contains C_4 as a subgraph. Then there exists $t \in \mathbb{N}$, such that the* Hom*(H) problem cannot be solved in time $2^{o(n)}$ in n-vertex P_t-free graphs, unless the ETH fails.*

Proof. We reduce from the LHom(H) problem in P_t-free graphs. Let (G, L) be an n-vertex instance of LHom(H) such that G is P_t-free. Since H is projective, Corollary 3.2.6 asserts that for each $v \in V(G)$ there exists an $L(v)$-construction (F_v, φ_v, x_v).

We perform the reduction in two steps. First, we define an instance (G', φ) of ExtHom(H) as follows. We take the graph G and, for every $v \in V(G)$, we introduce the $L(v)$-construction (F_v, φ_v, x_v). Then, for every $v \in V(G)$ we identify v with x_v. This way we obtain the graph G', the function φ arises from taking the union of all the functions φ_v over all $v \in V(G)$. We note that as G is P_t-free, it follows from the definition of G'

that the longest path in G' has at most $t' = 2\max_{v \in V(G)} |V(F_v)| + t$ vertices. Indeed, note that the way we build G' guarantees that for every $v \in V(G)$ the vertices of $V(F_v) \setminus \{x_v\}$ occur consecutively in any path in G'. Moreover, if such a path P contains vertices from $V(F_v) \setminus \{x_v\}$, at least one endpoint of P belongs to $V(F_v) \setminus \{x_v\}$. Thus, an induced path in G' contains vertices from at most two such sets, and at most $t-1$ vertices from G. Since the size of each F_v depends only on $|V(H)|$, which is a constant, t' is also a constant.

Now, having an instance (G', φ) of ExtHom(H) such that G' is $P_{t'+1}$-free, we use Theorem 3.2.9 to transform it into an instance G'' of Hom(H), such that G' is an induced subgraph of G'' and $|V(G'')| = |V(G')| + |V(H)|$. We claim that there exists $t'' \in \mathbb{N}$ such that G'' is $P_{t''}$-free. Consider a longest induced path P in G''. Clearly, in contains at most $|V(H)|$ vertices that do not belong to $V(G')$. Since G' is $P_{t'+1}$ free, there are at most t' consecutive vertices of $V(G')$ that appear on P. Thus P has at most $(|V(H)|+1) \cdot t'$ vertices, and G'' is $P_{t''}$-free for $t'' = (|V(H)|+1) \cdot t' + 1$.

Thus, if we can solve Hom(H) in time $2^{o(|V(G'')|)} = 2^{o(n)}$ in $P_{t''}$-free graphs, we can solve LHom(H) in P_t-free graphs in time $2^{o(n)}$, and this, by Theorem 1.3.4, contradicts the ETH. □

Looking from the broader perspective, we can provide a lower bound for the general case if we additionally assume Conjecture 1 and Conjecture 2. Following these premises, we conjecture that the lower bound for the Hom(H) problem needs to take into consideration the factorization (w.r.t. the direct product) of the target graph H.

Conjecture 3. *Let H be a non-trivial graph with factorization $(H_1, \ldots, H_m)$. If there exists $i \in [m]$ such that H_i is not square-free, then there exists $t \in \mathbb{N}$ such that the Hom(H) problem in NP-complete and cannot be solved in time $2^{o(n)}$ in n-vertex P_t-free graphs, unless the ETH fails.*

$S_{t,t,t}$-free graphs. Next, we turn our attention to $S_{t,t,t}$-free graphs. Again, since we can assume that H is non-trivial, connected, irreflexive and undecomposable, this simplifies the definition of safe graphs: these are precisely graphs that are triangle-free and square-free, or, equivalently, with girth at least 5. Analogously as in the P_t-free graphs case, a straightforward consequence of Theorem 1.3.5 (a) and Observation 3.2.2 (5) is the following.

Theorem 7.3.5. *Let H be a core with factorization $(H_1, \ldots, H_m)$. If for every $i \in [m]$ graph H_i is of girth at least 5, then, for every $t \in \mathbb{N}$, the* Hom*(H) problem can be solved in time $2^{\mathcal{O}(\sqrt{n}\log^2 n)}$ in n-vertex $S_{t,t,t}$-free graphs.*

Now, if H admits a factorization $(H_1, \ldots, H_m)$ and there is $i \in [m]$ such that H_i contains C_4 as a subgraph, then H satisfies the assumptions of Conjecture 3. Moreover, the smallest core that is square-free but have girth smaller than 5 is K_3; recall that it is known that 3-Coloring in $S_{1,1,1}$-free graphs is NP-complete and cannot be solved in subexponential time unless the ETH fails [70,86]. This motivates the following conjecture.

Conjecture 4. *Let H be a non-trivial graph with factorization $(H_1, \ldots, H_m)$. If there exists $i \in [m]$ such that H_i has girth at most 4, then there exists $t \in \mathbb{N}$ such that the* Hom*(H) problem is NP-complete cannot be solved in time $2^{o(n)}$ in n-vertex $S_{t,t,t}$-free graphs, unless the ETH fails.*

Chapter 8

Related results

In the last chapter we briefly survey the other results that are related to the graph homomorphism problem and co-authored by the author of the dissertation. Two of these [56,90] were already discussed in Chapter 7, here we focus on the remaining works.

Other graph classes. Another natural family of hereditary graph classes that one can investigate to find subexponential-time algorithms for various graph problems is formed by *geometric intersection graphs* [8, 27, 44]. These include in particular interval graphs, circular-arc graphs (already considered in Chapter 5), disk graphs or string graphs.

In [98], we show a dichotomy for a more general *weighted* homomorphism problem WHOM(H) (considered also in [56]) for string graphs, i.e., the intersection graphs of continuous curves in the plane. The dichotomy states that (i) the WHOM(H) problem in string graphs can be solved in subexponential time if H does not have two vertices with two common neighbors, and (ii) that the problem is NP-complete and cannot be solved in subexponential time, assuming the ETH, otherwise. Recall that the algorithmic part of the theorem coincides with our result from [56], that if H does not have two vertices with two common neighbors, WHOM(H) can be solved in subexponential time in P_t-free graphs for all $t \in \mathbb{N}$. In [98] we also complete the dichotomy for P_t-free graphs, showing the lower bounds for the remaining cases of H.

In [81] we focus on the LHOM(H) problem, and investigate in which classes of geometric intersection graphs it can be solved in subexponential time. We provide a series of results—both, algorithmic and lower bounds—for various intersection classes of so called *fat* objects. It turns out that the property of the graph H that seems to allow the existence of faster algorithms in these classes is the size of a maximum reflexive clique

in H. In particular, if the target graph H is irreflexive, the subexponential algorithms solving the homomorphism problem exist for all classes of graphs we consider (and thus $\textsc{Hom}(H)$ can be always solved in subexponential time in these classes). We also study further the class of string graphs. It turns out that for $\textsc{LHom}(H)$ the complexity of the problem in string graphs also coincides with its complexity in P_t-free graphs (namely, Theorem 1.3.4). Indeed, we show that if H is non-predacious, then $\textsc{LHom}(H)$ can be solved in time $2^{\mathcal{O}(n^{2/3}\log n)}$ in n-vertex string graphs, and otherwise, there is no subexponential algorithm for $\textsc{LHom}(H)$ in string graphs, assuming the ETH.

Other types of restrictions. As already discussed in the introduction, the major challenge with studying the $\textsc{Hom}(H)$ and $\textsc{LHom}(H)$ problems is the rich structure of target graphs that is difficult to capture by combinatorial tools. Thus, another popular line of research is focusing on particular families of target graphs, like odd cycles [16, 79] or Kneser graphs [116]. This way we can exploit the symmetric structure of the target.

In [28] we investigate the (L)$\textsc{Hom}(H)$ problem in F-free graphs for targets H that are *k-wheels* W_k, i.e., targets obtained from a k-cycle by adding a universal vertex. Among other results, we discover an interesting non-monotonicity of the $\textsc{Hom}(H)$ problem in restricted graph classes. Indeed, we show that $\textsc{Hom}(W_5)$ is polynomial-time solvable in $K_{1,3}$-free graphs. On the other hand note, W_5 has K_3 as a subgraph, and $\textsc{Hom}(K_3)$ in $K_{1,3}$-free graphs is NP-comlete and cannot be solved in time $2^{o(n)}$. This kind of behavior does not occur if we consider the $\textsc{Hom}(H)$ problem in general graphs: if H' contains H as a subgraph, and $\textsc{Hom}(H)$ is NP-complete, so is $\textsc{Hom}(H')$. This also cannot happen in the list variant, even if we restrict the class of instances, as each instance of $\textsc{LHom}(H)$ is also an instance of $\textsc{LHom}(H')$.

Other ways of measuring complexity. While analyzing the running time of the algorithms is one of the most prominent ways to study the complexity of particular problems, it is not the only one we can investigate. In [15] we show that the tools that we developed to study the complexity of $\textsc{LHom}(H)$ problem—in this particular case some of the gadgets that we used for the lower bounds in Chapter 6—can be also applied to study the *sparsification* algorithms for $\textsc{LHom}(H)$. It turns out that there is no non-bi-arc graph H such that every n-vertex instance of $\textsc{LHom}(H)$ can be in polynomial time reduced to an equivalent instance of bitsize $\mathcal{O}(n^{2-\varepsilon})$ for some $\varepsilon > 0$ unless an unlikely complexity-theoretic collapse occurs.

Appendix: Variants of satisfiability problems

A *formula* $(X, \mathcal{C})$ consists of a set $X = \{x_1, \ldots, x_n\}$ of variables, and a set $\mathcal{C}$ of clauses, where each clause contains *literals* of X, i.e., variables and their negations (called, respectively, positive and negative literals). A *truth assignment* for X is a function $\gamma : X \to \{0, 1\}$. A truth assignment γ *satisfies* a positive (resp. negative) literal z if $\gamma(z) = 1$ (resp. $\gamma(z) = 0$). Below we define the SAT problems that are considered in the thesis.

k-SAT, $k \geqslant 3$ **Input:** A formula $(X, \mathcal{C})$ where each clause has at most k elements. **Question:** Does there exist a truth assignment for X that satisfies at least one literal in each clause?

POS-1-IN-3-SAT **Input:** A pair $(X, \mathcal{C})$ where each clause contains precisely 3 positive literals **Question:** Does there exist a truth assignment for X that satisfies exactly one literal in each clause?

The first one is a classic NP-complete problem [49], and the subjects of, respectively, the SETH and the ETH. The 3-SAT and POS-1-IN-3-SAT problems are used as the starting points of our reductions. The following theorem can be found e.g. in [29].

Theorem 8.0.1 ([29], Theorem 24). *The 1-IN-3-SAT problem in NP-complete, and it cannot be solved in time $2^{o(n+m)}$, where n and m are, resp., the number of variables and clauses of an instance, unless the ETH fails.*

Bibliography

[1] Kenneth .I Appel and Wolfgang Haken. *Every planar map is four colorable*, volume 98. American Mathematical Soc., 1989.

[2] Stefan Arnborg, Derek G. Corneil, and Andrzej Proskurowski. Complexity of finding embeddings in a k-tree. *SIAM J. Algebraic Discrete Methods*, 8(2):277–284, April 1987.

[3] Michael Atiyah. *Introduction to commutative algebra*. CRC Press, 2018.

[4] Philippe Baptiste, Philippe Laborie, Claude Le Pape, and Wim Nuijten. Constraint-based scheduling and planning. In Francesca Rossi, Peter van Beek, and Toby Walsh, editors, *Handbook of Constraint Programming*, volume 2 of *Foundations of Artificial Intelligence*, pages 761–799. Elsevier, 2006.

[5] John Lane Bell and Alan B. Slomson. *Models and ultraproducts: An introduction*. Courier Corporation, 2006.

[6] Andreas Björklund, Thore Husfeldt, Petteri Kaski, and Mikko Koivisto. Fourier meets Möbius: fast subset convolution. In David S. Johnson and Uriel Feige, editors, *Proceedings of the 39th Annual ACM Symposium on Theory of Computing, San Diego, California, USA, June 11-13, 2007*, pages 67–74. ACM, 2007.

[7] Hans L. Bodlaender. Dynamic programming on graphs with bounded treewidth. In Timo Lepistö and Arto Salomaa, editors, *Automata, Languages and Programming, 15th International Colloquium, ICALP88, Tampere, Finland, July 11-15, 1988, Proceedings*, volume 317 of *Lecture Notes in Computer Science*, pages 105–118. Springer, 1988.

[8] Marthe Bonamy, Édouard Bonnet, Nicolas Bousquet, Pierre Charbit, Panos Giannopoulos, Eun Jung Kim, Paweł Rzążewski, Florian Sikora, and Stéphan Thomassé. EPTAS and subexponential algorithm for maximum clique on disk and unit ball graphs. *J. ACM*, 68(2):9:1–9:38, 2021.

[9] Édouard Bonnet, Colin Geniet, Eun Jung Kim, Stéphan Thomassé, and Rémi Watrigant. Twin-width III: Max Independent Set, Min Dominating Set, and Coloring. In Nikhil Bansal, Emanuela Merelli, and James Worrell, editors, *48th International Colloquium on Automata, Languages, and Programming, ICALP 2021, July 12-16, 2021, Glasgow, Scotland (Virtual Conference)*, volume 198 of *LIPIcs*, pages 35:1–35:20. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2021.

[10] Flavia Bonomo, Maria Chudnovsky, Peter Maceli, Oliver Schaudt, Maya Stein, and Mingxian Zhong. Three-coloring and list three-coloring of graphs without induced paths on seven vertices. *Comb.*, 38(4):779–801, 2018.

[11] Graham R. Brightwell and Peter Winkler. Graph homomorphisms and phase transitions. *J. Comb. Theory, Ser. B*, 77(2):221–262, 1999.

[12] Andrei A. Bulatov. Tractable conservative constraint satisfaction problems. In *18th IEEE Symposium on Logic in Computer Science (LICS 2003), 22-25 June 2003, Ottawa, Canada, Proceedings*, page 321. IEEE Computer Society, 2003.

[13] Andrei A. Bulatov. H-coloring dichotomy revisited. *Theor. Comput. Sci.*, 349(1):31–39, 2005.

[14] Andrei A. Bulatov. A short story of the CSP dichotomy conjecture. In *34th Annual ACM/IEEE Symposium on Logic in Computer Science, LICS 2019, Vancouver, BC, Canada, June 24-27, 2019*, page 1. IEEE, 2019.

[15] Hubie Chen, Bart M. P. Jansen, Karolina Okrasa, Astrid Pieterse, and Paweł Rzążewski. Sparsification lower bounds for list h-coloring. In Yixin Cao, Siu-Wing Cheng, and Minming Li, editors, *31st International Symposium on Algorithms and Computation, ISAAC 2020, December 14-18, 2020, Hong Kong, China (Virtual Conference)*, volume 181 of *LIPIcs*, pages 58:1–58:17. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2020.

[16] Maria Chudnovsky, Shenwei Huang, Paweł Rzążewski, Sophie Spirkl, and Mingxian Zhong. Complexity of c_k-coloring in hereditary classes of graphs. *Inf. Comput.*, 292:105015, 2023.

[17] Maria Chudnovsky, Marcin Pilipczuk, Michał Pilipczuk, and Stéphan Thomassé. Quasi-polynomial time approximation schemes for the maximum weight independent set problem in h-free graphs. *CoRR*, abs/1907.04585, 2019.

[18] Maria Chudnovsky, Marcin Pilipczuk, Michał Pilipczuk, and Stéphan Thomassé. Quasi-polynomial time approximation schemes for the maximum weight independent set problem in h-free graphs. In Shuchi Chawla, editor, *Proceedings of the 2020 ACM-SIAM Symposium on Discrete Algorithms, SODA 2020, Salt Lake City, UT, USA, January 5-8, 2020*, pages 2260–2278. SIAM, 2020.

[19] Maria Chudnovsky, Neil Robertson, Paul Seymour, and Robin Thomas. The strong perfect graph theorem. *Annals of mathematics*, pages 51–229, 2006.

[20] Maria Chudnovsky and Paul D. Seymour. The three-in-a-tree problem. *Comb.*, 30(4):387–417, 2010.

[21] Maria Chudnovsky, Sophie Spirkl, and Mingxian Zhong. Four-coloring p_t-free graphs. In *Proceedings of the Thirtieth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2019, San Diego, California, USA, January 6-9, 2019*, pages 1239–1256. SIAM, 2019.

[22] Paul Moritz Cohn. *Basic algebra: groups, rings and fields*. Springer Science & Business Media, 2012.

[23] Bruno Courcelle. The monadic second-order logic of graphs. I. Recognizable sets of finite graphs. *Inf. Comput.*, 85(1):12–75, 1990.

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

[25] Marek Cygan, Fedor V. Fomin, Alexander Golovnev, Alexander S. Kulikov, Ivan Mihajlin, Jakub Pachocki, and Arkadiusz Socala. Tight lower bounds on graph embedding problems. *J. ACM*, 64(3):18:1–18:22, 2017.

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

[27] Mark de Berg, Hans L. Bodlaender, Sándor Kisfaludi-Bak, Dániel Marx, and Tom C. van der Zanden. A framework for exponential-time-hypothesis-tight algorithms and lower bounds in geometric intersection graphs. *SIAM J. Comput.*, 49(6):1291–1331, 2020.

[28] Michał Dębski, Zbigniew Lonc, Karolina Okrasa, Marta Piecyk, and Paweł Rzążewski. Computing homomorphisms in hereditary graph classes: The peculiar case of the 5-wheel and graphs with no long claws. In Sang Won Bae and Heejin Park, editors, *33rd International Symposium on Algorithms and Computation, ISAAC 2022, December 19-21, 2022, Seoul, Korea*, volume 248 of *LIPIcs*, pages 14:1–14:16. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2022.

[29] Michał Dębski, Zbigniew Lonc, Karolina Okrasa, Marta Piecyk, and Paweł Rzążewski. Computing homomorphisms in hereditary graph classes: the peculiar case of the 5-wheel and graphs with no long claws. *CoRR*, abs/2205.13270, 2022.

[30] Keith Edwards. The complexity of colouring problems on dense graphs. *Theor. Comput. Sci.*, 43:337–343, 1986.

[31] László Egri, Dániel Marx, and Paweł Rzążewski. Finding list homomorphisms from bounded-treewidth graphs to reflexive graphs: a complete complexity characterization. In Rolf Niedermeier and Brigitte Vallée, editors, *35th Symposium on Theoretical Aspects of Computer Science, STACS 2018, February 28 to March 3, 2018, Caen, France*, volume 96 of *LIPIcs*, pages 27:1–27:15. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, 2018.

[32] Thomas Emden-Weinert, Stefan Hougardy, and Bernd Kreuter. Uniquely colourable graphs and the hardness of colouring graphs of large girth. *Comb. Probab. Comput.*, 7(4):375–386, 1998.

[33] Herbert B. Enderton. *A mathematical introduction to logic*. Elsevier, 2001.

[34] Paul Erdős, Arthur L. Rubin, and Herbert Taylor. Choosability in graphs. *Congr. Numer*, 26(4):125–157, 1979.

[35] Wolfgang Espelage, Frank Gurski, and Egon Wanke. How to solve NP-hard graph problems on clique-width bounded graphs in polynomial time. In Andreas Brandstädt and Van Bang Le, editors, *Graph-Theoretic Concepts in Computer Science, 27th International Workshop, WG 2001, Boltenhagen, Germany, June 14-16, 2001, Proceedings*, volume 2204 of *Lecture Notes in Computer Science*, pages 117–128. Springer, 2001.

[36] Tomás Feder and Pavol Hell. Edge list homomorphisms. http://theory.stanford.edu/~tomas/edgelist.ps.

[37] Tomás Feder and Pavol Hell. List homomorphisms to reflexive graphs. *Journal of Combinatorial Theory, Series B*, 72(2):236 – 250, 1998.

[38] Tomás Feder, Pavol Hell, and Jing Huang. List homomorphisms and circular arc graphs. *Combinatorica*, 19(4):487–505, 1999.

[39] Tomás Feder, Pavol Hell, and Jing Huang. Bi-arc graphs and the complexity of list homomorphisms. *Journal of Graph Theory*, 42(1):61–80, 2003.

[40] Tomás Feder, Pavol Hell, and Jing Huang. List homomorphisms of graphs with bounded degrees. *Discrete Mathematics*, 307(3-5):386–392, 2007.

[41] Tomás Feder, Pavol Hell, and Bojan Mohar. Acyclic homomorphisms and circular colorings of digraphs. *SIAM J. Discret. Math.*, 17(1):161–169, 2003.

[42] Tomás Feder and Moshe Y. Vardi. The computational structure of monotone monadic SNP and constraint satisfaction: A study through datalog and group theory. *SIAM J. Comput.*, 28(1):57–104, 1998.

[43] Fedor V. Fomin, Pinar Heggernes, and Dieter Kratsch. Exact algorithms for graph homomorphisms. *Theory Comput. Syst.*, 41(2):381–393, 2007.

[44] Jacob Fox and János Pach. Computing the independence number of intersection graphs. In Dana Randall, editor, *Proceedings of the Twenty-Second Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2011, San Francisco, California, USA, January 23-25, 2011*, pages 1161–1165. SIAM, 2011.

[45] Michael Freedman, László Lovász, and Alexander Schrijver. Reflection positivity, rank connectivity, and homomorphism of graphs. *Journal of the American Mathematical Society*, 20(1):37–51, 2007.

[46] Anna Galluccio, Pavol Hell, and Jaroslav Nešetřil. The complexity of H-colouring of bounded degree graphs. *Discret. Math.*, 222(1-3):101–109, 2000.

[47] Robert Ganian, Thekla Hamm, Viktoriia Korchemna, Karolina Okrasa, and Kirill Simonov. The fine-grained complexity of graph homomorphism parameterized by clique-width. In Mikolaj Bojanczyk, Emanuela Merelli, and David P. Woodruff, editors, *49th International Colloquium on Automata, Languages, and Programming, ICALP 2022, July 4-8, 2022, Paris, France*, volume 229 of *LIPIcs*, pages 66:1–66:20. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2022.

[48] Robert Ganian, Eun Jung Kim, and Stefan Szeider. Algorithmic applications of tree-cut width. *SIAM J. Discret. Math.*, 36(4):2635–2666, 2022.

[49] Michael R. Garey and David S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W. H. Freeman & Co., New York, NY, USA, 1979.

[50] Michael R. Garey, David S. Johnson, G. L. Miller, and Christos H. Papadimitriou. The complexity of coloring circular arcs and chords. *SIAM J. Algebraic Discret. Methods*, 1(2):216–227, 1980.

[51] Peter Gartland and Daniel Lokshtanov. Independent set on p_k-free graphs in quasi-polynomial time. In Sandy Irani, editor, *61st IEEE Annual Symposium on Foundations of Computer Science, FOCS 2020, Durham, NC, USA, November 16-19, 2020*, pages 613–624. IEEE, 2020.

[52] Peter Gartland, Daniel Lokshtanov, Tomás Masarík, Marcin Pilipczuk, Michał Pilipczuk, and Paweł Rzążewski. Maximum weight independent set in graphs with no long claws in quasi-polynomial time. *CoRR*, abs/2305.15738, 2023.

[53] Petr A. Golovach, Matthew Johnson, Daniël Paulusma, and Jian Song. A survey on the computational complexity of coloring graphs with forbidden subgraphs. *J. Graph Theory*, 84(4):331–363, 2017.

[54] Petr A. Golovach, Daniël Paulusma, and Jian Song. Closing complexity gaps for coloring problems on h-free graphs. *Inf. Comput.*, 237:204–214, 2014.

[55] Albert Gräf, Martin Stumpf, and Gerhard Weißenfels. On coloring unit disk graphs. *Algorithmica*, 20(3):277–293, 1998.

[56] Carla Groenland, Karolina Okrasa, Paweł Rzążewski, Alex D. Scott, Paul D. Seymour, and Sophie Spirkl. h-colouring p_t-free graphs in subexponential time. *Discret. Appl. Math.*, 267:184–189, 2019.

[57] Martin Grötschel, László Lovász, and Alexander Schrijver. Polynomial algorithms for perfect graphs. In *North-Holland mathematics studies*, volume 88, pages 325–356. Elsevier, 1984.

[58] Gregory Z. Gutin, Pavol Hell, Arash Rafiey, and Anders Yeo. A dichotomy for minimum cost graph homomorphisms. *Eur. J. Comb.*, 29(4):900–911, 2008.

[59] András Gyárfás. *Problems from the world surrounding perfect graphs.* Number 177. MTA Számítástechnikai és Automatizálási Kutató Intézet, 1985.

[60] Hugo Hadwiger. Über eine Klassifikation der Streckenkomplexe. *Vierteljschr. Naturforsch. Ges. Zürich*, 88(2):133–142, 1943.

[61] Rudolf Halin. S-functions for graphs. *Journal of geometry*, 8:171–186, 1976.

[62] Richard Hammack, Wilfried Imrich, and Sandi Klavžar. *Handbook of product graphs.* Discrete Mathematics and its Applications (Boca Raton). CRC Press, Boca Raton, FL, second edition, 2011. With a foreword by Peter Winkler.

[63] Richard H. Hammack, Wilfried Imrich, and Sandi Klavžar. *Handbook of product graphs.* CRC press, 2011.

[64] Pavol Hell and Jing Huang. Two remarks on circular arc graphs. *Graphs Comb.*, 13(1):65–72, 1997.

[65] Pavol Hell and Jaroslav Nešetřil. The core of a graph. *Discrete Mathematics*, 109(1-3):117–126, 1992.

[66] Pavol Hell and Jaroslav Nešetřil. *Graphs and homomorphisms*. Oxford University Press, 2004.

[67] Pavol Hell and Jaroslav Nešetřil. On the complexity of H-coloring. *J. Comb. Theory, Ser. B*, 48(1):92–110, 1990.

[68] Pavol Hell and Jaroslav Nešetřil. Colouring, constraint satisfaction, and complexity. *Comput. Sci. Rev.*, 2(3):143–163, 2008.

[69] Chính T. Hoàng, Marcin Kamínski, Vadim V. Lozin, Joe Sawada, and Xiao Shu. Deciding k-colorability of p_5-free graphs in polynomial time. *Algorithmica*, 57(1):74–81, 2010.

[70] Ian Holyer. The NP-completeness of edge-coloring. *SIAM J. Comput.*, 10(4):718–720, 1981.

[71] Shenwei Huang. Improved complexity results on k-coloring p_t-free graphs. *Eur. J. Comb.*, 51:336–346, 2016.

[72] John F. Humphreys. *A course in group theory*, volume 6. Oxford University Press, USA, 1996.

[73] Russell Impagliazzo and Ramamohan Paturi. On the complexity of k-SAT. *J. Comput. Syst. Sci.*, 62(2):367–375, 2001.

[74] Russell Impagliazzo, Ramamohan Paturi, and Francis Zane. Which problems have strongly exponential complexity? *J. Comput. Syst. Sci.*, 63(4):512–530, 2001.

[75] Bart M. P. Jansen and Jesper Nederlof. Computing the chromatic number using graph decompositions via matrix rank. In Yossi Azar, Hannah Bast, and Grzegorz Herman, editors, *26th Annual European Symposium on Algorithms, ESA 2018, August 20-22, 2018, Helsinki, Finland*, volume 112 of *LIPIcs*, pages 47:1–47:15. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2018.

[76] Peter Jeavons. On the algebraic structure of combinatorial problems. *Theor. Comput. Sci.*, 200(1-2):185–204, 1998.

[77] Peter Jeavons, David A. Cohen, and Marc Gyssens. Closure properties of constraints. *J. ACM*, 44(4):527–548, 1997.

[78] Tommy R. Jensen and Bjarne Toft. *Graph coloring problems*. John Wiley & Sons, 2011.

[79] Marcin Kamiński and Anna Pstrucha. Certifying coloring algorithms for graphs without long induced paths. *Discret. Appl. Math.*, 261:258–267, 2019.

[80] Richard M. Karp. Reducibility among combinatorial problems. In Raymond E. Miller and James W. Thatcher, editors, *Proceedings of a symposium on the Complexity of Computer Computations, held March 20-22, 1972, at the IBM Thomas J. Watson Research Center, Yorktown Heights, New York, USA*, The IBM Research Symposia Series, pages 85–103. Plenum Press, New York, 1972.

[81] Sándor Kisfaludi-Bak, Karolina Okrasa, and Paweł Rzążewski. Computing list homomorphisms in geometric intersection graphs. In Michael A. Bekos and Michael Kaufmann, editors, *Graph-Theoretic Concepts in Computer Science - 48th International Workshop, WG 2022, Tübingen, Germany, June 22-24, 2022, Revised Selected Papers*, volume 13453 of *Lecture Notes in Computer Science*, pages 313–327. Springer, 2022.

[82] Michael Lampis. Finer tight bounds for coloring on clique-width. *SIAM J. Discret. Math.*, 34(3):1538–1558, 2020.

[83] Benoit Larose. Families of strongly projective graphs. *Discussiones Mathematicae Graph Theory*, 22:271–292, 2002.

[84] Benoit Larose. Strongly projective graphs. *Canadian Journal of Mathematics*, 54(4):757–768, 2002.

[85] Benoit Larose and Claude Tardif. Strongly rigid graphs and projectivity. *Multiple-Valued Logic*, 7:339–361, 2001.

[86] Daniel Leven and Zvi Galil. NP completeness of finding the chromatic index of regular graphs. *J. Algorithms*, 4(1):35–44, 1983.

[87] Daniel Lokshtanov, Dániel Marx, and Saket Saurabh. Known algorithms on graphs of bounded treewidth are probably optimal. *ACM Trans. Algorithms*, 14(2):13:1–13:30, 2018.

[88] Tomasz Łuczak and Jaroslav Nešetřil. Note on projective graphs. *J. Graph Theory*, 47(2):81–86, 2004.

[89] Frédéric Maffray and Grégory Morel. On 3-colorable p_5-free graphs. *SIAM J. Discret. Math.*, 26(4):1682–1708, 2012.

[90] Konrad Majewski, Tomás Masarík, Jana Novotná, Karolina Okrasa, Marcin Pilipczuk, Paweł Rzążewski, and Marek Sokołowski. Max weight independent set in graphs with no long claws: An analog of the Gyárfás' path argument. In Mikołaj Bojańczyk, Emanuela Merelli, and David P. Woodruff, editors, *49th International Colloquium on Automata, Languages, and Programming, ICALP 2022, July 4-8, 2022, Paris, France*, volume 229 of *LIPIcs*, pages 93:1–93:19. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2022.

[91] Kim Marriott and Peter J. Stuckey. *Programming with constraints: an introduction.* MIT press, 1998.

[92] Ross M. McConnell. Linear-time recognition of circular-arc graphs. *Algorithmica*, 37(2):93–147, 2003.

[93] Ralph McKenzie. Cardinal multiplication of structures with a reflexive relation. *Fundamenta Mathematicae*, 70(1):59–101, 1971.

[94] George B. Mertzios and Paul G. Spirakis. Algorithms and almost tight results for 3-colorability of small diameter graphs. *CoRR*, abs/1202.4665, 2012.

[95] Jaroslav Nešetřil, Mark H. Siggers, and László Zádori. A combinatorial constraint satisfaction problem dichotomy classification conjecture. *Eur. J. Comb.*, 31(1):280–296, 2010.

[96] Jaroslav Nešetřil and Xuding Zhu. On sparse graphs with given colorings and homomorphisms. *J. Comb. Theory, Ser. B*, 90(1):161–172, 2004.

[97] Karolina Okrasa, Marta Piecyk, and Paweł Rzążewski. Full complexity classification of the list homomorphism problem for bounded-treewidth graphs. In Fabrizio Grandoni, Grzegorz Herman, and Peter Sanders, editors, *28th Annual European Symposium on Algorithms, ESA 2020, September 7-9, 2020, Pisa, Italy (Virtual*

Conference), volume 173 of *LIPIcs*, pages 74:1–74:24. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2020.

[98] Karolina Okrasa and Paweł Rzążewski. Subexponential algorithms for variants of the homomorphism problem in string graphs. *J. Comput. Syst. Sci.*, 109:126–144, 2020.

[99] Karolina Okrasa and Paweł Rzążewski. Complexity of the list homomorphism problem in hereditary graph classes. In Markus Bläser and Benjamin Monmege, editors, *38th International Symposium on Theoretical Aspects of Computer Science, STACS 2021, March 16-19, 2021, Saarbrücken, Germany (Virtual Conference)*, volume 187 of *LIPIcs*, pages 54:1–54:17. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2021.

[100] Karolina Okrasa and Paweł Rzążewski. Fine-grained complexity of the graph homomorphism problem for bounded-treewidth graphs. *SIAM J. Comput.*, 50(2):487–508, 2021.

[101] Marta Piecyk and Paweł Rzążewski. Fine-grained complexity of the list homomorphism problem: feedback vertex set and cutwidth. *CoRR*, abs/2009.11642, 2020.

[102] Marta Piecyk and Paweł Rzążewski. Fine-grained complexity of the list homomorphism problem: Feedback vertex set and cutwidth. In Markus Bläser and Benjamin Monmege, editors, *38th International Symposium on Theoretical Aspects of Computer Science, STACS 2021, March 16-19, 2021, Saarbrücken, Germany (Virtual Conference)*, volume 187 of *LIPIcs*, pages 56:1–56:17. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2021.

[103] Marcin Pilipczuk, Michał Pilipczuk, and Paweł Rzążewski. Quasi-polynomial-time algorithm for independent set in p_t-free graphs via shrinking the space of induced paths. In *4th Symposium on Simplicity in Algorithms, SOSA 2021, Virtual Conference, January 11-12, 2021*, pages 204–209. SIAM, 2021.

[104] Bert Randerath and Ingo Schiermeyer. 3-Colorability in P for p_6-free graphs. *Discret. Appl. Math.*, 136(2-3):299–313, 2004.

[105] Neil Robertson and Paul D. Seymour. Graph minors. III. Planar tree-width. *J. Comb. Theory, Ser. B*, 36(1):49–64, 1984.

[106] Neil Robertson and Paul D. Seymour. Graph minors. II. Algorithmic aspects of tree-width. *J. Algorithms*, 7(3):309–322, 1986.

[107] Paweł Rzążewski. Exact algorithm for graph homomorphism and locally injective graph homomorphism. *Inf. Process. Lett.*, 114(7):387–391, 2014.

[108] Thomas J. Schaefer. The complexity of satisfiability problems. In Richard J. Lipton, Walter A. Burkhard, Walter J. Savitch, Emily P. Friedman, and Alfred V. Aho, editors, *Proceedings of the 10th Annual ACM Symposium on Theory of Computing, May 1-3, 1978, San Diego, California, USA*, pages 216–226. ACM, 1978.

[109] Mark H. Siggers. Dichotomy for bounded degree h-colouring. *Discret. Appl. Math.*, 157(2):201–210, 2009.

[110] Mark H. Siggers. A new proof of the h-coloring dichotomy. *SIAM J. Discret. Math.*, 23(4):2204–2210, 2010.

[111] Alexander Soifer. *The mathematical coloring book: Mathematics of coloring and the colorful life of its creators*. Springer, 2009.

[112] Jeremy P. Spinrad. Circular-arc graphs with clique cover number two. *J. Comb. Theory, Ser. B*, 44(3):300–306, 1988.

[113] Vadim G. Vizing. Vertex coloring with given colors, diskret. analiz. 29 (1976) 3–10. *Discrete Anal. (in Russian)*, 29:3–10, 1976.

[114] Magnus Wahlström. New plain-exponential time classes for graph homomorphism. *Theory Comput. Syst.*, 49(2):273–282, 2011.

[115] Paul M. Weichsel. The Kronecker product of graphs. *Proceedings of the American mathematical society*, 13(1):47–52, 1962.

[116] Xuding Zhu. Recent developments in circular colouring of graphs. *Topics in Discrete Mathematics: Dedicated to Jarik Nešetřil on the Occasion of his 60th Birthday*, pages 497–550, 2006.