WARSAW UNIVERSITY OF TECHNOLOGY

DISCIPLINE OF SCIENCE - MATHEMATICS
FIELD OF SCIENCE - NATURAL SCIENCES

Ph.D. Thesis

Marta Piecyk, M.Sc.

Graph Homomorphisms –
Exploring the Boundaries of Tractability

Supervisors
Professor Zbigniew Lonc, Ph.D., D.Sc.
Paweł Rzążewski, Ph.D., D.Sc.

WARSAW 2024

Acknowledgements

I am extremely grateful to Paweł for being great supervisor, for all his time, for everything he taught me, and for inspiring me to start this journey. Without him I would not be in the place I am today.

I wish to thank all the people I had the pleasure of working with. Special thanks to Jajko, Carla, Isja, and Jesper – the co-authors of the results included in this dissertation. I am also thankful to Zbigniew for taking the responsibility of being my supervisor.

Many thanks to all my collegues from MiNI, especially the ones from, broadly defined, KTGiZU group, for making this time enjoyable. Thanks should also go to Ania, Karolina, Marcin, and Hubert for sharing (not always easy) experience of PhD studies – I cannot imagine going through all of this without them. I also cannot imagine this time without Ola and Aneta who I could always talk to.

Lastly, I would like to thank my family – my parents, Nati, and Adam – for all their support and love. The ones who always believed in me, were always there when I needed, and (each one in their own way) helped me go through my darkest times.

Abstract

For a fixed graph H, in the graph homomorphism problem, denoted by $\textsc{Hom}(H)$, we are given a graph G, and we have to determine whether there exists an edge preserving mapping $\varphi : V(G) \to V(H)$, i.e., for every $uv \in E(G)$, we have $\varphi(u)\varphi(v) \in E(H)$. This problem is a generalization of the well-studied k-Coloring, as $\textsc{Hom}(K_k)$, where K_k is the complete graph on k vertices, is equivalent to k-Coloring.

It is known that $\textsc{Hom}(H)$ is polynomial-time solvable when H is bipartite or contains a vertex with a loop. Hell and Nešetřil [JCTB, 1990] proved that for every other H, the problem is NP-hard. For a problem that is hard in general, a natural question is if it becomes tractable when we restrict the family of the input graphs G. One of possible restrictions is to bound some parameter of G. In this dissertation, we consider two parameters: the cutwidth ($\mathrm{ctw}(G)$) and the diameter ($\mathrm{diam}(G)$), whose behavior is very different and unusual.

For a linear ordering $v_1, \ldots, v_n$ of vertices of a graph G, the width of the ordering is the maximum number of edges from $\{v_1, \ldots, v_i\}$ to $\{v_{i+1}, \ldots, v_n\}$ over all i. The **cutwidth** of G is the minimum width over all linear orderings of $V(G)$. Jansen and Nederlof [TCS, 2019] proved that every instance G of k-Coloring can be solved in (randomized) time $2^{\mathrm{ctw}(G)} \cdot |G|^{\mathcal{O}(1)}$. This is very uncommon, as usually, for many graph parameters $t(G)$, the optimal running time of an algorithm solving k-Coloring is $f(k)^{t(G)} \cdot |G|^{\mathcal{O}(1)}$, where f is some increasing function of k.

In this dissertation, we try to find, for every H, a constant c_H such that we can solve every instance G of $\textsc{Hom}(H)$ in time $c_H^{\mathrm{ctw}(G)} \cdot |G|^{\mathcal{O}(1)}$, but there is no algorithm with running time $(c_H - \varepsilon)^{\mathrm{ctw}(G)} \cdot |G|^{\mathcal{O}(1)}$ for any $\varepsilon > 0$, unless the SETH fails. We define an asymptotic parameter, called $\mathrm{mimsup}^*(H)$, which is closely related to the sizes of maximum induced matchings in powers of H (with respect to direct product). As an evidence that $\mathrm{mimsup}^*(H)$ is a good candidate for c_H, we first show that the maximum number of states in a natural dynamic programming is precisely $\mathrm{mimsup}^*(H)^{\mathrm{ctw}(G)}$. Furthermore, we prove that for almost every graph H (and one can remove "almost" assuming two conjectures from early 2000's), there is no algorithm solving every instance G of $\textsc{Hom}(H)$ in time $(\mathrm{mimsup}^*(H) - \varepsilon)^{\mathrm{ctw}(G)} \cdot |G|^{\mathcal{O}(1)}$, for any $\varepsilon > 0$, unless the SETH

fails. Finally, we provide an algorithm solving every instance G of $\textsc{Hom}(H)$ in time $\exp(\mathcal{O}(\text{mimsup}^*(H) \cdot \text{ctw}(G) \log \text{ctw}(G))) \cdot |G|^{\mathcal{O}(1)}$. Let us point out that this is the first time when the parameterized complexity of a problem is linked to such an asymptotic parameter.

For the **diameter**, we do not even know if 3-Coloring can be solved in polynomial time for diameter-2 graphs. It is known that for $k \geqslant 4$, we cannot solve k-Coloring in subexponential time, unless the ETH fails. Mertzios and Spirakis [Algorithmica, 2016] proved that 3-Coloring is NP-hard for diameter-3 graphs and provided a subexponential-time algorithm for 3-Coloring for diameter-2 n-vertex graphs with running time $2^{\mathcal{O}(\sqrt{n \log n})}$.

In the dissertation we study the complexity of 3-Coloring for diameter-2 and -3 graphs. We show that 3-Coloring can be solved in time

- $2^{\mathcal{O}(n^{2/3} \log^{2/3} n)}$ on n-vertex diameter-3 graphs,
- $2^{\mathcal{O}(n^{1/3} \log^2 n)}$ on n-vertex diameter-2 graphs.

Furthermore, we consider $\textsc{Hom}(H)$ on bounded-diameter graphs for target graphs H other than triangles. We show that if H is triangle-free, the $\textsc{Hom}(H)$ problem can be solved in polynomial time on diameter-2 graphs.

Finally, we study the $\textsc{Hom}(C_{2k+1})$ problem, where C_{2k+1} is the cycle on $2k+1$ vertices and $k \geqslant 2$, i.e., we consider all odd cycles other than triangles. We prove that:

(1) every diameter-$(k+1)$ instance of $\textsc{Hom}(C_{2k+1})$ can be solved in polynomial time,

(2) every n-vertex diameter-$(k+2)$ instance of $\textsc{Hom}(C_{2k+1})$ can be solved in time $\exp(\mathcal{O}((n \log n)^{\frac{k+1}{k+2}}))$,

(3) there is no algorithm that solves every diameter-$(2k+2)$ instance of $\textsc{Hom}(C_{2k+1})$ in subexponential time, unless the ETH fails.

Let us point out that for $k = 1$, a result as in (1) would be precisely a polynomial-time algorithm for 3-Coloring on diameter-2 graphs.

Keywords: graph homomorphism, graph coloring, cutwidth, diameter, fine-grained complexity

Streszczenie

Dla ustalonego grafu H, w problemie homomorfizmu grafu, ozn. $\textsc{Hom}(H)$, mamy dany graf G i pytamy, czy istnieje funkcja $\varphi : V(G) \to V(H)$, która zachowuje krawędzie, tj. dla każdej $uv \in E(G)$, mamy $\varphi(u)\varphi(v) \in E(H)$. Ten problem jest uogólnieniem klasycznego problemu kolorowania grafów, ponieważ k-kolorowanie jest równoważne $\textsc{Hom}(K_k)$, gdzie K_k to graf pełny o k wierzchołkach.

Wiadomo, że problem $\textsc{Hom}(H)$ jest wielomianowy, gdy H zawiera wierzchołek z pętlą lub jest dwudzielny. Hell i Nešetřil [JCTB, 1990] pokazali, że dla każdego innego H, problem $\textsc{Hom}(H)$ jest NP-trudny. W przypadku problemu, który jest trudny w ogólności, naturalnym pytaniem jest, czy może stać się łatwy, jeśli ograniczymy klasę wejściowych grafów. Jedną z możliwości jest ograniczenie jakiegoś parametru wejściowego grafu. W tej rozprawie rozważamy dwa parametry grafu G – szerokość cięciową ($\mathrm{ctw}(G)$) i średnicę ($\mathrm{diam}(G)$).

Dla ustalonego liniowego porządku $v_1, \ldots, v_n$ wierzchołków grafu G, szerokość takiego porządku to największa liczba krawędzi z $\{v_1, \ldots, v_i\}$ do $\{v_{i+1}, \ldots, v_n\}$ po wszystkich i. **Szerokość cięciowa** grafu G to najmniejsza szerokość po wszystkich liniowych porządkach na $V(G)$. Jansen i Nederlof [TCS, 2019] pokazali, że każdą instancję G problemu k-kolorowania można rozwiązać w (randomizowanym) czasie $2^{\mathrm{ctw}(G)} \cdot |G|^{\mathcal{O}(1)}$. Jest to bardzo nietypowe, ponieważ dla wielu parametrów $t(G)$, optymalny czas, w którym można rozwiązać k-kolorowanie, to $f(k)^{t(G)} \cdot |G|^{\mathcal{O}(1)}$, gdzie f jest rosnącą funkcją k.

W tej rozprawie próbujemy dla każdego H znaleźć stałą c_H taką, że można rozwiązać każdą instancję G problemu $\textsc{Hom}(H)$ w czasie $c_H^{\mathrm{ctw}(G)} \cdot |G|^{\mathcal{O}(1)}$, ale, zakładając SETH, nie istnieje algorytm działający w czasie $(c_H - \varepsilon)^{\mathrm{ctw}(G)} \cdot |G|^{\mathcal{O}(1)}$ dla żadnego $\varepsilon > 0$. Definiujemy asymptotyczny parametr $\mathrm{mimsup}^*(H)$, który jest związany z rozmiarami największych indukowanych skojarzeń w kolejnych potęgach (w sensie iloczynu prostego) grafu H. Jako dowód na to, że $\mathrm{mimsup}^*(H)$ jest dobrym kandydatem na c_H, najpierw pokazujemy, że największa liczba stanów w naturalnym programowaniu dynamicznym wynosi dokładnie $\mathrm{mimsup}^*(H)^{\mathrm{ctw}(G)}$. Następnie pokazujemy, że dla *prawie każdego* grafu H (jeśli założymy dwie hipotezy ze wczesnych lat 2000, to słowo „prawie” może być pominięte), nie istnieje algorytm, który rozwiązuje każdą instancję G problemu $\textsc{Hom}(H)$ w

czasie $(\text{mimsup}^*(H) - \varepsilon)^{\text{ctw}(G)} \cdot |G|^{\mathcal{O}(1)}$ dla żadnego $\varepsilon > 0$, zakładając SETH. Ponadto, przedstawiamy algorytm, który rozwiązuje każdą instancję G problemu $\textsc{Hom}(H)$ w czasie $\exp(\mathcal{O}(\text{mimsup}^*(H) \cdot \text{ctw}(G) \log \text{ctw}(G))) \cdot |G|^{\mathcal{O}(1)}$. Podkreślmy, że to jest pierwszy raz, kiedy parametryzowana złożoność problemu została naturalnie połączona z parametrem zdefiniowanym asymptotycznie.

W przypadku **średnicy**, nie wiemy nawet, czy problem 3-kolorowania jest wielomianowy w grafach o średnicy dwa. Wiadomo, że dla $k \geqslant 4$, nie możemy rozwiązać k-kolorowania w czasie podwykładniczym, zakładając ETH. Mertzios i Spirakis [Algorithmica, 2016] udowodnili, że 3-kolorowanie jest NP-trudne w grafach o średnicy 3 oraz przedstawili podwykładniczy algorytm dla 3-kolorowania w n-wierzchołkowych grafach o średnicy 2 działający w czasie $2^{\mathcal{O}(\sqrt{n \log n})}$.

W tej rozprawie badamy złożoność 3-kolorowania dla grafów o średnicy dwa i trzy. Pokazujemy, że 3-kolorowanie może być rozwiązane w czasie

- $2^{\mathcal{O}(n^{2/3} \log^{2/3} n)}$ dla n-wierzchołkowych grafów o średnicy trzy,
- $2^{\mathcal{O}(n^{1/3} \log^2 n)}$ dla n-wierzchołkowych grafów o średnicy dwa.

Następnie rozważamy $\textsc{Hom}(H)$ dla grafów o ograniczonej średnicy oraz H innych niż trójkąty. Pokazujemy, że jeśli H nie zawiera trójkątów, to $\textsc{Hom}(H)$ w grafach o średnicy dwa może być rozwiązany w czasie wielomianowym.

Ponadto, badamy problem $\textsc{Hom}(C_{2k+1})$, gdzie C_{2k+1} jest cyklem o $2k+1$ wierzchołkach oraz $k \geqslant 2$, tj. rozważamy wszystkie cykle nieparzyste oprócz trójkątów. Pokazujemy, że

(1) każda instancja $\textsc{Hom}(C_{2k+1})$ o średnicy $(k+1)$ może być rozwiązana w czasie wielomianowym,

(2) każda n-wierzchołkowa instancja $\textsc{Hom}(C_{2k+1})$ o średnicy $(k+2)$ może być rozwiązana w czasie $\exp(\mathcal{O}((n \log n)^{\frac{k+1}{k+2}}))$,

(3) nie istnieje podwykładniczy algorytm rozwiązujący każdą instancję $\textsc{Hom}(C_{2k+1})$ o średnicy $(2k+2)$ zakładając ETH.

Zauważmy, że dla $k = 1$, wynik taki jak w (1) byłby dokładnie wielomianowym algorytmem dla 3-kolorowania grafów o średnicy dwa.

Słowa kluczowe: homomorfizm grafów, kolorowanie grafów, szerokość cięciowa, średnica, złożoność drobnoziarnista

Contents

Chapter 1

Introduction

One of the important tasks in the area of computational mathematics and theoretical computer science is to understand where the boundaries between tracable and hard problems lie. The classical distinction is based on the hypothesis that $\mathsf{P}\neq\mathsf{NP}$. However, nowadays the information that a problem is just NP-hard is not always satisfying. First, one can ask if on NP-hard problem can be solved in subexponential time, i.e., in time $2^{o(n)} \cdot n^{\mathcal{O}(1)}$, where n is the size of the input instance. Furthermore, for a problem hard in general, we can ask if it can become tractable when we restrict the input instances.

For graph problems, possible restrictions include forbidding some fixed graph or some family of graphs as induced subgraphs [27–32], bounding some parameter of the input graph [10, 11, 61, 62, 65, 71, 97, 107], restricting to intersection graphs of geometric objects [6, 7, 75, 102, 123], etc. In case of bounded parameters, one can also ask how the complexity of the problem depends on the parameter of the input instance. In last years, the so-called *parameterized complexity* of many problems has been widely studied [11, 36, 61, 64, 71, 73, 84, 86, 88, 97, 107, 116]. In a parameterized problem Π, the input consists of an instance I and some parameter k. We say that Π is *fixed-parameter tractable (FPT)* if every instance (I, k) of Π such that the size of I is n can be solved in time $f(k) \cdot n^{\mathcal{O}(1)}$, where f is a function that depends only on the parameter k. Note that if a problem Π is FPT with respect to some parameter $t(G)$ of the input graph G, then for the class of graphs with $t(G)$ bounded by a constant, Π is polynomial-time solvable.

Proving the existence of an FPT algorithm may still not be satisfying if the function f is large. In the so-called *fine-grained complexity* we are interested in obtaining algorithms with running time $f(k) \cdot n^{\mathcal{O}(1)}$ such that the function f is optimal under some standard

complexity assumptions, which we will describe later.

Now let us give some examples of parameters that can be used in designing efficient algorithms. Arguably, one of the most studied graph parameters is the treewidth [9, 52, 55, 55, 61, 62, 101, 108, 121, 124]. Intuitively, treewidth is a parameter that measures how close a graph is to a tree. Since usually problems are tractable on trees and can be solved using dynamic programming, the treewidth is a useful parameter in designing algorithms. Another "width" parameters are for instance: pathwidth [44, 45, 101, 121], cliquewidth [36, 65, 97], cutwidth [11, 71, 87], and twinwidth [13–15, 72].

While "width" parameters measure how "thick" a graph is, usually comparing to a tree, we can ask for parameters that measure how "high" our graph is, and introduce "depth" parameters. The best known "depth" parameter is the treedepth, a recursive analogue of the treewidth [77, 88, 116, 117]. The treedepth measures how close a graph is to a star. There have been also other "depth" analogues of the "width" parameters introduced: shrubdepth (which is related to the cliquewidth), branchdepth, and rankdepth [42, 66].

Other types of graph parameters are distances to some fixed hereditary graph classes [22, 22, 64, 73, 73, 83, 83, 88, 97, 101, 128], or some variations of well-known parameters, for instance: $\mathcal{H}$-elimination distance [19, 20, 80], $\mathcal{H}$-treewidth [53], tree-independence number [39, 40].

Lower bounds. In order to prove better lower bounds we need stronger assumptions than $\mathsf{P}\neq\mathsf{NP}$. The standard hypothesis that $\mathsf{P}\neq\mathsf{NP}$ implies that there is no algorithm solving every instance of 3-Sat in polynomial time. Impagliazzo and Paturi [81, 82] introduced the Exponential Time Hypothesis (ETH) and the Strong Exponential Time Hypothesis (SETH) in terms of the complexity of Sat problems, which for our purposes can be stated as follows.

Conjecture 1.1 (Exponential Time Hypothesis)**.** *There exists $\delta > 0$ such that there is no algorithm that solves every instance of 3-Sat with n variables in time $2^{\delta \cdot n} \cdot n^{\mathcal{O}(1)}$.*

Conjecture 1.2 (Strong Exponential Time Hypothesis)**.** *There is no algorithm that solves every instance of CNF-Sat with n variables and m clauses in time $(2-\varepsilon)^n \cdot (n+m)^{\mathcal{O}(1)}$, for any $\varepsilon > 0$.*

Let us mention that the ETH is often used in its stronger form, where we can assume that that the number of clauses is linear in number of variables (in general we can only

bound this number by $\mathcal{O}(n^3)$).

Theorem 1.3 (Sparsification Lemma [82]). *If the ETH is true, then there exists $\delta > 0$ such that there is no algorithm that solves every instance of 3-Sat with n variables and $\mathcal{O}(n)$ clauses in time $2^{\delta \cdot n} \cdot n^{\mathcal{O}(1)}$.*

1.1 Graph coloring

One of the most classic graph problems is k-Coloring, where we are given a graph G and we have to determine whether there is an assignment $c : V(G) \to \{1, \ldots, k\}$ such that for every $uv \in E(G)$, we have $c(u) \neq c(v)$. It is known that k-Coloring is polynomial-time solvable for $k \leqslant 2$, and NP-hard otherwise [33]. Furthermore, the problem can be solved in time $2^n \cdot n^{\mathcal{O}(1)}$ for n-vertex graphs [8, 92]. On the other hand, there is no algorithm that solves every n-vertex instance of k-Coloring in time $2^{o(n)} \cdot n^{\mathcal{O}(1)}$. The k-Coloring problem has been widely studied on various graph classes [7, 27, 30, 87, 101, 105, 111] and under various parameterizations [22, 34, 64, 71, 73, 83, 84, 87, 97, 101]. For treewidth, by standard dynamic programming, every instance G of k-Coloring can be solved in time $k^t \cdot |G|^{\mathcal{O}(1)}$, provided that G is given along with a tree decomposition of width t. Lokshtanov, Marx, and Saurabh showed that this running time is optimal under the SETH, i.e., there is no algorithm that solves every instance G of k-Coloring in time $(k - \varepsilon)^{\mathrm{tw}(G)} \cdot |G|^{\mathcal{O}(1)}$ for any $\varepsilon > 0$, unless the SETH fails [101]. In fact, in their lower bound one can replace treewidth with pathwidth or the size of a minimum feedback vertex set (both parameters are lower bounded by the treewidth, so by such a replacement, we get stronger lower bounds). In case of cliquewidth ($\mathrm{cw}(G)$), Lampis proved that every instance G of k-Coloring given with a clique expression of width t can be solved in time $(2^k - 2)^t \cdot |G|^{\mathcal{O}(1)}$, which optimal under the SETH [97].

1.2 Cutwidth

Given a linear ordering $v_1, \ldots, v_n$ of vertices of a graph G, the width of the ordering is the maximum number of edges from $\{v_1, \ldots, v_i\}$ to $\{v_{i+1}, \ldots, v_n\}$ over all i. The cutwidth of G, denoted by $\mathrm{ctw}(G)$, is the minimum width over all possible linear orderings of G.

Jansen and Nederlof provided a randomized algorithm that solves every instance G of

k-Coloring in time $2^t \cdot |G|^{\mathcal{O}(1)}$ (and deterministic one with running time $2^{k\omega} \cdot |G|^{\mathcal{O}(1)}$, where $\omega \leqslant 2.37$ is the matrix multiplication constant [135]) provided that G is given with linear ordering of width t. This is very uncommon, as for many graph parameters $t(G)$ (eg. treewidth, cliquewidth), every instance G of k-Coloring can be solved in time $f(k)^{t(G)} \cdot |G|^{\mathcal{O}(1)}$ for some increasing function f of k and cannot be solved in time $f(k)^{o(t(G))} \cdot |G|^{\mathcal{O}(1)}$, unless the ETH fails. In particular, the base of the exponent in the running time depends on the number of colors. Thus the case of q-Coloring parameterized by the cutwidth of the input graph seems very special. A natural question is how far the phenomenom of k-Coloring and cutwidth can be generalized. One can ask if the results for the decision version of the problem can be applied to the counting version. Groenland *et al.* [71] studied the complexity of counting k-colorings modulo prime p, i.e., $\#_p k$-Coloring. They provided tight bounds for the complexity of $\#_p k$-Coloring, where the base of the exponent is either k or $k-1$, depending on the relation of k and p.

1.3 Diameter

For a connected graph G, the diameter ($\mathrm{diam}(G)$) is the maximum possible distance (the number of edges on a shortest path) of two distinct vertices of G. We say that G is a *diameter-d* graph if $\mathrm{diam}(G) \leqslant d$. Recently, bounded-diameter graphs received a lot of attention [12, 16, 23, 48, 89, 105, 106, 111]. It is known that almost all graphs have diameter 2 [93], i.e., the probability that a random graph on n vertices has diameter 2 tends to 1 when n tends to infinity. Moreover, graphs from "real-life" applications often have bounded diameter, for instance social networks tend to have bounded diameter [131]. Furthermore, a diameter-2 graph can contain arbitrarily complicated structure. Indeed, consider any graph G and let G^+ be obtained by adding a universal vertex (connected to all other vertices) to G. Then G^+ has diameter at most 2.

Therefore, if for some $d \geqslant 2$, we have an efficient algorithm for the class of diameter-d graphs, then this algorithm is efficient for a "very wide" class of graphs. On the other hand, not all of the standard approaches can be used on bounded-diameter graphs – note that the class of diameter-d graphs is not closed under vertex deletion. Furthermore, many problems are hard already for diameter-2 graphs, for instance Independent Set cannot be solved in subexponential time on diameter-2, unless the ETH fails. In particular, most

of the hardness proofs use the construction of G^+. As an example, let G be an instance of k-Coloring. Observe that G is k-colorable if and only if G^+ is $(k+1)$-colorable. This simple observation yields that for $k \geqslant 4$, there is no subexponential-time algorithm for k-Coloring on diameter-2 graphs, unless the ETH fails. Thus, for bounded-diameter graphs, it is only interesting to consider k-Coloring when $k = 3$.

A textbook reduction from NAE-Sat implies that already for $d = 4$, the 3-Coloring problem is NP-hard and cannot be solved in subexponential time on diameter-d graphs, unless the ETH fails [126, Theorem 9.8]. Mertzios and Spirakis [111] proved that 3-Coloring is NP-hard on diameter-3 graphs. However, their reduction is quadratic, so it only excludes an algorithm with running time $2^{o(\sqrt{n})}$ under the ETH. Actually, the authors carefully analyzed how the lower bound depends on the minimum degree δ of the input graph, and presented three hardness reductions, each for a different range of δ. Furthermore, they showed that the problem can be solved in time $2^{\mathcal{O}\left(\min(\delta\cdot\Delta, \frac{n\log\delta}{\delta})\right)}$, where Δ is the maximum degree. Let us point out that if $\Delta = \Theta(n)$ and $\delta = \mathcal{O}(1)$, then the running time is exponential in n.

For diameter-2 graphs, Mertzios and Spirakis [111] provided a subexponential-time algorithm for 3-Coloring with running time $2^{\mathcal{O}(\sqrt{n\log n})}$ on n-vertex instances. Let us point out that the bound $\sqrt{n}$ appears naturally for various parameters of diameter-2 graphs, for example the maximum degree of such a graph is $\Omega(\sqrt{n})$. The question of whether 3-Coloring on diameter-2 graphs is polynomial-time solvable remains open despite some work on the problem: the 3-Coloring problem on bounded-diameter graphs was intensively studied on instances with some additional restrictions, i.e., on graphs with some forbidden induced subgraphs – and on such graph classes polynomial-time algorithms were given [89, 105, 106].

1.4 Graph homomorphisms

For a fixed graph H, called *target*, in the graph homomorphism problem, denoted by Hom(H), we are given a graph G and the task is to determine whether there exists a homomorphism φ from G to H, i.e., a mapping $\varphi : V(G) \to V(H)$ such that for every $uv \in E(G)$, it holds $\varphi(u)\varphi(v) \in E(H)$. In the list homomorphism problem, denoted by LHom(H), the graph G is given along with a list function $L : V(G) \to 2^{V(H)}$, and the

task is to determine whether there is a homomorphism φ from G to H that additionally respects the lists L, i.e., for every $v \in V(G)$, it holds $\varphi(v) \in L(v)$.

If H contains a vertex with a loop or is bipartite, the $\textsc{Hom}(H)$ problem can be solved in polynomial time. Hell and Nešetřil [78] proved that for every other graph H, the $\textsc{Hom}(H)$ problem is NP-hard. The complexity dichotomy was also provided in case of $\textsc{LHom}(H)$ – note that for every H, if $\textsc{Hom}(H)$ is NP-hard, then so is $\textsc{LHom}(H)$. However, the $\textsc{LHom}(H)$ problem can be also hard if H is bipartite or contains a vertex with a loop. The complexity dichotomy was proved in three steps: (i) for reflexive target graphs (every vertex has a loop), where the polynomial-time cases are precisely interval graphs [56], (ii) for bipartite target graphs [57], (iii) for general graphs [58].

The graph homomorphism problem has been widely studied [18, 21, 25, 26, 28, 35, 51, 65, 130]. From the fine-grained perspective, a lot of attention was put in the parameterization by treewidth of the instance graph [52, 55, 62, 121, 124].

Let us point out that since in general graph homomorphism problem is not that symmetric as k-Coloring, investigating its complexity for various parameters can tell us more about the parameters themselves.

1.5 Results

In this dissertation, we study the $(\textsc{L})\textsc{Hom}(H)$ problem on bounded-cutwidth and bounded-diameter graphs. These two parameters present extremely different behavior. Recall that almost all graphs have diameter two. This implies that almost all graphs have cutwidth $\Omega(\sqrt{n})$ – note that n-vertex diameter-2 graph needs to have a vertex of degree at least $\sqrt{n}$ and the cutwidth is always lower bounded by $\lfloor \frac{\Delta(G)}{2} \rfloor$, where $\Delta(G)$ is the maximum degree of G.

On the other hand, for cutwidth, many problems are not only polynomial-time solvable when we bound cutwidth by a constant, but also FPT with respect to cutwidth, and even the function of the parameter in the running time is unusually small. This is not the case for the diameter as already for diameter-2 graphs, many problems are NP-hard and cannot be solved in subexponential time under the ETH.

The results presented in the dissertation are based on the following papers of the author.

[70] Carla Groenland, Isja Mannens, Jesper Nederlof, Marta Piecyk, Paweł Rzążewski. Towards tight bounds for the graph homomorphism problem parameterized by cutwidth via asymptotic matrix parameters. *51st International Colloquium on Automata, Languages, and Programming, ICALP 2024*, volume 297 of LIPIcs, pages 77:1–77:21, Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2024.

[47] Michał Dębski, Marta Piecyk, Paweł Rzążewski. Faster 3-Coloring of bounded-diameter graphs. *29th Annual European Symposium on Algorithms, ESA 2021*, volume 204 of LIPIcs, pages 37:1–37:15, Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2021.

[48] Michał Dębski, Marta Piecyk, Paweł Rzążewski. Faster 3-Coloring of bounded-diameter graphs. *SIAM Journal on Discrete Mathematics*, 36(3):2205–2224, 2022.

[127] Marta Piecyk. C_{2k+1}-Coloring of bounded-diameter graphs. *49th International Symposium on Mathematical Foundations of Computer Science, MFCS 2024*, volume 306 of LIPIcs, pages 78:1–78:15, Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2024.

Moreover, we also use some results from other papers of the author.

[121] Karolina Okrasa, Marta Piecyk, Paweł 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, Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2020.

[128] Marta Piecyk, Paweł Rzążewski. Fine-grained complexity of the list homomorphism problem: feedback vertex set and cutwidth. *38th International Symposium on Theoretical Aspects of Computer Science, STACS 2021*, volume 187 of LIPIcs, pages 56:1–56:17, Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2021.

The above two papers are not included in the dissertation as they were submitted before starting PhD studies. In particular, the second paper is based on the Master thesis of the author.

1.5.1 Cutwidth

The first parameter we consider is the cutwidth. By standard dynamic programming, every instance G of $\textsc{Hom}(H)$ given with a linear ordering of width w can be solved in time $|H|^w \cdot |G|^{\mathcal{O}(1)}$. Recall that k-Coloring can be solved in deterministic time $2^{\omega \operatorname{ctw}(G)} \cdot |G|^{\mathcal{O}(1)}$ [87] – in particular the base of the exponent in the running time does not depend on k. A natural question, which was asked by Jansen [85], is if this can be generalized to arbitrary target graph H, i.e., is there a constant c such that for every H, there is an algorithm solving every instance G of $\textsc{Hom}(H)$ in time $c^{\operatorname{ctw}(G)} \cdot |G|^{\mathcal{O}(1)}$? Piecyk and Rzążewski [128] answered this question in the negative – in fact such a constant cannot exist (under the ETH) even if we restrict the family of target graphs to odd cycles. However, the lower bounds provided in [128] do not match any known upper bounds.

Therefore, we would like to solve the following open problem.

Open Problem 1. *Describe, for any fixed non-bipartite graph H, a constant c_H such that:*

1. *There is an algorithm that, for all $k, n \in \mathbb{N}$, given an n-vertex graph G with linear ordering of width k, solves $\textsc{Hom}(H)$ in time $c_H^k \cdot n^{\mathcal{O}(1)}$, and*
2. *Assuming the SETH, for any $\varepsilon > 0$, there is no algorithm that, for all $k, n \in \mathbb{N}$, given an n-vertex graph G with linear ordering of width k, solves $\textsc{Hom}(H)$ in time $(c_H - \varepsilon)^k \cdot n^{\mathcal{O}(1)}$.*

For every non-bipartite graph H, we define a parameter $\operatorname{mimsup}(H)$, and we give strong evidence that $c_H = \operatorname{mimsup}(H)$.

Let us first discuss the definition of mimsup. Let A be a 0-1 matrix. By $\operatorname{mim}(A)$ we denote the maximum ℓ such that A contains the $\ell \times \ell$ identity matrix as a submatrix[1]. It can be easily seen that if A is the bi-adjacency matrix of a bipartite graph H', then $\operatorname{mim}(A)$ equals the size of a maximum induced matching in H'. Let us point out that the results of [128] imply that c_H needs to be at least $\operatorname{mim}(A_H)$, if A_H is the adjacency matrix of H.

[1] A submatrix of a matrix A is any matrix that can be obtained from A by removing and reordering any of its rows and columns.

Next, by $\otimes$ we denote the Kronecker product, i.e., for two matrices A, B, where $A = (a_{i,j})_{i\in[n],j\in[m]}$ is $n \times m$ matrix,

$$A \otimes B = \begin{pmatrix} a_{1,1}B & a_{1,2}B & \dots & a_{1,m}B \\ a_{2,1}B & a_{2,2}B & \dots & a_{2,m}B \\ & \dots & & \\ a_{n,1}B & a_{n,2}B & \dots & a_{n,m}B \end{pmatrix}.$$

For an integer k and a matrix A, we define $A^{\otimes k}$ to be the Kronecker product of k copies of A. Now, for a matrix A, we can define

$$\mathrm{mimsup}(A) := \sup_{k\in\mathbb{N}} \mathrm{mim}(A_H^{\otimes k})^{1/k}.$$

For a graph H, we define $\mathrm{mimsup}(H) = \mathrm{mimsup}(A_H)$, where A_H is the adjacency matrix of H. We remark that $\mathrm{mimsup}(H)$ can be also defined in purely graph-theoretic way, in terms of the size of a maximum matching in a certain graph product. See Chapter 4 for more thorough definitions and details.

Representative sets. Let us present some intuition behind the mimsup parameter and its connection to so-called *representative sets*. The algorithm for k-Coloring parameterized by the cutwidth from [87] is based on the fact that in the dynamic programming, instead of keeping track of all possible colorings of some set of vertices, only some small subset, that still carries the same information about the solution, is remembered. The idea of representative sets has been used for example to obtain fast algorithms for the k-Path problem [112] or connectivity problems parameterized by treewidth [9, 63].

Let us describe the notion of representative sets in our setting. Let G, H be graphs, let $v_1, \dots, v_n$ be a linear ordering of $V(G)$ of width w and let $X_i \subseteq \{v_1, \dots, v_i\}$ (resp., $Y_i \subseteq \{v_{i+1}, \dots, v_n\}$) be the set of these vertices v such that v has a neighbor in $\{v_{i+1}, \dots, v_n\}$ (resp., $\{v_1, \dots, v_i\}$). We aim to determine if $G \to H$. Let $\mathcal{A}$ be some set of colorings of X_i (by "colorings" here we mean the homomorphisms to H). We say that a set $\mathcal{A}' \subseteq \mathcal{A}$ *represents* $\mathcal{A}$ if for every coloring φ of Y_i, if there is a coloring $\psi \in \mathcal{A}$ such that $\varphi \cup \psi$ respects the edges between X_i and Y_i, then there is $\psi' \in \mathcal{A}'$ such that $\varphi \cup \psi'$ respect the edges between X_i and Y_i. For simplicity of this description, assume that $|X_i| = |Y_i|$ and the edges between X_i and Y_i (let us call them E_i) form a matching. So let us consider a set $\mathcal{A}$ which does not have a proper subset that represents $\mathcal{A}$. This means that for every

$\psi \in \mathcal{A}$ there exists some coloring φ of Y_i such that $\varphi \cup \psi$ respects the edges of E_i, but for every other $\psi' \in \mathcal{A}$, the coloring $\psi' \cup \varphi$ does not respect at least one edge of E_i.

Now let $k := |E_i|$ and let A_H be the adjacency matrix of H. Each row index of the matrix $A_H^{\otimes k}$ can be interpreted as a coloring ψ of X_i, each column of $A_H^{\otimes k}$ can be interpreted as a coloring φ of Y_i, and $A_H^{\otimes k}[\psi, \varphi] = 1$ if and only if the coloring $\psi \cup \varphi$ respects the edges of E_i. Therefore, the largest possible size of $\mathcal{A}$ is precisely $\mathrm{mim}(A_H^{\otimes k})$. Recall that the width of the ordering $v_1, \ldots, v_n$ is w and thus $k \leqslant w$. So $\mathrm{mim}(A_H^{\otimes k}) \leqslant (\mathrm{mim}(A_H^{\otimes k})^{1/k})^w \leqslant \mathrm{mimsup}(H)^w$.

By a reasoning as above, we prove the following theorem (here we only present an informal statement), which is the first evidence that $c_H = \mathrm{mimsup}(H)$.

Theorem 1.4 (Informal statement of Theorem 4.3)**.** *In the context of the natural dynamic programming algorithm for* Hom(H) *parameterized by cutwidth w, there exist representative sets of size at most* $\mathrm{mimsup}(H)^w$.

Computing the representative sets. The reason why Theorem 4.3 does not imply an algorithm for Hom(H) with running time $\mathrm{mimsup}(H)^{\mathrm{ctw}(G)} \cdot |G|^{\mathcal{O}(1)}$ (and thus we lack tight bounds for Hom(H) parameterized by the cutwidth of the input graph) is that we do not know how to compute representative sets efficiently. Similar issues with the computation of representative sets are present in different problems: It withholds us, for example, from getting faster algorithms for connectivity problems such as Traveling Salesperson (both parameterized by treewidth [9, 38] and the classic parameterization by the number of cities [115, Theorem 3]), and polynomial kernelization algorithms for Odd Cycle Transversal [94].

In our case we try to deal with it in two ways.

Half-induced matching. First, we introduce a parameter him, whose matrix definition is similar to mim, where instead of an identity submatrix we look for a triangular matrix with ones on the diagonal. For a graph H, we again set $\mathrm{him}(H) = \mathrm{him}(A_H)$, where A_H is the adjacency matrix of H. We will refer to him as *the size of a maximum half-induced matching*. In the following theorem we describe the relation between him and mimsup.

Theorem 1.5. *For every non-bipartite graph H with adjacency matrix A_H and $k \in \mathbb{N}$,*

$$\mathrm{him}(A_H) \leqslant \mathrm{mimsup}(A_H) = \lim_{k \to \infty} \mathrm{mim}(A_H^{\otimes k})^{1/k} \quad \textit{and} \quad \mathrm{mim}(A_H^{\otimes k}) \leqslant k^{\mathrm{him}(A_H)k}.$$

The upper bound in Theorem 1.5 uses an argument similar to the 'neighborhood chasing' argument for the upper bounds on multi-colored Ramsey numbers [54]. This argument can in fact be made algorithmic in the sense that it can be used to compute representative sets for $\textsc{Hom}(H)$ of size at most $\mathcal{O}(k^{\mathrm{him}(H)k})$ in time $\mathcal{O}(k^{2\cdot\mathrm{him}(H)k})$ for input graphs given with a linear ordering of width at most k. We can use this to obtain the following result.

Theorem 1.6. *For any graphs G and H, where G is given with a linear ordering of width k, in time $\mathcal{O}(k^{2k\cdot\mathrm{him}(H)}\cdot|H|^4|G|)$ one can decide whether G admits a homomorphism to H.*

The bound from Theorem 1.5 yields the following.

Theorem 1.7. *For any graphs G and H, where G is given with a linear ordering of width k, in time $\mathcal{O}(k^{2k\cdot\mathrm{mimsup}(H)}\cdot|H|^4|G|)$ one can decide whether G admits a homomorphism to H.*

Recall that standard dynamic programming for $\textsc{Hom}(H)$ runs in time $|H|^{\mathrm{ctw}(G)}\cdot|G|^{\mathcal{O}(1)}$. Therefore, if H is fixed, then the running time of the algorithm from Theorem 1.7 can be worse. However, in Theorems 1.6 and 1.7, we do not assume that H is fixed, and in particular, H can be part of the input. In this case, the algorithm from Theorem 1.7 can be significantly faster than standard dynamic programming when H has small mimsup.

It also should be noted that a similar notion of half-induced matching of a compatibility matrix was already introduced in previous work in the context of representative sets of the AntiFactor problem [108] parameterized by treewidth and list size. However, in that setting, the authors were only able to provide a lower bound for their problem, and they did not manage to make the connection with half-induced matchings algorithmic. Additionally, their compatibility matrix has a very specific structure: it is indexed with integers and the value of an entry only depends on the sum of the values associated with the row and column.

Support-rank. Another attempt to solve the $\textsc{Hom}(H)$ problem is based on a low-rank matrix approach, which is a generalization of the algorithm for k-Coloring [87]. For a matrix A, we define a parameter support-rank, which is the smallest rank over any field of a matrix A' with the same support as A (i.e., $A'[i,j]=0$ if and only if $A[i,j]=0$). We prove that $\mathrm{mimsup}(H)$ is always at most the support-rank of the adjacency matrix of H. Furthermore, we prove the following generalization of the result from [87].

Theorem 1.8. *Let H be a non-bipartite graph on h vertices. Suppose we are given an $h \times h$ matrix over a field $\mathbb{F}$ with the same support as the adjacency matrix of H and rank r. Then there exists an algorithm that, given a linear ordering of an n-vertex graph G of width k, decides whether $G \to H$ in time $\mathcal{O}\Big((r^{k\cdot\omega}hk + h^3)|G|\Big)$.*

The caveat in Theorem 1.8 is that a small-rank matrix with the same support as the adjacency matrix of H must be given. If $\mathbb{F}$ is a finite field, an optimal such matrix can be found in time $|\mathbb{F}|^{h^2} \cdot h^{\mathcal{O}(1)}$ by brute-force, which is constant if both $|\mathbb{F}|$ and h are constants. However, in general it is not clear how to find such an optimal matrix.

Therefore, we present a combinatorial approach for finding a small-rank matrix with the same support, which does not necessarily achieve the support-rank, but can be computed efficiently. The approach is based on a parameter $\mathrm{cov}(H)$, which is closely related to *product dimension* (also known as *Prague dimension* or just *dimension*) [118, 119] and a *biclique covering number* studied under names *bipartite degree*, *local biclique cover number* [43, 60], and which is as special case of so-called *local covering numbers* also studied in the literature [17, 91]. We prove the following.

Theorem 1.9. *Let H be a fixed non-bipartite graph and let $r = \mathrm{cov}(H)$. The Hom(H) problem on n-vertex instances given with a linear ordering of width k can be solved in time $\mathcal{O}\Big((r+1)^{rk\cdot\omega}n^2\Big)$.*

We point out that for fixed H, Theorem 1.8 provides better upper bounds than Theorems 1.6 and 1.7. On the other hand, we know that there are families of graphs $\mathcal{H}$ for which him is bounded by a constant and the support-rank is unbounded. Therefore, for such graph classes $\mathcal{H}$, the upper bounds from Theorem 1.6 are better.

Lower bound. The next evidence that $c_H = \mathrm{mimsup}(H)$ is the following lower bound.

Theorem 1.10.

1. *There exists $\delta > 0$, such that for every non-bipartite projective core H, there is no algorithm solving every instance G of Hom(H) in time $\mathrm{mimsup}(H)^{\delta\cdot\mathrm{ctw}(G)} \cdot |G|^{\mathcal{O}(1)}$, unless the ETH fails.*

2. *Let H be a connected non-bipartite projective core. There is no algorithm solving every instance G of Hom(H) in time $(\mathrm{mimsup}(H) - \varepsilon)^{\mathrm{ctw}(G)} \cdot |G|^{\mathcal{O}(1)}$ for any $\varepsilon > 0$, unless the SETH fails.*

Let us point out that in the statement we assume H to be a so-called *projective core* (see Chapter 3 for definitions). It is known that almost all graphs are projective cores [79, 104]. Actually, we provide similar lower bounds for every non-bipartite graph H with a refined version of mimsup: we define a parameter mimsup* such that we always have mimsup$^*(H) \leqslant$ mimsup(H) and if H is a projective core, then mimsup$^*(H) =$ mimsup(H). The definition of mimsup* is related to standard preprocessing that can be performed on H. In particular, in all the upper bounds that we presented so far, we can replace mimsup with mimsup* (see Section 4.3). We extend the results of Theorem 1.10 (where mimsup is replaced with mimsup*) to every non-bipartite graph H, assuming two conjectures from early 2000s. These conjectures are assumed in all similar lower bounds for $\textsc{Hom}(H)$ under different parameterizations [65, 124].

Mimsup and other graph parameters. One of our main conceptual contributions is the introduction of the mimsup parameter in the context of parameterized algorithms. It is actually the first asymptotic rank parameter shown to be relevant in this context. Our mimsup parameter is very similar to the *asymptotic induced matching number* studied by Arunachalam et al. [3] which was introduced for k-partite, k-uniform hypergraphs (and so, in the graph setting, only for bipartite graphs).

Since our parameter is defined in an asymptotic way, it is not clear whether it can be computed efficiently or even if it can be computed at all. An indication that computing mimsup can be hard can be the case of a parameter called *Shannon capacity* (denoted by $\Theta(G)$) [103, 132], which is also defined in an asymptotic way (for exact definition we refer to Section 4.5.3). It is even not clear whether determining if the Shannon capacity is at most some given value α is decidable [2, Question 6]. Furthermore computing it seems very challenging even for very basic graphs. For instance, it is not known what the Shannon capacity of the cycle on 7 vertices is [68].

As we already mentioned, we compare mimsup with some other graph parameters – in fact this is similar approach as the ones used for Shannon capacity, where in order to compute $\Theta(G)$, some other parameters that lower/upper bound $\Theta(G)$ were investigated [103]. We prove that for every graph H, we have him$(H) \leqslant$ mimsup$(H) \leqslant$ support-rank(H). Furthermore, we present families of graphs for which the him(H) is bounded by a constant and the support-rank is unbounded, and thus him(H) and support-rank(H) are not functionally equivalent. However, we do not know if mimsup is functionally equivalent to

any of the two parameters.

1.5.2 Diameter

The second parameter we consider in the dissertation is the diameter. We first turn our attention to a special case of graph homomorphims, which is 3-Coloring. We first give a subexponential-time algorithm for diameter-3 graphs. Let us point out that all the algorithmic results can actually solve a more general, List 3-Coloring, where every vertex v of an input graph G is equipped with a list $L(v) \subseteq \{1, 2, 3\}$, and the desired coloring has to respect the lists.

Theorem 1.11. *The List 3-Coloring problem on n-vertex graphs with diameter at most 3 can be solved in time $2^{\mathcal{O}(n^{2/3} \cdot \log^{2/3} n)}$.*

Note that the running time bound does not depend on the maximum nor the minimum degree of the input graph. In particular, this is the *first* algorithm for List 3-Coloring, whose complexity is subexponential for *all* diameter-3 graphs.

Let us present a high-level overview of the proof. First, we partition the vertex set of the input graph G into V_1, V_2, V_3, where V_i is the set of vertices with lists of size i. If G contains a vertex $v \in V_2 \cup V_3$ with many neighbors in $V_2 \cup V_3$, then we branch on coloring the vertex v with some color c or not – note that since v has many neighbors, the lists of many vertices will be reduced. On the other hand, if all vertices of $V_2 \cup V_3$ have bounded number of neighbors in $V_2 \cup V_3$, then for some fixed v, we can branch on the coloring of the set S of vertices of $V_2 \cup V_3$ at distance at most 2 in $G[V_2 \cup V_3]$ from v. In this case the size of S is bounded and the set S *almost* dominates G (the only vertices that are not dominated by S are all contained in $V_1 \cup V_2$). Therefore, after guessing the coloring of the set S, all lists are reduced to size at most 2 and then we can finish in polynomial time, using the result of Edwards [50].

Next, we provide a faster subexponential-time algorithm for List 3-Coloring on diameter-2 graphs.

Theorem 1.12. *The List 3-Coloring problem on n-vertex graphs with diameter 2 can be solved in time $2^{\mathcal{O}(n^{1/3} \cdot \log^2 n)}$.*

Again, let us give some intuition about the proof. We partition the vertex set of G into (V_1, V_2, V_3), as previously. We aim to empty the set V_3, as then the problem can be solved

in polynomial time. We first apply three branching rules, which are slightly technical, so we do not describe them now, but in all three of them we branch on coloring a vertex or a pair of vertices with some color or not.

The main combinatorial insight that is used in our algorithm is as follows. Consider an instance (G, L), where G is of diameter 2 and none of the previous branching rules can be applied. Suppose that G has a proper 3-coloring φ that respects lists L. Then there is a color $a \in \{1, 2, 3\}$ and sets $S \subseteq V_3 \cap \varphi^{-1}(a)$ and $\widetilde{S} \subseteq V_3 \setminus \varphi^{-1}(a)$, each of size $\mathcal{O}(n^{1/3} \log n)$, with the following property:

($\star$) $S \cup \widetilde{S} \cup \Big(N(S) \cap N(\widetilde{S})\Big)$ dominates at least $\frac{1}{6}$-fraction of V_3,

where $N(S)$ (resp. $N(\widetilde{S})$) denotes the set of vertices from $V(G) \setminus S$ (resp. $V(G) \setminus \widetilde{S}$) with a neighbor in S (resp. $\widetilde{S}$). The existence of the sets S and $\widetilde{S}$ is shown using a probabilistic argument.

Now we proceed as follows. We enumerate all pairs of disjoint sets S and $\widetilde{S}$, each of size $\mathcal{O}(n^{1/3} \log n)$. If they satisfy the property ($\star$), we exhaustively guess the color a used for every vertex of S and the coloring of $\widetilde{S}$ with colors $\{1, 2, 3\} \setminus \{a\}$. Then we update the lists of the neighbors of colored vertices. Note that the color of every vertex from $N(S) \cap N(\widetilde{S})$ is now uniquely determined. Thus, for at least $\frac{1}{6}$-fraction of vertices $v \in V_3$, they are either already colored or have a colored neighbor, so their lists are of size at most 2. Therefore, our instance was significantly simplified and we can proceed recursively.

A natural question is how far the results of Theorem 1.11 and Theorem 1.12 can be generalized. We first consider a generalization of 3-Coloring, which is Weighted 3-Coloring. In Weighted 3-Coloring, the input graph G is given along with an integer k and a weight function, i.e., for every vertex $v \in V(G)$ and every color i, we have a "cost" of coloring v with i. The task is to determine whether there exists a 3-coloring of a total cost at most k. A prominent special case of this problem is Independent Odd Cycle Transversal, where we ask for a minimum-sized independent set which intersects all odd cycles. Clearly it is equivalent to solving an instance G of Weighted 3-Coloring where for every $v \in V(G)$ the cost of coloring v with 1 or 2 is 0 and the cost of coloring v with 3 is 1.

We show that the we can extend the result from Theorem 1.12 to the weighted setting.

Theorem 1.13. *The Weighted 3-Coloring problem on n-vertex diameter-2 graphs can be solved in time $2^{\mathcal{O}(n^{1/3} \log^2 n)}$.*

However, it is not possible (under the ETH) to obtain a similar strenghtening of Theorem 1.11.

Theorem 1.14. *The* Weighted 3-Coloring *problem on n-vertex diameter-3 graphs cannot be solved in time $2^{o(n)}$, unless the ETH fails.*

Other target graphs. In order to understand the peculiar case of 3-Coloring and diameter-2 graphs, we study closely related problems. As a natural direction, we consider other target graphs H – recall that 3-Coloring can be seen as Hom(K_3), where K_3 is the triangle. First, we still focus on diameter-2 input graphs, but we change the target graph to some arbitrary H, which is triangle-free. Note that it only makes sense to consider graphs H of diameter 2, since the homomorphic image of a diameter-2 graph has to induce a diameter-2 subgraph. We provide a polynomial-time algorithm for triangle-free target graphs – the algorithm works in the more general list setting. We point out that the class of triangle-free diameter-2 graphs is still very rich, for instance, contains all Mycielski graphs [113].

Theorem 1.15. *Let H be a triangle-free graph. Then the LHom(H) problem can be solved in polynomial time on diameter-2 graphs.*

Furthermore, we consider a special class of target graphs, which are odd cycles. We point out that odd cycles in homomorphism problems received considerable attention [5, 49, 67, 96, 133]. Moreover, note that the cycle on 5 vertices is the smallest graph H such that the Hom(H) problem is not equivalent to graph coloring. We first provide the following result. Again, the algorithm works in more general list setting.

Theorem 1.16. *Let $k \geqslant 2$. Then LHom(C_{2k+1}) can be solved in polynomial time on diameter-$(k+1)$ graphs.*

Note that such a result for $k = 1$ would yield a polynomial-time algorithm for 3-Coloring on diameter-2 graphs. Let us discuss the crucial points where this algorithm cannot be applied directly for $k = 1$. The first property, which holds for every cycle except C_3 and C_6, is that if for some set S of vertices, any two of them have a common neighbor, then there is a vertex that is a common neighbor of all vertices of S. Furthermore, for every cycle of length at least 5 except C_6, for a set S of 3 vertices, every vertex of S has

a private neighbor with respect to S, i.e., a neighbor that is non-adjacent to any other vertex of S.

We first show that for an instance of LHOM(C_{2k+1}), for each vertex v, we can deduce the set of vertices it can be mapped to and update the list of v so that all lists are of size at most 3. The properties discussed above allow us to encode coloring of a vertex with list of size 3 using its neighbors with lists of size two, and such a reduced instance of a slightly more general problem (we have more constraints than just the edges, but all of them are binary) can be solved in polynomial time by reduction to 2-SAT, similar to the one of Edwards [50].

Furthermore, we provide a subexponential-time algorithm for diameter-$(k+2)$ graphs.

Theorem 1.17. *Let $k \geqslant 1$. Then LHOM(C_{2k+1}) can be solved in time $\exp\left(\mathcal{O}\left((n \log n)^{\frac{k+1}{k+2}}\right)\right)$ on diameter-$(k+2)$ n-vertex graphs.*

Let us point out that the case $k = 1$ is precisely Theorem 1.11. However, we prefer to present first a simpler proof, and then move to the more complicated one. In fact the branching part of the algorithm from Theorem 1.17 is similar to the one in Theorem 1.11. The more involved part is to show that after applying braching and reduction rules we are left with an instance that can be solved in polynomial time. Similar to Theorem 1.16, we first analyze the lists of all vertices, and then reduce to an instance of more general problem where every vertex has list of size at most 2.

We complement Theorem 1.16 and Theorem 1.17 with the following NP-hardness result – since our reduction from 3-SAT is linear, we also prove that the problem cannot be solved in subexponential time under the ETH.

Theorem 1.18. *Let $k \geqslant 1$. The HOM(C_{2k+1}) problem is NP-hard on graphs of radius $k+1$ (and thus diameter $(2k+2)$) and cannot be solved in subexponential time, unless the ETH fails.*

We summarize the current state of knowledge about the complexity of HOM(C_{2k+1}) in Table 1.1. We also point out that, again, all upper bounds, i.e., Theorem 1.15, Theorem 1.16, and Theorem 1.17 hold in the more general list setting, while the lower bound from Theorem 1.18 holds already in the non-list setting.

$k \setminus$ diam	2	3	4	5	6	7	8	9	10	11	$\geqslant 12$
1	[111]	1.11	folklore								
2		1.16	1.17	5.23	1.18						
3			1.16	1.17	?	?	1.18				
4				1.16	1.17	?	?	?	1.18		
$\geqslant 5$					1.16	1.17	?	?	?	?	1.18

Table 1.1: Complexity of $\textsc{Hom}(C_{2k+1})$ on bounded-diameter graphs. The color in the cell (k, d) denotes that $\textsc{Hom}(C_{2k+1})$ on diameter-d, respectively, green – is polynomial-time solvable, blue – can be solved in subexponential time, and red – cannot be solved in subexponential time, assuming the ETH. The table is filled due to [111] and the results of this dissertation – the theorem numbers are placed in appropriate cells.

1.5.3 Organization of the dissertation

In Chapter 2 we define basic notions and problems. In Chapter 3 we present some existing results on $\textsc{Hom}(H)$ and $\textsc{LHom}(H)$, which will be useful in this dissertation. In Chapter 4, which is based on [70], we consider the $\textsc{Hom}(H)$ problem parameterized by the cutwidth of the input graph. In Chapter 5, which is based on [48] and [127], we first consider a special case of $\textsc{Hom}(H)$, which is 3-Coloring, for bounded-diameter graphs, and then we consider other target graphs H. In Chapter 6 we discuss some other results of the author, which were not selected for the dissertation.

Chapter 2

Preliminaries

By $[n]$ we denote the set $\{1, 2, \ldots, n\}$ and by $[n]_0$ we denote $\{0, 1, \ldots, n\}$. For integers a, b with $a < b$, we write $[a, b] = \{a, a+1, \ldots, b\}$. Whenever we write $\log x$, i.e., we do not specify the base of the logarithm, then we mean the natural logarithm. For a set X, by 2^X we denote the family of all subsets of X.

Throughout this paper all graphs we consider are simple, i.e., have no loops nor multiple edges. For a graph G and $V' \subseteq V(G)$ (resp. $E' \subseteq E(G)$), by $G[V']$ (resp. $G[E']$) we denote the subgraph of G induced by V' (resp. E'). We define $|G| := |V(G)|$. By $\overline{G}$ we denote the complement of G, i.e., $V(\overline{G}) = V(G)$ and $E(\overline{G}) = \binom{V(G)}{2} \setminus E(G)$. By P_n, C_n, and K_n we denote, respectively, the path on n vertices, the cycle on n vertices, and the complete graph on n vertices. By $\mathrm{dist}_G(u, v)$ we denote the length (number of edges) of a shortest u-v path in G. For a vertex v, by $N_G(v)$ we denote its *open neighborhood*, i.e., the set of all vertices adjacent to v. The *closed neighborhood* of v is defined as $N_G[v] := N_G(v) \cup \{v\}$. For an integer p, by $N_G^{\leqslant p}[v]$ we denote the set of vertices at distance at most p from v, and define $N_G^{\leqslant p}(v) := N_G^{\leqslant p}[v] \setminus \{v\}$. For a set X of vertices, we define $N_G(X) := \bigcup_{v \in X} N_G(v) \setminus X$ and $N_G[X] := N_G(X) \cup X$. For sets $X, Y \subseteq V$, we say that X *dominates* Y if $Y \subseteq N_G[X]$. By $\deg_G(v)$ we denote the *degree* of a vertex v, i.e., $|N_G(v)|$. By $\Delta(G)$ we denote the maximum vertex degree in G. If G is clear from the contex, we omit the subscript G and simply write $N(v)$, $N^{\leqslant d}(v)$, $\mathrm{dist}(u, v)$, etc. Two vertices $u, v \in V(G)$ are *incomparable* if their neighborhoods are not contained in each other, i.e., $N(u) \not\subseteq N(v)$ and $N(v) \not\subseteq N(u)$. A set of vertices is *incomparable* if all vertices in this set are pairwise incomparable. Finally, a graph G is *incomparable* if $V(G)$ is an incomparable set.

For a connected graph G, the *diameter* of G, denoted by $\mathrm{diam}(G)$, is the maximum $\mathrm{dist}(u,v)$ over all pairs of vertices $u,v \in V(G)$. We say that G is *diameter-d* graph if $\mathrm{diam}(G) \leqslant d$. The *radius* of G, denoted by $\mathrm{radius}(G)$, is the minimum integer r such that there is a vertex $z \in V(G)$ such that for every $v \in V(G)$, it holds that $\mathrm{dist}(v,z) \leqslant r$. Observe that we always have $\mathrm{radius}(G) \leqslant \mathrm{diam}(G) \leqslant 2 \cdot \mathrm{radius}(G)$.

Let G be a graph and consider a linear ordering $\sigma = (v_1, \ldots, v_n)$ of its vertices. For $i \in [n-1]$, the *i-th cut* is the partition of $V(G)$ into sets $\{v_1, \ldots, v_i\}$ and $\{v_{i+1}, \ldots, v_n\}$. The *width* of such a cut is the number of edges with one endvertex in $\{v_1, \ldots, v_i\}$ and the other in $\{v_{i+1}, \ldots, v_n\}$. The *width* of σ is the maximum width of a cut of σ. The *cutwidth* of G, denoted by $\mathrm{ctw}(G)$, is the minimum width of a linear ordering of the vertices of G.

Let H be a graph. By H^* we denote the *associated bipartite graph* defined as follows.

$$V(H^*) = \{u', u'' \mid u \in V(H)\},$$
$$E(H^*) = \{u'w'', u''w' \mid uw \in E(H)\}.$$

Colorings and homomorphisms. Let $q \in \mathbb{N}$. A *q-coloring* of a graph G is a function $c : V(G) \to [q]$. A q-coloring c of G is *proper* if for every $uv \in E(G)$, we have $c(u) \neq c(v)$. For a list function $L : V(G) \to 2^{[q]}$, we say that a q-coloring c *respects lists* L if for every $v \in V(G)$, we have $c(v) \in L(v)$. We will also call such a coloring an *L-coloring*.

For graphs G, H, a *homomorphism* from G to H is an edge-preserving mapping $\varphi : V(G) \to V(H)$, i.e., for every $uv \in E(G)$, it holds $\varphi(u)\varphi(v) \in E(H)$. For a list function $L : V(G) \to 2^{V(H)}$, a *list homomorphism* from (G, L) to H is a homomorphism that additionally respects the lists L, i.e., for every $v \in V(G)$ it holds $\varphi(v) \in L(v)$. Sometimes we will refer to L as *H-lists*. We will write $\varphi : G \to H$ (resp. $\varphi : (G, L) \to H$) if φ is a (list) homomorphism from G to H, and $G \to H$ (resp. $(G, L) \to H$) to indicate that such a (list) homomorphism exists.

Since any proper (list) q-coloring can be seen as a (list) homomorphism to K_q, we will often refer to a homomorphism as a coloring and to vertices of H as colors.

Problem definitions. We define the main computational problems of this paper.

For a fixed $q \in \mathbb{N}$, we define the q-Coloring problem.

q-Coloring
Input: Graph G.
Question: Is there a proper q-coloring of G?

We also define a list version of the problem, where each vertex of the input graph is equipped in a list of its allowed colors.

List-q-Coloring
Input: Graph G with list function $L : V(G) \to 2^{[q]}$.
Question: Is there a proper q-coloring of G, which respects lists L?

In the Weighted q-Coloring problem the instance graph G is given along with a weight function $\mathfrak{w} : V(G) \times [q] \to \mathbb{N}$ and an integer k. We might think of $\mathfrak{w}(v, i)$ as the cost of coloring v with the color i. We ask if there exists a q-coloring c of G with the total cost at most k, i.e., $\sum_{v \in V(G)} \mathfrak{w}(v, c(v)) \leqslant k$.

Weighted q-Coloring
Input: Graph G with weight function $\mathfrak{w} : V(G) \times [q] \to \mathbb{N}$, and an integer k.
Question: Is there a proper q-coloring of G such that $\sum_{v \in V(G)} \mathfrak{w}(v, c(v)) \leqslant k$?

We point out that Weighted List q-Coloring is actually equivalent to Weighted q-Coloring (see Section 5.1.3). However, sometimes it will be convenient for us to consider Weighted List q-Coloring instead of Weighted q-Coloring.

For fixed H, called *target*, in the homomorphism problem, denoted by Hom(H), we are given a graph G, called *input*, and we have to determine whether there exists a homomorphism from G to H.

Hom(H)
Input: Graph G.
Question: Is there a homomorphism $\varphi : G \to H$?

In the list homomorphism problem, denoted by LHom(H), G is given along with lists $L : V(G) \to 2^{V(H)}$, and we have to determine if there is a homomorphism φ from G to H which additionally respects lists.

LHom(H)
Input: Graph G with list function $L : V(G) \to 2^{V(H)}$.
Question: Is there a list homomorphism $\varphi : (G, L) \to H$?

We will also consider a variant of the homomorphism problem, Hom, where the target graph H is part of the input.

Hom
Input: Graphs G and H.
Question: Is there a homomorphism $\varphi : G \to H$?

Finally, we define the LHOM problem, which is the list version of HOM.

LHOM
Input: Graphs G and H, and a list function $L : V(G) \to 2^{V(H)}$.
Question: Is there a list homomorphism $\varphi : (G, L) \to H$?

For fixed integers q, r, in the Constraint Satisfaction Problem, denoted by $\mathrm{CSP}(q, r)$ problem we have a fixed set D (domain) of size q, and in the input we are given a set of variables V and set $\mathcal{C}$ of constraints which are of form $R(v_1, \ldots, v_r)$, where $R \subseteq D^r$. The task is to determine whether there exists an assignment $f : V \to D$ such that for every constraint $R(v_1 \ldots, v_r) \in \mathcal{C}$, we have $(f(v_1), \ldots, f(v_r)) \in R$.

$\mathrm{CSP}(q, r)$
Input: Set of variables V and set $\mathcal{C}$ of constraints.
Question: Does there exist an assignment $f : V \to D$ such that for every contraint $R(v_1 \ldots, v_r) \in \mathcal{C}$, we have $(f(v_1), \ldots, f(v_r)) \in R$?

Chapter 3

Graph homomorphisms – basic tools

In this chapter we present basic tools for dealing with graph homomorphisms – we introduce definitions and show useful properties. We first consider graph homomorphisms in the non-list setting and then we consider list homomorphisms.

3.1 Non-list variant

In what follows we assume that H is non-bipartite and has no vertices with loops; recall that otherwise $\textsc{Hom}(H)$ can be solved in polynomial time [78].

Disconnected graphs. Suppose that H is a disconnected graph and let $H_1, \ldots, H_\ell$ be the connected components of H. Let G be a graph with connected components $G_1, \ldots, G_p$. Observe that $G \to H$ if and only if $G_i \to H$ for every $i \in [p]$. Furthermore, for $i \in [p]$, we have $G_i \to H$ if and only if $G_i \to H_j$ for at least one $j \in [\ell]$. Therefore, in order to determine whether $G \to H$, it is sufficient to determine whether $G_i \to H_j$ for every pair $i \in [p], j \in [\ell]$.

Cores. A graph H is a *core* if it does not admit a homomorphism to any of its proper subgraphs. A *core* of H is a subgraph C of H, such that C is a core and there is a homomorphism $H \to C$. As shown by Hell and Nešetřil [79], every graph has a unique core (up to isomorphism). Thus we can talk about *the* core of a graph H and denote it by $\mathsf{core}(H)$. It is straightforward to verify that $\mathsf{core}(H)$ is actually an induced subgraph of H. Furthermore, for non-bipartite graphs H, $\mathsf{core}(H)$ is non-bipartite.

We will require the following straightforward properties of cores, see e.g. [124].

Proposition 3.1. *Let H be a core.*

(1) Every homomorphism $\varphi : H \to H$ is an automorphism.

(2) The graph H is incomparable. In particular, the neighborhoods of vertices in H are pairwise distinct.

Observe that homomorphisms are transitive: if $G_1 \to G_2$ and $G_2 \to G_3$, then $G_1 \to G_3$. Furthermore, for every graph H we have $\mathsf{core}(H) \to H$. Indeed, the function mapping each vertex of $\mathsf{core}(H)$ to itself is clearly a homomorphism. Consequently, G admits a homomorphism to H if and only if it admits a homomorphism to $\mathsf{core}(H)$. Thus, in order to solve $\textsc{Hom}(H)$, one can equivalently focus on solving $\textsc{Hom}(\mathsf{core}(H))$.

Let us point out that the problem of deciding whether a given graph H is a core is co-NP-hard [79].

One of the examples of cores are odd cycles. The following observation, which shows that we cannot map a smaller odd cycle to a larger one, will be useful for us.

Observation 3.2. *Let $k \geqslant 2$ and let G be a graph such that $G \to C_{2k+1}$. Then, for any $\ell < k$, the graph G does not contain a $(2\ell+1)$-cycle.*

Direct products. For graphs H_1, H_2, their *direct product* is the graph $H_1 \times H_2$ defined as follows

$$\begin{aligned} V(H_1 \times H_2) =& \{(v_1, v_2) \mid v_1 \in V(H_1) \text{ and } v_2 \in V(H_2)\} \\ E(H_1 \times H_2) =& \{(v_1, v_2)(u_1, u_2) \mid v_1u_1 \in E(H_1) \text{ and } v_2u_2 \in E(H_2)\}. \end{aligned}$$

Graphs H_1 and H_2 are *factors* of H and $H_1 \times H_2$ is a *factorization* of H (formally, a factorization is a sequence of factors). These definitions can be naturally generalized to more factors. A graph H with at least two vertices is *prime* if it cannot be written as a direct product of at least two graphs, each with at least two vertices. A factorization, where each factor is prime and has at least two vertices is called a *prime factorization*. For a graph $H = H_1 \times \ldots \times H_p$ and for $i \in [p]$, by π_i we will denote the *projection on ith coordinate*, i.e., the mapping $\pi_i : V(H) \to V(H_i)$ such that for $v = (v_1, \ldots, v_p) \in V(H)$, we have $\pi_i(v) = v_i$. It is straightforward to verify that a projection (on any coordinate) is always a homomorphism.

For a graph H, and for a positive integer k, by $H^{\times k}$ we will denote the direct product of k copies of H.

The following statement follows from a well-known result of McKenzie [110]; see also [74, Theorem 8.17].

Theorem 3.3 (McKenzie [110])**.** *Any connected non-bipartite core has a unique (up to reordering of factors) prime factorization. Furthermore, such a factorization can be found in polynomial time.*

The following properties of direct products are straightforward, see e.g. [124].

Proposition 3.4. *Let $H = H_1 \times \ldots \times H_\ell$.*

(1) If H is connected, then for every $i \in [\ell]$, the graph H_i is connected.

(2) If H is a core, then for every $i \in [\ell]$, the graph H_i is a core.

(3) If H is non-bipartite, then for every $i \in [\ell]$, the graph H_i is non-bipartite.

The fact why direct products play an important role in the study of graph homomorphisms is the following observation.

Observation 3.5. *Let $H = H_1 \times H_2 \times \ldots \times H_p$. Then $G \to H$ if and only if $G \to H_i$ for every $i \in [p]$.*

Proof. First, suppose that there exists $\varphi : G \to H$. For $i \in [p]$, let us define $\varphi_i = \pi_i \circ \varphi$. Let us verify that φ_i is a homomorphism. Let $uv \in E(G)$ and let $(x_1, \ldots, x_p) = \varphi(u)$ and $(y_1, \ldots, y_p) = \varphi(v)$. Then $\varphi_i(u) = x_i$ and $\varphi_i(v) = y_i$. Since φ is a homomorphism, $\varphi(u)\varphi(v) \in E(H)$, which means that $x_i y_i \in E(H_j)$ for every $i \in [p]$, and thus φ_i is indeed a homomorphism.

Now suppose that for every $i \in [p]$, there exists a homomorphism $\varphi_i : G \to H_i$. Let us define $\varphi : G \to H$ so that for $v \in V(G)$, we have $\varphi(v) = (\varphi_1(v), \ldots, \varphi_p(v))$. Let us verify that φ is a homomorphism. Let $uv \in E(G)$. For every $i \in [p]$, we have $\varphi_i(u)\varphi_i(v) \in E(H_i)$ since φ_i is a homomorphism. Then $\varphi(u)\varphi(v) = (\varphi_1(u), \ldots, \varphi_p(u))(\varphi_1(v), \ldots, \varphi_p(v)) \in E(H)$, which completes the proof. □

By Observation 3.5, we can solve the problem for each H_i independently and then return the conjunction of answers.

Now, for a graph H, let us define a set of graphs $\mathsf{atoms}(H)$ as follows.

$$\begin{aligned}\mathsf{atoms}(H) =\{&H_{i,j} \mid H_i \text{ is a connected component of } H,\\ &H_{i,1} \times \ldots \times H_{i,p}, \text{ is the prime factorization of } \mathsf{core}(H_i)\}.\end{aligned}$$

Let us point out that by Proposition 3.4 (1) and (2), every $H' \in \mathsf{atoms}(H)$ is a connected prime core. Moreover, by Proposition 3.4 (3), if H is non-bipartite, then every $H' \in \mathsf{atoms}(H)$ is non-bipartite.

By all the discussions above, we can state the following.

Observation 3.6. *Assume that we can solve every instance (G', H') of* Hom *where H' is a connected non-bipartite core in time $f(H', G') \cdot |G'| + |H'|)^{\mathcal{O}(1)}$, for some function f which is non-decreasing with respect to taking induced subgraphs of G'. For graphs G, H, let us define*

$$f^*(H, G) = \max_{H' \in \mathsf{atoms}(H)} f(H', G).$$

Then every instance (G, H) of Hom *can be solved in time $f^*(H, G) \cdot (|G| + |H|)^{\mathcal{O}(1)}$, provided that H is given along with cores of all its connected components.*

3.1.1 Projective graphs and constructions

In this section we present notion of *projectivity* and its connections to *constructions*.

Projective graphs. We first consider so-called *projective graphs*. Let H be a graph. We say that a homomorphism $\varphi : H^{\times \ell} \to H$, for some $\ell \geqslant 2$, is *idempotent* if for every $v \in V(H)$, we have $\varphi(v, \ldots, v) = v$. A graph H is *projective* if for every $\ell \geqslant 2$, every idempotent homomorphism $\varphi : H^{\times \ell} \to H$ is a projection.

Larose and Tardiff [100] provided a characterization of projective graphs using a notion of *constructions*.

Definition 3.7. Let H be a graph and let $S \subseteq V(H)$. A *construction of S* consists of:

- a graph $\mathrm{C}(S)$,
- a tuple $(x_1, x_2, \ldots, x_\ell)$ of vertices of H,
- a tuple $(y_1, y_2, \ldots, y_\ell)$ of vertices of $\mathrm{C}(S)$,

- one special vertex y_0 of $\mathrm{C}(S)$,

and the following property has to be satisfied:

$$S = \{\varphi(y_0) \mid \varphi : \mathrm{C}(S) \to H \text{ such that } \varphi(y_1) = x_1, \ldots, \varphi(y_\ell) = x_\ell\}.$$

In other words, if we map each vertex x_i to its corresponding vertex y_i, then the set of possible images of y_0 over all extensions of this partial mapping to a homomorphism from $\mathrm{C}(S)$ to H is exactly S.

The following charactarization of projective graphs will be crucial for us.

Theorem 3.8 (Larose, Tardif [100])**.** *A graph H with at least three vertices is projective if and only if every $S \subseteq V(H)$ has a construction.*

Non-projective graphs. Now we would like to be able to have a similar property for remaining graphs H. We will use methods introduced by Okrasa and Rzążewski [124]. In [124] they assume two conjectures from early 2000s.

Conjecture 3.9 ([100])**.** *Let H be connected non-bipartite core. Then H is projective if and only if H is prime.*

Assuming Conjecture 3.9, if a graph H is a connected non-bipartite core, and H is not projective, then there exist projective cores $H_1, \ldots, H_\ell$ such that $H = H_1 \times \ldots \times H_\ell$. This motivates the following definition.

Definition 3.10. Let $H = H_1 \times W$, let $S \subseteq V(H)$, and let $w \in V(W)$. A *construction of (S, w) consists of*:

- a graph $\mathrm{C}(S, w)$,
- a tuple $(x_1, x_2, \ldots, x_\ell)$ of vertices of H,
- a tuple $(y_1, y_2, \ldots, y_\ell)$ of vertices of $\mathrm{C}(S)$,
- one special vertex y_0 of $\mathrm{C}(S)$,

and the following properties have to be satisfied:

1. for every vertex $s \in S$, there exists a homomorphism $\varphi : G \to H$ such that $\varphi(y_0) = (s, w)$,

2. for $s' \in V(H_1)$, if there is $w \in V(W)$ and a homomorphism $\varphi : G \to H$ such that $\varphi(y_0) = (s', w')$, then $s' \in S$.

In order to introduce the second conjecture we need a few more definitions. Let H_1, H_2 be non-bipartite graphs, and let $H = H_1 \times H_2$. We say that a homomorphism $\varphi : H_1^{\times \ell} \times H_2 \to H_1$, for some $\ell \geqslant 2$, is H_1-*idempotent* if for every $u \in V(H_1), v \in V(H_2)$, it holds $\varphi(u, \ldots, u, v) = u$. Then H is H_1-*projective* if for every $\ell \geqslant 2$, every H_1-idempotent homomorphism $\varphi : H_1^{\times \ell} \times H_2 \to H_1$ is a projection.

Conjecture 3.11 ([98, 99])**.** *Let $H = H_1 \times H_2$ be a core such that H_1 is projective. Then H is H_1-projective.*

Now assuming that Conjecture 3.9 and Conjecture 3.11 are true, we can have an analogue of Theorem 3.8.

Theorem 3.12 ([120], Lemma 3.2.7)**.** *Let H be a connected non-bipartite core and let $H = H_1 \times \ldots \times H_\ell$ be its prime factorization. Assuming Conjecture 3.9 and Conjecture 3.11, for every $S \subseteq V(H_1)$ and $w \in V(W) = V(H_2 \times \ldots \times H_\ell)$, there exists a construction of (S, w).*

3.2 List homomorphisms

In this section we focus on the list variant of the problem, i.e., LHOM(H) – in order to prove hardness in Theorem 1.10, we will first prove it for the list version of the problem, and then, using methods described in the first part of this chapter, we will extend the results to the non-list version.

Note that in the list setting it makes sense if vertices of H have loops, but for the scope of this dissertation we will focus on simple graphs. In such a case the dichotomy provided by Feder, Hell, and Huang [57], can be formulated as follows (a *circular-arc graph* is the intersection graph of arcs on a given circle).

Theorem 3.13 (Feder, Hell, Huang [57])**.** *Let H be a simple graph. Then LHOM(H) is polynomial-time solvable if H is bipartite and its complement is a circular-arc graph, and NP-complete otherwise.*

The following property of circular-arc graphs will be useful for us.

Proposition 3.14 ([57])**.** *Let H be a bipartite graph whose complement is a circular-arc graph. Then H does not contain an induced cycle on at least* 6 *vertices.*

Similarly as in [121, 128], we will first prove hardness for bipartite target graphs, and then extend to the general case using properties of associated bipartite graphs. Recall that for a graph H, the associated bipartite graph H^* is defined as follows:

$$V(H^*) = \{u', u'' \mid u \in V(H)\},$$
$$E(H^*) = \{u'w'', u''w' \mid uw \in E(H)\}.$$

For a connected, bipartite graph H with bipartition classes X, Y, we say that (G, L) is a *consistent instance* of $\text{LHOM}(H)$, if the following conditions are satisfied:

1. G is connected and bipartite with bipartition classes X_G, Y_G,
2. $\bigcup_{v \in X_G} L(v) \subseteq X$ and $\bigcup_{v \in Y_G} L(v) \subseteq Y$.

The following proposition allows us to extend hardness results for bipartite target graphs to general target graphs.

Proposition 3.15 (Okrasa, Piecyk, Rzążewski [121])**.** *Let H be a graph, and let (G, L) be a consistent instance of $\text{LHOM}(H^*)$. Define $L' : V(G) \to 2^{V(H)}$ as $L'(v) = \{u \mid \{u', u''\} \cap L(v) \neq \emptyset\}$. Then $(G, L) \to H^*$ if and only if $(G, L') \to H$.*

Therefore, through the rest of this chapter, we will consider bipartite target graphs.

For a connected bipartite graph H with bipartition classes X, Y, we say that a subset of $V(H)$ is *one-sided* if it is contained either in X or in Y.

Decomposable graphs. Okrasa, Piecyk, and Rzążewski [121] defined the following decomposition of biparitte graphs which turned out to be useful in solving $\text{LHOM}(H)$.

Definition 3.16 (Bipartite decomposition)**.** Let H be a bipartite graph with bipartition classes X, Y. A partition of $V(H)$ into an ordered triple of sets (D, N, R) is a *bipartite decomposition* if the following conditions are satisfied (see Figure 3.1)

1. N is non-empty and separates D and R,
2. $|D \cap X| \geqslant 2$ or $|D \cap Y| \geqslant 2$,

3. N induces a biclique in H,

4. $(D \cap X) \cup (N \cap Y)$ and $(D \cap Y) \cup (N \cap X)$ induce bicliques in H.

If H admits a bipartite decomposition, we call it *decomposable*, otherwise we call it *undecomposable*.

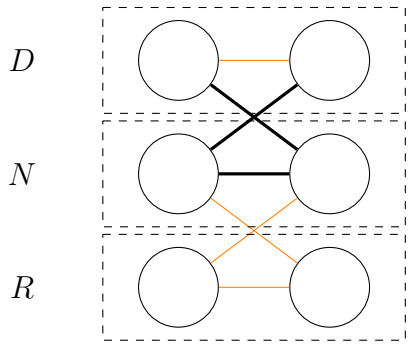

Figure 3.1: Bipartite decomposition (D, N, R). Circles denote independent sets. A black line denotes that there are all possible edges between sets, an orange one that there might be some edges, and the lack of a line denotes that there are no edges between sets. The figure is taken from [121].

We observe that the property of being incomparable is actually stronger that being undecomposable.

Proposition 3.17. *Every connected bipartite incomparable graph is undecomposable.*

Proof. Let H be a connected bipartite incomparable graph with bipartition classes X, Y, and, for contradiction, suppose that it admits a decomposition (D, N, R).

Note that the neighborhood of every vertex in $D \cap X$ (resp. $D \cap Y$) is contained in the neighborhood of any vertex in $N \cap X$ (resp. $N \cap Y$). Thus we have $D \cap X = \emptyset$ or $N \cap X = \emptyset$, and $D \cap Y = \emptyset$ or $N \cap Y = \emptyset$.

Suppose that $D \cap X = \emptyset$ (the case that $D \cap Y = \emptyset$ is symmetric). This implies that $|D \cap Y| \geqslant 2$, and all vertices in $D \cap Y$ have the same neighborhood (i.e., $N \cap Y$). So we conclude that $N \cap X = N \cap Y = \emptyset$, a contradiction with N being non-empty. □

3.2.1 Expressing relations

Throughout this section we assume that H is a connected undecomposable bipartite graph, whose complement is not a circular-arc graph, and (α, β) is a fixed pair of vertices from one bipartition class of H.

Let $k \in \mathbb{N}$, and for $i \in [k]$, let $S_i \subseteq V(H)$ be a one-sided incomparable set. Let $R \subseteq S_1 \times \ldots \times S_k$ be a k-ary relation. We say that a graph F with H-lists L and k vertices $v_1, \ldots, v_k$ is an *R-gadget* if

$$R = \{(\varphi(v_1), \ldots, \varphi(v_k)) \mid \varphi : (F, L) \to H\}.$$

In the following theorem we show that for every relation R, an R-gadget exists.

Theorem 3.18. *Let H be a connected undecomposable bipartite graph, whose complement is not a circular-arc graph. Let $k \in \mathbb{N}$ and let $R \subseteq S_1 \times \ldots \times S_k$, where for every $i \in [k]$, the set $S_i \subseteq V(H)$ is one-sided and incomparable. Then there exists an R-gadget.*

The first building block for proving Theorem 3.18 is a gadget of a k-ary relation called a $\mathrm{NAND}_k(\alpha, \beta)$.

$$\mathrm{NAND}_k(\alpha, \beta) := \{\alpha, \beta\}^k \setminus \{(\beta, \ldots, \beta)\}.$$

The second building block is gadget of a binary relation called *indicator* (S, s, α, β), where $S \subseteq V(H)$ and $s \in S$.

$$I(S, s, \alpha, \beta) := S \times \{\alpha, \beta\} \setminus \{(s, \alpha)\}.$$

For an indicator gadget $I(S, s, \alpha, \beta)$ with interface vertices (x, y), we will call x, y, respectively, *input* and *output*.

By the following lemma from [121], we can construct both, the NAND_k-gadget and the indicator gadget.

Lemma 3.19 (Okrasa, Piecyk, Rzążewski [121])**.** *Let H be an undecomposable, connected, bipartite graph, whose complement is not a circular-arc graph. Let $S \subseteq V(H)$ be an incomparable one-sided set. There exists a pair (α, β) of incomparable vertices from one bipartition class such that there exists a $\mathrm{NAND}_k(\alpha, \beta)$-gadget and for every $s \in S$ there exists an $I(S, s, \alpha, \beta)$-gadget.*

Now we are ready to prove Theorem 3.18.

Proof of Theorem 3.18. Let α, β be the pair given by Lemma 3.19. First, let us introduce the interface vertices $v_1, \ldots, v_k$ with lists $L(v_i) = S_i$. For every tuple $(s_1, \ldots, s_k) \in S_1 \times \ldots \times S_k \setminus R$ we introduce the following gadgets. For $i \in [k]$, we introduce an indicator gadget $I(S_i, s_i, \alpha, \beta)$ and identify its input vertex with v_i. Then we introduce a

$\mathrm{NAND}_k(\alpha,\beta)$-gadget and identify its interface vertices with the output vertices of indicator gadgets. This completes the construction of the R-gadget (F,L).

Let us verify that the constructed graph is indeed an R-gadget. First let $(s_1,\ldots,s_k) \in S_1 \times \ldots \times S_k \setminus R$ and suppose that there is a list homomorphism $\varphi : (F,L) \to H$ such that $\varphi(v_i) = s_i$ for every $i \in [k]$. Then for every indicator gadget $I(\alpha,\beta)$ introduced for the tuple $(s_1,\ldots,s_k)$, its output vertex must be mapped to β. These output vertices were identified with interface vertices of a $\mathrm{NAND}_k(\alpha,\beta)$-gadget, and thus cannot all be mapped to β, which is a contradiction.

So now let $(s_1,\ldots,s_k) \in R$. We set $\varphi(v_i) = s_i$ for every $i \in [k]$. It remains to show that φ can be extended to the remaining vertices of F, i.e., vertices of gadgets introduced for some tuples $(s'_1,\ldots,s'_k)$. Consider such a tuple $(s'_1,\ldots,s'_k)$, i.e., any tuple from $S_1 \times \ldots \times S_k \setminus R$. There must be $i \in [k]$ such that $s_i \neq s'_i$. Therefore the output vertex of the indicator gadget $I(S_i,s_i,\alpha,\beta)$ can be mapped to α. We can extend φ to the vertices of the remaining indicator gadgets so that their output vertices are mapped either to α or β. Since at least one of the interface vertices is not mapped to β, we can extend φ to the remaining vertices of the $\mathrm{NAND}_k(\alpha,\beta)$-gadget. This completes the proof. □

Chapter 4

Cutwidth

In this chapter as the parameter of the input graph the cutwidth. Let us first define crucial notions of this chapter.

Throughout this chapter, for a bipartite graph H, we will sometimes write $H = (X, Y, E)$, where $V(H) = X \cup Y$, $E(H) = E$, and X, Y are bipartition classes of H. In such a case we will consider H as a graph with fixed bipartition classes X, Y, which will allow us to avoid confusion, when H is not connected and bipartition classes are not uniquely defined. However, we will mostly consider connected graphs, for which bipartition classes are always uniquely defined. Finally, the values of all the parameters we will define for bipartite graphs with fixed bipartition classes, are independent on the choice of bipartition classes – we fix them only for clarity.

Induced matchings and half-induced matchings. A set $M \subseteq E$ of edges of a graph H forms an *induced matching* if the edges in M are disjoint and no edge in $E(H)$ is incident with two edges from M. We may also view this as two sequences of distinct vertices $v_1, \ldots, v_m$ and $u_1, \ldots, u_m$ where $v_i u_j \in E(H)$ if and only if $i = j$. For a bipartite graph H, by $\mathrm{mim}(H)$ we denote the size of a maximum induced matching in H. For non-bipartite H, we define $\mathrm{mim}(H) := \mathrm{mim}(H^*)$ – recall that by H^* we denote the associated bipartite graph of H.

A *half-induced matching* in a bipartite graph H with bipartition classes X, Y consists of two sequences $v_1, \ldots, v_m \in X$ and $u_1, \ldots, u_m \in Y$ of distinct vertices where $v_i u_i \in E$ for $i \in [m]$ and $u_i v_j \notin E$ if $1 \leqslant i < j \leqslant m$ (see Figure 4.1). For a bipartite graph H,

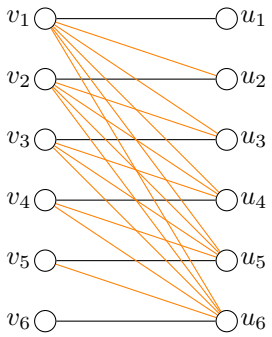

Figure 4.1: A half-induced matching of size 6. Black edges denote the edges that must be present, orange ones denote edges that might exist, and there are no other edges. If none of the orange edges exists, then we have an induced matching, and if all of them exist, then we have a halfgraph.

we denote the size of a largest half-induced matching in H by $\mathrm{him}(H)$. We extend the definition to graphs H that are non-bipartite via $\mathrm{him}(H) = \mathrm{him}(H^*)$. This notion has been studied under the name *constrained matching* (a subset with a unique matching, see e.g. [24, 125, 134]), but we decided to use the name which appeared more recently in a similar setting to ours [108], since the word 'constrained matching' has also been used for various other purposes in the algorithmic community.

Note that an induced matching is in particular a half-induced matching, and thus we always have $\mathrm{mim}(H) \leqslant \mathrm{him}(H)$. The other extremal case is a *halfgraph*, when we decide that all possible edges exist, i.e., a *halfgraph of size* m is a bipartite graph H with bipartition classes $\{v_1, \ldots, v_m\}$ and $\{u_1, \ldots, u_m\}$ and we have $u_i v_j \in E(H)$ if and only if $i \geqslant j$. Note that for a halfgraph H of size m, we have $\mathrm{him}(H) = m$ and $\mathrm{mim}(H) = 1$.

Mim and him for matrices. Let $A \in \{0,1\}^{n \times m}$ be a matrix. Given a sequence $r \in [n]^\ell$ of ℓ distinct row indices and $c \in [m]^p$ of p distinct columns indices, for some integers $\ell \in [n], p \in [m]$, we write $A[r,c]$ for the $\ell \times p$ matrix with entries $A[r,c]_{i,j} = A_{r_i,c_j}$ for $i \in [\ell]$ and $j \in [p]$. We refer to any matrix which arises in such a manner as a *permuted submatrix* of A.

We write $\mathrm{mim}(A)$ for the maximum ℓ for which A has the $\ell \times \ell$ identity matrix as permuted submatrix (equivalently, the largest permutation submatrix). We write $\mathrm{him}(A)$ for the largest ℓ for which A has an $\ell \times \ell$ triangular matrix with ones on the diagonal as permuted submatrix. We will also refer to such a submatrix as *half induced matching*. (A

matrix is called *triangular* if either all entries below the diagonal, or all entries above the diagonal are 0.)

For a bipartite graph $H = (X, Y, E)$, in the *bi-adjacency matrix* B_H rows are indexed by vertices of X and columns are indexed by the vertices of Y, and $B[u, v] = 1$ if $uv \in E$ and $B[u, v] = 0$ otherwise. For a bipartite graph H, there is a one-to-one correspondence between induced matchings of H of size m and $m \times m$ identity permuted submatrices of the bi-adjacency matrix of H. In particular, for a bi-adjacency matrix B_H of H, $\mathrm{mim}(B_H) = \mathrm{mim}(H)$. Similarly, $\mathrm{him}(B_H) = \mathrm{him}(H)$.

For a non-bipartite graph H, if A_H is its adjacency matrix, then A_H is also the bi-adjacency matrix of H^* with bipartition classes $V' = \{v' \mid v \in V(H)\}$ and $V'' = \{v'' \mid v \in V(H)\}$. This means that for non-bipartite H with adjacency matrix A_H, $\mathrm{mim}(H) = \mathrm{mim}(A_H)$ and $\mathrm{him}(H) = \mathrm{him}(A_H)$.

Mimsup. For a matrix A, we define

$$\mathrm{mimsup}(A) = \sup_k \mathrm{mim}(A^{\otimes k})^{1/k}.$$

Here $\otimes$ denotes the Kronecker product of the matrix. Given an $n \times m$ matrix $A = (a_{i,j})_{i\in[n],j\in[m]}$ and a matrix B, the Kronecker product is given by

$$A \otimes B = \begin{pmatrix} a_{1,1}B & a_{1,2}B & \dots & a_{1,m}B \\ a_{2,1}B & a_{2,2}B & \dots & a_{2,m}B \\ & \dots & & \\ a_{n,1}B & a_{n,2}B & \dots & a_{n,m}B \end{pmatrix}.$$

Moreover, by $A^{\otimes k}$ we denote the Kronecker product of k copies of A.

We point out that for every matrix $A \in \{0, 1\}^{n\times m}$ the value $\mathrm{mimsup}(A)$ is finite and is bounded by $\min\{n, m\}$. Indeed, for every $k \in \mathbb{N}$, the matrix $A^{\otimes k}$ has n^k rows and m^k columns, so it can only contain permutation submatrix with at most $\min\{n^k, m^k\}$ rows/columns, and thus $\mathrm{mim}(A^{\otimes k})^{1/k} \leqslant \min\{n, m\}$.

In fact, in the definition of mimsup, we can replace the supremum with the limit.

Theorem 4.1. *Let $A \in \{0, 1\}^{n\times m}$. Then*

$$\sup_{k\in\mathbb{N}} \mathrm{mim}(A^{\otimes k})^{1/k} = \lim_{k\to\infty} \mathrm{mim}(A^{\otimes k})^{1/k}.$$

We postpone the proof of Theorem 4.1 to Section 4.5.

For a non-bipartite graph H, with adjacency matrix A_H, we set

$$\operatorname{mimsup}(H) = \operatorname{mimsup}(A_H).$$

When $H = (X, Y, E)$ is bipartite with bi-adjacency matrix[1] B_H, $\operatorname{mimsup}(H) = \operatorname{mimsup}(B_H)$. The parameters can also be defined in purely graph theoretical terms, as we now explain.

For a bipartite graph $H = (X, Y, E)$, and for $k \in \mathbb{N}$, we define $H^{\otimes k}$ to be the graph on vertex set $X^k \cup Y^k$ where there is an edge $(x_1, \ldots, x_k)(y_1, \ldots, y_k)$ in $H^{\otimes k}$ if and only if $x_i y_i \in E(H)$ for every $i \in [k]$. With this definition of graph power, we define

$$\operatorname{mimsup}(H) = \begin{cases} \sup_{k \in \mathbb{N}} \operatorname{mim}(H^{\otimes k})^{1/k} & \text{if } H \text{ is bipartite,} \\ \operatorname{mimsup}(H^*) & \text{otherwise.} \end{cases}$$

We point out that the power $^{\otimes k}$ is very similar to taking powers with respect to the direct product $\times$. In fact for a bipartite graph $H = (X, Y, E)$, the graph $H^{\otimes k}$ is an induced subgraph of $H^{\times k}$, i.e., we consider only vertices $(z_1, \ldots, z_k) \in V(H)^k$ such that either for every $i \in [k]$, we have $z_i \in X$, or for every $i \in [k]$, we have $z_i \in Y$.

The following property of mimsup is straightforward.

Observation 4.2. *If H is an induced subgraph of G, then* $\operatorname{mimsup}(H) \leqslant \operatorname{mimsup}(G)$.

4.1 Algorithm

In this section we discuss how we can use representative sets to create fast algorithms for Hom. Let us first discuss the high-level sketch of the algorithm and the idea behind the so-called *representative sets*. Suppose we aim to solve Hom for input graphs G and H, where H is non-bipartite. We assume that G is given with a linear ordering $v_1, \ldots, v_n$ of width at most w. For an integer $i \in [n]$ we refer to $G[\{v_1, \ldots, v_i\}]$ as the *left-hand side of the graph* and

$$X_i := \{v \in \{v_1, \ldots, v_i\} \mid \exists v' \in \{v_{i+1}, \ldots, v_n\}, vv' \in E(G)\}$$

as *the left-hand side of the i-th cut.*

[1]Note that mimsup is invariant under row and column permutations. This means that the choice of bi-adjacency matrix does not affect the mimsup on matrices and thus mimsup on bipartite graphs is well-defined.

Similarly, we refer to

$$Y_i := \{v \in \{v_{i+1}, \ldots, v_n\} \mid \exists v' \in \{v_1, \ldots, v_i\}, vv' \in E(G)\}\,.$$

as *the right-hand side of the i-th cut.*

Given a 0-1 matrix M, with rows indexed by a set $\mathcal{R}$ and $\mathcal{A} \subseteq \mathcal{R}$, we are interested in knowing whether for a column c, there is a row $r \in \mathcal{A}$ with $M[r, c] = 1$. In our case,

- each row represents a coloring of the left-hand side of the cut;
- $\mathcal{A}$ contains the colorings that can be extended to the left-hand side of the (input) graph;
- each column represents a coloring of the right-hand side of the cut;
- $M[r, c] = 1$ if and only if the colorings represented by row r and column c respect the edges crossing the cut.

This makes the following definition very natural. We say that a subset $\mathcal{A}' \subseteq \mathcal{A}$ *M-represents* $\mathcal{A}$, if for any column j we have that if there is a row index $i \in \mathcal{A}$ such that $M[i, j] = 1$, then there is also $i' \in \mathcal{A}'$ such that $M[i', j] = 1$. Intuitively, this means that we do not "lose any solutions" by restricting $\mathcal{A}$ to $\mathcal{A}'$. We will also refer to $\mathcal{A}'$ as an *M-representative set* of $\mathcal{A}$. We may omit M if it is clear from context.

We remark that representing is transitive: if $\mathcal{A}''$ represents $\mathcal{A}'$ and $\mathcal{A}'$ represents $\mathcal{A}$, then $\mathcal{A}''$ represents $\mathcal{A}$.

We will be interested in representative sets with respect to $M = A_H^{\otimes k}$ for integers k, where A_H is the adjacency matrix of H. Thus, we may also refer to H-representative sets rather than A_H-representative sets.

Suppose there are k edges crossing the ith cut: $\{a_1, b_1\}, \ldots, \{a_k, b_k\} \in E(G)$ with $a_1, \ldots, a_k \in \{v_1, \ldots, v_i\}$ and $b_1, \ldots, b_k \in \{v_{i+1}, \ldots, v_n\}$. Let $L_i = (a_1, \ldots, a_k)$ and $R_i = (b_1, \ldots, b_k)$. Note that $\{a_1, \ldots, a_k\} = X_i$ but some elements may be repeated. A row r of the matrix $M = A_H^{\otimes k}$ is a k-tuple $(r_1, \ldots, r_k) \in V(H)^k$, which corresponds to a coloring $X_i \to V(H)$ if $r_j = r_{j'}$ whenever $a_j = a_{j'}$. If similarly $c \in V(H)^k$ represents a coloring of the 'right-hand side of the cut', then $M[r, c] = 1$ if and only if $a_j b_j \in E(H)$ for all $j \in [k]$, i.e., the colorings are compatible. So indeed we capture the properties informally claimed above.

The main idea behind the use of representative sets in an algorithmic setting is as follows. We solve the problem with a standard dynamic programming approach, where the cells are indexed by the elements of the set $\mathcal{A}$. A representative set then forms a small subset of these indices, which still carries enough information to solve the problem. Therefore, by reducing the current set of indices to a smaller representative set after each step, we can effectively run our dynamic programming algorithm on only a small subset of the cells in the table.

4.1.1 Connection to Mimsup

We now show that the largest size of a set $\mathcal{A} \subseteq V(H)^k$ without a smaller $A_H^{\otimes k}$-representative set, equals $\text{mimsup}(H)^k$ (for k an integer, H a non-bipartite graph and A_H its adjacency matrix).

Theorem 4.3. *Let H be a non-bipartite graph and let A_H be its adjacency matrix.*

- *For each integer $k \in \mathbb{N}$, for any $\mathcal{A} \subseteq V(H)^k$, there is a subset $\mathcal{A}' \subseteq \mathcal{A}$ of size at most $\text{mimsup}(H)^k$ that $A_H^{\otimes k}$-represents $\mathcal{A}$.*
- *Conversely, for each $\varepsilon > 0$, for each sufficiently large k, there is $\mathcal{A} \subseteq V(H)^k$, for which no $\mathcal{A}' \subseteq \mathcal{A}$ of size at most $(\text{mimsup}(H) - \varepsilon)^k$ can $A_H^{\otimes k}$-represent $\mathcal{A}$.*

Proof. Let $M = A_H^{\otimes k}$. Let $\mathcal{A}' \subseteq \mathcal{A}$ be of minimum size among the subsets that M-represent $\mathcal{A}$. Then no proper subset of it M-represents $\mathcal{A}$. This means that for each $a \in \mathcal{A}'$ it cannot be removed from $\mathcal{A}'$ to get a set that M-represents $\mathcal{A}$. Thus, for each $a \in \mathcal{A}'$ there is some $\mu(a) \in V(H)^k$ such that $M[a, \mu(a)] = 1$, but for every $a' \in \mathcal{A}' \setminus \{a\}$ we have that $M[a', \mu(a)] = 0$. Hence the rows and columns from $\{a, \mu(a) : a \in \mathcal{A}'\}$ form a permutation submatrix in $A_H^{\otimes k}$ of size $|\mathcal{A}'|$. This shows that $|\mathcal{A}'| \leqslant \text{mim}(A_H^{\otimes k})$. By definition of mimsup, $\text{mim}(A_H^{\otimes k}) \leqslant \text{mimsup}(H)^k$.

Conversely, recall that by Theorem 4.1, we can replace the supremum with the limit in the definition of mimsup. Therefore, by the definition of a limit, for each $\varepsilon > 0$ there is k_0 such that $\text{mim}(A_H^{\otimes k}) \geqslant (\text{mimsup}(A_H) - \varepsilon)^k$ for all $k \geqslant k_0$. Any permutation submatrix has no smaller representative sets, so it suffices to consider the set of rows $\mathcal{A}$ of a permutation submatrix in $A_H^{\otimes k}$ of size $\text{mim}(A_H^{\otimes k})$. □

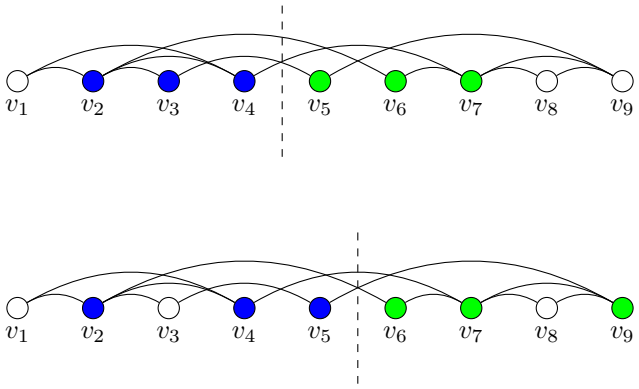

Figure 4.2: Sets X_i (blue) and Y_i (green) for $i = 4$ (above) and $i = 5$ (below).

4.1.2 Exploiting Representative Sets in Dynamic Programming

In the following theorem we formally show how to use representative sets for solving Hom. Let us emphasize that H is not assumed to be fixed here but rather given as an input.

Theorem 4.4. *Let H be a non-bipartite graph on h vertices. Let R be a reduction algorithm that, given an integer $k \geqslant 2$ and a subset $\mathcal{A} \subseteq V(H)^k$, outputs a set $\mathcal{A}'$ of size at most $\mathsf{size}(H,k)$ that $A_H^{\otimes k}$-represents $\mathcal{A}$, running in time $\mathsf{time}(|\mathcal{A}|,H,k)$. Then there exists an algorithm that, given a linear ordering of an n-vertex graph G of width w, decides whether $G \to H$ in time*

$$\mathcal{O}\Big(\big(\mathsf{size}(H,w)\cdot h + \mathsf{time}\left(\mathsf{size}(H,w)\cdot h, H, w\right)\big)n\Big).$$

Proof. We can assume that G is connected, as otherwise we can solve the problem for every connected component of G separately (see Section 3.1). Let $v_1,\ldots,v_n$ be a linear ordering of G of width w. For $i \in [n]$, by E_i we denote the set of edges that cross the i-th cut, i.e., those with one endpoint in $\{v_1,\ldots,v_i\}$ and the other in $\{v_{i+1},\ldots,v_n\}$. Recall that X_i and Y_i are defined as follows.

$$X_i := \{v \in \{v_1,\ldots,v_i\} \mid \exists v' \in \{v_{i+1},\ldots,v_n\}, vv' \in E(G)\},$$
$$Y_i := \{v \in \{v_{i+1},\ldots,v_n\} \mid \exists v' \in \{v_1,\ldots,v_i\}, vv' \in E(G)\}.$$

Note that we have $|X_i| \leqslant |E_i| \leqslant w$ and $X_1 = \{v_1\}$ (since G is connected). Let us point out that we always have $X_i \subseteq X_{i-1} \cup \{v_i\}$ (see Figure 4.2).

For a mapping $c : X_i \to V(H)$, we define the table entry $T_i[c]$ as true if there exists a homomorphism $\varphi : G[\{v_1, \ldots, v_i\}] \to V(H)$, such that for all $v \in X_i$ we have $\varphi(v) = c(v)$. (In other words, the keys are given by the H-colorings of X_i and the value of the table is true if there is an extension of the coloring to the graph induced on the left-hand side of the cut.)

This table can be easily computed in time $h^{w+1} \cdot n^{\mathcal{O}(1)}$ by the following naive dynamic programming procedure. We initiate every entry $T_i[c]$ to be false and every entry $T_1[c]$ to be true. Then, for every $i \in [2, n]$, every mapping $c' : X_{i-1} \to V(H)$, such that $T_{i-1}[c']$ is true, and every $u \in V(H)$, we check whether $c : X_{i-1} \cup \{v_i\} \to V(H)$ defined as

$$c(v) = \begin{cases} u & \text{if } v = v_i, \\ c'(v) & v \in X_{i-1}. \end{cases} \tag{4.1.1}$$

is a homomorphism from $G[X_{i-1} \cup \{v_i\}]$ to H. If so, we set $T_i[c|_{X_i}]$ to true.

We first outline why this correctly computes the table entries (that is, that at the end $T_i[c]$ is true if and only if c extends to a coloring of $G[\{v_1, \ldots, v_i\}]$) and then explain how to improve on this naive algorithm. We prove the correctness by induction on i. For $i = 1$, the homomorphism only assigns a color to v_1 and does not need to be extended (and it automatically respects the empty set of edges). Now suppose that the claim has been shown for $i = 1, \ldots j$ and let $\alpha : X_{j+1} \to V(H)$ be a coloring. If this extends to a coloring ϕ of $G[\{v_1, \ldots, v_{j+1}\}]$, then $T_j[c']$ is true for $c' = \phi|_{X_j}$ (by the induction hypothesis) and we could obtain α as a restriction of c from (4.1.1) with $u = \phi(v_{j+1})$ and $i = j + 1$. So $T_{j+1}[\alpha]$ is true. Vice versa, if $T_{j+1}[\alpha]$ has been set to true, then there is a $c' : X_{i-1} \to V(H)$ and $u \in V(H)$ such that c (again defined as in (4.1.1)) is a homomorphism $G[X_j \cup \{v_{j+1}\}] \to H$ which restricts to α on X_{j+1}. By the induction hypothesis, there exists a homomorphism ϕ' that extends c' to $G[\{v_1, \ldots, v_j\}]$ and we extend this to a homomorphism ϕ of $G[\{v_1, \ldots, v_{j+1}\}]$ by setting $\phi(v_{j+1}) = u$. Then ϕ still restricts to α and all of the edge constraints have been verified by c and/or ϕ' – let us point out that all the neighbors of v_{j+1} in $\{v_1, \ldots, v_j\}$ are contained in X_j and thus are colored by ϕ', so indeed ϕ respects the edges containing v_{j+1}. In particular, $G \to H$ if and only if $T_n[\emptyset]$ is true, where $\emptyset$ denotes the empty mapping ($X_n = \emptyset$). Since for every $i \in [n]$ the number of table entries of T_i is bounded by the number of possible mappings from X_i to H, which is h^w (recall that $|X_i| \leqslant |E_i| \leqslant w$) and we construct T_{i+1} by checking pairs: an entry of T_i and a vertex of H, the running time is indeed bounded by $h^{w+1} \cdot n^{\mathcal{O}(1)}$.

We will speed up this naive version of the dynamic program by computing a representative table T' as follows. We first set $T'_1 = T_1$. For $i = 1, 2, \ldots, n-1$ we proceed as follows. Let $k = |E_i| \leqslant w$ and $M = A_H^{\otimes k}$. Let $\{a_1, b_1\}, \ldots, \{a_k, b_k\} \in E_i$ be an enumeration of the edges, with $a_j \in \{v_1, \ldots, v_i\}$ for all $j \in [k]$. For each $c : X_i \to V(H)$ such that $T'_i[c]$ is set to true, we put the k-tuple $(c(a_1), \ldots, c(a_k))$ in $\mathcal{A}_i$. When $k \geqslant 2$, we apply the reduction algorithm R to $\mathcal{A}_i$, resulting in a set $\mathcal{A}'_i$ of size at most $\mathsf{size}(H, k)$ that $A_H^{\otimes k}$-represents $\mathcal{A}_i$. When $k = 1$, we set $\mathcal{A}'_i = \mathcal{A}_i$. We then compute the next table entries similarly as in the previous approach. Each element of $\mathcal{A}'_i$ corresponds to a coloring $c' : X_i \to V(H)$. For $u \in V(H)$, we check whether $c : X_i \cup \{v_{i+1}\} \to V(H)$ with $c(v_{i+1}) = u$ and $c|_{X_i} = c'$ is a homomorphism from $G[X_i \cup \{v_{i+1}\}]$ to H. If so, we set $T'_{i+1}[c|_{X_{i+1}}] = 1$. We repeat this for all pairs (c', u).

The procedure above is repeated for $i = 1, \ldots, n-1$, after which we return $T'_n[\emptyset]$ as the answer.

Running time. When $|\mathcal{A}'_i| \leqslant \mathsf{size}(H, k)$, we find that $|\mathcal{A}_{i+1}| \leqslant \mathsf{size}(H, k)h$ (for $k = |E_i| \leqslant w$ and $\mathsf{size}(H, k) = h$ for $k = 1$). We may assume size is a non-decreasing function on each coordinate. So the total running time is as claimed:

$$\mathcal{O}\Big(\Big(\mathsf{size}(H, w) \cdot h + \mathsf{time}\left(\mathsf{size}(H, w) \cdot h, H, w\right)\Big)n\Big).$$

Correctness. The fact that the dynamic programming steps preserve representation follows from transitivity of representation, but let us spell out the details.

Let Y_{i+1} be the set of endpoints on the right-hand side of the $(i+1)$th cut and enumerate the edges in E_{i+1} as $\{x_1, y_1\}, \ldots, \{x_k, y_k\}$, with $x_j \in X_{i+1}$ and $y_j \in Y_{i+1}$. We will show that for every $i \in [n-1]$, if $\mathcal{A}'_i$ represents the set $\mathrm{True}_i := \{(c(x_1), \ldots, c(x_k)) \mid T_i[c] = \mathrm{True}\}$, then $\mathcal{A}_{i+1}$ represents the set True_{i+1}. We started with setting $T'_1 = T_1$, so $\mathcal{A}'_1$ indeed represents True_1.

Suppose that $\mathcal{A}'_i$ represents the set True_i for some $i \in [n-1]$. We need to show that $\mathcal{A}_{i+1}$ represents the set True_{i+1}. The same then holds for $\mathcal{A}'_{i+1}$ by transitivity.

Let us first unravel the definitions to see what we need to show. Let $i \in [n-1]$ and suppose that $c : G[X_{i+1}] \to H$ extends to a homomorphism $\phi : G[\{v_1, \ldots, v_{i+1}\}] \to H$ (i.e., $T_i[c] = \mathrm{True}$). Since we want to verify representation, we will then assume there is a homomorphism $d : G[Y_{i+1}] \to H$ for which $c \cup d$ respects all edges from the $(i+1)$th cut

(those in E_{i+1}), i.e., this corresponds to a "one-entry in the compatibility matrix". What needs to be shown is that this "one-entry" can also be generated via a coloring coming from $\mathcal{A}_{i+1}$, that is, there is $\alpha : G[X_{i+1}] \to H$, such that $(\alpha(x_1), \ldots, \alpha(x_k)) \in \mathcal{A}_{i+1}$ and $(\alpha \cup d)|_{G[E_{i+1}]}$ is a homomorphism.

By assumption, $\phi \cup d$ respects all the edges with at least one endpoint in $\{v_1, \ldots, v_{i+1}\}$, and in particular those with one endpoint in $\{v_1, \ldots, v_i\}$. Since $\mathcal{A}'_i$ is a representative set of True_i, there must be $c' : G[X_i] \to H$ such that $(c'(x'_1), \ldots, c'(x'_{k'})) \in \mathcal{A}'_i$, for $\{x'_1, \ldots, x'_{k'}\} = X_i$, and where $c' \cup \phi|_{\{v_{i+1}\}} \cup d$ respects all the edges with at least one endpoint in $\{v_1, \ldots, v_i\}$. We set $\alpha = (c' \cup \phi|_{\{v_{i+1}\}})|_{X_{i+1}}$. Then $(\alpha(x_1), \ldots, \alpha(x_k)) \in \mathcal{A}_{i+1}$, by definition of how we obtain $\mathcal{A}_{i+1}$ from $\mathcal{A}'_i$. Moreover, $\alpha \cup d$ is a homomorphism $G[E_{i+1}] \to H$, as desired. □

4.2 Representative sets

In this section we focus on upper bounds for the size of representative sets and how to actually compute H-representative sets, where the size guarantee for the resulting representative sets is given in terms of two different parameters of H. The first of these two algorithms is one of main technical contributions of this chapter, and it is rather general since it finds representative sets non-trivially fast for any large Kronecker power of a matrix with small size of a maximum half-induced matching. The second of these two algorithms uses the notion of the so-called *support-rank* and is a natural generalization of the algorithm from [87]. We compare the size of a maximum half-induced matching to the support-rank in Section 4.5.2.

4.2.1 Computing representative sets via half-induced matchings

In this section we show how to compute small representative sets for graphs with no large half-induced matching. In order to do this, we will show how to find a representative set that has one fewer element, by finding some element that can be safely removed. We then use this intermediate result to find our final reduction algorithm, which will result in the following lemma.

Lemma 4.5. *Let $\ell \geqslant 1$ and $k \geqslant 2$ be integers. Let $A \in \{0,1\}^{h \times h}$ be a matrix with $\text{him}(A) < \ell$, and let $\mathcal{A} \subseteq [h]^k$. Then we can compute $\mathcal{A}' \subseteq \mathcal{A}$ that $A^{\otimes k}$-represents $\mathcal{A}$ with*

$|\mathcal{A}'| \leqslant k^{k\ell}$ *in time* $\mathcal{O}(|\mathcal{A}|^2 h^2 k^2)$.

Before we prove Lemma 4.5, let us discuss its consequences. When we combine the reduction algorithm described in Lemma 4.5 with Theorem 4.4 we find the following result.

Theorem 1.6. *For any graphs G and H, where G is given with a linear ordering of width k, in time $\mathcal{O}(k^{2k\cdot \mathrm{him}(H)} \cdot |H|^4|G|)$ one can decide whether G admits a homomorphism to H.*

We emphasize that the algorithm does not need to know the value of $\mathrm{him}(H)$. Since $\mathrm{him}(H) \leqslant \mathrm{mimsup}(H)$ (see Lemma 4.26), we immediately obtain Theorem 1.7 as a corollary from Theorem 1.6.

Theorem 1.7. *For any graphs G and H, where G is given with a linear ordering of width k, in time $\mathcal{O}(k^{2k\cdot \mathrm{mimsup}(H)} \cdot |H|^4|G|)$ one can decide whether G admits a homomorphism to H.*

Theorem 1.5 and Theorem 1.6 also easily follow from Lemma 4.5. We postpone the details of the combinatorial bound to Section 4.5.1 and for now focus on the algorithmic aspects.

Proof of Theorem 1.6. Let $h = |V(H)|$ and let A_H be the adjacency matrix of H. Recall that $\mathrm{him}(A_H) = \mathrm{him}(H)$ is always an integer. By Lemma 4.5 we have a reduction algorithm R that for every $\mathcal{A} \subseteq V(H)^k$ returns a representative set $\mathcal{A}' \subseteq \mathcal{A}$ of size at most $\mathsf{size}(H,k) \leqslant k^{k\cdot(\mathrm{him}(H)-1)}$ in time $\mathsf{time}(|\mathcal{A}|, H, k) = \mathcal{O}(|\mathcal{A}|^2 h^2 k^2)$. Then

$$\mathsf{time}(\mathsf{size}(H,k)\cdot h, H, k) = \mathcal{O}\left(k^2 \cdot k^{2k\cdot(\mathrm{him}(H)-1)} \cdot h^4\right).$$

By Theorem 4.4 we find an algorithm that decides Hom in time

$$\mathcal{O}\left((\mathsf{size}(H,k)\cdot h + \mathsf{time}(\mathsf{size}(H,k)\cdot h, H, k))|G|\right) = \mathcal{O}(k^{2k\cdot \mathrm{him}(H)} h^4 \cdot |G|).$$

This completes the proof. □

In order to prove Lemma 4.5, we will perform a recursive algorithm for which we no longer want to treat all the coordinates symmetrically. We therefore define

$$g_k(\ell_1, \ldots, \ell_k) = \binom{\sum_i \ell_i}{\ell_1, \ldots, \ell_k}.$$

When $\ell_1 = \cdots = \ell_k = \ell$, we have $g_k(\ell, \ldots, \ell) = \binom{k\ell}{\ell, \ldots, \ell} \leqslant k^{k\ell}$. The lemma will follow easily from the following more complicated statement.

Lemma 4.6. *Let $k \geqslant 2, \ell_1, \ldots, \ell_k \geqslant 1$ be integers. Let $A \in \{0,1\}^{h \times h}$ be a matrix and let $\mathcal{A} \subseteq [h]^k$ with $|\mathcal{A}| \geqslant g_k(\ell_1, \ldots, \ell_k)$. Suppose that for every $i \in [k]$, for the set of rows $\mathcal{R}_i = \{r_i \mid r \in \mathcal{A}\}$, we have $\operatorname{him}(A[\mathcal{R}_i, \cdot]) < \ell_i$. Then there exists $v \in \mathcal{A}$ such that $\mathcal{A} \setminus \{v\}$ $A^{\otimes k}$-represents $\mathcal{A}$. Moreover, v can be found in time $\mathcal{O}(\sum_{i=1}^{k} \ell_i \cdot |\mathcal{A}|hk)$.*

Proof. Note that $|\mathcal{A}| \geqslant g_k(\ell_1, \ldots, \ell_k) \geqslant 1$ for $\ell_1, \ldots, \ell_k, k \geqslant 1$, so it is non-empty.

For $i \in [k]$ and $u \in [h]$, let

$$\mathcal{A}_u^i = \{v = (v_1, \ldots, v_k) \in \mathcal{A} \mid A[v_i, u] = 0\}$$

We choose $v \in \mathcal{A}$ (arbitrarily). We then iterate over $u \in [h]$ and $i \in [k]$ to find if there is (u, i) for which

- $A[v_i, u] = 1$, and
- $|\mathcal{A}_u^i| \geqslant g_k(\ell_1, \ldots, \ell_i - 1, \ldots, \ell_k)$.

This step can be performed in time $\mathcal{O}(|\mathcal{A}|hk)$.

If we cannot find such (u, i) pair for v, then we return v as the row to be removed from $\mathcal{A}$ (and the algorithm terminates).

Otherwise, we did find (u, i). If $\ell_i = 1$, then since $\operatorname{him}(A[\mathcal{R}_i, \cdot]) < \ell_i$, we know $A[\mathcal{R}_i, \cdot]$ has all zero-entries and so $A[v_i, u] = 1$ would not have been possible. This means that $\ell_i \geqslant 2$. We apply the same process after updating $\ell_i \leftarrow \ell_i - 1$ and $\mathcal{A} \leftarrow \mathcal{A}_u^i$. Note that $v \notin \mathcal{A}_u^i$ and $\ell_i - 1 \geqslant 1$. We will show that

(1.) when v is returned, then indeed $\mathcal{A} \setminus \{v\}$ $A^{\otimes k}$-represents $\mathcal{A}$, and

(2.) when we recursively apply the algorithm, the conditions of the lemma are again satisfied, for which it remains to show that $\operatorname{him}(A[\mathcal{R}_i', \cdot]) < \ell_i - 1$ for $\mathcal{R}_i' = \{r_i \mid r \in \mathcal{A}_u^i\}$.

Since we reduce $\sum_{i=1}^{k} \ell_i$ by one in each recursive call, the algorithm will terminate. Moreover, the number of recursive calls is at most $\sum_{i=1}^{k} \ell_i$. This shows that assuming (1.) and (2.), the time complexity is as stated.

Correctness. We first show (1.): if the algorithm outputs v, then indeed it can be removed. Note that when for some subset $\mathcal{A}' \subseteq \mathcal{A}$, it is the case that $\mathcal{A}' \setminus \{v\}$ represents $\mathcal{A}'$, then

$$\mathcal{A}' \setminus \{v\} \cup (\mathcal{A} \setminus \mathcal{A}') = \mathcal{A} \setminus \{v\}$$

will also represent $\mathcal{A}$. This means we only have to check the claims in the "base case", i.e., it is sufficient to show that if the algorithm outputs v at some step, then $\mathcal{A} \setminus \{v\}$ represents $\mathcal{A}$. Suppose towards a contradiction that we wrongly outputted $v \in \mathcal{A}$, so

- there exists $u = (u_1, \ldots, u_k) \in [h]^k$ such that $A^{\otimes k}[v, u] = 1$ yet $A^{\otimes k}[v', u] = 0$ for all $v' \in \mathcal{A} \setminus \{v\}$ (since we wrongly outputted v, there needs to be a reason why we could not remove it),
- for this u, for all $i \in [k]$, $|\mathcal{A}^i_{u_i}| < g_k(\ell_1, \ldots, \ell_i - 1, \ldots, \ell_k)$ (else the algorithm would have recursed instead of outputting v).

The fact that $A^{\otimes k}[v', u] = 0$ in the first condition means that each $v' \in \mathcal{A} \setminus \{v\}$ is an element of $\mathcal{A}^i_{u_i}$ for some $i \in [k]$. In particular,

$$|\mathcal{A} \setminus \{v\}| \leqslant \sum_{i=1}^{k} |\mathcal{A}^i_{u_i}| \leqslant \sum_{i=1}^{k} g_k(\ell_1, \ldots, \ell_i - 1, \ldots, \ell_k) - k = g_k(\ell_1, \ldots, \ell_k) - k,$$

which contradicts the assumptions of the lemma since $k \geqslant 2$.

We now prove (2.): the conditions of the lemma are satisfied when we recurse. By assumption, $\ell_i \geqslant 1$ for all i and the new $\mathcal{A}$ is sufficiently large. Moreover, him can only decrease when taking submatrices, so indeed we only need to show that $\mathrm{him}(A[\mathcal{R}'_i, \cdot]) < \ell_i - 1$ for $\mathcal{R}'_i = \{r_i \mid r \in \mathcal{A}^i_{u_i}\}$. If there is a half-induced matching of size $\ell_i - 1$, induced on rows $w_1, \ldots, w_{\ell_i - 1} \in \mathcal{R}'_i$ and columns $z_1, \ldots, z_{\ell_i - 1}$, then there is a half-induced matching of size ℓ_i in $A[\mathcal{R}_i, \cdot]$ by considering rows $w_1, \ldots, w_{\ell_i - 1}, v_i \in \mathcal{R}_i$ and columns $z_1, \ldots, z_{\ell - 1}, u$. But by assumption this does not exist, so indeed $\mathrm{him}(A[\mathcal{R}'_i, \cdot]) < \ell_i - 1$. This completes the proof. □

Now we are ready to prove Lemma 4.5.

Proof of Lemma 4.5. Suppose that $|\mathcal{A}| \geqslant g_k(\ell, \ldots, \ell)$. For $i \in [k]$, set $\mathcal{R}_i = \{r_i \mid r \in \mathcal{A}\}$. Then $\mathrm{him}(A[\mathcal{R}_i, \cdot]) < \ell$ for each i. By Lemma 4.6 we can find a row v in $\mathcal{A}$ such that $\mathcal{A} \setminus \{v\}$ $A^{\otimes k}$-represents $\mathcal{A}$ in time $\mathcal{O}(\ell k \cdot |\mathcal{A}| hk) = \mathcal{O}(|\mathcal{A}| h^2 k^2)$, where we use that $\ell \leqslant h$.

We repeat this at most $|\mathcal{A}| - g_k(\ell, \ldots, \ell)$ times until we find the desired representative set in time $\mathcal{O}(|\mathcal{A}|^2 h^2 k^2)$. □

4.2.2 Computing representative sets via support rank

The *support-rank* (called also 'non-deterministic rank' in [41], see also [114]) of a matrix $M \in \{0,1\}^{n\times m}$ over a field $\mathbb{F}$ is defined as

$$\text{support-rank}(M) = \min\{\text{rank}_{\mathbb{F}}(M') \mid M' \in \mathbb{F}^{n\times m} \text{ and } M[i,j] = 0 \iff M'[i,j] = 0\}.$$

The algorithm of Jansen and Nederlof [87] that solves every instance (G, L) of List q-Coloring in time $2^{\omega k} \cdot |G|^{\mathcal{O}(1)}$ provided that G is given with a linear ordering of its vertices of width k builds on the fact that support-rank of the adjacency matrix of K_q is at most 2, since the sum of two rank-one matrices

$$\begin{pmatrix} 1 & 1 & \dots & 1 \\ 2 & 2 & \dots & 2 \\ \vdots & & \ddots & \vdots \\ n & n & \dots & n \end{pmatrix} + \begin{pmatrix} -1 & -2 & \dots & -n \\ -1 & -2 & \dots & -n \\ \vdots & & \ddots & \vdots \\ -1 & -2 & \dots & -n \end{pmatrix}$$

has the same non-zero entries (and we always have $\text{rank}(A+B) \leqslant \text{rank}(A) + \text{rank}(B)$). Small support-rank allows representative sets to be computed efficiently using row elimination. We record here what a generalization of this approach would give for our setting.

The following lemma is a generalization of the approach of Jansen and Nederlof [87].

Lemma 4.7. *Let $\mathbb{F}$ be any field.*[2] *Let $A \in \{0,1\}^{h\times h}$ be a matrix and let $B \in \mathbb{F}^{h\times h}$ be a matrix with the same support as A (i.e., $B[i,j] = 0 \iff A[i,j] = 0$). For any $\ell \in \mathbb{N}$ and set of rows $\mathcal{R} \subseteq [h]^\ell$ of $A^{\otimes\ell}$, we can find an $A^{\otimes\ell}$-representative set $\mathcal{R}'$ for $\mathcal{R}$ of size at most $\text{rank}(B)^\ell$ in time $\mathcal{O}(|\mathcal{R}|\text{rank}(B)^{\ell\cdot(\omega-1)} \cdot \ell + h^3)$.*

Here by $\omega \leqslant 2.371552$ we denote the matrix multiplication exponent [135].

Before we prove Lemma 4.7, we present some well-known facts from linear algebra that will be crucial for us. Let $\mathbb{F}$ be any field.

(L1) Let $A \in \mathbb{F}^{n\times n}$ be a matrix and let $r = \text{rank}_{\mathbb{F}}(A)$. Using Gaussian elimination, in time $\mathcal{O}(n^3)$ we can find matrices $L \in \mathbb{F}^{n\times r}$ and $R \in \mathbb{F}^{r\times n}$ such that $A = LR$.

(L2) Let $A \in \mathbb{F}^{n\times m}$ be such that $A = LR$ for some matrices L, R. Suppose that $\mathcal{B} \subseteq [n]$ is a row basis for L. Then $\mathcal{B}$ is also a row basis for A.

[2] We assume that all arithmetic operations over $\mathbb{F}$ are performed in constant time.

(L3) Let $A \in \mathbb{F}^{n_1 \times m_1}$, $B \in \mathbb{F}^{n_2 \times m_2}$. Then $\operatorname{rank}_{\mathbb{F}}(A \otimes B) = \operatorname{rank}_{\mathbb{F}}(A) \cdot \operatorname{rank}_{\mathbb{F}}(B)$. Moreover, if $\mathcal{B}_1 \subseteq [n_1]$ is a row basis for A and $\mathcal{B}_2 \subseteq [n_2]$ is a row basis for B, then $\mathcal{B} = \mathcal{B}_1 \times \mathcal{B}_2$ is a row basis for $A \otimes B$.

(L4) Let $A \in \mathbb{F}^{n_1 \times m_1}$, $B \in \mathbb{F}^{n_2 \times m_2}$. Then $\operatorname{rank}_{\mathbb{F}}(AB) \leqslant \min\{\operatorname{rank}_{\mathbb{F}}(A), \operatorname{rank}_{\mathbb{F}}(B)\}$.

(L5) Let $A \in \mathbb{F}^{n_1 \times m_1}$, $B \in \mathbb{F}^{n_2 \times m_2}$, and let $k \in \mathbb{N}$. Then $(AB)^{\otimes k} = A^{\otimes k} B^{\otimes k}$.

The next fact follows from [9, Lemma 3.15]. Let us point out that the lemma in the paper is written in the setting that $\mathbb{F} = \mathbb{F}_2$, but this assumption is actually unnecessary.

(L6) A row basis for an $n \times m$ matrix with $m \leqslant n$ and entries in $\mathbb{F}$ can be computed in time $\mathcal{O}(nm^{\omega-1})$.

Now we are ready to prove Lemma 4.7.

Proof of Lemma 4.7. We may assume that $|\mathcal{R}| > \operatorname{rank}(B)^\ell$, since otherwise we are done. We wish to compute a row basis $\mathcal{R}'$ for the matrix $B^{\otimes \ell}[\mathcal{R}, \cdot]$, but that matrix has too many columns. By (L1), B can be expressed as $B = LR$ for L of dimensions $h \times \operatorname{rank}(B)$ and R of dimensions $\operatorname{rank}(B) \times h$ in time $\mathcal{O}(h^3)$ by Gaussian elimination. Then by (L5), $B^{\otimes \ell} = L^{\otimes \ell} R^{\otimes \ell}$ and so, by (L2), a row basis for $L^{\otimes \ell}[\mathcal{R}, \cdot]$ is also a row basis for $B^{\otimes \ell}[\mathcal{R}, \cdot]$. Note that we also cannot permit ourselves to compute $L^{\otimes \ell}$ since it has too many rows again. However, we can compute an entry $L^{\otimes \ell}[x, y] = L[x_1, y_1] L[x_2, y_2] \cdots L[x_\ell, y_\ell]$. This allows us to compute the $|\mathcal{R}| \times \operatorname{rank}(B)^\ell$ matrix $L' = L^{\otimes \ell}[\mathcal{R}, \cdot]$ in time $\mathcal{O}(|\mathcal{R}| \operatorname{rank}(B)^\ell \cdot \ell)$. Since $|\mathcal{R}| > \operatorname{rank}(B)^\ell$, we can now compute a row basis $\mathcal{R}'$ for L' in time $\mathcal{O}(|\mathcal{R}| \operatorname{rank}(B)^{\ell \cdot (\omega-1)})$ by (L6). This is then also a row basis for $B^{\otimes \ell}[\mathcal{R}, \cdot]$.

We claim such a row basis forms the desired representative set. Suppose that $A^{\otimes \ell}[r, c] = 1$ for some row $r \in \mathcal{R}$ and column $c \in [h]^\ell$ of $A^{\otimes \ell}$. We need to prove that $A^{\otimes \ell}[r', c] = 1$ for some $r' \in \mathcal{R}'$. Since $\mathcal{R}'$ is a row basis, there exist coefficients $a_{r'} \in \mathbb{F}$ such that

$$B^{\otimes \ell}[r, c] = \sum_{r' \in \mathcal{R}} a_{r'} B^{\otimes \ell}[r', c].$$

Combined with the fact that B has the same support as A, we find

$$A^{\otimes \ell}[r, c] \neq 0 \implies B^{\otimes \ell}[r, c] \neq 0 \implies B^{\otimes \ell}[r', c] \neq 0 \text{ for some } r' \implies A^{\otimes \ell}[r', c] \neq 0 \text{ for some } r'.$$

This shows that indeed $\mathcal{R}'$ is a representative set for $\mathcal{R}$. Moreover, the size of the row basis is at most the rank of $B^{\otimes \ell}$, which by (L3), is $\operatorname{rank}(B)^\ell$ as claimed. □

Combining Lemma 4.7 with Theorem 4.4, we immediately obtain the following.

Theorem 1.8. *Let H be a non-bipartite graph on h vertices. Suppose we are given an $h \times h$ matrix over a field $\mathbb{F}$ with the same support as the adjacency matrix of H and rank r. Then there exists an algorithm that, given a linear ordering of an n-vertex graph G of width k, decides whether $G \to H$ in time $\mathcal{O}\Big((r^{k\cdot\omega}hk + h^3)|G|\Big)$.*

We saw in the previous section that there is always an $A^{\otimes k}$-representative set of size at most $\mathrm{mimsup}(A)^k$. The size constraint of Lemma 4.7 is possibly worse, but in this case we can guarantee that the representative sets can also be computed efficiently. We remark that Lemma 4.7 in particular implies that the support-rank is an upper bound on mimsup. This can also be seen directly and we record this fact in the following observation.

Observation 4.8. *Let $A \in \{0,1\}^{n\times n}$ be a matrix and let $\mathbb{F}$ be a field. Let $B \in \mathbb{F}^{n\times n}$ be a matrix with the same support as A. Then $\mathrm{mim}(A^{\otimes k}) \leqslant \mathrm{rank}_{\mathbb{F}}(B^{\otimes k}) = \mathrm{rank}_{\mathbb{F}}(B)^k$ for every $k \in \mathbb{N}$. In particular,*

$$\mathrm{mimsup}(A) \leqslant \text{support-rank}(A).$$

We do not know whether mimsup and support-rank are functionally equivalent, but we do provide a separation between support-rank and him in Section 4.5.2.

4.2.3 Bounding support rank via local biclique covers

The caveat in Theorem 1.8 is that a small-rank matrix with the same support as the adjacency matrix of H must be given. If $\mathbb{F}$ is a finite field, an optimal such matrix can be found in time $|\mathbb{F}|^{h^2} \cdot h^{\mathcal{O}(1)}$ by brute-force, which is constant if both $|\mathbb{F}|$ and h are constants. We will now present a combinatorial approach for finding a small-rank matrix with the same support, which does not necessarily achieve the support-rank, but can be computed efficiently.

Let F be a bipartite graph with bipartition classes X, Y. By F^c we denote the *bipartite complement* of F, i.e., the bipartite graph with bipartition classes X, Y, where $uv \in X \times Y$ is an edge if and only if $uv \notin E(F)$.

For a bipartite graph F, let $\mathcal{B} = \{B_1, \ldots, B_s\}$ be a family of subgraphs of F, such that (i) each B_i is a biclique, (ii) $\bigcup_{i=1}^{s} E(B_i) = E(F)$, and (iii) every $v \in V(F)$ is in at most r bicliques of $\mathcal{B}$. Then we say that $\mathcal{B}$ *r-covers* F. The minimum r for which there exists a

family that r-covers F, has been studied under names *bipartite degree*, *local biclique cover number* [43, 60], and is as special case of so-called *local covering numbers* also studied in the literature [17, 91].

Lemma 4.9. *Let H be a non-bipartite graph and assume we are given a family $\mathcal{B}$ of bicliques that r-covers the bipartite complement $(H^*)^c$ of H^*. Then we can compute a matrix A'_H with the same support as the adjacency matrix A_H of H with $\operatorname{rank}_{\mathbb{R}}(A'_H) \leqslant (r+1)^r$ in time $\mathcal{O}(h^2r^2)$. In particular the support rank of A_H is at most $(r+1)^r$.*

Proof. Define an arbitrary ordering $B_1, B_2, \ldots, B_s$ of the elements of $\mathcal{B}$. Moreover, for every vertex of $V(H^*)$, we fix the ordering of bicliques containing it. For each $v \in V(H)$ and $i \in [r]$, define $\sigma_i(v) = p$ (resp. $\delta_i(v) = p$) if the i-th biclique covering v' (resp., v'') is B_p. If v' (resp., v'') is covered by $r' < r$ bicliques, then for $i = r'+1, \ldots, r$, we define $\sigma_i(v) = s+1$ (resp., $\delta_i(v) = s+2$). For $u, v \in V(H)$, we define:

$$A'_H[u,v] = \prod_{i=1}^{r} \prod_{j=1}^{r} (\sigma_i(u) - \delta_j(v)).$$

Observe that such a product is non-zero if and only if there is no biclique from $\mathcal{B}$ that contains both u', v'', and this in turn happens if and only if $u'v'' \in E(H^*)$ which is equivalent to $uv \in E(H)$. Therefore, A'_H has the same support as A_H. We claim that the rank of A'_H is at most $(r+1)^r$.

In order to bound the rank of A'_H we can rewrite

$$\begin{aligned} A'_H[u,v] &= \prod_{i=1}^{r} \Big(\sum_{\ell=0}^{r} \sigma_i(u)^\ell \cdot \sum_{J \subseteq [r] \,:\, |J| = r-\ell} \prod_{j \in J} (-\delta_j(v)) \Big) \\ &= \sum_{(\ell_1,\ldots,\ell_r) \,:\, \ell_i \in [r]_0} \prod_{i=1}^{r} \sigma_i(u)^{\ell_i} \cdot \prod_{i=1}^{r} \sum_{J_i \subseteq [r] \,:\, |J_i| = r-\ell_i} \prod_{j_i \in J_i} (-\delta_{j_i}(v)). \end{aligned}$$

If for $i \in [r], \ell_i \in [r]_0$ we define

$$\begin{aligned} L[u, (\ell_1, \ldots, \ell_r)] &= \prod_{i=1}^{r} \sigma_i(u)^{\ell_i}, \\ R[(\ell_1, \ldots, \ell_r), v] &= \prod_{i=1}^{r} \sum_{J_i \subseteq [r] \,:\, |J_i| = r-\ell_i} \prod_{j_i \in J_i} (-\delta_{j_i}(v)), \end{aligned}$$

then we see that A'_H is the product of two matrices L, R such that number of columns of L and number of rows of R is $(r+1)^r$. Therefore both L, R have rank at most $(r+1)^r$ and by (L4), we conclude that $\operatorname{rank}(A'_H) \leqslant (r+1)^r$, which completes the proof. □

For a non-bipartite graph H, let $\mathrm{cov}(H)$ denote the minimum r for which there exists a family that r-covers $(H^*)^c$. Note that if H is assumed to be fixed, i.e., in the setting of the $\textsc{Hom}(H)$ problem, the value of $\mathrm{cov}(H)$ and the actual covering family can be computed in constant time by brute force. Thus, combining Lemma 4.9 and Theorem 1.8 we obtain the following.

Theorem 1.9. *Let H be a fixed non-bipartite graph and let $r = \mathrm{cov}(H)$. The $\textsc{Hom}(H)$ problem on n-vertex instances given with a linear ordering of width k can be solved in time $\mathcal{O}\Big((r+1)^{rk\cdot\omega}n^2\Big)$.*

4.3 Prime factorizations and algorithms

In this section we will discuss how to improve the algorithms from Section 4.2 using methods discussed in Chapter 3.

First, let us define refinements of the parameters him and mimsup. Recall that we defined:

$$\begin{aligned}\mathsf{atoms}(H) =\{&H_{i,j} \mid H_i \text{ is a connected component of } H,\\ &H_{i,1} \times \ldots \times H_{i,p}, \text{ is the prime factorization of } \mathsf{core}(H_i)\}.\end{aligned}$$

So we can define:

$$\begin{aligned}\mathrm{him}^*(H) &= \max_{H' \in \mathsf{atoms}(H)} \mathrm{him}(H')\\ \mathrm{mimsup}^*(H) &= \max_{H' \in \mathsf{atoms}(H)} \mathrm{mimsup}(H').\end{aligned}$$

Examples. Let us show that the parameters him and mimsup can be arbitrarily larger than him* and mimsup*. As the first example consider a graph H_1 on $3h$ vertices which is a collection of h disjoint triangles. Since H_1 is non-bipartite, $\mathrm{him}(H_1) = \mathrm{him}(H_1^*)$ and $\mathrm{mimsup}(H_1) = \mathrm{mimsup}(H_1^*)$. It is easy to verify that H_1^* is a disjoint union of h copies of C_6. Then, the maximum induced matching of H_1^* is of size $2h$ (we can take two edges from each C_6), and thus $\mathrm{mimsup}(H_1) \geqslant \mathrm{him}(H_1) \geqslant 2h$. On the other hand, the set $\mathsf{atoms}(H_1)$ constists only of a triangle (K_3 is a prime core), and thus it can be easily verified that $\mathrm{him}^*(H_1) = \mathrm{mimsup}^*(H_1) = 2$ (the equality $\mathrm{him}^*(H_1) = 2$ is straightforward and $\mathrm{mimsup}^*(H_1) = 2$ follows from the fact that mimsup is lowerbounded by him and upperbounded by the support-rank of the adjacency matrix).

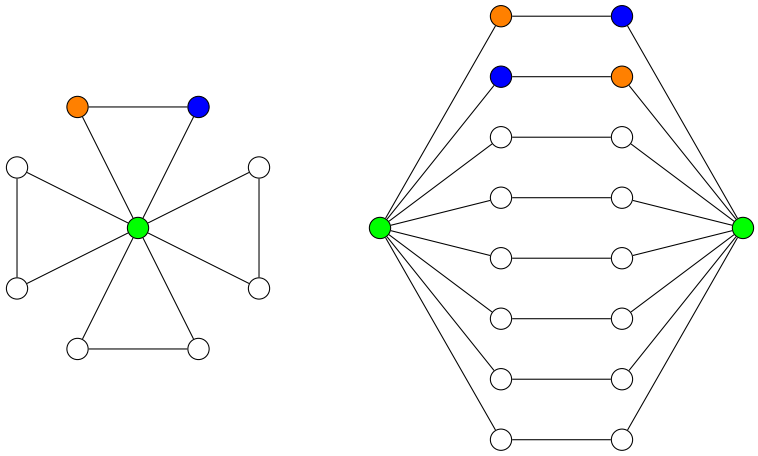

Figure 4.3: The graph H_2 for $h = 4$ (left) and H_2^* (right). The colors of vertices indicate the correspondence between the vertices in H_2 and H_2^*.

As the second example, consider a graph H_2 obtained from h triangles by identifying one vertex from each triangle into one vertex (see Figure 4.3). Again, $\mathrm{him}(H_2) = \mathrm{him}(H_2^*)$ and $\mathrm{mimsup}(H_2) = \mathrm{mimsup}(H_2^*)$. It can be verified that H_2^* contains an induced matching of size $2h$ and thus $\mathrm{mimsup}(H_2) \geqslant \mathrm{him}(H_2) \geqslant 2h$. On the other hand, the core of H_2 is the triangle and thus $\mathrm{mimsup}^*(H_2) = \mathrm{him}^*(H_2) = 2$.

Improved algorithms. By Observation 3.6, we can state the following strenghtenings of Theorems 1.6 and 1.7, where we replace him and mimsup with him^* and mimsup^*, respectively.

Theorem 4.10. *The* Hom *problem on an instance (G, H), where G is given with a linear ordering of width k and H is given along with cores of all its connected components, can be solved in time $\mathcal{O}(k^{2k\cdot\mathrm{him}^*(H)} \cdot |H|^4|G|)$.*

Theorem 4.11. *The* Hom *problem on an instance (G, H), where G is given with a linear ordering of width k and H is given along with cores of all its connected components, can be solved in time $\mathcal{O}(k^{2k\cdot\mathrm{mimsup}^*(H)} \cdot |H|^4|G|)$.*

Similarly, we can strenghten Theorem 1.8 and Theorem 1.9.

Theorem 4.12. *Let H be a non-bipartite graph on h vertices. Suppose that for every $H' \in \mathsf{atoms}(H')$ on h' vertices we are given an $h' \times h'$ matrix over a field $\mathbb{F}$ with the same*

support as the adjacency matrix of H' and rank at most r. Then there exists an algorithm that, given a linear ordering of an n-vertex graph G of width k, decides whether $G \to H$ in time $\mathcal{O}\Big((r^{k\cdot\omega}hk + h^3)|G|\Big)$.

Theorem 4.13. *Let H be a fixed non-bipartite graph and let $r = \max_{H' \in \mathsf{atoms}(H)} \mathrm{cov}(H')$. The Hom($H$) problem on n-vertex instances given with a linear ordering of width k can be solved in time $\mathcal{O}\Big((r+1)^{rk\cdot\omega}n^2\Big)$.*

4.4 Lower bound

We prove lower bounds for Hom(H) parameterized by cutwidth of the input graph G in three steps: (i) first we prove lower bounds for the list version of the problem, i.e., LHom(H), for bipartite target graphs H, (ii) then we extend the results to every target graph H, and (iii) by using some gadgets that can imitate the lists, we reduce from LHom(H) to Hom(H).

4.4.1 List homomorphisms and bipartite target graphs

The main technical contribution is the following lower bound.

Theorem 4.14. *Let $\mathcal{H}_0$ denote the set of connected incomparable bipartite graphs whose complement is not a circular-arc graph.*

(1.) Assuming the ETH, there exists $\delta > 0$ such that for every $H \in \mathcal{H}_0$ the following holds. There is no algorithm that solves every consistent instance (G, L) of LHom(H), given with a linear ordering of $V(G)$ of width t, in time $\mathrm{mimsup}(H)^{\delta\cdot t} \cdot |G|^{\mathcal{O}(1)}$.

(2.) Assuming the SETH, for every $\varepsilon > 0$ and $H \in \mathcal{H}_0$ the following holds. There is no algorithm that solves every consistent instance (G, L) of LHom(H), given with a linear ordering of $V(G)$ of width t, in time $(\mathrm{mimsup}(H) - \varepsilon)^t \cdot |G|^{\mathcal{O}(1)}$.

4.4.2 Gadgets

We start with introducing two gadgets which will be basic building blocks in the hardness proof of Theorem 1.10. We define both gadgets as R-gadgets for some relation R.

Definition 4.15 (Assignment relation)**.** Let H be a connected undecomposable bipartite graph whose complement is not a circular-arc graph. Let S be a one-sided set in H, let $v \in S$, and let (α, β, γ) be a triple of vertices from one bipartition class of H. We define a binary relation on $V(H)$ as:

$$\mathsf{Assign}(S, v, \alpha, \beta, \gamma) = S \times \{\alpha, \beta, \gamma\} \setminus \{(u, \gamma) \mid u \neq v\}.$$

For an $\mathsf{Assign}(S, v, \alpha, \beta, \gamma)$-gadget, we will call its interface vertices as the x-vertex and the y-vertex. The definition of $\mathsf{Assign}(S, v, \alpha, \beta, \gamma)$ implies that if the y-vertex is mapped to γ, then the x-vertex has to be mapped to v.

The second building block is a ternary relation is called a *switching relation*.

Definition 4.16 (Switching relation)**.** Let H be a connected undecomposable bipartite graph whose complement is not a circular-arc graph. Let (α, β, γ) be a triple of vertices from one bipartition class of H. We define a ternary relation

$$\mathrm{Switch}(\alpha, \beta, \gamma) = \{\alpha, \beta\} \times \{\alpha, \beta, \gamma\} \times \{\alpha, \beta\} \setminus \{(\alpha, \alpha, \beta), (\alpha, \beta, \beta)\}.$$

For a $\mathrm{Switch}(\alpha, \beta, \gamma)$-gadget, we will call its interface vertices, respectively, the *p-vertex*, the *q-vertex*, and the *r-vertex*. Mapping both p- and r-vertex to the same vertex, i.e., mapping both to α or both to β, or mapping the p-vertex to β and the r-vertex to α allows us to map the q-vertex to one of α, β, but "switching sides" from α to β forces mapping the q-vertex to γ.

The existence of both gadgets follows from Theorem 3.18, but we also point out that these gadgets (with some additional properties) were constructed in [128].

4.4.3 Reduction

Now we will use the introduced gadgets to reduce the *Constraint Satisfaction Problem* to LHOM(H). Recall that for fixed integers q, r, in the Constraint Satisfaction Problem, denoted by CSP(q, r) problem we have a fixed set D (domain) of size q, and in the input we are given a set of variables V and set $\mathcal{C}$ of constraints which are of form $R(v_1, \ldots, v_r)$, where $R \subseteq D^r$. The task is to determine whether there exists an assignment $f : V \to D$ such that for every constraint $R(v_1 \ldots, v_r) \in \mathcal{C}$, we have $(f(v_1), \ldots, f(v_r)) \in R$.

The construction in the following lemma is a refinement of the construction from [128].

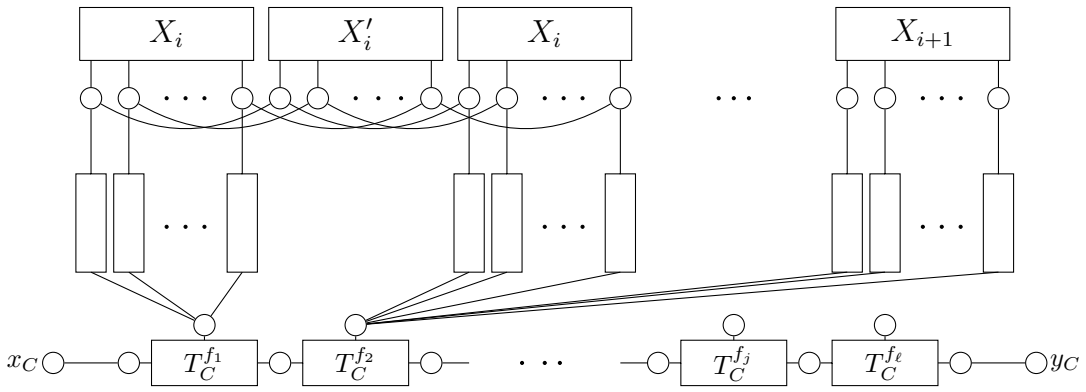

Figure 4.4: The path P_C for a constraint C and variable gadgets connected to P_C with assignment gadgets.

Lemma 4.17. *Let $q, r, k \in \mathbb{N}$ and let $H \in \mathcal{H}_0$ be such that $\mathrm{mim}(H^{\otimes k}) \geqslant q$. Let $(V, \mathcal{C})$ be an instance of CSP(q, r) with n variables and m constraints. In time polynomial in $(n+m)$ we can construct a graph G with H-lists L and with a linear ordering σ of $V(G)$, such that:*

(1.) $(V, \mathcal{C})$ is a yes-instance of CSP(q, r) if and only if $(G, L) \to H$,

(2.) the width of σ is at most $k \cdot n + r \cdot g(k, H)$, where g is some function depending only on H and k,

(3.) $|G| = (n+m)^{\mathcal{O}(1)}$.

Proof. Recall that by Proposition 3.17, H is undecomposable, so it satisfies the assumptions of all the lemmas in Section 3.2. Let (α, β, γ) be a triple of incomparable vertices from one bipartition class of H, and such that there exist $\alpha' \in N(\alpha) \setminus N(\beta)$ and $\beta' \in N(\beta) \setminus N(\alpha)$ – for a bipartite graph H whose complement is not a circular-arc graph there always exists such a triple, see [121]. Let D be the domain of the instance $(V, \mathcal{C})$. We construct (G, L) as follows.

Constraint gadgets. For each constraint $C \in \mathcal{C}$, we construct a *constraint gadget* P_C with H-lists L, as follows. We start with vertices x_C and y_C with lists $L(x_C) = \{\alpha'\}$ and $L(y_C) = \{\beta'\}$. Then for every assignment f of the variables in C that satisfies C we introduce a switching gadget, i.e., Switch(α, β, γ)-gadget, T_C^f with q-vertex q_C^f. We connect the introduced gadgets by identifying the r-vertex of the preceding switching gadget with the p-vertex of the following one. Moreover, we add edges from the p-vertex

of the first switching gadget to x_C and from the r-vertex from the last switching gadget to y_C. This completes the construction of constraint gadgets.

Variable gadgets. Let $V = \{v_1, \ldots, v_n\}$. Let $S, S' \subseteq V(H^{\otimes k})$ be the sets of endpoints of the induced matching M of size q in $H^{\otimes k}$ and such that $S \subseteq X^k$, $S' \subseteq Y^k$ for X, Y being the bipartition classes of H. By S_i (resp. S'_i) we denote the projection of S (resp. S') on ith coordinate. Note that each S_i and each S'_i is incomparable (since H is incomparable) and one-sided. For each variable v_i, for each constraint C that contains v_i, and for each satisfying assignment f of the variables of C, we call Theorem 3.18 to introduce an S-gadget $X_{i,f,C}$ and an S'-gadget $X'_{i,f,C}$, with interface vertices respectively $x^1_{i,f,C}, \ldots, x^k_{i,f,C}$ and $x'^1_{i,f,C}, \ldots, x'^k_{i,f,C}$. We will refer to $X_{i,f,C}$ and $X'_{i,f,C}$ as variable gadgets.

Connecting variables and constraints. Note that since $|S| = q$, there is a bijection between S and possible assignments of one variable from V, let us fix one, say $\mu : S \to D$. Let $X_{i,f,C}$ be the variable gadget introduced for the variable v_i and a satisfying assignment f for a constraint C. Let $(s_1, \ldots, s_k)$ be the tuple from S that corresponds to $f(v_i)$, i.e., $\mu((s_1, \ldots, s_k)) = f(v_i)$. For each interface vertex $x^j_{i,f,C}$ we introduce an $\mathsf{Assign}(S, s_j, \alpha, \beta, \gamma)$-gadget A_{s_j} and identify its x-vertex with $x^j_{i,f,C}$ and its y-vertex with q^f_C. Then we add the edges $x^j_{i,f,C}x'^j_{i,f,C}$. Finally, let us fix an arbitrary ordering of the constraints in $\mathcal{C}$, and an ordering of q-vertices so that q_1 precedes a vertex q_2 if:

- q_1 belongs to P_C and q_2 belongs to $P_{C'}$, and C precedes C', or
- q_1 and q_2 belong to the same constraint gadget P_C and q_1 was introduced before q_2 on P_C.

Let q^f_C and $q^{f'}_{C'}$ be consecutive q-vertices. We add all edges $x'^j_{i,f,C}x^j_{i,f',C'}$. This completes the construction of (G, L) (see Figure 4.4).

Correctness. Let us verify that $(G, L) \to H$ if and only if $(V, \mathcal{C})$ is satisfiable.

Claim 4.17.1. *If $(G, L) \to H$, then $(V, \mathcal{C})$ is satisfiable.*

Proof of Claim: Let φ be a list homomorphism from (G, L) to H. First observe that since M is an induced matching in $H^{\otimes k}$, all variable gadgets $X_{i,f',C}$ corresponding to the variable v_i have its interface vertices mapped to the same tuple $(s_1, \ldots, s_k)$ of S. Indeed, since for

every variable gadget $X_{i,f',C}$ its interface vertices are adjacent to corresponding interface vertices of the gadget $X'_{i,f',C}$, mapping interface vertices of $X_{i,f',C}$ to $(s_1, \ldots, s_k)$ forces mapping the interface vertices of $X'_{i,f',C}$ to a tuple $(s'_1, \ldots, s'_k)$ such that $s_\ell s'_\ell \in E(H)$ for every $\ell \in [k]$, and thus $(s_1, \ldots, s_k)$ is adjacent to $(s'_1, \ldots, s'_k)$ in $H^{\otimes k}$. Since the interface vertices can be mapped only to tuples that are endpoints of the induced matching M in $H^{\otimes k}$, the tuple $(s_1, \ldots, s_k)$ uniquely determines $(s'_1, \ldots, s'_k)$. Similarly, any mapping of the interface vertices of $X'_{i,f',C}$ uniquely determines the mapping of the interface vertices of the variable gadget $X_{i',f'',C'}$ following $X'_{i,f',C}$. Therefore we can set $f(v_i) = \mu((s_1, \ldots, s_k))$, for $(s_1, \ldots, s_k)$ being the tuple of vertices that the interface vertices of any variable gadget $X_{i,f',C}$ are mapped to.

Since the vertices $\alpha, \alpha', \beta, \beta'$ induce a matching in H, for every constraint C, the p-vertex of the first switching gadget on P_C must be mapped to α and the r-vertex of the last switching gadget on P_C must be mapped to β. Therefore, there must be a switching gadget $T_C^{f'}$ such that its p-vertex is mapped to α and its r-vertex is mapped to β. By the definition of $\mathsf{Switch}(\alpha, \beta, \gamma)$-gadget, the vertex $q_C^{f'}$ must be mapped to γ. Recall that for every variable v_i that appears in C, we introduced k assignment gadgets A_{s_j}, where $(s_1, \ldots, s_k)$ is the tuple such that $f'(v_i) = \mu((s_1, \ldots, s_k))$. By the definition of the $\mathsf{Assign}(S, s_j, \alpha, \beta, \gamma)$-gadget, each vertex $x^j_{i,f',C}$ is mapped by φ to s_j. Thus $f = f'$ on the variables from C, so f satisfies C. This completes the proof of the claim. ⌟

Claim 4.17.2. *If $(V, \mathcal{C})$ is satisfiable, then $(G, L) \to H$.*

Proof of Claim: Let f be a satisfying assignment of the variables of $(V, \mathcal{C})$. We construct $\varphi : (G, L) \to H$ as follows. For every variable gadget $X_{i,f',C}$ we set its interface vertices $x^j_{i,f,C}$ to s_j, where $(s_1, \ldots, s_k) = \mu^{-1}(f(v_i))$ and we map the vertices $x'^j_{i,f,C}$ to s_j, where $(s'_1, \ldots, s'_k)$ is the private neighbor of $(s_1, \ldots, s_k)$ in $H^{\otimes k}$ in the set S'. By the definition of the S-gadget and the S'-gadget, this mapping can be extended to every variable gadget. Now consider a constraint C and a switching gadget T_C^f. For this gadget, we map its p-vertex to α, its q-vertex to γ, its r-vertex to β, and we extend φ to all other vertices of this gadget by the definition of $\mathsf{Switch}(\alpha, \beta, \gamma)$-gadget. For all other switching gadgets on P_C that precede T_C^f, we map their p-vertices and r-vertices to α and we extend φ to every switching gadget so that no q-vertex is mapped to γ, which again can be done the definition of $\mathsf{Switch}(\alpha, \beta, \gamma)$-gadget. Similarly, for all switching gadgets that follow T_C^f on P_C we map their p-vertices and r-vertices to β and extend φ on remaining vertices of the gadgets so

that no q-vertex is mapped to γ. It remains to map the vertices of the assignment gadgets. Let A be an assignment gadget that joins an interface vertex $x^j_{i,f',C}$ with a q-vertex $q^{f'}_C$. If $f' = f$, then $\varphi(q^f_C) = \gamma$ and $\varphi(x^j_{i,f}) = s_j$, where $(s_1, \ldots, s_k) = \mu^{-1}(f(v_j))$. Recall that A must be an assignment gadget A_{s_j}. Therefore, by the definition of the $\mathsf{Assign}(S, s_j, \alpha, \beta, \gamma)$-gadget, the homomorphism φ can be extended to remaining vertices of the gadget. Finally, if $f' \neq f$, then $\varphi(q^{f'}_C) \in \{\alpha, \beta\}$, and again the definition of the $\mathsf{Assign}(S, s_j, \alpha, \beta, \gamma)$-gadget implies that the homomorphism φ can be extended to remaining vertices of the gadget. This completes the proof of the claim. ⌟

Cutwidth. Now we will construct a linear ordering σ of $V(G)$ of width at most $k \cdot n + r \cdot g(k, H)$, where g is a function that depends only on k and H. First we order vertices of the constraint gadgets so that we first put the vertices from P_{C_1}, and among the vertices of one constraint gadget, we order vertices according to the ordering of the switching gadgets (among vertices of one switching gadget the order is arbitrary), then from P_{C_2}, and so on. Then we modify this ordering by inserting, immediately after each q-vertex q^f_C, the vertices from the gadgets $X_{i,f,C}$ and $X'_{i,f,C}$ such that v_i is a variable that appears in C, and we place there also the vertices of the assignment gadgets that connect $X_{i,f,C}$ with q^f_C. The ordering of the vertices of these gadgets is arbitrary. This completes the construction of the ordering of $V(G)$. Now let us verify that it has desired width. Consider any cut. The edges that can cross this cut are:

- at most $g_1(H)$ edges from a constraint gadget, where g_1 depends only on H,
- at most $r \cdot g_2(k, H)$ edges of variable gadgets and assignment gadgets corresponding to the same q-vertex, where g_2 depends only on k and H,
- at most $n \cdot k$ edges that connect consecutive variable gadgets.

Therefore the cutwidth of G is at most $n \cdot k + r \cdot g(k, H)$, where g depends only on k and H.

Finally, observe that the construction of (G, L) is performed in time polynomial in $(n + m)$ and thus $|G| = (n + m)^{\mathcal{O}(1)}$. This completes the proof. □

We will use Lemma 4.17 to provide a series of lower bounds for the complexity of LHom(H). We will use the folowing result of Lampis [97]. The first part of the theorem was not stated in the paper, but it follows from the proof of the second part.

Theorem 4.18 (Lampis [97]). *(1.) Assuming the ETH, there exists $\delta > 0$ such that for every $q \geqslant 2$ the following holds. The CSP$(q,3)$ problem on n variables and m clauses cannot be solved in time $q^{\delta \cdot n} \cdot (n+m)^{\mathcal{O}(1)}$.*

(2.) Assuming the SETH, for every $q \geqslant 2$ and $\varepsilon > 0$ there exists r such that the following holds. The CSP(q,r) problem on n variables and m clauses cannot be solved in time $(q-\varepsilon)^n \cdot (n+m)^{\mathcal{O}(1)}$.

By combining Theorem 4.18 with Lemma 4.17, we obtain the following.

Theorem 4.19. *Let $\mathcal{H}_0$ denote the set of connected incomparable bipartite graphs whose complement is not a circular-arc graph.*

(1.) Assuming the ETH, there exists $\delta > 0$ such that for every $H \in \mathcal{H}_0$ and for every $k \in \mathbb{N}$, the following holds. There is no algorithm solving every instance (G, L) of LHOM(H) given with a linear ordering of width t in time $\mathrm{mim}(H^{\otimes k})^{\frac{1}{k} \cdot \delta \cdot t} \cdot |G|^{\mathcal{O}(1)}$.

(2.) Assuming the SETH, for every $H \in \mathcal{H}_0$, for every $k \in \mathbb{N}$, and for every $\varepsilon > 0$, the following holds. There is no algorithm solving every instance (G, L) of LHOM(H) given with a linear ordering of width t in time $(\mathrm{mim}(H^{\otimes k})^{\frac{1}{k}} - \varepsilon)^t \cdot |G|^{\mathcal{O}(1)}$.

Proof of (1.) First assume the ETH and let $\delta > 0$ be given by Theorem 4.18 (1.). Furthermore let $H \in \mathcal{H}_0$ and let $q = \mathrm{mim}(H^{\otimes k})$. Suppose that there is an algorithm $\mathcal{A}_1$ that solves every instance (G, L) of LHOM(H) given with a linear ordering of width t in time $\mathrm{mim}(H^{\otimes k})^{\frac{1}{k} \cdot \delta \cdot t} \cdot |G|^{\mathcal{O}(1)}$. Let $(V, \mathcal{C})$ be an instance of CSP$(q,3)$ on n variables and m clauses. We call Lemma 4.17 to construct an instance (G, L) of LHOM(H) with a linear ordering σ of width t satisfying the conditions given in Lemma 4.17. Thus we can solve the instance $(V, \mathcal{C})$ by calling $\mathcal{A}_1$ in time:

$$\mathrm{mim}(H^{\otimes k})^{\frac{1}{k} \cdot \delta \cdot t} \cdot |G|^{\mathcal{O}(1)} \leqslant q^{\frac{1}{k} \cdot \delta \cdot (k \cdot n + r \cdot g(k,H))} \cdot (n+m)^{\mathcal{O}(1)} = q^{\delta \cdot n} \cdot (n+m)^{\mathcal{O}(1)},$$

which is a contradiction by Theorem 4.18. □

Proof of (2.) Now assume the SETH. Let $\varepsilon > 0$, let $H \in \mathcal{H}_0$ and suppose there is an algorithm $\mathcal{A}_2$ that solves every instance (G, L) of LHOM(H) given with a linear ordering of width t in time $(\mathrm{mim}(H^{\otimes k})^{\frac{1}{k}} - \varepsilon)^t \cdot |G|^{\mathcal{O}(1)}$. Let $q = \mathrm{mim}(H^{\otimes k})$, let $\varepsilon' = q - (q^{\frac{1}{k}} - \varepsilon)^k > 0$, and let $r = r(q, \varepsilon')$ be given by Theorem 4.18 (2.). Let $(V, \mathcal{C})$ be an instance of

$\mathrm{CSP}(q, r)$ on n variables and m clauses. We call Lemma 4.17 to construct an instance (G, L) of $\mathrm{LHOM}(H)$ with a linear ordering σ of width t satisfying the conditions given in Lemma 4.17.

Since the instance (G, L) is equivalent to the instance $(V, \mathcal{C})$, we can use $\mathcal{A}_2$ to solve $(V, \mathcal{C})$. The reduction is performed in time polynomial in $(n + m)$, and $\mathcal{A}_2$ runs in time:

$$(\mathrm{mim}(H^{\otimes k})^{\frac{1}{k}} - \varepsilon)^t \cdot |G|^{\mathcal{O}(1)} \leqslant (q^{\frac{1}{k}} - \varepsilon)^{k \cdot n + r \cdot g(k,H)} \cdot (n+m)^{\mathcal{O}(1)} = (q^{\frac{1}{k}} - \varepsilon)^{k \cdot n} \cdot (n+m)^{\mathcal{O}(1)}$$
$$= (q - \varepsilon')^n \cdot (n+m)^{\mathcal{O}(1)},$$

which is a contradiction by Theorem 4.18. □

By definition, for a bipartite H, we have $\mathrm{mimsup}(H) = \sup_{k \in \mathbb{N}} \mathrm{mim}(H^{\otimes k})^{1/k}$, so Theorem 4.19 yields the following.

Theorem 4.14. *Let $\mathcal{H}_0$ denote the set of connected incomparable bipartite graphs whose complement is not a circular-arc graph.*

(1.) Assuming the ETH, there exists $\delta > 0$ such that for every $H \in \mathcal{H}_0$ the following holds. There is no algorithm that solves every consistent instance (G, L) of $\mathrm{LHOM}(H)$, given with a linear ordering of $V(G)$ of width t, in time $\mathrm{mimsup}(H)^{\delta \cdot t} \cdot |G|^{\mathcal{O}(1)}$.

(2.) Assuming the SETH, for every $\varepsilon > 0$ and $H \in \mathcal{H}_0$ the following holds. There is no algorithm that solves every consistent instance (G, L) of $\mathrm{LHOM}(H)$, given with a linear ordering of $V(G)$ of width t, in time $(\mathrm{mimsup}(H) - \varepsilon)^t \cdot |G|^{\mathcal{O}(1)}$.

Proof of (1.) Let $H \in \mathcal{H}_0$ and let (G, L) be an instance of $\mathrm{LHOM}(H)$ given with a linear ordering of width t. Let $\delta' > 0$ be given by Theorem 4.19 and let $0 < \delta < \delta'$. Furthermore, let $k \in \mathbb{N}$ be such that $\mathrm{mim}(H^{\otimes k})^{\frac{1}{k}} > \mathrm{mimsup}(H)^{\frac{\delta}{\delta'}}$ – it exists by the definition of mimsup and the fact that $\frac{\delta}{\delta'} < 1$. If (G, L) can be solved in time

$$\mathrm{mimsup}(H)^{\delta \cdot t} \cdot |G|^{\mathcal{O}(1)} < \mathrm{mim}(H^{\otimes k})^{\frac{\delta'}{k} \cdot t} \cdot |G|^{\mathcal{O}(1)},$$

then, by Theorem 4.19, it contradicts the ETH. □

Proof of (2.) Let $H \in \mathcal{H}_0$ and let (G, L) be an instance of $\mathrm{LHOM}(H)$ given with a linear ordering of width t. Let $\varepsilon > 0$ and let $\varepsilon' = \frac{\varepsilon}{2}$. By the definition of $\mathrm{mimsup}(H)$, there

exists $k \in \mathbb{N}$ such that $\mathrm{mim}(H^{\otimes k})^{\frac{1}{k}} - \varepsilon' \geqslant \mathrm{mimsup}(H) - \varepsilon$. If (G, L) can be solved in time

$$(\mathrm{mimsup}(H) - \varepsilon)^t \cdot |G|^{\mathcal{O}(1)} \leqslant (\mathrm{mim}(H^{\otimes k})^{\frac{1}{k}} - \varepsilon')^t \cdot |G|^{\mathcal{O}(1)},$$

then, by Theorem 4.19, it contradicts the SETH. □

4.4.4 List homomorphisms and general target graphs

Now we extend Theorem 4.14 to general target graphs. Recall Proposition 3.15.

Proposition 3.15 (Okrasa, Piecyk, Rzążewski [121])**.** *Let H be a graph, and let (G, L) be a consistent instance of LHOM(H^*). Define $L' : V(G) \to 2^{V(H)}$ as $L'(v) = \{u \mid \{u', u''\} \cap L(v) \neq \emptyset\}$. Then $(G, L) \to H^*$ if and only if $(G, L') \to H$.*

Combining Theorem 4.14 with Proposition 3.15 yields the following lower bound.

Theorem 4.20. *Let $\mathcal{H}_1$ denote the set of connected incomparable non-bipartite graphs.*

(1.) Assuming the ETH, there exists $\delta > 0$ such that for every $H \in \mathcal{H}_1$ the following holds. There is no algorithm that solves every instance (G, L) of LHOM(H), given with a linear ordering of $V(G)$ of width k, in time $\mathrm{mimsup}(H)^{\delta \cdot k} \cdot |G|^{\mathcal{O}(1)}$.

(2.) Assuming the SETH, for every $\varepsilon > 0$ and $H \in \mathcal{H}_1$ the following holds. There is no algorithm that solves every instance (G, L) of LHOM(H), given with a linear ordering of $V(G)$ of width k, in time $(\mathrm{mimsup}(H) - \varepsilon)^k \cdot |G|^{\mathcal{O}(1)}$.

Proof. Fix any $H \in \mathcal{H}_1$. Observe that $H^* \in \mathcal{H}_0$, where $\mathcal{H}_0$ is defined as in Theorem 4.14. Indeed, the connectivity of H^* follows easily from the fact that H is connected and non-bipartite, see e.g. [124, Observation 2.5]. The fact that H^* is incomparable follows from the fact that H is incomparable and has no isolated vertices. Finally, H contains an (induced) odd cycle, so H^* contains an induced cycle with at least 6 vertices, and by Proposition 3.14, H^* cannot be the complement of a circular-arc graph.

Now suppose that we have an algorithm $\mathcal{A}$ that solves every instance $(\widetilde{G}, \widetilde{L})$ of LHOM(H) in time $f(H, \widetilde{G})$, where f is some function that depends on H and $\widetilde{G}$. Proposition 3.15 implies that that for any consistent instance (G, L) of LHOM(H^*), we can solve it by calling $\mathcal{A}$ on the instance (G, L') of LHOM(H), defined as in Proposition 3.15, in time $f(H, G)$. Recall that for non-bipartite H, we have $\mathrm{mimsup}(H) = \mathrm{mimsup}(H^*)$. Thus the statement of the theorem follows directly from Theorem 4.14. □

We point out that in the instance $(G.L')$ constructed in the reduction above, the graph G is bipartite. This yields the following corollary.

Corollary 4.21. *Let $\mathcal{H}_1$ denote the set of connected incomparable non-bipartite graphs.*

(1.) Assuming the ETH, there exists $\delta > 0$ such that for every $H \in \mathcal{H}_1$ the following holds. There is no algorithm that solves every instance (G, L) of $\text{LHOM}(H)$ such that G is bipartite, given with a linear ordering of $V(G)$ of width k, in time $\text{mimsup}(H)^{\delta \cdot k} \cdot |G|^{\mathcal{O}(1)}$.

(2.) Assuming the SETH, for every $\varepsilon > 0$ and $H \in \mathcal{H}_1$ the following holds. There is no algorithm that solves every instance (G, L) of $\text{LHOM}(H)$ such that G is bipartite, given with a linear ordering of $V(G)$ of width k, in time $(\text{mimsup}(H) - \varepsilon)^k \cdot |G|^{\mathcal{O}(1)}$.

4.4.5 Hardness of Hom(H)

Finally, we are ready to prove lower bounds for the non-list variant of the problem. We will extend Theorem 4.20 using methods described in Chapter 3.

Theorem 4.22. *(A) Let $\mathcal{H}_2$ be a set of connected projective non-bipartite cores.*

(1) There exists $\delta > 0$, such that for every $H \in \mathcal{H}_1$, there is no algorithm solving every instance G of $\text{HOM}(H)$ in time $\text{mimsup}^*(H)^{\delta \cdot \text{ctw}(G)} \cdot |G|^{\mathcal{O}(1)}$, unless the ETH fails.

(2) Let $H \in \mathcal{H}_2$. There is no algorithm solving every instance G of $\text{HOM}(H)$ in time $(\text{mimsup}^*(H) - \varepsilon)^{\text{ctw}(G)} \cdot |G|^{\mathcal{O}(1)}$ for any $\varepsilon > 0$, unless the SETH fails.

(B) Let $\mathcal{H}_3$ be a set of connected non-projective non-bipartite cores. If Conjecture 3.9 and Conjecture 3.11 are true, then the following holds.

(1) There exists $\delta > 0$, such that for every $H \in \mathcal{H}_3$, there is no algorithm solving every instance G of $\text{HOM}(H)$ in time $\text{mimsup}^*(H)^{\delta \cdot \text{ctw}(G)} \cdot |G|^{\mathcal{O}(1)}$, unless the ETH fails.

(2) Let $H \in \mathcal{H}_3$. There is no algorithm solving every instance G of $\text{HOM}(H)$ in time $(\text{mimsup}^*(H) - \varepsilon)^{\text{ctw}(G)} \cdot |G|^{\mathcal{O}(1)}$ for any $\varepsilon > 0$, unless the SETH fails.

Proof. The general framework of the proof is the same in (A) and (B), so we will prove both in parallel. Let $H \in \mathcal{H}_2 \cup \mathcal{H}_3$. If $H \in \mathcal{H}_2$ (case (A)), then we set $H' = H$. If $H \in \mathcal{H}_3$ (case (B)), then by Conjecture 3.9 there are projective graphs $H_1, \ldots, H_p$ such that $H = H_1 \times \ldots \times H_p$. Without loss of generality we can assume that H_1 has the largest mimsup among all factors of H. We set $H' = H_1$. Moreover, let us denote $W = H_2 \times \ldots \times H_p$, so $H = H' \times W$. Observe that in both cases we have $\mathrm{mimsup}(H') = \mathrm{mimsup}^*(H)$. Furthermore, since H is a connected non-bipartite core, then so is H' – in case (A) it is clear since $H' = H$, and in case (B) if follows from Proposition 3.4. Moreover, by Proposition 3.1 (2), H' is incomparable and thus $H' \in \mathcal{H}_1$ (where $\mathcal{H}_1$ is defined as in Theorem 4.20).

Therefore, we can reduce from $\mathrm{LHOM}(H')$ – recall that by Corollary 4.21, it is sufficient to reduce from an instance where the input graph is bipartite. So let (G, L) be an instance of $\mathrm{LHOM}(H')$, given along with a linear ordering $\sigma = (v_1, v_2, \ldots, v_{|V(G)|})$ of $V(G)$ of width k and such that G is bipartite. Let X_G, Y_G be the bipartition classes of G.

We will construct, in time polynomial in $|G|$, an instance $\widetilde{G}$ of $\mathrm{HOM}(H)$ with the following properties:

1. $\widetilde{G} \to H$ if and only if $(G, L) \to H'$,
2. $|\widetilde{G}| = |G| \cdot f(H)$,
3. $\mathrm{ctw}(\widetilde{G}) \leqslant k + g(H)$,

where f and g are functions whose value depends only on H.

We start constructing $\widetilde{G}$ by taking a copy of G. Then for every vertex $v \in V(G)$ we proceed as follows. In case (A), using Theorem 3.8, we introduce a construction of $L(v)$, i.e., for $S = L(v)$, we introduce the graph $\mathsf{C}(S)$ with special vertices $y_0, \ldots, y_\ell$. Moreover, let $(x_1, \ldots, x_\ell)$ be the fixed tuple of vertices of $V(H)$ from Definition 3.7. We identify y_0 with v.

In case (B), let ab be an edge of W – it exists since W is non-bipartite. If $v \in X_G$ (resp., $v \in Y_G$), then, using Theorem 3.12, we introduce a construction of $(L(v), a)$ (resp., $(L(v), b)$), i.e., we introduce the graph $\mathsf{C}(S, a)$ (resp. $\mathsf{C}(S, b)$) with special vertices $y_0, \ldots, y_\ell$. Moreover, let $(x_1, \ldots, x_\ell)$ be the fixed tuple of vertices of $V(H)$ from Definition 3.10. Again, we identify y_0 with v.

Next, (in both cases) we introduce a copy H^v of H; for any $z \in V(H)$, let z^v denote the copy of z in H^v. For each $i \in [\ell]$ we identify x_i^v with y_i.

Finally, for every pair $v_j, v_{j+1} \in V(G)$, we proceed as follows. For every $wz \in E(H)$, we add an edge between w^{v_j} and $z^{v_{j+1}}$. This completes the construction of $\widetilde{G}$. Clearly the construction is performed in time polynomial in $|G|$, and $|\widetilde{G}| = |G| \cdot f(H)$ for some function f that depends only on H.

Correctness. Let us verify the equivalence of the instances $\widetilde{G}$ and (G, L).

Claim 4.22.1. *If $(G, L) \to H'$, then $\widetilde{G} \to H$.*

Proof of Claim: Suppose that there exists $\varphi' : (G, L) \to H'$. We define $\varphi : \widetilde{G} \to H$ as follows.

Case (A). We set $\varphi|_{V(G)} = \varphi'$ and for every copy H^v of H we define φ on its vertices as the identity function. Note that the function defined so far respects the edges inside $V(G)$ since φ' is a homomorphism, and the edges between copies of H as vertices of two consecutive copies of H are adjacent only to vertices that correspond to their neighbors in their own copy. It remains to extend φ to the remaining vertices of graphs $\mathsf{C}(S)$. We can do it independently for every graph $\mathsf{C}(S)$ as there are no edges between them. Consider such $\mathsf{C}(S)$ introduced for some $v \in V(G)$ and $S = L(v)$. Since $\varphi(y_0) = \varphi(v) \in L(v)$ and $\varphi(y_i) = x_i$ for every $i \in [\ell]$, Theorem 3.8 implies that the mapping φ can indeed be extended to a homomorphism from $\widetilde{G}$ to H.

Case (B). Let $v \in V(G)$. If $v \in X_G$, then we set $\varphi(v) = (\varphi'(v), a)$, otherwise we set $\varphi(v) = (\varphi'(v), b)$. Furthermore, we set φ on every copy H^v of H to the identity function. As in the previous case, the function defined so far respects the edges. Indeed, note that for an edge $uv \in E(G)$, say $u \in X_G$ and $v \in Y_G$, we have $\varphi(u) = (\varphi'(u), a)$ and $\varphi(v) = (\varphi'(v), b)$. Moreover, $\varphi'(u)\varphi'(v) \in E(H')$ since φ' is a homomorphism and $ab \in E(W)$, so $\varphi(u)\varphi(v) \in E(H)$. It remains to extend φ to the remaining vertices of graphs $\mathsf{C}(S, a)$ and $\mathsf{C}(S, b)$. If $v \in X_G$ (resp. $v \in Y_G$), then $\varphi(y_0) = (\varphi'(v), a)$ (resp. $\varphi(y_0) = (\varphi'(v), b)$), $\varphi'(v) \in L(v)$, and $\varphi(y_i) = x_i$ for every $i \in [\ell]$. Therefore, by the definition of construction, φ can be extended to the remaining vertices of $\mathsf{C}(S, a)$ (resp. $\mathsf{C}(S, b)$). ⌟

Claim 4.22.2. *If $\widetilde{G} \to H$, then $(G, L) \to H'$.*

Proof of Claim: Suppose that there is $\varphi : \widetilde{G} \to H$. Let us define $\varphi' : G \to H'$.

Case (A). We define $\varphi = \varphi'|_{V(G)}$. Clearly φ is a homomorphism. Let us verify that it respects lists. First, since H is a core, Proposition 3.1 (1) implies that φ' restricted to any copy H^v of H is an automorphism. We claim that for each copy it is actually the same automorphism of H. Indeed, let H^{v_j} and $H^{v_{j+1}}$ be consecutive copies and let $z \in V(H)$. Let $s := \varphi'(z^{v_j})$ and $u := \varphi'(z^{v_{j+1}})$. For contradiction, suppose that $s \neq u$. Since φ' restricted to $H^{v_{j+1}}$ is an automorphism, the image of $N_{H^{v_{j+1}}}(z^{v_{j+1}})$ is precisely $N_H(u)$. Furthermore, z^{v_j} is adjacent to every vertex $w^{v_{j+1}} \in N_{H^{v_{j+1}}}(z^{v_{j+1}})$, and thus $\varphi'(w^{v_{j+1}}) \in N_H(s)$ for every $w^{v_{j+1}} \in N_{H^{v_{j+1}}}(z^{v_{j+1}})$. Therefore, $N_H(u) \subseteq N_H(s)$, which by Proposition 3.1 (2), is a contradiction with the fact that H is incomparable.

So since now we can assume that for each copy H^v, we have the same automorphism of H. Without loss of generality assume that this "global" automorphism of H is the identity. Now consider any $v \in V(G)$. Since each vertex x_i from the copy of $\mathsf{C}(L(v))$ corresponding to v is mapped to y_i and $\mathsf{C}(L(v))$ is a construction of $L(v)$, we conclude that $v(= y_0)$ is mapped to some element of $L(v)$. Thus φ' respects lists.

Case (B). For every $v \in V(G)$, for $(s, w) = \varphi(v)$ such that $s \in V(H'), w \in V(W)$, we set $\varphi'(v) = s$, i.e., $\varphi' = \pi_1 \circ \varphi$, where π_1 is the projection to the first coordinate. Clearly φ' is a homomorphism. Let us verify that it respects lists. As in the previous case, φ restricted to any copy H^v of H is an automorphism, and we may assume that this automorphism is the identity. Now consider any $v \in V(G)$. Assume that $v \in X_G$ (the case $v \in Y_G$ is symmetric) and let $(s, w) = \varphi(v)$. Since each vertex y_i from the copy of $\mathsf{C}(L(v), a)$ introduced for v is mapped to x_i, by the definition of $\mathsf{C}(L(v), a)$, we conclude that $s \in L(v)$, which means that $v(= y_0)$ is mapped by φ' to some element of $L(v)$. Thus φ' respects lists. ⌟

Cutwidth of $\widetilde{G}$. Now let us define a linear ordering $\widetilde{\sigma}$ of $V(\widetilde{G})$. First we order the vertices of G according to σ. Then we modify this ordering by inserting right after a vertex v the vertices of H^v and, in case (A) the graph $\mathsf{C}(L(v))$ introduced for v, and in case (B) the graph $\mathsf{C}(L(v), a)$ or $\mathsf{C}(L(v), b)$ introduced for v; the order among these vertices is arbitrary. This completes the definition of $\widetilde{\sigma}$. Consider any cut in $\widetilde{\sigma}$. Among the edges crossing this cut there can be:

a) at most k edges of $E(G)$,

b) edges from at most one copy of H and at most one graph $\mathsf{C}(S)$ or $\mathsf{C}(S,a)$ or $\mathsf{C}(S,b)$,

c) edges joining two consecutive copies of H.

Observe that the number of edges in b) and c) can be bounded by a constant that depends only on H, say $g(H)$. Therefore, we can conclude that $\mathrm{ctw}(\widetilde{G}) \leqslant k + g(H)$.

Wrapping up the proof. Now consider δ from Theorem 4.20 and suppose there is an algorithm $\mathcal{A}_1$ that solves every instance G' of $\textsc{Hom}(H)$ in time $\mathrm{mimsup}^*(H)^{\delta\cdot\mathrm{ctw}(G')} \cdot |G'|^{\mathcal{O}(1)}$. Then, we can call the above reduction for any instance (G,L) of $\textsc{LHom}(H')$ given along with a linear ordering of $V(G)$ of width k and use $\mathcal{A}_1$ to solve it in time:

$$\begin{aligned}&\mathrm{mimsup}^*(H)^{\delta\cdot\mathrm{ctw}(\widetilde{G})} \cdot |\widetilde{G}|^{\mathcal{O}(1)} = \mathrm{mimsup}(H')^{\delta\cdot\mathrm{ctw}(\widetilde{G})} \cdot |\widetilde{G}|^{\mathcal{O}(1)}\\ &\leqslant \mathrm{mimsup}(H')^{\delta\cdot(k+g(H))} \cdot |G|^{\mathcal{O}(1)} = \mathrm{mimsup}(H')^{\delta\cdot k} \cdot |G|^{\mathcal{O}(1)},\end{aligned}$$

which by Theorem 4.20 (1.) contradicts the ETH.

Finally, let $\varepsilon > 0$ and suppose there is an algorithm $\mathcal{A}_2$ that solves every instance G' of $\textsc{Hom}(H)$ in time $(\mathrm{mimsup}^*(H)-\varepsilon)^{\mathrm{ctw}(G')}\cdot|G'|^{\mathcal{O}(1)}$. Then, we can call the above reduction for any instance (G,L) of $\textsc{LHom}(H')$ given along with a linear ordering of $V(G)$ of width k and use $\mathcal{A}_2$ to solve it in time:

$$\begin{aligned}&(\mathrm{mimsup}^*(H)-\varepsilon)^{\mathrm{ctw}(\widetilde{G})} \cdot |V(\widetilde{G})|^{\mathcal{O}(1)} = (\mathrm{mimsup}(H')-\varepsilon)^{\mathrm{ctw}(\widetilde{G})} \cdot |\widetilde{G}|^{\mathcal{O}(1)}\\ &\leqslant (\mathrm{mimsup}(H')-\varepsilon)^{k+g(H)} \cdot |G|^{\mathcal{O}(1)} = (\mathrm{mimsup}(H')-\varepsilon)^{k} \cdot |G|^{\mathcal{O}(1)},\end{aligned}$$

which, by Theorem 4.20 (2.), contradicts the SETH. □

4.5 Comparison of parameters

In this section, we state various combinatorial results about the parameters studied in this chapter. Before we move to comparing mimsup with other parameters, we give some properties of the parameter itself.

In the following observation we show that mim is *supermultiplicative with respect to* $\otimes$, i.e., we always have $\mathrm{mim}(A\otimes B) \geqslant \mathrm{mim}(A)\,\mathrm{mim}(B)$. This also implies that sequences such as $\left(\mathrm{mim}(A^{\otimes 2^k})^{1/2^k}\right)_{k\in\mathbb{N}}$ are non-decreasing.

Lemma 4.23. *For two 0-1 matrices A and B, it holds that* $\mathrm{mim}(A\otimes B) \geqslant \mathrm{mim}(A)\,\mathrm{mim}(B)$.

Proof. Suppose that $(r_1, c_1), \ldots, (r_a, c_a)$ is such that $A[(r_1, \ldots, r_a), (c_1, \ldots, c_a)]$ is the identity matrix. Similarly, suppose the submatrix of B induced on rows $r'_1, \ldots, r'_b$ and columns $c'_1, \ldots, c'_b$ (in that order) is the identity matrix. In $A \otimes B$, we may consider the submatrix induced on ab rows

$$\{(r_i, r'_j) \mid i \in [a], j \in [b]\}$$

and columns

$$\{(c_i, c'_j) \mid i \in [a], j \in [b]\}.$$

In this case, $(A \otimes B)[(r_i, r'_j), (c_v, c'_w)] = A[r_i, c_v]B[r'_j, c'_w]$ is 1 if and only if $i = v$ and $j = w$. This gives an induced matching of size $ab = \operatorname{mim}(A) \operatorname{mim}(B)$, as desired. □

In order to prove next theorem, we will use the Fekete's lemma [59].

Theorem 4.24 (Fekete's lemma [59])**.** *For any sequence $(a_n)_{n\in\mathbb{N}}$ which is* subadditive *($a_{n+m} \leqslant a_n + a_m$ for all $n, m \in \mathbb{N}$), it holds that $\lim_{n\to\infty} \frac{a_n}{n}$ exists and is equal to $\inf_{n\in\mathbb{N}} \frac{a_n}{n}$.*

We can now prove that in the definition of mimsup, we can replace the supremum with the limit.

Theorem 4.1. *Let $A \in \{0, 1\}^{n\times m}$. Then*

$$\sup_{k\in\mathbb{N}} \operatorname{mim}(A^{\otimes k})^{1/k} = \lim_{k\to\infty} \operatorname{mim}(A^{\otimes k})^{1/k}.$$

Proof. Since mim is supermultiplicative with respect to $\otimes$, the sequence defined by

$$a_k = -k \log_2(\operatorname{mim}(A^{\otimes k})^{1/k}) = -\log_2(\operatorname{mim}(A^{\otimes k}))$$

is subadditive.

By Theorem 4.24, the limit $\lim_{k\to\infty} \frac{a_k}{k}$ exists and is equal to $\inf_{k\in\mathbb{N}} \frac{a_k}{k}$. Therefore,

$$\lim_{k\to\infty} -\log_2(\operatorname{mim}(A^{\otimes k})^{1/k}) = \inf_k -\log_2(\operatorname{mim}(A^{\otimes k})^{1/k}).$$

That, in turn, implies that $\lim_{k\to\infty} \operatorname{mim}(A^{\otimes k})^{1/k}$ exists and is equal to $\sup_{k\in\mathbb{N}} \operatorname{mim}(A^{\otimes k})^{1/k}$, which completes the proof. □

Since by Theorem 4.1, the limit $\lim_{k\to\infty} \operatorname{mim}(A^{\otimes k})^{1/k}$ exists, the subsequence $(\operatorname{mim}(A^{\otimes k\ell})^{1/k\ell})_\ell$ has the same limit as $(\operatorname{mim}(A^{\otimes k'})^{1/k'})_{k'}$, that is,

$$\operatorname{mimsup}(A^{\otimes k})^{1/k} = \lim_{\ell\to\infty} \operatorname{mim}(A^{\otimes k\ell})^{1/k\ell} = \lim_{k'\to\infty} \operatorname{mim}(A^{\otimes k'})^{1/k'} = \operatorname{mimsup}(A).$$

This yields the following.

Theorem 4.25. *For a matrix A, it holds that* $\operatorname{mimsup}(A^{\otimes k}) = \operatorname{mimsup}(A)^k$.

4.5.1 Comparing him and mimsup

We first make some simple observations and then show that (perhaps surprisingly) mimsup can be much larger than him.

Lemma 4.26. *Let A be a 0-1 matrix. Then* $\text{mimsup}(A) \geqslant \text{him}(A) \geqslant \text{mim}(A)$.

Proof. The second inequality follows directly since each induced matching is a half-induced matching. We prove the first inequality.

Let $R = \{a_1, \ldots, a_i\}$ and $C = \{b_1, \ldots, b_i\}$ be the rows and columns, respectively, of a maximum half-induced matching in A. We may assume that these are ordered such that $A[a_j, b_j] = 1$ for all $j \in [i]$ and $A[a_k, b_j] = 0$ for all $k < j$. For integer $s \geqslant 1$, we consider the submatrix of $A^{\otimes is}$ induced on the rows consisting of "balanced" sequences

$$\{(r_1, \ldots, r_{is}) \in R^{is} \mid |\{\ell : r_\ell = a_j\}| = s, \text{ for every } j \in [i]\}$$

and similarly for the columns

$$\{(c_1, \ldots, c_{is}) \in C^{is} \mid |\{\ell : c_\ell = b_j\}| = s, \text{ for every } j \in [i]\}.$$

We claim this forms an induced matching of size $\binom{is}{s,\ldots,s} = i^{(1+o(1))is}$, which shows that $\text{mimsup}(A) \geqslant i = \text{him}(A)$. Since the size is clear from the definition, it only remains to check that it indeed forms an induced matching. To show this, we explain why the row of

$$r = (a_1, \ldots, a_1, a_2, \ldots, a_2, \ldots, a_i, \ldots, a_i)$$

has a single one entry in column

$$c = (b_1, \ldots, b_1, b_2, \ldots, b_2, \ldots, b_i, \ldots, b_i).$$

The other cases follow by symmetry. It is clear that $A^{\otimes is}[r, c] = 1$. Any column c' with $A^{\otimes is}[r, c'] = 1$ must have $A[a_1, c'_j] = 1$ for all $j \in [s]$. But $A[a_1, b_j] = 0$ when $j > 1$, so this implies that $\{j : c'_j = b_1\} = [s]$. Similarly, we require $A[a_2, c'_j] = 1$ for all $j \in [s+1, 2s]$, but c' has already 'used' all its b_1's and so $\{j \mid c'_j = b_2\}$ needs to be $[s+1, 2s]$. Continuing inductively, we find that $\{j \mid c'_j = b_k\} = [(k-1)s+1, ks]$ for all $k \in [i]$, that is, $c' = c$. □

Since by Theorem 4.1, the limit $\lim_{k\to\infty} \text{mim}(A^{\otimes k})^{1/k}$ exists, and by Theorem 4.25, we always have $\text{mimsup}(A^{\otimes k}) = \text{mimsup}(A)^k$, applying inequality from Lemma 4.26, for every $k \in \mathbb{N}$, we obtain:

$$\operatorname{mim}(A^{\otimes k}) \leqslant \operatorname{him}(A^{\otimes k}) \leqslant \operatorname{mimsup}(A^{\otimes k}) = \operatorname{mimsup}(A)^k.$$

When k tends to infinity, we obtain the following corollary.

Corollary 4.27. *Let $A \in \{0,1\}^{n\times m}$. Then*

$$\operatorname{mimsup}(A) = \lim_{k\to\infty} \operatorname{him}(A^{\otimes k})^{1/k},$$

We are now ready to prove previously claimed bounds.

Theorem 1.5. *For every non-bipartite graph H with adjacency matrix A_H and $k \in \mathbb{N}$,*

$$\operatorname{him}(A_H) \leqslant \operatorname{mimsup}(A_H) = \lim_{k\to\infty} \operatorname{mim}(A_H^{\otimes k})^{1/k} \quad \textit{and} \quad \operatorname{mim}(A_H^{\otimes k}) \leqslant k^{\operatorname{him}(A_H)k}.$$

Proof. Lemma 4.26 shows the first inequality. By Lemma 4.5, for a matrix A and any set $\mathcal{R}$ of rows of $A^{\otimes k}$, there is always a representative set $\mathcal{R}'$ of $\mathcal{R}$ of size at most $k^{\operatorname{him}(A)k}$. On the other hand, if we consider $\mathcal{R}$ to be the set of rows that (together with appropriate columns) form an induced matching of size $\operatorname{mim}(A^{\otimes k})$ in $A^{\otimes k}$, then $\operatorname{mim}(A^{\otimes k}) \leqslant |\mathcal{R}'|$, and thus the desired inequality follows. □

In principle, $\operatorname{mim}(A^{\otimes k})^{1/k}$ could grow very slowly and it could take very long to become as large as $\operatorname{him}(A)$ (and in fact it may stay smaller for all $k \in \mathbb{N}$). Our next result gives some form of guarantee: already for $k = 2$, $\operatorname{mim}(A^{\otimes k})^{1/k} \geqslant \sqrt{\operatorname{him}(A)}$.

Lemma 4.28. $\operatorname{mim}(A^{\otimes 2}) \geqslant \operatorname{him}(A)$.

Proof. Let $a_1, \dots, a_i$ and $b_1, \dots, b_i$ be the rows and columns, respectively, of a maximum half-induced matching in A. We may assume that these are ordered such that $A[a_j, b_j] = 1$ for all $j \in [i]$ and $A[a_k, b_j] = 0$ for all $k < j$. In $A^{\otimes 2}$ we claim that the rows $r_1 = (a_1, a_i), r_2 = (a_2, a_{i-1}), \dots, r_i = (a_i, a_1)$ and the columns $c_1 = (b_1, b_i), c_2 = (b_2, b_{i-1}), \dots, c_i = (b_i, b_1)$ induce a matching of size $i = \operatorname{him}(A)$. Indeed, for $j, j' \in [i]$, by definition

$$A[c_j, r_{j'}] = A[a_j, b_{j'}]A[a_{i+1-j}, b_{i+1-j'}],$$

for which both terms are 1 if $j = j'$ and at least one term is 0 if $j \neq j'$. □

At first glance, it may be natural to conjecture that in fact $\operatorname{mimsup}(A) = \operatorname{him}(A)$ for all matrices A. This is however not true, as the following result shows.

Theorem 4.29. *There is a constant $C > 1$, such that for all integers $h \geqslant C$, there is a symmetric matrix $A \in \{0,1\}^{2h \times 2h}$ with $\mathrm{him}(A) \leqslant 10 \log_2 h$ and $\mathrm{mimsup}(A) \geqslant \sqrt{h}$.*

In order to ensure that a matrix has no large half-induced matching, we will in fact ensure it has no large blocks of zeros. Moreover, we will show that not just the mimsup is large, we already find a large induced matching in the second Kronecker power. This lemma immediately implies the theorem above, since if M has a half-induced matching of size 2ℓ, then it has an $\ell \times \ell$ all zeros square submatrix.

Lemma 4.30. *There is a constant $C > 1$, such that for all $h \geqslant C$, there is a symmetric matrix $A \in \{0,1\}^{2h \times 2h}$ without $2\lceil 2\log_2 h \rceil \times 2\lceil 2\log_2 h \rceil$ blocks of zeros, but satisfies $\mathrm{mim}(A^{\otimes 2}) \geqslant h$.*

Proof. We write $\cdot$ for the Hadamard product: for two $n \times m$ matrices A, B, the Hadamard product $A \cdot B$ is $n \times m$ matrix such that $(A \cdot B)[i,j] = A[i,j] \cdot B[i,j]$. Then, for two $n' \times m'$ submatrices A_1, A_2 of A, we can observe that $A_1 \cdot A_2$ is a submatrix of $A^{\otimes 2}$. Indeed, let $r_1^{(1)}, \ldots, r_{n'}^{(1)}$ (resp., $r_1^{(2)}, \ldots, r_{n'}^{(2)}$) be the row indices of A present in A_1 (resp., A_2) and let $c_1^{(1)}, \ldots, c_{n'}^{(1)}$ (resp., $c_1^{(2)}, \ldots, c_{n'}^{(2)}$) be the column indices of A present in A_1 (resp., A_2). Then for $i \in [n'], j \in [m']$, we have

$$A_1 \cdot A_2[i,j] = A_1[i,j] \cdot A_2[i,j] = A[r_i^{(1)}, c_j^{(1)}] \cdot A[r_i^{(2)}, c_i^{(2)}] = A^{\otimes 2}[(r_i^{(1)}, r_i^{(2)}), (c_i^{(1)}, c_i^{(2)})],$$

and thus the matrix $A_1 \cdot A_2$ is the submatrix of $A^{\otimes 2}$ with rows $(r_1^{(1)}, r_1^{(2)}), \ldots, (r_{n'}^{(1)}, r_{n'}^{(2)})$ and columns $(c_1^{(1)}, c_1^{(2)}), \ldots, (c_{m'}^{(1)}, c_{m'}^{(2)})$.

Let M be a random $(h \times h)$-matrix, with 1's on the diagonal and the entries below the diagonal sampled independently uniformly at random from $\{0,1\}$, where the entries above the diagonal are chosen in order to make a symmetric matrix. Let M' be the "complement" of M: the diagonal entries are still 1 ($M'[i,i] = M[i,i] = 1$) but all other entries are flipped ($M'[i,j] = 1 - M[i,j]$ for $i \neq j$). By construction, $M \cdot M'$ is the $(h \times h)$-identity matrix and M, M' are both symmetric with the same distribution.

We first show that with high probability, M does not contain an $\ell \times \ell$ block of zeros for $\ell = \lceil 2\log_2 h \rceil$. Note that any such block is given by rows $r_1, \ldots, r_\ell$ and columns $c_1, \ldots, c_\ell$. All must be distinct (since the diagonal entries are non-zero). In particular, all $M[r_i, c_j]$ entries are sampled independently. By the union bound, the probability that M has an $\ell \times \ell$ block consisting of only zeros is at most

$$\binom{h}{\ell}^2 \left(\frac{1}{2}\right)^{\ell^2} \leqslant \left(\frac{eh}{\ell}\right)^{2\ell} \left(\frac{1}{2}\right)^{\ell^2}. \tag{4.5.1}$$

Since $\ell \geqslant 2\log_2 h$, the above can be upperbounded (see Appendix for details) by

$$\left(\frac{eh}{2\log_2 h}\right)^{4\log_2 h}\left(\frac{1}{2}\right)^{(2\log_2 h)^2} = \left(\frac{eh}{2\log_2 h}\right)^{4\log_2 h}\left(\frac{1}{h}\right)^{4\log_2 h} = \left(\frac{e}{2\log_2 h}\right)^{4\log_2 h}, \quad (4.5.2)$$

which tends to 0 when h tends to infinity. So for h sufficiently large, such a block does not exist in M with high probability. Since M and M' have the same distribution, the same holds for M'. We set

$$A = \begin{pmatrix} M & M' \\ M' & M \end{pmatrix}.$$

Then A has no $2\ell \times 2\ell$ block of zeros. At the same time, $A^{\otimes 2}$ contains the Hadamard product $M \cdot M'$ as submatrix, which is the $h \times h$ identity matrix. So

$$\operatorname{him}(A \otimes A) \geqslant \operatorname{mim}(A \otimes A)\operatorname{mim}(M \cdot M') = h.$$

We showed that for all h sufficiently large, there is a matrix A without $2\lceil 2\log_2 h\rceil \times 2\lceil 2\log_2 h\rceil$ block of zeros yet $\operatorname{him}(A^{\otimes 2}) \geqslant h$, as desired. □

The property of being a half-induced matching can also be weakened to $M[i,i] = 1$ and $M[i,j] + M[j,i] \leqslant 1$ for all $i \neq j$ ('sub-antisymmetric'). Let $g(M)$ denote the largest submatrix with this property. Then $g(M) \geqslant \operatorname{him}(M)$ and $\operatorname{him}(M) \geqslant \lfloor \log_2(g(M))\rfloor$ by greedily selecting the row r with the most zeros, removing the columns that have a 1 entry in row r. Each time we remove at most half of the remaining columns, so this can continue for at least $\lfloor \log_2(g(M))\rfloor$ steps. This shows these two parameters are functionally equivalent.

4.5.2 Comparing him and support rank

Our algorithm in Lemma 4.6 is based on the size of the largest half-induced matching, while the algorithm in Corollary **??** is based on the support-rank. We next show that him and the support rank are not functionally equivalent. In particular, this shows that bounds from Theorem 1.6 can be significantly better than the ones from Corollary **??** when k is small (when H may be chosen arbitrarily from the infinite family of matrices below).

Theorem 4.31. *For each field $\mathbb{F}$ and integer $t \geqslant 3$, for all n sufficiently large, there exists a symmetric matrix $A \in \{0,1\}^{n\times n}$ with ones on the diagonal, such that*

- $\mathrm{him}(A) \leqslant 2t$, *and*
- *any matrix* $B \in \mathbb{F}^{n\times n}$ *with the same support as* A *has rank at least* $n^{1-2/t}$.

Rather than proving this result directly, we will conclude it from results for a related notion. The *minrank* of a graph G on the set of vertices $[n]$ over a field $\mathbb{F}$, denoted by $\mathrm{minrank}_{\mathbb{F}}(G)$, is the minimum possible rank of a matrix $M \in \mathbb{F}^{n\times n}$ with nonzero diagonal entries such that $M_{i,j} = 0$ whenever i and j are distinct non-adjacent vertices of G. Note that here, in contrary to the support-rank of the adjacency matrix of G, we do not demand that $M_{i,j} \neq 0$ whenever i and j are distinct adjacent vertices of G. The minrank gives an upper bound on the Shannon capacity of the graph (see Section 4.5.3 for definition), but can also be used to give lower bounds on the smallest dimension d for which the graph admits certain geometric representations in $\mathbb{R}^d$, e.g. orthogonal representations, unit distance graphs or touching spheres (see [1, 69, 76]).

If we denote by $A \in \{0,1\}^{n\times n}$ the adjacency matrix of G and write

$$\mathcal{M}(A,\mathbb{F}) = \{M \in \mathbb{F}^{n\times n} \mid A[i,j] = 0 \implies M[i,j] = 0 \text{ for } i \neq j \text{ and } M[i,i] \neq 0 \text{ for all } i\},$$

then the minrank of G is $\min\{\mathrm{rank}(M) \mid M \in \mathcal{M}(A,\mathbb{F})\}$. With

$$\mathcal{M}'(A,\mathbb{F}) = \{M \in \mathbb{F}^{n\times n} \mid M[i,j] = 0 \iff A[i,j] = 0\},$$

the support rank of A is $\min\{\mathrm{rank}(M) \mid M \in \mathcal{M}'(A)\}$. We see that the set $\mathcal{M}(A,\mathbb{F})$ remains the same when all the diagonal entries of A are changed to ones. When A has ones on the diagonal, then $\mathcal{M}'(A,\mathbb{F}) \subseteq \mathcal{M}(A,\mathbb{F})$ and so the support-rank is lower bounded by minrank.

For a graph H with $h \geqslant 3$ vertices, let $m_2(H)$ denote the maximum value of $\frac{|E(H')|-1}{|V(H')|-2}$ over all subgraphs H' of H on at least 3 vertices. We will use the following result of Alon, Balla, Gishboliner, Mond and Mousset [1].

Theorem 4.32 ([1, Theorem 5.3])**.** *Let* H *be a graph with* $h \geqslant 3$ *vertices. Then there is a constant* $c = c(H) > 0$ *such that for every (finite or infinite) field* $\mathbb{F}$ *and every integer* n *there is a graph* G *on* n *vertices whose complement contains no copy of* H*, so that*

$$\mathrm{minrank}_{\mathbb{F}}(G) \geqslant \frac{cn^{1-1/m_2(H)}}{\log_2 n}.$$

Now we can prove Theorem 4.31.

Proof of Theorem 4.31. Let $t \geqslant 3$. For $K_{t,t}$, the complete bipartite graph with sides of size t, it holds that

$$m_2(K_{t,t}) \geqslant \frac{t^2-1}{2(t-1)} = \frac{t+1}{2}.$$

Applying Theorem 4.32 with $H = K_{t,t}$, we find that there is a graph G on n vertices, such that $\overline{G}$ does not contain $K_{t,t}$, and

$$\operatorname{minrank}(G) \geqslant \frac{c(K_{t,t})n^{1-1/m_2(K_{t,t})}}{\log_2 n}.$$

When n is chosen to be sufficiently large,

$$\frac{c(K_{t,t})n^{1-1/m_2(K_{t,t})}}{\log_2 n} > n^{1-2/t}.$$

Let A be the matrix obtained from an adjacency matrix A_G of G by setting all diagonal entries to ones. By our previous discussion, the support rank of A is lower-bounded by the minrank of A, and thus is at least $n^{1-2/t}$.

We claim that $\operatorname{him}(A) < 2t$. Indeed, if A contained a half-induced matching of size $2t$, then there are t rows $r_1, \ldots, r_t$ and t columns $c_1, \ldots, c_t$ that form an all-zero block in A. In particular, this contains no diagonal entries so $r_i \neq c_j$ for all $i, j \in [t]$. But then $\overline{G}[\{r_1, \ldots, r_t, c_1, \ldots, c_t\}]$ has a complete bipartite graph $K_{t,t}$ as subgraph, a contradiction. This completes the proof. □

4.5.3 Comparing mimsup and Shannon capacity

In this section, we shortly discuss the relation between our new parameter and the Shannon capacity [103, 132]. For two graphs G, H, the *strong product* $G \boxtimes H$ has vertex set $V(G) \times V(H)$ and two vertices $(u, x), (v, y)$ are adjacent if one of the following holds: (i) $uv \in E(G)$ and $xy \in E(H)$, or (ii) $uv \in E(G)$ and $x = y$, or (iii) $u = v$ and $xy \in E(H)$. This can be naturally generalized to more factors. Then by $G^{\boxtimes k}$ we denote the strong product of k copies of G. The *Shannon capacity* is defined as

$$\Theta(G) = \limsup_{k \to \infty} \alpha(G^{\boxtimes k}),$$

where $\alpha(G)$ is the independence number of G .

Let G be a graph with line graph $L(G)$, i.e., $V(L(G)) = E(G)$ and $ef \in E(L(G))$ if and only if $e \cap f \neq \emptyset$. Denote by $L(G)^2$ the square of the line graph: the vertex set is $E(G)$ where $e, e' \in E(G)$ are adjacent if $e \cap e''$ and $e' \cap e''$ are both non-empty for some

$e'' \in E(G)$ (possibly $e'' \in \{e, e'\}$). Then there is a one-to-one correspondence between independent sets in $L(G)^2$ and induced matchings of G: two edges $e \neq e'$ can be together in an induced matching if and only if no edge e'' intersects both e and e'. Therefore, we observe the following.

Observation 4.33. *Let G be a graph. Then we have $\alpha(L(G)^2) = \mathrm{mim}(G)$.*

However, $L(G^{\otimes 2})^2$ is not isomorphic to $(L(G)^2)^{\boxtimes 2}$ and in fact $\mathrm{mimsup}(G)$ cannot be upperbounded in terms of $\Theta(L(G)^2)$ as the following example shows. Let H be a half-graph on vertices $v_1, \ldots, v_r$ on one side and $w_1, \ldots, w_r$ on the other, where the edges are $v_i w_j$ for $i \leqslant j$. Consider two edges $e = v_i w_j$ and $f = v_s w_r$ (so $i \leqslant j$ and $s \leqslant r$). If $i = s$ or $j = r$, then ef is an edge in $L(H)$, and thus in $L(H)^2$. Otherwise, without loss of generality assume that $i < s \leqslant r$. Then there is an edge $v_i w_r$ in H, and thus ef is an edge in $L(H)^2$. So $(L(H))^2$ is the complete graph on $\frac{r(r+1)}{2}$ vertices. Any strong product power $((L(H))^2)^{\boxtimes k}$ is also a complete graph, so the Shannon capacity of $(L(H))^2$ is 1, while $\mathrm{mimsup}(H) \geqslant \mathrm{him}(H) = r$.

On the other hand, we do find the following relationship.

Lemma 4.34. *For any bipartite graph G it holds that $\Theta(L(G)^2) \leqslant \mathrm{mimsup}(G)$.*

Proof. Let G be a bipartite graph with bipartition classes U, V. Let A be the bi-adjacency matrix of G. Let $k \geqslant 1$ be an integer. Recall that $G^{\otimes k}$ is the bipartite graph on vertex set $U^k \cup V^k$ whose bi-adjacency matrix is $A^{\otimes k}$. It suffices to show that

$$\alpha((L(G)^2)^{\boxtimes k}) \leqslant \alpha(L(G^{\otimes k})^2) = \mathrm{mim}(G^{\otimes k}) = \mathrm{mim}(A^{\otimes k}),$$

where the first equality follows from Observation 4.33 and the second is by definition. We prove the inequality above by showing that $G_1 = L(G^{\otimes k})^2$ is isomorphic to a subgraph of $G_2 = (L(G)^2)^{\boxtimes k}$ – note that both graphs have the same number of vertices, which is $|E(G)|^k$, so in fact we will show that G_2 is isomorphic to a graph obtained by adding some (possibly zero) edges to G_1, and thus every independent set in G_2 is also an independent set in G_1.

The vertices of G_1 are of the form $\{u, v\}$ with $u = (u_1, \ldots, u_k) \in U^k$ and $v = (v_1, \ldots, v_k) \in V^k$ and $e_i = u_i v_i \in E(G)$ for all $i \in [k]$. We let $\mu(\{u, v\}) = (e_1, \ldots, e_k) \in V((L(G)^2)^{\boxtimes k})$. Then $\mu : V(G_1) \to V(G_2)$ is an injective function. It remains to show that $\mu(e)\mu(f) \in E(G_2)$ when $ef \in E(G_1)$. Let $ef \in E(G_1)$, and let $e = \{u, v\}$ and $f = \{u', v'\}$

for $u, u' \in U^k$ and $v, v' \in V^k$. Since $ef \in E(L(G^{\otimes k})^2)$, either $\{u, v'\}$ or $\{u', v\}$ must be an element of $E(G^{\otimes k})$ (note that this is also the case if $u = u'$ or $v = v'$). By symmetry, we may assume $g = \{u, v'\} \in E(G^{\otimes k})$. But by definition, that means that for all $i \in [k]$, $g_i := u_i v_i' \in E(G)$ is incident with both $e_i := u_i v_i \in E(G)$ and $f_i := u_i' v_i' \in E(G)$. So $(e_1, \ldots, e_k)$ is adjacent to $(f_1, \ldots, f_k)$ in G_2. The claim now follows as $\mu(e) = (e_1, \ldots, e_k)$ and $\mu(f) = (f_1, \ldots, f_k)$. □

Let us point out that since for non-bipartite graphs H we defined $\mathrm{mimsup}(H) = \mathrm{mimsup}(H^*)$, Lemma 4.34 implies that for every non-bipartite graph H, it holds that $\Theta(L(H^*)^2) \leqslant \mathrm{mimsup}(H)$.

4.5.4 Support rank, covering by bicliques, and Prague dimension

In this section we compare the support-rank with parameters cov (introduced in Section 4.2.3) and the so-called *Prague dimension*.

Let H be a graph on n vertices. The *Prague dimension* of a graph H (sometimes also called *product dimension* or just *dimension*) introduced by Nešetřil, Pultr, and Rödl [118, 119] is the least integer p such that there exist integers $n_1, \ldots, n_p$ for which H is an induced subgraph of the direct product of p cliques, i.e., $K_{n_1} \times \ldots \times K_{n_p}$. Note that without loss of generality we may assume that $n_1 = n_2 = \ldots = n_p = n$. We denote the Prague dimension of H by $\dim(H)$. Note that for two vertices $u = (u_1, \ldots, u_p), v = (v_1, \ldots, v_p)$ of $K_n \times \ldots \times K_n$, they are adjacent if and only if $u_i \neq v_i$ for every $i \in [p]$. Therefore equivalently we can say that $\dim(H) \leqslant p$ if we can encode each vertex of H as a sequence of length p so that the vertices are adjacent if and only if their corresponding sequences differ on every coordinate.

Theorem 4.35. *Let H be a bipartite graph.*

(1) There exists a family $\mathcal{B}$ that $\dim(H)$-covers H^c.

(2) If H^c can be r-covered, then $\dim(H) \leqslant r^2 + 2$.

Proof of (1). Let $n = |V(H)|$, $r = \dim(H)$, and let $\mu : V(H) \to [n]^r$ be a mapping whose image induces a copy of H in r-fold direct product of K_n with vertex set $[n]$; it exists by the definition of $\dim(H)$. For $i \in [r], j \in [n]$, let $B_{i,j}$ be the subgraph of H^c induced by

these vertices $v \in V(H)$ for which the projection of $\mu(v)$ to the i-th coordinate is equal to j. We define $\mathcal{B} = \{B_{i,j} \mid i \in [r], j \in [n]\}$.

Observe that all vertices of $B_{i,j}$ are pairwise non-adjacent in H since for all tuples $\mu(v)$ their i-th coordinate is the same. Therefore $B_{i,j}$ is a biclique of H^c.

Now let us verify that $\mathcal{B}$ r-covers H^c. First consider a vertex $v \in V(H^c) = V(H)$ and let $\mu(v) = (v_1, \ldots, v_p)$. The vertex v is contained only in bicliques B_{i,v_i} for $i \in [r]$, so v is in at most r bicliques. So now consider any edge uv of H^c, and let $\mu(u) = (u_1, \ldots, u_r)$ and $\mu(v) = (v_1, \ldots, v_r)$. By the definition of H^c, we have that $uv \notin E(H)$ and therefore there exists $i \in [r]$ such that $u_i = v_i$. Thus both u, v are contained in the biclique B_{i,u_i}. This completes the proof. □

Proof of (2). Let the bipartition classes of H be X, Y. Let $\mathcal{B}$ be a family of bicliques that r-covers H^c and let $B_1, \ldots, B_s$ be fixed arbitrary ordering of $\mathcal{B}$. Define $I = [r]^2 \cup \{(0,1),(1,0)\}$, which will be our set of indices. Note that $|I| = r^2 + 2$. Let $q = \max(|V(H)|, s+2)$. We aim to define a mapping $\mu : V(H) \to [q]^{|I|}$ which will define an induced copy of H in $|I|$-fold direct product of K_q with vertex set $[q]$ (here it will be convenient not to assume that the cliques have $|V(H)|$ vertices).

Fix a pair $(i,j) \in [r]^2$ and consider $u \in X$ (resp. $v \in Y$). Let u (resp. v) be covered by $r' \leqslant r$ bicliques in $\mathcal{B}$. We define

$$\mu_{i,j}(u) = \begin{cases} \ell & \text{if } i \leqslant r' \text{ and } B_\ell \text{ is the} \\ & i\text{-th biclique covering } u \\ s+1 & \text{if } i > r', \end{cases} \qquad \mu_{i,j}(v) = \begin{cases} \ell & \text{if } j \leqslant r' \text{ and } B_\ell \text{ is the} \\ & j\text{-th biclique covering } v \\ s+2 & \text{if } i > r'. \end{cases}$$

Moreover, we define $\mu_{1,0} : V(H) \to [q]$ so that each vertex receives distinct value (this is possible as $q \geqslant |V(H)|$), and $\mu_{0,1} : V(H) \to [q]$ that maps all vertices from X to 1 and all vertices from Y to 2.

Now, $\mu : V(H) \to [q]^I$ is defined in a way that the projection of μ on the coordinate $(i,j) \in I$ is precisely $\mu_{i,j}$. This completes the definition of μ.

Let us verify that the image $\mu(V(H))$ induces a copy of H. By the definition of $\mu_{1,0}$, the mapping is injective and by the definition of $\mu_{0,1}$ for u, v from the same bipartition class we have that $\mu(u)$ and $\mu(v)$ are non-adjacent in $K_q \times \ldots \times K_q$ (we remark that $\mu_{1,0}$ is not needed if all vertices have pairwise distinct neighborhoods).

Consider $uv \in E(H)$ with $u \in X$, $v \in Y$. This means that $uv \notin E(H^c)$ and thus there is no biclique that contains both u, v. Therefore $\mu(u)$ and $\mu(v)$ differ on every coordinate and hence are adjacent.

So now consider $uv \notin E(H)$ with $u \in X$, $v \in Y$. Since $uv \in E(H^c)$, there must be at least one biclique $B_\ell \in \mathcal{B}$ that contains uv. Therefore for some pair (i, j) we defined $\mu_{i,j}(u) = \mu_{i,j}(v) = \ell$, so $\mu(u)$ is non-adjacent to $\mu(v)$. This completes the proof. □

We conclude this section with the following bound.

Corollary 4.36. *The following bounds hold.*

1. *If H is not bipartite, then the support rank of the adjacency matrix of H is at most $(\dim(H^*) + 1)^{\dim(H^*)}$.*

2. *If H is bipartite, then the support rank of the bi-adjacency matrix of H is at most $(\dim(H) + 1)^{\dim(H)}$.*

Proof. The first statement is an immediate corollary of Lemma 4.9 and Theorem 4.35 (1). We point out that Lemma 4.9 holds, if we consider covering H^c instead of $(H^*)^c$ and the bi-adjacency matrix of H instead the adjacency matrix. Therefore, the second statement follows as well. □

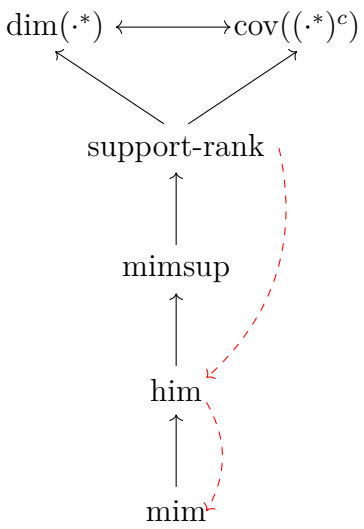

Figure 4.5: Relations between the parameters considered in Section 4.5. Let us point out that for a non-bipartite graph H, we compare parameters mim, him, and mimsup of H with (i) the support rank of the adjacency matrix A_H of H, (ii) the dimension of H^*, and (iii) the covering number of the bipartite complement $(H^*)^c$ of H^*. A black arrow from a parameter p_1 to a parameter p_2 denotes that p_1 can be bounded by a function of p_2. A dashed red arrow from a parameter p_1 to a parameter p_2 denotes that p_1 cannot be bounded by a function of p_2. For some pairs of parameters, we do not know if they are functionally equivalent, i.e., we do not know if: (i) mimsup can be bounded by some function of him, (ii) support-rank can be bounded by some function of mimsup, (iii) dim can be bounded by some function of support-rank.

Chapter 5

Diameter

In this chapter we consider the diameter as the parameter of the input graph. Throughout this chapter, whenever H is a target graph, we will always consider it as fixed, and thus its size will be always constant. Let us first define some additional notions and present basic tools used in this chapter.

Cycles. Whenever C_{2k+1} is the target graph, we will denote its vertex set by $[2k]_0$, unless explicitly stated otherwise. Moreover, whenever we refer to the vertices of the $(2k+1)$-cycle, i.e., cycle on $2k+1$ vertices, by $+$ and $-$ we denote respectively the addition and the subtraction modulo $2k+1$. We say that a list $L(v)$ is *of type* $(\ell_1, \ldots, \ell_r)$ if $|L(v)| = r+1$ and its vertices can be ordered $c_0, \ldots, c_r$ so that for every $i \in [r-1]_0$, we have that $c_{i+1} = c_i + \ell_i$. For example, for $k \geqslant 4$, one of the types of the list $\{1, 4, 6, 7\}$ is $(3, 2, 1)$.

Subinstances. Let H be a graph, let (G, L) be an instance of the LHOM(H) problem, and let x, y be two distinct vertices of G. By *identifying x and y into z* we mean the operation of adding a new vertex z to the graph, making it adjacent to every vertex in $N(u) \cup N(v)$, and removing u and v. We will always identify non-adjacent vertices. The list of z becomes equal to $L(x) \cap L(y)$. The following observation is straightforward.

Observation 5.1. *Let G be a graph, let $x, y \in V(G)$, and let G' be the graph obtained from G by identifying x and y. Then* $\operatorname{diam}(G') \leqslant \operatorname{diam}(G)$.

For two instances (G, L) and (G', L') of LHOM(H), we say that (G', L') is a *subinstance* of (G, L) if it can be obtained from (G, L) by a (possibly empty) sequence consisting of

two types of operations: (i) removing some colors from the list of a vertex, and (ii) identifying two non-adjacent vertices with the same list. Intuitively, a subinstance is formed by a series of decisions: that some vertex will *not* be colored with certain color (this corresponds to operation (i)), and that some pair of vertices will receive the same color (this corresponds to operation (ii)). Note that if (G, L) is a no-instance of LHOM(H), then each subinstance of (G, L) is a no-instance as well.

Observe that if (G', L') is a subinstance of (G, L), then there is a surjective mapping $\mathsf{trace} : V(G) \to V(G')$, which can be defined as follows. Let $(G_1, L_1), (G_2, L_2), \ldots, (G_\ell, L_\ell)$ be a sequence of subinstances such that $(G_1, L_1) = (G, L)$, $(G_\ell, L_\ell) = (G', L')$, and for $i \in [\ell-1]$, the instance (G_{i+1}, L_{i+1}) can be obtained from (G_i, L_i) by a single application of one of the operations (i) and (ii). For $i \in [\ell]$, we define $\mathsf{trace}_i : V(G) \to V(G_i)$ recursively as follows.

1. We set trace_1 to be the identity function.
2. If $i > 1$ and (G_i, L_i) is obtained from (G_{i-1}, L_{i-1}) by operation (i), then $V(G_i) = V(G_{i-1})$, and for every $v \in V(G)$, we set $\mathsf{trace}_i(v) = \mathsf{trace}_{i-1}(v)$.
3. If $i > 1$ and (G_i, L_i) is obtained from (G_{i-1}, L_{i-1}) by identifying vertices x, y into z, then for $v \in V(G)$, if $\mathsf{trace}_{i-1}(v) \in \{x, y\}$, we set $\mathsf{trace}_i(v) = z$, otherwise we set $\mathsf{trace}_i(v) = \mathsf{trace}_{i-1}(v)$.

Finally we can define $\mathsf{trace} = \mathsf{trace}_\ell$.

Let (G, L), (G', L'), and trace be as above. A list homomorphism $\varphi : (G, L) \to H$ is *compatible* with (G', L') if

- for each $x, y \in V(G)$, if $\mathsf{trace}(x) = \mathsf{trace}(y)$, then $\varphi(x) = \varphi(y)$,
- for each $x \in V(G)$ it holds that $\varphi(x) \in L'(\mathsf{trace}(x))$.

Intuitively, φ is compatible with (G', L') if it agrees will all decisions we made when forming (G', L') from (G, L).

Reduction rules. Let H be a graph and let (G, L) be an instance of LHOM(H). We define the following reduction rules.

(R1) If there is $v \in V(G)$ such that $L(v) = \emptyset$, then return NO.

(R2) For a vertex $x \in V(H)$, and vertices $u, v \in V(G)$ such that $L(u) = L(v) = \{x\}$, if $uv \in E(G)$, then return NO, otherwise identify u with v.

(R3) For every $uv \in E(G)$, if there is $x \in L(u)$ such that $N_H(x) \cap L(v) = \emptyset$, then remove x from $L(u)$.

Clearly, each of the above reduction rules can be applied in polynomial time, we can bound this time by $\mathcal{O}(|G|^2)$. Moreover, in each reduction rule we either decrease the number of vertices, decrease the size of a list, or return NO – therefore the reduction rules can be applied at most $\mathcal{O}(|G|)$ times in total. Thus the exhaustive application of the reduction rules to an instance of LHOM(H) with n vertices can be performed in time $\mathcal{O}(n^3)$.

Let us consider a special case of LHOM(H), i.e., LIST 3-COLORING. Then the reduction rule (R3) can be read as follows:

(R3) if there is $v \in V(G)$ such that $L(v) = \{a\}$, then remove a from $L(u)$ for every $u \in N_G(v)$.

Let us verify that the reduction rules are safe.

Lemma 5.2. *After applying each reduction rule to an instance (G, L) of LHOM(H), we obtain an equivalent instance with diameter at most* $\operatorname{diam}(G)$.

Proof. (R1) If any list of a vertex is an empty set, then we are dealing with a no-instance and thus (R1) is safe.

(R2) If two vertices have the same one-element list, then they must be mapped to the same vertex. Recall that we only consider loopless graphs H, adjacent vertices of G cannot be mapped to the same vertex. Therefore, if two vertices u, v of G have lists $L(v) = L(u) = \{x\}$ for some $x \in V(H)$, then, if $uv \in E(G)$, we are dealing with a no-instance. Otherwise, we can identify u and v and thus (R2) is safe.

(R3) Let $uv \in E(G)$ be such that there is $x \in L(u)$ such that $N(x) \cap L(v) = \emptyset$. Suppose that there is a list homomorphism $\varphi : (G, L) \to C_{2k+1}$ such that $\varphi(u) = x$. Then v must be mapped to a vertex from $N(x) \cap L(v) = \emptyset$, a contradiction. Thus we can safely remove x from $L(u)$, and (R3) is safe. □

An instance (G, L) for which none of the reduction rules (R1)–(R3) can be applied is called *reduced*. Note that after applying the reduction rules we obtain a subinstance of the original instance.

Layer structure. Let (G, L) be a reduced instance of LHOM(H). For $i \in [|H|]$, let V_i be the set of vertices v of G, such that $|L(v)| = i$. We also define $V_{\geqslant i} = \bigcup_{j \geqslant i} V_j$. Sometimes we will refer to vertices of V_1 as *precolored vertices* as their colors are uniquely determined by the lists. Note that $(V_1, \ldots, V_{|H|})$ is a partition of $V(G)$; we will call it the *layer structure* of (G, L). If H is a cycle, then since (R3) cannot be applied to a reduced instance, we can observe the following.

Observation 5.3. *Let $\ell \geqslant 3$ and let (G, L) be a reduced instance of LHOM(C_ℓ). Then $N(V_1) \subseteq V_2$, i.e., there are no edges between V_1 and V_i for $i \geqslant 3$.*

Binary CSP and 2-Sat. The Binary Constraint Satisfaction Problem, denoted by BCSP, is a special case of CSP(q, r) where $r = 2$. It will be convenient for us to (i) treat the set of constraints as a function, i.e., $C : V \times V \to 2^{D \times D}$, and (ii) consider instances which are equipped with a list function $L : V \to 2^D$, although L can be easily encoded in C by removing from $C(v, v)$ pairs $(a, a) \in D \times D$ such that $a \notin L(v)$. Therefore, for a given set (domain) D, an instance of BCSP consists of a set V of variables, a list function $L : V \to 2^D$, and a constraint function $C : V \times V \to 2^{D \times D}$. The task is to determine whether there exists an assignment $f : V \to D$ such that for every $v \in V$, we have $f(v) \in L(v)$ and for every pair $(u, v) \in V \times V$, we have $(f(u), f(v)) \in C(u, v)$.

Clearly, any instance of LHOM(H) can be seen as an instance of BCSP, where the domain D is $V(H)$, the list function remains the same, and for every edge $uv \in E(G)$ we set $C(u, v) = \{(x, y) \mid xy \in E(H)\}$ and for every $uv \notin E(G)$ we set $C(u, v) = V(H) \times V(H)$. We will denote by BCSP(H, G, L) the instance of BCSP corresponding to the instance (G, L) of LHOM(H).

Edwards [50] proved that an instance of LIST q-COLORING with all lists of size at most two can be solved in polynomial time by a reduction to 2-SAT. In fact, with the same approach, this result can be generalized to instances of BCSP with all list of size at most two.

Theorem 5.4 (Edwards [50])**.** *Let (V, L, C) be an instance of BCSP over the domain D. Assume that for every $v \in V$ it holds $|L(v)| \leqslant 2$. Then we can solve the instance (V, L, C) in polynomial time.*

Proof. We will construct in polynomial time an instance ϕ of 2-SAT with at most $2|V|$ variables and such that ϕ is a yes-instance of 2-SAT if and only if (V, L, C) is a yes-instance

of BCSP. Since 2-SAT is polynomial-time solvable [95], the statement will follow.

Note that we can assume that every list is of size exactly two. Indeed, if there is $v \in V$ with $L(v) = \emptyset$, we immediately can conclude that (V, L, C) is a no-instance. Furthermore, for v with $L(v) = \{a\}$, for every $u \in V \setminus \{v\}$, and for every $b \in L(u)$ such that $(a, b) \notin C(v, u)$, we can remove b from $L(u)$, and then we can remove v from V. We can repeat these operations until all variables have lists of size two or we return NO.

For each $v \in V$, we arbitrarily fix the order of the two elements of $L(v)$. For every v, we introduce two variables, $x_{v,1}$ and $x_{v,2}$, which correspond, respectively, to setting v to the first and the second element of $L(v)$. We also introduce clauses $(x_{v,1} \vee x_{v,2})$ and $(\neg x_{v,1} \vee \neg x_{v,2})$. Finally, for every pair $(u, v) \in V^2$, for every $(a, b) \in L(u) \times L(v) \setminus C(u, v)$ such that a is the ith element of $L(u)$ and b is the jth element of $L(v)$, we introduce the clause $(\neg x_{v,i} \vee \neg x_{u,j})$. The formula ϕ is obtained by taking the conjuction of all introduced clauses.

Observe that in any satisfying truth assignment of the variables of ϕ, for every $v \in V$, exactly one of the variables $x_{v,1}$, $x_{v,2}$ must be set to true. Indeed, since the clause $(x_{v,1} \vee x_{v,2})$ has to be satisfied, at least one of them must be set to true, and since the clause $(\neg x_{v,1} \vee \neg x_{v,2})$ has to be satisfied, at most one of them can be set to true.

Clearly, ϕ is an instance of 2-SAT as each clause contains precisely two literals. Moreover, ϕ has at most $2|V|$ variables and the construction of ϕ can be performed in polynomial time. Let us verify the equivalence of instances. First assume that there is a satisfying truth assignment ψ of the variables of ϕ. Let us construct a satisfying assignment $f : V \to D$. Let $v \in V$, let $L(v) = \{a_1, a_2\}$ and assume that a_1 is the first element of v. If $\psi(x_{v,1}) = 1$, then set $f(v) = a_1$, otherwise set $f(v) = a_2$. Clearly, f respects the lists L. Let $(u, v) \in V^2$ and suppose that $(f(u), f(v)) \notin C(u, v)$. Let $f(u)$ be the ith element of $L(u)$ and let $f(v)$ be the jth element of $L(v)$. Recall that in such case we introduced a clause $(\neg x_{u,i} \vee \neg x_{v,j})$. On the other hand, since we set u and v to, respectively, ith element of $L(u)$ and jth element of $L(v)$, we must have $\psi(x_{u,i}) = 1$ and $\psi(x_{v,j}) = 1$, a contradiction.

So now assume that there is a satisfying assignment $f : V \to D$. We define the truth assignment ψ of the variables of ϕ as follows. For every $v \in V$, if $f(u)$ is the first element of $L(v)$, we set $\psi(x_{v,1}) = 1$ and $\psi(x_{v,2}) = 0$. Otherwise, we set $\psi(x_{v,1}) = 0$ and $\psi(x_{v,2}) = 1$. Let us verify that ψ satisfies ϕ. Since for every $v \in V$, we set precisely one of $x_{v,1}$ and

$x_{v,2}$ to true, all clauses of type $(x_{v,1} \vee x_{v,2})$ and $(\neg x_{v,1} \vee \neg x_{v,2})$ are satisfied. It remains to consider a clause $(\neg x_{u,i} \vee \neg x_{v,j})$ introduced for a pair (u, v) and a pair $(a, b) \in L(u) \times L(v)$ such that a, b are, respectively, the ith element of $L(u)$ and the jth element of $L(v)$, and $(a, b) \notin C(u, v)$. Since f is a satisfying assignment, we have $(f(u), f(v)) \neq (a, b)$, and, by the definition, at least one of $x_{u,i}$, $x_{v,j}$ must be set to false. Thus the clause $(\neg x_{u,i} \vee \neg x_{v,j})$ is satisfied, which completes the proof. □

Clearly, Theorem 5.4 in particular implies that an instance (G, L) of LHOM(H) with all lists of size at most 2 can be solved in polynomial time.

5.1 List 3-Coloring

In this section we study the complexity of LIST 3-COLORING, i.e., LHOM(K_3), on diameter-2 and -3 graphs, and we prove Theorem 1.12 and Theorem 1.11.

The following proposition describes important properties of layer structures of graphs with diameter at most 3.

Proposition 5.5. *Let (G, L) be a reduced instance of LIST 3-COLORING, where G has diameter $d \leqslant 3$, and let (V_1, V_2, V_3) be the layer structure of G. Then, for any $u, v \in V_2 \cup V_3$, at least one of the following holds:*

a) u and v are at distance at most d in $G[V_2 \cup V_3]$, or

b) $\{u, v\} \cap V_2 \neq \emptyset$.

Proof. If $V_1 = \emptyset$, then the first outcome follows, since $G = G[V_2 \cup V_3]$. So assume that $V_1 \neq \emptyset$. Consider $u, v \in V_3$ and suppose that they are not at distance at most d in $G[V_2 \cup V_3]$. Since they are at distance at most d in G, all shortest u-v-paths in G must intersect V_1. However, by Observation 5.3, for any $x \in V_1$, it holds that $\text{dist}(u, x) \geqslant 2$ and $\text{dist}(v, x) \geqslant 2$. Thus $\text{dist}(u, v) \geqslant 4$, contradicting the fact that $\text{diam}(G) \leqslant 3$. □

Observe that Proposition 5.5 does not generalize to diameter-4 graphs: consider, e.g., a 5-vertex path P_5 with consecutive vertices v_1, v_2, v_3, v_4, v_5, where $V_1 = \{v_3\}$. Vertices v_1 and v_5 are in V_3, they are at distance 4 in P_5, but not in $P_5[V_2 \cup V_3] = P_5 - \{v_3\}$.

Proposition 5.5 immediately yields the following corollary.

Corollary 5.6. *Let (G, L) be a reduced instance of* List 3-Coloring*, where G has diameter $d \in \{2, 3\}$, and let (V_1, V_2, V_3) be the layer structure of (G, L). For every $v \in V_3$, the set $N^{\leqslant d-1}_{G[V_2 \cup V_3]}[v]$ dominates V_3.*

5.1.1 Diameter-3 graphs

In this section we present a simple proof of Theorem 1.11. Actually, we will show the following more general result, which also yields a $2^{\mathcal{O}(\sqrt{n \log n})}$-algorithm for diameter-2 graphs; it implies the result of Mertzios and Spirakis [111], but the proof is different than the original one. This will serve as a warm-up before showing the more complicated proof of Theorem 1.12.

Theorem 5.7. *The* List 3-Coloring *problem on n-vertex graphs G can be solved in time:*

1. $2^{\mathcal{O}(n^{1/2} \log^{1/2} n)}$, *if* $\operatorname{diam}(G) = 2$,
2. $2^{\mathcal{O}(n^{2/3} \log^{2/3} n)}$, *if* $\operatorname{diam}(G) = 3$.

Proof. Let (G, L) be an instance of List 3-Coloring, where G has n vertices and diameter $d \in \{2, 3\}$, and let (V_1, V_2, V_3) be the layer structure of (G, L). Without loss of generality we may assume that (G, L) is reduced. If $V_3 = \emptyset$, then the problem can be solved in polynomial time using Theorem 5.4. Thus let us assume that $V_3 \neq \emptyset$. Let us define a measure $\mu := 2|V_2| + 3|V_3|$.

First, consider the case that there is a vertex $v \in V_2 \cup V_3$ with at least $(\mu \log \mu)^{1/d}$ neighbors in $V_2 \cup V_3$. Since each vertex of $V_2 \cup V_3$ has one of four possible lists – $\{1, 2\}$, $\{1, 3\}$, $\{2, 3\}$, and $\{1, 2, 3\}$ – there is a subset of at least $\frac{(\mu \log \mu)^{1/d}}{4}$ neighbors of v that all have the same list L'. Note that there is $a \in L(v) \cap L'$ since both are subsets of size at least 2 of a set of size 3. We branch on coloring the vertex v with color a or not. In other words, in the first branch we remove from $L(v)$ all elements but a, and in the other one we remove a from $L(v)$. Note that in the first branch, after reducing the obtained instance, at least $\frac{(\mu \log \mu)^{1/d}}{4}$ vertices will lose at least one element from their list.

We can bound the number of instances produced by applying this step exhaustively as follows:

$$F(\mu) \leqslant F\left(\mu - \frac{(\mu \log \mu)^{1/d}}{4}\right) + F(\mu - 1).$$

Solving this inequality, we obtain that $F(\mu) = \mu^{\mathcal{O}\left(\frac{\mu}{(\mu \log \mu)^{1/d}}\right)} = 2^{\mathcal{O}\left((\mu \log \mu)^{1-1/d}\right)}$. For more details on how we solve this and other recursions, we refer the reader to the Appendix at the end of the dissertation.

Now consider the remaining case that $\Delta(G[V_2 \cup V_3]) < (\mu \log \mu)^{1/d}$. Recall that $V_3 \neq \emptyset$; pick any vertex $v \in V_3$. Define $X := N^{\leqslant d-1}_{G[V_2 \cup V_3]}[v]$; by Corollary 5.6, the set X dominates V_3. Furthermore

$$|X| \leqslant 1 + \Delta(G[V_2 \cup V_3])^{d-1} = \mathcal{O}((\mu \log \mu)^{(d-1)/d}).$$

We exhaustively guess the coloring of X, which results in at most $3^{|X|} = 2^{\mathcal{O}\left((\mu \log \mu)^{1-1/d}\right)}$ branches. As X dominates V_3, after applying the reduction rule (R3) to every vertex of X, in each branch there are no vertices with three-element lists. Therefore, the instance obtained in each of the branches is solved in polynomial time using Theorem 5.4. The total running time comes from multiplying the bounds obtained in both cases and is bounded by $2^{\mathcal{O}\left((\mu \log \mu)^{1-1/d}\right)} \cdot n^{\mathcal{O}(1)} = 2^{\mathcal{O}\left((\mu \log \mu)^{1-1/d}\right)}$. The claimed bound follows since $\mu \leqslant 3n$. □

5.1.2 Diameter-2 graphs

In this section we prove Theorem 1.12. Let us recall the following variant of the Chernoff concentration bound.

Theorem 5.8 ([109, Theorem 2.3])**.** *Let $X_1, \ldots, X_n$ be independent random variables with $0 \leqslant X_i \leqslant 1$ for each i. Let $X = \sum_{i=1}^{n} X_i$ and $\overline{X} = \mathsf{E}[X]$.*

(1) For any $\varepsilon > 0$,

$$\mathsf{Pr}\left(X \geqslant (1+\varepsilon)\overline{X}\right) \leqslant e^{-\frac{\varepsilon^2 \overline{X}}{2(1+\varepsilon/3)}}.$$

(2) For any $\varepsilon > 0$,

$$\mathsf{Pr}\left(X \leqslant (1-\varepsilon)\overline{X}\right) \leqslant e^{-\frac{\varepsilon^2 \overline{X}}{2}}.$$

It will be more convenient to work with random variables for which we only know bounds on the expected value. For this reason we will use the following corollary of Theorem 5.8.

Corollary 5.9. *Let $X_1, \ldots, X_n$ be independent random variables with $0 \leqslant X_i \leqslant 1$ for each i. Let $X = \sum_{i=1}^{n} X_i$.*

(1) For any $\varepsilon > 0$ and $\widehat{X} \geqslant \mathsf{E}[X]$,

$$\mathsf{Pr}\left(X \geqslant (1+\varepsilon)\widehat{X}\right) \leqslant e^{-\frac{\varepsilon^2\widehat{X}}{2(1+\varepsilon/3)}}.$$

(2) For any $\varepsilon > 0$ and $\widehat{X} \leqslant \mathsf{E}[X]$,

$$\mathsf{Pr}\left(X \leqslant (1-\varepsilon)\widehat{X}\right) \leqslant e^{-\frac{\varepsilon^2\widehat{X}}{2}}.$$

Proof. Clearly, if $\widehat{X} = \mathsf{E}[X]$, then (1) and (2) follow directly from Theorem 5.8. So since now assume that this is not the case. In order to prove (1) let us consider a random variable $Y = X + Y_1 + Y_2 + \ldots + Y_k$, where $k = \left\lceil \widehat{X} - \mathsf{E}[X] \right\rceil$ and each Y_i is a constant equal to $\frac{\widehat{X}-\mathsf{E}[X]}{k}$. Clearly Y can be written as a sum of independent random variables $Y'_1, \ldots, Y'_r$ such that $0 \leqslant Y'_i \leqslant 1$. Furthermore,

$$\mathsf{E}[Y] = \mathsf{E}[X + Y_1 + Y_2 + \ldots + Y_k] = \mathsf{E}[X] + \sum_{i=1}^{k} E[Y_i] = \mathsf{E}[X] + k \cdot \frac{\widehat{X} - \mathsf{E}[X]}{k} = \widehat{X},$$

and $Y \geqslant X$, so by Theorem 5.8 (1), we have

$$\mathsf{Pr}\left(X \geqslant (1+\varepsilon)\widehat{X}\right) \leqslant \mathsf{Pr}\left(Y \geqslant (1+\varepsilon)\widehat{X}\right) \leqslant e^{-\frac{\varepsilon^2\widehat{X}}{2(1+\varepsilon/3)}},$$

,

and thus (1) follows.

For (2), we can define $Y = X \cdot \frac{\widehat{X}}{\mathsf{E}[X]}$ – again, since $\widehat{X} \leqslant \mathsf{E}[X]$, the random variable Y can be written as a sum $\sum_{i=1}^{n} Y'_i$, where $Y'_i = X_i \cdot \frac{\widehat{X}}{\mathsf{E}[X]}$ and thus $0 \leqslant Y'_i \leqslant 1$. Furthermore, the expected value of Y is equal

$$E[Y] = \mathsf{E}\left[X \cdot \frac{\widehat{X}}{\mathsf{E}[X]}\right] = \frac{\widehat{X}}{\mathsf{E}[X]} \cdot \mathsf{E}[X] = \widehat{X},$$

and since $Y \leqslant X$, by Theorem 5.8 (2), we obtain that

$$\mathsf{Pr}\left(X \leqslant (1-\varepsilon)\widehat{X}\right) \leqslant \mathsf{Pr}\left(Y \leqslant (1-\varepsilon)\widehat{X}\right) \leqslant e^{-\frac{\varepsilon^2\widehat{X}}{2}},$$

which completes the proof. □

We start with a technical lemma that is the crucial ingredient of our algorithm.

Lemma 5.10. *There exists an absolute constant K such that the following is true. Let G be a 3-colorable graph with n vertices such that*

(i) $\Delta(G) \leqslant n^{2/3}$,

(ii) for every $v \in V(G)$, *the set* $N_G^{\leqslant 2}(v)$ *contains at least* $n - \frac{1}{36}n^{2/3}$ *vertices,*

(iii) for every two vertices $u, v \in V(G)$ *there are at most* $n^{2/3}$ *vertices* w *such that* $N_G(u) \cap N_G(v) \cap N_G(w) \neq \emptyset$.

Let ϕ *be a proper* 3*-coloring of* G, *where* $a \in [3]$ *is the color that appears most frequently. Define* $A := \phi^{-1}(a)$. *Then there exist sets* $S \subseteq A$ *and* $\widetilde{S} \subseteq V(G) \setminus A$, *each of size at most* $K \cdot n^{1/3} \log n$, *such that* $S \cup \widetilde{S} \cup \left(N(S) \cap N(\widetilde{S})\right)$ *dominates at least* $\frac{n}{6}$ *vertices.*

Before we prove Lemma 5.10, let us explain its purpose. Suppose that G is a graph with diameter at most 2 and we are trying to find a list 3-coloring of G under the promise that it exists. Note that if we correctly guess a set S of vertices of the most frequent color a and a set $\widetilde{S}$ of vertices together with its coloring using colors $[3] \setminus \{a\}$, then we can deduce the color of each vertex in $N(S) \cap N(\widetilde{S})$. Hence, our reduction rules will remove at least one color from the list of each vertex in V_3 dominated by $S \cup \widetilde{S} \cup \left(N(S) \cap N(\widetilde{S})\right)$. If the sets S and $\widetilde{S}$ are as in the lemma, then we have just removed at least $\frac{n}{6}$ colors from all the lists by guessing the coloring of only $\mathcal{O}(n^{1/3} \log n)$ vertices. This is roughly why our algorithm is much faster than an exhaustive search.

The assumptions of the lemma can be read as follows: (i) vertices in G do not have too many neighbors, (ii) G is almost a graph with diameter 2 and (iii) common neighbors of every two vertices u and v do not dominate too many vertices of the graph. As we will see later, these assumptions arise naturally when trying to solve the problem using simple branching rules – if any of them is violated, then searching for a list 3-coloring of G becomes easier because of other reasons.

Proof of Lemma 5.10. Note that we can assume that $n \geqslant n_0$, where n_0 is a constant that implicitly follows from the reasoning below. Indeed, otherwise it is sufficient to set $K := n_0$, $S := A$, and $\widetilde{S} := V(G) \setminus A$. Thus from now on we assume that n is sufficiently large.

For every two vertices $u, v \in V(G)$ such that $N[u] \cap N[v] \neq \emptyset$, let x_{uv} be a fixed vertex from $N[u] \cap N[v]$. Fix some vertex $v_a \in A$ and a function $f : N^{\leqslant 2}(v_a) \to N(v_a)$ defined such that $f(u) \in N[u] \cap N(v_a)$.

We start by selecting $\widetilde{S}$ as a subset of neighbors of v_a. For such a set $\widetilde{S}$ we say that a vertex $u \in A$ *threatens* a vertex $w \in A$ if

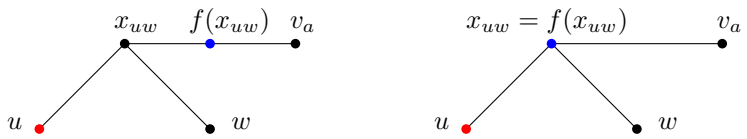

Figure 5.1: The vertex u threatens w: if $f(x_{uw}) \in \widetilde{S}$ and $u \in S$, then w has a neighbor with uniquely determined color. The vertex $f(x_{uw})$ can be either a neighbor of x_{uv} (left) or be the same vertex as x_{uv} (right).

(1) $N[u] \cap N[w] \neq \emptyset$,

(2) $x_{uw} \in N^{\leqslant 2}(v_a)$, and

(3) $f(x_{uw}) \in \widetilde{S}$.

Note that the definition depends on the choice of x_{uv} and f (fixed earlier). Intuitively, u threatens w if selecting u to S would undoubtedly cause w to be dominated by $S \cup \widetilde{S} \cup \left(N(S) \cap N(\widetilde{S})\right)$, see Figure 5.1. The following claim gives us a set $\widetilde{S}$ such that each vertex of A is threatened by many vertices.

Claim 5.10.1. *There exists a set $\widetilde{S} \subseteq N(v_a)$ of size at most $200n^{1/3}\log n$ such that for at least half of vertices $w \in A$ there are at least $8n^{2/3}\log n$ vertices from A that threaten w.*

Proof of Claim: We select $\widetilde{S}$ randomly in such a way that each neighbor of v_a is included in $\widetilde{S}$ independently with probability $\widetilde{p} = 100n^{-1/3}\log n$. We will show that $\widetilde{S}$ satisfies the desired properties with positive probability.

Note that the size of $\widetilde{S}$ is a sum of $\deg(v_a)$ independent random Boolean variables and the expected value of $|\widetilde{S}|$ is $\deg(v_a) \cdot \widetilde{p}$. Recall that by the assumption (i) we have $\deg(v_a) \leqslant n^{2/3}$. Therefore by Corollary 5.9 (1) applied with $\varepsilon = 1$ we deduce that

$$\mathsf{Pr}\left(|\widetilde{S}| > 200n^{1/3}\log n\right) \leqslant e^{-37.5n^{1/3}\log n}.$$

Let $A' \subseteq A$ be the set of those $v \in A$, for which the set $N\left(N(v) \setminus N^{\leqslant 2}(v_a)\right)$ contains fewer than half of vertices from A. We will show that $|A'| \geqslant \frac{1}{2}|A|$. First, let us estimate the number P of ordered pairs of vertices (u, v) such that u and v have a common neighbor outside of $N^{\leqslant 2}(v_a)$. By (i) each vertex outside of $N^{\leqslant 2}(v_a)$ can be a common neighbor for at most $n^{4/3}$ pairs of vertices, so (ii) implies that $P \leqslant \frac{1}{36}n^2$. Note that a vertex from A is not contained in A' only if it is in at least $|A|$ ordered pairs that contribute to P, i.e.,

there are at least $\frac{|A|}{2}$ pairs with v on the first position and at least $\frac{|A|}{2}$ with v on the second position. It follows that A' contains at least $|A| - \frac{2P}{|A|}$ vertices. Since a is most frequent color used by the 3-coloring ϕ, we have $|A| \geqslant \frac{1}{3}n$, and thus $|A'| \geqslant \frac{1}{2}|A|$, as desired.

We will show that, with high probability, A' is a subset of A required to prove the claim. Fix a vertex w from A'. Consider a random variable X_w that counts the number of vertices u from A such that u threatens w and $N(u) \cap N(w) \subseteq N^{\leqslant 2}(v_a)$. Our plan is to use Corollary 5.9 to show that X_w is at least $8n^{2/3}\log n$ with high probability.

We start by estimating the expected value of X_w. Let U be the set of vertices $u \in A$ such that $N(u) \cap N(w) \subseteq N^{\leqslant 2}(v_a)$; note that by the definition of U, there is a vertex in $N[u] \cap N[w] \cap N^{\leqslant 2}(v_a)$, so x_{uw} and $f(x_{uw})$ exist for all vertices $u \in U$. Each vertex $u \in U$ contributes 1 to X_w if and only if $f(x_{uw}) \in \widetilde{S}$, i.e., with probability $\widetilde{p}$. Since $w \in A'$, the size of U is at least $\frac{1}{2}|A|$ minus the number of vertices outside of $N^{\leqslant 2}(w)$, which totals to at least $\frac{n}{6} - \frac{1}{36}n^{2/3}$ by (ii). Therefore, $\mathsf{E}[X_w] \geqslant 16n^{2/3}\log n$ for large enough n.

Now we express X_w as a sum of a number of independent random variables. Fix an ordering $t_1, t_2, \ldots, t_{\deg(v_a)}$ of neighbors of v_a and define U_i as the set of vertices u from U such that $x_{uw} \in N^{\leqslant 2}(v_a)$ and $f(x_{uw}) = t_i$. For $i = 1, 2, \ldots, \deg(v_a)$ let X_i be a random variable that is equal to $|U_i|$ if $t_i \in \widetilde{S}$ and 0 otherwise. Clearly $X_w = \sum_i X_i$ and all the variables $X_1, \ldots, X_{\deg(v_a)}$ are independent by the independent selection of $\widetilde{S}$.

By (iii), applied for w and t_i, we obtain that $X_i \leqslant n^{2/3}$ for all i. Therefore the sequence of random variables $\frac{X_i}{n^{2/3}}$ satisfies the assumptions of Corollary 5.9 (2); since it sums up to $X = \frac{X_w}{n^{2/3}}$, by setting $\widehat{X} = 16n^{2/3}\log n$ and $\varepsilon = \frac{1}{2}$ in the referenced corollary we deduce that

$$\mathsf{Pr}\left(\frac{X_w}{n^{2/3}} \leqslant 8\log n\right) \leqslant e^{-\frac{16}{8}\log n},$$

which gives that

$$\mathsf{Pr}\left(X_w \leqslant 8n^{2/3}\log n\right) \leqslant n^{-2}.$$

By the union bound we obtain that the probability that $\widetilde{S}$ has more than $200n^{1/3}\log n$ vertices or that $X_w < 8n^{2/3}\log n$ for any $w \in A'$ is at most $n^{-37.5n^{1/3}} + n \cdot n^{-2}$. It follows that $\widetilde{S}$ (together with the subset A' of A) satisfies the properties required in the claim with probability at least $1 - n^{-37.5n^{1/3}} - n \cdot n^{-2}$, which is positive for large enough n; therefore, the proof is complete. ⌟

Having selected $\widetilde{S}$, we proceed to selecting S as a subset of A that guarantees the desired domination property.

Claim 5.10.2. *There exists a set $S \subseteq A$ of order at most $2n^{1/3}$ such that at least half of the vertices $w \in A$ are dominated by $S \cup \widetilde{S} \cup \Big(N(S) \cap N(\widetilde{S})\Big)$.*

Proof of Claim: We randomly select S so that each vertex from A is in S independently with probability $p = n^{-2/3}$. Note that by Corollary 5.9 (1) the size of S is at most $2n^{1/3}$ with probability at least $1 - e^{-\frac{3}{8}n^{2/3}}$.

Let w be a vertex from A that is threatened by at least $8n^{2/3}\log n$ vertices from A. The probability that w is not dominated by $N(S) \cap N(\widetilde{S})$ is at most

$$(1-p)^{8n^{2/3}\log n} \leqslant e^{-8pn^{2/3}\log n} \leqslant e^{-8\log n} \leqslant n^{-8}.$$

By the union bound it follows that with probability at least $1 - n^{-7}$ all vertices threatened by at least $8n^{2/3}\log n$ vertices from A are dominated by $N(S) \cap N(\widetilde{S})$. Claim 5.10.1 implies that there are at least $\frac{1}{2}|A|$ such vertices. This completes the proof of claim. ⌟

Set $K := \max(n_0, 200)$. Now the statement of the lemma follows from Claim 5.10.2 by observing that since A is the most frequent color, we have $\frac{1}{2}|A| \geqslant \frac{1}{6}n$. □

Our algorithm for diameter-2 graphs consists of two phases. The following lemma encapsulates the outcome of the first phase.

Lemma 5.11. *Let (G, L) be an instance of the* List 3-Coloring *problem, where G has n vertices and is of diameter at most 2. In time $2^{\mathcal{O}(n^{1/3}\log^2 n)}$ we can enumerate a family $\mathcal{X}$ of subinstances of (G, L), such that:*

1. *$|\mathcal{X}| = 2^{\mathcal{O}(n^{1/3}\log^2 n)}$,*
2. *for each $(G', L') \in \mathcal{X}$ and every $v \in V(G')$ it holds that $|L'(v)| \leqslant 2$,*
3. *for any 3-coloring φ of G, respecting the lists L, there is $(G', L') \in \mathcal{X}$, such that φ is compatible with (G', L').*

Proof. Let (G, L) be an instance of the List 3-Coloring problem, where G has n vertices and is of diameter at most 2. Again, we start by applying reduction rules exhaustively, so we can assume that (G, L) is reduced. Let (V_1, V_2, V_3) be the layer structure of (G, L). If $V_3 = \emptyset$, then we can return $\mathcal{X} = \{(G, L)\}$ and terminate; clearly $\mathcal{X}$ satisfies all required properties. So from now on let us assume that $V_3 \neq \emptyset$ and set $\mu := |V_3|$.

We use one of the four branching rules to produce a number of subinstances of the problem, each with fewer vertices with lists of size 3. The family $\mathcal{X}$ consists of all subinstances obtained in the leaves of the recursion tree.

The following branching rules are applied in the given order – it is essential that (B4) is executed only if the rules (B1), (B2), and (B3) cannot be applied.

(B1) If there exists a vertex $v \in V_2 \cup V_3$ such that v has more than $\mu^{2/3}$ neighbors in V_3, then for every color $a \in L(v)$ call the algorithm recursively for a subinstance obtained by replacing $L(v)$ with $\{a\}$ and exhaustively applying the reduction rules.

(B2) If there exists a vertex $v \in V_3$ such that for at least $\frac{1}{36}\mu^{2/3}$ vertices $u \in V_3$ a common neighbor of u and v is in V_2, then for every color $a \in L(v)$ call the algorithm recursively for a subinstance obtained by replacing $L(v)$ with $\{a\}$ and exhaustively applying the reduction rules.

(B3) If there are two vertices $u, v \in V_3$ such that for at least $\mu^{2/3}$ vertices w from V_3 the set $N(u) \cap N(v) \cap N(w)$ is nonempty, then for every two distinct colors a, b construct a subinstance by setting $L(u) := \{a\}$ and $L(v) := \{b\}$ and, if $uv \notin E(G)$, one additional instance obtained by identifying u with v. Apply the reduction rules to each of those subinstances and call the algorithm recursively.

(B4) Let K be the constant from Lemma 5.10. For every tuple $(a, S, \widetilde{S}, \varphi)$, where

- $a \in [3]$ is a color,
- $S \subseteq V_3$ is a set of size at most $K \cdot \mu^{1/3} \log \mu$,
- $\widetilde{S} \subseteq V_3 \setminus S$ is a set of size at most $K \cdot \mu^{1/3} \log \mu$,
- φ is a list coloring of $(\widetilde{S}, L)$ using colors $[3] \setminus \{a\}$,

construct a subinstance by setting $L(v) := \{a\}$ for each $v \in S$ and $L(v) = \{\varphi(v)\}$ for $v \in \widetilde{S}$. Apply the reduction rules to each of those subinstances for which $S \cup \widetilde{S} \cup \left(N(S) \cap N(\widetilde{S})\right)$ dominates at least $\frac{1}{6}\mu$ vertices from V_3 and call the algorithm recursively.

Let us show that the above algorithm is correct. Branching rules (B1) and (B2) are clearly correct, because if there is a solution to the given instance of the List 3-Coloring problem, then it assigns to v one color from $L(v)$. The rule (B3) is correct because if

there is a solution f to the given instance of the problem, then it either assigns two different colors to u and v, or assigns the same color to u and v, hence at least one of the constructed subinstances will admit a compatibile solution. Recall that identifying the vertices u and v does not increase the diameter. Now consider the branching rule (B4). Recall that it is applied only when rules (B1), (B2) and (B3) are inapplicable, so in this case the graph $G[V_3]$ satisfies the assumptions (i)-(iii) of Lemma 5.10. Indeed, it is immediate for (i) and (iii), and to see (ii) suppose that $G[V_3]$ does not satisfy (ii), i.e., there is a vertex $v \in V_3$ such that $N^{\leqslant 2}_{G[V_3]}(v)$ contains fewer than $\mu - \frac{1}{36}\mu^{2/3}$ vertices. Then there are at least $\frac{1}{36}\mu^{2/3}$ vertices u such that u has a common neighbor with v in V_2 – this follows from the fact that G has diameter at most 2, and if two vertices of V_3 are not at distance at most two in $G[V_3]$, they must have a common neighbor in V_2. In this case we would apply branching rule (B2), a contradiction. Therefore, if the original instance has a solution f, then by Lemma 5.10 at least one subinstance constructed in (B4) admits a compatibile solution. On the other hand, each subinstance is obtained by fixing the colors of vertices in $S \cup \widetilde{S}$ and both sets are contained in V_3, so every vertex of $S \cup \widetilde{S}$ has all 3 colors on its lists, and thus each such a coloring respects lists L. Furthermore, if this coloring is improper, then the application of reductions rules (R1) and (R2) will cause the algorithm to reject the instance. Hence, the branching rule (B4) is correct.

Let us denote by $F(x)$ the maximum running time of the algorithm on an instance with at most x vertices with lists of size 3.

Now we will bound the running time of the algorithm on our instance (G, L) with μ vertices with lists of size 3, depending on which branching rule was applied.

Case 1: (B1) was applied Note that this branching produced at most three instances of the problem, each with at most $\mu - \mu^{2/3}$ vertices with lists of size 3. This is because for every vertex $u \in V_3$ that is a neighbor of v the color a was removed from $L(u)$. Therefore, in this case the running time is at most

$$3F\left(\mu - \mu^{2/3}\right) + 3\mathsf{p}(n),$$

where $\mathsf{p}(n) = \mathcal{O}(n^3)$ is a polynomial such that we can exhaustively apply the reduction rules to an n-vertex instance of List 3-Coloring in time $\mathsf{p}(n)$.

Case 2: (B2) was applied Let $\{a, b, c\} = [3]$ and let N_a (respectively N_b or N_c) be the number of vertices $u \in V_3$ such that the list of a common neighbor of v and u in V_2 does not contain a (respectively b or c). We can assume that $N_a \leqslant N_b \leqslant N_c$. Note that if a vertex u contributes to N_c, then after the application of reduction rules b (respectively a) is removed from $L(u)$ in the instance constructed for the color a (respectively b). It follows that the running time of the algorithm in this case is at most

$$F(\mu - 1) + 2F\left(\mu - \frac{1}{108}\mu^{2/3}\right) + 3\mathsf{p}(n).$$

Case 3: (B3) was applied Let w be a vertex from V_3 such that the set $N(u) \cap N(v) \cap N(w)$ is nonempty. Note that if we set $L(u)$ to $\{a\}$ and $L(v)$ to $\{b\}$, for $a \neq b$, then after applying the reduction rules common neighbors of u and v will have lists of size 1, hence the size of the list of w will be at most 2. Therefore, in this case the running time is at most

$$F(\mu - 1) + 6F\left(\mu - \mu^{2/3}\right) + 7\mathsf{p}(n),$$

where the first term corresponds to the subinstance with two vertices identified into one, and the second term corresponds to the six subinstances obtained by fixing the colors of two vertices.

Case 4: (B4) was applied Note that in the constructed instances, after applying the reduction rules, all vertices from $S \cup \widetilde{S} \cup \left(N(S) \cap N(\widetilde{S})\right)$ have lists of size 1, so all vertices dominated by $S \cup \widetilde{S} \cup \left(N(S) \cap N(\widetilde{S})\right)$ have lists of size at most 2. Therefore, all instances that are solved recursively have at most $\mu - \frac{1}{6}\mu$ vertices with lists of size 3. The total number of those instances can be upper bounded by the number of all possible tuples $(a, S, \widetilde{S}, \varphi)$. Recall that a is one of 3 possible colors, S and $\widetilde{S}$ are subsets of V_3, each of size at most $K \cdot \mu^{1/3} \cdot \log \mu$, and thus there are at most $\mu^{K\mu^{1/3}\log\mu}$ choices for each of them, and φ is a 2-coloring of $\widetilde{S}$, so for a fixed $\widetilde{S}$, there are at most $2^{K\mu^{1/3}\log\mu}$ choices for φ. Therefore, the total number of instances created in this case is at most

$$3 \cdot \mu^{2K\mu^{1/3}\log\mu} \cdot 2^{K\mu^{1/3}\log\mu} < 2^{K'\mu^{1/3}\log^2\mu},$$

for some constant K'. Furthermore, the total running time in this case is at most

$$2^{K'\mu^{1/3}\log^2\mu} F\left(\frac{5}{6}\mu\right) + 2^{K'\mu^{1/3}\log^2\mu}\mathsf{p}(n)$$

As the considered cases cover all possibilities, we conclude that $F(\mu)$ is bounded by the maximum of the expressions obtained in all four cases. Solving these recursions and using that $\mu \leqslant n$ (again, see the Appendix for details) we obtain that the overall complexity of the algorithm is bounded by $2^{\mathcal{O}(n^{1/3}\log^2 n)}$. From the bound on the running time we also obtain $|\mathcal{X}| = 2^{\mathcal{O}(n^{1/3}\log^2 n)}$. □

Now Theorem 1.12 follows easily from Lemma 5.11 and Theorem 5.4.

Theorem 1.12. *The* List 3-Coloring *problem on n-vertex graphs with diameter 2 can be solved in time $2^{\mathcal{O}(n^{1/3}\cdot\log^2 n)}$.*

Proof. Let (G, L) be an instance of the List 3-Coloring problem, where G has n vertices and is of diameter at most 2. Apply Lemma 5.11 to (G, L) to obtain a family $\mathcal{X}$ of subinstances. By statement 2 of Lemma 5.11, each $(G', L') \in \mathcal{X}$ can be solved in polynomial time using Theorem 5.4. Since elements of $\mathcal{X}$ are subinstances of (G, L) and by statement 3 of Lemma 5.11 we observe that (G, L) is a yes-instance of List 3-Coloring if and only if at least one element of $\mathcal{X}$ is a yes-instance. The time bound follows from the time bound in Lemma 5.11 (needed to construct the family $\mathcal{X}$) and the size bound on $\mathcal{X}$ from statement 1. of Lemma 5.11. □

5.1.3 Weighted coloring

In this section we consider a generalization of List 3-Coloring, called Weighted 3-Coloring. Actually, we consider Weighted List 3-Coloring, where the instance has both, weights and lists.

Weighted List q-Coloring
Input: Graph G with weight function $\mathfrak{w} : V(G) \times [q] \to \mathbb{N}$, list function $L : V(G) \to 2^{[q]}$, and an integer k.
Question: Is there a proper q-coloring of G, which respects the lists L, and such that $\sum_{v \in V(G)} \mathfrak{w}(v, c(v)) \leqslant k$?

Observe that Weighted List q-Coloring is equivalent to Weighted q-Coloring, as lists can be easily expressed with weights: to indicate that $i \notin L(v)$ we can set $\mathfrak{w}(v, i)$ to some large value (larger than k). However, introducing lists gives us an easy way to indicate that some neighbors of a vertex are already colored, without the need of updating the weight function.

Let us discuss which of the phases of the algorithms from Theorem 1.11 and Theorem 1.12 can be simply applied in the weighted version.

Subinstances. We can generalize the notion of subinstances to the case of Weighted List 3-Coloring, where the only difference is the following: for an instance $(G, L, \mathfrak{w}, k)$ of Weighted List 3-Coloring, when identifying some vertices u, v of G into z, we set $\mathfrak{w}(z, i) = \mathfrak{w}(u, i) + \mathfrak{w}(v, i)$ for $i \in [3]$. Thus, in case of Weighted List 3-Coloring, we can also say that $I' = (G', L', \mathfrak{w}', k')$ is a subinstance of $I = (G, L, \mathfrak{w}, k)$, if I' can be obtained from I by a series of operations: (i) removing some colors from the list of a vertex, and (ii) identifying two non-adjacent vertices with the same list. Observe that we actually have $k' = k$ as none of the two operations changes the budget k. Similarly, we can generalize the notion of compatible colorings. Let $\mathsf{trace} : V(G) \to V(G')$ be defined as in the non-weighted setting, i.e., $\mathsf{trace} : V(G) \to V(G')$ is a surjective mapping, where

a) if $x \in V(G) \cap V(G')$, i.e., x was never identified with some other vertex when (G', L') was obtained from (G, L), then $\mathsf{trace}(x) = x$,

b) if $x \in V(G) \setminus V(G')$, i.e., there is some $z \in V(G')$, such that x was identified into z (note that it is possible that x was identified with x' into y and then y was identified with y' into z), then $\mathsf{trace}(x) = z$.

Again, we can say that a coloring c certifying that $(G, L, \mathfrak{w}, k)$ is a yes-instance of Weighted List 3-Coloring is *compatible* with $(G', L', \mathfrak{w}', k)$ if

(1) for each $x, y \in V(G)$, if $\mathsf{trace}(x) = \mathsf{trace}(y)$, then $c(x) = c(y)$,

(2) for each $x \in V(G)$ it holds that $c(x) \in L'(\mathsf{trace}(x))$.

Let us point out that the coloring c' obtained by restricting c to vertices of G' (we can use the name "restriction" because of (1)) has the same total weight as c, which is at most k.

Reduction rules. Observe that all of the reduction rules (R1), (R2), and (R3) are safe when solving Weighted List 3-Coloring, as they only identify the vertices that have to obtain the same color, or remove a color that cannot be used, or return NO, if there is a vertex with empty list.

Branching rules. Now we claim that the branching phases in our algorithms from Theorem 1.11 and Theorem 1.12 can handle the weights: in each branching rule, except (B4), we simply guess a coloring for some set of vertices, and thus branching rules (B1), (B2), and (B3) can be safely applied when solving Weighted List 3-Coloring. In case of (B4) we guess some sets of vertices and their coloring, but we also discard some instances – the ones where the guessed sets $S, \widetilde{S}$ are such that $S \cup \widetilde{S} \cup (N(S) \cap N(\widetilde{S}))$ does not dominate sufficiently many vertices. However, by Lemma 5.10, if $I = (G, L, \mathfrak{w}, k)$ is a yes-instance of Weighted List 3-Coloring, and c is a coloring certifying that I is a yes-instance, then whenever (B4) produces an instance $I' = (G', L', \mathfrak{w}', k)$ compatible with c, then I' will not be discarded. Therefore, (B4) can be also safely applied in weighted setting.

Therefore, we can write the following strenghtening of Lemma 5.11

Lemma 5.12. *Let $I = (G, L, \mathfrak{w}, k)$ be an instance of the Weighted List 3-Coloring problem, where G has n vertices and is of diameter at most 2. In time $2^{\mathcal{O}(n^{1/3}\log^2 n)}$ we can enumerate a family $\mathcal{X}$ of subinstances of $(G, L, \mathfrak{w}, k)$, such that:*

1. *$|\mathcal{X}| = 2^{\mathcal{O}(n^{1/3}\log^2 n)}$,*
2. *for each $(G', L', \mathfrak{w}', k') \in \mathcal{X}$ and every $v \in V(G')$ it holds that $|L'(v)| \leqslant 2$,*
3. *for any 3-coloring c of G, certifying that I is a yes-instance of Weighted List 3-Coloring, there is $(G', L', \mathfrak{w}', k) \in \mathcal{X}$, such that c is compatible with $(G', L', \mathfrak{w}', k)$.*

However, when solving Weighted List 3-Coloring we cannot simply apply the last phase, when the problem of coloring a graph with all lists of size at most two is reduced to 2-Sat using Theorem 5.4. It is known that a weighted variant of 2-Sat is NP-complete and admits no subexponential-time algorithm, unless the ETH fails [129].

It turns out that we can substitute the second phase of Theorem 1.12 with another procedure that can handle weights.

Lemma 5.13. *The Weighted List 3-Coloring problem on n-vertex diameter-2 graphs can be solved in time $n^{\mathcal{O}(\log n)}$, if the list of every vertex has at most 2 elements.*

Proof. Let $(G, \mathfrak{w}, L, k)$ be an instance of Weighted List 3-Coloring such that G has n vertices and is of diameter at most 2, and for each $v \in V(G)$ it holds that $|L(v)| \leqslant 2$.

Assume that none of the reduction rules (R1), (R2), (R3) can be applied. Furthermore, we introduce two more reduction rules and apply them in this order; clearly it can be done in polynomial time.

(R4) If there is an odd cycle whose all vertices have the same list, then terminate and report a no-instance.

(R5) If vertices u, v, w have the same 2-element list and $uv, vw \in E(G)$, then identify u and w.

The reduction rule (R4) is justified, as there is no way to color an odd cycle with (at most) two colors. Now let us discuss reduction rule (R5). Note that since (R4) cannot be applied, we have $uw \notin E(H)$. Since $L(u) = L(v) = L(w)$, we note that in any coloring respecting lists, u and w have the same color.

Suppose that no reduction rule can be applied and let $(V_1, V_2, \emptyset)$ be the layer structure. Define $\mu := |V_2|$.

Clearly if $\mu = 0$, the problem can be trivially solved in polynomial time. So from now on assume that $\mu > 0$. Since each vertex from V_2 has one of three possible lists – $\{1, 2\}$, $\{1, 3\}$, or $\{2, 3\}$ – there is a set $V_2' \subseteq V_2$ of size at least $\mu/3$, such that every vertex from V_2' has the same list. By symmetry let us assume that this list is $\{1, 2\}$.

First, consider the case that $V_2' = V_2$. Let $\mathcal{C}$ be the family of connected components of $G[V_2]$. Since (R4) cannot be applied, each $C \in \mathcal{C}$ is bipartite and thus has exactly two possible colorings. Furthermore, the coloring of $C \in \mathcal{C}$ does not influence the coloring of other elements of $\mathcal{C}$. Thus for each $C \in \mathcal{C}$ we can independently choose the coloring with smaller weight and check if the coloring of G obtained this way has total weight at most k. Summing up, in this case we can solve the problem in polynomial time.

Now consider the case that there is some vertex $x \in V_2 \setminus V_2'$; by symmetry assume that $L(x) = \{1, 3\}$. We introduce two branching rules, and we apply them in the given order.

(B1) Suppose there is a vertex $v \in V_2$ adjacent to at least $\mu/18$ vertices in V_2. For each $a \in L(v)$ we recursively solve the instance obtained by setting $L(v) = \{a\}$ and exhaustively applying the reduction rules.

(B2) For each $a \in L(x)$ we recursively solve the instance obtained by setting $L(x) = \{a\}$ and exhaustively applying the reduction rules.

Again, in both branching rules we simply guess coloring of some vertex, and thus the corectness of the algorithm follows.

Now let us analyze the complexity of our algorithm. Let $F(\ell)$ denote the maximum possible running time on an instance with at most ℓ vertices having a list of size 2, and let $\mathsf{p}(\ell)$ denote a polynomial such that the reduction rules (R1), (R2), (R3), (R4), and (R5) can be exhaustively applied for an ℓ-vertex instance. Consider the cases.

Case 1: (B1) was applied Note that at least $\mu/(18 \cdot 3)$ neighbors of v in V_2 have the same list L', and there is $a \in L' \cap L(v)$. Thus, in this case $F(\mu)$ is at most

$$F(\mu) \leqslant F(\mu - \mu/54) + F(\mu - 1) + 2\mathsf{p}(\mu).$$

The first term corresponds to a branch where we set $L(v) = \{a\}$ such that $a \in L'$ so the reduction rule (R3) will decrease sizes of lists of at least $\mu/54$ vertices of V_2. The second term corresponds to the other branch – we know that we at least decrease the size of $L(v)$.

Case 2: (B2) was applied Note that each vertex from V_2' is at distance at most 2 from x. Let A be the set of vertices in V_2' which are adjacent to x. For every $v \in V_2' \setminus A$ fix a common neighbor y_v of v and x. We claim that $|L(y_v)| = 2$. Suppose otherwise and let $L(y_v) = \{a\}$, for some $a \in [3]$. Observe that since $v \in V_2'$, $x \notin V_2'$, lists $L(v)$, $L(x)$ are distinct lists of size 2, and thus $L(v) \cup L(x) = [3]$, so a is in at least one of $L(v)$, $L(x)$. But then, the reduction rule (R3) could be applied to remove a from one of the lists, a contradiction. Therefore, we have $|L(y_v)| = 2$. Let us partition $V_2' \setminus A$ into sets B, C, D defined as follows

$$B := \{v \in V_2' \setminus A \mid L(y_v) = \{1, 2\}\},$$
$$C := \{v \in V_2' \setminus A \mid L(y_v) = \{1, 3\}\},$$
$$D := \{v \in V_2' \setminus A \mid L(y_v) = \{2, 3\}\}.$$

Note that $V_2' = A \cup B \cup C \cup D$ and it is indeed a partition, since the vertex y_v has been fixed for every v.

As (B1) could not be applied, we observe that $|A| \leqslant \mu/18$, otherwise x would be a vertex with more than $\mu/18$ neighbors in V_2. As (R5) and (B1) could not be applied, we observe that $|B| \leqslant \mu/18$ and $|C| \leqslant \mu/18$. Indeed, observe first that for any distinct

vertices $v, u \in B$, it cannot hold that $y_v = y_u$ since then we would apply (R5). Therefore, $|B| \geqslant \mu/18$ implies that x has at least $\mu/18$ neighbors in V_2 and we would apply (B1). For the upper bound on $|C|$, if for two distinct vertices $u, v \in C$ we have $y_u \neq y_v$, then we would apply (R5) to y_u, y_v, x. Therefore, we have that for any distinct $u, v \in C$ we have $y_u = y_v$ and $|C| \geqslant \mu/18$ implies that y_v is a vertex with at least $\mu/18$ neighbors in V_2 – a contradiction with a fact that (B1) cannot be applied. Thus, since $|V_2'| \geqslant \mu/3$, we obtain that $|D| \geqslant \mu/6$.

Note that in the branch where we set $L(x) = \{3\}$, for every $v \in D$, the reduction rules force $L(y_v)$ to be $\{2\}$ and thus the list of v is set to $\{1\}$. Thus the running time in this case is at most

$$F(\mu) \leqslant F(\mu - 1) + F(\mu - \mu/6) + 2\mathsf{p}(\mu).$$

Consequently, we obtain that $F(\mu) = \mu^{\mathcal{O}(\log \mu)}$ (again, see the Appendix for details). Since $\mu \leqslant n$, the proof is complete. □

Now, combining Lemma 5.12 with Lemma 5.13, we obtain the following strengthening of Theorem 1.12.

Theorem 1.13. *The* Weighted 3-Coloring *problem on n-vertex diameter-2 graphs can be solved in time $2^{\mathcal{O}(n^{1/3} \log^2 n)}$.*

In stark contrast, we show that it is not possible to strengthen Theorem 1.11 in a similar way.

Theorem 1.14. *The* Weighted 3-Coloring *problem on n-vertex diameter-3 graphs cannot be solved in time $2^{o(n)}$, unless the ETH fails.*

Proof. First, let us show the following claim.

Claim 1.14.1. *Let $(G, \mathfrak{w}, L, k)$ be an instance of the* Weighted List 3-Coloring *problem, where for each $v \in V(G)$ we have $|L(v)| = 2$. Then in polynomial time we can construct an instance $(G', \mathfrak{w}', k)$ of the* Weighted 3-Coloring *problem, such that:*

a) $(G, \mathfrak{w}, L, k)$ is a yes-instance of Weighted List 3-Coloring *if and only if $(G', \mathfrak{w}', k)$ is a yes-instance of* Weighted 3-Coloring*,*

b) $|G'| = |G| + 3$,

c) G' is of diameter at most 3.

Proof of Claim: Define $(G', \mathfrak{w}', k)$ as follows:

$$\begin{aligned} V(G') =& V(G) \cup \{v_1, v_2, v_3\}, \\ E(G') =& E(G) \cup \{v_1v_2, v_1v_3, v_2v_3\} \cup \bigcup_{i \in [3]} \Big\{vv_i \mid v \in V(G) \text{ and } L(v) = [3] \setminus \{i\}\Big\}, \end{aligned}$$

and for every $v \in V(G')$ and $i \in [3]$ we have

$$\mathfrak{w}'(v,i) = \begin{cases} \mathfrak{w}(v,i) & \text{if } v \in V(G), \\ 0 & \text{if } v = v_i, \\ k+1 & \text{if } v \in \{v_1, v_2, v_3\} \setminus \{v_i\}. \end{cases}$$

Clearly, $|G'| = |G| + 3$. Furthermore, each vertex of G' is adjacent to one vertex of the triangle $v_1v_2v_3$, and thus $\operatorname{diam}(G') \leqslant 3$. It remains to verify the equivalence. First assume that there is a list coloring $c : V(G) \to [3]$ of total weight at most k. We define $c' : V(G') \to [3]$ so that $c'(v) = c(v)$ for $v \in V(G)$, and $c'(v_i) = i$ for $i \in [3]$. Since $\mathfrak{w}(v_i, i) = 0$, the total weight of c' is at most k. It remains to verify that c' is a proper coloring. Suppose there is $uv \in E(G')$ such that $c'(u) = c'(v)$. Since c is a proper coloring, u, v cannot be both in $V(G)$, so let $u = v_i$ for some $i \in [3]$. Since v_1, v_2, v_3 are colored with three distinct colors, it must hold that $v \in V(G)$. But then $L(v) = [3] \setminus \{i\}$ and since c respects the lists, $c'(v) = c(v) \neq i = c'(v_i)$, a contradiction.

So now assume that there is a coloring $c' : V(G') \to [3]$ of total weight at most k. Define $c = c'|_{V(G)}$. Note that for every $i \in [3]$, it holds that $c'(v_i) = i$. Therefore, for every $v \in V(G) \cap N_{G'}(v_i)$, we have that $c(v) \in [3] \setminus \{i\} = L(v)$, and thus c respects the lists L. Moreover, the total weight of c' is at most k, which completes the proof of the claim. ⌟

By Claim 1.14.1 it is sufficient to prove the following claim.

Claim 1.14.2. *Assuming the ETH, there is no algorithm that solves every n-vertex instance of* Weighted List 3-Coloring *with all lists of size 2 in time $2^{o(n)}$.*

Proof of Claim: We reduce from the Vertex Cover problem. Let (G', k) be an arbitrary instance with N vertices and M edges. It is known that the existence of an algorithm solving (G', k) in time $2^{o(N+M)}$ would contradict the ETH [37, Theorem 14.6]. Let $V(G') = \{u_1, u_2, \ldots, u_N\}$.

We construct an instance $(G, \mathfrak{w}, L, k)$ of Weighted List 3-Coloring as follows. For each $i \in [N]$ we introduce a vertex x_i; this vertex corresponds to the vertex u_i of G'.

We set $L(x_i) = \{1, 2\}$, $\mathfrak{w}(x_i, 1) = \mathfrak{w}(x_i, 3) = 0$ and $\mathfrak{w}(x_i, 2) = 1$. Note that the value of $\mathfrak{w}(x_i, 3)$ is irrelevant, as $3 \notin L(x_i)$. We will interpret coloring x_i with the color 2 as choosing u_i to a vertex cover in G'.

Now for every $1 \leqslant i < j \leqslant N$ such that $u_iu_j \in E(G')$ we join x_i and x_j with a four-edge path with internal vertices a_{ij}, b_{ij}, c_{ij} and set $L(a_{ij}) = \{1, 2\}, L(b_{ij}) = \{2, 3\}, L(c_{ij}) = \{1, 3\}$; all weights associated with a_{ij}, b_{ij}, c_{ij} are 0. It is straightforward to verify that this gadget ensures that at least one of x_i, x_j will be colored 2 in every proper coloring of G respecting lists L.

It follows that G admits a proper coloring, respecting lists L, with total cost at most k if and only if G' has a vertex cover of size at most k. As the number of vertices of G is linear in $N + M$, the claim follows. ⌟

By Claims 1.14.1 and 1.14.2, an algorithm solving Weighted 3-Coloring problem on n-vertex graphs with diameter at most 3 in time $2^{o(n)}$ would falsify the ETH, which completes the proof. □

5.2 Other target graphs

In this section we turn our attention to other target graphs than triangles. We study the complexity of Hom(H) and LHom(H) on diameter-d graphs for special pairs (H, d): (i) triangle-free graphs H and diameter 2, and (ii) odd cycles C_{2k+1} other than the triangle and the diameter depending on k. All our algorithms work for the more general LHom(H) problem, while the lower bounds hold even in the non-list problem.

Triangle-free target graphs

First we focus on traingle-free target graphs and we restrict ourselves to input graphs with diameter at most 2. Since homomorphisms preserve edges, for graphs G, H, a homomorphism $\varphi : G \to H$ and a sequence of vertices $v_1, \ldots, v_k$ forming a path in G, the sequence $\varphi(v_1), \ldots, \varphi(v_k)$ forms a walk in H. The following observation is straightforward.

Observation 5.14. *Let G, H be graphs such that G is connected. If there exists a homomorphism $\varphi : G \to H$, then the image $\varphi(V(G))$ induces in H a subgraph with diameter at most* diam(G).

Therefore, for G with diameter at most 2, in order to determine whether $(G, L) \to H$, it suffices to verify whether $(G, L) \to H'$ for at least one diameter-2 subgraph H' of H. Hence, we can assume that the target graph H has also diameter at most 2.

Let us introduce one more reduction rule.

(R6) For every $v \in V(G)$, if there are $x, y \in L(v)$ such that for every $u \in N_G(v)$, we have $N_H(x) \cap L(u) \subseteq N_H(y) \cap L(u)$, then remove x from $L(v)$.

Clearly (R6) can be applied in polynomial time. Let us verify that it is safe. Assume there is $v \in V(G)$ and $x, y \in L(v)$ such that for every $u \in N_G(v)$, it holds $N_H(x) \cap L(u) \subseteq N_H(y) \cap L(u)$, and suppose there is a list homomorphism $\varphi : (G, L) \to H$ such that $\varphi(v) = x$. Then φ' defined so that $\varphi'(v) = y$ and $\varphi'(w) = \varphi(w)$ for $w \in V(G) \setminus \{v\}$ is also a list homomorphism $(G, L) \to H$. Therefore, (R6) is safe.

We prove Theorem 1.15.

Theorem 1.15. *Let H be a triangle-free graph. Then the LHOM(H) problem can be solved in polynomial time on diameter-2 graphs.*

Proof. Let (G, L) be an instance of LHOM(H). We guess the set of colors that will be used – by Observation 5.14 they should induce a diameter-2 subgraph H' of H. For each such H', we guess $h' = |H'|$ vertices $v_1, \ldots, v_{h'}$ of G that will be injectively mapped to $V(H') = \{x_1, \ldots, x_{h'}\}$. For each tuple $(H', v_1, \ldots, v_{h'})$ such that $x_i \in L(v_i)$ for $i \in [h']$, we solve the instance (G, L') of LHOM(H'), where $L'(v) = \{x_i\}$ for $v = v_i$, $i \in [h']$ and $L'(v) = L(v)$ otherwise. Note that (G, L) is a yes-instance if and only if at least one instance (G, L') is a yes-instance.

First, for every edge $x_i x_j \in E(H')$, if v_i, v_j are non-adjacent, we add the edge $v_i v_j$ to G – note that this operation is safe, since we cannot increase the diameter by adding edges and we only add edges between vertices that must be mapped to neighbors in H'. Therefore, we can assume that the set $V' = \{v_1, \ldots, v_{h'}\}$ induces a copy of H' in G (if not, then we have an extra edge, which means that we are dealing with a no-instance and we reject immediately). Furthermore, we exhaustively apply reduction rules (R1), (R2), (R3), and (R6).

So from now on we assume that for the instance (G, L'), none of the reduction rules (R1), (R2), (R3), (R6) can be applied. We claim that either (G, L') is a no-instance or

$V(G) = \{v_1, \ldots, v_{h'}\}$, i.e., after exhaustive application of the reduction rules, the graph G is isomorphic to H'. Note that in the latter case we can return YES as an answer.

Suppose there is $v \in V(G) \setminus V'$. Moreover, we choose such v which is adjacent to some vertex of V' (see Figure 5.2). Suppose that there exists $\varphi : (G, L') \to H'$ and let $x_i = \varphi(v)$. Then v cannot be adjacent to v_i since there are no loops in H'. Furthermore, the only neighbors of v in V' can be the neighbors of v_i. Suppose that there is $v_j \in N_G(v_i) \cap V'$ which is non-adjacent to v. Since the diameter of G is at most 2, then there must be $u \in N_G(v) \cap N_G(v_j)$. Observe that $u \notin V'$. Indeed, v does not have any neighbors in $V' \setminus N_G(v_i)$ and if $u \in N_G(v)$, then there is a triangle uv_iv_j in a copy of H', a contradiction. Furthermore, it must hold that $\varphi(u)$ is adjacent to x_i in H' as u is adjacent to v and $\varphi(v) = x_i$, and similarly, $\varphi(u)$ must be adjacent to x_j as u is adjacent to v_j. Then $\varphi(u)x_ix_j$ forms a triangle in H', a contradiction. Thus v must be adjacent to all vertices of $N(v_i) \cap V'$.

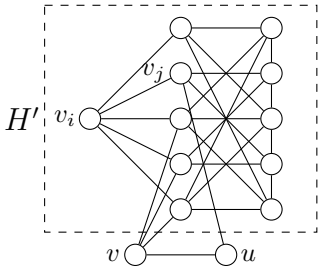

Figure 5.2: The copy of H' in G and a vertex v such that for some homomorphism φ, it holds $\varphi(v_i) = \varphi(v)$. We show that a vertex u which is a common neighbor of v and some neighbor v_j of v_i in the copy of H' cannot exist.

Since (R3) cannot be applied, each vertex of $L'(v)$ is adjacent to all vertices of $N_H(x_i)$. Moreover, since (R6) cannot be applied, it holds that $L'(v) = \{x_i\}$. Indeed, otherwise there is $x_{i'} \neq x_i$ such that $x_{i'} \in L'(v)$. Recall that $x_{i'}$ is adjacent to all vertices of $N_H(x_i)$. Therefore, $N_H(x_i) \subseteq N_H(x_{i'})$, and thus one of $x_i, x_{i'}$ should have been removed from $L'(v)$ by (R6). Furthermore, since (R2) cannot be applied, we must have $v = v_i \in V'$, a contradiction. This completes the proof. □

5.2.1 Odd cycles

Before we prove the main results of this section, we make a series of observations about the case where the target graph is an odd cycle other than the triangle. First, we describe the lists of vertices that are at some small distance of a precolored vertex.

Lemma 5.15. *Let $k \geqslant 1$ and let (G, L) be a reduced instance of LHOM(C_{2k+1}). Let $u \in V(G)$ be such that $L(u) = \{i\}$ and let $v \in V(G)$ be such that $\mathrm{dist}(u, v) = d$. Then*

a) $L(v) \subseteq \{i-d, i-d+2, \ldots, i-2, i, i+2, \ldots, i+d-2, i+d\}$ if d is even,

b) $L(v) \subseteq \{i-d, i-d+2, \ldots, i-1, i+1, \ldots, i+d-2, i+d\}$ if d is odd.

In particular, if $d \leqslant k-1$, then $L(v)$ is an independent set.

Proof. Let P be a shortest u-v path such that the consecutive vertices of P are $u = p_0, p_1, \ldots, p_d = v$. We have $L(p_0) = \{i\}$. Since the reduction rule (R3) cannot be applied for p_0p_1, we must have $L(p_1) \subseteq \{i-1, i+1\}$. Suppose now that there is $j \in [d]$ such that:

$$L(p_j) \subseteq \{i-j, i-j+2, \ldots, i-2, i, i+2, \ldots, i+j-2, i+j\},$$

if j is even, and

$$L(p_j) \subseteq \{i-j, i-j+2, \ldots, i-1, i+1, \ldots, i+j-2, i+j\},$$

if j is odd.

Then for p_{j+1}, which is adjacent to p_j and thus each vertex of $L(p_{j+1})$ must be adjacent to some vertex of $L(p_j)$, we have

$$L(p_{j+1}) \subseteq \{i-j-1, i-j+1, \ldots, i-1, i+1, \ldots, i+j-1, i+j+1\},$$

if j is even, and

$$L(p_{j+1}) \subseteq \{i-j-1, i-j+1, \ldots, i-2, i, i+2, \ldots, i+j-1, i+j+1\},$$

if j is odd. By the principle of induction the theorem follows. □

The next lemma immediately follows from Lemma 5.15.

Lemma 5.16. *Let $k \geqslant 1$ and let (G, L) be a reduced instance of LHOM(C_{2k+1}). Let $u, w \in V(G)$ be such that $L(u) = \{i\}$ and $L(w) = \{i+1\}$. Let $v \in V(G)$ be such that $\mathrm{dist}(u, v) = \mathrm{dist}(w, v) = k + \ell$ for some $\ell \geqslant 0$. Then $L(v) \subseteq \{i+k-\ell+1, i+k-\ell+2, \ldots, i+k+\ell+1\}$.*

In the following lemma we show that for a partial mapping $\varphi : V(G) \to [2k]_0$ for $k \geqslant 2$, for some vertex $v \in V(G)$, if every pair (a, b) of its neighbors is precolored so that $\varphi(a)$ and $\varphi(b)$ have a common neighbor in $L(v)$, then φ can be extended to v so that it preserves the edges containing v.

Lemma 5.17. *Let $k \geqslant 2$, let (G, L) be an instance of LHOM(C_{2k+1}) and let $v \in V(G)$. Let $\varphi : N(v) \to [2k]_0$ be a mapping such that for every $u, w \in N(v)$, we have that $N_{C_{2k+1}}(\varphi(u)) \cap N_{C_{2k+1}}(\varphi(w)) \cap L(v) \neq \emptyset$. Then $\bigcap_{u \in N(v)} N_{C_{2k+1}}(\varphi(u)) \cap L(v) \neq \emptyset$.*

Proof. Define $A = \{\varphi(u) \mid u \in N(v)\}$. If $|A| \leqslant 2$, then the statement clearly follows. We will show that this is the only case. So suppose that $|A| \geqslant 3$. If two distinct vertices of C_{2k+1} for $k \geqslant 2$ have a common neighbor, then they must be at distance exactly two. Without loss of generality, let $0, 2 \in A$ and $1 \in L(v)$. Moreover, let $i \in A \setminus \{0, 2\}$. By assumption $N_{C_{2k+1}}(i) \cap N_{C_{2k+1}}(0) \neq \emptyset$, so $i = 2k - 1$. On the other hand, $N_{C_{2k+1}}(i) \cap N_{C_{2k+1}}(2) \neq \emptyset$, so $i = 4$. Thus $2k - 1 = 4$, a contradiction. □

In the next lemma we show that for an odd cycle C and a vertex v there is at least one pair of consecutive vertices of C with equal distances to v.

Lemma 5.18. *Let G be a connected graph, let C be a cycle in G with consecutive vertices $c_0, \ldots, c_{2k}$ (indices computed modulo $2k + 1$), and let $v \in V(G) \setminus V(C)$. Then there is $i \in [2k]_0$ such that $\operatorname{dist}(v, c_i) = \operatorname{dist}(v, c_{i+1})$.*

Proof. First observe, that for all i, we have $|\operatorname{dist}(v, c_i) - \operatorname{dist}(v, c_{i+1})| \leqslant 1$ since $c_i c_{i+1} \in E(G)$. Therefore, going around the cycle, the distance from v to c_i can increase by 1, decrease by 1, or remain the same. Since we have to end up with the same value at the end and the length of the cycle is odd, there is at least one pair of consecutive vertices c_i, c_{i+1} such that $\operatorname{dist}(v, c_i) = \operatorname{dist}(v, c_{i+1})$. □

We will also use the result of Feder, Hell, and Huang [57].

Lemma 5.19 ([57])**.** *Let $t \geqslant 1$. Then every instance (G, L) of LHOM(P_t) can be solved in polynomial time.*

Odd cycles – polynomial-time algorithm

In this section we prove Theorem 1.16.

Theorem 1.16. *Let $k \geqslant 2$. Then LHOM(C_{2k+1}) can be solved in polynomial time on diameter-$(k+1)$ graphs.*

Proof. Let (G, L) be an instance of LHOM(C_{2k+1}) such that G has diameter at most $k+1$. First, for every $i \in [2k]_0$, we check whether there is a list homomorphism $\varphi : (G, L) \to C_{2k+1}$ such that no vertex is mapped to i. So we look for a list homomorphism to a path which can be done in polynomial time by Lemma 5.19.

If there is no such a homomorphism, then we know that all colors have to be used and thus we guess $2k+1$ vertices that will be mapped to distinct vertices of C_{2k+1}. Let $c_0, \ldots, c_{2k}$ be the vertices such that c_i is precolored with i. We check whether such a partial assignment respects the lists and satisfies the edges with both endpoints precolored. Moreover, for $i \in [2k]_0$, if c_i, c_{i+1} are non-adjacent, we add the edge $c_i c_{i+1}$ – this operation is safe as c_i, c_{i+1} are precolored with consecutive vertices of C_{2k+1} and adding an edge does not increase the diameter. Furthermore, we exhaustively apply the reduction rules. Finally, we check if G contains an odd cycle of length at most $2k-1$, and if so, then we reject the instance – recall that by Observation 3.2, there is no homomorphism from G to C_{2k+1}.

So since now we can assume that the instance (G, L) is reduced and G does not contain an odd cycle of length at most $2k-1$. Let us analyze (G, L).

Observe that the vertices $c_0, \ldots, c_{2k}$ induce a $(2k+1)$-cycle C. Suppose there is a vertex v that is not on C. By Lemma 5.18, there is $i \in [2k]_0$ such that $\operatorname{dist}(v, c_i) = \operatorname{dist}(v, c_{i+1}) =: \ell$.

First, we show that we cannot have $\ell < k$. Suppose otherwise. Let P_1, P_2 be shortest v-c_i-, and v-c_{i+1}-paths, respectively. Let u be the their last common vertex (it cannot be c_i or c_{i+1} as the distances are the same and c_i, c_{i+1} are adjacent). Note that since P_1, P_2 are shortest, the u-c_i-path P_1' obtained from P_1 and the u-c_{i+1}-path P_2' obtained from P_2 have the same length. Therefore we can construct a cycle by taking P_1', P_2' and the edge $c_i c_{i+1}$. The length of this cycle is odd, and it is at most $2k-1$, which is a contradiction.

Now suppose that $\ell = k$. By Lemma 5.16, we have $L(v) \subseteq \{i+k+1\}$. If $L(v) = \emptyset$, then the reduction rule (R1) would return NO, a contradiction. If $L(v) = \{i+k+1\}$, then (R2) would identify v with c_{i+k+1} or would return NO, a contradiction. Therefore we cannot have $\ell \leqslant k$, and thus, since $\operatorname{diam}(G) \leqslant k+1$, we have $\ell = k+1$.

By Lemma 5.16, we have that $L(v) \subseteq \{i+k, i+k+1, i+k+2\}$. Since v is an

arbitrary vertex outside C, we can conclude that all lists of our instance have size at most 3. Moreover, each vertex of V_3 (recall that by V_r we denote the set of vertices of $V(G)$ with lists of size r) has list of type $(1,1)$. Furthermore, since (R3) cannot be applied, for a vertex with list $\{j, j+1, j+2\}$, the possible lists of its neighbors in $G[V_3]$ are then $\{j-1, j, j+1\}$, $\{j, j+1, j+2\}$, and $\{j+1, j+2, j+3\}$.

For a list $\{j-1, j, j+1\}$ of type $(1,1)$, we will call j the *middle vertex* of $\{j-1, j, j+1\}$. For a homomorphism φ, we will say that a vertex $v \in V_3$ is *φ-middle*, if φ maps v to the middle vertex of its list.

Now consider a connected component S of $G[V_3]$, let $v \in V(S)$, and let $L(v) = \{j-1, j, j+1\}$. The following claim is straightforward.

Claim 1.16.1. *Suppose there is a list homomorphism $\varphi : (S, L) \to C_{2k+1}$. Then*

(1.) if v is φ-middle, then any $u \in N_S(v)$ with list $\{j-1, j, j+1\}$ cannot be φ-middle, and every $w \in N_S(v)$ with list $\{j-2, j-1, j\}$ or $\{j, j+1, j+2\}$ has to be φ-middle,

(2.) if v is not φ-middle, then every $u \in N_S(v)$ with list $\{j-1, j, j+1\}$ has to be φ-middle, and any $w \in N_S(v)$ with list $\{j-2, j-1, j\}$ or $\{j, j+1, j+2\}$ cannot be φ-middle.

Thus deciding if one vertex of S is φ-middle, already determines for every vertex of S if it is φ-middle or not. It is described more formally in the following claim.

Claim 1.16.2. *In polynomial time we can either (1) construct a partition (U_1, U_2) of $V(S)$ (U_1, U_2 might be empty) such that for every list homomorphism $\varphi : (S, L) \to C_{2k+1}$, either all vertices of U_1 are φ-middle and no vertex of U_2 is φ-middle, or all vertices of U_2 are φ-middle and no vertex of U_1 is φ-middle, or (2) conclude that we are dealing with a no-instance.*

Proof of Claim: Fix $v \in V(S)$. We start with $U_1 := \{v\}$ and $U_2 := \emptyset$. Let $U = U_1 \cup U_2$ and $j = |U| + 1$. We set $v_1 := v$. As long as $j \leqslant |S|$, we proceed as follows. We set v_j to be a vertex in $N(U_1 \cup U_2)$ – since U_1 is non-empty, $j \leqslant |S|$, and S is connected, such v_j always exists. Moreover, let N_j^1 be the set of neighbors of v_j with list $L(v_j)$ and let $N_j^2 = N(v_j) \setminus N_j^1$.

If (a) $N_j^1 \cap U \subseteq U_1$ and $N_j^2 \cap U \subseteq U_2$, then add v_j to U_2. If (b) $N_j^1 \cap U \subseteq U_2$ and $N_j^2 \cap U \subseteq U_1$, then add v_j to U_1. If none of the two cases holds, return NO. In the first

two cases we repeat the procedure. Clearly, the above procedure can be performed in polynomial time.

Let us verify the correctness. We will show the following.

($\star$) For every $j \in [|S|]$, for every list homomorphism $\varphi : (S[\{v_1, \ldots, v_j\}], L) \to C_{2k+1}$, it holds that either (i) all vertices of $U_1 \cap \{v_1, \ldots, v_j\}$ are φ-middle and no vertex of $U_2 \cap \{v_1, \ldots, v_j\}$ is φ-middle, or (ii) all vertices of $U_2 \cap \{v_1, \ldots, v_j\}$ are φ-middle and no vertex of $U_1 \cap \{v_1, \ldots, v_j\}$ is φ-middle.

We prove it by induction on j. For $j = 1$, we have $U_1 = \{v\}$ and $U_2 = \emptyset$, so the statement clearly follows. So now assume that the statement is true for some $i \in [|S| - 1]$. Suppose there is a list homomorphism $\varphi : (S[\{v_1, \ldots, v_{i+1}\}], L) \to C_{2k+1}$. By inductive assumption, for $\varphi|_{\{v_1,\ldots,v_i\}}$, either (i) every vertex of $\{v_1, \ldots, v_i\} \cap U_1$ is φ-middle and no vertex of $\{v_1, \ldots, v_i\} \cap U_2$ is φ-middle or (ii) every vertex of $\{v_1, \ldots, v_i\} \cap U_2$ is φ-middle and no vertex of $\{v_1, \ldots, v_i\} \cap U_1$ is φ-middle.

Case 1: v_{i+1} is φ-middle. By Claim 1.16.1 (1.), all neighbors of v_{i+1} in $\{v_1, \ldots, v_i\}$ with lists different than $L(v_{i+1})$ have to be φ-middle, so in case (i) all such neigbors have to be in U_1, and in case (ii) all such neighbors have to be in U_2. Furthermore, by Claim 1.16.1 (1.), all neighbors of v in $\{v_1, \ldots, v_i\}$ with list $L(v_{i+1})$ cannot be φ-middle, so in case (i) all such neigbors have to be in U_2, and in case (ii) all such neighbors have to be in U_1. Recall, that if $N_j^1 \cap U \subseteq U_2$ and $N_j^2 \cap U \subseteq U_1$ which is the case in (i), we added v_{i+1} to U_1, so the claim follows. Similarly, if $N_j^1 \cap U \subseteq U_1$ and $N_j^2 \cap U \subseteq U_2$ which is the case in (ii), we added v_{i+1} to U_2, and the claim also follows in this case.

Case 2: v_{i+1} is not φ-middle. By Claim 1.16.1 (2.), any neighbor of v_{i+1} in $\{v_1, \ldots, v_i\}$ with list different than $L(v_{i+1})$ cannot be φ-middle, so in case (i) all such neighbors have to be in U_2, and in case (ii) all such neighbors have to be in U_1. Furthermore, by Claim 1.16.1 (2.), all neighbors of v in $\{v_1, \ldots, v_i\}$ with list $L(v_{i+1})$ have to be φ-middle, so in case (i) all such neighbors have to be in U_1, and in case (ii) all such neighbors have to be in U_2. But then in cases (i) and (ii), respectively, we added v_{i+1} to U_2 and U_1, and in each of the cases the claim follows.

This completes the proof of ($\star$).

Finally, observe that since ($\star$) is true, if (G, L) is a yes-instance, then for any $j \in [|S|]$, for a vertex $v_j \in U_i$, where $\{i, i'\} = \{1, 2\}$, all neighbors of v_j with list $L(v_j)$ have to be

in $U_{i'}$ and all neighbors of v_j with list different than $L(v_j)$ have to be in U_i. Therefore, if we returned NO, then indeed, (G, L) is a no-instance. This completes the proof of the claim. ⌟

Therefore, for every connected component S of $G[V_3]$ we solve two subinstances:

1. $I_1(S) = (S, L_1)$, where for every $v \in V(S)$ with list $\{i-1, i, i+1\}$ for some $i \in [2k]_0$, $L_1(v) = \{i\}$ if $v \in U_1$ and $L_1(v) = \{i-1, i+1\}$ if $v \in U_2$,

2. $I_2(S) = (S, L_2)$, where for every $v \in V(S)$ with list $\{i-1, i, i+1\}$ for some $i \in [2k]_0$, $L_2(v) = \{i\}$ if $v \in U_2$ and $L_2(v) = \{i-1, i+1\}$ if $v \in U_1$.

Note that both subinstances have all lists of size at most two and thus can be solved in polynomial time by Theorem 5.4. If for some component in both cases we obtain NO, then we return NO.

Creating a BCSP instance. Let $(V(G), L, C) = \text{BCSP}(C_{2k+1}, G[V_1 \cup V_2], L)$ – note that since we only consider vertices of $V_1 \cup V_2$, all lists have size at most two. We will now modify the instance $(V(G), L, C)$ so that it is equivalent to the instance (G, L) of $\text{LHOM}(C_{2k+1})$. For every $v \in V_3$ and for every pair of $u, w \in N(v) \cap V_2$, we leave in $C(u, w)$ only these pairs of vertices that have a common neighbor in $L(v)$ – recall that by Lemma 5.17 this ensures us that there will be a color left for v. Furthermore, for every connected component S of $G[V_3]$, we add constraints according to which of the two possibilities S can be properly colored (possibly S can be colored in both cases) as follows. Let $\{p, p'\} = \{1, 2\}$.

1. If there is no list homomorphism $\varphi : (S, L) \to C_{2k+1}$ such that all vertices of U_p are φ-middle and all vertices of $U_{p'}$ are not φ-middle, then for $v \in V(S)$ with $L(v) = \{i-1, i, i+1\}$, if $v \in U_p$, we remove $i-1, i+1$ from the lists of neighbors of v, and if $v \in U_{p'}$, we remove i from the lists of neighbors of v.

2. Moreover, for every pair $u, v \in V(S)$ with $L(u) = \{j-1, j, j+1\}$ and $L(v) = \{i-1, i, i+1\}$, for every $u' \in V_2 \cap N(u)$, and for every $v' \in V_2 \cap N(v)$, if $u, v \in U_p$, then we remove from $C(u', v')$ the pairs $(j, i+1)$, $(j, i-1)$, $(j-1, i)$, $(j+1, i)$, and if $v \in U_p$, $v \in U_{p'}$, then we remove from $C(u', v')$ the pairs (j, i), $(j-1, i-1)$, $(j-1, i+1)$, $(j+1, i-1)$, $(j+1, i+1)$.

Let us describe the intuition behind both types of constraints. Suppose that we have colored $G[V_1 \cup V_2]$ so that all added constraints are satisfied and let S be a connected of $G[V_3]$. Constraints added in 1. ensure that if one of the instances $I_1(S)$ and $I_2(S)$ is a no-instance, then the colors left for vertices of S correspond only to that $I_p(S)$ for $p \in \{1, 2\}$, which is a yes-instance. Furthermore, 2. ensures us that the colors left for vertices of S correspond to the same instance $I_p(S)$.

This completes the construction of BCSP instance $(V(G), L, C)$. Recall that all lists have size at most 2 and thus by Theorem 5.4 we solve $(V(G), L, C)$ in polynomial time.

Correctness. It remains to show that $(V(G), L, C)$ is equivalent to (G, L). First suppose that there is a list homomorphism $\varphi : (G, L) \to C_{2k+1}$. Let us define $f = \varphi|_{V_1 \cup V_2}$. Clearly f respects the lists and the constraints coming from BCSP(C_{2k+1}, G, L). Furthermore, since φ is a homomorphism, for every $v \in V_3$ and every pair $u, w \in V_2$ of neighbors of v, it holds that $f(u)$ and $f(w)$ have a common neighbor in $L(v)$. Suppose now that for some vertex v with $L(v) = \{i-1, i, i+1\}$, for $v' \in V_2 \cap N(v)$, we removed from $L(v')$ color $\varphi(v')$. Let S be the connected component of $G[V_3]$ containing v. If $\varphi(v') = i$, then we must have $\varphi(v) \in \{i-1, i+1\}$ – but we only removed i from $L(v')$ if there was no list homomorphism from $(G[S], L)$ to C_{2k+1} mapping v to one of $\{i-1, i+1\}$, a contradiction. Similarly, if $\varphi(v') \in \{i-1, i+1\}$, then we must have $\varphi(v) = i$, but we only removed $i-1, i+1$ from $L(v')$ if there was no list homomorphism from $(G[S], L)$ to C_{2k+1} mapping v to i, a contradiction.

Finally, let us verify the constraints added in the last step. Let S be a connected component of $G[V_3]$, let $u, v \in V(S)$ with $L(v) = \{i-1, i, i+1\}$ and $L(u) = \{j-1, j, j+1\}$, let $v' \in N(v) \cap V_2$ let $u' \in N(u) \cap V_2$. Suppose that $(f(u'), f(v')) \notin C(u', v')$ because of the last step. Let $\{p, p'\} = \{1, 2\}$ and let $v \in U_p$. If $u \in U_p$, then either both u, v are φ-middle or none of them. In the first case $\varphi(v') = i$ and $\varphi(u') = j$ and in the second case $\varphi(v') \in \{i-1, i+1\}$ and $\varphi(u') \in \{j-1, j+1\}$. Recall that for $u, v \in U_p$, we did not remove any pair from $(\{i-1, i+1\} \times \{j-1, j+1\}) \cup \{(i, j)\}$, a contradiction. So suppose that $u \in U_{p'}$. Then v is φ-middle if and only if u is not φ-middle. Therefore, we either have $\varphi(v') \in \{i-1, i+1\}$ and $\varphi(u') = j$, or $\varphi(v') = i$ and $\varphi(v') = i$ and $\varphi(u') \in \{j-1, j+1\}$, but again no such a pair was removed in the last step, a contradiction.

So now suppose that there is $f : V(G) \to [2k]_0$ that satisfies all constraints of $(V(G), L, C)$. Define $\varphi(v) = f(v)$ for $v \in V_1 \cup V_2$. Note that since f satisfies all con-

straints, φ is a list homomorphism on $G[V_1 \cup V_2]$. We have to show that φ can be extended to vertices of V_3. Consider a connected component S of $G[V_3]$. Since the algorithm did not return NO, there is a list homomorphism $\varphi_S : (S, L) \to C_{2k+1}$. First, note that for any $v \in V_3$, its neighbors from V_2 are colored so that there is a color left on $L(v)$ for v. Moreover, recall that if $L(v) = \{i-1, i, i+1\}$, then the possible lists of neighbors of v in V_2 are $\{i-1, i\}$ and $\{i, i+1\}$. Therefore, either all neighbors of v in V_2 are colored with i or with $i-1$ and $i+1$. In the first case, colors left for v are both $i-1$ and $i+1$, and in the second case i is left for v. Furthermore, let $\{p, p'\} = \{1, 2\}$ and let $u, v \in V(S)$. If $u, v \in U_p$, then by the constraints added in 2., the colors left for u, v allow both of them to be φ-middle, or both of them to be not φ-middle. Similarly, if $v \in U_p$ and $u \in U_{p'}$, then the colors left for u, v allow one of them to be φ-middle and the other to be not φ-middle. Therefore, if we leave on the lists of vertices of S only those colors that match $\varphi[V_1 \cup V_2]$, we obtain one of the instances $I_1(S)$, $I_2(S)$, and by the constraints added in 1., we can only obtain a yes-instance. Thus we can extend φ to all vertices of S. This completes the proof. □

Odd cycles – subexponential-time algorithm

In this section we prove Theorem 1.17. Recall that for $k = 1$ Theorem 1.17 is precisely Theorem 1.11.

Theorem 1.17. *Let $k \geqslant 1$. Then LHOM(C_{2k+1}) can be solved in time $\exp\left(\mathcal{O}\left((n \log n)^{\frac{k+1}{k+2}}\right)\right)$ on diameter-$(k+2)$ n-vertex graphs.*

We start with defining branching rules crucial for our algorithm – actually they are analogous to the ones in Theorem 1.11. Recall that for an instance (G, L) of LHOM(C_{2k+1}), by V_ℓ we denote the subset of $V(G)$ such that for every $v \in V_\ell$, we have $|L(v)| = \ell$, and $V_{\geqslant \ell} = \bigcup_{i \geqslant \ell} V_i$.

Branching rules. Let $k \geqslant 2$, let $I = (G, L)$ be an instance of LHOM(C_{2k+1}), and let $d \geqslant \operatorname{diam}(G)$ – note that for Theorem 1.17 it suffices to consider only $d = k+1$, but we do it more generally so we can later use it also for other values of d. Let $\mu = \sum_{\ell=2}^{2k+1} \ell \cdot |V_\ell|$. We define the following branching rules.

(B1) For a vertex $v \in V_{\geqslant 2}$ and for a color $a \in L(v)$, we branch on coloring v with a or not, i.e., we create two subinstances of I: $I_a = (G, L_a)$, $I'_a = (G, L'_a)$ such

that $L_a(u) = L'_a(u) = L(u)$ for every $u \in V(G) \setminus \{v\}$, and $L_a(v) = \{a\}$ and $L'_a(v) = L(v) \setminus \{a\}$.

(B2) For a vertex v we branch on the coloring of $N^{\leqslant d-1}[v] \cap V_{\geqslant 2}$, i.e., for every mapping f of $N^{\leqslant d-1}[v] \cap V_{\geqslant 2}$ that respects the lists, we create a new subinstance $I_f = (G, L_f)$ such that $L_f(u) = L(u)$ for $u \notin N^{\leqslant d-1}[v] \cap V_{\geqslant 2}$ and $L_f(w) = \{f(w)\}$ for $w \in N^{\leqslant d-1}[v] \cap V_{\geqslant 2}$.

Algorithm `Recursion Tree`. Let us describe an algorithm that for fixed d takes an instance (G, L) of LHOM(C_{2k+1}) with a fixed precolored $(2k+1)$-cycle C and such that $\operatorname{diam}(G) \leqslant d$, and returns a rooted tree $\mathcal{R}$ whose nodes are labelled with subinstances of (G, L). We first introduce the root r of $\mathcal{R}$ and we label it with (G, L). Then for every node we proceed recursively as follows. Let s be a node labelled with an instance (G', L') of LHOM(C_{2k+1}). We first exhaustively apply to (G', L') reduction rules (R1), (R2), (R3), and if some of the reduction rules returns NO, then s does not have any children. Otherwise, if there is a vertex $v \in V_{\geqslant 2}$ with at least $(\mu \log \mu)^{1/d}$ neighbors in $V_{\geqslant 2}$, then we will apply (B1) for v and for $a \in L(v)$ chosen as follows. If on $N_{G'[V_{\geqslant 2}]}(v)$ there are no lists of type (2), then we take any $a \in L(v)$. Otherwise, let S be the most frequent list of type (2) on $N_{G'[V_{\geqslant 2}]}(v)$, and let $S = \{j-1, j+1\}$ for some $j \in [2k]_0$. Then we take any $a \in L(v) \setminus \{j\}$. After application of (B1), we exhausively apply reduction rules (R1), (R2), (R3), to each instance. Furthermore, for each instance created by (B1), we create a child node of s and we label it with that instance.

If there is no vertex $v \in V_{\geqslant 2}$ with at least $(\mu \log \mu)^{1/d}$ neighbors in $V_{\geqslant 2}$, then we apply the branching rule (B2) for some $v \in V_{\geqslant 2}$, we exhausively apply reduction rules (R1), (R2), (R3), and again for each instance created by (B2), we introduce a child node of s. The choice of v is not completely arbitrary. If possible, we choose v so that $\operatorname{dist}(v, C) \geqslant \lceil \frac{d}{2} \rceil$ – note that the cycle C is present in all instances of $\mathcal{R}$. The nodes corresponding to instances created by (B2) are leaves, i.e., we do not recurse on the children of s for which we applied (B2).

Let us analyze the running time of the algorithm `Recursion Tree` and properties of the constructed tree $\mathcal{R}$.

Lemma 5.20. *Given an instance (G, L) of LHOM(C_{2k+1}) with a fixed precolored $(2k+1)$-cycle C and such that $n = |G|$, $\operatorname{diam}(G) \leqslant d$, the algorithm `Recursion Tree` in*

time $\exp\left(\mathcal{O}((n\log n)^{\frac{d-1}{d}})\right)$ *returns a tree* $\mathcal{R}$ *whose nodes are labelled with subinstances of* (G,L)*, and* (G,L) *is a yes-instance if and only if at least one subinstance corresponding to a leaf of* $\mathcal{R}$ *is a yes-instance.*

Proof. First, we show that for every node s of $\mathcal{R}$ the corresponding instance is a yes-instance if and only if at least one instance corresponding to a child of s is a yes-instance. Let s be a node of $\mathcal{R}$ and let (G',L') be the corresponding instance. The algorithm `Recursion Tree` first applies reduction rules to (G',L') and by Lemma 5.2, we obtain an equivalent instance. Furthermore, we applied to (G',L') either (B1) or (B2) where the branches correspond to all possible colorings of some set of vertices, so indeed (G',L') is a yes-instance if and only if at least one instance corresponding to a child of s is a yes-instance. Since the root of $\mathcal{R}$ is labelled with (G,L), we conclude that (G,L) is a yes-instance if and only if at least one instance corresponding to a leaf of $\mathcal{R}$ is a yes-instance.

It remains to analyze the running time. Let $F(\mu)$ be an upper bound on the running time of `Recursion Tree` applied to an instance (G',L') with $\mu=\sum_{\ell=2}^{2k+1}\ell\cdot|V_\ell|$ and let $\mathsf{p}(n)=\mathcal{O}(n^3)$ be a polynomial such that exhaustive application of reduction rules (R1), (R2), (R3) to an n-vertex instance can be performed in time $\mathsf{p}(n)$. Observe that if we apply (B1) to (G',L'), then we obtain

$$F(\mu)\leqslant F\left(\mu-\frac{(\mu\log\mu)^{1/d}}{2k+1}\right)+F(\mu-1)+2\cdot\mathsf{p}(n).$$

Indeed, let v,a be, respectively, the vertex and the color to which we apply (B1) – recall that v has at least $(\mu\log\mu)^{1/d}$ neighbors in $V_{\geqslant 2}$ If there are no lists of type (2) on $N_{G'[V_{\geqslant 2}]}(v)$, then in the branch where we set $L(v)=\{a\}$, after application of reduction rules, every neighbor of v must have $L(v)\subseteq\{a-1,a+2\}$. If $|L(v)|\geqslant 2$ and $L(v)\neq\{a-1,a+1\}$, then $|L(v)|\cap\{a-1,a+1\}|<|L(v)|$. Therefore, in this case we decrease sizes of all lists on $N_{G'[V_{\geqslant 2}]}(v)$. Otherwise, we chose $a\in L(v)\setminus\{j\}$, where $\{j-1,j+1\}$ is the most frequent list of type (2) on $N_{G'[V_{\geqslant 2}]}(v)$. Since there are exactly $2k+1$ lists of type (2), at least $\frac{1}{2k+1}$-fraction of $N_{G'[V_{\geqslant 2}]}(v)$ has list of different type than (2) or has list $\{j-1,j+1\}$. Thus, for the branch where we set $L(v)=\{a\}$, the sizes of lists of at least $\frac{1}{2k+1}\cdot(\mu\log\mu)^{1/d}$ vertices decrease. In the branch where we remove a from $L(v)$, we decrease the size of $L(v)$ at least by one. In both branches we apply the reduction rules, so the desired inequality follows.

If we apply (B2) to (G', L') – recall that we stop recursing in this case – then we obtain

$$F(\mu) \leqslant (2k+1)^{(\mu \log \mu)^{\frac{d-1}{d}}} \cdot \mathsf{p}(n),$$

since we guess the coloring on $N^{\leqslant d-1}_{G'[V_{\geqslant 2}]}(v)$ whose size is bounded by $(\mu \log \mu)^{\frac{d-1}{d}}$ (in this case the degrees in $G'[V_{\geqslant 2}]$ are bounded by $(\mu \log \mu)^{1/d}$) and the number of possible colors is at most $2k+1$.

We can conclude that $F(\mu) \leqslant 2^{\mathcal{O}((\mu \log \mu)^{\frac{d-1}{d}})}$ (see Appendix) which, combined with the inequality $\mu \leqslant (2k+1)n = \mathcal{O}(n)$, completes the proof. □

In the following lemma we show that we can solve every instance corresponding to a leaf of $\mathcal{R}$ in polynomial time.

Lemma 5.21. *Let (G', L') be an instance of LHOM(C_{2k+1}) such that $\operatorname{diam}(G') \leqslant k+2$ and let C be a fixed precolored $(2k+1)$-cycle. Assume that we applied the algorithm* `Recursion Tree` *to (G', L') and let $\mathcal{R}$ be the resulting recursion tree. Let (G, L) be an instance corresponding to a leaf in $\mathcal{R}$. Then (G, L) can be solved in polynomial time.*

In order to prove Lemma 5.21, we first prove that we can solve every instance whose lists are of special form in polynomial time.

Lemma 5.22. *Let $k \geqslant 2$ and let (G, L) be a reduced instance of LHOM(C_{2k+1}) such that G is connected and for every vertex $v \in V(G)$, the list $L(v)$ either has size at most 2 or is of type $(2,2)$. Then (G, L) can be solved in polynomial time.*

Proof. If there are no lists of size 3, then we only have lists of size at most 2, and thus (G, L) can be solved in polynomial time by Theorem 5.4. So let $v \in V_3$ be a vertex with list $L(v) = \{i-2, i, i+2\}$ for some $i \in [2k]_0$ and let u be a neighbor of v. Observe that since the reduction rule (R3) cannot be applied, if $u \in V_2$, then u must have one of the lists: $\{i-1, i+1\}$ $\{i-3, i+1\}$, $\{i-1, i+3\}$, and if $u \in V_{\geqslant 3}$, then u must have one of the lists $\{i-3, i-1, i+1\}$, or $\{i-1, i+1, i+3\}$ (see Figure 5.3).

If for some vertex v with list $\{i-2, i, i+2\}$ there is no $u \in N(v)$ with one of lists $\{i-1, i+1\}$, $\{i-3, i+1\}$, $\{i-1, i+3\}$, then we add such a vertex u to G and make it adjacent to u. Note that now the diameter of G might increase, but in this lemma we only need G to be connected. Moreover, any list homomorphism on $G-u$ can be extended to u, since each color on $L(v)$ has a neighbor on $L(u)$. So since now, we can assume that

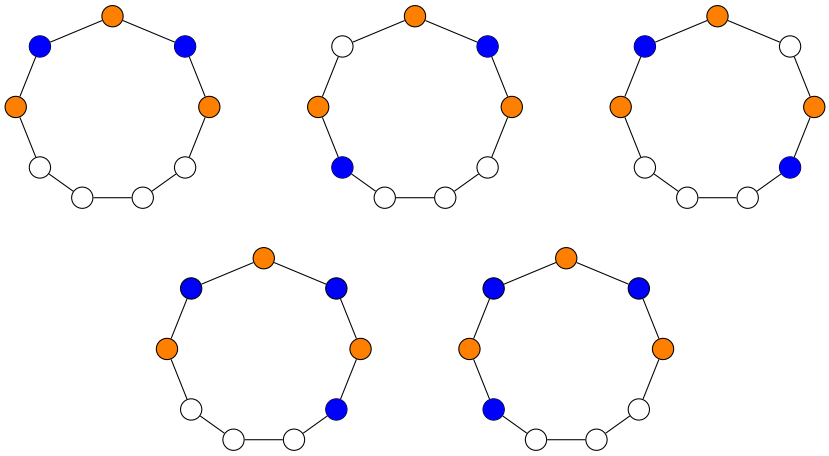

Figure 5.3: Case $k = 4$. Orange vertices denote a list of some vertex $v \in V_3$, blue vertices denote all possible lists of a neighbor u of v when (R3) cannot be applied, i.e., every vertex of $L(u)$ is a neighbor of a vertex of $L(v)$ and every vertex of $L(v)$ is a neighbor of a vertex of $L(u)$.

($\star$) for a vertex v with list $\{i-2, i, i+2\}$, all lists $\{i-1, i+1\}$, $\{i-3, i+1\}$, $\{i-1, i+3\}$ are present on $N(v)$.

Constructing a BCSP instance. We construct an instance of BCSP as follows. We start with BCSP$(C_{2k+1}, G - V_3, L)$ – note that in this instance all lists have size at most 2. First, for every vertex $v \in V_3$ and for every $v', v'' \in V_2 \cap N(v)$, we leave in $C(v', v'')$ only such pairs that have a common neighbor in $L(v)$. Furthermore, for every edge uv with $u, v \in V_3$ such that $L(u) = \{i-2, i, i+2\}$, $L(v) = \{i-1, i+1, i+3\}$, and for every pair $u', v' \in V_2$ such that $uu', vv' \in E(G)$, we remove (if they are present) from $C(u', v')$ the following pairs: $(i-3, i+2)$, $(i-1, i+4)$, and $(i+3, i-2)$. This completes the construction of the instance (V, L, C) of BCSP.

Clearly the instance (V, L, C) is constructed in polynomial time. Moreover, since all lists of (V, L, C) have size at most 2, by Theorem 5.4, this instance can be solved in polynomial time.

Correctness. It remains to show that the instance (G, L) is equivalent to the instance (V, L, C). First suppose that there is a list homomorphism $\varphi : (G, L) \to C_{2k+1}$. Consider the assignment $f = \varphi|_{V_1 \cup V_2}$. We have to verify that f satisfies all constraints of the instance (V, L, C). Clearly f satisfies all constraints coming from BCSP$(C_{2k+1}, G - V_3, L)$. Furthermore, since φ is a homomorphism from G to C_{2k+1}, for every $v \in V_3$ and every

pair $v', v'' \in N(v) \cap V_2$, we have that $f(v')$ and $f(v'')$ must have a common neighbor in $L(v)$. Finally, we have to verify that f satisfies constraints that were introduced for edges of $G[V_3]$. Suppose that there are u', v' such that $(f(u'), f(v'))$ was removed from $C(u', v')$ for an edge uv of $G[V_3]$ such that $uu', vv' \in E(G)$. Let $L(u) = \{i-2, i, i+2\}$ and $L(v) = \{i-1, i+1, i+3\}$. Suppose first that $f(u') = i-3$ and $f(v') = i+2$. Since φ is a list homomorphism and $uu' \in E(G)$, we must have $\varphi(u) = i-2$, and since $uv \in E(G)$ we must have $\varphi(v) = i-1$. However, $i-1$ is non-adjacent to $i+2$ and thus the edge vv' cannot be properly colored, a contradiction. Now suppose that $f(u') = i-1$ and $f(v') = i+4$. Similarly, we must have $\varphi(v) = i+3$ and $\varphi(u) \in \{i-2, i\}$ so the edge uv cannot be properly colored. Finally, suppose that $f(u') = i+3$ and $f(v') = i-2$. Then we must have $\varphi(u) = i+2$ and $\varphi(v) = i-1$ so again uv cannot be properly colored, a contradiction. Therefore, f satisfies all constraints.

Now suppose that there is a satisfying assignment $f : V \to V(H)$ of (V, L, C). Observe that if we consider $\varphi = f$ on $V_1 \cup V_2$, then φ is a list homomorphism on $G[V_1 \cup V_2]$. It remains to show that φ can be extended to vertices of V_3. Consider a vertex $v \in V_3$ with list $\{i-2, i, i+2\}$ for some $i \in [2k]_0$. Recall that by $(\star)$, v has neighbors with lists $\{i-3, i+1\}$ and $\{i-1, i+3\}$. Since f satisfies the constraints, for every pair of neighbors v', v'' of v in V_2, $f(v') \cap f(v'') \cap L(v) \neq \emptyset$, and thus by Lemma 5.17, there is a common neighbor of the colors on $N(v)$ in $L(v)$. Furthermore, we claim that this common neighbor is unique. Indeed, if the neighbor of v with list $\{i-3, i+1\}$ is colored with $i-3$, then we already know that v has to be mapped to $i-2$. Otherwise, v has to be mapped to one of $i, i+2$. If the neighbor of v with list $\{i-1, i+3\}$ is mapped to $i+3$, then v has to be mapped to $i+2$, otherwise, v has to be mapped to i. Therefore, we extend φ to vertices of V_3 in the only possible way. It remains to show that φ respects the edges of V_3. Suppose not and let uv be an edge which is not properly colored. Let $L(u) = \{i-2, i, i+2\}$ and $L(v) = \{i-1, i+1, i+3\}$. First suppose that $\varphi(u) = i-2$ and $\varphi(v) \in \{i+1, i+3\}$. A neighbor of u with list $\{i-3, i+1\}$ has to be mapped to $i-3$ and a neighbor of v with list $\{i-2, i+2\}$ has to be mapped to $i+2$, but then the mapping f does not satisfy the constraints since we removed the pair $(i-3, i+2)$. So now suppose that u is mapped to i and v is mapped to $i+3$. Then a neighbor of v with list $\{i, i+4\}$ is mapped to $i+4$ and a neighbor of u with list $\{i-1, i+3\}$ is mapped to $i-1$, but the pair $(i-1, i+4)$ was removed. Finally suppose that u is mapped to $i+2$ and v is mapped to $i-1$. Then

a neighbor of u with list $\{i-1, i+3\}$ is mapped to $i+3$ and a neighbor of v with list $\{i-2, i+2\}$ is mapped to $i-2$, which is a contradiction because we removed the pair $(i+3, i-2)$. Therefore φ is a list homomorphism, which completes the proof. □

Now we can prove Lemma 5.21.

Proof of Lemma 5.21. Recall that the instance (G, L) is reduced. Moreover, we check if G contains an odd cycle of length at most $2k-1$, and if so, we return NO since by Observation 3.2, there is no homomorphism from G to C_{2k+1} By Lemma 5.22, it is enough to show that every vertex of $V_{\geqslant 3}$ has list of type $(2,2)$. First observe that for every vertex u outside the cycle C, we have that $L(u) \subseteq \{i-2, i-1, i, i+1, i+2\}$ for some $i \in [2k]_0$. Indeed, by Lemma 5.18, there must be $i \in [2k]_0$ such that $\operatorname{dist}(u, c_i) = \operatorname{dist}(u, c_{i+1}) =: \ell \leqslant \operatorname{diam}(G) = k+2$. Similarly as in the proof of Theorem 1.16, it holds that $\ell \in \{k+1, k+2\}$, so by Lemma 5.16, $L(u) \subseteq \{i-2, i-1, i, i+1, i+2\}$, for some $i \in [2k]_0$.

Furthermore, observe that since for the branching rule (B2), if we could, we chose vertex v whose distance from C is at least $\lceil \frac{d}{2} \rceil$, each vertex of G is at distance $\lfloor \frac{k+2}{2} \rfloor$ from C. Indeed, every vertex u' that was in $N^{\leqslant k+2}_{G[V_{\geqslant 2}]}[v]$ either is already precolored or has a precolored neighbor after guessing the coloring on $N^{\leqslant k+1}_{G[V_{\geqslant 2}]}[v]$, and thus each such vertex u' has list of size at most 2. So for any vertex u that is still in $V_{\geqslant 3}$, the shortest u-v path (whose length is at most the diameter $\operatorname{diam}(G)$) should contain a vertex from C and length of that path is at least $\operatorname{dist}(v, C) + \operatorname{dist}(u, C)$. So either all vertices outside C were at distance at most $\lfloor \frac{k+2}{2} \rfloor$, or v was at distance at least $\lceil \frac{k+2}{2} \rceil$, and thus $\operatorname{dist}(u, C) \leqslant \operatorname{diam}(G) - \operatorname{dist}(v, C) \leqslant k + 2 - \lceil \frac{k+2}{2} \rceil = \lfloor \frac{k+2}{2} \rfloor$, so u is at distance at most $\lfloor \frac{k+2}{2} \rfloor$ from C.

For $k = 1$, we obtain that every vertex of G is at distance at most $\lfloor \frac{3}{2} \rfloor = 1$ from C. Therefore, by Lemma 5.15, we obtain that every vertex of G has list of size at most 2, and thus (G, L) can be solved in polynomial time by Theorem 5.4. So since now we can assume that $k \geqslant 2$. If $k > 2$, then $\lfloor \frac{k+2}{2} \rfloor < k$. Recall that by Lemma 5.15, for a vertex u that is at distance at most $k-1$ from a precolored vertex (and all vertices of C are precolored), the set $L(u)$ is an independent set. Combining it with the fact that each list of a vertex in $V_{\geqslant 3}$ is contained in $\{i-2, i-1, i, i+1, i+2\}$, for some $i \in [2k]_0$, we obtain that for every $u \in V_{\geqslant 3}$ we have $L(u) \subseteq \{i-2, i, i+2\}$, for some $i \in [2k]_0$. If $k = 2$, then each vertex is at distance at most 2 from C, and by Lemma 5.15, we obtain that for every $u \in V_{\geqslant 3}$, the list $L(u)$ is of type $(2,2)$, i.e., $L(u) = \{i-2, i, i+2\}$ for some $i \in [2k]_0$.

Therefore, every vertex of $V_{\geqslant 3}$ has list of type $(2,2)$, and thus, by Lemma 5.22, the instance (G,L) can be solved in polynomial time, which completes the proof. □

Now we are ready to prove Theorem 1.17.

Proof of Theorem 1.17. Let (G,L) be an instance of LHOM(C_{2k+1}) such that the diameter of G is at most $k+2$. As in Theorem 1.16, first for every $i \in [2k]_0$, we check in polynomial time whether there is a list homomorphism $\varphi : (G,L) \to C_{2k+1}$ such that no vertex is mapped to i – recall that this can be done by Lemma 5.19. If there is no such list homomorphism, we guess $2k+1$ vertices $c_0, \ldots, c_{2k}$ which will be colored so that c_i is mapped to i. We add the edges $c_i c_{i+1}$ and we obtain an induced $(2k+1)$-cycle C (if not, then we are dealing with a no-instance). Note that adding edges cannot increase the diameter and since the edges are added between vertices precolored with consecutive vertices, we obtain an equivalent instance.

Now for (G,L) and C as the fixed precolored $(2k+1)$-cycle we use the algorithm `Recursion Tree`, which by Lemma 5.20 in time $2^{\mathcal{O}((n \log n)^{\frac{k+1}{k+2}})}$ returns a tree $\mathcal{R}$. Moreover, in order to solve the instance (G,L) it is enough to solve every instance corresponding to a leaf of $\mathcal{R}$ by Lemma 5.20, and by Lemma 5.21, we can solve each such instance in polynomial time. Furthermore, since the size of $\mathcal{R}$ is bounded by the running time, the instance (G,L) can be solved in time $\exp\left(\mathcal{O}((n \log n)^{\frac{k+1}{k+2}})\right) \cdot n^{\mathcal{O}(1)} = \exp\left(\mathcal{O}((n \log n)^{\frac{k+1}{k+2}})\right)$, which completes the proof. □

We finish this section with a result which shows that it is sometimes possible to have a subexponential-time algorithm for LHOM(C_{2k+1}) for diameter-$(k+3)$ graphs, i.e., we show that LHOM(C_5) can be solved in polynomial time on diameter-5 graphs. It is not clear if this can be generalized to other values of k – here we use the fact that after standard branchings, every vertex is at distance at most $\lfloor \frac{k+3}{2} \rfloor$, which for $k=2$ is 2, from a precolored vertex. By Lemma 5.15, the list of every vertex is either of size at most two or of type $(2,2)$, and thus the instance can be solved in polynomial time by Lemma 5.22. For all larger values of k, we have $\lfloor \frac{k+3}{2} \rfloor \geqslant 3$, and thus the same argument does not work.

Theorem 5.23. *Every diameter-5 n-vertex instance (G,L) of LHOM(C_5) can be solved in time $\mathcal{O}\left(\exp\left((n \log n)^{\frac{4}{5}}\right)\right)$.*

Proof. Let (G,L) be an n-vertex instance of LHOM(C_5) such that $\operatorname{diam}(G) \leqslant 6$. First, as in Theorem 1.17, for every $i \in \{0,1,2,3,4\}$, by applying Lemma 5.19, we check in

polynomial time whether there is a list homomorphism $\varphi : (G, L) \to C_5$ such that no vertex is mapped to i. If no, we guess 5 vertices c_0, c_1, c_2, c_3, c_4 which will be colored so that c_i is mapped to i. We add the edges $c_i c_{i+1}$ and we obtain an induced 5-cycle C (if not, then we are dealing with a no-instance).

We use the algorithm `Recursion Tree` for (G, L) and C, which by Lemma 5.20 in time $2^{\mathcal{O}((n \log n)^{\frac{4}{5}})}$ returns a tree $\mathcal{R}$ such that in order to solve the instance (G, L) it is enough to solve every instance corresponding to a leaf of $\mathcal{R}$.

Let (G', L') be an instance corresponding to a leaf in $\mathcal{R}$. As in the proof of Lemma 5.21, since in (B2), if we could, we chose a vertex whose distance from C is at least $\lceil \frac{5}{2} \rceil$, each vertex of G' is at distance at most $\lfloor \frac{5}{2} \rfloor = 2$ from C. By Lemma 5.15, each vertex of $V_{\geqslant 3}$ has list of type $(2, 2)$, and thus by Lemma 5.22, (G', L') can be solved in polynomial time. This completes the proof. □

5.2.2 Hardness result

In this section we prove Theorem 1.18.

Theorem 1.18. *Let $k \geqslant 1$. The Hom(C_{2k+1}) problem is* NP*-hard on graphs of radius $k + 1$ (and thus diameter $(2k + 2)$) and cannot be solved in subexponential time, unless the ETH fails.*

Proof. We reduce from 3-Sat. Let Φ be an instance of 3-Sat with n variables $x_1, \ldots, x_n$ and m clauses $\gamma_1, \ldots, \gamma_m$. We assume that each clause has precisely three literals. We construct G as follows.

We start with a $(2k + 1)$-cycle C on vertices $\{v_0, \ldots, v_{2k}\}$. For every clause γ_j we add a copy of C_{2k+1} on vertices $a_j, b_j, c_j^1, c_j^2, \ldots, c_j^{2k-1}$ and we add edges $v_1 a_j, b_j v_2$. For every variable x_i, we add a copy of C_{2k+1} with vertices $x_i^0, \ldots, x_i^{2k}$ and identify x_i^0 with v_0.

For every clause γ_j, we fix ordering of its variables. Furthermore, for every variable x_i of γ_j we add a path P_{ij} as follows.

1. If x_i is the first variable of γ_j we add a path P_{ij} on $2k + 1$ vertices, identify its first vertex with a_j. Furthermore, for $\ell = 2, \ldots, 2k$, we make the ℓ-th vertex of the path adjacent to v_ℓ. Finally, if the occurence of x_i in γ_j is positive, then we identify the $(2k + 1)$th vertex of the path with x_i^{2k}. Otherwise, we identify the $(2k + 1)$th vertex with x_i^1.

2. If x_i is the second variable of γ_j, then we add a path P_{ij} on three vertices, identify the first vertex with b_j, and make the second vertex adjacent to v_1. Finally, if the occurence of x_i in γ_j is positive we identify the third vertex of the path with x_i^1. Otherwise, we identify it with x_i^{2k}.

3. If x_i is the third variable of γ_j, then we add a path P_{ij} on $k+2$ vertices, identify the first one with c_j^k, and make the $(k+1)$-th one adjacent to v_1. Finally, if the occurence of x_i in γ_j is positive, then we identify the last vertex of the path with x_i^1, and otherwise, we identify it with x_i^{2k}.

This completes the construction of G (see Figure 5.4). Note that $|V(G)| = \mathcal{O}(n+m)$. We first prove that ϕ is satisfiable if and only if $G \to C_{2k+1}$.

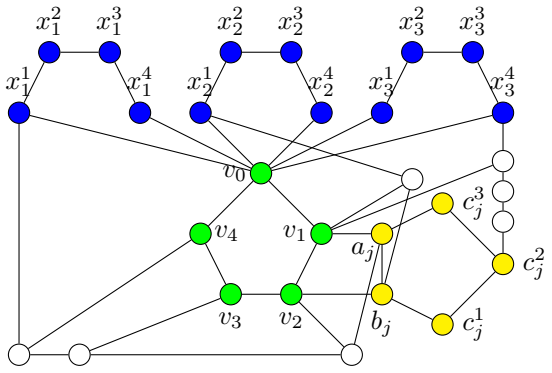

Figure 5.4: Construction of G for $k = 2$ and clause $\gamma_j = (\neg x_1 \vee x_2 \vee \neg x_3)$. Green vertices belong to the cycle C, blue vertices are those introduced for variables, and yellow ones are those introduced for the clause γ_j. Remaining vertices belong to paths P_{ij}.

Claim 1.18.1. *If $G \to C_{2k+1}$, then Φ is satisfiable.*

Proof of Claim: First suppose that there exists $\varphi : G \to C_{2k+1}$. Since C_{2k+1} is a core, φ is an automorphism. Without loss of generality, assume that $\varphi(v_i) = i$ for $i \in [2k]_0$. Note that for every variable, there are exactly two ways of coloring its corresponding copy of C_{2k+1}, i.e., one with $\varphi(x_i^{2k}) = 2k$, and the other with $\varphi(x_i^{2k}) = 1$. We define a truth assignment ψ of the variables, so that x_i is true if and only if $\varphi(x_i^{2k}) = 2k$.

Let us verify that ψ is a satisfying assignment of Φ. Consider a clause γ_j and its corresponding copy of C_{2k+1}. Observe that a_j is adjacent to vertex colored with 1 and

b_j is adjacent to a vertex colored with 2. Therefore, the pair (a_j, b_j) must be colored in one of three ways: $(0,1),(2,3),(2,1)$. First assume that (a_j, b_j) is colored with $(0,1)$ and let x_i be the first variable of C_j. Recall that P_{ij} is a path on $2k+1$ vertices with a_j as the first vertex. Moreover, for every $\ell = 2, \ldots, 2k$, the ℓth vertex of P_{ij} is adjacent to v_ℓ and we have $\varphi(v_\ell) = \ell$. Thus ℓth vertex of P_{ij} has to be mapped to one of $\{\ell - 1, \ell + 1\}$. Furthermore, $\varphi(a_j) = 0$, so the second vertex of P_{ij} has to be mapped to 1. Then ℓth vertex has to be mapped to $\ell - 1$, and thus the last vertex has to be mapped to $2k$. Recall that the last vertex of the path P_{ij} is x_i^{2k} if the occurence of x_i is positive and x_i^1, otherwise. In both cases by the definition of the truth assignment ψ, x_i satisfies C_j.

Now assume that the pair (a_j, b_j) is colored with $(2,3)$ and let x_i be the second variable of C_j. Then observe that the consecutive vertices of P_{ij} must be colored with $3, 2, 1$, respectively. Recall that the last vertex of P_{ij} is x_i^1 if the occurence of x_i is positive and x_i^{2k}, otherwise. In both cases by the definition of truth assignment ψ, x_i satisfies C_j.

Finally assume that the pair (a_j, b_j) is colored with $(2,1)$ and let x_i be the third variable of C_j. Observe that in this case c_j^k, which is also the first vertex of P_{ij}, must be colored with $k+2$. Since the $(k+1)$th vertex of P_{ij} is adjacent to v_1 colored with 1, it can be colored only with one of $0, 2$. However, if it is colored with 0, then there must be a walk in C_{2k+1} from 0 to $k+2$ of length exactly k, a contradiction. Therefore, the last vertex of P_{ij} must be colored with 1. As in the previous cases, x_i satisfies C_j. ⌟

Claim 1.18.2. *If Φ is satisfiable, then $G \to C_{2k+1}$.*

Proof of Claim: Let ψ be a truth assignment satisfying Φ. We define $\varphi : G \to C_{2k+1}$ as follows. First, we set $\varphi(v_i) := i$. Moreover, for every variable x_i, we extend φ to the vertices of the cycle introduced for x_i, so that $\varphi(x_i^1) = 1$ if $\psi(x_i) = 1$, and $\varphi(x_i^1) = 2k$ otherwise. Furthermore, for each clause γ_j we fix one variable x_i that satisfies γ_j in assignment ψ. Then, if the first variable satisfies γ_j, we color (a_j, b_j) with $(0,1)$, if it is the second variable, we color (a_j, b_j) with $(2,3)$, and if it is the third one, we color (a_j, b_j) with $(2,1)$. We extend φ to the remaining vertices of the cycle introduced for γ_j in the only possible way.

Observe that so far, φ respect all the edges. It remains to extend φ to vertices of paths P_{ij}. Consider such a path P_{ij} introduced for a clause γ_j and its variable x_i.

Case 1: x_i is the first variable of γ_j. In this case P_{ij} is a path on $2k+1$ vertices with a_j as the first vertex and x_i^1 or x_i^{2k} as the last vertex depending on the sign of the occurence of x_i in γ_j. Moreover, for $\ell = 2, \ldots, 2k$, the ℓth vertex of P_{ij} is adjacent to v_ℓ, so it has to be mapped to one of $\ell-1, \ell+1$. If a_j is colored with 2, then φ can be extended to P_{ij} so that the $2k$ consecutive vertices are mapped respectively to $2, 3, 4 \ldots, 2k, 0$. Now note that this respects all the edges of P_{ij} as the last vertex is colored either with 1 or $2k$, both adjacent to 0. So now assume that a_j is colored with 0. By the definition of φ, the variable x_i must satisfy γ_j, and thus the last vertex of P_{ij} has to be mapped to $2k$. Then we can extend φ to P_{ij} so that the consecutive vertices of P_{ij} are mapped respectively to $0, 1, \ldots, 2k$.

Case 2: x_i is the second variable of γ_j. Then P_{ij} is a path on 3 vertices with b_j being the first vertex. Moreover, the second vertex is adjacent to v_1, so it has to be mapped to 0 or 2. If b_j is mapped to 1, then we set φ on the second vertex of P_{ij} to 0, which is adjacent to both $1, 2k$, and thus φ respects the edges of P_{ij}. If b_j is mapped to 3, then the variable x_i satisfies γ_j and thus the last vertex of P_{ij} has to be mapped to 1. Therefore we can set φ on the second vertex of P_{ij} to 2.

Case 3: x_i is the third variable of γ_j. In this case P_{ij} is a path on $k+2$ vertices with c_j^k being the first vertex and the $(k+1)$th vertex adjacent to v_1. Note that if (a_j, b_j) is mapped to $(0,1)$, $(2,3)$, or $(2,1)$, then c_j^k is mapped respectively to $k+1$, $k+3$, $k+2$. In the first case, we can extend φ so that consecutive $k+1$ vertices of P_{ij} are mapped respectively to $k+1, k+2, \ldots, 2k, 0$ and since the $(k+1)$th vertex is mapped to 0 adjacent to both $1, 2k$, φ respects all the edges of P_{ij}. In the second case we extend φ to P_{ij} so that consecutive $k+1$ vertices of P_{ij} are mapped respectively to $k+3, k+4, \ldots, 2k, 0, 1, 0$ and again φ respects the edges of P_{ij}. So let us consider the third case. Recall that this happens when x_i satisfies C_j and thus the last vertex of P_{ij} is mapped to 1. We extend φ so that the consecutive vertices of P_{ij} are mapped respectively to $k+2, k+1, k, \ldots, 2, 1$, which clearly respects the edges of P_{ij}. This completes the proof of the claim. ⌟

Now we show that the radius (and thus the diameter) of G is bounded.

Claim 1.18.3. *The radius of G is at most $k+1$.*

Proof of Claim: We show that each vertex is at distance at most $k+1$ from v_1. It holds

for every vertex of the cycle C and every vertex that is adjacent to C. The remaining vertices are those introduced for variables, the vertices of cycles introduced for clauses, and those of paths P_{ij} introduced for clauses and their third variables.

The vertices introduced for variables are at distance at most k from v_0, and thus at most $k+1$ from v_1. For a cycle introduced for a clause γ_j, each its vertex is at distance at most k from a_j, which is adjacent to v_1. Finally, it remains to check the internal vertices of the paths introduced in the third case. Recall that each such a path consist of $k+2$ vertices (k internal vertices) and the $(k+1)$-th vertex is adjacent to v_1. This completes the proof of the claim. ⌟

Therefore the $\textsc{Hom}(C_{2k+1})$ problem is NP-hard on diameter-$(2k+2)$ graphs. Moreover, since $|G| = \mathcal{O}(n+m)$, there is no algorithm solving $\textsc{Hom}(C_{2k+1})$ in time $2^{o(|G|)} \cdot |G|^{\mathcal{O}(1)}$, unless the ETH fails. This completes the proof. □

Chapter 6

Other results

In this chapter we shortly describe other results of the author that were not selected for the dissertation, which are the following.

[121] K. Okrasa, M. Piecyk, and P. Rzążewski. Full Complexity Classification of the List Homomorphism Problem for Bounded-Treewidth Graphs. ESA 2020, volume 173 of LIPIcs, pages 74:1–74:24, Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2020.

[46] M. Dębski, Z. Lonc, K. Okrasa, M. Piecyk, and P. Rzążewski. Computing Homomorphisms in Hereditary Graph Classes: The Peculiar Case of the 5-Wheel and Graphs with No Long Claws. ISAAC 2022, volume 248 of LIPIcs, pages 14:1–14:16, Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2022.

[90] K. Kluk, H. La, and M. Piecyk. Graph Reconstruction with Connectivity Queries. To appear in WG 2024 Proceedings.

List homomorphisms and treewidth. The problem considered in [121] is LHOM(H) parameterized by the treewidth $\mathrm{tw}(G)$ of the input graph G. Arguably, treewidth is one of the most studied graph parameters [9, 52, 55, 55, 61, 62, 101, 108, 121, 124]. Recall, that for LHOM(H) it makes sense to consider H that contains a vertex with a loop or is bipartite. The dichotomy for LHOM(H) was proven in three steps: (i) for reflexive target graphs [56], (ii) for bipartite target graphs [57], and (iii) for general target graphs [58].

In case of LHOM(H) parameterized by treewidth, the problem was first studied by Egri, Marx, and Rzążewski [52] for reflexive graphs. In the paper [121], we consider

general target graphs H. For every H such that LHOM(H) is NP-hard, we define an invariant $i^*(H)$ and show the following.

Theorem 6.1 ([121])**.** *Let H be a graph such that LHOM(H) is NP-hard.*

1. *Even if H is part of the input, every instance (G, L) of LHOM(H) given along with tree decomposition of G of width t can be solved in time $i^*(H)^t \cdot |V(G)|^{\mathcal{O}(1)}$.*
2. *Even if H is fixed, for any $\varepsilon > 0$, there is no algorithm that solves every instance (G, L) of LHOM(H) in time $i^*(H)^t \cdot |V(G)|^{\mathcal{O}(1)}$, unless the SETH fails.*

Let us briefly discuss the definition of i^*. This invariant is related to incomparable sets and bipartite decompositions (see Section 3.2 to recall the definitions). First, for every connected bipartite graph H, we define $i(H)$ as the maximum size of a one-sided incomparable set of vertices of H. This definition is motivated by the fact that for an instance (G, L) of LHOM(H), we can assume that list of every vertex of G is one-sided and incomparable. Furthermore, we define $i^*(H)$ for every bipartite graph H to be maximum $i(H')$ over all induced subgraphs H' of H such that H is connected, undecomposable, and such that LHOM(H') is NP-hard – this in turn is motivated by the fact that if H admits a decomposition, then we can solve an instance (G, L) of LHOM(H) by solving some small number of instances (G', L') of LHOM(H'), where G' is induced subgraph of G and H' is a proper induced subgraph of H. Finally, we define $i^*(H)$ for non-bipartite H by setting $i^*(H) = i^*(H^*)$, which is connected to a correspondence between some special homomorphisms $G^* \to H^*$ and $G \to H$.

Let us point out that techniques developed in [121] are quite powerful and the gadgets introduced in [121] are used in this dissertation as basic building blocks in hardness reductions in Section 4.4.

Precoloring extension and forbidden induced subgraphs. In [46] we study the complexity of the graph homomorphism problem for another type of restriction on the class of the input graphs: instead of bounding some parameter of G, we forbid some fixed graph F as an induced subgraph. If G does not contain F as an induced subgraph, we say that G is *F-free*. Let us point out that if F' is an induced subgraph of F, then every F'-free graph is also F-free. Therefore, every problem that is polynomial-time solvable for F-free graphs is also polynomial-time solvable for F'-free graphs, and every problem that is NP-hard for F'-free graphs is also NP-hard for F-free graphs.

In the problem we considered in [46], called H-ColoringExt, we are given a graph G with some vertices already precolored, and we have to determine if there is a homomorphism from G to H that extends this partial mapping. Note that H-ColoringExt lies in between of Hom(H) and LHom(H), i.e., we can think of H-ColoringExt as LHom(H) where every vertex of the input graph has either full list (equal to $V(H)$) or a one-element list.

For $a, b, c \in \mathbb{N}$, by $S_{a,b,c}$ we denote the graph obtained from three paths P_{a+1}, P_{b+1}, P_{c+1} by identifying their first vertices into one. For $k \in \mathbb{N}$, by W_k we denote the *k-wheel*, i.e., a graph obtained from C_k by adding a new vertex and connecting it to every vertex of C_k.

Let F be a connected graph. Piecyk and Rzążewski proved that for H such that LHom(H) is NP-hard, LHom(H) is NP-hard and cannot be solved in subexponential time on F-free graphs if F is not a path nor $S_{a,b,c}$ for some $a, b, c \in \mathbb{N}$ [128]. Furthermore, the complexity dichotomy of LHom(H) was provided in case of P_t-free graphs [122]. Therefore, it is interesting to consider LHom(H) on $S_{a,b,c}$-graphs as an open case.

The main result of [46] is a polynomial-time algorithm for W_5-ColoringExt for $S_{2,1,1}$-free graphs. This is complemented with the proof that W_5-ColoringExt is NP-hard in $S_{3,3,3}$-free graphs. This shows a very unusal behavior of the complexity of the problem. It is known that 3-Coloring is NP-hard on claw-free graphs, and so is the more general 3-ColoringExt problem (K_3-ColoringExt). On the other hand, K_3 is an induced subgraph of W_5, so this shows that the complexity of H-ColoringExt is not monotone with respect to taking induced subgraphs. In contrast, this is not the case of LHom(H), as for an induced subgraph H' of H, every instance (G, L) of LHom(H') can be seen as an instance of LHom(H) where no vertex from $V(H) \setminus V(H')$ appears on a list. Thus if LHom(H) is polynomial-time solvable on some class of graphs $\mathcal{C}$, then LHom(H') is polynomial-time solvable on $\mathcal{C}$, and if LHom(H') is NP-hard on $\mathcal{C}$, then so is LHom(H'). Finally, usually problems that are hard on $S_{a,b,c}$-free graphs for some $a, b, c \in \mathbb{N}$ are already hard for claw-free ($S_{1,1,1}$-free) graphs, which is not the case for W_5-ColoringExt.

Graph reconstruction. The problem considered in [90] is not related to graph homomorphisms. The general concept is that instead of a graph, we are given the information which subsets of vertices induce connected subgraphs and which do not. The question is

whether such information can uniquely describe the graph. Now let us define the problem formally. Let k be a fixed integer. In the k-RECONSTRUCTION problem, we are given a triple $(V, \mathcal{S}_k, \overline{\mathcal{S}}_k)$, where $\mathcal{S}_k, \overline{\mathcal{S}}_k \subseteq \binom{V}{k}$ and $\mathcal{S}_k, \overline{\mathcal{S}}_k$ form a partition of $\binom{V}{k}$, i.e., the family of k-element subsets of V. The task is to determine if there exists a graph G *consistent* with $(\mathcal{S}_k, \overline{\mathcal{S}}_k)$, i.e., $V(G) = V$, for every $S \in \mathcal{S}_k$, the graph $G[S]$ is connected, and for every $S' \in \overline{\mathcal{S}}_k$, the graph $G[S']$ is disconnected. Another possible goals are: (i) determine whether such a graph is unique, and (ii) enumerate all such graphs. The problem was introduced in [4] for special case of $k = 3$.

In [90], for every $k \geqslant 4$, we show the following.

1. Given $(V, \mathcal{S}_k, \overline{\mathcal{S}}_k)$, in time polynomial in $|V|$, we can determine whether there exists a triangle-free connected graph G consistent with $(V, \mathcal{S}_k, \overline{\mathcal{S}}_k)$, and if the answer is positive, then we can actually find G. Moreover, for some function f, if $|V| \geqslant f(k)$, there is at most one connected triangle-free graph G consistent with $(V, \mathcal{S}_k, \overline{\mathcal{S}}_k)$. Therefore, for every given $(V, \mathcal{S}_k, \overline{\mathcal{S}}_k)$, in time polynomial in $|V|$ we can enumerate all consistent connected triangle-free graphs – if $|V| < f(k)$, then we can brute-force, and if $|V| \geqslant f(k)$, then, if we find G, we know that it is unique.

2. Let $d \in \mathbb{N}$. Given $(V, \mathcal{S}_k, \overline{\mathcal{S}}_k)$ in time polynomial in $|V|$, we can determine whether there exists a connected graph G with $\Delta(G) \leqslant d$ and consistent with $(V, \mathcal{S}_k, \overline{\mathcal{S}}_k)$. Moreover, our algorithm "enumerates" all such graphs G. Let us explain the expression "enumerate" here. In this case we can have exponentially many consistent graphs, so we cannot have a polynomial-time algorithm that actually enumerates all of them. Instead, we describe all consistent graphs in a compact way i.e., our algorithm returns a set:

   $$\mathcal{G} = \{(G', V_1, \ldots, V_\ell, \mathcal{C}_1, \ldots, \mathcal{C}_\ell) \mid V_i \subseteq V(G), |V_i| \leqslant d, \mathcal{C}_i \text{ is a set of graphs on } V_i\},$$

   where the connected graphs G with $\Delta(G) \leqslant d$ consistent with $(V, \mathcal{S}_k, \overline{\mathcal{S}}_k)$ are precisely the graphs G that for some tuple $(G', V_1, \ldots, V_\ell, \mathcal{C}_1, \ldots, \mathcal{C}_\ell) \in \mathcal{G}$, either $G = G'$ or G can be obtained from G' by replacing $G'[V_i]$ with some $G_i \in \mathcal{C}_i$ for $i \in [\ell]$.

Bibliography

[1] Noga Alon, Igor Balla, Lior Gishboliner, Adva Mond, and Frank Mousset. The minrank of random graphs over arbitrary fields. *Israel Journal of Mathematics*, 235:63–77, 2020.

[2] Noga Alon and Eyal Lubetzky. The shannon capacity of a graph and the independence numbers of its powers. *IEEE Trans. Inf. Theory*, 52(5):2172–2176, 2006.

[3] Srinivasan Arunachalam, Péter Vrana, and Jeroen Zuiddam. The asymptotic induced matching number of hypergraphs: Balanced binary strings. *The Electronic Journal of Combinatorics*, 27(3), 2020.

[4] Paul Bastide, Linda Cook, Jeff Erickson, Carla Groenland, Marc J. van Kreveld, Isja Mannens, and Jordi L. Vermeulen. Reconstructing graphs from connected triples. In Daniël Paulusma and Bernard Ries, editors, *Graph-Theoretic Concepts in Computer Science - 49th International Workshop, WG 2023, Fribourg, Switzerland, June 28-30, 2023, Revised Selected Papers*, volume 14093 of *Lecture Notes in Computer Science*, pages 16–29. Springer, 2023.

[5] Laurent Beaudou, Florent Foucaud, and Reza Naserasr. Smallest c_{2l+1}-critical graphs of odd-girth 2k+1. *Discret. Appl. Math.*, 319:564–575, 2022.

[6] Sujoy Bhore, Paz Carmi, Sudeshna Kolay, and Meirav Zehavi. Parameterized study of steiner tree on unit disk graphs. *Algorithmica*, 85(1):133–152, 2023.

[7] Csaba Biró, Édouard Bonnet, Dániel Marx, Tillmann Miltzow, and Paweł Rzążewski. Fine-grained complexity of coloring unit disks and balls. *J. Comput. Geom.*, 9(2):47–80, 2018.

[8] Andreas Björklund and Thore Husfeldt. Inclusion-exclusion based algorithms for graph colouring. *Electron. Colloquium Comput. Complex.*, TR06-044, 2006.

[9] Hans L. Bodlaender, Marek Cygan, Stefan Kratsch, and Jesper Nederlof. Deterministic single exponential time algorithms for connectivity problems parameterized by treewidth. *Inf. Comput.*, 243:86–111, 2015.

[10] Hans L. Bodlaender, Erik Jan van Leeuwen, Johan M. M. van Rooij, and Martin Vatshelle. Faster algorithms on branch and clique decompositions. In Petr Hlinený and Antonín Kucera, editors, *Mathematical Foundations of Computer Science 2010, 35th International Symposium, MFCS 2010, Brno, Czech Republic, August 23-27, 2010. Proceedings*, volume 6281 of *Lecture Notes in Computer Science*, pages 174–185. Springer, 2010.

[11] Narek Bojikian, Vera Chekan, Falko Hegerfeld, and Stefan Kratsch. Tight bounds for connectivity problems parameterized by cutwidth. In Petra Berenbrink, Patricia Bouyer, Anuj Dawar, and Mamadou Moustapha Kanté, editors, *40th International Symposium on Theoretical Aspects of Computer Science, STACS 2023, March 7-9, 2023, Hamburg, Germany*, volume 254 of *LIPIcs*, pages 14:1–14:16. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2023.

[12] Marthe Bonamy, Konrad K. Dabrowski, Carl Feghali, Matthew Johnson, and Daniël Paulusma. Independent feedback vertex sets for graphs of bounded diameter. *Inf. Process. Lett.*, 131:26–32, 2018.

[13] 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.

[14] Édouard Bonnet, Eun Jung Kim, Amadeus Reinald, Stéphan Thomassé, and Rémi Watrigant. Twin-width and polynomial kernels. *Algorithmica*, 84(11):3300–3337, 2022.

[15] Édouard Bonnet, Eun Jung Kim, Stéphan Thomassé, and Rémi Watrigant. Twin-width I: tractable FO model checking. *J. ACM*, 69(1):3:1–3:46, 2022.

[16] Christoph Brause, Petr A. Golovach, Barnaby Martin, Pascal Ochem, Daniël Paulusma, and Siani Smith. Acyclic, star, and injective colouring: Bounding the diameter. *Electron. J. Comb.*, 29(2), 2022.

[17] Csilla Bujtás, Akbar Davoodi, Ervin Győri, and Zsolt Tuza. Clique coverings and claw-free graphs. *European Journal of Combinatorics*, 88:103114, 2020. Selected papers of EuroComb17.

[18] Andrei A. Bulatov and Amirhossein Kazeminia. Complexity classification of counting graph homomorphisms modulo a prime number. In Stefano Leonardi and Anupam Gupta, editors, *STOC 2022*, pages 1024–1037. ACM, 2022.

[19] Jannis Bulian and Anuj Dawar. Graph isomorphism parameterized by elimination distance to bounded degree. *CoRR*, abs/1406.4718, 2014.

[20] Jannis Bulian and Anuj Dawar. Fixed-parameter tractable distances to sparse graph classes. *Algorithmica*, 79(1):139–158, 2017.

[21] Jin-Yi Cai and Ashwin Maran. The complexity of counting planar graph homomorphisms of domain size 3. In Barna Saha and Rocco A. Servedio, editors, *STOC 2023*, pages 1285–1297. ACM, 2023.

[22] Leizhen Cai. Parameterized complexity of vertex colouring. *Discret. Appl. Math.*, 127(3):415–429, 2003.

[23] Victor A. Campos, Guilherme de C. M. Gomes, Allen Ibiapina, Raul Lopes, Ignasi Sau, and Ana Silva. Coloring problems on bipartite graphs of small diameter. *Electron. J. Comb.*, 28(2):2, 2021.

[24] Airlie Chapman and Mehran Mesbahi. On strong structural controllability of networked systems: A constrained matching approach. In *2013 American Control Conference*, pages 6126–6131, 2013.

[25] Prasad Chaugule, Nutan Limaye, and Aditya Varre. Variants of homomorphism polynomials complete for algebraic complexity classes. *ACM Trans. Comput. Theory*, 13(4):21:1–21:26, 2021.

[26] Rajesh Chitnis, László Egri, and Dániel Marx. List h-coloring a graph by removing few vertices. *Algorithmica*, 78(1):110–146, 2017.

[27] Maria Chudnovsky, Jan Goedgebeur, Oliver Schaudt, and Mingxian Zhong. Obstructions for three-coloring and list three-coloring H-free graphs. *SIAM J. Discret. Math.*, 34(1):431–469, 2020.

[28] Maria Chudnovsky, Shenwei Huang, Pawel Rzazewski, Sophie Spirkl, and Mingxian Zhong. Complexity of C_k-coloring in hereditary classes of graphs. *Inf. Comput.*, 292:105015, 2023.

[29] Maria Chudnovsky, Marcin Pilipczuk, Michal Pilipczuk, and Stéphan Thomassé. Quasi-polynomial time approximation schemes for the maximum weight independent set problem in \(\boldsymbol{H}\)-free graphs. *SIAM J. Comput.*, 53(1):47–86, 2024.

[30] Maria Chudnovsky, Sophie Spirkl, and Mingxian Zhong. List 3-coloring P_t-free graphs with no induced 1-subdivision of $K_{1,s}$. *Discrete Mathematics*, 343(11):112086, 2020.

[31] Maria Chudnovsky, Sophie Spirkl, and Mingxian Zhong. Four-coloring \(\boldsymbol{P_6}\)-free graphs. II. finding an excellent precoloring. *SIAM J. Comput.*, 53(1):146–187, 2024.

[32] Maria Chudnovsky, Sophie Spirkl, and Mingxian Zhong. Four-coloring \(p_6\)-free graphs. i. extending an excellent precoloring. *SIAM J. Comput.*, 53(1):111–145, 2024.

[33] Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, and Clifford Stein. *Introduction to Algorithms, 3rd Edition*. MIT Press, 2009.

[34] Jean-François Couturier, Petr A. Golovach, Dieter Kratsch, and Daniël Paulusma. On the parameterized complexity of coloring graphs in the absence of a linear forest. *J. Discrete Algorithms*, 15:56–62, 2012.

[35] Radu Curticapean, Holger Dell, and Dániel Marx. Homomorphisms are a good basis for counting small subgraphs. In Hamed Hatami, Pierre McKenzie, and Valerie King, editors, *STOC 2017*, pages 210–223. ACM, 2017.

[36] Radu Curticapean and Dániel Marx. Tight conditional lower bounds for counting perfect matchings on graphs of bounded treewidth, cliquewidth, and genus. In Robert Krauthgamer, editor, *SODA 2016*, pages 1650–1669. SIAM, 2016.

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

[38] Marek Cygan, Stefan Kratsch, and Jesper Nederlof. Fast hamiltonicity checking via bases of perfect matchings. *J. ACM*, 65(3):12:1–12:46, 2018.

[39] Clément Dallard, Fedor V. Fomin, Petr A. Golovach, Tuukka Korhonen, and Martin Milanic. Computing tree decompositions with small independence number. In Karl Bringmann, Martin Grohe, Gabriele Puppis, and Ola Svensson, editors, *51st International Colloquium on Automata, Languages, and Programming, ICALP 2024, July 8-12, 2024, Tallinn, Estonia*, volume 297 of *LIPIcs*, pages 51:1–51:18. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2024.

[40] Clément Dallard, Martin Milanic, and Kenny Storgel. Treewidth versus clique number. II. tree-independence number. *J. Comb. Theory B*, 164:404–442, 2024.

[41] Ronald de Wolf. Nondeterministic quantum query and communication complexities. *SIAM Journal on Computing*, 32(3):681–699, 2003.

[42] Matt DeVos, O-joung Kwon, and Sang-il Oum. Branch-depth: Generalizing tree-depth of graphs. *Eur. J. Comb.*, 90:103186, 2020.

[43] Jinquan Dong and Yanpei Liu. On the decomposition of graphs into complete bipartite graphs. *Graphs Comb.*, 23(3):255–262, 2007.

[44] Louis Dublois, Michael Lampis, and Vangelis Th. Paschos. New algorithms for mixed dominating set. *Discret. Math. Theor. Comput. Sci.*, 23(1), 2021.

[45] Louis Dublois, Michael Lampis, and Vangelis Th. Paschos. Upper dominating set: Tight algorithms for pathwidth and sub-exponential approximation. In Tiziana Calamoneri and Federico Corò, editors, *Algorithms and Complexity - 12th International Conference, CIAC 2021, Virtual Event, May 10-12, 2021, Proceedings*, volume 12701 of *Lecture Notes in Computer Science*, pages 202–215. Springer, 2021.

[46] 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.

[47] Michał Dębski, Marta Piecyk, and Paweł Rzążewski. Faster 3-coloring of small-diameter graphs. In Petra Mutzel, Rasmus Pagh, and Grzegorz Herman, editors, *29th Annual European Symposium on Algorithms, ESA 2021, September 6-8, 2021, Lisbon, Portugal (Virtual Conference)*, volume 204 of *LIPIcs*, pages 37:1–37:15. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2021.

[48] Michał Dębski, Marta Piecyk, and Paweł Rzążewski. Faster 3-coloring of small-diameter graphs. *SIAM J. Discret. Math.*, 36(3):2205–2224, 2022.

[49] Oliver Ebsen and Mathias Schacht. Homomorphism thresholds for odd cycles. *Comb.*, 40(1):39–62, 2020.

[50] Keith Edwards. The complexity of colouring problems on dense graphs. *Theor. Comput. Sci.*, 43:337–343, 1986.

[51] László Egri, Andrei A. Krokhin, Benoît Larose, and Pascal Tesson. The complexity of the list homomorphism problem for graphs. In Jean-Yves Marion and Thomas Schwentick, editors, *STACS 2010*, volume 5 of *LIPIcs*, pages 335–346. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2010.

[52] 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, *STACS 2018*, volume 96 of *LIPIcs*, pages 27:1–27:15. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2018.

[53] Eduard Eiben, Robert Ganian, Thekla Hamm, and O-joung Kwon. Measuring what matters: A hybrid approach to dynamic programming with treewidth. *J. Comput. Syst. Sci.*, 121:57–75, 2021.

[54] Paul Erdős and George Szekeres. A combinatorial problem in geometry. *Compositio mathematica*, 2:463–470, 1935.

[55] Baris Can Esmer, Jacob Focke, Dániel Marx, and Pawel Rzazewski. List homomorphisms by deleting edges and vertices: Tight complexity bounds for bounded-treewidth graphs. In Timothy Chan, Johannes Fischer, John Iacono, and Grzegorz Herman, editors, *32nd Annual European Symposium on Algorithms, ESA 2024, September 2-4, 2024, Royal Holloway, London, United Kingdom*, volume 308 of *LIPIcs*, pages 39:1–39:20. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2024.

[56] Tomás Feder and Pavol Hell. List homomorphisms to reflexive graphs. *J. Comb. Theory B*, 72(2):236–250, 1998.

[57] Tomás Feder, Pavol Hell, and Jing Huang. List homomorphisms and circular arc graphs. *Comb.*, 19(4):487–505, 1999.

[58] Tomás Feder, Pavol Hell, and Jing Huang. Bi-arc graphs and the complexity of list homomorphisms. *J. Graph Theory*, 42(1):61–80, 2003.

[59] M. Fekete. Über die Verteilung der Wurzeln bei gewissen algebraischen Gleichungen mit ganzzahligen Koeffizienten. *Mathematische Zeitschrift*, 17(1):228–249, December 1923.

[60] Peter C. Fishburn and Peter L. Hammer. Bipartite dimensions and bipartite degrees of graphs. *Discrete Mathematics*, 160(1):127–148, 1996.

[61] Jacob Focke, Dániel Marx, Fionn Mc Inerney, Daniel Neuen, Govind S. Sankar, Philipp Schepper, and Philip Wellnitz. Tight complexity bounds for counting generalized dominating sets in bounded-treewidth graphs. In Nikhil Bansal and Viswanath Nagarajan, editors, *SODA 2023*, pages 3664–3683. SIAM, 2023.

[62] Jacob Focke, Dániel Marx, and Paweł Rzążewski. Counting list homomorphisms from graphs of bounded treewidth: tight complexity bounds. In Joseph (Seffi) Naor and Niv Buchbinder, editors, *Proceedings of the 2022 ACM-SIAM Symposium on Discrete Algorithms, SODA 2022, Virtual Conference / Alexandria, VA, USA, January 9 - 12, 2022*, pages 431–458. SIAM, 2022.

[63] Fedor V. Fomin, Daniel Lokshtanov, Fahad Panolan, and Saket Saurabh. Efficient computation of representative families with applications in parameterized and exact algorithms. *J. ACM*, 63(4):29:1–29:60, 2016.

[64] Jakub Gajarský, Michael Lampis, and Sebastian Ordyniak. Parameterized algorithms for modular-width. In Gregory Z. Gutin and Stefan Szeider, editors, *Parameterized and Exact Computation - 8th International Symposium, IPEC 2013, Sophia Antipolis, France, September 4-6, 2013, Revised Selected Papers*, volume 8246 of *Lecture Notes in Computer Science*, pages 163–176. Springer, 2013.

[65] 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.

[66] Robert Ganian, Petr Hlinený, Jaroslav Nesetril, Jan Obdrzálek, and Patrice Ossona de Mendez. Shrub-depth: Capturing height of dense graphs. *CoRR*, abs/1707.00359, 2017.

[67] A. M. H. Gerards. Homomorphisms of graphs into odd cycles. *J. Graph Theory*, 12(1):73–83, 1988.

[68] Chris Godsil. Problems in algebraic combinatorics. *Electr. J. Comb.*, 2, 01 1995.

[69] Alexander Golovnev, Oded Regev, and Omri Weinstein. The minrank of random graphs. *IEEE Transactions on Information Theory*, 64(11):6990–6995, 2018.

[70] Carla Groenland, Isja Mannens, Jesper Nederlof, Marta Piecyk, and Paweł Rzążewski. Towards tight bounds for the graph homomorphism problem parameterized by cutwidth via asymptotic matrix parameters. In Karl Bringmann, Martin Grohe, Gabriele Puppis, and Ola Svensson, editors, *51st International Colloquium on Automata, Languages, and Programming, ICALP 2024, July 8-12, 2024, Tallinn, Estonia*, volume 297 of *LIPIcs*, pages 77:1–77:21. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2024.

[71] Carla Groenland, Isja Mannens, Jesper Nederlof, and Krisztina Szilágyi. Tight bounds for counting colorings and connected edge sets parameterized by cutwidth. In Petra Berenbrink and Benjamin Monmege, editors, *STACS 2022*, volume 219 of *LIPIcs*, pages 36:1–36:20. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2022.

[72] Martin Grohe and Daniel Neuen. Isomorphism for tournaments of small twin width. In Karl Bringmann, Martin Grohe, Gabriele Puppis, and Ola Svensson, editors, *51st International Colloquium on Automata, Languages, and Programming, ICALP 2024, July 8-12, 2024, Tallinn, Estonia*, volume 297 of *LIPIcs*, pages 78:1–78:20. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2024.

[73] Gregory Z. Gutin, Diptapriyo Majumdar, Sebastian Ordyniak, and Magnus Wahlström. Parameterized pre-coloring extension and list coloring problems. *SIAM J. Discret. Math.*, 35(1):575–596, 2021.

[74] Richard H. Hammack, Wilfried Imrich, and Sandi Klavžar. *Handbook of product graphs*. CRC press, 2011.

[75] Elfarouk Harb, Zhengcheng Huang, and Da Wei Zheng. Shortest path separators in unit disk graphs. In Timothy Chan, Johannes Fischer, John Iacono, and Grzegorz Herman, editors, *32nd Annual European Symposium on Algorithms, ESA 2024, September 2-4, 2024, Royal Holloway, London, United Kingdom*, volume 308 of *LIPIcs*, pages 66:1–66:14. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2024.

[76] Ishay Haviv. On minrank and forbidden subgraphs. *ACM Transactions on Computation Theory*, 11(4), 2019.

[77] Falko Hegerfeld and Stefan Kratsch. Solving connectivity problems parameterized by treedepth in single-exponential time and polynomial space. In Christophe Paul and Markus Bläser, editors, *37th International Symposium on Theoretical Aspects of Computer Science, STACS 2020, March 10-13, 2020, Montpellier, France*, volume 154 of *LIPIcs*, pages 29:1–29:16. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2020.

[78] Pavol Hell and Jaroslav Nesetril. On the complexity of H-coloring. *J. Comb. Theory B*, 48(1):92–110, 1990.

[79] Pavol Hell and Jaroslav Nešetřil. The core of a graph. *Discrete Mathematics*, 109(1-3):117–126, 1992.

[80] Eva-Maria C. Hols, Stefan Kratsch, and Astrid Pieterse. Elimination distances, blocking sets, and kernels for vertex cover. In Christophe Paul and Markus Bläser, editors, *37th International Symposium on Theoretical Aspects of Computer Science, STACS 2020, March 10-13, 2020, Montpellier, France*, volume 154 of *LIPIcs*, pages 36:1–36:14. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2020.

[81] Russell Impagliazzo and Ramamohan Paturi. On the complexity of k-SAT. *J. Comput. Syst. Sci.*, 62(2):367 – 375, 2001.

[82] Russell Impagliazzo, Ramamohan Paturi, and Francis Zane. Which problems have strongly exponential complexity? *J. Comput. Syst. Sci.*, 63(4):512–530, 2001.

[83] Lars Jaffke and Bart M. P. Jansen. Fine-grained parameterized complexity analysis of graph coloring problems. In Dimitris Fotakis, Aris Pagourtzis, and Vangelis Th. Paschos, editors, *Algorithms and Complexity - 10th International Conference, CIAC 2017, Athens, Greece, May 24-26, 2017, Proceedings*, volume 10236 of *Lecture Notes in Computer Science*, pages 345–356, 2017.

[84] Lars Jaffke and Bart M. P. Jansen. Fine-grained parameterized complexity analysis of graph coloring problems. *Discret. Appl. Math.*, 327:33–46, 2023.

[85] Bart M. P. Jansen. Personal communication.

[86] Bart M. P. Jansen, Jari J. H. de Kroon, and Michał Włodarczyk. Vertex deletion parameterized by elimination distance and even less. In Samir Khuller and Virginia Vassilevska Williams, editors, *STOC '21: 53rd Annual ACM SIGACT Symposium on Theory of Computing, Virtual Event, Italy, June 21-25, 2021*, pages 1757–1769. ACM, 2021.

[87] Bart M. P. Jansen and Jesper Nederlof. Computing the chromatic number using graph decompositions via matrix rank. *Theor. Comput. Sci.*, 795:520–539, 2019.

[88] Ioannis Katsikarelis, Michael Lampis, and Vangelis Th. Paschos. Structurally parameterized d-scattered set. *Discret. Appl. Math.*, 308:168–186, 2022.

[89] Tereza Klimosová and Vibha Sahlot. 3-coloring C_4 or C_3-free diameter two graphs. In Pat Morin and Subhash Suri, editors, *Algorithms and Data Structures - 18th International Symposium, WADS 2023, Montreal, QC, Canada, July 31 - August 2, 2023, Proceedings*, volume 14079 of *Lecture Notes in Computer Science*, pages 547–560. Springer, 2023.

[90] Kacper Kluk, Hoang La, and Marta Piecyk. Graph reconstruction with connectivity queries, 2024.

[91] Kolja Knauer and Torsten Ueckerdt. Three ways to cover a graph. *Discrete Mathematics*, 339(2):745–758, 2016.

[92] Mikko Koivisto. An o*(2ˆn) algorithm for graph coloring and other partitioning problems via inclusion–exclusion. In *47th Annual IEEE Symposium on Foundations of Computer Science (FOCS 2006), 21-24 October 2006, Berkeley, California, USA, Proceedings*, pages 583–590. IEEE Computer Society, 2006.

[93] Aleksej Dmitrievich Korshunov. On the diameter of graphs. *Soviet Math*, 12:302:305, 1971.

[94] Stefan Kratsch and Magnus Wahlström. Representative sets and irrelevant vertices: New tools for kernelization. *J. ACM*, 67(3):16:1–16:50, 2020.

[95] M. R. Krom. The decision problem for a class of first-order formulas in which all disjunctions are binary. *Mathematical Logic Quarterly*, 13(1-2):15–20, 1967.

[96] Hong-Jian Lai. Unique graph homomorphisms onto odd cycles, II. *J. Comb. Theory, Ser. B*, 46(3):363–376, 1989.

[97] Michael Lampis. Finer tight bounds for coloring on clique-width. *SIAM J. Discret. Math.*, 34(3):1538–1558, 2020.

[98] Benoît Larose. Families of strongly projective graphs. *Discuss. Math. Graph Theory*, 22(2):271–292, 2002.

[99] Benoit Larose. Strongly projective graphs. *Canadian Journal of Mathematics*, 54(4):757–768, 2002.

[100] Benoit Larose and Claude Tardif. Strongly rigid graphs and projectivity. *Multiple-Valued Logic*, 7:339–361, 2001.

[101] 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.

[102] Daniel Lokshtanov, Fahad Panolan, Saket Saurabh, Jie Xue, and Meirav Zehavi. A 1.9999-approximation algorithm for vertex cover on string graphs. In Wolfgang Mulzer and Jeff M. Phillips, editors, *40th International Symposium on Computational Geometry, SoCG 2024, June 11-14, 2024, Athens, Greece*, volume 293 of *LIPIcs*, pages 72:1–72:11. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2024.

[103] L. Lovasz. On the shannon capacity of a graph. *IEEE Transactions on Information Theory*, 25(1):1–7, 1979.

[104] Tomasz Łuczak and Jaroslav Nešetřil. Note on projective graphs. *Journal of Graph Theory*, 47(2):81–86, 2004.

[105] Barnaby Martin, Daniël Paulusma, and Siani Smith. Colouring H-free graphs of bounded diameter. In Peter Rossmanith, Pinar Heggernes, and Joost-Pieter Katoen, editors, *44th International Symposium on Mathematical Foundations of Computer Science, MFCS 2019, August 26-30, 2019, Aachen, Germany*, volume 138 of *LIPIcs*, pages 14:1–14:14. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2019.

[106] Barnaby Martin, Daniël Paulusma, and Siani Smith. Colouring graphs of bounded diameter in the absence of small cycles. *Discret. Appl. Math.*, 314:150–161, 2022.

[107] Dániel Marx, Govind S. Sankar, and Philipp Schepper. Degrees and gaps: Tight complexity results of general factor problems parameterized by treewidth and cutwidth. In Nikhil Bansal, Emanuela Merelli, and James Worrell, editors, *ICALP 2021*, volume 198 of *LIPIcs*, pages 95:1–95:20. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2021.

[108] Dániel Marx, Govind S. Sankar, and Philipp Schepper. Anti-Factor Is FPT Parameterized by Treewidth and List Size (But Counting Is Hard). In Holger Dell and

Jesper Nederlof, editors, *IPEC 2022*, volume 249 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 22:1–22:23, Dagstuhl, Germany, 2022. Schloss Dagstuhl – Leibniz-Zentrum für Informatik.

[109] Colin McDiarmid. Concentration. In J. Ramirez-Alfonsin M. Habib, C. McDiarmid and B. Reed, editors, *Probabilistic methods for algorithmic discrete mathematics*, volume 16 of *Algorithms and Combinatorics*, pages 195–248. Springer, 1998.

[110] Ralph McKenzie. Cardinal multiplication of structures with a reflexive relation. *Fundamenta Mathematicae*, 70(1):59–101, 1971.

[111] George B. Mertzios and Paul G. Spirakis. Algorithms and almost tight results for 3-colorability of small diameter graphs. *Algorithmica*, 74(1):385–414, 2016.

[112] Burkhard Monien. The complexity of determining paths of length k. In Manfred Nagl and Jürgen Perl, editors, *WG '83*, pages 241–251. Universitätsverlag Rudolf Trauner, Linz, 1983.

[113] Jan Mycielski. Sur le coloriage des graphs. *Colloquium Mathematicae*, 3(2):161–162, 1955.

[114] Jesper Nederlof. Algorithms for np-hard problems via rank-related parameters of matrices. In Fedor V. Fomin, Stefan Kratsch, and Erik Jan van Leeuwen, editors, *Treewidth, Kernels, and Algorithms - Essays Dedicated to Hans L. Bodlaender on the Occasion of His 60th Birthday*, volume 12160 of *Lecture Notes in Computer Science*, pages 145–164. Springer, 2020.

[115] Jesper Nederlof. Bipartite TSP in $O(1.9999^n)$ time, assuming quadratic time matrix multiplication. In Konstantin Makarychev, Yury Makarychev, Madhur Tulsiani, Gautam Kamath, and Julia Chuzhoy, editors, *STOC 2020*, pages 40–53. ACM, 2020.

[116] Jesper Nederlof, Michał Pilipczuk, Céline M. F. Swennenhuis, and Karol Węgrzycki. Hamiltonian cycle parameterized by treedepth in single exponential time and polynomial space. In Isolde Adler and Haiko Müller, editors, *Graph-Theoretic Concepts in Computer Science - 46th International Workshop, WG 2020, Leeds, UK, June*

24-26, 2020, Revised Selected Papers, volume 12301 of *Lecture Notes in Computer Science*, pages 27–39. Springer, 2020.

[117] Jaroslav Nesetril and Patrice Ossona de Mendez. *Sparsity - Graphs, Structures, and Algorithms*, volume 28 of *Algorithms and combinatorics*. Springer, 2012.

[118] Jaroslav Nešetřil and Aleš Pultr. A Dushnik - Miller type dimension of graphs and its complexity. In Marek Karpiński, editor, *Fundamentals of Computation Theory*, pages 482–493, Berlin, Heidelberg, 1977. Springer Berlin Heidelberg.

[119] Jaroslav Nešetřil and Vojtéch Rödl. A simple proof of the Galvin-Ramsey property of the class of all finite graphs and a dimension of a graph. *Discrete Mathematics*, 23(1):49–55, 1978.

[120] Karolina Okrasa. Graph homomorphisms: From structure to algorithms. *PhD thesis, Warsaw University of Technology*, 2024.

[121] 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.

[122] Karolina Okrasa and Pawel 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.

[123] Karolina Okrasa and Paweł Rzążewski. Subexponential algorithms for variants of the homomorphism problem in string graphs. *J. Comput. Syst. Sci.*, 109:126–144, 2020.

[124] Karolina Okrasa and Paweł Rzążewski. Fine-grained complexity of the graph homo-

morphism problem for bounded-treewidth graphs. *SIAM J. Comput.*, 50(2):487–508, 2021.

[125] D.D. Olesky, Michael Tsatsomeros, and P. van den Driessche. Qualitative controllability and uncontrollability by a single entry. *Linear Algebra and its Applications*, 187:183–194, 1993.

[126] Christos H. Papadimitriou. *Computational complexity*. Addison-Wesley, 1994.

[127] Marta Piecyk. C_{2k+1}-coloring of bounded-diameter graphs. In Rastislav Královic and Antonín Kucera, editors, *49th International Symposium on Mathematical Foundations of Computer Science, MFCS 2024, August 26-30, 2024, Bratislava, Slovakia*, volume 306 of *LIPIcs*, pages 78:1–78:15. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2024.

[128] 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.

[129] Stefan Porschen. On variable-weighted exact satisfiability problems. *Ann. Math. Artif. Intell.*, 51(1):27–54, 2007.

[130] Marc Roth and Philip Wellnitz. Counting and finding homomorphisms is universal for parameterized complexity theory. In Shuchi Chawla, editor, *SODA 2020*, pages 2161–2180. SIAM, 2020.

[131] Sebastian Schnettler. A structured overview of 50 years of small-world research. *Soc. Networks*, 31(3):165–178, 2009.

[132] Claude E. Shannon. *The Zero Error Capacity of a Noisy Channel*, pages 221–238. 1993.

[133] Petra Sparl and Janez Zerovnik. Homomorphisms of hexagonal graphs to odd cycles. *Discret. Math.*, 283(1-3):273–277, 2004.

[134] Maguy Trefois and Jean-Charles Delvenne. Zero forcing number, constrained matchings and strong structural controllability. *Linear Algebra and its Applications*, 484:199–218, 2015.

[135] Virginia Vassilevska Williams, Yinzhan Xu, Zixuan Xu, and Renfei Zhou. New bounds for matrix multiplication: from alpha to omega. In David P. Woodruff, editor, *Proceedings of the 2024 ACM-SIAM Symposium on Discrete Algorithms, SODA 2024, Alexandria, VA, USA, January 7-10, 2024*, pages 3792–3835. SIAM, 2024.

Chapter 7

Appendix

7.1 Inequality from Lemma 4.30

In this section we prove inequality from the proof of Lemma 4.30. Recall that we want to upperbound the right-hand side of (4.5.1) with the expression from (4.5.2), i.e., we want to show that

$$\left(\frac{eh}{\ell}\right)^{2\ell}\left(\frac{1}{2}\right)^{\ell^2} \leqslant \left(\frac{eh}{2\log_2 h}\right)^{4\log_2 h}\left(\frac{1}{2}\right)^{(2\log_2 h)^2},$$

when $\ell = \lceil 2\log_2 h \rceil$ and h is a sufficiently large positive integer.

We will actually prove that for sufficiently large h, for $x \geqslant y \geqslant 2\log_2 h$ and $t = eh$, we have:

$$\left(\frac{t}{x}\right)^{2x}\left(\frac{1}{2}\right)^{x^2} \leqslant \left(\frac{t}{y}\right)^{2y}\left(\frac{1}{2}\right)^{y^2},$$

which implies the desired inequality since $\ell \geqslant 2\log_2 h$.

In other words, we will show that the function

$$f(x) = \left(\frac{t}{x}\right)^{2x}\left(\frac{1}{2}\right)^{x^2}$$

is non-increasing for $x \geqslant 2\log_2 h$.

Equivalently, we can show that the function $g(x) = \log_2 f(x)$ is non-increasing for $x \geqslant 2\log_2 h$. In order to do that we will show that $g'(x) \leqslant 0$, for $x \geqslant 2\log_2 h$.

First, let us rewrite the function g.

$$g(x) = \log_2 f(x) = 2x\cdot(\log_2 t - \log_2 x) + x^2\cdot(\log_2 1 - \log_2 2) = 2x\cdot\log_2 t - 2x\cdot\log_2 x - x^2.$$

Now let us compute $g'(x)$.

$$g'(x) = \left(2x \cdot \log_2 t - 2x \cdot \log_2 x - x^2\right)' = 2\log_2 t - (2x \cdot \log_2 x)' - 2x$$
$$= 2\log_2 t - \left(2\log_2 x + \frac{2x}{x \cdot \log 2}\right) - 2x = 2\log_2 t - 2\log_2 x - \frac{1}{\log 2} - 2x$$
$$= 2\left(\log_2 t - x\right) + \left(-2\log_2 x - \frac{1}{\log 2}\right)$$

Since h is sufficiently large, we can safely assume that $h \geqslant e$, and $\log_2 t = \log_2 e + \log_2 h \leqslant 2\log_2 h \leqslant x$, so the first term is at most 0. Similarly, we can assume that h is sufficiently large, so that for $x \geqslant 2\log_2 h$, the second term is negative. Therefore $g'(x) \leqslant 0$ for $x \geqslant 2\log_2 h$ as desired.

7.2 Solving recursive inequalities

In this section we provide the detailed description on how we solve recursive inequalities that describe the complexity of our algorithms in Theorem 5.7 and Lemma 5.11. By $F(\mu)$ we denote the complexity of an algorithm on instances of measure μ, and we always assume that n is the number of vertices of the whole graph. We assume that F is non-decreasing. In what follows we also assume that μ is sufficiently large, as we can deal with the case of bounded μ by adjusting the constants accordingly. Finally, we note that we make no attempt to optimize the constants in the final complexity bound.

Solving the recursion in Theorem 5.7 and Lemma 5.20. In order to solve the recursions in Theorem 5.7 and Lemma 5.20, is enough to find, for any fixed constant $c \in \mathbb{N}$ and for any fixed polynomial $\mathsf{p}(\mu)$, an upper bound on $F(\mu)$, where F satisfies the inequality:

$$F(\mu) \leqslant F\left(\mu - \frac{(\mu \log \mu)^{1/d}}{c}\right) + F(\mu - 1) + \mathsf{p}(n).$$

We solve the recursion as follows.

$$
\begin{aligned}
F(\mu) &\leqslant F\left(\mu - \frac{(\mu \log \mu)^{1/d}}{c}\right) + F(\mu - 1) + \mathsf{p}(n) \\
&\leqslant 2F\left(\mu - \frac{(\mu \log \mu)^{1/d}}{c}\right) + F(\mu - 2) + 2\mathsf{p}(n) \leqslant \cdots \\
&\leqslant \mu F\left(\mu - \frac{(\mu \log \mu)^{1/d}}{c}\right) + \mu \cdot \mathsf{p}(n) = \mu^{\mathcal{O}\left(\frac{\mu}{(\mu \log \mu)^{1/d}}\right)} \\
&= 2^{\mathcal{O}\left((\mu \log \mu)^{1-1/d}\right)}.
\end{aligned}
$$

Solving the recursion in Lemma 5.11 Here $N_0, N_1, \ldots$ are appropriately chosen constants. Recall that we have:

$$
F(\mu) \leqslant \max \begin{cases} 3F(\mu - \mu^{2/3}) + 3\mathsf{p}(n), \\ F(\mu - 1) + 2F(\mu - \frac{1}{108}\mu^{2/3}) + 3\mathsf{p}(n), \\ F(\mu - 1) + 6F(\mu - \mu^{2/3}) + 7\mathsf{p}(n), \\ 2^{K'\mu^{1/3}\log^2 \mu} F\left(\frac{5}{6}\mu\right) + 2^{K'\mu^{1/3}\log^2 \mu}\mathsf{p}(n), \end{cases}
$$

where n is the number of vertices of the whole graph and p is some polynomial function. As μ is large, $\mu \leqslant n$ and we can assume that $\mathsf{p}(n) = o(F(n))$ (in fact, the function F is superpolynomial), we can write the following.

$$
F(\mu) \leqslant F(n) \leqslant \max\left(F(n-1) + 6F\left(n - \frac{n^{2/3}}{108}\right), 2^{N_0 n^{1/3}\log^2 n} F\left(\frac{5}{6}n\right)\right).
$$

Clearly we have

$$
F(n) \leqslant F(n-1) + 6F\left(n - \frac{n^{2/3}}{108}\right) + 2^{N_0 n^{1/3}\log^2 n} F\left(\frac{5}{6}n\right).
$$

We apply this inequality n times to the first expression, obtaining:

$$
\begin{aligned}
F(n) \leqslant & F(n-1) + 6F\left(n - \frac{n^{2/3}}{108}\right) + 2^{N_0 n^{1/3}\log^2 n} F\left(\frac{5}{6}n\right) \\
\leqslant & F(n-2) + 2 \cdot 6F\left(n - \frac{n^{2/3}}{108}\right) + 2 \cdot 2^{N_0 n^{1/3}\log^2 n} F\left(\frac{5}{6}n\right) \\
\leqslant & F(n-3) + 3 \cdot 6F\left(n - \frac{n^{2/3}}{108}\right) + 3 \cdot 2^{N_0 n^{1/3}\log^2 n} F\left(\frac{5}{6}n\right) \\
\leqslant \ldots \leqslant & N_1 + n \cdot 6F\left(n - \frac{n^{2/3}}{108}\right) + n \cdot 2^{N_0 n^{1/3}\log^2 n} F\left(\frac{5}{6}n\right).
\end{aligned}
$$

As n is large and F is non-decreasing, we can write

$$
F(n) \leqslant n \cdot 7F\left(n - \frac{n^{2/3}}{108}\right) + n \cdot 2^{N_0 n^{1/3}\log^2 n} F\left(\frac{5}{6}n\right).
$$

Now let us apply the above expression to its first term.

$$\begin{aligned}
F(n) \leqslant & n \cdot 7F\left(n - \frac{n^{2/3}}{108}\right) + n \cdot 2^{N_0 n^{1/3}\log^2 n} F\left(\frac{5}{6}n\right) \\
\leqslant & 7n\left(n \cdot 7F\left(n - \frac{n^{2/3}}{108} - \frac{1}{108}\left(n - \frac{n^{2/3}}{108}\right)^{2/3}\right) + n \cdot 2^{N_0 n^{1/3}\log^2 n} F\left(\frac{5}{6}n\right)\right) \\
& + n \cdot 2^{N_0 n^{1/3}\log^2 n} F\left(\frac{5}{6}n\right) \\
\leqslant & (7n)^2 F\left(n - \frac{n^{2/3}}{108} - \frac{1}{108}\left(n - \frac{n^{2/3}}{108}\right)^{2/3}\right) + 8n^2 \cdot 2^{N_0 n^{1/3}\log^2 n} F\left(\frac{5}{6}n\right) \\
\leqslant & (7n)^2 F\left(n - \frac{n^{2/3}}{108} - \frac{0.99n^{2/3}}{108}\right) + 8n^2 \cdot 2^{N_0 n^{1/3}\log^2 n} F\left(\frac{5}{6}n\right)
\end{aligned}$$

Let us repeat this $N_2 \cdot n^{1/3}$ times, until the argument in the first term drops to at most $\frac{5}{6}n$.

$$F(n) \leqslant \ldots \leqslant (7n)^{N_2 n^{1/3}} \cdot F\left(\frac{5}{6}n\right) + (8n)^{N_2 n^{1/3}} \cdot 2^{N_0 n^{1/3}\log^2 n} F\left(\frac{5}{6}n\right).$$

By choosing N_3 sufficiently large, we can thus write

$$F(n) \leqslant 2^{N_3 n^{1/3}\log^2 n} F\left(\frac{5}{6}n\right).$$

Now we solve the recursion as follows.

$$\begin{aligned}
F(n) \leqslant & \; 2^{N_3 n^{1/3}\log^2 n} F\left(\frac{5}{6}n\right) + 2^{N_3 n^{1/3}\log^2 n} \\
\leqslant & \; 2^{N_3 n^{1/3}\log^2 n \cdot (1+(5/6)^{1/3})} F\left(\left(\frac{5}{6}\right)^2 n\right) + 2 \cdot 2^{N_3 n^{1/3}\log^2 n} \leqslant \cdots \leqslant \\
\leqslant & \; 2^{N_3 n^{1/3}\log^2 n \cdot \sum_{i=0}^{\infty}(5/6)^{i/3}} \cdot N_4 + \log_{6/5} n \cdot 2^{N_3 n^{1/3}\log^2 n} = 2^{\mathcal{O}(n^{1/3}\log^2 n)}.
\end{aligned}$$

Solving the recursion in Lemma 5.13

$$\begin{aligned}
F(\mu) \leqslant & \; \max\Big(F(\mu-1) + F(\mu - \mu/54), F(\mu-1) + F(\mu - \mu/6)\Big) \\
= & \; F(\mu-1) + F(\mu - \mu/54) = F(\mu-1) + F(53/54\ \mu) \\
\leqslant & \; F(\mu-2) + 2F(53/54\ \mu) \\
\leqslant & \; \cdots \leqslant \mu \cdot F(53/54\ \mu) \\
\leqslant & \; \mu^2 \cdot F((53/54)^2\mu) \leqslant \; \mu^{\log_{54/53}\mu} \cdot \mathcal{O}(1) = \mu^{\mathcal{O}(\log\mu)}.
\end{aligned}$$