Politechnika Warszawska
Warsaw University of Technology
http://repo.pw.edu.pl

Publikacja / Publication	Fine-Grained Complexity of the List Homomorphism Problem: Feedback Vertex Set and Cutwidth, Piecyk Marta, Rzążewski Paweł
DOI wersji wydawcy / Published version DOI	http://dx.doi.org/10.4230/LIPIcs.STACS.2021.56
Adres publikacji w Repozytorium URL / Publication address in Repository	http://repo.pw.edu.pl/info/article/WUT9ff13ee84b8c428284b1b6e08241322a/
Data opublikowania w Repozytorium / Deposited in Repository on	Apr 2, 2021
Rodzaj licencji / Type of licence	Uznanie Autorstwa (CC-BY)
Cytuj tę wersję / Cite this version	Piecyk Marta, Rzążewski Paweł: Fine-Grained Complexity of the List Homomorphism Problem: Feedback Vertex Set and Cutwidth, In: Proceedings of the 38th International Symposium on Theoretical Aspects of Computer Science (STACS 2021) / Bläser Markus, Monmege Benjamin (eds.), Leibniz International Proceedings in Informatics, vol. 187, 2021, Schloss Dagstuhl-Leibniz-Center for Informatics, ISBN 978-3-95977-180-1, 56:1-56:17, DOI:10.4230/LIPIcs.STACS.2021.56

Fine-Grained Complexity of the List Homomorphism Problem: Feedback Vertex Set and Cutwidth

Marta Piecyk
Faculty of Mathematics and Information Science, Warsaw University of Technology, Poland

Paweł Rzążewski
Faculty of Mathematics and Information Science, Warsaw University of Technology, Poland
Institute of Informatics, University of Warsaw, Poland

Abstract

For graphs G, H, a homomorphism from G to H is an edge-preserving mapping from $V(G)$ to $V(H)$. In the list homomorphism problem, denoted by LHOM(H), we are given a graph G, whose every vertex v is equipped with a list $L(v) \subseteq V(H)$, and we need to determine whether there exists a homomorphism from G to H which additionally respects the lists L. List homomorphisms are a natural generalization of (list) colorings.

Very recently Okrasa, Piecyk, and Rzążewski [ESA 2020] studied the fine-grained complexity of the problem, parameterized by the treewidth of the instance graph G. They defined a new invariant $i^*(H)$, and proved that for every relevant graph H, i.e., such that LHOM(H) is NP-hard, this invariant is the correct base of the exponent in the running time of any algorithm solving the LHOM(H) problem.

In this paper we continue this direction and study the complexity of the problem under different parameterizations. As the first result, we show that $i^*(H)$ is also the right complexity base if the parameter is the size of a minimum feedback vertex set of G, denoted by $\mathrm{fvs}(G)$. In particular, for every relevant graph H, the LHOM(H) problem

- can be solved in time $i^*(H)^{\mathrm{fvs}(G)} \cdot |V(G)|^{\mathcal{O}(1)}$, if a minimum feedback vertex set of G is given,
- cannot be solved in time $(i^*(H) - \varepsilon)^{\mathrm{fvs}(G)} \cdot |V(G)|^{\mathcal{O}(1)}$, for any $\varepsilon > 0$, unless the SETH fails.

Then we turn our attention to a parameterization by the cutwidth $\mathrm{ctw}(G)$ of G. Jansen and Nederlof [TCS 2019] showed that LIST k-COLORING (i.e., LHOM(K_k)) can be solved in time $c^{\mathrm{ctw}(G)} \cdot |V(G)|^{\mathcal{O}(1)}$ for an absolute constant c, i.e., the base of the exponential function does not depend on the number of colors. Jansen asked whether this behavior extends to graph homomorphisms. As the main result of the paper, we answer the question in the negative. We define a new graph invariant $mim^*(H)$, closely related to the size of a maximum induced matching in H, and prove that for all relevant graphs H, the LHOM(H) problem cannot be solved in time $(mim^*(H) - \varepsilon)^{\mathrm{ctw}(G)} \cdot |V(G)|^{\mathcal{O}(1)}$ for any $\varepsilon > 0$, unless the SETH fails. In particular, this implies that, assuming the SETH, there is no constant c, such that for every odd cycle the non-list version of the problem can be solved in time $c^{\mathrm{ctw}(G)} \cdot |V(G)|^{\mathcal{O}(1)}$.

2012 ACM Subject Classification Mathematics of computing → Graph coloring; Theory of computation → Problems, reductions and completeness; Theory of computation → Graph algorithms analysis; Theory of computation → Parameterized complexity and exact algorithms

Keywords and phrases list homomorphisms, fine-grained complexity, SETH, feedback vertex set, cutwidth

Digital Object Identifier 10.4230/LIPIcs.STACS.2021.56

Related Version *Full Version*: `https://arxiv.org/abs/2009.11642` [35]

Funding Supported by Polish National Science Centre grant no. 2018/31/D/ST6/00062.

Acknowledgements We are grateful to Bart M. P. Jansen for introducing us to the problem and to Karolina Okrasa for many fruitful discussions.

© Marta Piecyk and Paweł Rzążewski;
licensed under Creative Commons License CC-BY 4.0

38th International Symposium on Theoretical Aspects of Computer Science (STACS 2021).
Editors: Markus Bläser and Benjamin Monmege; Article No. 56; pp. 56:1–56:17

Leibniz International Proceedings in Informatics
Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

Pobrano z http://repo.pw.edu.pl / Downloaded from Repository of Warsaw University of Technology 2021-09-19

1 Introduction

The k-Coloring problem, which asks whether an input graph G admits a proper coloring with k colors, is arguably one of the best studied computational problems. The problem is known to be notoriously hard: it is polynomial-time solvable (and, in fact, very simple) only for $k \leqslant 2$, and NP-complete otherwise, even on very restricted classes of graphs [15, 19, 20, 26].

For such a hard problem, an interesting direction of research is to study their *fine-grained complexity* depending on some parameters of input instances, in order to understand where the boundary of easy and hard cases lies. Such investigations usually follow two paths in parallel. On one hand, we extend our algorithmic toolbox in order to solve the problem efficiently in various settings. On the other hand, we try to show hardness of the problem, using appropriate reductions.

In order to obtain meaningful lower bounds, the basic assumption of the classical complexity theory, i.e., $\mathsf{P} \neq \mathsf{NP}$, is not strong enough. The usual assumptions used in this context are the Exponential Time Hypothesis (ETH) and the Strong Exponential Time Hypothesis (SETH), both formulated by Impagliazzo and Paturi [21, 22]. The ETH asserts that 3-Sat with n variables cannot be solved in time $2^{o(n)} \cdot n^{\mathcal{O}(1)}$, while the SETH says that CNF-Sat with n variables and m clauses cannot be solved in time $(2-\varepsilon)^n \cdot (n+m)^{\mathcal{O}(1)}$ for any $\varepsilon > 0$.

In case of k-Coloring, the most natural parameter is the number of vertices. While the brute-force approach to solve the problem on an instance G takes time $k^{|V(G)|} \cdot |V(G)|^{\mathcal{O}(1)}$, we know better algorithms where the base of the exponential function does not depend on k. The currently best algorithm is due to Björklund et al. [3] and has complexity $2^{|V(G)|} \cdot |V(G)|^{\mathcal{O}(1)}$. On the other hand, the standard hardness reduction shows that the problem cannot be solved in time $2^{o(|V(G)|)} \cdot |V(G)|^{\mathcal{O}(1)}$, unless the ETH fails [9].

Similarly, we can ask how the complexity depends on some parameters, describing the structure of the instance. The most famous structural parameter is arguably the *treewidth* of the graph, denoted by $\mathrm{tw}(G)$ [2, 5, 36]. Intuitively, treewidth measures how tree-like the graph is. Thus, on graphs with bounded treewidth, we can mimick the bottom-up dynamic programming algorithms that works very well on trees. In case of k-Coloring, the complexity of such a straightforward approach is $k^{\mathrm{tw}(G)} \cdot |V(G)|^{\mathcal{O}(1)}$, provided that G is given along with its tree decomposition of width $\mathrm{tw}(G)$. One might wonder whether this could be improved, in particular, if one can design an algorithm with running time $c^{\mathrm{tw}(G)} \cdot |V(G)|^{\mathcal{O}(1)}$, where c is a constant that does not depend on k, as it was possible in the case if the parameter is $|V(G)|$. Lokshtanov, Marx, and Saurabh [30] proved that this is unlikely, and an algorithm with running time $(k-\varepsilon)^{\mathrm{tw}(G)} \cdot |V(G)|^{\mathcal{O}(1)}$, for any $\varepsilon > 0$, would contradict the SETH. This lower bounds holds even if we replace treewidth with pathwidth $\mathrm{pw}(G)$; the latter result is stronger, as we always have $\mathrm{tw}(G) \leqslant \mathrm{pw}(G)$.

Another way to measure how close a graph G is to a tree or a forest is to analyze the size $\mathrm{fvs}(G)$ of a minimum *feedback vertex set*, i.e., the minimum number of vertices that need to be removed from G to break all cycles. If G is given with a minimum feedback vertex set S, we can solve k-Coloring by enumerating all possible colorings of S, and trying to extend them on the forest $G-S$ using dynamic programming. The running time of such a procedure is $k^{\mathrm{fvs}(G)} \cdot |V(G)|^{\mathcal{O}(1)}$. This is complemented by a hardness result of Lokshtanov et al. [30], who showed that the problem cannot be solved in time $(k-\varepsilon)^{\mathrm{fvs}(G)} \cdot |V(G)|^{\mathcal{O}(1)}$ for any $\varepsilon > 0$, unless the SETH fails. Let us point that $\mathrm{pw}(G)$ and $\mathrm{fvs}(G)$ are incomparable parameters, so this result is incomparable with the previously mentioned lower bound. These two lower bounds were later unified by Jaffke and Jansen [23], who considered the parameterization by the *distance to a linear forest*.

Pobrano z http://repo.pw.edu.pl / Downloaded from Repository of Warsaw University of Technology 2021-09-19

The above examples show a behavior which is typical for many other parameters: the running time of the algorithm depends on the number k of colors and this dependence is necessary under standard complexity assumptions [16, 27, 23]. Thus, it was really surprising that Jansen and Nederlof [25] showed that for any k, the k-Coloring problem can be solved in time $c^{\mathrm{ctw}(G)} \cdot |V(G)|^{\mathcal{O}(1)}$, where c is an absolute constant and $\mathrm{ctw}(G)$ is the *cutwidth* of G. Intuitively, we can imagine $\mathrm{ctw}(G)$ as follows. We fix some ordering of the vertices of G and place them on a horizontal line in this ordering. The edges of G are drawn as arcs above the line; we do not care about intersections. Now, the width of this layout is the maximum number of edges that can be cut by a vertical line. The cutwidth is the minimum width over all linear layouts of vertices of G. The substantial difference between cutwidth and the previously mentioned parameters is that cutwidth corresponds to a number of edges, not a number of vertices. Also, it is known that $\mathrm{pw}(G) \leqslant \mathrm{ctw}(G)$ [4].

Actually, Jansen and Nederlof [25] presented two algorithms for k-Coloring, parameterized by the cutwidth. The first one is deterministic and has running time $2^{\omega \cdot \mathrm{ctw}(G)} \cdot |V(G)|^{\mathcal{O}(1)}$, where $\omega < 2.373$ is the matrix multiplication exponent [7, 39]. The second one is randomized and works in time $2^{\mathrm{ctw}(G)} \cdot |V(H)|^{\mathcal{O}(1)}$. Also, the authors show that the latter complexity is optimal under the SETH.

Let us point out that all the algorithms mentioned above work also for the more general List k-Coloring problem, where each vertex v of G is equipped with a list $L(v) \subseteq \{1, 2, \ldots, k\}$, and we additionally require that the assigned color comes from this list. The general direction of our work is to investigate how further the techniques developed for k-Coloring can be generalized.

Graph homomorphisms. A homomorphism from a graph G to a graph H (called *target*) is an edge-preserving mapping from $V(G)$ to $V(H)$. In the $\textsc{Hom}(H)$ problem we ask if the input graph G admits a homomorphism to H, which is usually treated as a fixed graph. Observe that if H is K_k, i.e., a complete graph on k vertices, then $\textsc{Hom}(H)$ is equivalent to k-Coloring. The complexity classification of $\textsc{Hom}(H)$ was provided by the seminal paper by Hell and Nešetřil [18]: the problem is polynomial-time solvable if H is bipartite or has a vertex with a loop, and NP-complete otherwise. This problem can also be considered in a list setting, where every vertex v of G is equipped with an *H-list* $L(v) \subseteq V(H)$, and we ask for a homomorphism from G to H, which additionally respects lists L. The corresponding computational problem is denoted by $\textsc{LHom}(H)$.

The complexity dichotomy for $\textsc{LHom}(H)$ was proven in three steps: first, for reflexive graphs H (i.e., where every vertex has a loop) by Feder and Hell [11], then for irreflexive graphs H (i.e., with no loops) by Feder, Hell, and Huang [12], and finally, for all graphs H, again by Feder, Hell, and Huang [13]. The problem appears to be polynomial-time solvable if H is a so-called *bi-arc graph*. We will now skip the definition of this class and just mention a special case if H is irreflexive and bipartite: then the $\textsc{LHom}(H)$ problem is in P if the complement of H is a circular-arc graphs, and otherwise the problem is NP-complete. This special case will play a prominent role in our paper.

Let us point out that despite the obvious similarity of $\textsc{Hom}(H)$ and $\textsc{LHom}(H)$, the methods used to prove lower bounds are very different. For $\textsc{Hom}(H)$, all hardness results use some algebraic tools, which allow us to capture the structure of the whole graph H. On the other hand, hardness proofs for $\textsc{LHom}(H)$ are purely combinatorial and are based on the analysis of some small subgraphs of H.

A brute-force approach to solving an instance G of $\textsc{Hom}(H)$ (and $\textsc{LHom}(H)$) has complexity $|V(H)|^{|V(G)|} \cdot |V(G)|^{\mathcal{O}(1)}$. This can be improved if H has some special structure: several algorithms with running time $\mathcal{O}^*(f(H)^{|V(G)|})$ were obtained, where f is a function

Pobrano z http://repo.pw.edu.pl / Downloaded from Repository of Warsaw University of Technology 2021-09-19

of some structural parameter of H [14, 38, 37]. A natural open question was whether one can obtain a $c^{|V(G)|} \cdot |V(G)|^{\mathcal{O}(1)}$ algorithm, where c is a constant that does not depend on H [38]. This question was finally answered in the negative by Cygan et al. [8], who proved that the brute force algorithm is essentially optimal under the ETH.

The fine-grained complexity of the Hom(H) problem, parameterized by the treewidth of G, was studied recently by Okrasa and Rzążewski [34]. The analogous question for LHom(H) was first investigated by Egri et al. [10] for reflexive graphs H, and then by Okrasa et al. [32] for the general case. The authors defined a new graph invariant $i^*(H)$, and proved the following, tight bounds.

▶ **Theorem 1** (Okrasa, Piecyk, Rzążewski [32]). *Let H be a connected, non-bi-arc graph.*

a) *Every instance (G, L) of LHom(H) can be solved in time $i^*(H)^t \cdot |V(G)|^{\mathcal{O}(1)}$, provided that G is given along with a tree decomposition of width t.*

b) *There is no algorithm that solves every instance (G, L) of LHom(H) in time $(i^*(H) - \varepsilon)^{\mathrm{pw}(G)} \cdot |V(G)|^{\mathcal{O}(1)}$ for any $\varepsilon > 0$, unless the SETH fails.*

To the best of our knowledge, the complexity depending on other structural parameters of G was not investigated. In this paper, we make some progress to fill this gap. In particular, our main motivation is the following question by Jansen [24], repeated by Okrasa, Piecyk, Rzążewski [32].

▶ **Question 2** (Jansen [24]). *Is there a universal constant c, such that for every H, every instance G of the Hom(H) problem can be solved in time $c^{\mathrm{ctw}(G)} \cdot |V(G)|^{\mathcal{O}(1)}$?*

Our results. As our first result, we complement the recent result of Okrasa et al. [32] and show tight complexity bounds, parameterized by the size of a minimum feedback vertex set of the instance.

▶ **Theorem 3.** *Let H be a connected, non-bi-arc graph.*

a) *Every instance (G, L) of LHom(H) can be solved in time $i^*(H)^s \cdot |V(G)|^{\mathcal{O}(1)}$, provided that G is given along with a feedback vertex set of size s.*

b) *There is no algorithm that solves every instance (G, L) of LHom(H) in time $(i^*(H) - \varepsilon)^{\mathrm{fvs}(G)} \cdot |V(G)|^{\mathcal{O}(1)}$ for any $\varepsilon > 0$, unless the SETH fails.*

Let us point out that the algorithmic part of the theorem, i.e., the statement a), follows directly from Theorem 1 a), as given a graph G and its feedback vertex set S, we can in polynomial time construct a tree decomposition of G with width $|S| + 1$. The proof of the lower bound follows the general direction of the hardness proof for k-Coloring by Lokshtanov et al. [30]. However, as we are showing hardness for all non-bi-arc graphs H, the gadgets are significantly more complicated. In their construction we use some machinery developed by Okrasa et al. [32]. Unfortunately, most of the gadgets used by Okrasa et al. [32] cannot be used as a black box, as they contain many vertex-disjoint cycles. However, we are able to adjust the constructions so that they work in our setting.

Furthermore, similarly to the proof of Theorem 1 b), the proof of Theorem 3 b) is split into two parts: first we prove hardness for the special case if H is bipartite, and then we reduce the general case to the bipartite one.

Then we turn our attention to the setting, where the parameter is the cutwidth of the instance graph. Recall that $\mathrm{ctw}(G) \geqslant \mathrm{pw}(G) \geqslant \mathrm{tw}(G)$. Furthermore, given a linear layout of G with width w, we can in polynomial time construct a tree decomposition of G with width at most w [4]. Thus by Theorem 1 a) we know that LHom(H) can be solved in time $(i^*(H))^{\mathrm{ctw}(G)} \cdot |V(G)|^{\mathcal{O}(1)}$. On the other hand, we know that this algorithm cannot be optimal for all H, as $i^*(K_k) = k$, while List k-Coloring, i.e., LHom(K_k), can be solved in time $2^{\omega \cdot \mathrm{ctw}(G)} \cdot |V(G)|^{\mathcal{O}(1)}$ [25].

Pobrano z http://repo.pw.edu.pl / Downloaded from Repository of Warsaw University of Technology 2021-09-19

We introduce another parameter $mim^*(H)$, closely related to the size of a maximum induced matching in H, and show the following lower bound.

▶ **Theorem 4.** *For every connected non-bi-arc graph H, there is no algorithm that solves every instance (G, L) of LHom(H) in time $(mim^*(H) - \varepsilon)^{\mathrm{ctw}(G)} \cdot |V(G)|^{\mathcal{O}(1)}$ for any $\varepsilon > 0$, unless the SETH fails.*

As a sanity check, we point out that $mim^*(K_k) = 2$, so our lower bounds are consistent with the results of Jansen and Nederlof [25].

Next, we focus on the non-list variant of the problem, i.e., Hom(H). Note that here we only consider graphs H that are irreflexive and non-bipartite, as otherwise the problem is polynomial-time solvable. Furthermore, we restrict our attention to graphs H that are *projective cores* (see Section 6 for a characterization of these graphs). It is known that almost all irreflexive graphs are non-bipartite, connected projective cores [1, 17, 31, 34]. For this class of graphs H, we show the following lower bounds, answering Question 2 in the negative.

▶ **Theorem 5.** *For every connected non-bipartite, irreflexive projective core H, there is no algorithm that solves every instance G of Hom(H) in time $(mim^*(H) - \varepsilon)^{\mathrm{ctw}(G)} \cdot |V(G)|^{\mathcal{O}(1)}$ for any $\varepsilon > 0$, unless the SETH fails.*

In particular, odd cycles are projective cores [28]. Furthermore, for any odd cycle C_k it holds that $mim^*(C_k) = \lfloor 2k/3 \rfloor$. Thus, we obtain the following as a corollary from Theorem 5.

▶ **Corollary 6.** *Assuming the SETH, there is no universal constant c, such that for every odd cycle C, the Hom(C) problem can be solved in time $c^{\mathrm{ctw}(G)} \cdot |V(G)|^{\mathcal{O}(1)}$ for every instance G.*

We conclude the paper with pointing out some directions for future investigations.

Full version. Due to the page limit, the proofs of some statements, marked with (♠), are omitted or just sketched. The complete proofs can be found in the full version that is available on arXiv [35]. There we also discuss the consequences of our hardness results, if we only assume the ETH. Finally, we generalize the algorithm for k-Coloring by Jansen and Nederlof [25] so that it could be used to solve LHom(H) for every graph H, as well as some other closely related problems.

2 Notation and preliminaries

For a positive integer n, we define $[n] := \{1, \ldots, n\}$. For a set X, by 2^X we denote the set of all subsets of X. Unless explicitly stated otherwise, all logarithms are of base 2, i.e., $\log x := \log_2 x$.

Let G be a graph. For a set $S \subsetneq V(G)$, by $G - S$ we denote the graph induced by $V(G) \setminus S$. For a vertex $v \in V(G)$, by $N_G(v)$ we denote the set of neighbors of v and by $\deg_G(v)$ its degree, i.e., $|N_G(v)|$. If the graph G is clear from the context, we write $N(v)$ and $\deg(v)$ instead of $N_G(v)$ and $\deg_G(v)$. Note that $v \in N(v)$ if and only if v is a vertex with a loop. We say that two vertices $u, v \in V(G)$ are *incomparable* if $N(u) \not\subseteq N(v)$ and $N(v) \not\subseteq N(u)$. A set $S \subseteq V(G)$ is *incomparable* if all its vertices are pairwise incomparable. Equivalently, we can say that for every distinct $u, v \in S$, there is a vertex $u' \in N(u) \setminus N(v)$. A set $S \subseteq V(G)$ is *strongly incomparable* if for every $u \in S$ there exists its *private neighbor* $u' \in N(u)$, such that u' is non-adjacent to every vertex in $S \setminus \{u\}$. Clearly, a strongly incomparable set of vertices is incomparable.

Pobrano z http://repo.pw.edu.pl / Downloaded from Repository of Warsaw University of Technology 2021-09-19

For two graphs G and H, we write $\varphi : G \to H$ if φ is a homomorphism from G to H. If G is given with H-lists L, we write $\varphi : (G, L) \to H$ if φ is a homomorphism from G to H, respecting lists L. We also write $G \to H$ (resp., $(G, L) \to H$) to indicate that some homomorphism $\varphi : G \to H$ (resp., $\varphi : (G, L) \to H$) exists. As graph homomorphisms generalize graph colorings, we will often use the term *coloring* to refer to a homomorphism. Moreover, we refer to the vertices of H as *colors*.

For a graph G, H-lists L, and a set $S \subseteq V(G)$ we define $L(S) := \bigcup_{v \in S} L(v)$. If it does not lead to confusion, for a set V such that $V(G) \subseteq V$ and H-lists $L : V \to 2^{V(H)}$, we will denote the instance $(G, L|_{V(G)})$ by (G, L), in order to simplify the notation.

Let H be a graph. A *walk* $\mathcal{P}$ in H is a sequence $p_1, \ldots, p_\ell$ of vertices of H such that $p_i p_{i+1} \in E(H)$ for $i \in [\ell - 1]$. We define the *length* of a walk $\mathcal{P} = p_1, \ldots, p_\ell$ as $\ell - 1$. We also write $\mathcal{P} : p_1 \to p_\ell$ to emphasize that $\mathcal{P}$ starts in p_1 and ends in p_ℓ. For walks $\mathcal{P} = p_1, \ldots, p_\ell$ and $\mathcal{Q} = q_1, \ldots, q_\ell$ of equal length we say $\mathcal{P}$ *avoids* $\mathcal{Q}$ if $p_1 \neq q_1$ and for every $i \in [\ell - 1]$ it holds that $p_i q_{i+1} \notin E(H)$. By $\overline{\mathcal{P}}$ we denote walk $\mathcal{P}$ reversed, i.e., if $\mathcal{P} = p_1, \ldots, p_\ell$, then $\overline{\mathcal{P}} = p_\ell, \ldots, p_1$. It is straightforward to observe that if $\mathcal{P}$ avoids $\mathcal{Q}$, then $\overline{\mathcal{Q}}$ avoids $\overline{\mathcal{P}}$.

Graph parameters. By $\mathrm{tw}(G)$ and $\mathrm{pw}(G)$ we denote, respectively, the *treewidth* and the *pathwidth* of a graph G. A set $F \subseteq V(G)$, such that $G - F$ does not contain any cycle, is called a *feedback vertex set* of G. We denote the size of a minimum feedback vertex set in G by $\mathrm{fvs}(G)$.

Let $\pi = (v_1, \ldots, v_n)$ be a linear ordering of vertices of G, we will call it a *linear layout* of G. A *cut* of π is a partition of $V(G)$ into two subsets: $\{v_1, \ldots, v_p\}$ and $\{v_{p+1}, \ldots, v_n\}$, for some $p \in [n - 1]$. We say that an edge $v_i v_j$, where $i < j$, *crosses* the cut $(\{v_1, \ldots, v_p\}, \{v_{p+1}, \ldots, v_n\})$, if $i \leqslant p$ and $j > p$. The *width* of the linear layout π is the maximum number of edges that cross any cut of π. Finally, we define the *cutwidth* $\mathrm{ctw}(G)$ of G as the minimum width over all linear layouts of G.

Given a linear layout of G with width k, we can in polynomial time construct a path decomposition of G with width at most k, so in particular $\mathrm{pw}(G) \leqslant \mathrm{ctw}(G)$ [4]. On the other hand, for every graph G it holds that $\mathrm{ctw}(G) \leqslant \mathrm{pw}(G) \cdot \Delta(G)$ [6]. As we also have that $\mathrm{ctw}(G) \geqslant \Delta(G)/2$, we can intuitively think that $\mathrm{ctw}(G)$ is bounded if and only if both $\Delta(G)$ and $\mathrm{pw}(G)$ are bounded.

Bipartite graphs H, for which LHom(H) is NP-hard. Recall that Feder et al. [12] proved that in this case the $\textsc{LHom}(H)$ problem is polynomial-time solvable if H is a complement of a circular-arc graph and NP-complete otherwise. We will rely on the following structural result.

▶ **Lemma 7** (Okrasa et al. [32, 33]). *Let H be a bipartite graph, whose complement is not a circular-arc graph. Then in each bipartition class there exists a triple (α, β, γ) of vertices such that:*

(1) *there exist $\alpha', \beta' \in V(H)$, such that the edges $\alpha\alpha', \beta\beta'$ induce a matching in H,*
(2) *vertices α, β, γ are pairwise incomparable,*
(3) *there exist walks $\mathcal{X}, \mathcal{X}' : \alpha \to \beta$ and $\mathcal{Y}, \mathcal{Y}' : \beta \to \alpha$, such that $\mathcal{X}$ avoids $\mathcal{Y}$ and $\mathcal{Y}'$ avoids $\mathcal{X}'$,*
(4) *at least one of the following holds:*
  a) *H contains an induced C_6 with consecutive vertices $w_1, \ldots, w_6$ and $\alpha = w_1, \beta = w_5, \gamma = w_3$,*
  b) *H contains an induced C_8 with consecutive vertices $w_1, \ldots, w_8$ and $\alpha = w_1, \beta = w_5, \gamma = w_3$,*

Pobrano z http://repo.pw.edu.pl / Downloaded from Repository of Warsaw University of Technology 2021-09-19

c) *the set* $\{\alpha, \beta, \gamma\}$ *is strongly incomparable and for any* a, b, c*, such that* $\{a, b, c\} = \{\alpha, \beta, \gamma\}$*, there exist walks* $\mathcal{X}_c : \alpha \to a$ *and* $\mathcal{Y}_c : \alpha \to b$*, and* $\mathcal{Z}_c : \beta \to c$*, such that* $\mathcal{X}_c, \mathcal{Y}_c$ *avoid* $\mathcal{Z}_c$ *and* $\mathcal{Z}_c$ *avoids* $\mathcal{X}_c, \mathcal{Y}_c$*.*

Incomparable sets, decompositions, and main invariants. In this section we still consider H to be a bipartite graph. First, let us define parameters $i(H)$ and $\mathrm{mim}(H)$.

▶ **Definition 8** ($i(H)$ and $mim(H)$)**.** *Let* H *be a bipartite graph. By* $i(H)$ *(resp.* $mim(H)$*) we denote the maximum size of an incomparable set (resp. strongly incomparable set) in* H*, which is fully contained in one bipartition class.*

Let S be a strongly incomparable set, contained in one bipartition class, and let S' be the set of private neighbors of vertices of S. We observe that the set $S \cup S'$ induces a matching in H of size $|S|$. On the other hand, if M is an induced matching, then the endpoints of edges from M contained in one bipartition class form a strongly incomparable set of size $|M|$. Thus $mim(H)$ can be equivalently defined as the size of a maximum induced matching in H.

Okrasa et al. [32] studied a certain decomposition of bipartite graphs. Its exact definition is not important for us, so we skip it in the conference version. The only thing we need to know is that every bipartite graph, whose complement is not a circular-arc graph, contains an induced subgraph, which is *undecomposable* (i.e., does not admit this decomposition) and its complement is not a circular-arc graph, see e.g. [33, Theorem 46]. This leads to the following definitions.

▶ **Definition 9** ($i^*(H)$ and $mim^*(H)$ for bipartite H)**.** *Let* H *be a bipartite graph, whose complement is not a circular-arc graph. Define*

$$\begin{aligned} i^*(H) &:= \max\{i(H') \colon H' \text{ is an undecomposable, connected, induced} \\ &\qquad \text{subgraph of } H, \text{ whose complement is not a circular-arc graph}\}, \\ mim^*(H) &:= \max\{mim(H') \colon H' \text{ is an undecomposable, connected, induced} \\ &\qquad \text{subgraph of } H, \text{ whose complement is not a circular-arc graph}\}. \end{aligned}$$

Observe that if H is bipartite, connected, undecomposable, and the complement of H is not a circular-arc graph, then $i^*(H) = i(H)$ and $mim^*(H) = mim(H)$.

3 Bipartite H, parameter: the size of a minimum feedback vertex set

In this section we prove Theorem 3 b) in the case that H is bipartite. Assume that the complement of H is not a circular-arc graph, and let (α, β, γ) be the triple given by Lemma 7.

We will introduce two gadgets. The first one is a graph called an *assignment gadget* and has two special vertices. Its main goal is to ensure that a certain coloring of one special vertex forces a certain coloring of the other special vertex.

▶ **Definition 10** (Assignment gadget)**.** *Let* S *be an incomparable set in* H *contained in the same bipartition class as* α, β, γ *and let* $v \in S$*. An* assignment gadget *is a graph* A_v *with* H*-lists* L *and with special vertices* x, y*, such that:*

(A1.) $L(x) = S$ *and* $L(y) = \{\alpha, \beta, \gamma\}$*,*

(A2.) *for every* $u \in S$ *and for every* $a \in \{\alpha, \beta\}$ *there exists a list homomorphism* $\varphi : (A_v, L) \to H$ *such that* $\varphi(x) = u$ *and* $\varphi(y) = a$*,*

(A3.) *there exists a list homomorphism* $\varphi : (A_v, L) \to H$ *such that* $\varphi(x) = v$ *and* $\varphi(y) = \gamma$*,*

(A4.) *for every list homomorphism* $\varphi : (A_v, L) \to H$ *it holds that if* $\varphi(y) = \gamma$*, then* $\varphi(x) = v$*,*

Pobrano z http://repo.pw.edu.pl / Downloaded from Repository of Warsaw University of Technology 2021-09-19

(A5.) $A_v - \{x\}$ *is a tree,*
(A6.) $\deg(x) = (|S|-1)^2$ *and* $\deg(y) = |S| - 1$.

The second gadget is called a *switching gadget*. It is a path T with a special internal vertex q, whose list is $\{\alpha, \beta, \gamma\}$, and endvertices with the same list $\{\alpha, \beta\}$. Coloring both endvertices of T with the same color, i.e., coloring both with α or both with β, allows us to color q with one of α, β, but "switching sides" from α to β forces coloring q with γ.

▶ **Definition 11** (Switching gadget). *A* switching gadget *is a path T of even length with H-lists L, endvertices p, r, called respectively the* input *and the* output *vertex, and one special internal vertex q, called a* q-vertex*, in the same bipartition class as p, r, such that:*
(S1.) $L(p) = L(r) = \{\alpha, \beta\}$ *and* $L(q) = \{\alpha, \beta, \gamma\}$,
(S2.) *for every $a \in \{\alpha, \beta\}$ there exists a list homomorphism $\varphi : (T, L) \to H$, such that $\varphi(p) = \varphi(r) = a$ and $\varphi(q) \neq \gamma$,*
(S3.) *there exists a list homomorphism $\varphi : (T, L) \to H$, such that $\varphi(p) = \alpha$, $\varphi(r) = \beta$, and $\varphi(q) = \gamma$,*
(S4.) *for every list homomorphism $\varphi : (T, L) \to H$, if $\varphi(p) = \alpha$ and $\varphi(r) = \beta$, then $\varphi(q) = \gamma$.*
Note that in a switching gadget we do not care about homomorphisms that map p to β and r to α.

Later, when discussing assignment and switching gadgets, we will use the notions of x-, y-, p-, q-, and r-vertices to refer to the appropriate vertices introduced in the definitions of the gadgets.

▶ **Lemma 12 (♠).** *Let H be an undecomposable, connected, bipartite graph, whose complement is not a circular-arc graph. Let (α, β, γ) be the triple from Lemma 7. Let S be an incomparable set in H contained in the same bipartition class as α, β, γ, such that $|S| \geqslant 2$. Then for every $v \in S$ there exists an assignment gadget A_v.*

Sketch of proof. The construction of an assignment gadget A_v is performed in three steps. First, for every $u \in S \setminus \{v\}$, we construct a gadget $\widetilde{F}_u$ with two special vertices x_u, c_u with lists $L(x_u) = S$ and $L(c_u) = \{\alpha, \beta\}$, such that there are list homomorphisms $\varphi : (\widetilde{F}_u, L) \to H$ that map c_u to β and x_u to any vertex from S, or map c_u to α and x_u to any vertex from $S \setminus \{u\}$, but mapping c_u to α and x_u to u is forbidden. The gadget was first introduced in [33, first step in Lemma 4], but our construction is slightly different as we additionally ensure that $\widetilde{F}_u - \{x_u\}$ is a tree and $\deg(x_u) = |S| - 1$ and $\deg(c_u) = 1$. We point out that using the original gadgets intruduces many vertex-disjoint cycles, which increases the size of a smallest feedback vertex set in the constructed graph.

In the second step, for every $u \in S \setminus \{v\}$ we construct a path P_u with endvertices c'_u and y_u with lists $L(c'_u) = \{\alpha, \beta\}$ and $L(y_u) = \{\alpha, \beta, \gamma\}$, such that it is possible to map the pair (c'_u, y_u) to any pair of $\{\alpha, \beta\}^2$, or to (α, γ), but the pair (β, γ) is forbidden.

The construction of P_u depends on the case in Lemma 7 (4). In case (4a), P_u is the path with lists of consecutive vertices $\{w_1, w_5\}, \{w_2, w_6\}, \{w_1, w_3, w_5\}$. In case (4b), P_u is the path with lists of consecutive vertices $\{w_1, w_5\}, \{w_2, w_6\}, \{w_1, w_3, w_7\}, \{w_2, w_4, w_6, w_8\}, \{w_1, w_3, w_5\}$.

Finally, in case (4c), we will use walks given by Lemma 7 (4c). We construct an auxiliary path P'_u, such that the list of its i-th vertex is the set of i-th vertices of the walks $\mathcal{X}_\alpha, \mathcal{Y}_\alpha, \mathcal{Z}_\alpha$. Similarly we construct a path P''_u using walks $\overline{\mathcal{X}_\gamma}, \overline{\mathcal{Y}_\gamma}, \overline{\mathcal{Z}_\gamma}$ and a path P'''_u using walks $\mathcal{X}_\gamma, \mathcal{Y}_\gamma, \mathcal{Z}_\gamma$. We obtain P_u by identifying the last vertex of P'_u with the first vertex of P''_u, and the last vertex of P''_u with the first vertex of P'''_u. We set c'_u to be the first vertex of P'_u and y_u to be the last vertex of P'''_u.

Pobrano z http://repo.pw.edu.pl / Downloaded from Repository of Warsaw University of Technology 2021-09-19

Next, we introduce a copy of $\widetilde{F}_u$ and identify the vertex c_u from $\widetilde{F}_u$ with the vertex c'_u from P_u. Now, if x_u is mapped to some vertex from $S \setminus \{u\}$, then y_u can be mapped to any of α, β, γ. However, if x_u is mapped to u, then y_u can only be mapped to α or β. Let F_u be the graph obtained in this step.

Finally, we obtain the assignment gadget A_v by introducing the gadget F_u for every $u \in S \setminus \{v\}$, and identifying all x_u's into one vertex x and all y_u's into one vertex y. ◀

▶ **Lemma 13 (♠).** *Let H be an undecomposable, connected, bipartite graph, whose complement is not a circular-arc graph. Let (α, β, γ) be the triple from Lemma 7. Then there exists a switching gadget T.*

Sketch of proof. Again, we consider cases of Lemma 7 (4). In cases (4a) and (4b) the path T is the path with lists of consecutive vertices: $\{w_1, w_5\}$, $\{w_2, w_4\}$, $\{w_1, w_3, w_5\}$, $\{w_2, w_4\}$, $\{w_1, w_5\}$. We set p, q, r to be, respectively, the first, the third, and the fifth vertex of T.

In case (4c) we construct T similarly as we constructed P_u in the proof of Lemma 12, using walks given by Lemma 7 (3) and (4c). We construct a path T' using walks $\mathcal{X}_\beta, \mathcal{Y}_\beta, \mathcal{Z}_\beta$, a path T'' using walks $\overline{\mathcal{X}}_\alpha, \overline{\mathcal{Y}}_\alpha, \overline{\mathcal{Z}}_\alpha$, and a path T''' using walks $\mathcal{X}', \mathcal{Y}'$. We obtain T by identifying the last vertex of T' with the first vertex of T'', and the last vertex of T'' with the first vertex of T'''. The vertices p, q, r are, respectively, the first vertex of T, the last vertex of T', and the last vertex of T. ◀

Reduction. Suppose that we can construct both, the assignment gadget and the switching gadget. Let us show that this is sufficent to prove Theorem 3 b) in the case that H is bipartite. The proof is an extension of the construction of Lokshtanov et al. for the List k-Coloring problem [30].

▶ **Theorem 14.** *For every connected bipartite graph H, whose complement is not a circular-arc graph, there is no algorithm that solves every instance (G, L) of LHom(H) in time $(i^*(H) - \varepsilon)^{\mathrm{fvs}(G)} \cdot |V(G)|^{\mathcal{O}(1)}$ for any $\varepsilon > 0$, unless the SETH fails.*

Proof. Let us point out that it is sufficient to show the theorem if we additionally assume that H is undecomposable. Indeed, assume the SETH and suppose the theorem holds for every bipartite undecomposable graph H', and it does not hold for every bipartite graph H. Then there exist a connected, bipartite graph H, whose complement is not a circular-arc graph, and an algorithm that solves LHom(H) for every instance (G, L) in time $(i^*(H) - \varepsilon)^{\mathrm{fvs}(G)} \cdot |V(G)|^{\mathcal{O}(1)}$ for some $\varepsilon > 0$. Let H' be an induced subgraph of H such that H' is connected, undecomposable, is not a complement of a circular-arc graph, and $i(H') = i^*(H)$. Any instance (G, L) of LHom(H') can be seen as an instance of LHom(H) such that only vertices of H' appear on lists L. Thus we can solve any instance (G, L) of LHom(H') in time $(i^*(H) - \varepsilon)^{\mathrm{fvs}(G)} \cdot |V(G)|^{\mathcal{O}(1)} = (i(H') - \varepsilon)^{\mathrm{fvs}(G)} \cdot |V(G)|^{\mathcal{O}(1)}$, a contradiction.

So from now on we assume that H is undecomposable. In particular, $i^*(H) = i(H)$. Let ϕ be an instance of CNF-Sat with n variables and m clauses. Let $\varepsilon > 0$ and $k := i(H)$. Let S be a maximum incomparable set contained in one bipartition class of H, i.e., $|S| = k$. Let α, β, γ be the vertices of H, in the same bipartition class as S, given by Lemma 7. Let α', β' be the vertices such that edges $\alpha\alpha', \beta\beta'$ induce a matching in H, they exist by Lemma 7. Observe that $k \geqslant 3$, since vertices α, β, γ are pairwise incomparable. Moreover, we define $\lambda := \log_k(k - \varepsilon)$. Observe that $\lambda < 1$. We choose a positive integer p sufficiently large so that $\lambda^{\frac{p}{p-1}} < 1$ and define $t := \left\lceil \frac{n}{\lfloor \log k^p \rfloor} \right\rceil = \left\lceil \frac{n}{\lfloor p \cdot \log k \rfloor} \right\rceil$.

Pobrano z http://repo.pw.edu.pl / Downloaded from Repository of Warsaw University of Technology 2021-09-19

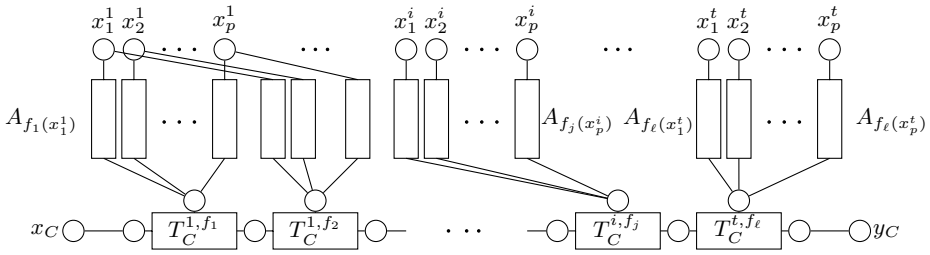

Figure 1 The path P_C for a clause C and vertices x_s^i for $i \in [t], s \in [p]$.

We will construct a graph G with H-lists L such that $\mathrm{fvs}(G) \leqslant t \cdot p$ and $(G, L) \to H$ if and only if ϕ is satisfiable. We partition the variables of ϕ into t sets $F_1, \ldots, F_t$ called *groups*, such that $|F_i| \leqslant \lfloor \log k^p \rfloor$. For each $i \in [t]$ we introduce p vertices $x_1^i, \ldots, x_p^i$ and for every $s \in [p]$ we set $L(x_s^i) := S$. We will interpret a coloring of these vertices as a truth assignment of variables in F_i. Note that there are at most $2^{\lfloor \log k^p \rfloor} \leqslant k^p$ possible truth assigments of variables in F_i and there are k^p possible colorings of $x_1^i, \ldots, x_p^i$, respecting lists L. Thus we can define an injective mapping that assigns a distinct coloring of vertices $x_1^i, \ldots, x_p^i$ to each truth assignment of the variables in F_i, note that some colorings may remain unassigned.

For every clause C of ϕ we introduce a path P_C constructed as follows. Consider a group F_i that contains at least one variable from C, and a truth assignment of F_i that satisfies C. Recall that this assignment corresponds to a coloring f of vertices $x_1^i, \ldots, x_p^i$. We introduce a switching gadget $T_C^{i,f}$, whose q-vertex is denoted by $q_C^{i,f}$. We fix an arbitrary ordering of all switching gadgets introduced for the clause C. For every switching gadget but the last one, we identify its output vertex with the input vertex of the succesor. We add vertices x_C with $L(x_C) = \{\alpha'\}$ and y_C with $L(y_C) = \{\beta'\}$. We add an edge between x_C and the input of the first switching gadget, and between y_C and the output of the last switching gadget. This completes the construction of P_C.

Now consider a switching gadget $T_C^{i,f}$ introduced in the previous step. Recall that C is a clause of ϕ, and f is a coloring of $x_1^i, \ldots, x_p^i$ corresponding to a truth assignment of variables in F_i, which satisfies C. Let us define $v_s := f(x_s^i)$ for $s \in [p]$. For every $s \in [p]$, we call Lemma 12 to construct the assignment gadget A_{v_s}. We identify the x-vertex of A_{v_s} with x_s^i and the y-vertex with $q_C^{i,f}$. This completes the construction of (G, L) (see Figure 1). The properties of the gadgets ensure that ϕ is satisfiable if and only if $(G, L) \to H$ (♠). Furthermore, $|V(G)| = (n+m)^{\mathcal{O}(1)}$ and $\bigcup_{i=1}^{t} \{x_1^i, \ldots, x_p^i\}$ is a feedback vertex set in G, so $\mathrm{fvs}(G) \leqslant t \cdot p$ (♠).

Suppose that the instance (G, L) of $\mathrm{LHom}(H)$ can be solved in time $(k-\varepsilon)^{\mathrm{fvs}(G)} \cdot |V(G)|^{\mathcal{O}(1)} \leqslant (k-\varepsilon)^{t \cdot p} \cdot |V(G)|^{\mathcal{O}(1)}$. Recall that ϕ is satisfiable if and only if $(G, L) \to H$. By a careful analysis of the exponent in the complexity bound (♠) we conclude that ϕ can be solved in time $(2-\delta)^n \cdot (n+m)^{\mathcal{O}(1)}$ for some $\delta > 0$, which contradicts the SETH. ◀

Let us point out that the pathwidth of the graph constructed in the proof above is bounded by $t \cdot p + f(H)$, for some function f of H (see also [30]).

4 Bipartite H, parameter: cutwidth

Similarly as in the previous section, let us first prove Theorem 4 in the case that H is bipartite. We will modify the reduction from Theorem 14. To get an intuition about what needs to be done, recall that in order to obtain a bound on the cutwidth, we need to bound

Pobrano z http://repo.pw.edu.pl / Downloaded from Repository of Warsaw University of Technology 2021-09-19

the pathwidth and the maximum degree. Also, as we already observed, the pathwidth of the instance constructed in Theorem 14 is upper-bounded by the correct value, so we need to take care of vertices of large degree.

▶ **Theorem 15.** *For every connected bipartite graph H, whose complement is not a circular-arc graph, there is no algorithm that solves every instance (G, L) of LHOM(H), given with a linear layout of width at most w, in time $(mim^*(H) - \varepsilon)^w \cdot |V(G)|^{\mathcal{O}(1)}$ for any $\varepsilon > 0$, unless the SETH fails.*

Proof. First, similarly to the proof of Theorem 14 in the case that H is bipartite, it is sufficient to show the proof in case that H is undecomposable. In particular $mim^*(H) = mim(H)$.

Let S be a strongly incomparable set in H of size $k = mim(H)$, contained in one bipartition class. Let S' be a set such that $S \cup S'$ induces a matching of size k in H, and let (α, β, γ) be the triple given by Lemma 7, such that α, β, γ are in the same bipartition class as S. Let ϕ be an instance of CNF-SAT with n variables and m clauses. Let $\varepsilon > 0$. As in the proof of Theorem 14, we choose an integer p so that $\log_k(k - \varepsilon) \cdot \frac{p}{p-1} < 1$ and set $t := \left\lceil \frac{n}{\lfloor p \cdot \log k \rfloor} \right\rceil$. We will construct an instance $(\widetilde{G}, \widetilde{L})$ of LHOM(H) with a linear layout of width at most $t \cdot p + f(H)$, where f is some function of H, such that $(\widetilde{G}, \widetilde{L}) \to H$ if and only if ϕ is satisfiable. We repeat the construction of the instance (G, L) of LHOM(H), such that $(G, L) \to H$ if and only if ϕ is satisfiable, from the proof of Theorem 14. This is possible since S is in particular incomparable. Furthermore, in the construction of (G, L) we did not use the fact that S was maximum, we only needed that $|S| \geqslant 2$, which is the case as $\{\alpha, \beta\}$ is strongly incomparable. We are going to modify the instance (G, L) into the desired instance $(\widetilde{G}, \widetilde{L})$.

Before we do that, let us fix an arbitrary ordering of clauses $C_1, \ldots, C_m$ in ϕ, which implies the ordering of paths P_C in G. Then we can fix an ordering of all q-vertices in G, so that a q-vertex q_1 precedes a q-vertex q_2, if:

- q_1 belongs to the path P_{C_i} and q_2 belongs to the path P_{C_j}, such that $i < j$, or
- q_1 and q_2 belong to the same path P_C, and q_1 precedes q_2 on P_C (the order of the vertices of each path P_C is such that x_C is the first vertex and y_C is the last vertex).

Finally, we fix an ordering of the assignment gadgets in G. Recall that every q-vertex $q_C^{i,f}$ is a y-vertex of p assignment gadgets whose x-vertices are, respectively, $x_1^i, \ldots, x_p^i$. We fix an ordering of the assignment gadgets so that the assignment gadget A_1 precedes the assignment gadget A_2 if:

- the y-vertex of A_1 precedes the y-vertex of A_2 in the fixed order of the q-vertices, or
- A_1 and A_2 have the same y-vertex $q_C^{i,f}$ and x-vertices of A_1 and A_2 are, respectively, x_j^i and x_s^i, with $j < s$.

Now we are ready to modify the instance (G, L). It turns out that we only need to take care of q-vertices and x-vertices, as their large degree forces large cutwidth. The construction of $(\widetilde{G}, \widetilde{L})$ will be thus performed in two steps.

Step 1. Splitting q-vertices. Recall that every q-vertex of a switching gadget is a y-vertex of p assignment gadgets and the degree of each y-vertex in the assignment gadget is $k - 1$. For every q-vertex q, in order to reduce its degree, we will split q into $p \cdot (k - 1)$ vertices $q_1, \ldots, q_{p \cdot (k-1)}$. In this step, the construction depends on the structure of H. Let us consider two cases.

Case I. The set $\{\alpha, \beta, \gamma\}$ is strongly incomparable. Let $\overline{\alpha}, \overline{\beta}, \overline{\gamma}$ be vertices such that edges $\alpha\overline{\alpha}, \beta\overline{\beta}, \gamma\overline{\gamma}$ induce a matching in H. We replace every q-vertex q from a path P_C with $p \cdot (k - 1)$ vertices $q_1, \ldots, q_{p \cdot (k-1)}$, each for every neighbor of q inside assignment gadgets.

Pobrano z http://repo.pw.edu.pl / Downloaded from Repository of Warsaw University of Technology 2021-09-19

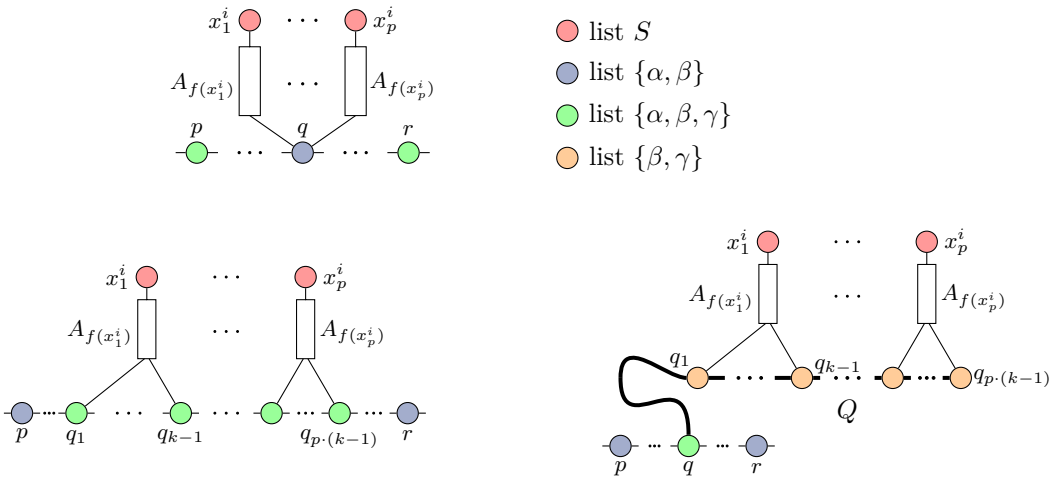

Figure 2 The switching gadget T and the group of vertices x_s^i for $s \in [p]$ before the step of splitting q-vertices (top), after the step in the case that $\{\alpha,\beta,\gamma\}$ is a strongly incomparable set (bottom, left), and after introducing the path Q (marked by the bold curve) in the case that $\{\alpha,\beta,\gamma\}$ is not strongly incomparable (bottom, right).

For every $j \in [p\cdot(k-1)-1]$ we introduce a path Q_j of length 2 with lists of consecutive vertices $\{\alpha,\beta,\gamma\},\{\overline{\alpha},\overline{\beta},\overline{\gamma}\},\{\alpha,\beta,\gamma\}$, and we identify its endvertices with q_j and q_{j+1}. In the same way, we introduce paths Q_0 and $Q_{p\cdot(k-1)}$ and we identify endvertices of Q_0 with q_1 and the vertex preceding q on P_C, and we identify endvertices of $Q_{p\cdot(k-1)}$ with $q_{p\cdot(k-1)}$ and the vertex following q on P_C (see Figure 2). Finally, let us fix an ordering $a_1,a_2,\ldots,a_{p\cdot(k-1)}$ of neighbors of q in assignment gadgets such that for $j \in [p-1]$ vertices of the assignment gadget with the x-vertex x_j^i precede vertices of the assignment gadget with the x-vertex x_{j+1}^i. The order of the neighbors from the same assignment gadget is arbitrary. For every $j \in [p\cdot(k-1)]$ we add an edge between q_j and a_j (see Figure 2). This completes the step of splitting q-vertices in this case.

Case II: The set $\{\alpha,\beta,\gamma\}$ is not strongly incomparable. By Lemma 7 this means that H contains an induced C_6 or C_8 with consecutive vertices $w_1,\ldots,w_6(,w_7,w_8)$ and $\alpha = w_1$, $\beta = w_5$, $\gamma = w_3$. In this case we leave each q-vertex q in the graph, but we introduce a path Q with H-lists L, with q as one of endvertices, special vertices q_j for $j \in [p\cdot(k-1)]$, with list $L(q_j) = \{\beta,\gamma\}$ and such that:

- for every list homomorphism $\varphi : (Q,L) \to H$, if q is mapped to γ, then for every $j \in [p\cdot(k-1)]$ the vertex q_j is mapped to γ.
- there exists a list homomorphism $\varphi : (Q,L) \to H$ such that $\varphi(q) = \gamma$ and $\varphi(q_j) = \gamma$ for every $j \in [p\cdot(k-1)]$.
- for every $c \in \{\alpha,\beta\}$ there exists a list homomorphism $\varphi : (Q,L) \to H$ such that q is mapped to c and for every $j \in [p\cdot(k-1)]$ the vertex q_j is mapped to β.

The path Q is constructed using the walks from Lemma 7, similarly as we did in Lemma 12 and Lemma 13 (♠). Again, for each neighbor a_j of q (the neighbors of $q_C^{i,f}$ are ordered as in the previous case) we add an edge $q_j a_j$ and remove the edge $q a_j$ (see Figure 2).

This completes the Step 1. We will refer to the newly introduced vertices q_j as q-vertices.

Step 2. Splitting x-vertices. The only vertices that might still have large degree are vertices from $\{x_j^i \mid i \in [t], j \in [p]\}$. More precisely, the degree of the x-vertex in an assignment gadget is $(k-1)^2$, and thus the degree of an x-vertex x is $d = d(x)\cdot(k-1)^2$, where $d(x)$

Pobrano z http://repo.pw.edu.pl / Downloaded from Repository of Warsaw University of Technology 2021-09-19

is the number of the assignment gadgets, whose x-vertex is x. We replace the vertex x with d vertices $x_1, \ldots, x_d$, each with list S. For every $s \in [d-1]$ we introduce a path X_s of length 2, lists of consecutive vertices S, S', S, and we identify its endvertices with x_s and x_{s+1}, respectively. We fix an ordering $b_1, \ldots, b_d$ of neighbors of x, such that if b_i and b_j belong, respectively, to assignment gadgets A_i and A_j, and A_i precedes A_j in the fixed order of the assignment gadgets, then b_i precedes b_j. The order of the neighbors from the same assignment gadget is arbitrary. For every $s \in [d]$ we add an edge $b_s x_s$. We will refer to the new vertices x_j introduced in this step also as x-vertices.

It can be verified that $(\widetilde{G}, \widetilde{L}) \to H$ if and only if ϕ is satisfiable (♠). Furthermore, we can specify a linear layout π of $\widetilde{G}$ with width at most $w := t \cdot p + f(H)$, for some function f, as follows (♠). We order the vertices of the original paths P_C (those from graph G), such that the vertices from P_{C_j} precede vertices of $P_{C_{j+1}}$, and the vertices from one path P_C are ordered in a natural way (the vertex x_C is the first one and the vertex y_C is the last one). Then, if Case 1. in Step 1. was applied, we replace each q-vertex q with vertices q_j and vertices of paths Q_j in the following order: $Q_0, q_1, Q_1, \ldots, q_{p\cdot(k-1)}, Q_{p\cdot(k-1)}$. If Case 2. was applied, we insert the vertices from the path Q just after q, in the natural order with q being the first one.

Now we need to place the vertices of assignment gadgets and of paths X_s. We insert the vertices of an assignment gadget A_v, whose y-vertex was q, just after q-vertices adjacent to A_v, which were introduced for q in Step 1. We also insert there the vertices from those paths X_s, whose endvertices are adjacent to A_v.

To see that the width of π is at most $t \cdot p + f(H)$ observe that we placed vertices from each assignment gadget close to each other and to the vertices adjacent to that gadget. The number of the edges with at least one endpoint in a fixed assignment gadget is bounded by some constant $f(H)$. The only edges between vertices that are possibly "far" are edges from paths X connecting x-vertices adjacent to different assignment gadgets and in each cut their number is bounded by the number of original x-vertices, i.e., $t \cdot p$.

Now suppose there is an algorithm that solves every instance (G, L) of LHom(H) in time $(k-\varepsilon)^w \cdot |V(G)|^{\mathcal{O}(1)}$. Then for an instance ϕ of CNF-Sat we can construct the instance $(\widetilde{G}, \widetilde{L})$ as above and we can solve $(\widetilde{G}, \widetilde{L})$ in time $(k-\varepsilon)^{t\cdot p+f(H)} \cdot |V(G)|^{\mathcal{O}(1)}$, which is equivalent to solving the instance ϕ. As in the proof of Theorem 14, we conclude that this implies that CNF-Sat with n variables and m clauses can be solved in time $(2-\delta)^n \cdot (n+m)^{\mathcal{O}(1)}$ for some $\delta > 0$, which contradicts the SETH. ◀

5 Hardness for general target graphs

For a graph H, the *associated bipartite graph* H^* is the graph with vertex set $V(H^*) = \{v', v'' \mid v \in V(H)\}$, whose edge set contains those pairs $u'v''$, for which $uv \in E(H)$.

Recall that for general graphs H, Feder et al. [13] showed that the LHom(H) problem is polynomial-time solvable if H is a bi-arc graph, and NP-complete otherwise. They also observed that H is a bi-arc graph if and only if H^* is the complement of a circular-arc graph. Furthermore, an irreflexive graph is bi-arc if and only if it is bipartite and its complement is a circular-arc graph. Thus "hard" cases of LHom(H) correspond to the "hard" cases of LHom(H^*).

Observe that if H is bipartite, then H^* consists of two disjoint copies of H. Thus for bipartite H it holds that $i^*(H^*) = i^*(H)$ and $mim^*(H^*) = mim^*(H)$. On the other hand, if H is non-bipartite and additionally connected, then H^* is connected. This motivates the following extension of the definition of i^* and mim^* to non-bipartite H.

Pobrano z http://repo.pw.edu.pl / Downloaded from Repository of Warsaw University of Technology 2021-09-19

▶ **Definition 16** ($i^*(H)$ and $mim^*(H)$)**.** *Let H be a non-bi-arc graph. Define:*

$$i^*(H) := i^*(H^*) \qquad \text{and} \qquad mim^*(H) := mim^*(H^*).$$

Observe that that the instances (G, L) constructed in the proofs of Theorem 14 or Theorem 15 are bipartite. Indeed, the instance constructed in Theorem 14 consists of paths P_C and assignment gadgets, whose vertices were appropriately identified. More precisely, we identify some q-vertices (from switching gadgets belonging to paths P_C) with y-vertices (from addignment gadgets), and x-vertices with another x-vertices (from assignment gadgets). Recall that each assignment gadget is bipartite by property (A3.) of Definition 10. Moreover, for each assignment gadget, the x-vertex is in the same bipartition class as the y-vertex. Similarly, for each path P_C, all q-vertices are in the same bipartition class. Therefore, the instance (G, L) constructed in Theorem 14 is bipartite. The instance from Theorem 15 was obtained from the instance (G, L) from Theorem 14 by splitting some vertices into a set of vertices joined by paths of even lenght, so the instance remains bipartite.

Furthermore, if the bipartition classes of G are X and Y, then $L(X)$ is contained in one bipartition class of H, and $L(Y)$ is contained in the other one. Finally, without loss of generality we can assume that for each $v \in V(G)$, the set $L(v)$ is incomparable. Indeed, if $L(v)$ contains two distincts vertices x, y, such that $N_H(x) \subseteq N_H(y)$, we can safely remove x from $L(v)$, obtaining an equivalent instance. Instances satisfying these three conditions are called *consistent*.

▶ **Proposition 17** ([32, 33])**.** *Let H be a graph and let (G, L) be a consistent instance of LHOM(H^*). Define L' as $L'(x) := \{u \colon \{u', u''\} \cap L(x) \neq \emptyset\}$. Then $(G, L) \to H^*$ if and only if $(G, L') \to H$.*

Now we can prove Theorem 3 b) and Theorem 4 (♠).

Sketch of proof of Theorem 3 b) and Theorem 4. If H is bipartite, we are done by Theorem 14 or Theorem 15. Otherwise H^* is connected. Let (G, L) be an instance of LHOM(H^*) constructed in the proof of Theorem 14 or Theorem 15. Let (G, L') be an equivalent instance of LHOM(H) given by Proposition 17. As the instance graph remains the same, the lower bound holds. ◀

Note that the statement of Theorem 15 actually implies the following, slightly stronger result.

▶ **Corollary 18.** *For every connected non-bi-arc graph H, there is no algorithm that solves every instance (G, L) of LHOM(H), given with a linear layout of width at most w, in time $(mim^*(H) - \varepsilon)^w \cdot |V(G)|^{\mathcal{O}(1)}$ for any $\varepsilon > 0$, unless the SETH fails.*

6 Hardness of Hom(H)

In this section we extend Theorem 4 to the non-list case, i.e., we prove Theorem 5. Let us first discuss the graph class mentioned in the statement. Recall that HOM(H) is NP-hard if H is non-bipartite and has no loops [18]. In particular, this implies that H has at least three vertices. We say that a graph H is a *core* if every homomorphism $\varphi : H \to H$ is an automorphism, i.e., is injective and surjective. We also need the following characterization of projective graphs.

Pobrano z http://repo.pw.edu.pl / Downloaded from Repository of Warsaw University of Technology 2021-09-19

▶ **Theorem 19** (Larose, Tardif [29]). *Let H be graph with at least three vertices. Then H is projective if and only if for every $L \subseteq V(H)$ there exist a tuple $(x_1, \ldots, x_\ell)$ of vertices in H and a graph F_L with a tuple of its vertices $(y_0, y_1, \ldots, y_\ell)$ such that*

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

Now we are ready to prove Theorem 5.

▶ **Theorem 5.** *For every connected non-bipartite, irreflexive projective core H, there is no algorithm that solves every instance G of $\textsc{Hom}(H)$ in time $(mim^*(H) - \varepsilon)^{\mathrm{ctw}(G)} \cdot |V(G)|^{\mathcal{O}(1)}$ for any $\varepsilon > 0$, unless the SETH fails.*

Sketch of proof. Let H be as in the statement. As non-bipartite irreflexive graphs are not bi-arc graphs [13], we can use Corollary 18. Let (G, L) be an instance of $\textsc{LHom}(H)$, and let $\pi = (v_1, \ldots, v_{|V(G)|})$ be a linear layout of G of width w. Consider an instance $\widetilde{G}$ of $\textsc{Hom}(H)$ constructed as follows. For every $v_i \in V(G)$ we call Theorem 19 to obtain the tuple $(x_1^{(i)}, \ldots, x_{\ell_i}^{(i)})$ of vertices in H and a graph $F_{L(v_i)}$ with special vertices $y_0^{(i)}, \ldots, y_{\ell_i}^{(i)}$. For every v_i we introduce a copy $H^{(i)}$ of the graph H and identify vertices $y_1^{(i)}, \ldots, y_{\ell_i}^{(i)}$, respectively with $x_1^{(i)}, \ldots, x_{\ell_i}^{(i)}$ in the copy $H^{(i)}$. Moreover, we identify $y_0^{(i)}$ with v_i. Finally, for every $i \in [|V(G)| - 1]$ we add edges between the copies $H^{(i)}$ and $H^{(i+1)}$ as follows. For every vertex $z^{(i)}$ in $H^{(i)}$ and its corresponding copy $z^{(i+1)}$ in $H^{(i+1)}$ we add all edges between $z^{(i)}$ and $N_{H^{(i+1)}}(z^{(i+1)})$. This completes the construction of $\widetilde{G}$.

Theorem 19 implies that $(G, L) \to H$ if and only if $\widetilde{G} \to H$: every graph $F_{L(v)}$ together with a copy of H forces that a vertex v can be colored only with vertices from $L(v)$ and thus it imitates the list of v (♠). Furthermore, $\mathrm{ctw}(\widetilde{G}) \leqslant w + g(H)$ for some function g of H: the copies of H are connected in the appropriate order and thus the linear layout π of G can be easily modified to a linear layout $\widetilde{\pi}$ of $\widetilde{G}$ with width larger than w only by a constant depending on H (♠).

Now suppose that $\textsc{Hom}(H)$ can be solved for every instance G' in time $(mim^*(H) - \varepsilon)^{\mathrm{ctw}(G')} \cdot |V(G')|^{\mathcal{O}(1)}$ for some $\varepsilon > 0$. Then, for an instance (G, L) of $\textsc{LHom}(H)$ with a linear layout π of width w, we can construct in polynomial time the instance $\widetilde{G}$ of $\textsc{Hom}(H)$ as above. We solve the instance $\widetilde{G}$ in time $(mim^*(H) - \varepsilon)^{\mathrm{ctw}(\widetilde{G})} \cdot |V(\widetilde{G})|^{\mathcal{O}(1)}$, which is equivalent to solving the instance (G, L) in time $(mim^*(H) - \varepsilon)^{w} \cdot |V(G)|^{\mathcal{O}(1)}$. By Corollary 18, this contradicts the SETH. ◀

7 Conclusion

A natural open question is to close the gap between lower and upper bounds for the complexity of $\textsc{LHom}(H)$, parameterized by the cutwidth. As a concrete problem, we believe that a good starting point is to understand the complexity of $\textsc{LHom}(C_k)$, where $k \geqslant 5$. Recall that we have a lower bound $mim^*(C_k)^{\mathrm{ctw}(G)} \cdot |V(G)|^{\mathcal{O}(1)}$ and an upper bound $i^*(C_k)^{\mathrm{ctw}(G)} \cdot |V(G)|^{\mathcal{O}(1)}$. The value of $mim^*(C_k)$ is $\lfloor k/3 \rfloor$ if k is even, and $\lfloor 2k/3 \rfloor$ if k is odd. On the other hand, $i^*(C_k)$ is $k/2$ if k is even, and k if k is odd. Where does the truth lie? To be even more specific, what is the complexity of $\textsc{LHom}(C_6)$?

Another research direction that we find exciting is to study the complexity of $\textsc{Hom}(H)$ and $\textsc{LHom}(H)$, depending on different parameters of the instance graph. In particular, Lampis [27] showed that k-Coloring on a graph G can be solved in time $\mathcal{O}^*((2^k - 2)^{\mathrm{cw}(G)})$, where $\mathrm{cw}(G)$ is the *clique-width* of G. Furthermore, an algorithm with a running time $\mathcal{O}^*((2^k - 2 - \varepsilon)^{\mathrm{cw}(G)})$, for any $\varepsilon > 0$, would contradict the SETH. We believe it is exciting to investigate how these results generalize to non-complete target graphs H.

Pobrano z http://repo.pw.edu.pl / Downloaded from Repository of Warsaw University of Technology 2021-09-19

References

1 Noga Alon and Joel H. Spencer. *The Probabilistic Method, Third Edition*. Wiley-Interscience series in discrete mathematics and optimization. Wiley, 2008.

2 Stefan Arnborg and Andrzej Proskurowski. Linear time algorithms for NP-hard problems restricted to partial k-trees. *Discret. Appl. Math.*, 23(1):11–24, 1989. doi:10.1016/0166-218X(89)90031-0.

3 Andreas Björklund, Thore Husfeldt, and Mikko Koivisto. Set partitioning via inclusion-exclusion. *SIAM J. Comput.*, 39(2):546–563, 2009. doi:10.1137/070683933.

4 H. L. Bodlaender. Classes of graphs with bounded tree-width. *Bulletin of EATCS*, pages 116–128, 1988.

5 Hans L. Bodlaender and Arie M. C. A. Koster. Combinatorial optimization on graphs of bounded treewidth. *Comput. J.*, 51(3):255–269, May 2008. doi:10.1093/comjnl/bxm037.

6 Fan R. K. Chung and Paul D. Seymour. Graphs with small bandwidth and cutwidth. *Discret. Math.*, 75(1-3):113–119, 1989. doi:10.1016/0012-365X(89)90083-6.

7 Don Coppersmith and Shmuel Winograd. Matrix multiplication via arithmetic progressions. *J. Symb. Comput.*, 9(3):251–280, 1990. doi:10.1016/S0747-7171(08)80013-2.

8 Marek Cygan, Fedor V. Fomin, Alexander Golovnev, Alexander S. Kulikov, Ivan Mihajlin, Jakub Pachocki, and Arkadiusz Socala. Tight lower bounds on graph embedding problems. *J. ACM*, 64(3):18:1–18:22, 2017. doi:10.1145/3051094.

9 Marek Cygan, Fedor V. Fomin, Łukasz Kowalik, Daniel Lokshtanov, Dániel Marx, Marcin Pilipczuk, Michał Pilipczuk, and Saket Saurabh. *Parameterized Algorithms*. Springer, 2015. doi:10.1007/978-3-319-21275-3.

10 László Egri, Dániel Marx, and Paweł Rzążewski. Finding list homomorphisms from bounded-treewidth graphs to reflexive graphs: a complete complexity characterization. In Rolf Niedermeier and Brigitte Vallée, editors, *35th Symposium on Theoretical Aspects of Computer Science, STACS 2018, February 28 to March 3, 2018, Caen, France*, volume 96 of *LIPIcs*, pages 27:1–27:15. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, 2018. doi:10.4230/LIPIcs.STACS.2018.27.

11 Tomás Feder and Pavol Hell. List homomorphisms to reflexive graphs. *Journal of Combinatorial Theory, Series B*, 72(2):236–250, 1998. doi:10.1006/jctb.1997.1812.

12 Tomás Feder, Pavol Hell, and Jing Huang. List homomorphisms and circular arc graphs. *Combinatorica*, 19(4):487–505, 1999. doi:10.1007/s004939970003.

13 Tomás Feder, Pavol Hell, and Jing Huang. Bi-arc graphs and the complexity of list homomorphisms. *Journal of Graph Theory*, 42(1):61–80, 2003. doi:10.1002/jgt.10073.

14 Fedor V. Fomin, Pinar Heggernes, and Dieter Kratsch. Exact algorithms for graph homomorphisms. *Theory Comput. Syst.*, 41(2):381–393, 2007. doi:10.1007/s00224-007-2007-x.

15 M.R. Garey, D.S. Johnson, and L. Stockmeyer. Some simplified NP-complete graph problems. *Theoretical Computer Science*, 1(3):237–267, 1976. doi:10.1016/0304-3975(76)90059-1.

16 Petr A. Golovach, Daniel Lokshtanov, Saket Saurabh, and Meirav Zehavi. Cliquewidth III: the odd case of graph coloring parameterized by cliquewidth. In Artur Czumaj, editor, *Proceedings of the Twenty-Ninth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2018, New Orleans, LA, USA, January 7-10, 2018*, pages 262–273. SIAM, 2018. doi:10.1137/1.9781611975031.19.

17 Pavol Hell and Jaroslav Nešetřil. *Graphs and homomorphisms*. Oxford University Press, 2004.

18 Pavol Hell and Jaroslav Nešetřil. On the complexity of H-coloring. *J. Comb. Theory, Ser. B*, 48(1):92–110, 1990. doi:10.1016/0095-8956(90)90132-J.

19 Ian Holyer. The NP-completeness of edge-coloring. *SIAM J. Comput.*, 10(4):718–720, 1981. doi:10.1137/0210055.

20 Shenwei Huang. Improved complexity results on k-coloring P_t-free graphs. *Eur. J. Comb.*, 51:336–346, 2016. doi:10.1016/j.ejc.2015.06.005.

21 Russell Impagliazzo and Ramamohan Paturi. On the complexity of k-SAT. *Journal of Computer and System Sciences*, 62(2):367–375, 2001. doi:10.1006/jcss.2000.1727.

Pobrano z http://repo.pw.edu.pl / Downloaded from Repository of Warsaw University of Technology 2021-09-19

22 Russell Impagliazzo, Ramamohan Paturi, and Francis Zane. Which problems have strongly exponential complexity? *J. Comput. Syst. Sci.*, 63(4):512–530, 2001. doi:10.1006/jcss.2001.1774.

23 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. doi:10.1007/978-3-319-57586-5_29.

24 Bart M. P. Jansen. Personal communication.

25 Bart M. P. Jansen and Jesper Nederlof. Computing the chromatic number using graph decompositions via matrix rank. *Theor. Comput. Sci.*, 795:520–539, 2019. doi:10.1016/j.tcs.2019.08.006.

26 Sanjeev Khanna, Nathan Linial, and Shmuel Safra. On the hardness of approximating the chromatic number. *Combinatorica*, 20(3):393–415, 2000. doi:10.1007/s004930070013.

27 Michael Lampis. Finer tight bounds for coloring on clique-width. In Ioannis Chatzigiannakis, Christos Kaklamanis, Dániel Marx, and Donald Sannella, editors, *45th International Colloquium on Automata, Languages, and Programming, ICALP 2018, July 9-13, 2018, Prague, Czech Republic*, volume 107 of *LIPIcs*, pages 86:1–86:14. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2018. doi:10.4230/LIPIcs.ICALP.2018.86.

28 Benoît Larose. Families of strongly projective graphs. *Discuss. Math. Graph Theory*, 22(2):271–292, 2002. doi:10.7151/dmgt.1175.

29 Benoit Larose and Claude Tardif. Strongly rigid graphs and projectivity. *Multiple-Valued Logic*, 7:339–361, 2001.

30 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. doi:10.1145/3170442.

31 Tomasz Łuczak and Jaroslav Nešetřil. Note on projective graphs. *Journal of Graph Theory*, 47(2):81–86, 2004.

32 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. doi:10.4230/LIPIcs.ESA.2020.74.

33 Karolina Okrasa, Marta Piecyk, and Paweł Rzążewski. Full complexity classification of the list homomorphism problem for bounded-treewidth graphs. *CoRR*, abs/2006.11155, 2020. arXiv:2006.11155.

34 Karolina Okrasa and Paweł Rzążewski. Fine-grained complexity of graph homomorphism problem for bounded-treewidth graphs. In *Proceedings of the 2020 ACM-SIAM Symposium on Discrete Algorithms*, pages 1578–1590, 2020. doi:10.1137/1.9781611975994.97.

35 Marta Piecyk and Paweł Rzążewski. Fine-grained complexity of the list homomorphism problem: feedback vertex set and cutwidth. *CoRR*, abs/2009.11642, 2020. arXiv:2009.11642.

36 Neil Robertson and Paul D. Seymour. Graph minors. II. algorithmic aspects of tree-width. *J. Algorithms*, 7(3):309–322, 1986. doi:10.1016/0196-6774(86)90023-4.

37 Paweł Rzążewski. Exact algorithm for graph homomorphism and locally injective graph homomorphism. *Inf. Process. Lett.*, 114(7):387–391, 2014. doi:10.1016/j.ipl.2014.02.012.

38 Magnus Wahlström. New plain-exponential time classes for graph homomorphism. *Theory Comput. Syst.*, 49(2):273–282, 2011. doi:10.1007/s00224-010-9261-z.

39 Virginia Vassilevska Williams. Multiplying matrices faster than Coppersmith-Winograd. In Howard J. Karloff and Toniann Pitassi, editors, *Proceedings of the 44th Symposium on Theory of Computing Conference, STOC 2012, New York, NY, USA, May 19 - 22, 2012*, pages 887–898. ACM, 2012. doi:10.1145/2213977.2214056.

Pobrano z http://repo.pw.edu.pl / Downloaded from Repository of Warsaw University of Technology 2021-09-19