Tight Lower Bounds on Graph Embedding Problems

MAREK CYGAN, Institute of Informatics, University of Warsaw
FEDOR V. FOMIN, University of Bergen; St. Petersburg Department of Steklov Institute of Mathematics of the Russian Academy of Sciences
ALEXANDER GOLOVNEV, New York University; St. Petersburg Department of Steklov Institute of Mathematics of the Russian Academy of Sciences
ALEXANDER S. KULIKOV, St. Petersburg Department of Steklov Institute of Mathematics of the Russian Academy of Sciences
IVAN MIHAJLIN, University of California—San Diego
JAKUB PACHOCKI, Carnegie Mellon University
ARKADIUSZ SOCAŁA, Institute of Informatics, University of Warsaw

18

We prove that unless the Exponential Time Hypothesis (ETH) fails, deciding if there is a homomorphism from graph G to graph H cannot be done in time $|V(H)|^{o(|V(G)|)}$. We also show an exponential-time reduction from Graph Homomorphism to Subgraph Isomorphism. This rules out (subject to ETH) a possibility of $|V(H)|^{o(|V(H)|)}$-time algorithm deciding if graph G is a subgraph of H. For both problems our lower bounds asymptotically match the running time of brute-force algorithms trying all possible mappings of one graph into another. Thus, our work closes the gap in the known complexity of these fundamental problems.

Moreover, as a consequence of our reductions, conditional lower bounds follow for other related problems such as Locally Injective Homomorphism, Graph Minors, Topological Graph Minors, Minimum Distortion Embedding and Quadratic Assignment Problem.

CCS Concepts: • **Theory of computation → Parameterized complexity and exact algorithms**; *Graph algorithms analysis;*

Additional Key Words and Phrases: Lower bounds, graph homomorphism, subgraph isomorphism, graph embedding, exponential time hypothesis

ACM Reference Format:
Marek Cygan, Fedor V. Fomin, Alexander Golovnev, Alexander S. Kulikov, Ivan Mihajlin, Jakub Pachocki, and Arkadiusz Socała. 2017. Tight lower bounds on graph embedding problems. J. ACM 64, 3, Article 18 (June 2017), 22 pages.
DOI: http://dx.doi.org/10.1145/3051094

The research of Alexander Golovnev is supported by NSF grant 1319051. The research of Jakub Pachocki is supported by NSF grant CCF-1065106. The research of Marek Cygan and Arkadiusz Socała is supported by National Science Centre of Poland, Grant Number UMO-2013/09/B/ST6/03136. Preliminary versions of this work were presented at ICALP 2015 and SODA 2016.
The research leading to these results has received funding from the Government of the Russian Federation (grant 14.Z50.31.0030).
Authors' addresses: M. Cygan and A. Socała, Institute of Informatics, Banacha 2, 02-097 Warszawa, Poland; emails: {cygan, as277575}@mimuw.edu.pl; F. Fomin, Department of Informatics, University of Bergen, N-5020 Bergen, Norway; email: fomin@ii.uib.no; A. Golovnev, 251 Mercer St, New York, NY 10021, USA; email: alexgolovnev@gmail.com; A. S. Kulikov, Fontanka 27, 191023 St. Petersburg, Russia; email: alexanders.kulikov@gmail.com; I. Mihajlin, 9500 Gilman Drive, La Jolla, CA 92093-0404 U.S.A; email: imikhail@cs.ucsd.edu; J. Pachocki, Carnegie Mellon School of Computer Science, 5000 Forbes Ave, Pittsburgh, PA 15213, USA; email: meret@seas.harvard.edu.

DOI: http://dx.doi.org/10.1145/3051094

1. INTRODUCTION

We establish tight conditional lower bounds on the complexity of several fundamental graph embedding problems, including Graph Homomorphism, Subgraph Isomorphism, Graph Minor, Topological Graph Minor, and Minimum Distortion Embedding. For given undirected graphs G and H, all these problems can be solved in time $n^{\mathcal{O}(n)}$ by a brute-force algorithm that tries all possible embeddings of G into H, where n is the total number of vertices in G and H. We show that unless the Exponential Time Hypothesis (ETH) fails, the running time $n^{\Omega(n)}$ is unavoidable. This resolves a number of open problems about graph embeddings that can be found in the literature. We start by defining embedding problems and providing for each of the problems a brief overview of the related previous results.

Graph Homomorphism. A *homomorphism* $G \to H$ from an undirected graph G to an undirected graph H is a mapping from the vertex set of G to that of H such that the image of every edge of G is an edge of H. In other words, there is $G \to H$ if and only if there exists a mapping $g : V(G) \to V(H)$, such that for every edge $uv \in E(G)$, we have $g(u)g(v) \in E(H)$. Then the Graph Homomorphism problem HOM(G, H) is defined as follows.

> Graph Homomorphism
> **Input:** Undirected graphs G and H.
> **Task:** Decide whether there is a homomorphism $G \to H$.

Many combinatorial structures in G, for example, cliques, independent sets, and proper vertex colorings, may be viewed as graph homomorphisms to a particular graph H, see Hell and Nešetřil [2004] for a thorough introduction to the topic. It is well known that Coloring is a special case of graph homomorphism. More precisely, a graph G can be colored with at most h colors if and only if $G \to K_h$, where K_h is a complete graph on h vertices. Due to this, very often in the literature HOM(G, H), when $h = |V(H)|$, is referred as H-coloring of G. It was shown by Feder and Vardi [1998] that the Constraint Satisfaction Problem (CSP) can be interpreted as a homomorphism problem on relational structures, and thus Graph Homomorphism encompasses a large family of problems generalizing Coloring but is less general than CSP.

Hell and Nešetřil showed that for every fixed simple graph H, the problem whether there exists a homomorphism from G to H is solvable in polynomial time if H is bipartite and NP-complete if H is not bipartite [Hell and Nešetřil 1990]. Since then, algorithms for and the complexity of graph homomorphisms (and homomorphisms between other discrete structures) have been studied intensively [Austrin 2010; Barto et al. 2008; Grohe 2007; Marx 2010; Raghavendra 2008].

There are two different ways graph homomorphisms are used to extract useful information about graphs. Let us consider two homomorphisms, from a "small" graph F into a "large" graph G and from a "large" graph G into a "small" graph H, which can be represented by the following formula (here we borrow the intuitive description from Lovász [2012]):

$$F \to G \to H.$$

The "left-homomorphisms" from various small graphs F into G are useful to study the local structure of G. For example, if F is a triangle, then the number of "left-homomorphisms" from F into G is the number of triangles in graph G. This type of information is closely related to sampling, and we refer to Lovász [2012], which provides many applications of homomorphisms. "Right-homomorphisms" into "small" different graphs H are related to global properties of graph G.

The trivial brute-force algorithm solving "left-homomorphism" from an f-vertex graph F into an n-vertex graph G runs in time $2^{\mathcal{O}(f \log n)}$: We try all possible vertex subsets of G of size at most f, which is $n^{\mathcal{O}(f)}$, and then for each subset try all possible f^f mappings into it from F. Interestingly, this naïve algorithm is asymptotically optimal. Indeed, as shown by Chen et al. [2006], assuming Exponential Time Hypothesis, there is no $g(k)n^{o(k)}$ time algorithm deciding if an input n-vertex graph G contains a clique of size at least k for any computable function g. Since this is a very special case of Graph Homomorphism with F being a clique of size k, the result of Chen et al. rules out algorithms for Graph Homomorphism of running time $g(f)2^{o(f \log n)}$, from F to G, when the number of vertices f in F is significantly smaller than the number of vertices n in G. Actually, the reduction of Chen et al. [2006] may be slightly adjusted (see the remark right after Theorem 1.1) so a lower bound of the form $2^{\omega(f \log n)}$ follows whenever n is superpolynomial in f, under the Exponential Time Hypothesis.

The interest in "right-homomorphisms" is due to the recent developments in the area of exact exponential algorithms for Coloring and 2-CSP (CSP where all constraints have arity at most 2) problems. The area of exact exponential algorithms is about solving intractable problems significantly faster than the trivial exhaustive search, though still in exponential time [Fomin and Kratsch 2010]. For example, as for Graph Homomorphism, a naïve brute-force algorithm for coloring an n-vertex graph G in h colors is to try for every vertex a possible color, resulting in the running time $\mathcal{O}^*(h^n) = 2^{\mathcal{O}(n \log h)}$.[1] Since h can be of order $\Omega(n)$, the brute-force algorithm computing the chromatic number runs in time $2^{\mathcal{O}(n \log n)}$. It was already observed in 1970s by Lawler [1976] that the brute-force for the Coloring problem can be beaten by making use of dynamic programming over maximal independent sets resulting in single-exponential running time $\mathcal{O}^*((1+\sqrt[3]{3})^n) = \mathcal{O}(2.45^n)$. Almost 30 years later Björklund et al. [2009] succeeded to reduce the running time to $\mathcal{O}^*(2^n)$. And as we observed already, for H-coloring, the brute-force algorithm solving H-coloring runs in time $2^{\mathcal{O}(n \log h)}$. In spite of all the similarities between graph coloring and homomorphism, no substantially faster algorithm was known and it was an open question in the area of exact algorithms if there is a single-exponential algorithm solving H-coloring in time $2^{\mathcal{O}(n+h)}$ [Fomin et al. 2007; Rzążewski 2014; Wahlström 2010, 2011], see also Fomin and Kratsch [2010, Chapter 12].

On the other hand, Graph Homomorphism is a special case of 2-CSP with n variables and domain of size h. It was shown by Traxler [2008] that unless the Exponential Time Hypothesis fails, there is no algorithm solving 2-CSP with n variables and domain of size h in time $h^{o(n)} = 2^{o(n \log h)}$. This excludes (up to ETH) the existence of a single-exponential c^n time algorithm for some constant $c > 1$ for 2-CSP.

Another interesting variant of Graph Homomorphism is related to graph labelings. A homomorphism $f : G \to H$ is called *locally injective* if for every vertex $u \in V(G)$, its neighborhood is mapped injectively into the neighborhood of $f(u)$ in H, that is, if every two vertices with a common neighbor in G are mapped onto distinct vertices in H.

Locally Injective Graph Homomorphism
Input: Undirected graphs G and H.
Task: Decide whether there is a locally injective homomorphism $G \to H$.

As graph homomorphism generalizes graph coloring, locally injective graph homomorphism can be seen as a generalization of graph distance constrained labelings. An

[1] $\mathcal{O}^*(\cdot)$ hides polynomial factors in the input length. Most of the algorithms considered in this article take graphs G and H as an input. By saying that such an algorithm has a running time $\mathcal{O}^*(f(G, H))$, we mean that the running time is upper bounded by $(|V(G)| + |E(G)| + |V(H)| + |E(H)|)^{\mathcal{O}(1)} \cdot f(G, H)$.

$L(2,1)$-labeling of a graph G is a mapping from $V(G)$ into the nonnegative integers such that the labels assigned to vertices at distance 2 differ while labels assigned to adjacent vertices differ by at least 2. This problem was studied intensively in combinatorics and algorithms, see, for example, Griggs and Yeh [1992] and Fiala et al. [2008]. Fiala and Kratochvíl suggested the following generalization of $L(2,1)$-labeling, and we refer the reader to Fiala and Kratochvíl [2008] for the survey. For graphs G and H, an $H(2,1)$-labeling is a mapping $f : V(G) \to V(H)$ such that for every pair of distinct adjacent vertices $u, v \in V(G)$, images $f(u)$ $f(v)$ are distinct and nonadjacent in H. Moreover, if the distance between u and v in G is two, then $f(u) \neq f(v)$. It is easy to see that a graph G has an $L(2,1)$-labeling with maximum label at most k if and only if there is an $H(2,1)$-labeling for H being a k-vertex path. Then the following is known, see, for example, Fiala and Kratochvíl [2008]: There is an $H(2,1)$-labeling of a graph G if and only if there is a locally injective homomorphism from G to the complement of H.

Several single-exponential algorithms for $L(2,1)$-labeling can be found in the literature, the most recent algorithm is due to Junosza-Szaniawski et al. [2013] that runs in time $\mathcal{O}(2.6488^n)$. For $H(2,1)$-labeling, or equivalently for locally injective homomorphisms, single-exponential algorithms were known only for special cases when the maximum degree of H is bounded [Havet et al. 2011] or when the bandwidth of the complement of H is bounded [Rzażewski 2014].

Subgraph Isomorphism. We say that an undirected G is a *subgraph* of H if one can remove some edges and vertices of H, so what remains is isomorphic to G. In other words, G is a subgraph of H if and only if there exists an injective mapping $g : V(G) \to V(H)$, such that for each edge $uv \in E(G)$, $g(u)g(v) \in E(H)$. We define

> Subgraph Isomorphism
> **Input:** Undirected graphs G and H.
> **Task:** Decide whether G is a subgraph of H.

Subgraph Isomorphism is an important and very general problem. Several flagship graph problems can be viewed as instances of Subgraph Isomorphism:

—Hamiltonicity(G): Is C_n (a cycle with n vertices) a subgraph of G?
—Clique(G, k): Is K_k a subgraph of G?
—3-Coloring(G): Is G a subgraph of $K_{n,n,n}$, a tripartite graph with n vertices in each of its three independent sets?
—Bandwidth(G, k): Is G a subgraph of P_n^k (a kth power of an n-vertex path)?

All of the mentioned problems are NP-complete, and the best-known algorithms for all the listed special cases work in exponential time. In fact, all those problems are well studied from the exact exponential algorithms perspective [Beigel and Eppstein 2005; Björklund 2014; Bourgeois et al. 2012; Cygan and Pilipczuk 2012; Feige 2000; Held and Karp 1962; Lawler 1976; Robson 1986; Tarjan and Trojanowski 1977], where the goal is to obtain an algorithm of running time $\mathcal{O}(c^n)$ for the smallest possible value of c. Furthermore, the Subgraph Isomorphism problem was very extensively studied from the viewpoint of fixed parameter tractability, see Marx and Pilipczuk [2014] for a discussion of 19 different possible parametrizations. All the mentioned special cases of Subgraph Isomorphism admit $\mathcal{O}(c^n)$ time algorithms, by using branching, inclusion-exclusion principle, or dynamic programming. On the other hand, a simple exhaustive search for the Subgraph Isomorphism problem—enumerating all possible mappings from the pattern graph to the host graph—runs in $2^{\mathcal{O}(n \log n)}$ time, where n is the total number of vertices of the host graph and pattern graph.

Therefore, a natural question is whether Subgraph Isomorphism admits an $\mathcal{O}(c^n)$ time algorithm. This was repeatedly posed as an open problem [Amini et al. 2012; Cygan et al. 2014; Fomin et al. 2008; Husfeldt et al. 2013]. In particular, in the monograph of Fomin and Kratsch [2010] the existence of $\mathcal{O}(c^n)$ time algorithm for SUBGRAPH ISOMORPHISM was put among the few questions in the open problems section.

SUBGRAPH ISOMORPHISM is a special case of QUADRATIC ASSIGNMENT PROBLEM, which is

> QUADRATIC ASSIGNMENT PROBLEM (QAP)
> **Input:** $n \times n$ matrices $A = (a_{ij})$ and $B = (b_{ij})$ with real entries.
> **Task:** Find a permutation π minimizing $\sum_{i=1}^{n}\sum_{j=1}^{n} a_{\pi(i)\pi(j)} b_{ij}$.

Indeed, G is a subgraph of H if and only if for the instance of QAP with A and B being adjacency matrices of G and the complement of H the optimum value is 0.[2] Problem 7.6 in the influential survey of Woeginger on exact algorithms [Woeginger 2003] is to prove that QAP cannot be solved in time $\mathcal{O}(c^n)$ for any fixed value c (under some reasonable assumption).

GRAPH MINOR. For a graph G and an edge $uv \in G$, we define the operation of *contracting edge uv* as follows: We delete vertices u and v from G and add a new vertex w_{uv} adjacent to all vertices that u or v was adjacent to in G. We say that a graph G is a *minor* of H, if G can be obtained from some subgraph of H by a series of edge contractions. Equivalently, we may say that G is a minor of H if G can be obtained from H itself by a series of edge deletions, edge contractions, and vertex deletions.

> GRAPH MINOR
> **Input:** Undirected graphs G and H.
> **Task:** Decide whether G is a minor of H.

GRAPH MINOR is a fundamental problem in graph theory and graph algorithms. By the theorem of Robertson and Seymour [1995], there exists a computable function f and an algorithm that, for given graphs G and H, checks in time $f(G)|V(H)|^3$ whether G is a minor of H. However, when the size of the graph G is not constant, nothing beyond a brute-force algorithm trying all possible partitions of a vertex set of H was known.

Related notion of graph embedding is the notion of topological minor. We say that a graph G is a subdivision of a graph H if H can be obtained from G by contracting only edges incident with vertices of degree two. In other words, G is obtained from H by replacing edges with paths. A graph G is called a *topological minor* of a graph H if a subdivision of G is isomorphic to a subgraph of H.

> TOPOLOGICAL GRAPH MINOR
> **Input:** Undirected graphs G and H.
> **Task:** Decide whether G is a topological minor of H.

Lingas and Wahlen [2009] gave an algorithm of running time $\mathcal{O}^*(\binom{n}{p}p!2^{n-p})$ solving TOPOLOGICAL GRAPH MINOR for n-vertex graph H and p-vertex graph G.

MINIMUM DISTORTION EMBEDDING. Given an undirected connected graph G with the vertex set $V(G)$ and the edge set $E(G)$, the *graph metric* of G is $M(G) = (V(G), D_G)$, where the distance function D_G is the shortest path distance between u and v for every pair of vertices $u, v \in V(G)$. Given a graph metric M and another metric space M'

[2]If G has smaller number of vertices than H, then it should be first padded with isolated vertices to make the number of vertices in both graphs equal.

with distance functions D and D', a mapping $f : M \to M'$ is called an *embedding* of M into M'. The mapping f is *non-contracting* if for every pair of points p, q in M, $D(p, q) \leq D'(f(p), f(q))$. The *distortion* of embedding f is the minimum number d_f such that $D(p, q) \cdot d_f \geq D'(f(p), f(q))$. We define

> MINIMUM DISTORTION EMBEDDING
> **Input:** Undirected graphs G and H.
> **Task:** Find a non-contracting embedding of G into H of minimum distortion.

Most of exact algorithms for MINIMUM DISTORTION EMBEDDING deal with a special case when the host graph H is a path or a tree of bounded degree [Bădoiu et al. 2005a, 2005b; Cygan and Pilipczuk 2012; Fellows et al. 2013; Fomin et al. 2011; Kenyon et al. 2009]. In particular, an optimal-distortion embedding into a line can be found in time $2^{\mathcal{O}(n)}$ [Cygan and Pilipczuk 2012; Fomin et al. 2011].

Our Results. In this article, we show that from the algorithmic perspective, the behavior of "right-homomorphism" is, unfortunately, much closer to 2-CSP than to COLORING. This result will also imply similar lower bounds for many other graph embedding and containment problems. All lower bounds obtained in this article are conditional, and they hold unless the Exponential Time Hypothesis [Impagliazzo and Paturi 2001; Impagliazzo et al. 2001a] fails. ETH is an established assumption; many interesting lower bounds have been found under this hypothesis (see Cygan et al. [2015] and Lokshtanov et al. [2011] for surveys). We formulate ETH in the next section.

The first main result of this article is the following theorem, which excludes (up to ETH) resolvability of HOM(G, H) in time $2^{o(n \log h)}$, thus resolving the open qestion from Fomin et al. [2007], Rzażewski [2014], Wahlström [2010], and Wahlström [2011].

THEOREM 1.1. *Unless ETH fails, for any constant $D > 0$ there exists a constant $c = c(D) > 0$ such that for any non-decreasing function $3 \leq h(n) \leq n^D$, there is no algorithm solving* GRAPH HOMOMORPHISM *from an n-vertex graph G to a graph H with at most $h(n)$ vertices in time*

$$\mathcal{O}\big(2^{cn \log h(n)}\big). \tag{1}$$

Note that for $h(n) = n$ Theorem 1.1 implies that there is no $2^{\mathcal{O}(n+h)}$ time algorithm for GRAPH HOMOMORPHISM under ETH. Let us remark that to obtain more general results, in all lower bounds proven in this article we assume implicitly that the number h of vertices of the graph H is a function of the number n of the vertices of the graph G. At the same time, to exclude some pathological cases we assume that the function $h(n)$ is "reasonable," meaning that it is non-decreasing and time constructible. Also, it is worth noting that from previous work [Chen et al. 2006; Lokshtanov et al. 2013], it follows that there is a reduction that transforms a given instance of 3-COLORING into deciding whether a graph with $k \cdot 3^{n/k}$ vertices admits a clique of size k (see the proof of Theorem 14.21 in Cygan et al. [2015]). By setting k appropriately to values smaller than $n/\log n$, one can obtain hardness for the cases not covered by Theorem 1.1, that is, when $h(n)$ is superpolynomial, consequently covering the whole spectrum of $h(n)$.

With a tiny modification, the proof of Theorem 1.1 can be adapted to show a similar lower bound for LOCALLY INJECTIVE GRAPH HOMOMORPHISM.

THEOREM 1.2. *Unless ETH fails, for any constant $D > 0$ there exists a constant $c = c(D) > 0$ such that for any non-decreasing function $3 \leq h(n) \leq n^D$, there is no algorithm deciding if there is a locally injective homomorphism from an n-vertex graph G to a graph H with at most $h(n)$ vertices in time $\mathcal{O}(2^{cn \log h(n)})$.*

The second main result of this article is about SUBGRAPH ISOMORPHISM, resolving the open question asked in Amini et al. [2012], Cygan et al. [2014], Fomin et al. [2008], Fomin and Kratsch [2010], and Husfeldt et al. [2013].

THEOREM 1.3. *Unless ETH fails, there is no algorithm solving* SUBGRAPH ISOMORPHISM *for graphs G and H in time $2^{o(n \log n)}$, where $n = |V(G)| = |V(H)|$.*

Theorem 1.3 implies that QAP cannot be solved in time $2^{o(n \log n)}$ unless ETH fails and hence provides the answer to the open problem of Woeginger [2003].

An important feature of our proof is that it rules out solvability of SUBGRAPH ISOMORPHISM in time $2^{o(n \log n)}$ even for the special case when $|V(G)| = |V(H)| = n$, since in this special case a graph G is a (topological) minor of H if and only if G is a subgraph of H. Thus the case of GRAPH MINOR and TOPOLOGICAL GRAPH MINOR when $|V(G)| = |V(H)| = n$ cannot be resolved in time $2^{o(n \log n)}$ as well. Similar arguments work for various modifications of GRAPH MINOR like SHALLOW GRAPH MINOR, and so on.

To see how the bound on SUBGRAPH ISOMORPHISM yields the bound on MINIMUM DISTORTION EMBEDDING, we observe that an n-vertex graph G admits a non-contracting embedding of distortion 1 into an n-vertex graph H if and only if H is a subgraph of G.

Methods. To establish lower bounds for graph homomorhisms, we proceed in two steps. First, we obtain lower bounds for LIST GRAPH HOMOMORPHISM by reducing to it the 3-coloring problem on graphs of bounded degree. More precisely, for a given graph G with vertices of small degrees, we construct an instance (G', H') of LIST GRAPH HOMOMORPHISM, such that G is 3-colorable if and only if there exists a list homomorphism from G' to H'. Moreover, our construction guarantees that a "fast" algorithm for list homomorphism implies an algorithm for 3-coloring violating ETH. The reduction is based on a "grouping" technique; however, to do the required grouping, we need a trick exploiting the condition that G has a bounded maximum vertex degree and thus can be colored in a bounded number of colors in polynomial time. In the second step of reductions, we proceed from list homomorphisms to normal homomorphisms. Here we need specific gadgets with a property that any homomorphism from such a graph to itself preserves an order of its specific structures.

The remaining part of the article is organized as follows. Section 2 contains all necessary definitions. In Section 3 we give technical lemmata and reductions that are used to prove lower bounds for the GRAPH HOMOMORPHISM problem in Section 4.1 and for the SUBGRAPH ISOMORPHISM in Section 4.2. We conclude with some open problems in Section 5.

2. PRELIMINARIES

Graphs. We consider simple undirected graphs, where $V(G)$ denotes the set of vertices and $E(G)$ denotes the set of edges of a graph G. For a given subset S of $V(G)$, $G[S]$ denotes the subgraph of G induced by S, and $G - S$ denotes the graph $G[V(G) \setminus S]$. A vertex set S of G is an *independent set* if $G[S]$ is a graph with no edges, and S is a *clique* if $G[S]$ is a complete graph. The set of neighbors of a vertex v in G is denoted by $N_G(v)$, and the set of neighbors of a vertex set S is $N_G(S) = \bigcup_{v \in S} N_G(v) \setminus S$. By $N_G[S]$ we denote the closed neighborhood of the set S, that is, the set S together with all its neighbors: $N_G[S] = S \cup N_G(S)$. For an integer n, we use $[n]$ to denote the set of integers $\{1, \ldots, n\}$.

The complete graph on k vertices is denoted by K_k. A *coloring* of a graph G is a function assigning a color to each vertex of G such that adjacent vertices have different colors. A k-coloring of a graph uses at most k colors, and the *chromatic number* $\chi(G)$ is the smallest number of colors in a coloring of G.

Throughout the article, we implicitly assume that there is a total order on the set of vertices of a given graph. This allows us to treat a k-coloring of an n-vertex graph simply as a vector in $[k]^n$.

Let G be an n-vertex graph, $1 \leq r \leq n$ be an integer, and $V(G) = B_1 \sqcup B_2 \sqcup \ldots \sqcup B_{\lceil \frac{n}{r} \rceil}$ be a partition of the set of vertices of G. Then *the grouping* of G with respect to the partition $V(G) = B_1 \sqcup B_2 \sqcup \ldots \sqcup B_{\lceil \frac{n}{r} \rceil}$ is a graph G_r with vertices $B_1, \ldots, B_{\lceil \frac{n}{r} \rceil}$ such that B_i and B_j are adjacent if and only if there exist $u \in B_i$ and $v \in B_j$ such that $uv \in E(G)$. To distinguish vertices of the graphs G and G_r, the vertices of G_r will be called *buckets*.

For a graph G, its *square* G^2 has the same set of vertices as G and $uv \in E(G^2)$ if and only if there is a path of length at most 2 between u and v in G (thus, $E(G) \subseteq E(G^2)$). It is easy to see that if the degree of G is at most d then the degree of G^2 is at most d^2.

Homomorphisms and List Homomorphisms. Let G and H be graphs. A mapping $\varphi : V(G) \to V(H)$ is a *homomorphism* if for every edge $uv \in E(G)$ its image $\varphi(u)\varphi(v) \in E(H)$. If there exists a homomorphism from G to H, then we often write $G \to H$. The Graph Homomorphism problem HOM(G, H) asks whether $G \to H$.

Assume that for each vertex v of G we are given a list $\mathcal{L}(v) \subseteq V(H)$. A *list homomorphism* of G to H, also known as a list H-coloring of G, with respect to the lists $\mathcal{L}$, is a homomorphism $\varphi : V(G) \to V(H)$, such that $\varphi(v) \in \mathcal{L}(v)$ for all $v \in V(G)$. The List Graph Homomorphism problem LIST-HOM(G, H) asks whether graph G with lists $\mathcal{L}$ admits a list homomorphism to H with respect to $\mathcal{L}$.

Exponential Time Hypothesis. Our lower bounds are based on a well-known complexity hypothesis formulated by Impagliazzo et al. [2001b].

Exponential Time Hypothesis (ETH): There is a constant $q > 0$ such that 3-CNF-SAT with n variables and m clauses cannot be solved in time $2^{qn}(n+m)^{\mathcal{O}(1)}$.

This hypothesis is widely applied in the theory of exact exponential algorithms, and we refer to Cygan et al. [2015] and Lokshtanov et al. [2013] for an overview of ETH and its implications.

In this article, we use the following well-known application of ETH with respect to 3-Coloring (see, e.g., Theorem 3.2 in Lokshtanov et al. [2013], and Exercise 7.27 in Sipser [2005]). The 3-Coloring problem is the problem to decide whether the given graph can be properly colored in three colors.

Proposition 2.1. *Unless ETH fails, there exists a constant $q > 0$ such that* 3-Coloring *on n-vertex graphs of average degree four cannot be solved in time $\mathcal{O}^*(2^{qn})$.*

It is well known that 3-Coloring remains NP-complete on graphs of maximum vertex degree four. Moreover, the classical reduction, see, for example, Garey and Johnson [1979], allows for a given n-vertex graph G to construct a graph G' with maximum vertex degree at most four and $|V(G')| = \mathcal{O}(|E(G)|)$ such that G is 3-colorable if and only if G' is. Thus Proposition 2.1 implies the following (folklore) lemma that will be used in our proofs.

Lemma 2.2. *Unless ETH fails, there exists a constant $q > 0$ such that there is no algorithm solving* 3-Coloring *on n-vertex graphs of maximum degree four in time $\mathcal{O}^*(2^{qn})$.*

3. AUXILIARY LEMMATA

In this section, we provide reductions and auxiliary lemmata about colorings that will be used to prove lower bounds for Graph Homomorphism and Subgraph Isomorphism.

3.1. Balanced Colorings

To prove slightly superexponential lower bound based on the ETH, we need a slightly sublinear reduction. From Lemma 2.2, we know that 3-COLORING requires exponential time even for graphs of maximum degree four. Given a graph G that needs to be 3-colored, our general strategy is to partition the set of vertices of G into buckets of size roughly $\epsilon \log n$ (for some constant $0 < \epsilon < 1$). This way the number of buckets is $\mathcal{O}(n/\log n)$, where $n = |V(G)|$. Observe that each bucket has only $3^{\epsilon \log n} = n^{\mathcal{O}(\epsilon)}$ valid 3-colorings. Consequently, we may create graphs G', H', such that vertices of G' correspond to buckets of G, whereas vertices of H' correspond to potential 3-colorings of a bucket, obtaining a sublinear reduction as $|V(G')| = \mathcal{O}(n/\log n)$ and $|V(H')| = n^{\mathcal{O}(\epsilon)}$. Unfortunately, in our reduction we are unable to implement this strategy for all bucketings. In particular, we are unable to verify consistency of colorings between buckets if there is more than one edge between them. For this reason, in this section we will prove two lemmata that will be used to construct a very particular bucketing of G, suitable for reductions presented in Section 3.2.

In the following, we show how to construct a specific "balanced" coloring of a graph in polynomial time. Let G be a graph of constant maximum degree. The coloring of G we want to construct should satisfy three properties. First, it should be a proper coloring of G^2. Then the size of each color class should be bounded as well as the number of edges between vertices from different color classes. More precisely, we prove the following lemma.

LEMMA 3.1. *For any constant $d \geq 1$, there exist constants $\alpha, \beta, \tau > 1$ and a polynomial time algorithm that for a given graph G on n vertices of maximum degree d and an integer $\tau \leq L \leq \frac{n}{4d^2}$, finds a coloring $c : V(G) \to [L]$ satisfying the following properties:*

(*i*) *The coloring c is a proper coloring of G^2.*

(*ii*) *There are only a few vertices of each color: for all $i \in [L]$,*

$$|c^{-1}(i)| \leq \left\lceil \alpha \cdot \frac{n}{L} \right\rceil . \tag{2}$$

(*iii*) *There are only a few edges of G between each pair of colors: For all $i \neq j \in [L]$, we have*

$$k_{i,j} := |\{uv \in E(G) : c(u) = i, c(v) = j\}| \leq K_{i,j} := \left\lceil \beta \cdot \frac{\min\{|c^{-1}(i)|, |c^{-1}(j)|\}}{L} \right\rceil .$$

PROOF. The algorithm starts by constructing greedily an independent set I of G^2 of size $\lceil \frac{n}{d^2+1} \rceil$. Since the maximum vertex degree of G^2 does not exceed d^2, this is always possible. We construct a partial coloring of G^2 by coloring the vertices of I in L colors such that the obtained coloring is a balanced coloring of $G^2[I]$, meaning that the number of vertices of each color is $\lfloor |I|/L \rfloor$ or $\lceil |I|/L \rceil$. Since I is an independent set in G^2, such a coloring can be easily constructed in polynomial time. In the obtained partial equitable coloring, we have that for every $i \in [L]$

$$|c^{-1}(i)| \geq \left\lfloor \frac{n}{L(d^2+1)} \right\rfloor \geq \left\lfloor \frac{n}{2Ld^2} \right\rfloor \geq \frac{n}{4Ld^2}, \tag{3}$$

because $L \leq \frac{n}{4d^2}$ and $\lfloor x \rfloor \geq x/2$ for any $x \geq 2$. Let us note that the obtained precoloring of G^2 clearly satisfies the first and the third conditions of the lemma. Since the size of every $c^{-1}(i)$, $i \in [L]$, does not exceed $|c^{-1}(i)| \leq \lceil \frac{n}{L} \rceil$, the second condition of the lemma also holds for every $\alpha \geq 1$.

We extend the precoloring of G^2 to the required coloring by the following greedy procedure: We select an arbitrary uncolored vertex v and color it by a color from $[L]$

such that the new partial coloring also satisfies the three conditions of the lemma. In what follows, we prove that such a greedy choice of a color is always possible.

Coloring of a vertex v with a color i can be forbidden only because it breaks one of the three conditions. Let us count, how many colors can be forbidden for v by each of the three constraints.

(i) Vertex v has at most d^2 neighbors in G^2, so the first constraint forbids at most d^2 colors.
(ii) The second constraint forbids all the colors that are "fully packed" already. The number of such colors is at most $\frac{n}{(\frac{\alpha n}{L})} = \frac{L}{\alpha}$.
(iii) To estimate the number of colors forbidden by the third condition, we go through all the neighbors of v. A neighbor $u \in N_G(v)$ forbids a color i if coloring v by i exceeds the allowed bound on $k_{i,c(u)}$. Hence to estimate the number of such forbidden colors i (for every fixed vertex u) we need to estimate how many values of $k_{i,c(u)}$ can reach the allowed upper bound $K_{i,c(u)}$. We have that

$$|\{i : k_{i,c(u)} = K_{i,c(u)}\}| \overset{\text{by}\,(3)}{\leq} \left|\left\{i : k_{i,c(u)} \geq \frac{\beta n}{4L^2d^2}\right\}\right| = \left|\left\{i : k_{i,c(u)} \cdot \frac{4L^2d^2}{\beta n} \geq 1\right\}\right|$$
$$\leq \sum_{i \in [L]} k_{i,c(u)} \cdot \frac{4L^2d^2}{\beta n}.$$

The number of edges between vertices of the same color $c(u)$ and all other vertices of the graph does not exceed the cardinality of the color class $c(u)$ times d. Thus we have

$$\sum_{i \in [L]} k_{i,c(u)} \cdot \frac{4L^2d^2}{\beta n} \leq d|c^{-1}(c(u))| \cdot \frac{4L^2d^2}{\beta n} \overset{\text{by}\,(2)}{\leq} d\left\lceil \frac{\alpha n}{L} \right\rceil \cdot \frac{4L^2d^2}{\beta n}$$
$$\leq d\frac{2\alpha n}{L} \cdot \frac{4L^2d^2}{\beta n} = \frac{8\alpha L d^3}{\beta},$$

where the last inequality is due to $\alpha > 1$ and $L \leq n$. Therefore,

$$|\{i : k_{i,c(u)} = K_{i,c(u)}\}| \leq \frac{8\alpha L d^3}{\beta}.$$

Since the degree of v in G does not exceed d, we have that the number of colors forbidden by the third constraint is at most $\frac{8\alpha L d^4}{\beta}$.

Thus, the total number of colors forbidden by all three constraints for the vertex v is at most

$$d^2 + \frac{L}{\alpha} + \frac{8\alpha L d^4}{\beta}.$$

By taking sufficiently large constants α, β, and τ, say, $\alpha = 4$, $\beta = 32\alpha d^4$, and $\tau = 4d^2$, we may upper bound this expression by $L/4 + L/4 + L/4 = \frac{3L}{4}$, which is guaranteed not to exceed $L - 1$ for every $L \geq \tau \geq 4$. Therefore, there always exists a vacant color for the vertex v, which concludes the proof. □

We would like to note that the properties of Lemma 3.1 cannot be improved significantly, as on average a color class has n/L vertices, while the average number of edges between two color classes is roughly $dn/L^2 = d \cdot \frac{n/L}{L}$.

Now with help of Lemma 3.1, we describe a way to construct a specific grouping of a graph. The properties of such groupings are crucial for the final reduction.

LEMMA 3.2. *For any constant $d \geq 1$, there exist positive integers $\lambda = \lambda(d)$, $n_0 = n_0(d)$ and a polynomial time algorithm that for a given graph G on $n \geq n_0$ vertices of maximum degree d and a positive integer $r \leq \sqrt{\frac{n}{2\lambda}}$, finds a grouping $\tilde{G}$ of G and a coloring $\tilde{c} : V(\tilde{G}) \to [\lambda r]$ such that*

(*i*) *The number of buckets of $\tilde{G}$ is*

$$|V(\tilde{G})| \leq \frac{|V(G)|}{r}\,;$$

(*ii*) *The coloring $\tilde{c}$ is a proper coloring of $\tilde{G}^2$;*

(*iii*) *Each bucket $B \in V(\tilde{G})$ is an independent set in G, that is, for every $u, v \in B$, $uv \notin E(G)$;*

(*iv*) *For every pair of buckets $B_1, B_2 \in V(\tilde{G})$ there is at most one edge between them in G, that is,*

$$|\{uv \in E(G) : u \in B_1, v \in B_2\}| \leq 1\,.$$

PROOF. Let $\beta = \beta(d)$, $\tau = \tau(d)$ be constants provided by Lemma 3.1 and let $L = \lambda r$ for $\lambda = \lambda(d) = \max(\lceil\tau\rceil, \lceil 2d\beta\rceil)$. Clearly, $L \geq \tau$. To ensure $L \leq n/(4d^2)$, and consequently satisfy the prerequisites of Lemma 3.1 for any $r \leq \sqrt{n/(2\lambda)}$, we set $n_0 = 8\lambda d^4$ and assume $n \geq n_0$, as then

$$L = \lambda r \leq \sqrt{\frac{n\lambda}{2}} = n \cdot \sqrt{\frac{\lambda}{2n}} \leq n \cdot \sqrt{\frac{\lambda}{2 \cdot 8\lambda d^4}} = \frac{n}{4d^2}\,.$$

Let also c be a coloring of G in L colors provided by Lemma 3.1. We want to construct a grouping $\tilde{G}$ of G such that for all buckets $B \in V(\tilde{G})$ and all $u \neq v \in B$,

$$\begin{gathered} c(u) = c(v) \text{ and } c(u') \neq c(v') \\ \text{for all } u' \in N_G(u),\, v' \in N_G(v). \end{gathered} \tag{4}$$

In other words, all vertices of the same bucket are of the same color while any two neighbors of such two vertices are of different colors.

For each color $i \in [L]$, we introduce an auxiliary constraint graph F_i. The vertex set of F_i is $V(F_i) = c^{-1}(i)$ and its edge set is

$$E(F_i) = \{uv : \exists u' \in N_G(u),\, v' \in N_G(v),\, c(u') = c(v')\}.$$

In our construction, each bucket of $\tilde{G}$ will be an independent set in some F_i. Note that this will immediately imply Equation (4). The degree of any vertex $v \in V(F_i)$ is at most

$$\deg_{F_i}(v) \leq \sum_{v' \in N_G(v)} \left(K_{c(v),c(v')} - 1\right) \leq d\left(\left\lceil \frac{\beta|c^{-1}(c(v))|}{L} \right\rceil - 1\right) \leq \frac{d\beta|V(F_i)|}{L} \leq \frac{|V(F_i)|}{2r}\,,$$

where the last inequality follows from $L \geq \lceil 2d\beta \rceil r \geq 2d\beta r$. This means that the greedy algorithm finds a proper coloring of each F_i in at most $\frac{|V(F_i)|}{2r} + 1$ colors, which splits each F_i in at most $\frac{|V(F_i)|}{2r} + 1$ independent sets. We create a separate bucket of $\tilde{G}$ from each independent set of each F_i. Now we show that the four conditions from the lemma statement hold.

(i) For the first property, the number of independent sets in each F_i is at most $\frac{|V(F_i)|}{2r}+1$. Thus the number of buckets in $\tilde{G}$ is

$$|V(\tilde{G})| \leq \sum_{i\in[L]} \left(\frac{|V(F_i)|}{2r}+1\right) = \sum_{i\in[L]} \left(\frac{|c^{-1}(i)|}{2r}+1\right) = \frac{n}{2r} + L \leq \frac{n}{r},$$

since $L = \lambda r$ and $2\lambda r^2 \leq n$.

(ii) For the second property, by Lemma 3.1, the coloring c is proper in G^2. We can convert c to a coloring $\tilde{c} : V(\tilde{G}) \to [\lambda r]$ by assigning each bucket the color of its vertices (all of them have the same color). The resulting coloring $\tilde{c}$ is a proper coloring of $\tilde{G}^2$ by Equation (4) and the fact that c is proper in G^2.

(iii) All buckets of $\tilde{G}$ are monochromatic with respect to c, thus, each bucket $B \in V(\tilde{G})$ is an independent set in G and the third property holds.

(iv) Finally, by Equation (4), there is at most one edge in G between vertices corresponding to any pair of buckets in $\tilde{G}$.

Thus, the constructed grouping and its coloring satisfy all conditions of the lemma. □

We would like to note that the properties of Lemma 3.2 do not upper bound the size of each bucket explicitly. However, for any graph without isolated vertices we may upper bound the size of each bucket as follows. Consider a bucket B. By property (iii), we know that B is an independent set, and if there are no isolated vertices in the graph, then each vertex of B has at least one incident edge going outside of B. By property (iv) each of those edges goes to a different bucket, and by property (ii) each of those buckets has a different color. Consequently, the number of vertices of a bucket is not greater than the total number of colors being λr.

3.2. Reductions

This section constitutes the main technical part of the article and contains all the necessary reductions used in the lower bounds proofs. Using these reductions as building blocks, the lower bounds follow from careful calculations. The general pipeline is as follows. To prove a lower bound, we take a graph G of maximum degree four that needs to be 3-colored and construct an equisatisfiable instance (G', H') of List Graph Homomorphism using Lemma 3.3 . We then use Lemma 3.4 to transform (G', H') into an equisatisfiable instance (G'', H'') of Graph Homomorphism. Thus, an algorithm checking whether there exists a homomorphism from G'' to H'' can be used to check whether the initial graph G can be 3-colored. At the same time, we know a lower bound for 3-Coloring under ETH (Lemma 2.2). This gives us a lower bound for Graph Homomorphism under the ETH assumption (Section 4.1). To prove the hardness of Subgraph Isomorphism in Section 4.2, we show an exponential-time reduction from Graph Homomorphism to Subgraph Isomorphism.

Lemma 3.3 (3-Coloring → List Graph Homomorphism). *There exists an algorithm that takes as input a graph G on n vertices of maximum degree d that needs to be 3-colored and an integer $r = o(\sqrt{n})$ and finds an equisatisfiable instance (G', H') of LIST-HOM, where $|V(G')| \leq n/r$ and $|V(H')| \leq \gamma(d)^r$, where $\gamma(d)$ is a function of the graph degree. The running time of the algorithm is polynomial in n and the size of the output graphs.*

Proof. *Constructing the graph G'.* Let G' be the grouping of G and $c : V(G') \to [L]$ be the coloring provided by Lemma 3.2, where $L = \lambda(d)r$. To distinguish colorings of G and G', we call $c(B)$, for a bucket $B \in V(G')$, a *label* of B. Consider a bucket $B \in V(G')$, that is, a subset of vertices of G, and a label $i \in [L]$. From item (ii) of Lemma 3.2, we know that c is a proper coloring of $(G')^2$. This, in particular, means that there is at most one $B' \in N_{G'}(B)$ such that $c(B') = i$. Moreover, if such B' exists then, by item (iv) of Lemma 3.2, there exists a unique $u \in B$ and unique $u' \in B'$ such that $uu' \in E(G)$. This allows us to define the following mapping $\phi_B : [L] \to B \cup \{0\}$: $\phi_B(i) = u$ if such B' exists and $\phi_B(i) = 0$ if B has no neighbor B' of label i. Without loss of generality we assume that G does not have isolated vertices. Since each vertex has a neighbor outside its bucket (it cannot have a neighbor in its own bucket, as buckets are independent), $B \subseteq \phi_B(L)$.

Constructing the graph H'. We now define a redundant encoding of a 3-coloring of a bucket $B \in V(G')$. Namely, let $\mu_B : (f : B \to \{1, 2, 3\}) \to \{0, 1, 2, 3\}^L$. That is, for a 3-coloring $f : B \to \{1, 2, 3\}$ of B, $\mu_B(f)$ is a vector v of length L. For $i \in [L]$, by $v[i]$ we denote the ith component of v. The value of $v[i]$ is defined as follows: If $\phi_B(i) = 0$, then $v[i] = 0$, otherwise $v[i] = f(\phi_B(i))$. In other words, for a given bucket B and a 3-coloring f of its vertices, for each possible label $i \in [L]$, $\mu_B(f)[i]$ is the color of the vertex $u \in B$ that has a neighbor in a bucket with label i and 0 if there is no such vertex u.

We are now ready to construct the graph H'. The set of vertices of H' is defined as follows:

$$V(H') = \{(R, l) : R \in \{0, 1, 2, 3\}^L \text{ and } l \in [L]\}.$$

Intuitively, a vertex of H' is a pair consisting of a redundant encoding of a 3-coloring of a bucket and a label of a bucket. By using list constraints, we will ensure that $R = \mu_B(f)$ for some 3-coloring f of B, where B is a bucket mapped to (R, l).

Formally, the list constraints of this instance of List Graph Homomorphism are defined as follows: a bucket $B \in V(G')$ is allowed to be mapped to $(R, l) \in V(H')$ if and only if $l = c(B)$, and there is a 3-coloring f of B such that $\mu_B(f) = R$. Informally, two vertices in $V(H')$ are joined by an edge if they define two consistent 3-colorings. Formally, $(R_1, l_1)(R_2, l_2) \in E(H')$ if and only if $R_1[l_2] \neq R_2[l_1]$. Observe that $|V(G')| \leq n/r$ by Lemma 3.2 and $|V(H')| \leq 4^L \cdot L \leq 4^L \cdot 2^L = 8^{\lambda(d)r} = \gamma(d)^r$ for $\gamma(d) = 8^{\lambda(d)}$.

In the described construction, it might happen that we add an edge to H' even if $R_1[l_2] = 0$ or $R_2[l_1] = 0$. We would like to note that this is, however, irrelevant to the following proof of correctness: If the edges would be added to H' only when $R_1[l_2], R_2[l_1] \neq 0$, then the same reasoning would also work.

Running time of the reduction. The reduction clearly takes time polynomial in the size of input and output.

Correctness of the reduction. It remains to show that G is 3-colorable if and only if (G', H') is a yes-instance of List Graph Homomorphism.

Assume that G is 3-colorable and take a proper 3-coloring g of G. It defines a homomorphism from G' to H' in a natural way: $B \in V(G')$ is mapped to $(\mu_B(g|_B), c(B))$, where $g|_B$ is the function g with its domain restricted to B. Each list constraint is satisfied by definition. To show that each edge is mapped to an edge, consider an edge $BB' \in E(G')$. Then, by item (iv) of Lemma 3.2, there is a unique edge $uu' \in E(G)$ such that $u \in B, u' \in B'$. Note that B and B' are mapped to vertices (R, l) and (R', l') such that $R[l'] = g(u)$ and $R'[l] = g(u')$, since g is a proper 3-coloring of G, $g(u) \neq g(u')$. This, in turn, means that $(R, l)(R', l') \in E(H')$, and hence the edge BB' is mapped to this edge in H'.

For the reverse direction, consider a homomorphism $h : G' \to H'$. For each bucket $B \in V(G')$, $h(B)$ defines a proper 3-coloring of B, that is, $\mu_B^{-1}(h(B))$. Note that $\mu_B^{-1}(h(B))$ is well defined as μ_B is injective due to the list constraints and the assumption that G has no isolated vertices. Together, they define a 3-coloring g of G and we need to show that g is proper. Assume, to the contrary, that there is an edge $uu' \in E(G)$ such that $g(u) = g(u')$. By item (iii) of Lemma 3.2, u and u' belong to different buckets $B, B' \in V(G')$. By the definition of grouping, $BB' \in E(G')$. Since h is a homomorphism,

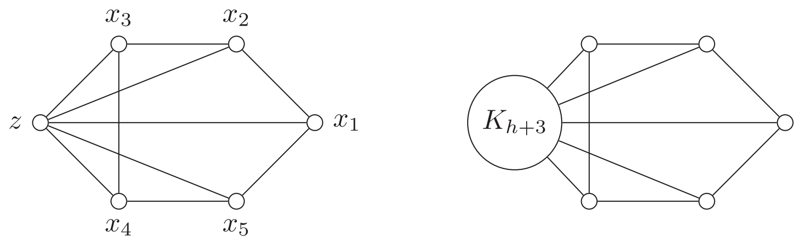

Fig. 1. The graphs D' (left) and D (right). The encircled clique K_{h+3} is the canonical clique of D. An edge from a clique to a vertex of a cycle means that each vertex of the clique is joined to this vertex.

$(R, l)(R', l') := h(B)h(B') \in E(H')$. At the same time, $R[l'] = g(u) = g(u') = R'[l]$, which contradicts the fact that $(R, l)(R', l')$ is an edge in H'. □

In the following lemma, we show a reduction from LIST-HOM to HOM, but before formally proving the lemma, let us discuss the approach we take. Let (G, H) be an instance of LIST-HOM equipped with lists $\mathcal{L} : V(G) \to 2^{V(H)}$. A natural idea could be as follows. Create graphs $G' = G \cup X$ and $H' = H \cup X$, where $X = \{v' : v \in V(H)\}$, that is add to both graphs G and H a set X of $|V(H)|$ vertices. If we could guarantee that any homomorphism from G' to H' maps each v' of G' to its corresponding copy v' of H', then we could emulate the list constraints $\mathcal{L}$ by appropriate connections with X:

—connect each vertex $x \in V(G)$ to $\{v' : v \in V(H) \setminus \mathcal{L}(x)\}$ in G', that is, connect it to the copies of forbidden vertices,
—connect each vertex $v \in V(H)$ to $X \setminus \{v'\}$ in H', that is, connect it to all the vertices of X except its copy.

Now for a homomorphism from G' to H' mapping bijectively X to X, where the bijection is an identity function, a vertex $x \in V(G)$ cannot be mapped to vertex $v \in V(H)$ with $v \notin \mathcal{L}(x)$, as the edge xv of G' would not be mapped to an edge of H'.

To implement the above idea, we construct *anchors*, which can be only mapped to themselves by any homomorphism.

LEMMA 3.4 (LIST GRAPH HOMOMORPHISM → GRAPH HOMOMORPHISM). *There is a polynomial-time algorithm that from an instance (G, H) of LIST-HOM where $|V(G)| = n$, $|V(H)| = h \geq 3$ constructs an equisatisfiable instance (G', H') of HOM where $|V(G')| \leq n + \Delta$, $|V(H')| \leq \Delta$ for $\Delta = 25h^2$.*

PROOF. *Preparations.* We start with a simple six-vertex gadget D' consisting of a 5-cycle together with an apex vertex adjacent to all the vertices of the cycle, see Figure 1.

An important property of D' is that for each homomorphism $\phi : D' \to D'$ and $i \in [5]$,

$$\phi(z) = z \text{ and } \phi(z) \neq \phi(x_i),$$

which means that z is always mapped to z and nothing else is mapped to z. Indeed, because the vertex z is adjacent to all the remaining vertices of D', we have that $\phi(z) \neq \phi(x_i)$ as otherwise $\phi(z)$ would not be adjacent to $\phi(x_i)$, since D' contains no self-loops. For the same reason, we have that for every $i \in [5]$, $\phi(x_i) \in N_{D'}(\phi(z))$. But for every x_i, its open neighborhood $N_{D'}(x_i)$ induces a bipartite graph. On the other hand, the chromatic number of the cycle $C = x_1x_2x_3x_4x_5$ is 3, and thus it cannot be mapped by ϕ to $N_{D'}(x_i)$ for any $i \in [5]$. Therefore, $\phi(z) = z$.

In order for the $\phi(z) = z$ argument to work in a bigger graph, we replace z by a clique K_{h+3} of size $h + 3$, called the *canonical clique* of the gadget. The obtained graph with $(h + 3) + 5$ vertices is denoted as D (see Figure 1).

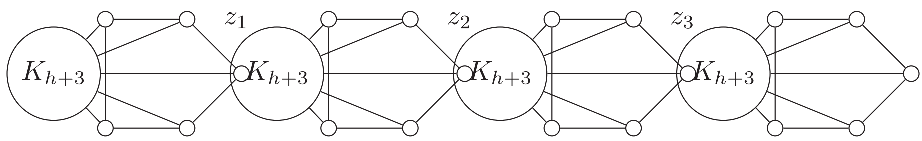

Fig. 2. The gadget T_k for $k = 3$.

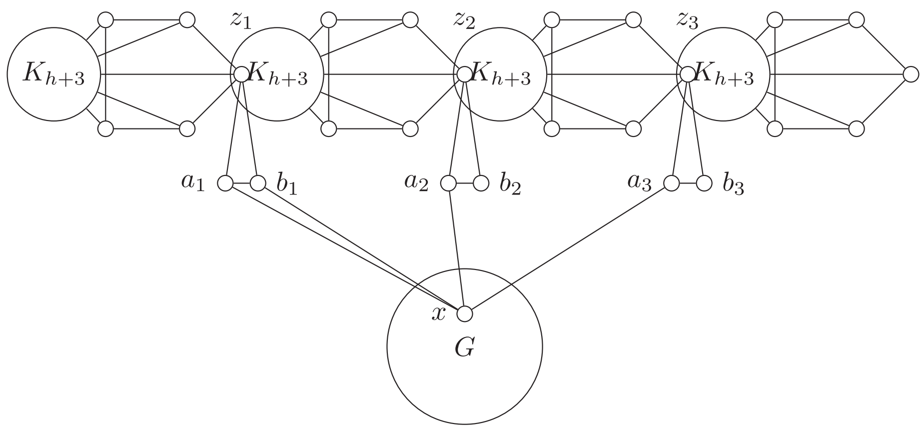

Fig. 3. The graph G'. A vertex $x \in V(G)$ is connected to b_j if and only if $v_j \notin \mathcal{L}(x)$, where $\mathcal{L}(x)$ is the list associated with the vertex $x \in V(G)$ and $V(H) = \{v_1, \ldots, v_h\}$.

Let $D_0, \ldots, D_k$ be $k + 1$ copies of the graph D. We join those $k + 1$ graphs isomorphic to D to construct a larger gadget T_k as follows (see Figure 2). For each $i \in [k]$, we select an arbitrary vertex from the canonical clique of D_i, denote this vertex as z_i, and identify it with one arbitrary vertex of D_{i-1} that does not belong to the canonical clique of D_{i-1}, that is, with a vertex of the 5-cycle (see Figure 2). Denote the new graph by T_k. Observe that each D_i is a block of T_k, and we call D_i the ith block of T_k. Note that two consecutive blocks D_{i-1} and D_i have exactly one common vertex, namely z_i.

The reason we are using those canonical cliques instead of single vertices in the construction of T_k is that those canonical cliques are big enough to behave as anchors. That is, we will prove that canonical cliques can only be mapped to themselves and not to other parts of the graph, in particular, for each $i \in [k]$ and homomorphism $\phi : T_k \to T_k$, $\phi(z_i) = z_i$.

Constructing G'. As we take into consideration list constraints, let us denote $V(H) = \{v_1, \ldots, v_h\}$. Let A_h be a graph consisting of a matching with h edges $\{a_1b_1, \ldots, a_hb_h\}$. Then the graph G' consists of a copy of G, a copy of T_h, and a copy of A_h with the following additional edges: The vertex z_i from the ith block of T_h is adjacent to the vertices a_i and b_i. Also we add edges from G to A_h: For a vertex $x \in V(G)$, we add an edge xa_j for every j, and an edge xb_j if $v_j \notin \mathcal{L}(x)$ (see Figure 3). The number of vertices in G' is at most $n + 2h + (h + 1)(h + 3 + 5) \le n + (h + 1)(h + 11) \le n + 25h^2$.

Constructing H'. Recall we denote $V(H) = \{v_1, \ldots, v_h\}$. The graph H' is constructed similarly as G'. It consists of a copy of H, a copy of T_h, and a copy of A_h. For every i, we add edges z_ia_i and z_ib_i as before. Also, each vertex v_i of H is adjacent to all the vertices from A_h except for b_i (see Figure 4). The number of vertices in H' is at most $h + 2h + (h + 1)(h + 3 + 5) \le (h + 1)(h + 11) \le 25h^2$.

Correctness. We now turn to prove that the instance (G, H) of LIST-HOM is equisatisfiable to an instance (G', H') of HOM.

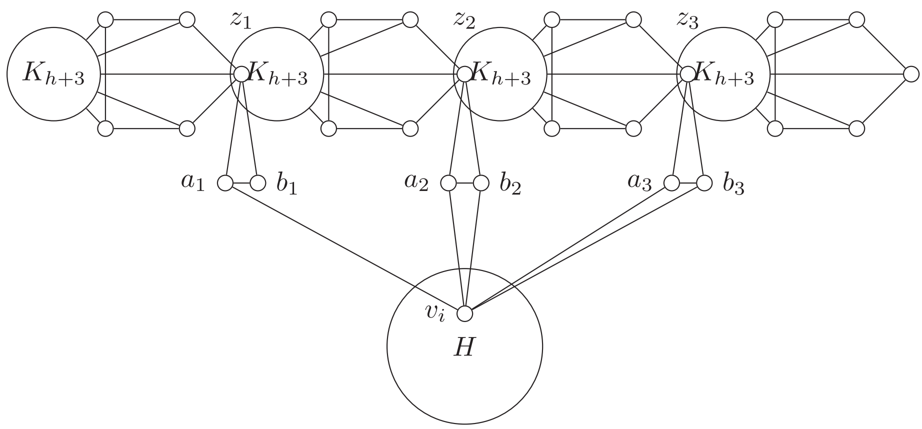

Fig. 4. The graph H'. A vertex $v_i \in V(H)$ is connected to all a_j's and all b_j's except for b_i; here we illustrate the case $i = 1$.

Claim 3.5. *Any homomorphism ϕ from G' to H' maps T_h into T_h.*

Proof of the Claim. No pair of vertices of the same clique of T_h is mapped to the same vertex in H', because H' has no self-loops. Therefore, canonical cliques from T_h are mapped to some cliques from T_h, as H' has no more cliques of size $h + 3$. The remaining vertices of T_h have at least $h + 3$ neighbors from canonical cliques, and therefore they must be mapped to vertices from T_h. □

Claim 3.6. *Any homomorphism ϕ from G' to H' bijectively maps T_h to T_h so the order of z's is preserved, that is, for each $i \in [h]$, $\phi(z_i) = z_i$.*

Proof of the Claim.

(1) *Every canonical clique is mapped to a canonical clique.* First note that a canonical clique is mapped into one block. Indeed, there are no vertices outside a block that are connected to more than one vertex of the block. Assume, to the contrary, that some canonical clique is mapped to one block but not to a canonical clique. Then its image has to contain one or two vertices of the 5-cycle from that block. If the image contains only one vertex of the 5-cycle, then the image of the 5-cycle has at most three vertices: one vertex from the canonical clique K_{h+3} and two neigbors of the vertex from the 5-cycle (because all the vertices of the image of the 5-cycle must be connected to all the vertices of the image of the clique). Note that these three vertices do not form a triangle, and therefore the 5-cycle cannot be mapped to them. If the image of the clique contains two vertices outside of the canonical clique, then for the same reason the image of the 5-cycle must contain only two vertices, which is not possible. This analysis shows that every canonical clique K_{h+3} must be mapped to a canonical clique K_{h+3}.
(2) *Every block is mapped to a block.* We already know that every canonical clique is mapped to a canonical clique. The 5-cycle from the same block must be mapped to the corresponding 5-cycle, because it is the only image that contains a closed walk of odd length and every vertex of which is connected to the clique (recall that the images of the canonical clique and the 5-cycle do not intersect, since their preimages are joined by edges). Note that since the canonical clique and the cycle are mapped to themselves, z_i has to be mapped to some z_j.
(3) *If D_i is mapped to D_j, then D_{i+1} is mapped to D_{j+1}.* The cycle from D_i shares a vertex with the canonical clique from D_{i+1}, and therefore if D_i is mapped to D_j,

then D_{i+1} can only be mapped to D_{j+1} or D_j. However, D_{i+1} cannot be mapped into the same block as D_i. Indeed, in this case the canonical clique of D_{i+1} would be mapped to the canonical clique of D_j, but we already know that z_{i+1} is mapped to the 5-cycle of D_j. Therefore, D_i and D_{i+1} must be mapped in consecutive blocks.

The above proves that for every $i \in \{0, \dots, h\}$, D_i is mapped to D_i, which implies that any homomorphism preserves the order of z's. □

CLAIM 3.7. *Any homomorphism ϕ from G' to H' bijectively maps A_h to A_h so $\{a_i, b_i\}$ is mapped to $\{a_i, b_i\}$.*

PROOF OF THE CLAIM. Every pair $\{a_i, b_i\}$ is connected to $z_i \in T_h$, so it can be mapped either to $\{a_i, b_i\}$ or to some vertices of T_h. But in the latter case it would not have paths of length 2 to all other pairs $\{a_j, b_j\}$ for $h \geq 3$. □

CLAIM 3.8. *Any homomorphism ϕ from G' to H' maps G to H.*

PROOF OF THE CLAIM. Assume, to the contrary, that a vertex $x \in V(G)$ is mapped to a vertex $v \in V(T_h)$ or a vertex $a \in V(A_h)$. The vertex x is adjacent to at least h vertices from A_h, but v and a are adjacent to at most two vertices from A_h (recall that by the previous claim every $\{a_i, b_i\}$ is mapped to $\{a_i, b_i\}$). □

Now we show that the two instances are equisatisfiable. Let ϕ be a list homomorphism from G to H. We show that its natural extension ϕ' mapping T_h to T_h and A_h to A_h is a correct homomorphism from G' to H'. This is non-trivial only for edges of G' from G to A_h. Consider an edge from a vertex x of G to a vertex b_j. The presence of this edge means that x is not mapped to $v_j \in V(H)$ by ϕ. Recall that the b_j is mapped by ϕ' to b_j, as we have assumed that ϕ' extends ϕ by the identity function on T_h and A_h. This means that the considered edge in G' is mapped to an edge in H' by ϕ'.

For the reverse direction, let ϕ' be a homomorphism from G' to H'. We show that its natural projection is a list homomorphism from G to H. Since ϕ' maps G to H (by Claim 3.8), it is enough to check that all list constraints are satisfied. For this, consider a vertex x from G and assume that $v_i \notin \mathcal{L}(x)$. Then ϕ' does not map x to v_i, as otherwise there would be no image for one of the edges xa_i or xb_i.

Running time of the reduction. The reduction clearly takes time polynomial in the input length. □

4. LOWER BOUNDS

4.1. Graph Homomorphism

We are ready to prove our main result about graph homomorphisms, that is, Theorem 1.1.

THEOREM 4.1 (THEOREM 1.1 RESTATED). *Unless ETH fails, for any constant $D > 0$ there exists a constant $c = c(D) > 0$ such that for any non-decreasing function $3 \leq h(n) \leq n^D$, there is no algorithm solving* GRAPH HOMOMORPHISM *from an n-vertex graph G to a graph H with at most $h(n)$ vertices in time*

$$\mathcal{O}\big(2^{cn \log h(n)}\big). \tag{5}$$

PROOF. The outline of the proof of the theorem is as follows. Assuming that there is a "fast" algorithm for GRAPH HOMOMORPHISM, we show that there is also a "fast" algorithm solving LIST GRAPH HOMOMORPHISM, which, in turn, implies "fast" algorithm for 3-COLORING on degree 4 graphs, contradicting ETH. In what follows, we specify what we mean by "fast."

Let $h_0 = 25^2$. If $h(n) < h_0$ for all values of n, then an algorithm with running time $\mathcal{O}(h^{cn})$ would solve 3-Coloring in time $\mathcal{O}(h_0^{cn}) = \mathcal{O}(2^{cn \log h_0})$ (recall that $h(n) \geq 3$). Therefore, by choosing a small-enough constant c such that $c \log h_0 < q$, we arrive at a contradiction with Lemma 2.2.

From now on, we assu.me that $h(n) \geq h_0$ for large-enough values of n. Let $c = \frac{q}{8D \log \gamma}$, where q is the constant from Lemma 2.2, and $\gamma := \gamma(4)$ is the constant from Lemma 3.3. For the sake of contradiction, let us assume that there exists an algorithm $\mathcal{A}$ deciding whether $G \to H$ in time $\mathcal{O}(h^{cn}) = \mathcal{O}(2^{cn \log h})$, where $|V(G)| = n$, $|V(H)| = h := h(n)$. Now we show how to solve 3-coloring on n'-vertex graphs of maximum degree four in time $2^{qn'}$, which would contradict Lemma 2.2.

Let G' be an n'-vertex graph of maximum degree four that needs to be 3-colored. Let $r = \frac{\log h}{4D \log \gamma}$ and $n = \frac{2n'}{r}$. Using Lemma 3.3 (note that $r = o(\sqrt{n'})$ as required), we construct an instance (G_1, H_1) of List Graph Homomorphism that is satisfiable if and only if the initial graph G' is 3-colorable, and $|V(G_1)| \leq \frac{n'}{r}$, $|V(H_1)| \leq \gamma^r$. By Lemma 3.4, this instance is equisatisfiable to an instance (G, H) of Graph Homomorphism where $|V(H)| < 25\gamma^{2r} = 25h^{\frac{1}{2D}} \leq h$ (since $D \geq 1$ and $h(n) \geq h_0$), and

$$|V(G)| \leq \frac{n'}{r} + 25\gamma^{2r} \leq \frac{n}{2} + 25h^{\frac{1}{2D}} \leq \frac{n}{2} + 25\sqrt{n} \leq n$$

(for sufficiently large values of n).

Now, to solve 3-coloring for G', we construct an instance (G, H) with $|V(G)| \leq n$ and $|V(H)| \leq h$ of Graph Homomorphism and invoke the algorithm $\mathcal{A}$ on this instance. The running time of $\mathcal{A}$ is

$$\mathcal{O}(2^{cn \log h}) = \mathcal{O}(2^{\frac{2cn'}{r} \log h}) = \mathcal{O}(2^{2cn' \log h \cdot \frac{4D \log \gamma}{\log h}}) = \mathcal{O}(2^{8cDn' \log \gamma}) = \mathcal{O}(2^{qn'})$$

and hence we can find a 3-coloring of G' in time $\mathcal{O}(2^{qn'})$, which contradicts ETH (see Lemma 2.2). □

Theorem 4.2 (Theorem 1.2 restated). *Unless ETH fails, for any constant $D > 0$ there exists a constant $c = c(D) > 0$ such that for any non-decreasing function $3 \leq h(n) \leq n^D$, there is no algorithm deciding if there is a locally injective homomorphism from an n-vertex graph G to a graph H with at most $h(n)$ vertices in time $\mathcal{O}(2^{cn \log h(n)})$.*

Proof. The proof is almost identical to the proof of Theorem 4.1.

Let us observe that in the reduction in Lemma 3.3, in graph G', we take a coloring (in the proof we refer to such coloring as to labeling) of the square of G'. Thus for every bucket v of G', all its neighbors are labeled by different colors. The way we construct the lists, only buckets with the same labels can be mapped to the same vertex of H'. Thus for every vertex v of G', no pair of its neighbors can be mapped to the same vertex. Hence every list homomorphism from G' to H' is locally injective. Therefore, the result of Lemma 3.3 holds for locally injective list homomorphisms as well, and we obtain the following lemma. □

Lemma 4.3. *There exists an algorithm that takes as input a graph G on n vertices of maximum degree d that needs to be 3-colored and an integer $r = o(\sqrt{n})$ and finds an equisatisfiable instance (G', H') of* Locally Injective List Graph Homomorphism, *where $|V(G')| \leq n/r$ and $|V(H')| \leq \gamma(d)^r$, where $\gamma(d)$ is a function of the graph degree. The running time of the algorithm is polynomial in n and the size of the output graphs.*

In the reduction of Lemma 3.4, we established that every homomorphism from G' to H' bijectively maps T_h to T_h and A_h to A_h so $\{a_i, b_i\}$ is mapped to $\{a_i, b_i\}$. Thus for vertices of these structures, every homomorphism is locally injective. By Claim 3.8, any homomorphism ϕ from G' to H' maps G to H. Therefore, there is a locally injective homomorphism from G' to H' if and only if there is a locally injective list homomorphism from G to H. Then by making use of Lemma 4.3, the calculations performed in the proof of Theorem 4.1, we conclude with the proof of the theorem.

4.2. Subgraph Isomorphism

To prove a lower bound for SUBGRAPH ISOMORPHISM, we need a reduction, which, given an instance of GRAPH HOMOMORPHISM, produces a single exponential number of instances of SUBGRAPH ISOMORPHISM. Even though from the perspective of polynomial time algorithms such a reduction gives no implication in terms of which problem is harder, in our setting it is enough to obtain a lower bound for SUBGRAPH ISOMORPHISM.

THEOREM 4.4. *Given an instance (G, H) of* GRAPH HOMOMORPHISM *, one can in* $\mathrm{poly}(p)2^p$ *time create 2^p instances of* SUBGRAPH ISOMORPHISM *with $|V(G)|$ vertices in each of the graphs, where $p = |V(G)| + |V(H)|$, such that (G, H) is a yes-instance if and only if at least one of the created instances of* SUBGRAPH ISOMORPHISM *is a yes-instance.*

PROOF. Let (G, H) be an instance of GRAPH HOMOMORPHISM and let $p = |V(G)|+|V(H)|$. Note that any homomorphism h from G to H can be associated with some sequence of non-negative numbers $(|h^{-1}(v)|)_{v\in V(H)}$, being the numbers of vertices of G mapped to particular vertices of H. The sum of the numbers in such a sequence equals exactly $|V(G)|$. As the number of such sequences is $\binom{V(G)+V(H)-1}{V(H)-1} \le 2^p$, we can enumerate all such sequences in time $2^p\,\mathrm{poly}(p)$. For each such sequence $(a_v)_{v\in V(H)}$ we create a new instance (G', H') of SUBGRAPH ISOMORPHISM, where the pattern graph remains the same, that is, $G' = G$, and in the host graph H' each vertex of $v \in V(H)$ is replicated exactly a_v times (possibly zero). Observe that $|V(H')| = |V(G')|$.

We claim that G admits a homomorphism to H if and only if for some sequence $(a_v)_{v\in V(H)}$ the graph G' is a subgraph of H'. First, assume that G admits a homomorphism h to H. Consider the instance (G', H') created for the sequence $a_v = |h^{-1}(v)|$ and observe that we can create a bijection $h' : V(G') \to V(H')$ by assigning $v \in V(G')$ to its private copy of $h(v)$. As h is a homomorphism, so is h', and as h' is at the same time a bijection, we infer that G' is a subgraph of H'.

On the other hand, if for some sequence $(a_v)_{v\in V(H)}$ the constructed graph G' is a subgraph of H', then projecting the witnessing injection $g : V(G') \to V(H')$ so $g'(v)$ is defined as the prototype of the copy $g(v)$, gives a homomorphism from G to H, as copies of each $v \in V(H)$ form independent sets in H'. □

Combining Theorem 4.1 with Theorem 4.4, we immediately obtain the following lower bound.

THEOREM 4.5 (THEOREM 1.3 RESTATED). *Unless ETH fails, there exists a constant $c > 0$ such that there is no algorithm deciding whether a given n-vertex graph G contains a subgraph isomorphic to a given n-vertex graph H in time $\mathcal{O}(n^{cn})$.*

5. CONCLUSION AND OPEN PROBLEMS

In this work, we resolved a number of questions about exact exponential algorithms. Our lower bounds suggest several directions for further research.

"Fine-grained" Dichotomy. The classical results of Hell and Nešetřil [1990] establishes the following dichotomy for GRAPH HOMOMORPHISM subject to P$\neq$ NP : For every fixed simple graph H, the problem of whether there exists a homomorphism from G to H is solvable in polynomial time if and only if H is bipartite. Is there anything similar to that in the world of exponential algorithms for HOM(G, H)?

More precisely, for graph classes $\mathcal{G}$ and $\mathcal{H}$, we denote by HOM($\mathcal{G}$, $\mathcal{H}$) the restriction of the graph homomorphism problem to input graphs $G \in \mathcal{G}$ and $H \in \mathcal{H}$. If $\mathcal{G}$ or $\mathcal{H}$ is the class of all graphs, then we use the placeholder "_" instead of a letter. Thus the result of Hell-Nešetřil states that unless P=NP, HOM(_, $\mathcal{H}$) is in P if and only if $\mathcal{H}$ is a class of bipartite graphs.

Now we know that solving HOM(_, _) with input graphs G and H in time $|V(H)|^{o(|V(G)|)}$ would refute ETH. On the other hand, when $\mathcal{H}$ is the class of graphs consisting of complete graphs, HOM(_, $\mathcal{H}$) is equivalent to computing the chromatic number of G and thus is solvable in time $\mathcal{O}(2^{|V(G)|})$ [Björklund et al. 2009]. More generally, let $\mathcal{H}$ be a graph class such that for some constant t, either the clique width or the maximum vertex degree of the core of every graph in $\mathcal{H}$ is at most t. Wahlström [2010] has shown that in this case HOM(_, $\mathcal{H}$) is solvable in single-exponential time $\mathcal{O}(f(t)^{|V(G)|}) = 2^{\mathcal{O}(|V(G)|)}$, where f is some function of $\mathcal{H}$ only. Is it possible to characterize (up to some complexity assumption) graph classes $\mathcal{H}$, where HOM(_, $\mathcal{H}$) is solvable in single-exponential time?

What about the fine-grained complexity of GRAPH HOMOMORPHISM for HOM($\mathcal{G}$, _) and HOM($\mathcal{G}$, $\mathcal{H}$)? Of course, similar questions are interesting for SUBGRAPH ISOMORPHISM, as well as for counting versions of GRAPH HOMOMORPHISM and SUBGRAPH ISOMORPHISM.

Some Concrete Problems. Are the following problems solvable in single-exponential time?

—SUBGRAPH ISOMORPHISM with instance (G, H) when the maximum vertex degree of G is 3. (When degree of G does not exceed 2, the problem is solvable in single-exponential time, see, for example, Held and Karp [1962].)
—Deciding if graph G can be obtained from graph H only by edge contractions.
—Deciding if graph G is an immersion of graph H.
—Deciding if G is a minor of a graph H for the special case when G is a clique.
—Finding a minimum distortion embedding into a cycle. We remark that embedding in a path can be done in time $2^{\mathcal{O}(|V(G)|)}$ [Cygan and Pilipczuk 2012; Fomin et al. 2011].

ACKNOWLEDGMENTS

We thank Gregory Gutin for pointing us to QAP and the anonymous reviewers for their helpful comments.

REFERENCES

Omid Amini, Fedor V. Fomin, and Saket Saurabh. 2012. Counting subgraphs via homomorphisms. *SIAM J. Discr. Math.* 26, 2 (2012), 695–717. DOI:http://dx.doi.org/10.1137/100789403

Per Austrin. 2010. Towards sharp inapproximability for any 2-CSP. *SIAM J. Comput.* 39, 6 (2010), 2430–2463.

Mihai Bădoiu, Julia Chuzhoy, Piotr Indyk, and Anastasios Sidiropoulos. 2005a. Low-distortion embeddings of general metrics into the line. In *Proceedings of the 37th Annual ACM Symposium on Theory of Computing (STOC'05)*. ACM, 225–233.

Mihai Bădoiu, Kedar Dhamdhere, Anupam Gupta, Yuri Rabinovich, Harald Räcke, R. Ravi, and Anastasios Sidiropoulos. 2005b. Approximation algorithms for low-distortion embeddings into low-dimensional spaces. In *Proceedings of the 16th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA'05)*. SIAM, 119–128.

Libor Barto, Marcin Kozik, and Todd Niven. 2008. Graphs, polymorphisms and the complexity of homomorphism problems. In *Proceedings of the 40th Annual ACM Symposium on Theory of Computing (STOC'08)*. 789–796.

Richard Beigel and David Eppstein. 2005. 3-coloring in time $O(1.3289^n)$. *J. Algor.* 54, 2 (2005), 444–453.

Andreas Björklund. 2014. Determinant sums for undirected hamiltonicity. *SIAM J. Comput.* 43, 1 (2014), 280–299. DOI:http://dx.doi.org/10.1137/110839229

Andreas Björklund, Thore Husfeldt, and Mikko Koivisto. 2009. Set partitioning via inclusion–exclusion. *SIAM J. Comput.* 39, 2 (2009), 546–563.

Nicolas Bourgeois, Bruno Escoffier, Vangelis Th. Paschos, and Johan M. M. van Rooij. 2012. Fast algorithms for max independent set. *Algorithmica* 62, 1–2 (2012), 382–415. DOI:http://dx.doi.org/10.1007/s00453-010-9460-7

Jianer Chen, Xiuzhen Huang, Iyad A. Kanj, and Ge Xia. 2006. Strong computational lower bounds via parameterized complexity. *J. Comput. Syst. Sci.* 72, 8 (2006), 1346–1367.

Marek Cygan, Fedor Fomin, Bart M. P. Jansen, Łukasz Kowalik, Daniel Lokshtanov, Dániel Marx, Marcin Pilipczuk, Michał Pilipczuk, and Saket Saurabh. 2014. School on Parameterized Algorithms and Complexity—Open Problems. Retrieved from http://fptschool.mimuw.edu.pl/opl.pdf.

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

Marek Cygan and Marcin Pilipczuk. 2012. Bandwidth and distortion revisited. *Discr. Appl. Math.* 160, 4–5 (2012), 494–504. DOI:http://dx.doi.org/10.1016/j.dam.2011.10.032

Tomás Feder and Moshe Y. Vardi. 1998. The computational structure of monotone monadic SNP and constraint satisfaction: A study through datalog and group theory. *SIAM J. Comput.* 28, 1 (1998), 57–104. DOI:http://dx.doi.org/10.1137/S0097539794266766

Uriel Feige. 2000. Coping with the NP-hardness of the graph bandwidth problem. In *Proceedings of the 7th Scandinavian Workshop on Algorithm Theory (SWAT'00)*. Lecture Notes in Computer Science, Vol. 1851. Springer, Berlin, 10–19.

Michael R. Fellows, Fedor V. Fomin, Daniel Lokshtanov, Elena Losievskaja, Frances A. Rosamond, and Saket Saurabh. 2013. Distortion is fixed parameter tractable. *ACM Trans. Comput. Theory* 5, 4 (2013), 16.

Jirí Fiala, Petr A. Golovach, and Jan Kratochvíl. 2008. Computational complexity of the distance constrained labeling problem for trees (extended abstract). In *Proceedings of the 35th International Colloquium of Automata, Languages and Programming (ICALP'08).* Lecture Notes in Computer Science, Vol. 5125. Springer, 294–305.

Jirí Fiala and Jan Kratochvíl. 2008. Locally constrained graph homomorphisms - structure, complexity, and applications. *Comput. Sci. Rev.* 2, 2 (2008), 97–111. DOI:http://dx.doi.org/10.1016/j.cosrev.2008.06.001

Fedor Fomin, Kazuo Iwama, and Dieter Kratsch. 2008. Moderately exponential time algorithms (dagstuhl seminar 08431). In Dagstuhl Reports. Retrieved from http://drops.dagstuhl.de/opus/volltexte/2008/1798/pdf/08431.SWM.Paper.1798.pdf. 1.

Fedor V. Fomin, Pinar Heggernes, and Dieter Kratsch. 2007. Exact algorithms for graph homomorphisms. *Theor. Comput. Syst.* 41, 2 (2007), 381–393.

Fedor V. Fomin and Dieter Kratsch. 2010. *Exact Exponential Algorithms*. Springer.

Fedor V. Fomin, Daniel Lokshtanov, and Saket Saurabh. 2011. An exact algorithm for minimum distortion embedding. *Theor. Comput. Sci.* 412, 29 (2011), 3530–3536. DOI:http://dx.doi.org/10.1016/j.tcs.2011.02.043

Michael R. Garey and David S. Johnson. 1979. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W. H. Freeman.

Jerrold R. Griggs and Roger K. Yeh. 1992. Labelling Graphs with a Condition at Distance 2. *SIAM J. Discr. Math.* 5, 4 (1992), 586–595.

Martin Grohe. 2007. The complexity of homomorphism and constraint satisfaction problems seen from the other side. *J. ACM* 54, 1 (2007).

Frédéric Havet, Martin Klazar, Jan Kratochvíl, Dieter Kratsch, and Mathieu Liedloff. 2011. Exact algorithms for $L(2, 1)$-labeling of graphs. *Algorithmica* 59, 2 (2011), 169–194. DOI:http://dx.doi.org/10.1007/s00453-009-9302-7

Michael Held and Richard M. Karp. 1962. A dynamic programming approach to sequencing problems. *J. SIAM* 10 (1962), 196–210.

Pavol Hell and Jaroslav Nešetřil. 1990. On the complexity of H-coloring. *J. Combin. Theory Ser. B* 48, 1 (1990), 92–110.

Pavol Hell and Jaroslav Nešetřil. 2004. *Graphs and Homomorphisms*. Oxford Lecture Series in Mathematics and its Applications, Vol. 28. Oxford University Press, Oxford.

Thore Husfeldt, Ramamohan Paturi, Gregory B. Sorkin, and Ryan Williams. 2013. Exponential algorithms: Algorithms and complexity beyond polynomial time (dagstuhl seminar 13331). In Dagstuhl Reports. Retrieved from http://drops.dagstuhl.de/opus/volltexte/2013/4342/pdf/dagrep_v003_i008_p040_s13331.pdf. 63.

Russell Impagliazzo and Ramamohan Paturi. 2001. On the complexity of k-SAT. *J. Comput. Syst. Sci.* 62, 2 (2001), 367–375. DOI:http://dx.doi.org/10.1006/jcss.2000.1727

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

Russell Impagliazzo, Ramamohan Paturi, and Francis Zane. 2001b. Which problems have strongly exponential complexity. *J. Comput. Syst. Sci.* 63, 4 (2001), 512–530.

Konstanty Junosza-Szaniawski, Jan Kratochvíl, Mathieu Liedloff, Peter Rossmanith, and Paweł Rzażewski. 2013. Fast exact algorithm for $L(2, 1)$-labeling of graphs. *Theor. Comput. Sci.* 505 (2013), 42–54. DOI:http://dx.doi.org/10.1016/j.tcs.2012.06.037

Claire Kenyon, Yuval Rabani, and Alistair Sinclair. 2009. Low distortion maps between point sets. *SIAM J. Comput.* 39, 4 (2009), 1617–1636. DOI:http://dx.doi.org/10.1137/080712921

Eugene L. Lawler. 1976. A note on the complexity of the chromatic number problem. *Inf. Process. Lett.* 5, 3 (1976), 66–67.

Andrzej Lingas and Martin Wahlen. 2009. An exact algorithm for subgraph homeomorphism. *J. Discr. Algorithms* 7, 4 (2009), 464–468. DOI:http://dx.doi.org/10.1016/j.jda.2008.10.003

Daniel Lokshtanov, Dániel Marx, and Saket Saurabh. 2011. Lower bounds based on the exponential time hypothesis. *Bull. EATCS* 105 (2011), 41–72.

Daniel Lokshtanov, Dániel Marx, and Saket Saurabh. 2013. Lower bounds based on the exponential time hypothesis. *Bull. EATCS* 3, 105 (2013).

László Lovász. 2012. *Large Networks and Graph Limits*. Vol. 60. American Mathematical Soc.

Dániel Marx. 2010. Can you beat treewidth? *Theor. Comput.* 6, 1 (2010), 85–112. DOI:http://dx.doi.org/10.4086/toc.2010.v006a005

Dániel Marx and Michal Pilipczuk. 2014. Everything you always wanted to know about the parameterized complexity of Subgraph Isomorphism (but were afraid to ask). In *Proceedings of the 31st International Symposium on Theoretical Aspects of Computer Science (STACS'14)*. 542–553. DOI:http://dx.doi.org/10.4230/LIPIcs.STACS.2014.542

Prasad Raghavendra. 2008. Optimal algorithms and inapproximability results for every CSP? In *Proceedings of the 40th Annual ACM Symposium on Theory of Computing (STOC'08)*. 245–254.

Neil Robertson and Paul D. Seymour. 1995. Graph minors. XIII. The disjoint paths problem. *J. Combin. Theory Ser. B* 63, 1 (1995), 65–110.

J. M. Robson. 1986. Algorithms for maximum independent sets. *J. Algor.* 7, 3 (1986), 425–440.

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

Michael Sipser. 2005. *Introduction to the Theory of Computation*. Cengage Learning.

Robert Endre Tarjan and Anthony E. Trojanowski. 1977. Finding a maximum independent set. *SIAM J. Comput.* 6, 3 (1977), 537–546.

Patrick Traxler. 2008. The time complexity of constraint satisfaction. In *Parameterized and Exact Computation*. Springer, 190–201.

Magnus Wahlström. 2010. Problem 5.21. time complexity of graph homomorphism. In *Exact Complexity of NP-Hard Problems. Dagstuhl Seminar 10441 Final Report*, Ramamohan Paturi Thore Husfeldt, Dieter Kratsch and Gregory Sorkin (Eds.). Dagstuhl.

Magnus Wahlström. 2011. New plain-exponential time classes for graph homomorphism. *Theor. Comput. Syst.* 49, 2 (2011), 273–282.

Gerhard J. Woeginger. 2003. Exact algorithms for NP-hard problems: A survey. In *Combinatorial Optimization—Eureka, You Shrink!*Lecture Notes in Computer Science, Vol. 2570. Springer-Verlag, Berlin, 185–207.

Received February 2016; revised October 2016; accepted February 2017