Hitting Forbidden Induced Subgraphs on Bounded Treewidth Graphs

Ignasi Sau, Uéverton dos Santos Souza

Hitting Forbidden Induced Subgraphs on Bounded Treewidth Graphs

Ignasi Sau
LIRMM, Université de Montpellier, CNRS, France
ignasi.sau@lirmm.fr

Uéverton dos Santos Souza
Instituto de Computação, Universidade Federal Fluminense, Niterói, Brazil
ueverton@ic.uff.br

Abstract

For a fixed graph H, the H-IS-Deletion problem asks, given a graph G, for the minimum size of a set $S \subseteq V(G)$ such that $G \setminus S$ does not contain H as an induced subgraph. Motivated by previous work about hitting (topological) minors and subgraphs on bounded treewidth graphs, we are interested in determining, for a fixed graph H, the smallest function $f_H(t)$ such that H-IS-Deletion can be solved in time $f_H(t) \cdot n^{\mathcal{O}(1)}$ assuming the Exponential Time Hypothesis (ETH), where t and n denote the treewidth and the number of vertices of the input graph, respectively.

We show that $f_H(t) = 2^{\mathcal{O}(t^{h-2})}$ for every graph H on $h \geq 3$ vertices, and that $f_H(t) = 2^{\mathcal{O}(t)}$ if H is a clique or an independent set. We present a number of lower bounds by generalizing a reduction of Cygan et al. [MFCS 2014] for the subgraph version. In particular, we show that when H deviates slightly from a clique, the function $f_H(t)$ suffers a sharp jump: if H is obtained from a clique of size h by removing one edge, then $f_H(t) = 2^{\Theta(t^{h-2})}$. We also show that $f_H(t) = 2^{\Omega(t^h)}$ when $H = K_{h,h}$, and this reduction answers an open question of Mi. Pilipczuk [MFCS 2011] about the function $f_{C_4}(t)$ for the subgraph version.

Motivated by Cygan et al. [MFCS 2014], we also consider the colorful variant of the problem, where each vertex of G is colored with some color from $V(H)$ and we require to hit only induced copies of H with matching colors. In this case, we determine, under the ETH, the function $f_H(t)$ for every connected graph H on h vertices: if $h \leq 2$ the problem can be solved in polynomial time; if $h \geq 3$, $f_H(t) = 2^{\Theta(t)}$ if H is a clique, and $f_H(t) = 2^{\Theta(t^{h-2})}$ otherwise.

2012 ACM Subject Classification Theory of computation → Design and analysis of algorithms; Theory of computation → Graph algorithms analysis; Theory of computation → Parameterized complexity and exact algorithms

Keywords and phrases parameterized complexity, induced subgraphs, treewidth, hitting subgraphs, dynamic programming, lower bound, Exponential Time Hypothesis

Digital Object Identifier 10.4230/LIPIcs.MFCS.2020.82

Related Version A full version of the paper is available at `https://arxiv.org/abs/2004.08324`.

Funding *Ignasi Sau*: CAPES-PRINT Institutional Internationalization Program, process 88887.371209/ 2019-00, and projects DEMOGRAPH (ANR-16-CE40-0028) and ESIGMA (ANR-17-CE23-0010).
Uéverton dos Santos Souza: Grant E-26/203.272/2017 Rio de Janeiro Research Foundation (FAPERJ) and Grant 303726/2017-2 National Council for Scientific and Technological Development (CNPq).

1 Introduction

Graph modification problems play a central role in modern algorithmic graph theory. The most general form of such a problem is determined by a target graph class $\mathcal{G}$ and some prespecified set $\mathcal{M}$ of allowed *local* modifications, and the question is, given an input graph G and an integer k, whether it is possible to transform G to a graph in $\mathcal{G}$ by applying k

modification operations from $\mathcal{M}$. A wealth of graph problems can be formulated for different instantiations of $\mathcal{G}$ and $\mathcal{M}$, and applications span diverse topics such as computational biology, computer vision, machine learning, networking, and sociology [8, 11, 19].

Probably, the most studied local modification operation in the literature is vertex deletion and, among the target graph classes, of particular relevance are the ones defined by *excluding* the graphs in a family $\mathcal{F}$ according to some natural graph containment relation, such as minor, topological minor, subgraph, or induced subgraph. By the classical classification result of Lewis and Yannakakis [24], all interesting cases of these problems are NP-hard.

One of the most popular strategies to cope with NP-hard problems is that of parameterized complexity [12, 17], where the core idea is to identify a parameter k associated with an input of size n that allows for an algorithm in time $f(k) \cdot n^{\mathcal{O}(1)}$, called *fixed-parameter tractable* (or FPT for short). A natural goal within parameterized algorithms is the quest for the "best possible" function $f(k)$ in an FPT algorithm. Usually, the working hypothesis to prove lower bounds is the *Exponential Time Hypothesis* (ETH) that states, in a simplified version, that the 3-Sat problem on n variables cannot be solved in time $2^{o(n)}$; see [20, 21] for more details.

Among graph parameters, definitely one of the most successful ones is *treewidth*, which –informally speaking– quantifies the topological resemblance of a graph to a tree. The celebrated Courcelle's Theorem [10] states that every graph problem that can be expressed in Monadic Second Order logic is solvable in time $f(t) \cdot n$ on n-vertex graphs of treewidth at most t. In particular, it applies to most vertex deletion problems discussed above. A very active area in parameterized complexity during the last years consists in optimizing, under the ETH, the function $f(t)$ given by Courcelle's Theorem for several classes of vertex deletion problems. As a byproduct, several cutting-edge techniques for obtaining both lower bounds [25] and algorithms [6, 14, 18] have been obtained, which have become part of the standard toolbox of parameterized complexity. Obtaining tight bounds under the ETH for this kind of vertex deletion problems is, in general, a very hard task, as we proceed to discuss.

Let H be a fixed graph and let © be a fixed graph containment relation. In the H-©-Deletion (meta)problem, given an n-vertex graph G, the objective is to find a set $S \subseteq V(G)$ of minimum size such that $G \setminus S$ does not contain H according to containment relation ©. We parameterize the problem by the treewidth of G, denoted by t, and the objective is to find the smallest function $f_H(t)$ such that H-©-Deletion can be solved in time $f_H(t) \cdot n^{\mathcal{O}(1)}$.

The case © = "minor" has been object of intense study during the last years [6, 14, 16, 22, 26, 28], culminating in a tight dichotomy about the function $f_H(t)$ when H is connected [2–5].

The case © = "topological minor" has been also studied recently [3–5], but we are still far from obtaining a complete characterization of the function $f_H(t)$. For both minors and topological minors, so far there is no graph H such that $f_H(t) = 2^{\Omega(t^c)}$ for some $c > 1$.

Recently, Cygan et al. [13] started a systematic study of the case © = "subgraph", which turns out to exhibit a quite different behavior from the above cases: for every integer $c \geq 1$ there is a graph H such that $f_H(t) = 2^{\Theta(t^c)}$. Cygan et al. [13] provided a general upper bound and some particular lower bounds on the function $f_H(t)$, but a complete characterization seems to be currently out of reach. Previously, Mi. Pilipczuk [27] had studied the cases where H is a cycle, finding the function $f_{C_i}(t)$ for every $i \geq 3$ except for $i = 4$.

In this article we focus on the case © = "induced subgraph" that, to the best of our knowledge, had not been studied before in the literature, except for the case $K_{1,3}$, for which Bonomo-Braberman et al. [9] showed very recently that $f_{K_{1,3}}(t) = 2^{\mathcal{O}(t^2)}$.

Our results and techniques. We first show (Theorem 2) that, for every graph H on $h \geq 3$ vertices, $f_H(t) = 2^{\mathcal{O}(t^{h-2})}$. The algorithm uses standard dynamic programming over a nice tree decomposition of the input graph. However, in order to achieve the claimed running time, we need to use a slightly non-trivial encoding in the tables that generalizes an idea of Bonomo-Braberman et al. [9], by introducing an object that we call *rooted H-folio*, inspired by similar encodings in the context of graph minors [1, 5].

It turns out that when H is a clique or an independent set (in particular, when $|V(H)| \leq 2$), the problem can be solved in *single-exponential* time, that is, $f_H(t) = 2^{\mathcal{O}(t)}$. The case of cliques (Theorem 5), which coincides with the subgraph version, had been already observed by Cygan et al. [13], using essentially the folklore fact that every clique is contained in some bag of a tree decomposition. The case of independent sets (Theorem 8) is more interesting, as we exploit tree decompositions in a novel way, by showing (Lemma 6) that a chordal completion of the complement of a solution can be covered by a constant number of cliques, which implies (Lemma 7) that the complement of a solution is contained in a constant number of bags of the given tree decomposition.

Our main technical contribution consists of lower bounds. Somehow surprisingly, we show (Theorem 10) that when H deviates slightly from a clique, the function $f_H(t)$ suffers a sharp jump: if H is obtained from a clique of size h by removing one edge, then $f_H(t) = 2^{\Omega(t^{h-2})}$, and this bound is tight by Theorem 2. We also provide lower bounds for other graphs H that are "close" to cliques (Theorem 10; see also the full version), some of them being tight. In particular, we show (Theorem 12) that when $H = K_{h,h}$, we have that $f_H(t) = 2^{\Omega(t^h)}$. By observing that the proof of the latter lower bound also applies to occurrences of $K_{h,h}$ as a *subgraph*, the particular case $h = 2$ (Corollary 13) answers the open question of Mi. Pilipczuk [27] about the function $f_{C_4}(t)$. All these reductions are inspired by a reduction of Cygan et al. [13] for the subgraph version. We first present the general frame of the reduction together with some properties that the eventual instances constructed for each of the graphs H have to satisfy, yielding in a unified way (Lemma 9) lower bounds for the corresponding problems.

Motivated by the work of Cygan et al. [13], we also consider the *colorful* variant of the problem, where the input graph G comes equipped with a coloring $\sigma : V(G) \to V(H)$ and we are only interested in hitting induced subgraphs of G isomorphic to H such that their colors match. In this case, we first observe that essentially the same dynamic programming algorithm of the non-colored version (Theorem 3) yields the upper bound $f_H(t) = 2^{\mathcal{O}(t^{h-2})}$ for every graph H on $h \geq 3$ vertices. Again, our main contribution concerns lower bounds: we show (Theorem 14), by modifying appropriately the frame introduced for the non-colored version, that $f_H(t) = 2^{\Omega(t^{h-2})}$ for every graph H having a connected component on h vertices that is not a clique. Since the case where H is a clique can also be easily solved in single-exponential time (Theorem 5), which can be shown (Theorem 15) to be optimal, it follows that if H is a connected graph on $h \geq 3$ vertices, $f_H(t) = 2^{\Theta(t)}$ if H is a clique, and $f_H(t) = 2^{\Theta(t^{h-2})}$ otherwise. It is easy to see that the cases where $|V(H)| \leq 2$ can be solved in polynomial time by computing a minimum vertex cover in a bipartite graph.

Organization. In Section 2 we provide some basic preliminaries and formally define the problems. In Section 3 we present the algorithms for both problems, and in Section 4 (resp. Section 5) we provide the lower bounds for the non-colored (resp. colored) version. Finally, we conclude the article in Section 6 with some open questions. Due to space limitations, the proofs of the results marked with "($\star$)" can be found in the full version of this article, available at `https://arxiv.org/abs/2004.08324`.

2 Preliminaries

Graphs and functions. We use standard graph-theoretic notation, and we refer the reader to [15] for any undefined notation. We will only consider undirected graphs without loops nor multiple edges, and we denote an edge between two vertices u and v by $\{u, v\}$. A subgraph H of a graph G is *induced* if H can be obtained from G by deleting vertices. A graph G is *H-free* if it does not contain any induced subgraph isomorphic to H. For a graph G and a set $S \subseteq V(G)$, we use the notation $G \setminus S := G[V(G) \setminus S]$. A vertex v is *complete* (resp. *anticomplete*) to a set $S \subseteq V(G)$ if v is adjacent (resp. not adjacent) to every vertex in S. A vertex set S of a connected graph G is a *separator* if $G \setminus S$ is disconnected.

We denote by $\Delta(G)$ (resp. $\omega(G)$) the maximum vertex degree (resp. clique size) of a graph G. For an integer $h \geq 1$, we denote by P_h (resp. I_h, K_h) the path (resp. independent set, clique) on h vertices, and by $K_h - e$ the graph obtained from K_h by deleting one edge. For two integers $a, b \geq 1$, we denote by $K_{a,b}$ the bipartite graph with parts of sizes a and b. We denote the disjoint union of two graphs G_1 and G_2 by $G_1 + G_2$.

A graph property is *hereditary* if whenever it holds for a graph G, it holds for all its induced subgraphs as well. The *open* (resp. *closed*) *neighborhood* of a vertex v is denoted by $N(v)$ (resp. $N[v]$). A vertex is *simplicial* if its (open or closed) neighborhood induces a clique. A graph G is *chordal* if it does not contain induced cycles of length at least four or, equivalently, if $V(G)$ can be ordered $v_1, \ldots, v_n$ such that, for every $2 \leq i \leq n$, vertex v_i is simplicial in the subgraph of G induced by $\{v_1, \ldots, v_{i-1}\}$. Note that being chordal is a hereditary property.

Given a function $f : A \to B$ between two sets A and B and a subset $A' \subseteq A$, we denote by $f|_{A'}$ the restriction of f to A' and by $\mathsf{im}(f)$ the image of f, that is, $\mathsf{im}(f) = \{b \in B \mid \exists a \in A : f(a) = b\}$. For an integer $k \geq 1$, we let $[k]$ be the set containing all integers i with $1 \leq i \leq k$.

Tree decompositions. A *tree decomposition* of a graph G is a pair $\mathcal{D} = (T, \mathcal{X})$, where T is a tree and $\mathcal{X} = \{X_w \mid w \in V(T)\}$ is a collection of subsets of $V(G)$, called *bags*, such that:

- $\bigcup_{w \in V(T)} X_w = V(G)$,
- for every edge $\{u, v\} \in E$, there is a $w \in V(T)$ such that $\{u, v\} \subseteq X_w$, and
- for each $\{x, y, z\} \subseteq V(T)$ such that z lies on the unique path between x and y in T, $X_x \cap X_y \subseteq X_z$.

We call the vertices of T *nodes* of $\mathcal{D}$ and the sets in $\mathcal{X}$ *bags* of $\mathcal{D}$. The *width* of a tree decomposition $\mathcal{D} = (T, \mathcal{X})$ is $\max_{w \in V(T)} |X_w| - 1$. The *treewidth* of a graph G, denoted by $\mathsf{tw}(G)$, is the smallest integer t such that there exists a tree decomposition of G of width at most t. We need to introduce nice tree decompositions, which will make the presentation of the algorithms much simpler.

Nice tree decompositions. Let $\mathcal{D} = (T, \mathcal{X})$ be a tree decomposition of G, r be a vertex of T, and $\mathcal{G} = \{G_w \mid w \in V(T)\}$ be a collection of subgraphs of G, indexed by the vertices of T. A triple $(\mathcal{D}, r, \mathcal{G})$ is a *nice tree decomposition* of G if the following conditions hold:

- $X_r = \emptyset$ and $G_r = G$,
- each node of $\mathcal{D}$ has at most two children in T,
- for each leaf $\ell \in V(T)$, $X_\ell = \emptyset$ and $G_\ell = (\emptyset, \emptyset)$. Such an ℓ is called a *leaf node*,
- if $w \in V(T)$ has exactly one child w', then either
  - $X_w = X_{w'} \cup \{v_{\text{in}}\}$ for some $v_{\text{in}} \notin X_{w'}$ and $G_w = G[V(G_{w'}) \cup \{v_{\text{in}}\}]$. The node w is called an *introduce node* and the vertex v_{in} is the *introduced vertex* of X_w,
  - $X_w = X_{w'} \setminus \{v_{\text{out}}\}$ for some $v_{\text{out}} \in X_{w'}$ and $G_w = G_{w'}$. The node w is called a *forget node* node and v_{out} is the *forget vertex* of X_w.

- if $w \in V(T)$ has exactly two children w_1 and w_2, then $X_w = X_{w_1} = X_{w_2}$, $E(G_{w_1}) \cap E(G_{w_2}) = E(G[X_w])$, and $G_w = (V(G_{w_1}) \cup V(G_{w_2}), E(G_{w_1}) \cup E(G_{w_2}))$. The node w is called a *join node*.

For each $w \in V(T)$, we denote by V_w the set $V(G_w)$. Given a tree decomposition, it is possible to transform it in polynomial time to a *nice* one of the same width [23]. Moreover, by Bodlaender et al. [7] we can find in time $2^{\mathcal{O}(\mathsf{tw})} \cdot n$ a tree decomposition of width $\mathcal{O}(\mathsf{tw})$ of any graph G. Hence, since the running time of our algorithms dominates this function, we may assume that a nice tree decomposition of width $t = \mathcal{O}(\mathsf{tw})$ is given along with the input.

Definition of the problems. Before formally defining the problems considered in this article, we introduce some terminology, mostly taken from [13]. Given a graph H, an *H-coloring* of a graph G is a function $\sigma : V(G) \to V(H)$. A *homomorphism* (resp. *induced homomorphism*) from a graph H to a graph G is a function $\pi : V(H) \to V(G)$ such that $\{u, v\} \in E(H)$ implies (resp. if and only if) $\{\pi(u), \pi(v)\} \in E(G)$. When G is H-colored by a function σ, an *(induced) σ-homomorphism* from H to G is an (induced) homomorphism π from H to G with the additional property that every vertex is mapped to the appropriate color, that is, $\sigma(\pi(a)) = a$ for every vertex $a \in V(H)$. An *(induced) H-subgraph* of G is an (induced) injective homomorphism from H to G and, if G is H-colored by a function σ, an *(induced) σ-H-subgraph* of G is an (induced) injective σ-homomorphism from H to G. We say that a vertex set $X \subseteq V(G)$ *hits* an (induced) σ-H-subgraph π if $X \cap \pi(V(H)) \neq \emptyset$.

For a fixed graph H, the problems we consider in this article are defined as follows.

H-IS-Deletion
Input: A graph G.
Output: The minimum size of a set $X \subseteq V(G)$ that hits all induced H-subgraphs of G.

Colorful H-IS-Deletion
Input: A graph G and an H-coloring σ of G.
Output: The minimum size of a set $X \subseteq V(G)$ that hits all induced σ-H-subgraphs of G.

The H-S-Deletion and Colorful H-S-Deletion problems are defined similarly, just by removing the word "induced" from the above definitions. In the decision version of these problems, we are given a target budget k, and the objective is to decide whether there exists a hitting set of size at most k. Unless stated otherwise, we let n denote the number of vertices of the input graph of problem under consideration. When expressing the running time of an algorithm, we will sometimes use the $\mathcal{O}^*(\cdot)$ notation, which suppresses polynomial factors in the input size.

Exponential Time Hypothesis. The *Exponential Time Hypothesis* (ETH) of Impagliazzo and Paturi [20] implies that the 3-Sat problem on n variables cannot be solved in time $2^{o(n)}$. The Sparsification Lemma of Impagliazzo et al. [21] implies that if the ETH holds, then there is no algorithm solving a 3-Sat formula with n variables and m clauses in time $2^{o(n+m)}$. Using the terminology from Cygan et al. [13], a 3-Sat formula φ, in conjunctive normal form, is said to be *clean* if each variable of φ appears exactly three times, at least once positively and at least once negatively, and each clause of φ contains two or three literals and does not contain twice the same variable. Cygan et al. [13] observed the following useful lemma.

▶ **Lemma 1** (Cygan et al. [13]). *The existence of an algorithm in time $2^{o(n)}$ deciding whether a clean 3-Sat formula with n variables is satisfiable would violate the ETH.*

3 Algorithms

In this section we present algorithms for H-IS-Deletion and Colorful H-IS-Deletion. We start in Subsection 3.1 with a general dynamic programming algorithm that solves H-IS-Deletion and Colorful H-IS-Deletion in time $\mathcal{O}^*(2^{\mathcal{O}(t^{h-2})})$ for any graph H on at least $h \geq 3$ vertices. In Subsection 3.2 we focus on hitting cliques and independent sets.

3.1 A general dynamic programming algorithm

We present the algorithm for H-IS-Deletion, and then we discuss that essentially the same algorithm applies to Colorful H-IS-Deletion as well. Our algorithm to solve H-IS-Deletion in time $\mathcal{O}^*(2^{\mathcal{O}(t^{h-2})})$ uses standard dynamic programming over a nice tree decomposition of the input graph; we refer the reader to [12] for a nice exposition. However, in order to achieve the claimed running time, we need to use a slightly non-trivial encoding in the tables, which we proceed to explain.

Let $|V(H)| = h$ and assume that we are given a nice tree decomposition of the input graph G such that its bags contain at most t vertices (in a tree decomposition of width t, the bags have size at most $t+1$, but to simplify the exposition we assume that they have size at most t, which does not change the asymptotic complexity of the algorithm).

Intuitively, our algorithm proceeds as follows. At each bag X_w of the nice tree decomposition of G, a state is indexed by the intersection of the desired hitting set constructed so far with the bag, and the collection of *proper* subgraphs of H that occur as induced subgraphs in the graph obtained from G_w after removing the current solution. In order to be able to proceed with the dynamic programming routine while keeping the complement of the hitting set H-free, we need to remember how these proper subgraphs of H intersect with X_w, and this is the most expensive part of the algorithm in terms of running time. We encode this collection of rooted subgraphs of H (where the "roots" correspond to the vertices in X_w) with an object $\mathcal{H}_w$ that we call a *rooted H-folio*, inspired by similar encodings in the context of graph minors [1,5]. Since we need to remember proper subgraphs of H on at most $h-1$ vertices, and we have up to t choices to root each of their vertices in the bag X_w, the number of rooted proper subgraphs of H is at most t^{h-1}. Therefore, the number of rooted H-folios, each corresponding to a collection of rooted proper subgraphs of H, is bounded above by $2^{t^{h-1}}$. This encoding naturally leads to a dynamic programming algorithm to solve H-IS-Deletion in time $\mathcal{O}^*(2^{\mathcal{O}(t^{h-1})})$, where the hidden constants may depend on H.

In order to further reduce the exponent to $h-2$, we use the following trick inspired by the dynamic programming algorithm of Bonomo-Braberman et al. [9] to solve $K_{1,3}$-IS-Deletion in time $\mathcal{O}^*(2^{\mathcal{O}(t^2)})$. The crucial observation is the following: the existence of proper induced subgraphs of H that are *fully* contained in the current bag X_w can be checked *locally* within that bag, without needing to root their vertices. That is, we distinguish these *local occurrences* of proper induced subgraphs of H, and we encode them separately in $\mathcal{H}_w$, without rooting their vertices in X_w. Note that the number of choices for those local occurrences depends only on H. In particular, since the proper subgraphs of H have at most $h-1$ vertices, the previous observation implies that we never need to root exactly $h-1$ vertices of an induced subgraph of H, since such occurrences would be fully contained in X_w. This permits to improve the running time to $\mathcal{O}^*(2^{\mathcal{O}(t^{h-2})})$. The details can be found in the full version.

Note that we may assume that H has at least three vertices, as otherwise it is a clique or an independent set, and then H-IS-Deletion can be solved in single-exponential time by the algorithms in Subsection 3.2.

▶ **Theorem 2** (⋆). *For every graph H on $h \geq 3$ vertices, the H-IS-Deletion problem can be solved in time $2^{\mathcal{O}(t^{h-2})} \cdot n$, where n and t are the number of vertices and the treewidth of the input graph, respectively.*

A dynamic programming algorithm similar to the one provided in Theorem 2 can also solve the Colorful H-IS-Deletion problem in time $2^{\mathcal{O}(t^{h-2})} \cdot n$ for every graph H on $h \geq 3$ vertices. Indeed, the algorithm remains basically the same, except that we have to keep track only of *colorful* copies of proper subgraphs of H, and to discard only the states in which a *colorful* occurrence of H appears. In order to do that, in the tables of the dynamic programming algorithm we just need to replace rooted H-folios by *rooted σ-H-folios*, defined in the natural way. Since the number of further computations at each node in order to verify that the colors match in the obtained rooted subgraphs of H is a function dominated by $2^{\mathcal{O}(t^{h-2})}$, we obtain the same asymptotic running time. We omit the details.

▶ **Theorem 3.** *For every graph H on $h \geq 3$ vertices, the Colorful H-IS-Deletion problem can be solved in time $2^{\mathcal{O}(t^{h-2})} \cdot n$, where n and t are the number of vertices and the treewidth of the input graph, respectively.*

3.2 Hitting cliques and independent sets

The following folklore lemma follows easily from the definition of tree decomposition.

▶ **Lemma 4.** *Let G be a graph and let $\mathcal{D}$ be a tree decomposition of G. Then every clique of G is contained in some bag of $\mathcal{D}$.*

Note that if H is a clique, then the (Colorful) H-IS-Deletion problem is the same as the (Colorful) H-S-Deletion problem. Cygan et al. [13] observed that, by Lemma 4, in order to solve (Colorful) K_h-IS-Deletion it is enough to store, for every bag of a (nice) tree decomposition of the input graph, the subset of vertices of the bag that belongs to the partial hitting set, and to check locally within the bag that the remaining vertices do not induce a K_h. A typical dynamic programming routine yields the following result[1].

▶ **Theorem 5** (Cygan et al. [13]). *For every integer $h \geq 1$, K_h-IS-Deletion and Colorful K_h-IS-Deletion can be solved in time $2^{\mathcal{O}(t)} \cdot n$, where n and t are the number of vertices and the treewidth of the input graph, respectively.*

The case where H is an independent set, which is NP-hard by [24], turns out to be more interesting. We proceed to present a single-exponential algorithm for I_h-IS-Deletion, and we remark that this algorithm does *not* apply to the colorful version.

Note that I_2-IS-Deletion is dual to Maximum Clique, since a minimum I_2-hitting set is the complement of a maximum clique. This duality together with Lemma 4 yield the following key insight: in any graph G, after the removal of an optimal solution of I_2-IS-Deletion, all the remaining vertices are contained in a single bag of any tree decomposition of G. Our algorithm is based on a generalization of this property to any $h \geq 1$, stated in Lemma 7, which gives an alternative way to exploit tree decompositions in order to solve the H-IS-Deletion problem. We first need a technical lemma. A *clique cover* of a graph G is a collection of cliques of G that cover $V(G)$, and its *size* is the number of cliques in the cover.

▶ **Lemma 6.** *Every I_h-free chordal graph G admits a clique cover of size at most $h - 1$.*

[1] In fact, Cygan et al. [13] presented an algorithm only for Colorful K_h-S-Deletion, but the algorithm for K_h-S-Deletion is just a simplified version of the colorful version, just by forgetting the colors.

Proof. We prove the lemma by induction on h. For $h = 2$, G itself is a clique and the claim is trivial. Suppose inductively that any I_{h-1}-free chordal graph admits a clique cover of size at most $h - 2$, let G be an I_h-free chordal graph, and let v be a simplicial vertex of G. Since $N[v]$ is a clique and G is I_h-free, it follows that $G \setminus N[v]$ is I_{h-1}-free. Since being chordal is a hereditary property, $G \setminus N[v]$ is an I_{h-1}-free chordal graph, so by induction $G \setminus N[v]$ admits a clique cover of size at most $h - 2$. These $h - 2$ cliques together with $N[v]$ define a clique cover of G of size at most $h - 1$. ◀

▶ **Lemma 7.** *Let $h \geq 2$ be an integer, let G be a graph, let $\mathcal{D}$ be a tree decomposition of G, and let S be any solution for I_h-IS-Deletion on G. Then there are at most $h - 1$ bags $X_1, X_2, \ldots, X_{h-1}$ of $\mathcal{D}$ such that $V(G) \setminus S \subseteq \bigcup_{i \in [h-1]} X_i$.*

Proof. Let $\mathcal{D}$ be a tree decomposition of G, let S be a solution for I_h-IS-Deletion on G, and let $G^\star$ be the graph obtained from G by adding an edge between any pair of vertices contained in the same bag of $\mathcal{D}$. Note that $G^\star$ is a chordal graph, and that $\mathcal{D}$ is also a tree decomposition of $G^\star$. Since being a chordal graph is a hereditary property, it follows that $G^\star \setminus S$ is chordal. Since $G \setminus S$ is I_h-free, and the property of being I_h-free is closed under edge addition, we have that $G^\star \setminus S$ is also I_h-free. Thus, $G^\star \setminus S$ is an I_h-free chordal graph, and Lemma 6 implies that $G^\star \setminus S$ admits a clique cover of size at most $h - 1$. Since any clique in $G^\star \setminus S$ is also a clique in $G^\star$, and $\mathcal{D}$ is a tree decomposition of $G^\star$, Lemma 4 implies that every clique of $G^\star \setminus S$ is contained in some bag of $\mathcal{D}$, and therefore there are at most $h - 1$ bags of $\mathcal{D}$ that cover all vertices in $V(G^\star) \setminus S = V(G) \setminus S$. ◀

Recall that I_h-IS-Deletion is NP-hard even for $h = 2$, thus the problem cannot be solved in time $n^{f(h)}$ for any function f, unless P = NP.

▶ **Theorem 8.** *For every integer $h \geq 1$, I_h-IS-Deletion can be solved in time $2^{\mathcal{O}(t)} \cdot n^h$, where n and t are the number of vertices and the treewidth of the input graph, respectively.*

Proof. For $h = 1$ the problem can be trivially solved in linear time, so assume $h \geq 2$. Let $\mathcal{D}$ be a tree decomposition of G with width t, and let S be an (unknown) optimal solution for I_h-IS-Deletion on G. By Lemma 7, there are at most $h - 1$ bags $X_1, X_2, \ldots, X_{h-1}$ of $\mathcal{D}$ such that $V(G) \setminus S \subseteq \bigcup_{i \in [h-1]} X_i$. Since we may assume that $\mathcal{D}$ has $\mathcal{O}(n)$ nodes [23], we can enumerate the candidate sets of bags $X_1, X_2, \ldots, X_{h-1}$ in time $\mathcal{O}(n^{h-1})$. For each such fixed set $X_1, X_2, \ldots, X_{h-1}$, we generate all subsets $\bar{S} \subseteq \bigcup_{i \in [h-1]} X_i$, which are at most $2^{(h-1)(t+1)}$ many, and for each $\bar{S}$ we check whether the graph $G[\bar{S}]$ is I_h-free, in time $2^t \cdot t^{\mathcal{O}(1)} \cdot n$, by computing a maximum independent set of $G[\bar{S}]$ using dynamic programming based on treewidth [12] (note that having treewidth at most t is a hereditary property). Note that, by Lemma 7, there exists some of the considered sets $\bar{S}$ such that $V(G) \setminus \bar{S} = S$, and therefore an optimal solution S of I_h-IS-Deletion on G can be found in time $\mathcal{O}(n^{h-1} \cdot 2^{(h-1)(t+1)} \cdot 2^t \cdot t^{\mathcal{O}(1)} \cdot n) = 2^{\mathcal{O}(t)} \cdot n^h$, as claimed. ◀

We would like to mention that the approach used in the algorithm of Theorem 8 does not seem to be easily applicable to the colorful version of the problem. Indeed, the colored version of Lemma 6 fails: removing a clique from a σ-I_h-free chordal graph does not necessarily yield a σ-I_{h-1}-free chordal graph, and the inductive argument does not apply.

Note that, as for any graph H and any instance (G, σ) of Colorful H-IS-Deletion, any edge between two vertices u, v with $\sigma(u) = \sigma(v)$ can be safely deleted without affecting the instance, the Colorful K_2-IS-Deletion problem is equivalent to computing a minimum vertex cover in a bipartite graph, which can be done in polynomial time. Similarly, the Colorful I_2-IS-Deletion problem can also be solved in polynomial time, by computing a minimum vertex cover in the bipartite complement of the input graph. This is in sharp contrast to the uncolored version, where both problems are NP-hard [24].

4 Lower bounds for H-IS-Deletion

In this section we present lower bounds for the H-IS-Deletion problem. Our reductions will be from the 3-Sat problem restricted to clean formulas, and are strongly inspired by a reduction of Cygan et al. [13] for the H-S-Deletion problem when H is the graph obtained from $K_{2,h}$ by attaching a triangle to each of the two vertices of degree h. We start by presenting the general frame of the reductions together with some generic properties that our eventual instances of H-IS-Deletion will satisfy, which allow to prove in a unified way (cf. Lemma 9) the equivalence of the instances. Variations of this general frame will yield the concrete reductions for distinct graphs H (cf. Theorems 10 and 12).

General frame of the reductions. Given a clean 3-Sat formula φ with n variables and m clauses, we proceed to build a so-called *frame graph* $F_{H,\varphi}$. For each graph H considered in the reductions, $F_{H,\varphi}$ will be enhanced with additional vertices and edges, obtaining a graph $G_{H,\varphi}$ that will be the constructed instance of the H-IS-Deletion problem.

Let h be an integer depending on H, to be specified in each particular reduction, and let s be the smallest positive integer such that $s^h \geq 3n$, and note that $s = \mathcal{O}(n^{1/h})$. We introduce a set of vertices $M = \{w_{i,j} \mid i \in [s], j \in [h]\}$, which we call the *central part* of the frame. One may think of this set M as a matrix with s rows and h columns. We will sometimes add an extra set T of vertices to the central part, with $|T|$ depending on H, obtaining an *enhanced central part* $M' = M \cup T$.

Let L be a graph depending on each particular graph H. By *attaching a copy of* L *between two vertices* $u, v \in V(F_{H,\varphi})$ we mean adding a new copy of L, choosing two arbitrary distinct vertices of L, and identifying them with u and v respectively.

For each variable x of φ and for each clause C containing x in a literal $\ell \in \{x, \bar{x}\}$, we add to F_φ a new vertex $a_{x,C,\ell}$. We also introduce another "dummy" vertex a_x. Since φ is clean, we have introduced four vertices in $F_{H,\varphi}$ for each variable x. Let $a_{x,C_1,\ell}$, $a_{x,C_2,\bar{\ell}}$, $a_{x,C_3,\ell}$, a_x be the four introduced vertices (recall that x appears at least once positively and negatively in φ). We attach a copy of L between the following four pairs of vertices: $(a_{x,C_1,\ell}, a_{x,C_2,\bar{\ell}})$, $(a_{x,C_2,\bar{\ell}}, a_{x,C_3,\ell})$, $(a_{x,C_3,\ell}, a_x)$, and $(a_x, a_{x,C_1,\ell})$. We denote by A the union of all the vertices in these variable gadgets.

For each clause C of φ and for each literal ℓ in C, we add to F_φ a new vertex $b_{C,\ell}$. Since φ is clean, we have introduced two or three vertices in $F_{H,\varphi}$ for each clause C. We attach a copy of L between every pair of these vertices. We denote by B the union of all the vertices in these clause gadgets. This concludes the construction of the frame $F_{H,\varphi}$; cf. Figure 1.
In all our reductions, the graph $G_{H,\varphi}$ will satisfy the following property:

P1: All the connected components of $G_{H,\varphi} \setminus M'$ are of size bounded by a function of H.

Also, in all our reductions the budget that we set for the solution of H-IS-Deletion on $G_{H,\varphi}$ is $k := 2n + \sum_{C \in \varphi}(|C| - 1) = 5n - m$, where $|C|$ denotes the number of literals in clause C. For each fixed graph H, the choice of k, the edges within M', and the edges between M' and the sets A, B will force the following behavior in $G_{H,\varphi}$:

P2: For each gadget corresponding to a variable x, at least one of the pairs $(a_{x,C_1,\ell}, a_{x,C_3,\ell})$ and $(a_x, a_{x,C_2,\bar{\ell}})$ needs to be in the solution and, for each gadget corresponding to a clause C, at least $|C| - 1$ vertices in the set $\{b_{C,\ell} \mid \ell \in C\}$ need to be in the solution.

The above property together with the choice of k imply that the budget is *tight*: exactly one of the pairs $(a_{x,C_1,\ell}, a_{x,C_3,\ell})$ and $(a_x, a_{x,C_2,\bar{\ell}})$ is in the solution, thereby defining the

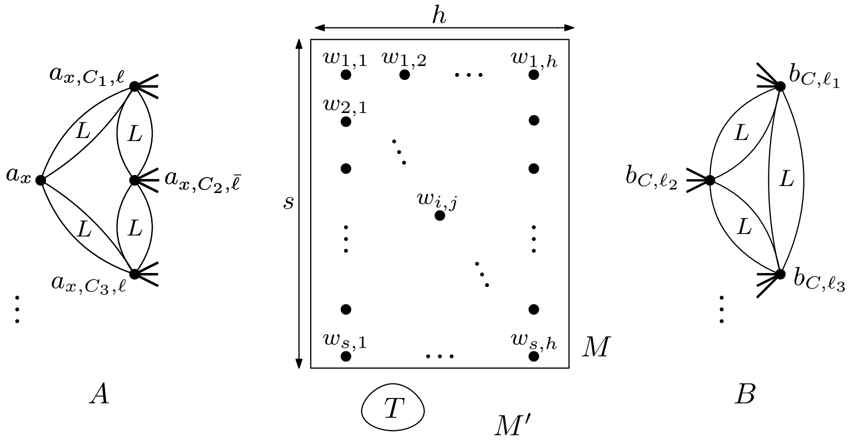

Figure 1 Illustration of the general frame graph $F_{H,\varphi}$.

true/false assignment of variable x; and exactly one of the vertices in $\{b_{C,\ell} \mid \ell \in C\}$ is *not* in the solution, corresponding to a satisfied literal in C. More precisely, our graph $G_{H,\varphi}$ will satisfy the following key property:

P3: Let $X \subseteq V(G_{H,\varphi})$ contain exactly one of $(a_{x,C_1,\ell}, a_{x,C_3,\ell})$ and $(a_x, a_{x,C_2,\bar{\ell}})$ for each variable x, and exactly $|C| - 1$ vertices in $\{b_{C,\ell} \mid \ell \in C\}$ for each clause C. Then all occurrences of H in $G_{H,\varphi} \setminus X$ as an induced subgraph contain exactly one vertex $a_{x,C,\ell} \in A$ and exactly one vertex $b_{C',\ell'} \in B$, with $(C, \ell) = (C', \ell')$. Moreover, each such a pair of vertices gives rise to an occurrence of H in $G_{H,\varphi} \setminus X$.

We now show that the above three properties are enough to construct the desired reductions.

▶ **Lemma 9** ($\star$)**.** *Let H be fixed graph and, given a clean 3-SAT formula φ, let $G_{H,\varphi}$ be a graph constructed starting from the frame graph $F_{H,\varphi}$ described above, where the central part M has h columns for some constant $h \geq 1$ depending on H. If $G_{H,\varphi}$ satisfies properties P1, P2, and P3, then the H-IS-DELETION problem cannot be solved in time $\mathcal{O}^*(2^{o(t^h)})$ unless the ETH fails, where t is the width of a given tree decomposition of the input graph.*

We now proceed to describe concrete reductions for several instantiations of H. In order to add edges between the enhanced central part M' and the sets A, B, we use the following nice trick introduced in [13]. To each pair (C, ℓ), where C is a clause of φ and ℓ is a literal in C, we assign a function $f_{C,\ell} : [h] \to [s]$. We assign these functions in such a way that $f_{C,\ell} \neq f_{C',\ell'}$ whenever $(C, \ell) \neq (C', \ell')$; note that this is possible by the choice of s and the fact that, since φ is clean, each clause contains at most three literals. We assume henceforth that these functions are fixed. We start with the following result that provides tight lower bounds for two graphs that are "close" to a clique.

▶ **Theorem 10.** *Let $h \geq 1$ be a fixed integer and let $H \in \{K_{h+2} - e, K_h + I_2\}$. Then, unless the ETH fails, the H-IS-DELETION problem cannot be solved in time $\mathcal{O}^*(2^{o(t^h)})$, where t is the width of a given tree decomposition of the input graph.*

Proof. For each graph $H \in \{K_{h+2} - e, K_h + I_2\}$, we will present a reduction from the 3-SAT problem restricted to clean formulas. Given such a formula φ, let $F_{H,\varphi}$ be the frame graph described above, where the constant h, the graph L, and the set T will be specified below for each H. In each case, we will build, starting from $F_{H,\varphi}$, an instance $G_{H,\varphi}$ of H-IS-DELETION with budget $k = 5n - m$ satisfying properties P1, P2, and P3, and then Lemma 9 will imply the claimed lower bound.

We focus here on the case $H = K_{h'+2} - e$ for some $h' \geq 1$, and the case $H = K_h + I_2$ can be found in the full version. Let $F_{H,\varphi}$ be the frame graph with $h = h'$, $L = K_{h+2} - e$, and $T = \emptyset$. We add an edge between any two vertices $w_{i,j}, w_{i',j'} \in M$ with $j \neq j'$. That is, we turn $G_{H,\varphi}[M]$ into a complete h-partite graph, where each part has size s. For each clause C and each literal ℓ in C, where $\ell \in \{x, \bar{x}\}$ for some variable x, we add the edges $\{a_{C,x,\ell}, w_{f_{C,\ell}(j),j}\}$ and $\{b_{C,\ell}, w_{f_{C,\ell}(j),j}\}$ for every $j \in [h]$. This concludes the construction of $G_{H,\varphi}$, which clearly satisfies property P1. By the choice of k and the fact that there is a copy of H between the corresponding vertices of A and B (cf. Figure 1), property P2 holds as well. Let $X \subseteq V(G_{H,\varphi})$ be a set as in property P3, and let $\tilde{H}$ be an induced subgraph of $G_{H,\varphi} \setminus X$ isomorphic to H. Since $\omega(G_{H,\varphi}[M]) = h$, $\omega(G_{H,\varphi}[(A \cup B) \setminus X]) \leq h$, and $\omega(H) = h + 1$, $\tilde{H}$ intersects both M and $A \cup B$. Moreover, since no two adjacent vertices in $(A \cup B) \setminus X$ have neighbors in M, necessarily $|V(\tilde{H}) \cap (A \cup B)| = 2$ and $V(\tilde{H}) \cap M$ induces a clique of size h, which implies that $\tilde{H}$ contains a vertex in each column of M. By the definition of the functions $f_{C,\ell}$ and the construction of $G_{H,\varphi}$, the two vertices in $V(\tilde{H}) \cap (A \cup B)$ must be $a_{x,C,\ell} \in A$ and $b_{C',\ell'} \in B$ with $(C, \ell) = (C', \ell')$, and therefore property P3 follows and we are done by Lemma 9. ◀

Note that, in the proof of Theorem 2 for $H = K_{h+2} - e$, all the occurrences of H in $G_{H,\varphi}$ are induced, and therefore the lower bound also applies to the $(K_{h+2} - e)$-S-Deletion problem. On the other hand, for $H = K_h + I_2$ the proof of Theorem 2 strongly uses the fact that H cannot occur as an *induced* subgraph. The following lemma explains why the proof does not work for the subgraph version: it can be easily solved in single-exponential time. This points out an interesting difference between both problems.

▶ **Lemma 11** ($\star$). *For every two fixed integers $h \geq 1$ and $\ell \geq 0$, the $(K_h + I_\ell)$-S-Deletion problem can be solved in time $\mathcal{O}^*(2^{\mathcal{O}(t)})$, where t is the width of a given tree decomposition of the input graph.*

By Theorem 2, the lower bounds presented in Theorem 10 for $H \in \{K_{h+2} - e, K_h + I_2\}$ are tight under the ETH. These two graphs are very symmetric, in the sense that each of them contains two non-adjacent vertices that are either complete or anticomplete to a "central" clique K_h (cf. Figure 2). Unfortunately, for graphs without two such non-adjacent symmetric vertices, our framework described above is not capable of obtaining tight lower bounds. Nevertheless, in the full version we show how to obtain lower bounds for other graphs, namely for the graph $K_{h+1} + v_x$ for $0 \leq x \leq h - 1$, defined as the graph obtained from K_{h+1} by adding a vertex v adjacent to x vertices in the clique (cf. Figure 2).

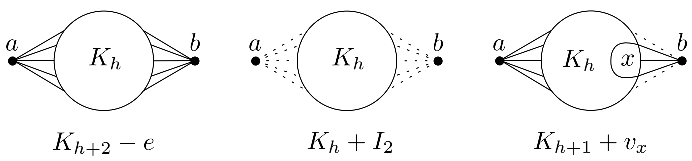

Figure 2 Graphs H considered in Theorem 10 and in the lower bounds presented in the full version.

Another direction for generalizing the lower bound of Theorem 10 to other graphs H is to consider complete bipartite graphs.

▶ **Theorem 12** ($\star$). *For any integer $h \geq 2$, the $K_{h,h}$-IS-Deletion problem cannot be solved in time $\mathcal{O}^*(2^{o(t^h)})$ unless the ETH fails, where t is the width of a given tree decomposition of the input graph.*

It can be easily verified that the proof of Theorem 12 works for both $K_{h,h}$-IS-Deletion and $K_{h,h}$-S-Deletion, since all the occurrences of $K_{h,h}$ in the constructed graph $G_{H,\varphi}$ are induced. Hence, as the particular case of Theorem 12 for $h = 2$ we get the following corollary, which answers a question of Mi. Pilipczuk [27] about the asymptotic complexity of C_4-S-Deletion parameterized by treewidth.

▶ **Corollary 13.** *Neither C_4-IS-Deletion nor C_4-S-Deletion can be solved in time $\mathcal{O}^*(2^{o(t^2)})$ unless the ETH fails, where t is the width of a given tree decomposition of the input graph.*

As mentioned in [27], C_4-S-Deletion can be easily solved in time $\mathcal{O}^*(2^{\mathcal{O}(t^2)})$. This fact together with Theorem 2 imply that both lower bounds of Corollary 13 are tight.

We can obtain lower bounds for other graphs H that are "close" to a complete bipartite graph. Indeed, note that the lower bound of Theorem 12 also applies to the graph H obtained from $K_{h,h}$ by turning one of the two parts into a clique: the same reduction works similarly, and the only change in the construction is to turn the whole central part M into a clique. We can also consider complete bipartite graphs $K_{a,b}$ with parts of different sizes, by letting the number of columns of the central part M be equal to $\max\{a, b\}$, hence obtaining a lower bound of $\mathcal{O}^*(2^{o(t^{\max\{a,b\}})})$. Similarly, we can also turn one of the two parts of $K_{a,b}$ into a clique, and obtain the same lower bound. In particular, in this way we can obtain a lower bound of $\mathcal{O}^*(2^{o(t^h)})$ for the graph H obtained from K_{h+3} by removing the edges in a triangle.

5 Lower bounds for Colorful H-IS-Deletion

Our main reduction for the colored version is again strongly inspired by the corresponding reduction of Cygan et al. [13] for the non-induced version. The main difference with respect to their reduction is that in the non-induced version, the graph H is required to contain a connected component that is neither a *clique* nor a *path*, while for the induced version we only require a component that is not a *clique*, and therefore we need extra arguments to deal with the case where all the connected components of H are paths.

▶ **Theorem 14** (⋆)**.** *Let H be a graph having a connected component on h vertices that is not a clique. Then Colorful H-IS-Deletion cannot be solved in time $\mathcal{O}^*(2^{o(t^{h-2})})$ unless the ETH fails, where t is the width of a given tree decomposition of the input graph.*

When H is a connected graph, the lower bound of Theorem 14 together with the algorithms given by Proposition 5 and Theorem 3 completely settle, under the ETH, the asymptotic complexity of Colorful H-IS-Deletion parameterized by treewidth. Note that, in particular, Theorem 14 applies when H is path, in contrast to the subgraph version that can be solved in polynomial time [13].

Therefore, what remains is to obtain tight lower bounds when H is disconnected. In particular, Theorem 14 cannot be applied at all when all the connected components of H are cliques, since the machinery that we developed (inspired by Cygan et al. [13]) using the framework graph $F_{H,\varphi}$ crucially needs two non-adjacent vertices in the same connected component. Let us now focus on those graphs, sometimes called *cluster* graphs in the literature.

As mentioned in Section 3, both Colorful K_2-IS-Deletion and Colorful I_2-IS-Deletion can be solved in polynomial time. In our next result we show that if H is slightly larger than these two graphs (namely, K_2 or I_2), then Colorful H-IS-Deletion becomes hard. Namely, we provide a single-exponential lower bound for the following three graphs H on three vertices that are *not* covered by Theorem 14: K_3, I_3, and $K_2 + K_1$. Note that these lower bounds are tight by the algorithm of Theorem 3.

▶ **Theorem 15** ($\star$). *Let $H \in \{K_3, I_3, K_2 + K_1\}$. Then, unless the ETH fails, the* Colorful *H-IS-*Deletion *problem cannot be solved in time $\mathcal{O}^*(2^{o(t)})$, where t is the width of a given tree decomposition of the input graph.*

The proof of Theorem 15 can be easily adapted to $H = P_3$ by complementing the appropriate neighborhoods, hence obtaining a lower bound of $\mathcal{O}^*(2^{o(t)})$ for P_3-IS-Deletion. Note, however, that this lower for P_3 bound already follows from Theorem 14.

It is also easy to adapt the proof of Theorem 15 to larger graphs, but then the lower bound of $\mathcal{O}^*(2^{o(t)})$ is not tight anymore. For example, for $H = 2K_2$ (the disjoint union of two edges), with $V(H) = \{z_1, z_2, z_3, z_4\}$ such that the edges are $\{z_1, z_2\}$ and $\{z_3, z_4\}$, it suffices to take the instance $(G_{K_2+K_1}, \sigma)$ of $(K_2 + K_1)$-IS-Deletion defined above and to add a private neighbor colored z_4 for every vertex of $G_{K_2+K_1}$ colored z_3. Also, for $H = K_h$ with $h \geq 4$, in the gadget L we just replace the triangles by cliques of size h, and for $H = I_h$ with $h \geq 4$, we take the h-partite complement of the previous instance of K_h-IS-Deletion.

6 Further research

Concerning H-IS-Deletion, the complexity gap is still quite large for most graphs H, as our lower bounds (Theorems 10 and 12) only apply to graphs H that are "close" to cliques or complete bipartite graphs. In particular, Theorem 10 provides tight bounds for P_3 or $K_4 - e$ (the *diamond*), but we do not know the tight function $f_H(t)$ for other small graphs H on four vertices such as P_4, $K_{1,3}$ (the *claw*), or $2K_2$.

We think that for most graphs H on h vertices, the upper bound $f_H(t) = 2^{\mathcal{O}(t^{h-2})}$ given by Theorem 2 is the asymptotically tight function, and that the single-exponential algorithms for cliques and independent sets are isolated exceptions. The reason is that, in contrast to the subgraph version, when hitting induced subgraphs, edges and non-edges behave essentially in the same way when performing dynamic programming, as one has to keep track of both the existence and the non-existence of edges in order to construct the tables, and storing this information seems to be unavoidable.

As for Colorful H-IS-Deletion, in view of Theorems 3, 5, 14, and 15, only the cases where H is a disjoint union of at least two cliques and $|V(H)| \geq 4$ remain open. In particular, when H is an independent set or a matching with $|V(H)| \geq 4$.

References

1 Isolde Adler, Frederic Dorn, Fedor V. Fomin, Ignasi Sau, and Dimitrios M. Thilikos. Faster parameterized algorithms for minor containment. *Theoretical Computer Science*, 412(50):7018–7028, 2011. doi:10.1016/j.tcs.2011.09.015.

2 Julien Baste, Ignasi Sau, and Dimitrios M. Thilikos. A complexity dichotomy for hitting connected minors on bounded treewidth graphs: the chair and the banner draw the boundary. In *Proc. of the 31st Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 951–970, 2020. doi:10.1137/1.9781611975994.57.

3 Julien Baste, Ignasi Sau, and Dimitrios M. Thilikos. Hitting minors on bounded treewidth graphs. II. Single-exponential algorithms. *Theoretical Computer Science*, 814:135–152, 2020. doi:10.1016/j.tcs.2020.01.026.

4 Julien Baste, Ignasi Sau, and Dimitrios M. Thilikos. Hitting minors on bounded treewidth graphs. III. Lower bounds. *Journal of Computer and System Sciences*, 109:56–77, 2020. doi:10.1016/j.jcss.2019.11.002.

5 Julien Baste, Ignasi Sau, and Dimitrios M. Thilikos. Hitting minors on bounded treewidth graphs. I. General upper bounds. Corresponding to Section 3 here, to appear in *SIAM Journal on Discrete Mathematics*.

6 Hans L. Bodlaender, Marek Cygan, Stefan Kratsch, and Jesper Nederlof. Deterministic single exponential time algorithms for connectivity problems parameterized by treewidth. *Information and Computation*, 243:86–111, 2015. doi:10.1016/j.ic.2014.12.008.

7 Hans L. Bodlaender, Pål Grønås Drange, Markus S. Dregi, Fedor V. Fomin, Daniel Lokshtanov, and Michal Pilipczuk. A $c^k n$ 5-Approximation Algorithm for Treewidth. *SIAM Journal on Computing*, 45(2):317–378, 2016. doi:10.1137/130947374.

8 Hans L. Bodlaender, Pinar Heggernes, and Daniel Lokshtanov. Graph modification problems (dagstuhl seminar 14071). *Dagstuhl Reports*, 4(2):38–59, 2014. doi:10.4230/DagRep.4.2.38.

9 Flavia Bonomo-Braberman, Julliano R. Nascimento, Fabiano S. Oliveira, Uéverton S. Souza, and Jayme L. Szwarcfiter. Linear-time algorithms for eliminating claws in graphs. *CoRR*, abs/2004.05672, 2020. URL: http://arxiv.org/abs/2004.05672.

10 Bruno Courcelle. The Monadic Second-Order Logic of Graphs. I. Recognizable Sets of Finite Graphs. *Information and Computation*, 85(1):12–75, 1990. doi:10.1016/0890-5401(90)90043-H.

11 Christophe Crespelle, Pål Grønås, Drange, Fedor V. Fomin, and Petr A. Golovach. A survey of parameterized algorithms and the complexity of edge modification. *CoRR*, abs/2001.06867, 2013. URL: https://arxiv.org/abs/2001.06867.

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

13 Marek Cygan, Dániel Marx, Marcin Pilipczuk, and Michal Pilipczuk. Hitting forbidden subgraphs in graphs of bounded treewidth. *Information and Computation*, 256:62–82, 2017. doi:10.1016/j.ic.2017.04.009.

14 Marek Cygan, Jesper Nederlof, Marcin Pilipczuk, Michal Pilipczuk, Johan M. M. van Rooij, and Jakub Onufry Wojtaszczyk. Solving connectivity problems parameterized by treewidth in single exponential time. In *Proc. of the 52nd Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 150–159, 2011. doi:10.1109/FOCS.2011.23.

15 Reinhard Diestel. *Graph Theory, 4th Edition*, volume 173 of *Graduate texts in mathematics*. Springer, 2012. URL: https://dblp.org/rec/books/daglib/0030488.bib.

16 Frederic Dorn, Eelko Penninkx, Hans L. Bodlaender, and Fedor V. Fomin. Efficient Exact Algorithms on Planar Graphs: Exploiting Sphere Cut Decompositions. *Algorithmica*, 58(3):790–810, 2010. doi:10.1007/s00453-009-9296-1.

17 Rodney G. Downey and Michael R. Fellows. *Fundamentals of Parameterized Complexity*. Texts in Computer Science. Springer, 2013. doi:10.1007/978-1-4471-5559-1.

18 Fedor V. Fomin, Daniel Lokshtanov, Fahad Panolan, and Saket Saurabh. Efficient computation of representative families with applications in parameterized and exact algorithms. *Journal of the ACM*, 63(4):29:1–29:60, 2016. doi:10.1145/2886094.

19 Fedor V. Fomin, Saket Saurabh, and Neeldhara Misra. Graph modification problems: A modern perspective. In *Proc. of the 9th International Frontiers in Algorithmics Workshop (FAW)*, volume 9130 of *LNCS*, pages 3–6, 2015. doi:10.1007/978-3-319-19647-3_1.

20 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.

21 Russell Impagliazzo, Ramamohan Paturi, and Francis Zane. Which Problems Have Strongly Exponential Complexity? *Journal of Computer and System Sciences*, 63(4):512–530, 2001. doi:10.1006/jcss.2001.1774.

22 Bart M. P. Jansen, Daniel Lokshtanov, and Saket Saurabh. A Near-Optimal Planarization Algorithm. In *Proc. of the 25th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 1802–1811, 2014. doi:10.1137/1.9781611973402.130.

23 Ton Kloks. *Treewidth. Computations and Approximations*. Springer-Verlag LNCS, 1994. doi:10.1007/BFb0045375.

24 John M. Lewis and Mihalis Yannakakis. The Node-Deletion Problem for Hereditary Properties is NP-Complete. *Journal of Computer and System Sciences*, 20(2):219–230, 1980. doi:10.1016/0022-0000(80)90060-4.

25 Daniel Lokshtanov, Dániel Marx, and Saket Saurabh. Lower bounds based on the Exponential Time Hypothesis. *Bulletin of the EATCS*, 105:41–72, 2011. URL: http://eatcs.org/beatcs/index.php/beatcs/article/view/92.

26 Marcin Pilipczuk. A tight lower bound for Vertex Planarization on graphs of bounded treewidth. *Discrete Applied Mathematics*, 231:211–216, 2017. doi:10.1016/j.dam.2016.05.019.

27 Michal Pilipczuk. Problems parameterized by treewidth tractable in single exponential time: A logical approach. In *Proc. of the 36th International Symposium on Mathematical Foundations of Computer Science (MFCS)*, volume 6907 of *LNCS*, pages 520–531, 2011. doi:10.1007/978-3-642-22993-0_47.

28 Juanjo Rué, Ignasi Sau, and Dimitrios M. Thilikos. Dynamic programming for graphs on surfaces. *ACM Transactions on Algorithms*, 10(2):8:1–8:26, 2014. doi:10.1145/2556952.