Incompressibility of H-Free Edge Modification Problems

Leizhen CAI* and Yufei CAI†

Department of Computer Science and Engineering
The Chinese University of Hong Kong
Shatin, Hong Kong SAR, China

lcai@cse.cuhk.edu.hk and cai@mathematik.uni-marburg.de

September 1, 2014

Abstract

Given a fixed graph H, the H-Free Edge Deletion (resp., Completion, Editing) problem asks whether it is possible to delete from (resp., add to, delete from or add to) the input graph at most k edges so that the resulting graph is H-free, i.e., contains no induced subgraph isomorphic to H. These H-free edge modification problems are well known to be fixed-parameter tractable for every fixed H.

In this paper we study the incompressibility, i.e., nonexistence of polynomial kernels, for these H-free edge modification problems in terms of the structure of H, and completely characterize their nonexistence for H being paths, cycles or 3-connected graphs. We also give a sufficient condition for the nonexistence of polynomial kernels for $\mathcal{F}$-Free Edge Deletion problems, where $\mathcal{F}$ is a finite set of forbidden induced subgraphs.

As an effective tool, we have introduced an incompressible constraint satisfiability problem Propagational-f Satisfiability to express common propagational behaviors of events, and we expect the problem to be useful in studying the nonexistence of polynomial kernels in general.

Keyword: Parameterized complexity, polynomial kernel, polynomial compression, incompressibility, and edge modification.

*Partially supported by GRF grants CUHK410409 and CUHK410212 of the Research Grants Council of Hong Kong.

†Current address: Philipps-Universitaet, Marburg Mehrzweckgebaeude 05D06, Hans-Meerwein Straβe, 35032 Marburg, Gemany.

1 Introduction

Edge modification problems are concerned with adding edges to or deleting edges from input graphs to obtain graphs with desired properties, and have been studied extensively under frameworks of both traditional and parameterized complexities. In this paper, we focus on edge modification problems concerning the property of being *H-free* for a fixed graph H, i.e., our target graph contains no induced subgraph isomorphic to H. Such problems are fundamental as every hereditary property is H-free for every graph H in a set of forbidden induced subgraphs.

In connection with edge modification, the H-Free Edge Deletion problem asks whether it is possible to delete from the input graph G at most k edges to obtain an H-free graph. Similarly we can form H-Free Edge Completion and H-Free Edge Editing problems by replacing "delete from" with "add to" and "delete from or add to" respectively.

For every fixed H, the above H-free edge modification problems with respect to parameter k are fixed-parameter tractable following a general result of Cai [5]. On the other hand, not much is known about the existence of polynomial kernels for such H-free edge modification problems, and Cai [1] raised the issue of determining the existence of polynomial kernels for H-Free Edge Deletion in IWPEC'06. Roughly speaking, a polynomial kernel of a problem instance is an equivalent instance that has size bounded by a polynomial of k and can be constructed in polynomial time. Polynomial kernels provide an effective way to compress problem instances. In this paper, we wish to establish the nonexistence of polynomial kernels, i.e., the *incompressibility*, for H-free edge modification problems in terms of the structure of H, and we hope that the work in the paper will lay the foundation for possible dichotomy theorems on the incompressibility of H-free edge modification problems.

Under the assumption that NP $\not\subseteq$ coNP/poly, Kratsch and Wahlström [14] constructed the first H for which neither H-Free Edge Deletion nor H-Free Edge Editing admits polynomial kernels, and Guillemot et al. [11] established the nonexistence of polynomial kernels for H-Free Edge Deletion when H is any path P_l with $l \geq 13$ or cycle C_t with $t \geq 12$, which has been improved recently to $l \geq 7$ and $t \geq 4$ by Guillemot et al. [12]. On the other hand, Gramm et al. [9] obtained polynomial kernels for P_3-Free Edge Deletion, Completion and Editing, and Guillemot et al. [11, 12] presented polynomial kernels for P_4-Free Edge Deletion, Completion and Editing. This summarizes known results about polynomial kernels of H-free edge modification problems.

In this paper, we significantly improve our knowledge on the incompressibility of H-free edge modification problems — we fully characterize 3-connected H for which H-free edge modification problems admit no polynomial kernel, and determine exact conditions for P_l-free (resp., C_t-free) edge modification problems to not admit polynomial kernel, assuming NP $\not\subseteq$ coNP/poly.

- For 3-connected H, H-Free Edge Deletion and Editing admit no polynomial kernel iff H is not a complete graph, and H-Free Edge Completion

admits no polynomial kernel iff H misses at least two edges.

- For H being a fixed path or cycle, H-Free Edge Deletion, Completion and Editing, respectively, admit no polynomial kernel iff H has at least 4 edges.
- For a finite set $\mathcal{F}$ of forbidden induced subgraphs, $\mathcal{F}$-Free Edge Deletion admits no polynomial kernel if all graphs in $\mathcal{F}$ are 3-connected and there is a graph $H \in \mathcal{F}$ with fewest edges such that one can add an edge to H to obtain a graph not in $\mathcal{F}$.

In order to obtain our results, we introduce a constraint satisfiability problem Propagational-f Satisfiability and prove its incompressibility, which is inspired by the Not-1-in-3 SAT problem of Kratsch and Wahlström [14] for proving the first incompressible H-free edge modification problem. Secondly, we try to ease the complication in dealing with edge modification problems by adding a restriction to edges that can be added/deleted to form "quarantined" H-free edge modification problems. Then we use Propagational-f Satisfiability as our seed problem and reduce it to quarantined H-free edge modification problems to establish their incompressibility, and finally we lift the quarantine by using "enforcers" to prevent edges from being added/deleted, which in turn yields our main results in the paper.

The work in the paper considerably enhances our knowledge on the incompressibility of H-free edge modification problems, and our Propagational-f Satisfiability problem is very effective in establishing their incompressibility. We hope that our ideas will be useful in the discovery of possible dichotomy theorems on the incompressibility of H-free edge modification problems, and we also expect Propagational-f Satisfiability to be useful in studying the nonexistence of polynomial kernels in general.

The rest of the paper is organized as follows. We fix notation and definitions, and also give some background for incompressibility in Section 2. We introduce Propagational-f Satisfiability and show its incompressibility in Section 3. In Section 4, we discuss basic components for reductions from Propagational-f Satisfiability to our quarantined H-free modification problems, and we use them in Section 5 to establish the incompressibility of quarantined H-free edge modification problems. In Section 6, we will lift the quarantine by using enforcers to establish our main results. We conclude the paper in Section 7 with some open problems and conjectures for further research.

2 Preliminaries

All graphs in this paper are simple undirected graphs. For a graph $G = (V, E)$, $\overline{G} = (V, \overline{E})$ denotes the *complement* of G. Edges of $\overline{G}$ are *antiedges* of G. Unless specified otherwise, we use E^- and E^+ to denote edges being deleted and antiedges being added. Any set of edges whose deletion from G results in an H-free graph is an *H-free deletion set*, and any set of antiedges whose addition to G results in an H-free graph is an *H-free completion set*. Note that technically speaking, edge addition

should be called antiedge addition: when we say "add edges to G", it is understood that we add to G edges of $\overline{G}$, i.e., antiedges of G. We use $\mathbf{N}$ for the set of natural numbers.

2.1 Edge modification problems

Our H-free edge modification problems are concerned with adding/deleting edges to obtain H-free graphs, and we give their technical definitions as follows, where H is a fixed graph.

H-Free Edge Deletion
Instance: Graph $G = (V, E)$, and *parameter* $k \in \mathbf{N}$.
Question: Is there $E^- \subseteq E$ with $|E^-| \leq k$ such that $G - E^-$ is H-free?

H-Free Edge Completion
Instance: Graph $G = (V, E)$, and *parameter* $k \in \mathbf{N}$.
Question: Is there $E^+ \subseteq \overline{E}$ with $|E^+| \leq k$ such that $G + E^+$ is H-free?

H-Free Edge Editing
Instance: Graph $G = (V, E)$, and *parameter* $k \in \mathbf{N}$.
Question: Are there $E^- \subseteq E$ and $E^+ \subseteq \overline{E}$ with $|E^-| + |E^+| \leq k$ such that $G - E^- + E^+$ is H-free?

As a useful auxiliary tool, we often add a restriction to edges/antiedges that can be deleted/added. For this purpose, we call a graph G an *edge-quarantined graph* if its edges are partitioned into *forbidden edges* and *allowed edges*, and we can delete from G allowed edges only. Similarly, we call a graph G an *antiedge-quarantined graph* if its antiedges are partitioned into *forbidden antiedges* and *allowed antiedges*, and we can add to G allowed antiedges only.

In all figures, forbidden edges are indicated by thick edges, allowed antiedges are indicated by dashed edges and forbidden antiedges are invisible.

2.2 Kernelization lower bounds

We assume that the reader is familiar with the general framework for kernelization lower bounds [2, 3, 4, 8] and only give necessary definitions and results, including a relaxation of composition algorithms. A mini survey in a recent paper of Bodlaender, Jansen and Kratsch [3] contains useful background materials. For an instance (I, k) of a parameterized problem P, the main input I is encoded with some finite alphabet $\mathbf{\Sigma}$ and parameter $k \in \mathbf{N}$ is encoded in unary. Kernels in this paper refer to *generalized kernels* defined below.

Definition 2.1 (see [2]) *A* generalized kernelization *from a parameterized problem P to another parameterized problem P' is an algorithm that, for input $(I, k) \in P$, takes*

time polynomial in $|I| + k$ and outputs an instance $(I', k') \in P'$ such that
(a) *(I, k) is a yes-instance of P iff (I', k') is a yes-instance of P', and*
(b) *both $|I'|$ and k' are bounded by a computable function $g(k)$.*
The output (I', k') is a kernel, *and* polynomial kernel *if $g(k)$ is a polynomial.*

The notion of polynomial kernels is naturally generalized to *polynomial compressions* by relaxing the target problem P' to any problem (instead of parameterized problem), i.e., language $L \subseteq \mathbf{\Sigma}^*$.

Definition 2.2 (see [3]) *Let P be a parameterized problem and $L \subseteq \mathbf{\Sigma}^*$ a language. A* polynomial compression *from P to L is an algorithm that, for input $(I, k) \in P$, takes time polynomial in $|I| + k$ and outputs a string $y \in \mathbf{\Sigma}^*$ such that*
(a) *(I, k) is a yes-instance of P iff $y \in L$, and*
(b) *the length of y is bounded by a polynomial of k.*

The following reduction, *polynomial parameter transformation* (*ppt-reduction* in short), plays a prominent role in demonstrating incompressibility and will be used extensively in our investigation of H-free modification problems.

Definition 2.3 (see [3, 4]) *A* ppt-reduction *from a parameterized problem P to another parameterized problem P' is an algorithm that, for input $(I, k) \in P$, takes time polynomial in $|I| + k$ and outputs an instance $(I', k') \in P'$ such that*
(a) *(I, k) is a yes-instance of P iff (I', k') is a yes-instance of P', and*
(b) *parameter k' is bounded by a polynomial of k.*

The definition of ppt-reductions immediately yields the following theorem that makes them a key tool for establishing incompressibility of parameterized problems.

Theorem 2.4 (see [3]) *If there is a ppt-reduction from a parameterized problem P to another parameterized problem P', then P' admits no polynomial compression (hence no polynomial kernel) whenever P admits no polynomial compression.*

We also need a relaxation of composition algorithms introduced by Bodlaender, Downey, Fellows, and Hermelin [2] in their pioneer work on incompressibility — we relax their requirement for parameter k' from polynomial in k to polynomial in $\max_i^t |I_i| + \log t$.

Definition 2.5 *A relaxed-composition algorithm for a parameterized problem P takes t instances $(I_1, k), \ldots, (I_t, k) \in P$ as input and, in time polynomial in $\sum_{i=1}^t |I_i| + k$, outputs an instance $(I', k') \in P$ such that*
(a) *(I', k') is a yes-instance of P iff some (I_i, k) is a yes-instance of P, and*
(b) *k' is polynomial in $\max_i^t |I_i| + \log t$.*

Let $n = \max_i^t |I_i|$ and note that the work of Bodlaender et al. [2] remains valid when $k' = n^{O(1)}$ (the last paragraph in the proof of their Lemma 1). As $\mathbf{\Sigma}$ is a finite alphabet, we may assume that $t \leq (|\mathbf{\Sigma}| + 1)^n$ and hence $\log t = O(n)$, implying $k' = (n + \log t)^{O(1)} = n^{O(1)}$ for relaxed compositions. Therefore Bodlaender et al. [2], together with a result of Fortnow and Santhanam [8], implicitly established the following theorem, which can also be regarded as a special case of Corollary 3.6 in the paper of Bodlaender, Jansen and Kratsch [3] on cross-compositions.

Theorem 2.6 (see [2, 3, 8]) *If an NP-complete parameterized problem admits a relaxed-composition algorithm, then it has no polynomial compression, hence no polynomial kernel, unless* NP $\subseteq$ coNP/poly.

Remark 2.7 Relaxed-compositions possess several advantages in establishing the incompressibility of a parameterized problem P. To prove that P has no polynomial kernel by a composition algorithm, one often produces an output (I', k') with $k' = O(k + \log t)$, and one needs an FPT algorithm of P to handle the case that $t > 2^k$. This is rather awkward — the method fails to work if P has no FPT algorithm, but in this case P actually has no kernel at all! Relaxed-compositions enable us to rid this awkward argument without any assumption about an FPT algorithm for P or the magnitude of t with respect to k. Secondly, relaxed-compositions do not need a polynomial equivalence relation required by the general technique of cross-compositions. Thirdly, k' is usually $(k + \log t)^{O(1)}$ but may indeed go up to $(n + \log t)^{O(1)}$, and finally the same relaxation also works for AND-compositions.

For simplicity in discussions, we call a parameterized problem *incompressible* when it admits no polynomial compression (hence no polynomial kernel) under the assumption that NP $\not\subseteq$ coNP/poly.

3 Satisfiability of propagational formulas

One main complication of H-free edge modification problems lies in the possibility of introducing new induced copies of H when we add/delete edges, which causes a propagation of edge additions/deletions. This propagational phenomenon is similar to the propagation of truth values of Boolean variables in certain Boolean formulas, and we introduce in this section a constraint satisfaction problem Propagational-f Satisfiability for propagational functions f. We will establish the incompressibility of the problem, and use it extensively in later sections to show the incompressibility of our edge modification problems. Note that we disallow negated variables as arguments of $f(x, y, z)$ in the following definition.

Definition 3.1 *A ternary Boolean function $f(x, y, z)$, where x, y and z are either Boolean variables or constants 0 or 1, is* propagational *if it satisfies $f(1, 0, 0) = 0$ and $f(0, 0, 0) = f(1, 0, 1) = f(1, 1, 0) = f(1, 1, 1) = 1$.*

In other words, $f(x, y, z)$ is propagational if it is true when either $x = y = z = 0$ or "$x = 1$ implies $y = 1$ or $z = 1$". There are eight different propagational functions f in total due to the freedom of defining values of f for the other three assignments of variables.

Example 3.2 *The following three functions are propagational:*

$$\begin{aligned} f_1(x, y, z) &= \overline{x} \vee y \vee z, \\ f_2(x, y, z) &= x \,\texttt{XOR}\, (y \,\texttt{NOR}\, z), \\ \texttt{Not-1-in-3}(x, y, z) &= (\overline{x} \vee y \vee z) \wedge (x \vee \overline{y} \vee z) \wedge (x \vee y \vee \overline{z}). \end{aligned}$$

Propagational functions $f(x, y, z)$ generalize function $\texttt{Not-1-in-3}(x, y, z)$ of Kratsch and Wahlström [14], and capture the relation describing "whatever happens to x must happen to either y or z", which is of great use when we deal with edge modification problems because of propagations of edge additions/deletions. The following example of C_4-Free Edge Deletion illustrates such a connection. Suppose that we want to delete some light edges from the graph in Fig. 1 to obtain a C_4-free graph. When we delete edge x, we create a new induced C_4 in the graph, and we must delete either edge y or edge z or both in order to destroy the new C_4. Therefore the propagation of edge deletions from x to y or z simulates a propagational function $f(x, y, z)$.

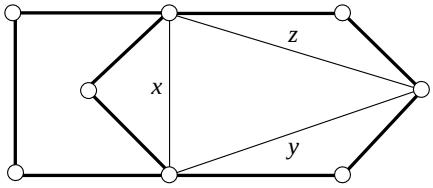

Figure 1: Realization of a propagational function $f(x, y, z)$ by C_4-free edge deletion.

For a Boolean function $f(x, y, z)$, a *conjunctive formula* φ is of the form

$$f(x_1, y_1, z_1) \wedge f(x_2, y_2, z_2) \wedge \cdots \wedge f(x_m, y_m, z_m).$$

Each $f(x_i, y_i, z_i)$ is a *clause* of φ, and the *weight* of a truth assignment of 0's and 1's to variables is the number of 1's in the assignment. For φ, the *degree of a variable* is its number of occurrences in φ, and the *degree of* φ is the maximum degree of its variables. A formula φ is *d-regular* if all its variables have degree d.

Propagational-f Satisfiability
Instance: Conjunctive formula φ of a propagational ternary function f with distinct variables inside each clause of φ, and *parameter* $k \in \mathbf{N}$.
Question: Does φ have a satisfying truth assignment of weight $\leq k$?

We remark that constants 0 and 1 are allowed to appear multiple times inside any clause of φ. Also, the above problem definition actually yields eight different

satisfiability problems, one for each of the eight different propagational functions f. When we consider the incompressibility of an H-free edge modification problem, we will use the structure of H to choose a proper f for a reduction from PROPAGATIONAL-f SATISFIABILITY.

As the first step towards the incompressibility of PROPAGATIONAL-f SATISFIABILITY we show the NP-completeness of the problem with a technical requirement on input formulas which appears to be crucial for establishing the incompressibility. The proof in the following lemma uses a single reduction from the classical VERTEX COVER problem on 3-regular graphs, and is valid for all eight different propagational f. The NP-completeness of this restricted VERTEX COVER was proved by Garey, Johnson and Stockmeyer [10].

Lemma 3.3 *For any propagational ternary Boolean function f,* PROPAGATIONAL-f SATISFIABILITY *is NP-complete for degree-3 conjunctive formulas with exactly one occurrence of constant 1.*

Proof. The problem is obviously in NP, and we first give a straightforward polynomial reduction from the classical VERTEX COVER problem on 3-regular graphs to our problem on 3-regular conjunctive formulas. For an arbitrary instance (G, k) of VERTEX COVER on 3-regular graph G with m edges, we construct a conjunctive formula φ' as follows: for each edge xy of G, form a clause $f(1, x, y)$ of φ' by regarding x and y as Boolean variables. Since f is propagational, we must set either x or y to 1 to satisfy $f(1, x, y)$, which is equivalent to choosing either vertex x or vertex y to cover edge xy. It follows that G has a vertex cover of size $\leq k$ iff φ' has a satisfying truth assignment of weight $\leq k$.

Now we convert φ' into a degree-3 conjunctive formula φ with exactly one occurrence of 1. For this purpose, we introduce m new variables u_i, $1 \leq i \leq m$, to replace all 1's in φ', and put m new clauses

$$(1, u_1, 0), (u_1, u_2, 0), \ldots, (u_{m-1}, u_m, 0)$$

into φ. Since f is propagational, clause $f(1, u_1, 0)$ will force $u_1 = 1$, which subsequently forces every $u_i = 1$, $1 \leq i \leq m$. We put into φ every clause of φ' after replacing the occurrence of 1 in the clause by a distinct u_i.

Our conversion from φ' to φ clearly takes polynomial time, and we can easily check that φ is a degree-3 formula with exactly one occurrence of 1, namely, in clause $f(1, u_1, 0)$, and each clause contains distinct variables. Furthermore, it is clear that φ' is satisfiable with weight $\leq k$ iff φ is satisfiable with weight $\leq m + k$. Therefore this special case of PROPAGATIONAL-f SATISFIABILITY is NP-complete. ∎

We now establish the incompressibility of PROPAGATIONAL-f SATISFIABILITY, and in fact we will do so for 3-regular conjunctive formulas, which improves our result on 6-regular formulas in the preliminary version of this paper [6]. It is this 3-regularity condition accompanied by the variety of f that makes the problem suitable and effective for ppt-reductions to H-free edge modification problems for various H.

Theorem 3.4 *For any propagational ternary Boolean function f,* Propagational-f Satisfiability *on 3-regular conjunctive formulas admits no polynomial compression, hence no polynomial kernel, unless* NP $\subseteq$ coNP/poly.

Proof. We begin by showing the incompressibility of the problem for degree-3 formulas with exactly one occurrence of 1. By Theorem 2.6 and Lemma 3.3, we need only present a relaxed-composition algorithm for this restricted case. Our proof is similar to that of Kratsch and Wahlström [14] for showing the incompressibility of their Not-1-in-3 SAT problem.

Let f be an arbitrary propagational function, and $(\varphi_0, k), \ldots, (\varphi_{t-1}, k)$ be t instances of Propagational-f Satisfiability where each φ_i has degree 3 and exactly one occurrence of 1. Without loss of generality, we may assume that $t = 2^h$ for some integer h since we can always duplicate some instances in the sequence to make up 2^h instances.

First we note that, since f is propagational, clause $f(1, x, y)$ forces either x or y to take value 1. This fact naturally leads us to the construction of a composition tree by introducing some new variables u_i and new clauses. For convenience, we regard subscript i of each u_i as a binary string, and describe our construction by a complete binary tree T of height $h = \log t$ as composition tree, where a vertex a and its two children b, c represent clause $f(a, b, c)$.

We start with root node 1 with two children u_0 and u_1, which yields a new clause $f(1, u_0, u_1)$ and thus forces either u_0 or u_1 to take value 1. For each internal node u_i, we add two new variables u_{i0} and u_{i1} and make them children of u_i, which results in a new clause $f(u_i, u_{i0}, u_{i1})$ and hence forces either u_{i0} or u_{i1} to take value 1 when $u_i = 1$. See Fig. 2 for the construction of T. Note that our construction forces all variables in some root-to-leaf path in T to take value 1, and hence at least one leaf of T has value 1.

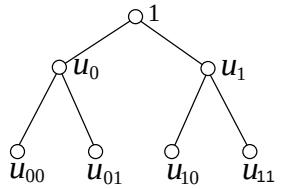

Figure 2: Composition tree T, where a vertex a and its two children b, c represent clause $f(a, b, c)$, and the variable of each leaf is used to replace the unique occurrence of 1 in a distinct φ_i.

Having the composition tree T in hand, we now construct a required instance (φ', k') from $(\varphi_0, k), \ldots, (\varphi_{t-1}, k)$ as follows.

1. Rename variables in all φ_i so that each formula φ_i has distinct variables.
2. For each $0 \leq i \leq t-1$, replace the unique occurrence of 1 in the i-th formula φ_i by leaf-variable u_i to form φ'_i (from now on we interpret subscript of each leaf

u_i as a decimal number).

3. Put all φ'_i, $0 \le i \le t-1$, and all clauses in T together to form φ' and set $k' = k + h = k + \log t$.

Note that φ' is a degree-3 conjunctive formula, and is constructed in time polynomial in $\sum_{i=0}^{t-1} |\varphi_i| + k$.

If some φ_i is satisfiable by a weight-k truth assignment ξ_i, then we construct a truth assignment ξ of weight $k' = k + h$ for φ' as follows: assign 1 to all variables on the path in T from root node 1 to leaf u_i, satisfy φ'_i in φ' by ξ_i together with $u_i = 1$, and assign 0 to all other variables in φ'. Note that for each $j \ne i$, the only occurrence of 1 in φ_j was replaced by variable u_j to form φ'_j and $\xi(u_j) = 0$. Therefore all variables in φ'_j have value 0 under ξ and thus every clause in φ'_j is satisfied as $f(0,0,0) = 1$ for propagational f. Furthermore, under truth assignment ξ, clauses in T have forms $f(0,0,0)$, $f(1,1,0)$, or $f(1,0,1)$, and therefore are satisfied as f is propagational.

Conversely, if φ' is satisfied with a truth assignment of weight $k' = k + \log t$, then all variables on the path in T from root node 1 to some leaf u_i are forced to be true, which sets $\log t$ variables to 1. Since T contains exactly t leaves, u_i appears in φ'_i and hence φ'_i is satisfied with $\le k$ true variables, implying that φ_i is satisfied with $\le k$ true variables. Therefore we have obtained a relaxed-composition algorithm for Propagational-f Satisfiability on degree-3 conjunctive formulas with exactly one occurrence of 1, and the incompressibility of the problem follows from Theorem 2.6 and Lemma 3.3.

Finally, we modify a degree-3 conjunctive formula into an equivalent 3-regular conjunctive formula as follows: for each variable x of degree $d < 3$, add $3 - d$ clauses of the form $f(1,1,x)$. This is clearly a ppt-reduction and therefore we have the result in the theorem. ∎

4 Components for representing formulas

We will establish the incompressibility of our H-free edge modification problems mainly by ppt-reductions from propagational satisfiability problems. This requires us to represent Boolean formulas by graphs, and we will discuss components for constructing such graphs in this section.

To ease the complication in dealing with edge modification problems, we first add a restriction to edges/antiedges that can be deleted/added to form edge-quarantined graphs or antiedge-quarantined graphs, and consider edge deletion/completion problems on such quarantined graphs. We will construct edge-quarantined (resp., antiedge-quarantined) graphs for edge deletion (resp., completion) to represent conjunctive formulas φ. The basic idea is to use a satisfaction-testing component $S(x,y,z)$ to represent a clause $f(x,y,z)$ in φ, form a truth-setting component $T(u)$ for a Boolean variable u to ensure the consistency for the value of u in different clauses, and connect

satisfaction-testing components and truth-setting components together in a proper way.

4.1 Satisfaction-testing components

We start with the definition of a satisfaction-testing component $S(x, y, z)$ for H-free edge deletion, which will be used to realize a propagational function $f(x, y, z)$.

Definition 4.1 *For H-free edge deletion, a satisfaction-testing component $S(x, y, z)$ is a constant-size edge-quarantined H-free graph with exactly three allowed edges $\{x, y, z\}$, called* variable-edges*, such that the following Boolean function is propagational: $f_S(x, y, z) = 1$ iff the graph obtained from $S(x, y, z)$ is H-free when we delete from $S(x, y, z)$ edges in $\{x, y, z\}$ with value 1.*

For H being C_4 or C_5, we have satisfaction-testing components for H-free deletion in Fig. 3(a) and (b) respectively. Both components represent the propagational function `Not-1-in-3`(x, y, z) as $S(x, y, z)$ in (a) (resp., (b)) itself is C_4-free (resp., C_5-free) and we need to delete at least two edges from $\{x, y, z\}$ to ensure that $S(x, y, z)$ remains C_4-free (resp., C_5-free).

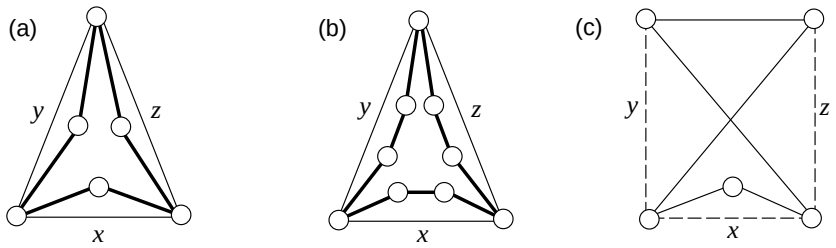

Figure 3: Satisfaction-testing components $S(x, y, z)$ for (a) C_4-free edge deletion (b) C_5-free edge deletion, and (c) C_4-free edge completion. In (a) and (b), forbidden edges are indicated by thick edges. In (c), allowed antiedges are indicated by dashed edges and forbidden antiedges are invisible. Components in (a) and (b) represent `Not-1-in-3`(x, y, z) and the component in (c) realizes $f_S(x, y, z) = x$ `XOR` $(y$ `NOR` $z)$.

We define satisfaction-testing components $S(x, y, z)$ for H-free edge completion in a similar way, which reflects the complementarity between edge deletion and edge completion.

Definition 4.2 *For H-free edge completion, a satisfaction-testing component $S(x, y, z)$ is a constant-size antiedge-quarantined graph with exactly three allowed antiedges $\{x, y, z\}$, called* variable-antiedges*, such that the following Boolean function is propagational: $f_S(x, y, z) = 1$ iff the graph obtained from $S(x, y, z)$ is H-free when we add to $S(x, y, z)$ antiedges in $\{x, y, z\}$ with value 1.*

Fig. 3(c) gives a satisfaction-testing component for C_4-free edge completion, which realizes the propagational function $f_S(x, y, z) = x$ `XOR` $(y$ `NOR` $z)$: when we add some

antiedges in $\{x, y, z\}$ to $S(x, y, z)$, the resulting graph is C_4-free iff the string xyz is 000, 101, 110 or 111.

For general H, it is also easy to construct satisfaction-testing components for H-free edge deletion/completion.

Lemma 4.3 *Let H be a connected fixed graph with at least 4 vertices. Satisfaction-testing components $S(x, y, z)$ exist for H-free edge deletion if H is not a complete graph, and for H-free edge completion if H has at least two antiedges.*

Proof. If H is not a complete graph, then it contains an antiedge x and two edges y and z as H is connected and has at least 4 vertices. Set $S(x, y, z)$ to $H + x$ with $\{x, y, z\}$ being only allowed edges in $H+x$ and all other edges of $H+x$ being forbidden edges. See Fig. 4(a) and (b) for an example of the construction of $S(x, y, z)$.

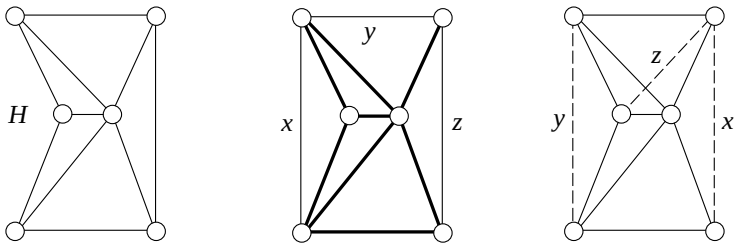

Figure 4: Construction of satisfaction-testing components $S(x, y, z)$ for general H: (a) graph H, (b) H-free edge deletion, and (c) H-free edge completion.

Since $H + x$ is H-free and the deletion of edge x will cause deletion of either edge y or edge z to maintain H-freeness, it is easily checked that

$$f_S(0, 0, 0) = f_S(1, 0, 1) = f_S(1, 1, 0) = f_S(1, 1, 1) = 1 \text{ but } f_S(1, 0, 0) = 0,$$

implying that f_S is propagational and thus $S(x, y, z)$ is a satisfaction-testing component for H-free edge deletion.

If H has at least two antiedges y and z, we arbitrarily take an edge x of H and set $S(x, y, z)$ to $H - x$ with $\{x, y, z\}$ being the only allowed antiedges and all other antiedges of $H - x$ being forbidden antiedges. Fig. 4(c) gives an example of $S(x, y, z)$. Since $H - x$ is H-free and the addition of antiedge x to $S(x, y, z)$ will cause the addition of either antiedge y or antiedge z in order to maintain H-freeness, it is easily checked that

$$f_S(0, 0, 0) = f_S(1, 0, 1) = f_S(1, 1, 0) = f_S(1, 1, 1) = 1 \text{ but } f_S(1, 0, 0) = 0.$$

Therefore f_S is propagational and $S(x, y, z)$ is a satisfaction-testing component for H-free edge completion. ∎

4.2 Truth-setting components

The purpose of a truth-setting component $T(u)$ for a Boolean variable u is to ensure the consistency for the value of u in different clauses by linking satisfaction-testing components together, and we define truth-setting components as follows.

Definition 4.4 *For H-free edge deletion (resp., edge completion), a truth-setting component $T(u)$ is a constant-size edge-quarantined (resp., antiedge-quarantined) H-free graph that contains, amongst other allowed edges (resp., antiedges), three allowed edges (resp., antiedges) without common vertex, which are also called* variable-edges *(resp.,* variable-antiedges*), and admits exactly two deletion sets (resp., completion sets): the empty set and the set containing all allowed edges (resp., antiedges).*

Fig. 5 gives truth-setting components $T(u)$ for C_4-free edge deletion, C_5-free edge deletion, and C_4-free edge completion. They are formed by taking three copies of their corresponding basic chain in the bottom of the figure and merging their leftmost triangles (resp., 4-cycles) together. For these basic chains, it is easy to see that, as far as H-freeness is concerned, the deletion of any allowed edge (resp., addition of any allowed antiedge) will cause the deletion of all allowed edges (resp., addition of all allowed antiedges) in the basic chain, which implies that $T(u)$ satisfies the required properties.

We now give a construction of truth-setting components $T(u)$ for fixed graphs H in general. We will construct a basic unit, use it to form a basic chain, and then link basic chains in a cyclic fashion, instead of ray-shaped, to form $T(u)$.

Lemma 4.5 *Let H be a 3-connected fixed graph that is not a complete graph. Then truth-setting components $T(u)$ exist for both H-free edge deletion and completion.*

Proof. First we show that H contains an antiedge e and an edge e' sharing no common vertex. By the assumption that H is not a complete graph, H has an antiedge $e = ab$. If there is an edge e' not incident with a or b, then e' is a required edge. Otherwise, all edges are incident with a or b and thus H is a bipartite graph. Since H is 3-connected, vertex a has at least three neighbors $\{a_1, a_2, a_3\}$ and we can set antiedge $e = a_2a_3$ and edge $e' = aa_1$, which are vertex-disjoint.

For H-free edge deletion, we construct a basic unit $U = H + e$, set e and e' as the only allowed edges in U and make all other edges in U forbidden edges.

Since e and e' are the only allowed edges of U, deletion of e from U will force deletion of e' to maintain H-freeness. This property enables us to construct the following basic chain $B(u)$ for a truth setting component $T(u)$: Take h vertex-disjoint copies $U_1, \ldots, U_h$ of U, where h is the number of vertices in H, and identify edge e' of U_i with edge e of U_{i+1} to form a chain of U's. See Fig. 6(a) for an example of the construction. Edge e of U_1 and edge e' of U_h are variable-edges of $B(u)$ for Boolean variable u. For convenience, we also refer to e and e', respectively, as the leftmost and rightmost edges of $B(u)$. Since U is H-free and H is 3-connected, $B(u)$ is H-free.

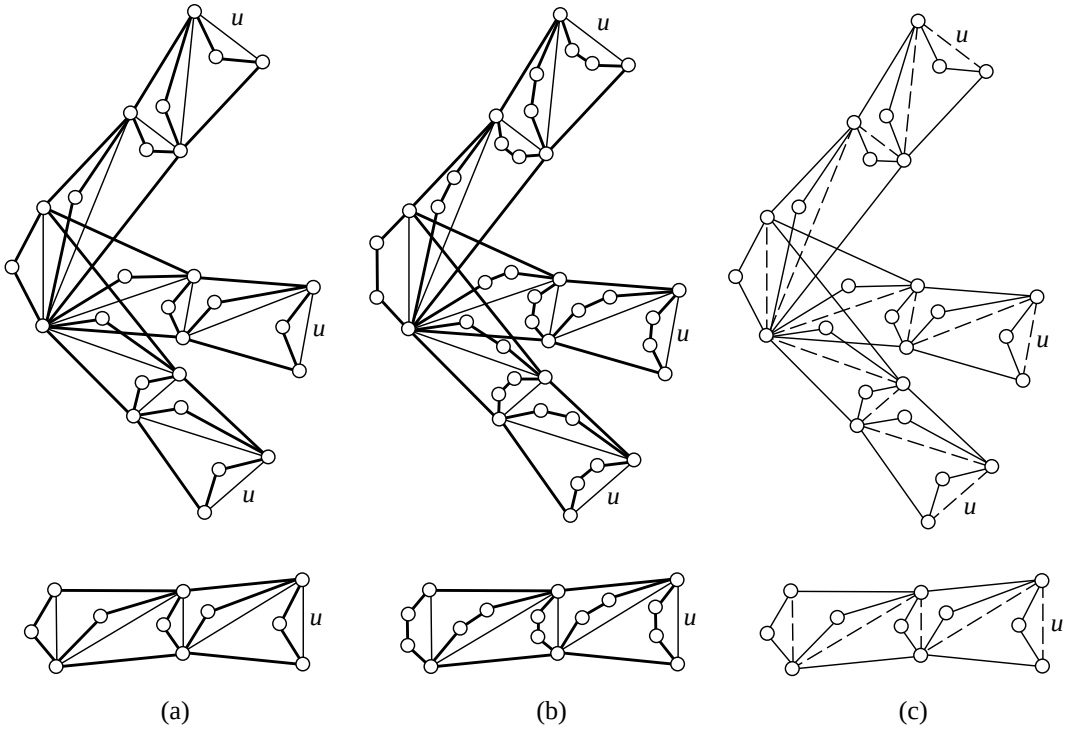

Figure 5: Truth-setting components $T(u)$ for (a) C_4-free edge deletion, (b) C_5-free edge deletion, and (c) C_4-free edge completion. Graphs in the bottom of the figure are basic chains for constructing $T(u)$. Note that $T(u)$ has 13 allowed edges (resp., antiedges) in these three cases.

It is easy to see that $B(u)$ has the following important propagation property: *To maintain H-freeness, deletion of an allowed edge e in $B(u)$ will force the deletion of all allowed edges in $B(u)$ to the right of e* (see Fig. 6(a)).

We now construct a truth-setting component $T(u)$ by taking three vertex-disjoint copies B_0, B_1, B_2 of $B(u)$ and link them into a cycle by identifying the rightmost edge of B_i with the leftmost edge of B_{i+1} for $0 \leq i \leq 2$, with index i modulo 3. In other words, $T(u)$ consists of $3h$ copies of the basic unit U linked in a cyclic fashion. Note that $T(u)$ contains exactly three variable-edges resulted from the identification of variable-edges of B_0, B_1, B_2. The propagation property of $B(u)$ ensures that $T(u)$ has exactly two deletion sets – the empty set and the set of all allowed edges, and therefore $T(u)$ is a valid truth-setting component.

For H-free edge completion, we also use edge e' and antiedge e in H. Recall that e' and e share no common vertex. We construct a basic unit $U = H - e'$, set e and e' as the only allowed antiedges in U and make all other antiedges in U' forbidden antiedges.

Since e and e' are the only allowed antiedges of U, the addition of e' to U will force the addition of e to maintain H-freeness. As with H-free edge deletion, we can link h copies of the basic unit U to form a basic chain $B(u)$ and then link three

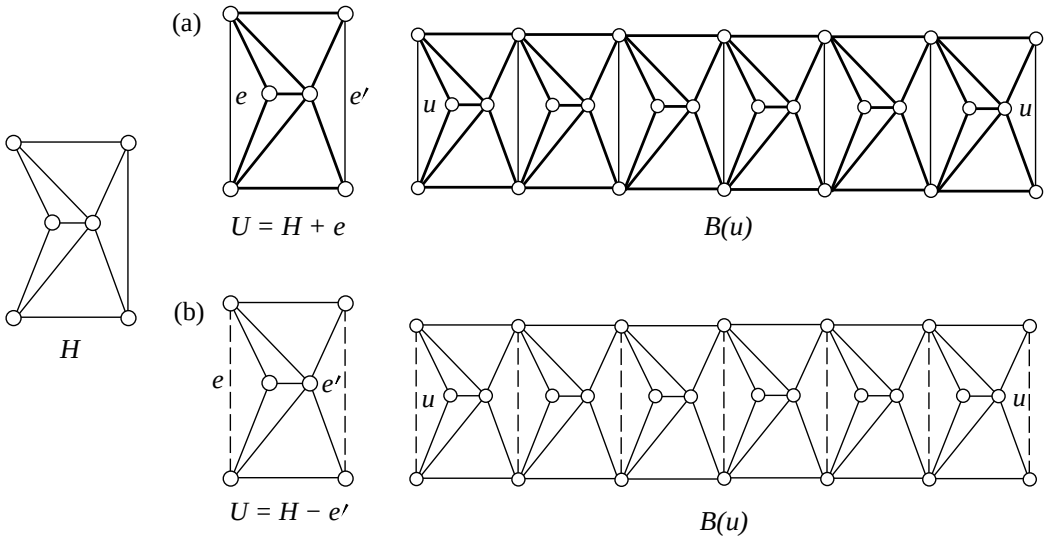

Figure 6: Construction of basic chains $B(u)$ of truth-setting components $T(u)$ for (a) H-free edge deletion, and (b) H-free edge completion. Note that e and e', respectively, are antiedge and edge of H sharing no common vertex.

vertex-disjoint copies of $B(u)$ into a cycle to form a truth setting component $T(u)$. Fig. 6(b) gives an example for the construction of $B(u)$. For this $B(u)$, the addition of an allowed antiedge e' in $B(u)$ will force the addition of all allowed antiedges in $B(u)$ to the left of e' (see Fig. 6(b)). This property of $B(u)$ ensures that $T(u)$ has two completion sets – the empty set and the set of all allowed antiedges, and therefore $T(u)$ is a valid truth-setting component. ∎

5 Quarantined H-free edge modification

Having satisfaction-testing components and truth-setting components in hand, we will establish in this section the incompressibility of the following "quarantined" versions of H-free edge deletion/completion problems, where we have a restriction on edges/antiedges that can be deleted/added. Results in this section form the base of our main results in the next section where we will lift the quarantine by using "enforcers".

Quarantined H-Free Edge Deletion
Instance: Graph G with a partition of its edges into *forbidden edges* and *allowed edges*, and *parameter* $k \in \mathbf{N}$.
Question: Can we delete at most k allowed edges from G to obtain an H-free graph?

Quarantined H-Free Edge Completion.
Instance: Graph G with a partition of its antiedges into *forbidden antiedges*

and *allowed antiedges*, and *parameter* $k \in \mathbf{N}$.
QUESTION: Can we add at most k allowed antiedges to G to obtain an H-free graph?

We will use ppt-reductions from propagational satisfiability problems to the above quarantined problems to show their incompressibility for H being a 4-cycle, 5-cycle, or 3-connected graph with one or two antiedges. For this purpose, we first give a general scheme for such reductions. As mentioned in the previous section, the main idea is to use satisfaction-testing components $S(x, y, z)$ to represent clauses, and truth-setting components $T(u)$ to connect satisfaction-testing components together to realize a conjunctive formula φ of some propagational function.

Let H be a fixed graph such that H-free edge deletion (resp., edge completion) has a satisfaction-testing component $S(x, y, z)$ for some propagational function $f(x, y, z)$ (i.e., $f_S(x, y, z) = f(x, y, z)$) and a truth-setting component $T(u)$ for Boolean variables u. For an arbitrary instance (φ, k) of PROPAGATIONAL-f SATISFIABILITY on 3-regular conjunctive formulas, we construct an instance (G, k') of QUARANTINED H-FREE EDGE DELETION (resp., EDGE COMPLETION) as follows (see Fig. 7(a) for an example). We remark that the 3-regularity requirement is essential for our ppt-reductions.

Reduction Scheme **PS-QED/QEC**

1. For each clause $f(x, y, z)$ of φ, construct its satisfaction-testing component $S(x, y, z)$. For each constant $c \in \{x, y, z\}$, its corresponding variable-edge (resp., variable-antiedge) in $S(x, y, z)$ is deleted (resp., added) if $c = 1$ and set as forbidden if $c = 0$. Note that all satisfaction-testing components are vertex disjoint.

2. For each variable u of φ, construct its truth-setting component $T(u)$ and identify each of the three variable-edges (resp., variable-antiedges) of $T(u)$ with its corresponding variable-edge (resp., variable-antiedge) in a distinct satisfaction-testing component. Note that φ has exactly three clauses containing u.

3. Let G be the graph obtained from the above construction and call it a *φ-graph*. Let t be the number of allowed edges (resp., allowed antiedges) in $T(u)$, which is a constant, and set $k' = tk$.

It is clear that our construction of (G, k') takes polynomial time as both satisfaction-testing and truth-setting components are of constant size. We now show that the φ-graph G is very close to what we require for representing formula φ, which sets up the framework for our ppt-reductions to QUARANTINED H-FREE EDGE DELETION and COMPLETION.

Lemma 5.1 *Formula φ is satisfiable with at most k true variables iff the φ-graph G contains at most k' allowed edges E^- (resp., allowed antiedges E^+) such that all satisfaction-testing components and truth-setting components in G are H-free after deleting E^- from G (resp., adding E^+ to G).*

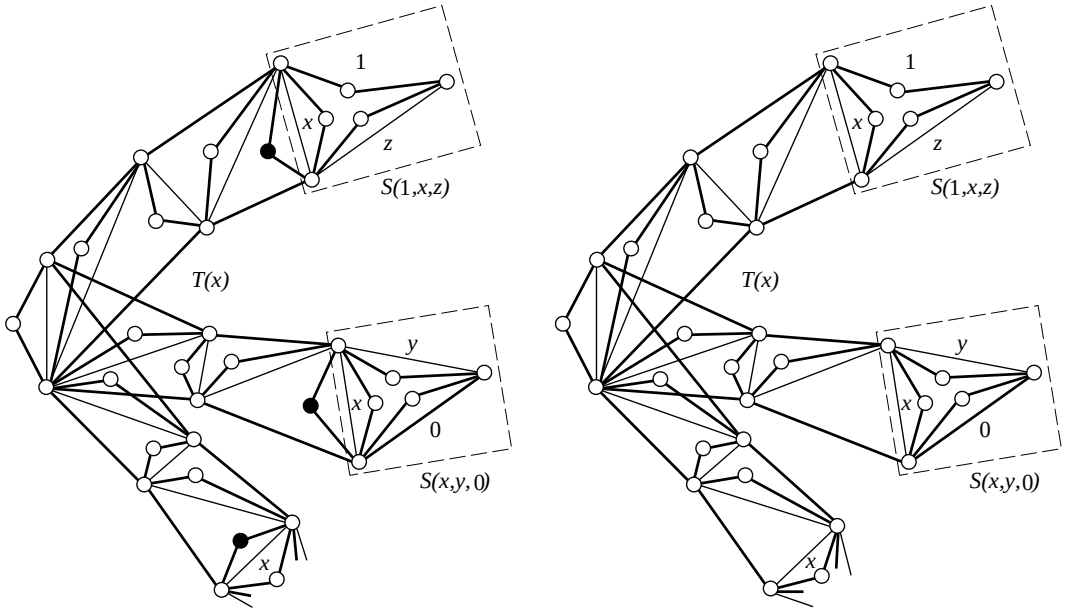

Figure 7: Part of the φ-graph G for Quarantined C_4-Free Edge Deletion that corresponds to a formula φ containing clauses $f(1, x, z)$ and $f(x, y, 0)$, using components in Fig. 3(a) and Fig. 5(a). In the figure, (a) is the original construction, and (b) is a modification from (a) by deleting black vertices in (a).

Proof. We will prove the lemma for edge deletion only, and a proof for edge completion is readily obtained by changing "allowed edges" to "allowed antiedges" and "deletion of edges" to "addition of antiedges" in the following arguments.

($\Rightarrow$) Consider a satisfying truth assignment of φ with $\leq k$ true variables, and let E^- be the allowed edges in all truth-setting components $T(u)$ in G with $u = 1$. Then $|E^-| \leq tk = k'$ as each $T(u)$ contains t allowed edges. By definitions of satisfaction-testing and truth-setting components, we see that each of them is H-free after deleting E^- from G.

($\Leftarrow$) Let E^- be $\leq k' = tk$ edges in G satisfying the assumption in the lemma. By the definition of a truth-setting component, either all or none of its allowed edges are in E^-. Therefore, as each component contains t allowed edges, E^- contains edges from at most k truth-setting components. For each variable u, set $u = 1$ if E^- contains the variable-edge in $T(u)$ and set $u = 0$ otherwise, which gives us a truth assignment with $\leq k$ true variables. For each clause $f_S(x, y, z)$, its corresponding satisfaction-testing component $S(x, y, z)$ is H-free after the deletion of E^-. It follows from the definition of $S(x, y, z)$ that each $f_S(x, y, z)$ is satisfied, and hence φ is satisfied. ∎

With Lemma 5.1 and components of the previous section in hand, we are now ready to establish our incompressibility results for Quarantined H-Free Edge Deletion and Completion. Because of Lemma 5.1, we only need to show that there

is no induced copy of H bestriding satisfaction-testing and truth-setting components.

Theorem 5.2 Quarantined H-Free Edge Deletion *is incompressible for H being C_4, C_5, or any fixed 3-connected graph that is not a complete graph.*

Proof. Our construction of (G, k') from (φ, k) in reduction scheme **PS-QED/QEC** will serve as ppt-reductions from Propagational-f Satisfiability on 3-regular conjunctive formulas to Quarantined H-Free Edge Deletion. In light of Lemma 5.1, we only need to show that every induced H of $G - E^-$, if any, resides inside a satisfaction-testing or truth-setting component. Recall that E^- is the set of all allowed edges in all truth-setting components $T(u)$ in G with $u = 1$. We consider H being 3-connected, C_5, and C_4 in this order.

Case 1. H is 3-connected but not complete.

Because of the 3-connectivity, this general case is actually the easiest one. We use the satisfaction-testing component in Lemma 4.3 and truth-setting component in Lemma 4.5 of the previous section to construct the φ-graph G. Note that $k' = 3hk$, where h is the number of vertices of H. In G, the distance between any two variable-edges of a Boolean variable u is at least h (note that all variables inside each clause are distinct), and any satisfaction-testing component shares at most two vertices with any truth-setting component. Therefore the 3-connectivity of H ensures that any induced H of $G - E^-$, if any, must reside inside a satisfaction-testing or truth-setting component. It follows from Lemma 5.1 that $G - E^-$ is H-free, and we have a required ppt-reduction for the incompressibility of Quarantined H-Free Edge Deletion in this case.

Case 2. $H = C_5$. We use the satisfaction-testing component in Fig. 3(b) and truth-setting component in Fig. 5(b) for the φ-graph G. Note that the truth-setting component contains $t = 13$ allowed edges and thus $k' = 13k$. Since the distance in G between any two variable-edges of a Boolean variable u is at least 4, the only possible induced 5-cycle C in $G - E^-$ must bestride a satisfaction-testing component S and a truth-setting component T.

Clearly the common edge of S and T vanishes in $G - E^-$ as otherwise it would be a chord of C. Therefore we have the situation in Fig. 8 as all allowed edges of T also vanish in $G - E^-$ by the property of T. Since S is C_5-free in $G - E^-$, at least one of edges x and y vanishes in $G - E^-$, which rules out the existence of induced 5-cycle C. Therefore $G - E^-$ is C_5-free and our construction of (G, k') gives a required ppt-reduction to Quarantined C_5-Free Edge Deletion, implying the incompressibility of the problem.

Case 3. $H = C_4$. We use the satisfaction-testing component in Fig. 3(a) and truth-setting component in Fig. 5(a) for the φ-graph G. Note again that $k' = 13k$. Similar to C_5, the only possible induced 4-cycle C in $G - E^-$ bestrides a satisfaction-testing component S and a truth-setting component T. Unfortunately, it is possible in this case: In Fig. 7(a), when variable-edge $x \in E^-$, $G - E^-$ clearly contains an induced 4-cycle formed by the four forbidden edges surrounding x.

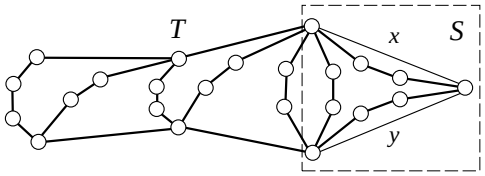

Figure 8: Impossibility of an induced 5-cycle bestriding T and S.

To avoid such induced 4-cycles, we make the following modification in constructing G: when we identify the variable-edge x of $T(x)$ with its corresponding variable-edge in a satisfaction-testing component S, we delete the degree-2 vertex v of $T(x)$ that is adjacent to the two ends of x. Another way of viewing this is that when we identify edge x with its corresponding variable-edge in S, we also identify the degree-2 vertices adjacent to the ends of these two edges. See the graph in Fig. 7(b), which is obtained from the graph in (a) by deleting such degree-2 vertices (indicated by black vertices). The function of v in $T(x)$ is to cause a propagation of edge deletions for C_4-freeness when x is deleted from $T(x)$, which is also accomplished by the degree-2 vertex in S adjacent to the two ends of x. Therefore Lemma 5.1 is still valid.

In $G-E^-$, the induced 4-cycle C would bestride S and T. Therefore, the common edge x of S and T vanishes in $G-E^-$, implying that all allowed edges of T also vanish in $G-E^-$ and we have the situation in Fig. 9. It is easy to see that no such C is possible, and therefore $G-E^-$ is C_4-free and we have a required ppt-reduction for incompressibility. ∎

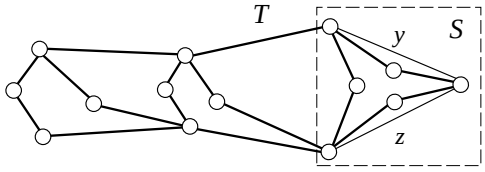

Figure 9: Impossibility of an induced 4-cycle bestriding T and S.

Remark 5.3 The φ-graph G for quarantined C_4-free edge deletion has an additional property that the graph formed by allowed edges contains no 4-cycle as a partial subgraph, which is crucial for the incompressibility of P_5-Free Edge Deletion and Editing in the proof of Theorem 6.8.

For Quarantined H-Free Edge Completion, we have the following similar result. Note that we will obtain the incompressibility of C_5-Free Edge Completion in a much easier way in the next section.

Theorem 5.4 Quarantined H-Free Edge Completion *is incompressible for H being C_4 or any fixed 3-connected graph with at least two antiedges.*

Proof. Like quarantined H-free edge deletion, we also use reduction scheme **PS-QED/QEC** to construct (G, k') from (φ, k), which serves as ppt-reductions from

Propagational-f Satisfiability on 3-regular conjunctive formulas to Quarantined H-Free Edge Completion. In light of Lemma 5.1, we only need to show that every induced H of $G + E^+$, if any, resides inside a satisfaction-testing or truth-setting component. Recall that E^+ is the set of all allowed antiedges in all truth-setting components $T(u)$ in G with $u = 1$.

Same as H-free edge deletion, the case for 3-connected H is easy because of the 3-connectivity. We use the satisfaction-testing component in Lemma 4.3 and truth-setting component in Lemma 4.5 of the previous section to construct the φ-graph G. Again $k' = 3hk$ with h being the number of vertices of H. In $G + E^+$, the distance between any two variable-edges of a Boolean variable u is still at least h, and any satisfaction-testing component shares at most two vertices with any truth-setting component. Therefore the 3-connectivity of H ensures that any induced H of $G+E^+$ must reside inside a satisfaction-testing component or a truth-setting component. It follows from Lemma 5.1 that $G+E^+$ is H-free, and we have a required ppt-reduction for the incompressibility of the problem for 3-connected H with at least two antiedges.

Our proof for $H = C_4$ is almost identical to that for Quarantined C_4-Free Edge Deletion. In constructing φ-graph G, we use the satisfaction-testing component in Fig. 3(c) and truth-setting component in Fig. 5(c) with the following modification similar to that for Quarantined C_4-Free Edge Deletion: when we identify the variable-antiedge x of $T(x)$ with its corresponding variable-antiedge of a satisfaction-testing component, we delete the degree-2 vertex of $T(x)$ adjacent to the two ends of x. Fig. 10 illustrates this modification.

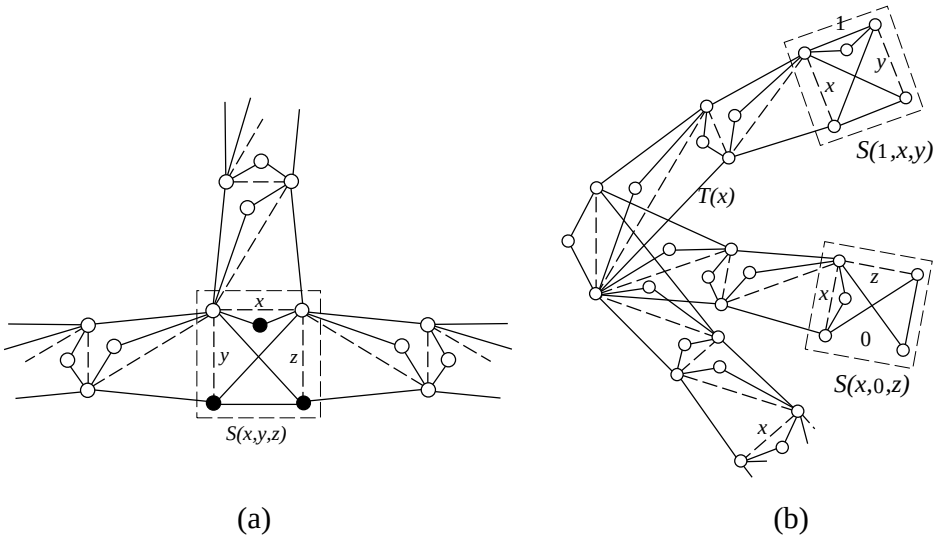

Figure 10: (a). Vicinity of a satisfaction-testing component $S(x, y, z)$ in G. The three black vertices takes the role of the three deleted degree-2 vertices of the three truth-setting components attached to $S(x, y, z)$. (b). Part of the φ-graph G for Quarantined C_4-Free Edge Completion that corresponds to a formula φ containing clauses $f(1, x, y)$ and $f(x, 0, z)$.

Now suppose that $G + E^+$ contains an induced 4-cycle C. Then by Lemma 5.1, C must bestride a satisfaction-testing component S and a truth-setting component T, as the distance in $G + E^+$ between any two variable-edges of a Boolean variable u is still at least 4. It is a routine matter to check that such C is impossible to exist as shown in Fig. 11. Therefore $G + E^+$ is C_4-free and we have a required ppt-reduction for the incompressibility of Quarantined C_4-Free Edge Completion. ∎

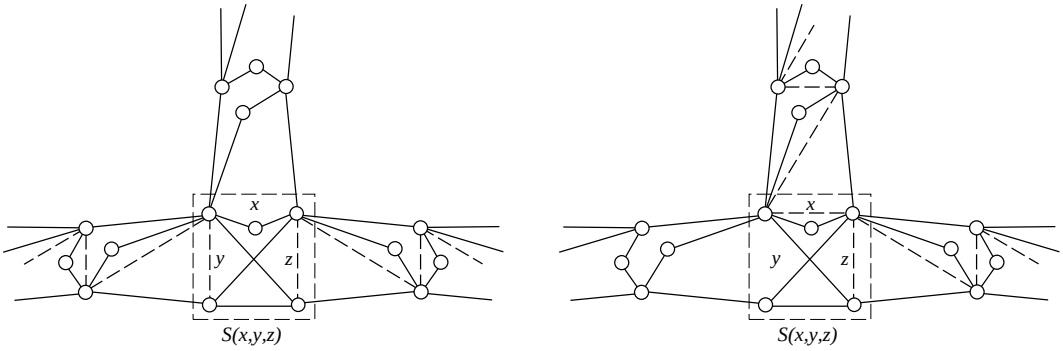

Figure 11: Impossibility of the existence of an induced 4-cycle bestriding a satisfaction-testing component S and a truth-setting component T. Note that if an allowed antiedge of a truth-setting component T is not added to G then no allowed antiedge of T is added to G.

Remark 5.5 The φ-graph G for quarantined C_4-free edge completion has an additional property that the graph formed by allowed antiedges contains no 4-cycle as a partial subgraph, which is crucial for the incompressibility of P_5-Free Edge Completion in the proof of Theorem 6.8.

6 Lifting the quarantine

As the final step towards our main results, we will lift the quarantine for the results in the previous section. For this purpose, we transform the incompressibility of quarantined H-free edge deletion/completion problems into their unquarantined versions by attaching "enforcers" to forbidden edges (resp., forbidden antiedges) to prevent them from being deleted (resp., added). As a big bonus, enforcers also enable us to establish incompressibility of H-free edge editing problems directly from that of H-free edge deletion/completion problems.

Attaching enforcers to edges/antiedges may create new induced copies of H, which can cause great complication and should be avoided. We take this into consideration and define enforcers for edge deletion/completion as follows.

Definition 6.1 *An* H-free deletion enforcer (X, e) *consists of an* H*-free graph* X *and a distinguished edge* e *of* X *such that (a)* $X - e$ *contains an induced* H*, and (b) for any graph* G *vertex-disjoint from* X *and any edge* e' *of* G*, all induced copies of* H *in*

the graph obtained by attaching X to G through identifying e with e' reside entirely inside G.

Definition 6.2 *An* H-free completion enforcer (X, e) *consists of an H-free graph X and a distinguished antiedge e of X such that (a) $X + e$ contains an induced H, and (b) for any graph G vertex-disjoint from X and any antiedge e' of G, all induced copies of H in the graph obtained by attaching X to G through identifying e with e' reside entirely inside G.*

Property (b) of enforcers makes it tricky to construct them, and some H (e.g. path P_l with $l \geq 3$) actually does not have enforcers. In this paper, we just need enforcers for cycles and 3-connected H, which are easily constructed.

Lemma 6.3 *The following statements hold for H being a cycle or a 3-connected graph.*

1. *For any cycle C_t with $t \geq 4$, adding any antiedge e to C_t yields a C_t-free deletion enforcer $(C_t + e, e)$, and deleting any edge e from C_t yields a C_t-free completion enforcer $(C_t - e, e)$.*
2. *For any 3-connected H with an antiedge e, $(H + e, e)$ is an H-free deletion enforcer.*
3. *For any 3-connected H and any edge e, $(H - e, e)$ is an H-free completion enforcer.*

Proof. It is obvious that enforcers in the lemma satisfy property (a) of enforcers, and it is straightforward to confirm that they also satisfy property (b) as an enforcer has exactly two vertices connecting to the outside. ∎

Once we have an enforcer (X, e), we can easily prevent an edge/antiedge e' in the input graph G from being deleted/added for H-free deletion/completion problems with $\leq k$ deletions/additions.

Definition 6.4 *To* k-forbid an edge (resp., antiedge) e', *we attach k vertex-disjoint copies of an H-free deletion enforcer (resp., completion enforcer) (X, e) to e' by identifying e with e'.*

Let G' be a graph obtained from G by k-forbidding each edge (resp., antiedge) of a set F of edges (resp., antiedges) using H-free deletion (resp., completion) enforcers. Then the definitions of enforcers ensure that

1. all induced copies of H in G' reside inside G, and
2. for graph G', no edge/antiedge in F can be deleted/added as its deletion/addition will create k induced copies of H that share no common edge/antiedge.

Therefore we can use enforcers to specify forbidden edges/antiedges F in quarantined H-free edge deletion/completion problems, which gives us equivalent unquarantined versions of H-free edge deletion/completion problems. Enforcers also enable us to force edge editing problems to be edge deletion problems or edge completion problems, which yields incompressibility of edge editing problems directly from that of edge deletion/completion problems.

Lemma 6.5 *For any fixed graph H, the following statements hold:*

1. H-Free Edge Deletion *is incompressible if* Quarantined H-Free Edge Deletion *is incompressible and there exists an H-free deletion enforcer.*
2. H-Free Edge Completion *is incompressible if* Quarantined H-Free Edge Completion *is incompressible and there exists an H-free completion enforcer.*
3. H-Free Edge Editing *is incompressible if either*
   (a) H-Free Edge Deletion *is incompressible and there exists an H-free completion enforcer or*
   (b) H-Free Edge Completion *is incompressible and there exists an H-free deletion enforcer.*

Proof. Let F be forbidden edges (resp., antiedges) in a quarantined graph G. By k-forbidding every edge (resp., antiedge) in F by H-free deletion enforcers (resp., completion enforcers), we can obtain an equivalent instance (G', k) of H-Free Edge Deletion (resp., Completion) of instance (G, k) for Quarantined H-Free Edge Deletion (resp., Completion) in polynomial time. Therefore the incompressibility of H-Free Edge Deletion (resp., Completion) follows from that of Quarantined H-Free Edge Deletion (resp., Completion) and Theorem 2.4.

For H-Free Edge Editing, we can construct graph G' from the input graph G of H-Free Edge Deletion by k-forbidding every antiedge of G using H-free completion enforcers. Then H-Free Edge Editing on G' is equivalent to H-Free Edge Deletion on G as the only way to destroy induced copies in G' by $\leq k$ edge deletions/additions is to destroy those inside G of G' by $\leq k$ edge deletions. Similarly, we can construct graph G' from the input graph G of H-Free Edge Completion by k-forbidding every edge of G using H-free deletion enforcers. Then H-Free Edge Editing on G' is equivalent to H-Free Edge Completion on G as the only way to destroy induced copies in G' by $\leq k$ edge deletions/additions is to destroy those inside G of G' by $\leq k$ edge additions. Therefore the incompressibility of H-Free Edge Editing follows from the assumptions of the lemma. ∎

There are also fundamental connections between H-free edge modification problems, which are easy to see yet very useful in dealing with the incompressibility of these problems.

Lemma 6.6 *For any fixed H, H-Free Edge Deletion is incompressible iff $\overline{H}$-Free Edge Completion is incompressible, and H-Free Edge Editing is incompressible iff $\overline{H}$-Free Edge Editing is incompressible.*

Proof. For a graph G, H-Free Edge Deletion on G is equivalent to $\overline{H}$-Free Edge Completion on $\overline{G}$, and H-Free Edge Editing on G is equivalent to $\overline{H}$-Free Edge Editing on $\overline{G}$. ∎

We are now ready to give a full characterization of the incompressibility of H-free edge modification problems for 3-connected H.

Theorem 6.7 *Let H be a fixed 3-connected graph and assume* NP $\not\subseteq$ coNP/poly.

1. *H-Free Edge Completion admits no polynomial compression (hence no polynomial kernel) iff H has ≥ 2 antiedges.*
2. *H-Free Edge Deletion (resp., H-Free Edge Editing) admits no polynomial compression (hence no polynomial kernel) iff H is not a complete graph.*

Proof. The incompressibility follows from that of quarantined H-free edge modification problems (Theorem 5.2 and Theorem 5.4), the existence of H-free deletion/completion enforcers (Lemma 6.3(2) and (3)), and Lemma 6.5.

For the cases that admit polynomial kernels, H-Free Edge Completion is trivial for H being a complete graph, and easily solved in $O(kn^t)$ for H being $K_t - e$ for some constant t (for each H found in G just add the missing edge): both solvable in polynomial time and thus have trivial kernels.

When H is a complete graph K_t for some constant t, K_t-Free Edge Editing is equivalent to K_t-Free Edge Deletion. The latter admits a polynomial kernel by reducing it to Hitting Set where each subset has size $\binom{t}{2}$, and we can make the kernel an instance of K_t-Free Edge Deletion if one insists [7]. ∎

In general, it seems difficult to establish the incompressibility of H-free edge modification problems when H has connectivity at most 2. However, for the two basic cases of 1-connected and 2-connected H, namely, paths and cycles, we have a complete characterization for incompressibility. Note that Gramm et al. [9] and Guillemot et al. [11, 12], respectively, have constructed polynomial kernels for P_3- and P_4-free edge modification problems. Also note that Guillemot et al. [11] have shown the incompressibility of H-Free Edge Deletion for H being a path P_l with $l \geq 13$ or a cycle C_t with $t \geq 12$, which has been improved recently to $l \geq 7$ and $t \geq 4$ by Guillemot et al. [12] with involved and complicated proofs.

Theorem 6.8 *Let H be a fixed path or cycle and assume* NP $\not\subseteq$ coNP/poly. *Then H-Free Edge Deletion (resp., H-Free Edge Completion, and H-Free Edge Editing) admits no polynomial compression (hence no polynomial kernel) iff H has at least 4 edges.*

Proof. As noted earlier, polynomial kernels exist for P_l-free edge modification problems with $l \leq 4$ [9, 11]. Since $C_3 = K_3$, polynomial kernels exist for C_3-free edge modification problems as discussed in the proof of Theorem 6.7.

For the incompressibility part, if $H = P_l$ or $H = C_l$ with $l \geq 6$ then $\overline{H}$ is 3-connected with at least two antiedges, and the incompressibility of these problems follow from that of $\overline{H}$-free edge modification problems (Theorem 6.7) and Lemma 6.6.

For $H = C_4$, the incompressibility follows from that of quarantined C_4-free edge modification problems (Theorem 5.2 and Theorem 5.4) using Lemma 6.5 with the aid of C_4-free deletion/completion enforcers (Lemma 6.3(1)).

For $H = C_5$, the incompressibility of C_5-Free Edge Deletion follows from that of Quarantined C_5-Free Edge Deletion (Theorem 5.2) and the existence of C_5-deletion enforcer (Lemma 6.3(1) and Lemma 6.5(1)). It follows that C_5-Free Edge Editing is incompressible with the aid of a C_5-free completion enforcer (Lemma 6.3(1)), and so is C_5-Free Edge Completion because of Lemma 6.6 and the fact that $\overline{C_5} = C_5$.

For the remaining case $H = P_5$, we consider its complement $\overline{P_5}$ — the house graph in Fig. 12(a), and give ppt-reductions from quarantined C_4-free edge modification problems with special properties.

For House-Free Edge Deletion, we construct a ppt-reduction from Quarantined C_4-Free Edge Deletion. For a quarantined graph G with forbidden edges F, we construct a graph G' by attaching a copy of the gadget in Fig. 12(c) to each forbidden edge $e \in F$ by identifying e with edge uv of the gadget. Note that all induced 4-cycles of G' are inside G. By Remark 5.3, we may assume that no four allowed edges in G form a 4-cycle. We show that G has a C_4-free k-deletion set iff G' has a house-free k-deletion set.

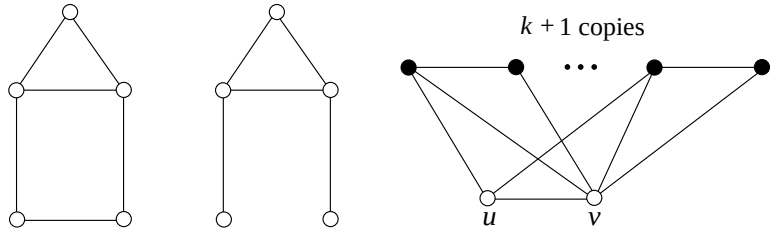

Figure 12: (a) The house graph $\overline{P_5}$, (b) a house-free completion enforcer, and (c) the gadget formed from $k + 1$ copies of the diamond graph, where edge uv will be identified with a forbidden edge to prevent the edge from being deleted.

($\Rightarrow$) Observe that deleting from G' any allowed edges of G can only create chordless 4-cycles entirely inside G. Since all induced 4-cycles of G' are inside G, any C_4-free k-deletion set of G, which consists of allowed edges only, is actually also a C_4-free k-deletion set of G', and hence a house-free k-deletion set of G'.

($\Leftarrow$) Let E^* be a house-free k-deletion set of G'. Then E^* contains no forbidden edge, for otherwise it would create too many induced houses to be destroyed by the remaining $\leq k - 1$ edge deletions. Let E^- be allowed edges of G in E^*. If $G - E^-$ contains a chordless 4-cycle C, then it contains a forbidden edge $e \in F$ as no four

allowed edges of G form a 4-cycle. Since some black vertex in Fig. 12(c) survives the deletion of E^*, it forms an induced house in $G' - E^*$ with C, a contradiction to $G' - E^*$ being house-free. Therefore E^- is a required C_4-free k-deletion set of G.

For House-Free Edge Completion, it suffices to show the incompressibility of Quarantined House-Free Edge Completion because of the house-free completion enforcer in Fig. 12(b). For this purpose, we give a ppt-reduction from Quarantined C_4-Free Edge Completion. For a quarantined graph G with allowed antiedges A, we construct from G a graph G' by adding, for each edge e of G, a new vertex v_e and two edges connecting v_e with the ends of e. Make G' a quarantined graph by also setting A as its allowed antiedges. By Remark 5.5, we may assume that no four allowed antiedges in A form a 4-cycle. We show that G has a C_4-free k-completion set iff G' has a house-free k-completion set.

($\Rightarrow$) Observe that adding to G' any allowed antiedges from A can only create chordless 4-cycles entirely inside G. Since all induced 4-cycles of G' are inside G, any C_4-free k-completion set of G, which consists of allowed antiedges only, is actually also a C_4-free k-completion set of G', and hence a house-free k-completion set of G'.

($\Leftarrow$) Let $E^+ \subseteq A$ be a house-free completion set of G'. If $G + E^+$ contains a chordless 4-cycle C, then C contains an edge e of G as no four allowed antiedges form a 4-cycle. But then vertex v_e and C together form an induce house in $G + E^+$, a contradiction. Therefore E^+ is a required C_4-free k-edge completion set of G.

Finally, the incompressibility of C_4-Free Edge Editing follows from that of C_4-Free Edge Deletion and the house-free completion enforcer in Fig. 12(b). By Lemma 6.6, we conclude that P_5-Free Edge Deletion, Completion, and Editing are all incompressible. ∎

7 Conclusion: towards dichotomy theorems

Our ambition in studying H-free edge modification problems aims at complete characterizations of their incompressibility in terms of the structure of H. We feel that, except $H = K_t$ or $K_t - e$, H-free edge modification problems may only admit polynomial kernels for a handful of small H, perhaps with at most 5 vertices. This is attested by our results for 3-connected H, simpliest 2-connected H (i.e., cycles), and simpliest 1-connected H (i.e., paths).

In fact, our incompressibility for 3-connected H reveals much more than it looks in Theorem 6.7. Because of Lemma 6.6, Theorem 6.7 implies the following result which covers a very extensive range of H.

Corollary 7.1 *For any fixed H, H-Free Edge Deletion (resp., Completion, and Editing) is incompressible whenever H or $\overline{H}$ is 3-connected with at least two antiedges.*

Indeed, we can deduce from the above result that H-free edge modification problems are incompressible for most disconnected H and for most trees H, as $\overline{H}$ is

3-connected for most such H. In fact for trees H, we know that H-free edge modification problems are incompressible for all but a small number of trees [7]. In this regards, $H = K_{1,3}$ (the claw graph) is a very challenging case.

Problem 7.2 *Determine whether claw-free edge modification problems admit polynomial kernels.*

The settlement of the above problem, especially if the answer is incompressible, will be important for the study of H-free edge modification problems with H being trees.

Problem 7.3 *Give a complete characterization of trees T for which T-free edge modification problems admit polynomial kernels.*

For general H, we pretty much know how blocks and connected components in H affect the incompressibility of H-free modification problems [7]. This leaves 2-connected H a very important case. Note that for $H = K_4 - e$ (the diamond graph), H-Free Edge Deletion admits a polynomial kernel [7].

Conjecture 7.4 *For any fixed 2-connected H with more than 5 vertices, H-Free Edge Deletion and Editing are incompressible unless H is complete, and H-Free Edge Completion is incompressible unless H has at most one antiedge.*

It is not hard to prove the above conjecture for certain 2-connected H, however, it seems quite difficult to control the occurrences of induced H in ppt-reductions, and clever new ideas will be needed if we are after a complete settlement of the conjecture. One possible way is to consider the following conjecture that enables us to examine small H only.

Conjecture 7.5 *For any fixed H, H-Free Edge Deletion (resp., Completion, and Editing) is incompressible whenever H contains an induced subgraph H' such that H'-Free Edge Deletion (resp., Completion, and Editing) is incompressible.*

Of course, H-free edge modification problems are only a starting point for edge modification problems regarding general hereditary properties described by a family $\mathcal{F}$ of forbidden induced subgraphs. We hope that our work will also shed light on the incompressibility of $\mathcal{F}$-free edge modification problems.

Problem 7.6 *Investigate connections between the incompressibility of $\mathcal{F}$-free edge modification problems and that of H-free edge modification problems for all $H \in \mathcal{F}$. In particular, does the incompressibility of H- and H'-free edge modification problems imply that of $\{H, H'\}$-free edge modification problem? If not, characterize those H and H' that do.*

We remark that if $|\mathcal{F}| \geq 3$, then $\mathcal{F}$-free edge modification problems may admit polynomial kernel while H-free edge modification problems are incompressible for every $H \in \mathcal{F}$. For instance, $\mathcal{F} = \{2K_2, C_4, C_5\}$ (note that $\mathcal{F}$-graphs are exactly split graphs and $2K_2 = \overline{C_4}$) for which $\mathcal{F}$-Free Edge Deletion admits a kernel of size $O(k^4)$ [13].

Finally, we finish the paper with a theorem in connection with the last problem, which also unveils the hidden power of our general scheme for ppt-reductions from Propagational-f Satisfiability.

Theorem 7.7 *Let $\mathcal{F}$ be a finite set of forbidden induced subgraphs, and assume* NP $\not\subseteq$ coNP/poly*. If all graphs in $\mathcal{F}$ are 3-connected and there is an $H \in \mathcal{F}$ with fewest edges such that $H + e \notin \mathcal{F}$ for some antiedge e of H, then $\mathcal{F}$-Free Edge Deletion admits no polynomial compression and hence no polynomial kernel.*

Proof. First let us consider the antiedge e. If all edges of H are incident with an end of e, then H is a bipartite graph and the two ends of e is a 2-cut of H, contradicting the 3-connectivity of H. Therefore H has an edge e' that shares no vertex with antiedge e. Let l be the maximum number of vertices for graphs in $\mathcal{F}$.

We use our construction of φ-graph G for H-Free Edge Deletion in Theorem 5.2 with the following specification/modification:

1. For satisfaction-testing component $S(x, y, z) = H + x$, we set $x = e$ and $\{y, z\}$ two arbitrary edges of H.
2. For truth-setting component $T(u)$, we use l copies of basic unit U to construct the basic chain $B(u)$.

We show that these two components are valid. Let h be the number of vertices of H, and we show first that the basic unit U is $\mathcal{F}$-free. By the assumption for H, $U = H + e \notin \mathcal{F}$. For any induced proper subgraph U' of U, U' contains at most $h - 2$ edges as U is also 3-connected. Therefore $U' \notin \mathcal{F}$ as H has the fewest edges in $\mathcal{F}$, and it follows that U is $\mathcal{F}$-free. Now we note that $U - \{e, e'\}$ is also $\mathcal{F}$-free as it has $h - 1$ edges. Therefore $T(u)$ is a valid truth-setting component for $\mathcal{F}$-free edge deletion.

The satisfaction-testing component $S(x, y, z)$ is the same as the basic unit U except different allowed edges, and thus is also $\mathcal{F}$-free. Deletion of x will cause the deletion of y or z, and the resulting graph has only $\leq h - 1$ edges and thus is also $\mathcal{F}$-free. It follows that

$$f_S(0, 0, 0) = f_S(1, 0, 1) = f_S(1, 1, 0) = f_S(1, 1, 1) = 1 \text{ but } f_S(1, 0, 0) = 0.$$

Therefore f_S is propagational and $S(x, y, z)$ is a valid satisfaction-testing component for $\mathcal{F}$-free edge deletion.

In G, the distance between any two variable-edges of a Boolean variable u is at least l, and thus the 3-connectivity of H ensures that any induced copy of any $H' \in \mathcal{F}$ must reside inside a satisfaction-testing component or a truth-setting component. The

same arguments of Theorem 5.2 establishes the incompressibility of Quarantined $\mathcal{F}$-Free Edge Deletion. Finally we use the H-free deletion enforcer, which is also $\mathcal{F}$-free, in Lemma 6.3(2) on G to lift the quarantine and obtain the incompressibility of $\mathcal{F}$-Free Edge Deletion. ∎

Acknowledgements

The paper is partially based on the MPhil Thesis of the 2nd author under the supervision of the 1st author. We thank the two anonymous reviewers for their constructive suggestions.

References

[1] H. L. Bodlaender, L. Cai, J. Chen, M. R. Fellows, J. A. Telle, and D. Marx, Open problems in parameterized and exact computation — IWPEC 2006. Utrecht University Technical Report UU-CS-2006-052, 2006.

[2] H. L. Bodlaender, R. G. Downey, M. R. Fellows, D. Hermelin, On problems without polynomial kernels. *Journal of Computer and System Sciences* 75(8), 423–434, 2009.

[3] H. L. Bodlaender, B. Jansen, S. Kratsch, Kernelization lower bounds by cross-compositions, *SIAM Journal on Discrete Mathematics* 28(1), 277-305, 2014.

[4] H. L. Bodlaender, S. Thomassé, A. Yeo, Kernel bounds for disjoint cycles and disjoint paths, *Theoretical Computer Science* 412(35), 4570-4578, 2011.

[5] L. Cai, Fixed-parameter tractability of graph modification problems for hereditary properties. *Information Processing Letters* 58(4), 157–206, 1996.

[6] L. Cai and Y. Cai, Incompressibility of H-free edge modification, In: G. Gutin and S. Szeider (eds.), 8th International Symposium on Parameterized and Exact Computation (IPEC 2013), Lecture Notes in Computer Science, vol. 8246, 84-96, 2013.

[7] Y. Cai, Polynomial Kernelisation of H-free Edge Modification Problems, MPhil Thesis, Department of Computer Science and Engineering, The Chinese University of Hong Kong, Hong Kong SAR, China, 2012. `http://www.uni-marburg.de/fb12/ps/team/cai-masterarbeit.pdf`.

[8] L. Fortnow and R. Santhanam, Infeasibility of instance compression and succinct PCPs for NP. *Journal of Computer and System Sciences* 77(1), 91–106, 2011.

[9] J. Gramm, J. Guo, F. Hüffner, and R. Niedermeier, Graph-modeled data clustering: fixed parameter algorithms for clique generation. *Theory of Computing Systems* 38(4), 373–392, 2005.

[10] M. R. Garey, D. S. Johnson, and L. Stockmeyer, Some simplified NP-complete graph problems, Theoretical Computer Science 1(3), 237-267, 1976.

[11] S. Guillemot, C. Paul, and A. Perez, On the (non-)existence of polynomial kernels for P_l-free edge modification problems. In: V. Raman and S. Saurabh (eds.), 5th International Symposium on Parameterized and Exact Computation (IPEC 2010), Lecture Notes in Computer Science, vol. 6478, 147–157, 2010.

[12] S. Guillemot, F. Havet, C. Paul, and A. Perez, On the (non-)existence of polynomial kernels for P_l-free edge modification problems. *Algorithmica* 65(4), 900-926, 2013.

[13] J. Guo, Problem kernels for NP-complete edge deletion problems: split and related graphs, In: T. Tokuyama (ed.), 18th International Symposium on Algorithms and Complexity (ISAAC 2007), Lecture Notes in Computer Science, vol. 4835, 915-926, 2007.

[14] S. Kratsch and M. Wahlström, Two edge modification problems without polynomial kernels. *Discrete Optimization* 10, 193-199, 2013.