



HAL
open science

On the Parameterized Complexity of Graph Modification to First-Order Logic Properties

Fedor V. Fomin, Petr A. Golovach, Dimitrios M. Thilikos

► **To cite this version:**

Fedor V. Fomin, Petr A. Golovach, Dimitrios M. Thilikos. On the Parameterized Complexity of Graph Modification to First-Order Logic Properties. *Theory of Computing Systems*, 2020, 64 (2), pp.251-271. 10.1007/s00224-019-09938-8 . hal-03002648

HAL Id: hal-03002648

<https://hal.science/hal-03002648>

Submitted on 20 Nov 2020

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

On the Parameterized Complexity of Graph Modification to first-order logic Properties*

Fedor V. Fomin[†] Petr A. Golovach[†] Dimitrios M. Thilikos[‡]

February 27, 2019

Abstract

We establish connections between parameterized/kernelization complexity of graph modification problems and expressibility in logic. For a first-order logic formula φ , we consider the problem of deciding whether an input graph can be modified by removing/adding at most k vertices/edges such that the resulting modification has the property expressible by φ . We provide sufficient and necessary conditions on the structure of the prefix of φ specifying when the corresponding graph modification problem is fixed-parameter tractable (parameterized by k) and when it admits a polynomial kernel.

Keywords: First-order logic, graph modification, parameterized complexity, descriptive complexity, kernelization

1 Introduction

A variety of algorithmic graph problems, called *modification problems*, can be formulated as problems of modifying a graph such that the resulting graph satisfies some fixed desired property. The study of graph modification problems is one of the most popular trends in graph algorithms, and in particular, in parameterized complexity. One of the classic results about graph modification problems is the work of Lewis and Yannakakis [16], which provides necessary and sufficient conditions (assuming $P \neq NP$) of polynomial time solvability of vertex-removal problems for hereditary properties. For other types of graph modification problems, like edge-removal problems [21], no such dichotomy is known. For the past 30 years graph modification problems served as a strong inspiration for developing new methods and techniques in parameterized/kernelization algorithms and complexity, see the books [6, 7, 9, 17] for an overview of the area.

In this paper we approach graph modification problems from the perspective of descriptive complexity. Descriptive complexity is the field of logic which studies the relations between computational complexity and expressibility in logic. The classic example of a theorem in descriptive complexity is the theorem of Fagin [8] asserting that a property

*The two first authors have been supported by the Research Council of Norway via the projects “CLASSIS” and “MULTIVAL”. The third author has been supported by projects DEMOGRAPH (ANR-16-CE40-0028) and ESIGMA (ANR-17-CE23-0010). All authors have been supported by the Research Council of Norway and the French Ministry of Europe and Foreign Affairs, via the Franco-Norwegian project PHC AURORA 2019.

Emails of authors: {fedor.fomin, petr.golovach}@ii.uib.no, sedthilk@thilikos.info.

[†]Department of Informatics, University of Bergen, Norway.

[‡]AIGCo project team, CNRS, LIRMM, Université de Montpellier, Montpellier, France.

of graphs is in NP if and only if it is definable by an existential second-order formula. We refer to the recent book of Grohe [12] for a modern overview of descriptive complexity. The significant amount of research in descriptive complexity is devoted to the study of prefix classes of certain logics. A prefix class is a syntactic fragment of first-order or second-order logic with formulas in prenex normal form and imposed constraints on the patterns of quantifiers in formulas. For example, the study of prefix classes of first-order logic is provided in the book of Börger, Grädel, and Gurevich [4], see also the work of Gottlob, Kolatis and Schwentick [11] on characterizing the computational complexity of prefix classes of second-order logic.

Our results. Let ϕ be an FOL formula on (undirected) graphs in prenex normal form. In particular, $\phi = \mathbf{Q}_1x_1\mathbf{Q}_2x_2\cdots\mathbf{Q}_tx_t\chi$, where t is some constant, each $\mathbf{Q}_i \in \{\forall, \exists\}$ is a quantifier, x_i is a variable, and χ is a quantifier-free part that depends on the variables x_1, \dots, x_t . We consider the following generic problems (we use “ $-$ ” for the vertex/edge removal, “ $+$ ” for the edge addition and “ Δ ” for the symmetric difference).

<p>VERTEX-REMOVAL TO ϕ</p> <p>Input: A graph G and an integer k.</p> <p>Question: Does there exist a vertex set $S \subseteq V(G)$ with $S \leq k$ such that $G - S \models \phi$?</p>	<p>Parameter: k</p>
<p>EDGE-REMOVAL TO ϕ</p> <p>Input: A graph G and an integer k.</p> <p>Question: Does there exist an edge set $F \subseteq E(G)$ with $F \leq k$ such that $G - F \models \phi$?</p>	<p>Parameter: k</p>
<p>EDGE-COMPLETION TO ϕ</p> <p>Input: A graph G and an integer k.</p> <p>Question: Does there exist $F \subseteq \binom{V(G)}{2} \setminus E(G)$ with $F \leq k$ such that $G + F \models \phi$?</p>	<p>Parameter: k</p>
<p>EDGE-EDITING TO ϕ</p> <p>Input: A graph G and an integer k.</p> <p>Question: Does there exist $F \subseteq \binom{V(G)}{2}$ with $F \leq k$ such that $G \Delta F \models \phi$?</p>	<p>Parameter: k</p>

For example, for $\phi = \forall u \forall v \neg(u \sim v)$, VERTEX-REMOVAL TO ϕ is equivalent to VERTEX COVER that is the graph modification problem asking whether one can remove at most k vertices such that the resulting graph has no edges (we use $u \sim v$ for the adjacency predicate). More generally, any vertex-removal problem to a graph class characterized by a finite set of forbidden subgraphs, can be expressed as VERTEX-REMOVAL TO ϕ for some ϕ with only \forall quantifications over variables, where the number of variables is the maximum number of vertices of a forbidden graph. Clearly, using FOL, we are able to express other properties. For example, the property that the diameter of a graph is at most two cannot be expressed using forbidden subgraphs but can easily be written as the FOL formula $\forall u \forall v \exists w [(u = v) \vee (u \sim v) \vee ((u \sim w) \wedge (v \sim w))]$. Similarly, the edge variants of modification problems to ϕ capture quite a few interesting and well-studied problems like CLUSTER EDITING, where the task is to change at most k adjacencies in the graph resulting in a disjoint union of cliques.

We consider modification problems, where the specification of a prefix class of formula ϕ is defined according to the arithmetic hierarchy (also known as Kleene-Mostowski hierarchy) used for classifications of the formulas in the first-order arithmetic language (see, e.g., [18]). We define prefix classes according to alternations of quantifiers, that is, switchings from \forall to \exists or vice versa in the prefix string of the formula. We allow a formula to have *free*, i.e., non-quantified, variables. Let $\Sigma_0 = \Pi_0$ be the classes of FOL-formulas

without quantifiers. For a positive integer s , the class Σ_s contains formulas that could be written in the form

$$\phi = \exists x_1 \exists x_2 \cdots \exists x_t \psi,$$

where ψ is a Π_{s-1} -formula, t is some integer, and x_1, \dots, x_t are free variables of ψ . Respectively, Π_s consists of formulas

$$\phi = \forall x_1 \forall x_2 \cdots \forall x_t \psi,$$

where ψ is a Σ_{s-1} -formula and x_1, \dots, x_t are free variables of ψ . Note that we allow $t = 0$, which implies that for $s' > s$, $\Sigma_s \cup \Pi_s \subseteq \Sigma_{s'} \cap \Pi_{s'}$.

We establish a number of algorithmic results about modification problems where the target property is definable in FOL. We complement these results by lower bounds, which in combination provide a neat dichotomy theorems about the parameterized complexity of such problems. Hence we establish sufficient and necessary conditions on the prefix classes of FOL-formulas such that the corresponding graph modification problems are fixed-parameter tractable and/or admit a polynomial kernel.

Our first result shows the following dichotomy (subject to $W[2] \neq FPT$) for VERTEX-REMOVAL TO ϕ , depending on the structure of the prefix class of ϕ .

Theorem 1.

- (i) For every $\phi \in \Sigma_3$ without free variables, VERTEX-REMOVAL TO ϕ is FPT.
- (ii) There is $\phi \in \Pi_3$ without free variables such that VERTEX-REMOVAL TO ϕ is $W[2]$ -hard.

In other words, if the prefix of an FOL-formula ϕ has at most two alternations of quantifiers and, in the case of exactly two alternations, if the first quantifier is \exists , then VERTEX-REMOVAL TO ϕ is FPT. For each other type of quantifier alternations, there exists a formula for which the problem becomes $W[2]$ -hard.

For kernelization complexity of VERTEX-REMOVAL TO ϕ , we establish the following dichotomy.

Theorem 2.

- (i) For every $\phi \in \Sigma_1 \cup \Pi_1$ without free variables, VERTEX-REMOVAL TO ϕ admits a polynomial kernel.
- (ii) There is $\phi \in \Sigma_2$ ($\phi \in \Pi_2$) without free variables such that VERTEX-REMOVAL TO ϕ admits no polynomial kernel unless $NP \subseteq coNP/poly$.

For edge-modification problems we prove the following.

Theorem 3.

- (i) For every $\phi \in \Sigma_2$ without free variables, EDGE-REMOVAL TO ϕ , EDGE-COMPLETION TO ϕ , and EDGE-EDITING TO ϕ are FPT.
- (ii) There exists $\phi \in \Pi_2$ without free variables such that EDGE-REMOVAL TO ϕ (respectively, EDGE-COMPLETION TO ϕ and EDGE-EDITING TO ϕ) is $W[2]$ -hard.

We observe that if $\phi \in \Sigma_1$, then all considered problems can be solved in polynomial time. Clearly, this means that they are FPT and have trivial polynomial kernels. We complement this with lower bounds and summarize these results in the following theorem.

Theorem 4.

- (i) For every $\phi \in \Sigma_1$ without free variables, EDGE-REMOVAL TO ϕ , EDGE-COMPLETION TO ϕ , and EDGE-EDITING TO ϕ admit polynomial kernels.
- (ii) There exists $\phi \in \Pi_1$ without free variables such that EDGE-REMOVAL TO ϕ (respectively, EDGE-COMPLETION TO ϕ and EDGE-EDITING TO ϕ) admits no polynomial kernel unless $\text{NP} \subseteq \text{coNP}/\text{poly}$.

This paper is organized as follows. In Section 2, we introduce basic notions and state some auxiliary results. In Section 3, we obtain the algorithmic upper bounds, that is, we show the claims (i) of Theorems 1–4. In Section 4, we complement these results by the lower bounds given in the claims (ii) of Theorems 1–4. We conclude with Section 5, where we discuss some possible extension of our results and mention some directions for further research.

2 Preliminaries

Sets. We use \mathbb{N} to denote the set of all non-negative numbers. Given some $k \in \mathbb{N}$, we denote $[k] = [1, k]$. Given a set A , we denote by 2^A the set of all its subsets and we define $\binom{A}{2} := \{e \mid e \in 2^A \wedge |e| = 2\}$. We denote by $\mathbf{a} = \langle a_1, \dots, a_r \rangle$ a sequence of elements of a set A and call \mathbf{a} an r -tuple of simply a tuple. Note that the elements of \mathbf{a} not necessarily pairwise distinct. We denote by \mathbf{ab} the concatenation of tuples \mathbf{a} and \mathbf{b} .

Graphs. All graphs in this paper are undirected, loop-less, and without multiple edges unless it is explicitly specified to be different. Given a graph G , we denote by $V(G)$ its vertex set and by $E(G)$ its edge set. For an edge $e = \{x, y\} \in E(G)$, we use instead the notation $e = xy$, that is equivalent to $e = yx$. We denote $|G| = |V(G)|$. Throughout the paper we use n to denote $|G|$ if it does not create confusion. For a vertex v , $d_G(v)$ denotes the degree of v . For any set of vertices $S \subseteq V(G)$, we denote by $G[S]$ the subgraph of G induced by the vertices from S . We also define $G - S := G[V(G) \setminus S]$. Given an edge set $F \subseteq E(G)$, we denote $G - F = (V(G), E(G) \setminus F)$. Also, given a set $F \subseteq \binom{V(G)}{2} \setminus E(G)$, i.e., F is a set of pairs of vertices that are not edges of G , we define $G + F = (V(G), E(G) \cup F)$, and for $F \subseteq \binom{V(G)}{2}$, we define $G \Delta F = (V(G), E(G) - E(G) \cap F + F \setminus E(G))$.

Formulas. In this paper we deal with logic formulas on graphs. In particular we will deal with formulas of first-order logic (FOL). The syntax of FOL-formulas on graphs includes the logical connectives \vee, \wedge, \neg , variables for vertices, the quantifiers \forall, \exists that are applied to these variables, the predicate $u \sim v$, where u and v are vertex variables and the interpretation is that u and v are adjacent, and the equality of variables representing vertices. It is also convenient to assume that we have the logical connectives \rightarrow and \leftrightarrow . An FOL-formula ϕ is in *prenex normal form* if it is written as $\phi = \mathbf{Q}_1 x_1 \mathbf{Q}_2 x_2 \cdots \mathbf{Q}_t x_t \chi$ where each $\mathbf{Q}_i \in \{\forall, \exists\}$ is a quantifier, x_i is a variable, and χ is a quantifier-free part that depends on the variables x_1, \dots, x_t . Then $\mathbf{Q}_1 x_1 \mathbf{Q}_2 x_2 \cdots \mathbf{Q}_t x_t$ is referred as the *prefix* of ϕ . From now on, when we mention the term ‘‘FOL-formula’’, we mean an FOL-formula on graphs that is in prenex normal form. For an FOL-formula ϕ without free variables and a graph G , we write $G \models \phi$ to denote that ϕ evaluates to *true* on G .

For technical reasons, we extend FOL-formulas on graphs to structures of a special type. We say that a pair (G, \mathbf{v}) , where $\mathbf{v} = \langle v_1, \dots, v_r \rangle$ is an r -tuple of vertices of G , is an r -structure. Let ϕ be an FOL-formula without free variables and let $\mathbf{x} = \langle x_1, \dots, x_r \rangle$ be an

r -tuple of pairwise distinct variables of ϕ . We denote by $\phi[\mathbf{x}]$ the formula obtained from ϕ by the deletion of the quantification over x_1, \dots, x_r , that is, these variables become the free variables of $\phi[\mathbf{x}]$. For an r -structure (G, \mathbf{v}) with $\mathbf{v} = \langle v_1, \dots, v_r \rangle$ and $\phi[\mathbf{x}]$, we write $(G, \mathbf{v}) \models \phi[\mathbf{x}]$ to denote that $\phi[\mathbf{x}]$ evaluates to *true* on G if x_i is assigned v_i for $i \in [r]$. If $r = 0$, that is, \mathbf{v} and \mathbf{x} are empty, then $(G, \mathbf{v}) \models \phi[\mathbf{x}]$ is equivalent to $G \models \phi$.

Parameterized Complexity. We refer to the books [6, 7, 9, 17] for the detailed introduction to the field. Here we only briefly review the basic notions.

Parameterized Complexity is a bivariate framework for studying the computational complexity of computational problems. One variable is the input size n and the other is a *parameter* k associated with the input. The main goal is to confine the combinatorial explosion in the running time of an algorithm for an NP-hard problem to depend only on k . Thus, a parameterized problem is defined formally as a language $\mathcal{L} \subseteq \Sigma^* \times \mathbb{N}$, where Σ^* is a set of string over a finite alphabet Σ . A parameterized problem is said to be *fixed parameter tractable* (or FPT) if it can be solved in time $f(k) \cdot n^{\mathcal{O}(1)}$ for some computable function f . Also, we say that a parameterized problem belongs in the class XP if it can be solved in time $O(n^{f(k)})$ for some computable function f . The complexity class FPT consists of all fixed parameter tractable problems. Parameterized complexity theory also provides tools to disprove the existence of FPT algorithms under plausible complexity-theoretic assumptions. For this, Downey and Fellows introduced a hierarchy of parameterized complexity classes, namely $\text{FPT} \subseteq \text{W}[1] \subseteq \text{W}[2] \subseteq \text{W}[3] \subseteq \dots \subseteq \text{XP}$ and conjectured that it is proper. This conjecture plays a central role in obtaining lower complexity bounds. The basic way to show that it is unlikely that a parameterized problem admit an FPT algorithm is to show that it is W[1] or W[2]-hard using a *parameterized reduction* from a known W[1] or W[2]-hard problem.

A *kernelization* for a parameterized problem is a polynomial time algorithm that maps each instance (I, k) of a parameterized problem with the input I and parameter k to an instance (I', k') of the same problem such that

- (i) (I, k) is a **yes**-instance if and only if (I', k') is a **yes**-instance, and
- (ii) $|I'| + k'$ is bounded by $f(k)$ for some computable function f .

The output (I', k') is called a *kernel*. The function f is said to be the *size* of the kernel. A kernel is *polynomial* if f is polynomial. While it can be shown that every decidable parameterized problem is FPT if and only if it admits a kernel, it is unlikely that every problem in FPT has a polynomial kernel. In particular, the now standard *composition* and *cross-composition* techniques [2, 3] allow to show that certain problems have no polynomial kernels unless $\text{NP} \subseteq \text{coNP}/\text{poly}$.

To solve all considered problems, we have to solve the MODEL CHECKING problem for first-order logic on graphs:

MODEL CHECKING

Input: A graph G and an FOL-formula ϕ .

Question: Does $G \models \phi$?

MODEL CHECKING is known to be PSPACE-complete [19]. The problem is also hard from the parameterized complexity viewpoint when parameterized by the size of the formula. It was proved by Frick and Grohe in [10] that the problem is AW[*]-complete for this parametrization (see, e.g., the book [9] for the definition of the class). Thus, it is

unlikely that MODEL CHECKING is FPT when parameterized by the formula size. This immediately implies that the problem VERTEX-REMOVAL TO ϕ as well as the problems EDGE-REMOVAL/COMPLETION/EDITING TO ϕ are AW[*]-hard when parameterized by the size of ϕ even for $k = 0$. However, MODEL CHECKING is in XP when parameterized by the number of variables. In particular, if ϕ has r variables and input size is n , then it can be solved in time $\mathcal{O}(n^r)$ by exhaustive search. The currently best algorithm is given by Williams in [20] who proved the following.

Theorem 5 ([20]). *MODEL CHECKING can be solved in time $\tilde{\mathcal{O}}(n^\omega)$ for formulas with 3 variables and if the number of variables $r \geq 3$, then it can be solved in time $\tilde{\mathcal{O}}(n^{r-3+\omega})$ where ω is the matrix-multiplication exponent. Moreover, if $r \geq 9$, then MODEL CHECKING can be solved in time $n^{r-1+o(1)}$.*

Here $\tilde{\mathcal{O}}(f(n))$ is used to denote an upper bound $\mathcal{O}(f(n) \log^c n)$ for some positive constant c . These algorithms are, in fact, asymptotically optimal up to the *Strong Exponential Time Hypothesis* (SETH) (see [6, 14] for the definition). It was shown by Williams [20] that if MODEL CHECKING for formulas with $r \geq 4$ can be solved in time $\mathcal{O}(n^{r-1-\varepsilon})$ for some $\varepsilon > 0$, then SETH is false.

Because of these results, we assume throughout the paper that the FOL-formulas in the considered modification problems have a *constant* number of variables and, therefore, constant sizes. In particular, the exponents of polynomials in running times and the sizes of kernels should depend on the length $|\phi|$ of the formula ϕ .

We conclude this section by observing that Theorems 3 and 4 claim the same complexity status for EDGE-REMOVAL TO ϕ and EDGE-COMPLETION TO ϕ . This is not surprising, because these problems are equivalent in the following sense. Denote by \bar{G} the *complement* of a graph G , that is, the graph with the same vertex set such that every two distinct vertices are adjacent in \bar{G} if and only if they are nonadjacent in G . For an FOL-formula ϕ , denote by $\bar{\phi}$ the formula obtained from ϕ by replacing each adjacency predicate by the subformula expressing non-adjacency of distinct vertices, that is, $u \sim v$ is replaced by $\neg(u = v) \wedge \neg(u \sim v)$. Then we can make the following straightforward observation.

Observation 1. *For every FOL-formula ϕ , (G, k) is a yes-instance of EDGE-REMOVAL TO ϕ if and only if (\bar{G}, k) is a yes-instance of EDGE-COMPLETION TO $\bar{\phi}$.*

By Observation 1, it is sufficient to show Theorems 3 and 4 for EDGE-REMOVAL TO ϕ and EDGE-EDITING TO ϕ .

3 Upper bounds

In this section we prove the claims (i) of Theorems 1–4. First, we consider VERTEX-REMOVAL TO ϕ .

Lemma 1. *For every $\phi \in \Sigma_3$ without free variables, VERTEX-REMOVAL TO ϕ can be solved in time $|\phi|^k \cdot n^{\mathcal{O}(|\phi|)}$.*

Proof. Consider an instance (G, k) of VERTEX-REMOVAL TO ϕ for

$$\phi = \exists x_1 \cdots \exists x_r \forall y_1 \cdots \forall y_s \exists z_1 \cdots \exists z_t \chi,$$

where $r, s, t \geq 0$ and χ is quantifier-free. Let $\mathbf{x} = \langle x_1, \dots, x_r \rangle$, $\mathbf{y} = \langle y_1, \dots, y_s \rangle$, and $\mathbf{z} = \langle z_1, \dots, z_t \rangle$.

Assume that (G, k) is a **yes**-instance of VERTEX-REMOVAL TO ϕ . This means that there is $S \subseteq V(G)$ of size at most k such that $G - S \models \phi$. Observe that $G - S \models \phi$ if and only if there is an r -tuple $\mathbf{u} = \langle u_1, \dots, u_r \rangle$ of vertices of $G - S$ such that $(G - S, \mathbf{u}) \models \phi[\mathbf{x}]$.

We use this observation, and for each r -tuple \mathbf{u} of vertices of G , check whether there is $S \subseteq V(G)$ of size at most k that has no common vertices with \mathbf{u} and it holds that $(G - S, \mathbf{u}) \models \phi[\mathbf{x}]$. If we find such a set S , we return this solution for the considered instance of VERTEX-REMOVAL TO ϕ . Otherwise, if we fail to find S for all r -tuples \mathbf{u} , we conclude that (G, k) is a **no**-instance. From now we assume that \mathbf{u} is given.

Suppose that $(G, \mathbf{u}) \models \phi[\mathbf{x}]$ does not hold. Then there is an s -tuple $\mathbf{v} = \langle v_1, \dots, v_s \rangle$ of vertices of G such that $(G, \mathbf{uv}) \models \phi[\mathbf{xy}]$ does not hold. Our algorithm is based on the following crucial claim.

Claim 1.1. *For every $S \subseteq V(G)$ such that S is disjoint with \mathbf{u} and $(G - S, \mathbf{u}) \models \phi[\mathbf{x}]$, S contains at least one vertex of \mathbf{v} .*

The proof is by contradiction. Assume that $(G - S, \mathbf{u}) \models \phi[\mathbf{x}]$ but S and \mathbf{v} are disjoint. Then $(G - S, \mathbf{uv}) \models \phi[\mathbf{xy}]$. By definition, this means that there is a t -tuple of vertices \mathbf{w} of $G - S$ such that $(G - S, \mathbf{uvw}) \models \phi[\mathbf{xyz}]$. In other words, χ evaluates to *true* if the x , y and z -variables are assigned to \mathbf{u} , \mathbf{v} and \mathbf{w} respectively. This immediately implies that $(G, \mathbf{uvw}) \models \phi[\mathbf{xyz}]$ and, therefore, $(G, \mathbf{uv}) \models \phi[\mathbf{xy}]$. This contradicts the assumption that $(G, \mathbf{uv}) \not\models \phi[\mathbf{xy}]$.

Claim 1.1 leads to the following recursive algorithm that find a solution S for the given \mathbf{u} (if such a solution exist). The algorithm receives as the input the current set S that is initially set to be empty and finds a solution $S^* \supseteq S$ as follows.

1. If $(G - S, \mathbf{uv}) \models \phi[\mathbf{xy}]$ for all s -tuples $\mathbf{v} = \langle v_1, \dots, v_s \rangle$ of vertices of $G - S$, then return S and stop.
2. Otherwise, for an s -tuple $\mathbf{v} = \langle v_1, \dots, v_s \rangle$ of vertices of $G - S$ such that $(G - S, \mathbf{uv}) \not\models \phi[\mathbf{xy}]$, do the following:
 - (i) if $|S| = k$ or all the vertices of \mathbf{v} are in \mathbf{u} , then stop;
 - (ii) else, for each $v_i \in \mathbf{v}$ that is not in \mathbf{u} , call the algorithm for $S' = S \cup \{v_i\}$.

The correctness of the algorithm follows from Claim 1.1. Concerning the running time of the algorithm. At each iteration we check at most n^s s -tuples \mathbf{v} and for each \mathbf{v} we verify in time $n^{\mathcal{O}(|\phi|)}$ whether $(G - S, \mathbf{uv}) \models \phi[\mathbf{xy}]$. Hence, each iteration takes time $n^{\mathcal{O}(|\phi|)}$. Also at each iteration we branch into at most s subproblems and the depth of the search tree produced by the algorithm is at most k . Thus the running time is $s^k \cdot n^{\mathcal{O}(|\phi|)}$. Recall that we call the algorithm for each r -tuple \mathbf{u} . Since there are n^r such tuples, we have that the total running time is $s^k \cdot n^{\mathcal{O}(|\phi|)}$, which can be rewritten as $|\phi|^k \cdot n^{\mathcal{O}(|\phi|)}$. \square

Lemma 1 immediately implies Theorem 1 (i).

We move to EDGE-REMOVAL TO ϕ and EDGE-EDITING TO ϕ .

Lemma 2. *For every $\phi \in \Sigma_2$ without free variables, EDGE-REMOVAL TO ϕ and EDGE-EDITING TO ϕ can be solved in time $|\phi|^{2k} \cdot n^{\mathcal{O}(|\phi|)}$.*

Proof. The proof is similar to the proof of Lemma 1. We show the claim for EDGE-REMOVAL TO ϕ and then explain how it should be modified for EDGE-EDITING TO ϕ .

Let (G, k) be an instance of EDGE-REMOVAL TO ϕ for

$$\phi = \exists x_1 \dots \exists x_r \forall y_1 \dots \forall y_s \chi,$$

where χ is quantifier-free. Let $\mathbf{x} = \langle x_1, \dots, x_r \rangle$ and $\mathbf{y} = \langle y_1, \dots, y_s \rangle$.

We observe that $F \subseteq E(G)$ of size at most k is a solution for an instance (G, k) of EDGE-REMOVAL TO ϕ if and only if there is an r -tuple $\mathbf{u} = \langle u_1, \dots, u_r \rangle$ of vertices of G such that $(G - F, \mathbf{u}) \models \phi[\mathbf{x}]$. Respectively, for each r -tuple \mathbf{u} of vertices of G , we check whether there is $F \subseteq E(G)$ of size at most k such that $(G - F, \mathbf{u}) \models \phi[\mathbf{x}]$. If we find such F , we return this solution and we obtain that (G, k) is a **no**-instance otherwise. Assume that \mathbf{u} is given.

If the property $(G, \mathbf{u}) \models \phi[\mathbf{x}]$ is not fulfilled, then there is an s -tuple $\mathbf{v} = \langle v_1, \dots, v_s \rangle$ of vertices of G such that it does not hold that $(G, \mathbf{uv}) \models \phi[\mathbf{xy}]$. We use the following claim.

Claim 2.1. *For every $F \subseteq E(G)$ such that $(G - F, \mathbf{u}) \models \phi[\mathbf{x}]$, F contains at least one edge with both end-vertices in \mathbf{uv} .*

To obtain a contradiction, assume that $(G - F, \mathbf{u}) \models \phi[\mathbf{x}]$ but every edge of F has at least one end-vertex outside the tuples \mathbf{u} and \mathbf{v} . This means that χ evaluates to *true* on $G - F$ if the x any y -variables are assigned to \mathbf{u} and \mathbf{v} respectively. Notice that every two vertices of \mathbf{uv} are adjacent in $G - F$ if and only if they are adjacent in G . Hence, χ evaluates to *true* on G if the x any y -variables are assigned to \mathbf{u} and \mathbf{v} respectively. This means that $(G, \mathbf{uv}) \models \phi[\mathbf{xy}]$; a contradiction.

We construct the following recursive branching algorithm that finds a solution F for the given \mathbf{u} if it exists. The algorithm takes as the input the current set F that is initially empty and finds a solution $F^* \supseteq F$:

1. If $(G - F, \mathbf{uv}) \models \phi[\mathbf{xy}]$ for all s -tuples $\mathbf{v} = \langle v_1, \dots, v_s \rangle$ of vertices of G , then return F and stop.
2. Otherwise, for an s -tuple $\mathbf{v} = \langle v_1, \dots, v_s \rangle$ of vertices of G such that $(G - F, \mathbf{uv}) \not\models \phi[\mathbf{xy}]$, do the following:
 - (i) set $L \subseteq E(G)$ be the set of edges with both end-vertices in \mathbf{uv} ,
 - (ii) if $|F| = k$ or $L = \emptyset$, then stop;
 - (iii) else, and for each $e \in L$, call the algorithm for $F' = F \cup \{e\}$.

The correctness of the algorithm follows from Claim 2.1. On each iteration we check at most n^s s -tuples \mathbf{v} , and for each \mathbf{v} , verify in time $n^{\mathcal{O}(|\phi|)}$ whether $(G - F, \mathbf{uv}) \models \phi[\mathbf{xy}]$. Hence, each iteration can be done in time $n^{\mathcal{O}(|\phi|)}$. Also on each iteration we have at most $|L| \leq \binom{r+s}{2}$ branches and the depth of the search tree produced by the algorithm is at most k . This implies that the running time is $(r+s)^{2k} \cdot n^{\mathcal{O}(|\phi|)}$. Recall that we call the algorithm for each r -tuple u . Since there are n^r such tuples, we have that the total running time is $|\phi|^{2k} \cdot n^{\mathcal{O}(|\phi|)}$.

For EDGE-EDITING TO ϕ , the algorithm is essentially the same. The difference is that in addition to edge removal we allowed to add edges. Respectively, we replace $G - F$ by $G \triangle F$ in the above algorithm and modify Steps 2 (i)—(iii):

- (i) set L be the set of pairs of distinct vertices of \mathbf{uv} ,
- (ii) if $|F| = k$ or $L = \emptyset$, then stop;
- (iii) else, and for each $e \in L$, call the algorithm for $F' = F \cup \{e\}$.

Notice that the variant of Claim 2.1, where $G - F$ is replaced by $G \triangle F$, holds and this implies correctness. The time analysis is the same. \square

Lemma 2 together with Observation 1 implies Theorem 3(i). Our next aim is show kernelization upper bounds. First, we observe that for Σ_1 -formulas, our problems can be solved in polynomial time.

Lemma 3. *For every $\phi \in \Sigma_1$ without free variables, VERTEX-REMOVAL TO ϕ , EDGE-REMOVAL TO ϕ , and EDGE-EDITING TO ϕ can be solved in time $n^{\mathcal{O}(|\phi|)}$.*

Proof. Assume that

$$\phi = \exists x_1 \cdots \exists x_r \chi$$

where χ is quantifier-free.

For VERTEX-REMOVAL TO ϕ , it is sufficient to observe that (G, k) is a **yes**-instance of the problem if and only if $G \models \phi$. We can use Theorem 5 and solve the problem in time $n^{\mathcal{O}(|\phi|)}$.

For EDGE-REMOVAL TO ϕ and EDGE-EDITING TO ϕ , we can observe that (G, k) is a **yes**-instance if and only if there is a set of vertices U of size $s = \min\{r, n\}$ such that $(G[U], k)$ is a **yes**-instance. We can check all such sets U in time $n^{\mathcal{O}(|\phi|)}$, and for each set, we use brute force to verify whether $(G[U], k)$ is a **yes**-instance. Since the brute force checking of all subsets of edges or pairs of vertices of $G[U]$ of size at most $k' = \min\{k, \binom{s}{2}\}$ can be done in time $s^{2k'}$, the total running time is $s^{2s^2} \cdot n^{\mathcal{O}(|\phi|)}$. Because $s \leq |\phi|$, we can write it as $n^{\mathcal{O}(|\phi|)}$. \square

Because every problem that can be solved in polynomial time has a trivial polynomial kernel, Lemma 3 together with Observation 1 implies Theorem 4(i). Clearly, the lemma also implies the claim of Theorem 2 (i) for Σ_1 -formulas. It remains to prove it for Π_1 -formulas. For this, we need the classic result of Lewis and Yannakakis [16]. A graph property P is said to be *hereditary* if for each graph G satisfying P , it hold that P holds for every induced subgraph of G . A property P is *nontrivial* if it is true for infinitely many graphs and it is false for infinitely many graphs. VERTEX-REMOVAL TO P asks, given a graph G and a positive integer k , whether it is possible to remove at most k vertices of G to obtain a graph satisfying P . It was proved by Lewis and Yannakakis [16] that the following dichotomy holds for a hereditary property P that can be tested in polynomial time: VERTEX-REMOVAL TO P can be solved in polynomial time if P is trivial, and the problem is NP-complete otherwise.

Lemma 4. *For every $\phi \in \Pi_1$ without free variables, VERTEX-REMOVAL TO ϕ admits a polynomial kernel.*

Proof. Let (G, k) be an instance of VERTEX-REMOVAL TO ϕ for

$$\phi = \forall x_1 \cdots \forall x_r \chi$$

where χ is quantifier-free. Let $\mathbf{x} = \langle x_1, \dots, x_r \rangle$. Observe that the graph property $G \models \phi$ is hereditary for Π_1 -formulas. If this property is trivial, we can solve VERTEX-REMOVAL TO ϕ in polynomial time [16] and conclude that the problem admits a trivial polynomial kernel. Assume from now that the property $G \models \phi$ is not trivial. By the result of Lewis and Yannakakis [16], VERTEX-REMOVAL TO ϕ is NP-complete.

For every tuple \mathbf{v} of vertices of G , denote by $U_{\mathbf{v}}$ the set of vertices contained in \mathbf{v} . Let

$$\mathcal{U} = \{U_{\mathbf{v}} \mid \mathbf{v} \text{ is an } r\text{-tuple of vertices of } G \text{ such that } (G, \mathbf{v}) \not\models \phi[\mathbf{x}]\}.$$

The crucial observation is that $S \subseteq V(G)$ of size at most k is a solution for (G, k) if and only if S is a *hitting set* for \mathcal{U} , that is, $S \cap U_{\mathbf{v}} \neq \emptyset$ for every $U_{\mathbf{v}} \in \mathcal{U}$. This observation is proved by the same arguments as Claim 1.1 assuming that \mathbf{u} is empty.

The s -HITTING SET problem that asks, given a family of sets \mathcal{U} of size at most s over some universe and a non-negative integer k , whether there is a hitting set S for \mathcal{U} is known to have a polynomial kernel of size at most $(2s - 1)k^{s-1} + k$ by the result of Abu-Khazam [1]. Apparently HITTING SET is in NP. Hence, there is a polynomial reduction from HITTING SET to the NP-complete problem VERTEX-REMOVAL TO ϕ . This implies that VERTEX-REMOVAL TO ϕ admits a polynomial kernel. \square

4 Lower bounds

Here we prove the hardness claims of Theorems 1–4. In Subsection 4.1, we give the technical result about reducing EDGE-REMOVAL TO ϕ to VERTEX-REMOVAL TO ψ . In Subsection 4.2, we show W[2]-hardness and in Subsection 4.3 we obtain kernelization lower bounds.

4.1 Reducing Edge Removal to Vertex Removal

In this section we construct a generic reduction of EDGE-REMOVAL TO ϕ to VERTEX-REMOVAL TO ψ that we use twice in the proofs of our complexity lower bounds.

We say that an FOL-formula ϕ is \forall -containing if the prefix of ϕ contains a \forall quantifier.

Lemma 5. *For every \forall -containing FOL-formula $\phi \in \Sigma_s \cup \Pi_s$ without free variables for $s \geq 1$, there is formula $\psi \in \Sigma_{s+1} \cup \Pi_{s+1}$ without free variables such that*

- (i) $|\psi| = \text{poly}(|\phi|)$,
- (ii) if $\phi \in \Sigma_s$ (resp. $\phi \in \Pi_s$), then $\psi \in \Sigma_{s+1}$ (resp. $\psi \in \Pi_{s+1}$),
- (iii) there is a polynomial reduction of EDGE-REMOVAL TO ϕ to VERTEX-REMOVAL TO ψ that transforms each instance (G, k) of EDGE-REMOVAL TO ϕ to an equivalent instance (G', k) of VERTEX-REMOVAL TO ψ , i.e., the parameter k remains the same.

Proof. Let (G, k) be an instance of EDGE-REMOVAL TO ϕ . We construct the instance (G', k) of VERTEX-REMOVAL TO ψ from (G, k) and then we construct ψ . The main idea is to replace edge removals by vertex removals switching to the *incidence* graph of G or, equivalently, by subdividing edges of G . Then we have to “label” the vertices of the original graph that should not be removed. We do it by making them adjacent to sufficiently many pendant vertices. Formally, we construct G' as follows.

- Construct a copy of G and subdivide each edge, that is, for each $e = xy \in E(G)$, delete e , construct a new vertex v_e and make it adjacent to x and y . We say that the vertices of G are *branching* vertices and the vertices obtained by the edge subdivisions are called *subdivision* vertices.
- For each branching vertex u , introduce $k + 3$ new vertices v_1, \dots, v_{k+3} and make them adjacent to u ; we call these vertices *pendant*.

Notice that every subdivision vertex has degree 2 and every branching vertex has degree at least 3. Moreover, if H is obtained from G' by the removal of at most k vertices, then still every remaining branching vertex has degree at least 3. Observe also that H has isolated vertices if and only if at least one branching vertex of G is removed in the construction of H .

Our next aim is to construct ψ from ϕ to ensure that (G, k) is a **yes**-instance of **EDGE-REMOVAL TO ϕ** if and only if (G', k) is a **yes**-instance of **VERTEX-REMOVAL TO ψ** . Let

$$\phi = \mathbf{Q}_1 x_1 \dots \mathbf{Q}_p x_p \chi$$

where $\mathbf{Q}_1, \dots, \mathbf{Q}_p$ are quantifiers, x_1, \dots, x_p are variables and χ is quantifier-free. We also assume that χ is written in the conjunctive normal form.

The construction of ψ is done in several steps. First, we take care of adjacencies in ϕ . Recall that two vertices u and v are adjacent in G if and only if they have a common neighbor in G' . Respectively, we modify the adjacency predicates in χ .

Let $\Pi = \{\pi_1, \dots, \pi_s\}$ be the family (multiset) of all predicates of the form $x_i \sim x_j$ that occur in χ without negations. If the same predicate $x_i \sim x_j$ occurs several times, then for each occurrence, we include it in Π . Similarly, let $\bar{\Pi} = \{\bar{\pi}_1, \dots, \bar{\pi}_t\}$ be the family (multiset) of all predicates of the form $\neg(x_i \sim x_j)$ that occur in χ . Then we do the following.

- Construct s new variables y_1, \dots, y_s and t variables z_1, \dots, z_t .
- For each $h \in \{1, \dots, s\}$, consider $\pi_h = x_i \sim x_j$ for some $i, j \in \{1, \dots, p\}$ and replace it by $\neg(x_i = x_j) \wedge (x_i \sim y_h) \wedge (y_h \sim x_j)$.
- For each $h \in \{1, \dots, t\}$, consider $\bar{\pi}_h = \neg(x_i \sim x_j)$ for some $i, j \in \{1, \dots, p\}$ and replace it by $(x_i = x_j) \vee \neg(x_i \sim y_h) \vee \neg(y_h \sim x_j)$.

- Denote the formula obtained from χ by χ' . Then

- if $\mathbf{Q}_p = \exists$, set $\sigma = \exists y_1 \dots \exists y_s \forall z_1 \dots \forall z_t \chi'$, and
- if $\mathbf{Q}_p = \forall$, set $\sigma = \forall z_1 \dots \forall z_t \exists y_1 \dots \exists y_s \chi'$.

Consider the formula $\alpha = \mathbf{Q}_1 x_1 \dots \mathbf{Q}_p x_p \sigma$. Observe that we added new quantified variables in the end of the prefix of ϕ in such a way that we obtain at most one additional alternation of quantifiers. That is, we obtain that $\alpha \in \Sigma_{s+1}$ if $\phi \in \Sigma_s$ and $\alpha \in \Pi_{s+1}$ if $\phi \in \Pi_s$. The crucial property of the above construction is given in the following straightforward claim.

Claim 5.1. $G \models \phi$ if and only if $G' \models \beta$ where

$$\beta = \mathbf{Q}_1 x_1 \dots \mathbf{Q}_p x_p \sigma \text{ with the domains of the variables } x_1, \dots, x_p \text{ restricted to } V(G).$$

Moreover, for every set of pendant vertices X of size at most k , $G \models \phi$ if and only if $(G' - X) \models \beta$.

Notice that in the formula of Claim 5.1 we insist to restrict the domains of the variables x_1, \dots, x_p in β to $V(G)$, that is, β is not an FOL-formula (and $\beta \neq \alpha$). Our next aim is to express these additional constraints in the first-order logic. We do it using the property that the branching vertices of G' have degrees at least 3 and all the other vertices have degrees at most 2.

We consecutively construct the formulas $\rho_{n+1}, \dots, \rho_1$. First, we set $\rho_{n+1} = \sigma$. Note that x_1, \dots, x_p are free variable for ρ_{n+1} . Assume inductively that $1 \leq i \leq p$ and ρ_{i+1} with free variables x_1, \dots, x_i is already constructed. Denote by P_{i+1} the prefix and μ_{i+1} the quantifier-free part of ρ_{i+1} respectively, that is, $\rho_{i+1} = P_{i+1} \mu_{i+1}$. The construction of ρ_i depends on the quantifier \mathbf{Q}_i .

- Introduce 3 new variables r_i^1, r_i^2, r_i^3 .

- If $Q_i = \exists$, then set

$$\rho_i = \exists x_i \exists r_i^1 \exists r_i^2 \exists r_i^3 P_{i+1} [(\neg(r_i^1 = r_i^2) \wedge \neg(r_i^1 = r_i^3) \wedge \neg(r_i^2 = r_i^3)) \wedge ((x_i \sim r_i^1) \wedge (x_i \sim r_i^2) \wedge (x_i \sim r_i^3)) \wedge \mu_{i+1}].$$

- If $Q_i = \forall$, then set

$$\rho_i = \forall x_i \forall r_i^1 \forall r_i^2 \forall r_i^3 P_{i+1} [((\neg(r_i^1 = r_i^2) \wedge \neg(r_i^1 = r_i^3) \wedge \neg(r_i^2 = r_i^3)) \wedge (((x_i \sim r_i^1) \wedge (x_i \sim r_i^2) \wedge (x_i \sim r_i^3)) \rightarrow \mu_{i+1}))].$$

Let $\gamma = \rho_1$. Note that in our construction of γ , we do not create new alternations of quantifications, that is, $\gamma \in \Sigma_{s+1}$ or in Π_{s+1} depending on whether $\alpha \in \Sigma_{s+1}$ or in Π_{s+1} . The construction of γ implies the next claim.

Claim 5.2. $G \models \phi$ if and only if $G' \models \gamma$. Moreover, for every set of pendant vertices X of size at most k , $G \models \phi$ if and only if $(G' - X) \models \gamma$.

Recall that the removal of the edges in G corresponds to the removal of subdivision vertices of G' . Respectively, our next aim is to ensure that the removal of a branching vertex of G' leads to the graph for which our formula is false. We use the property that for every set S of at most k vertices of G' , $G' - S$ has an isolated vertex if and only if S contains a branching vertex. We use the property that the condition that a graph has no an isolated can be expressed by the formula $\forall s_1 \exists s_2 (s_1 \sim s_2)$ and modify γ as follows. Let P be the prefix of γ and let μ be the quantifier-free part. We write P as the concatenation of 3 parts P_1 , P_2 and P_3 where P_1 and/or P_3 may be empty. Recall that the prefix of the original formula ϕ contains the $\forall x_i$ for some $i \in \{1, \dots, p\}$ by the condition of the lemma. Hence, the same holds for γ . Let P_1 be the first part of P until the first occurrence of the quantifier \forall . Then P_2 is the next part until the first occurrence of the quantifier \exists or until the end of P if such a quantifier does not exist. Respectively, P_3 is the remaining part. We define

$$\psi = P_1 \forall s_1 P_2 \exists s_2 P_3 [(s_1 \sim s_2) \wedge \mu]$$

using two new variables s_1 and s_2 .

Notice that the insertion of the new quantifications is done in such a way that we do not introduce new alternations of quantifiers unless P_3 is empty. But if P_3 is empty, then $\phi \in \Pi_1$ or $\phi \in \Sigma_2$ and the new alternations were not introduced in the construction of α from ϕ . We have that either $\phi, \gamma \in \Sigma_s$ and $\psi \in \Sigma_{s+1}$ or $\phi, \gamma \in \Pi_s$ and $\psi \in \Pi_{s+1}$.

We show the following claim.

Claim 5.3. *The instance (G, k) is a yes-instance of EDGE-REMOVAL TO ϕ if and only if (G', k) is a yes-instance of VERTEX-REMOVAL TO ψ .*

To show the claim, assume first that (G, k) is a yes-instance of EDGE-REMOVAL TO ϕ . Then there is $F \subseteq E(G)$ of size at most k such that $(G - F) \models \phi$. We define $S \subseteq V(G')$ be the the set of the subdivision vertices of G' corresponding to the edges of F , that is, $S = \{v_e \mid e \in F\}$. Clearly, $|S| \leq k$. Notice that our reduction algorithm for graphs produces $G' - S$ from $G - F$. Hence, by Claim 5.2, $(G' - F) \models \gamma$. Because $G' - S$ has no isolated vertices, we have that $(G' - S) \models \psi$. It means that (G', k) is a yes-instance of VERTEX-REMOVAL TO ψ .

Suppose now that (G', k) is a yes-instance of VERTEX-REMOVAL TO ψ . Then there is $S \subseteq V(G')$ of size at most k such that $(G' - S) \models \psi$. Since $\psi = P_1 \forall s_1 P_2 \exists s_2 P_3 [(s_1 \sim s_2) \wedge \mu]$, we have that $G' - S$ has no isolated vertices, that is, S contains only subdivision vertices

of G' or pendant vertices. Also this immediately implies that $(G' - S) \models \gamma$. Let S' be the set of subdivision vertices of S and let X be the set of pendant vertices in S . We define F to be the set of edges of G corresponding to the subdivision vertices in S' , that is, $F = \{e \in E(G) \mid v_e \in S'\}$. We have that $|F| \leq k$. Let also X be the set of pendant vertices in S . Observe again that our reduction algorithm for graphs produces $G' - S'$ from $G - F$. Then by Claim 5.2, we have that $(G - F) \models \phi$. We conclude that (G, k) is a yes-instance of EDGE-REMOVAL TO ϕ .

To complete the proof of the lemma, observe that the size of ψ is polynomial in the size of ϕ by our construction of the formula and this shows (i). To show (ii), observe that $\psi \in \Sigma_{s+1}$ if $\phi \in \Sigma_s$ and $\psi \in \Pi_{s+1}$ if $\phi \in \Pi_s$. The last claim (iii) of the lemma immediately follows from Claim 5.3. \square

4.2 W[2]-hardness

In this subsection we show that there are formulas ϕ in Π_2 and Π_3 for which EDGE-REMOVAL (EDITING) TO ϕ and VERTEX-REMOVAL TO ψ , respectively are W[2]-hard when parameterized by k . First, we show the claim for EDGE-REMOVAL TO ϕ and EDGE-EDITING TO ϕ .

Lemma 6. *There is an FOL-formula $\phi \in \Pi_2$ without free variables with 5 variables such that EDGE-REMOVAL TO ϕ and EDGE-EDITING TO ϕ are W[2]-hard.*

Proof. We define the formula ϕ as follows:

$$\phi = \forall x \exists y_1 \exists y_2 \exists y_3 \exists y_4 [(x \sim y_1) \wedge (x \sim y_2) \wedge (x \sim y_3) \wedge (x \sim y_4) \wedge (y_1 \sim y_2) \wedge (y_1 \sim y_3) \wedge (y_2 \sim y_3) \wedge (y_2 \sim y_4) \wedge (y_3 \sim y_4) \wedge \neg(y_1 = y_4) \wedge \neg(y_1 \sim y_4)].$$

In terms of graphs, $G \models \phi$ means that for every vertex x of G , there are vertices y_1, y_2, y_3, y_4 such that these vertices together with x induce the graph W shown in Fig. 1. We say that W is a ϕ -witness subgraph rooted in x .

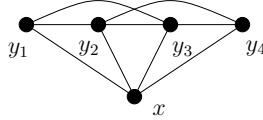


Figure 1: The ϕ -witness graph W .

We show hardness for EDGE-EDITING TO ϕ by reducing the SET COVER problem:

SET COVER

Input: A family of sets \mathcal{S} over the universe U and a positive integer k .

Question: Is there a subfamily $\mathcal{S}^* \subseteq \mathcal{S}$ of size at most k that covers U , that is, every element of U is in one of the set of \mathcal{S}^* ?

Parameter: k

It is well-known that SET COVER is W[2]-hard when parameterized by k [7].

Let (U, \mathcal{S}, k) be an instance SET COVER, $\mathcal{S} = \{S_1, \dots, S_m\}$ and $U = \{u_1, \dots, u_n\}$. We construct the graph G as follows.

- For each $i \in \{1, \dots, m\}$, construct the graph H_i with 4 root vertices $s_i^1, s_i^2, s_i^3, s_i^4$ as it is shown in Fig. 2 a).

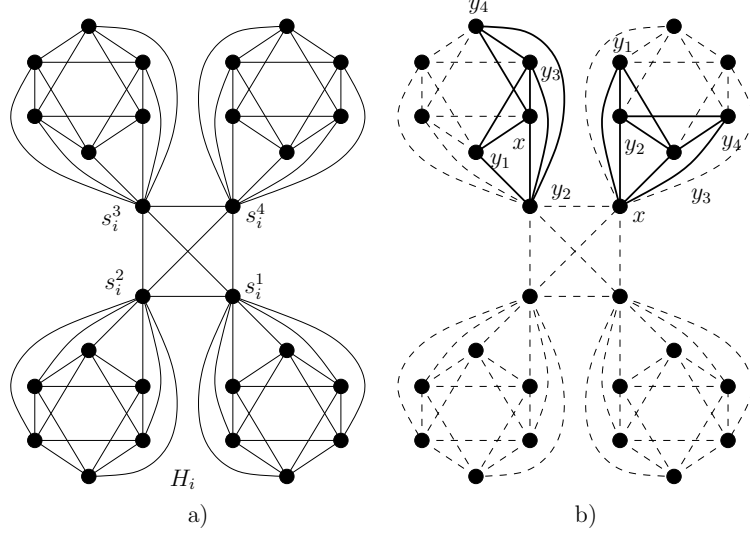


Figure 2: Construction of H_i and the witness subgraphs for the vertices of H_i . The witness subgraphs are shown by thick lines and the other edges are shown by dashed lines.

- For each $j \in \{1, \dots, n\}$, construct $2k + 1$ vertices u_j^0, \dots, u_j^{2k} and make them adjacent to the root vertices of all the gadgets H_r such that the element u_j of the universe U is in the set $S_r \in \mathcal{S}$.

We claim that (U, \mathcal{S}, k) is a **yes**-instance SET COVER if and only if (G, k) is a **yes**-instance of EDGE-EDITING TO ϕ .

Suppose that (U, \mathcal{S}, k) is a **yes**-instance SET COVER. Let $\mathcal{S}^* \subseteq \mathcal{S}$ be a family of size at most k that covers U . We construct the set of edges F of G as follows. For every $S_i \in \mathcal{S}^*$, we include the edge $s_i^1 s_i^4$ of the gadget H_i in F . Clearly, $|F| \leq k$. Let $G' = G \triangle F = G - F$. We show that $G' \models \phi$. Recall that we have to show that for every $x \in V(G')$, there are y_1, y_2, y_3, y_4 , such that $G[\{x, y_1, y_2, y_3, y_4\}]$ is a ϕ -witness subgraph rooted in x . For $x \in \bigcup_{i=1}^m V(H_i)$, such subgraphs are shown in Fig. 2 b). Let $x = u_j^h$ for $j \in \{1, \dots, n\}$ and $h \in \{0, \dots, 2k\}$. The element $u_j \in U$ is covered by some set $S_i \in \mathcal{S}^*$. Since $s_i^1 s_i^4 \in F$, we have that $G[u_j^h, s_i^1, s_i^2, s_i^3, s_i^4]$ is a ϕ -witness subgraph rooted in x . We conclude that $G' \models \phi$ and, therefore, (G, k) is a **yes**-instance of EDGE-EDITING TO ϕ .

Assume that (G, k) is a **yes**-instance of EDGE-EDITING TO ϕ . Then there is $F \subseteq (V(G))$ with $|F| \leq k$ such that for $G' = G \triangle F$, it holds that $G' \models \phi$. Consider the auxiliary graph $Q = (V(G), F)$. For $i \in \{1, \dots, m\}$, let $\delta_i = \sum_{v \in V(H_i)} d_Q(v)$. We define $\mathcal{S}^* = \{S_i \mid 1 \leq i \leq m, \delta_i \geq 2\}$. Because $|F| \leq k$, $\sum_{i=1}^m \delta_i \leq 2k$ and, therefore, $|\mathcal{S}^*| \leq k$. We claim that \mathcal{S}^* covers U . To obtain a contradiction, assume that there is $j \in \{1, \dots, n\}$ such that u_j is not covered by \mathcal{S}^* . Since $|F| \leq k$, there is $h \in \{0, \dots, 2k\}$ such that the vertex u_j^h is not incident to the pairs of F . Because $G' \models \phi$, there is a ϕ -witness subgraph rooted in $x = u_j^h$. Hence, there are $y_1, y_2, y_3, y_4 \in V(G)$ such that $G[\{x, y_1, y_2, y_3, y_4\}]$ is a ϕ -witness subgraph. Notice that $y_1, y_2, y_3, y_4 \in \bigcup_{i=1}^m V(H_i)$. Observe that if $y_s \in V(H_i)$, then $F \cap (V_2^{(H_i)}) = \emptyset$, because $\delta_i \leq 1$. Hence, it cannot happen that $y_1, y_2, y_3, y_4 \in V(H_i)$ for some $i \in \{1, \dots, m\}$, because it would mean that $\{y_1, y_2, y_3, y_4\} = \{s_i^1, s_i^2, s_i^3, s_i^4\}$ but $G'[\{s_i^1, s_i^2, s_i^3, s_i^4\}] = K_4$, a contradiction. Therefore, there are distinct $i, i' \in \{1, \dots, n\}$ such that $V(H_i) \cap \{y_1, y_2, y_3, y_4\} \neq \emptyset$ and $V(H_{i'}) \cap \{y_1, y_2, y_3, y_4\} \neq \emptyset$. Since $\delta_i, \delta_{i'} \leq 1$, there is a unique pair ab of F such that $a \in V(H_i)$ and $b \in V(H_{i'})$. Moreover, ab is a bridge of $G'[\bigcup_{s=1}^m V(H_s)]$ and, therefore, ab is a bridge of $G'[\{y_1, y_2, y_3, y_4\}]$. This contradicts the fact that $W - x$ is 2-connected. We conclude that \mathcal{S}^* covers U . Hence, (U, \mathcal{S}, k) is a

yes-instance SET COVER.

This concludes the $W[2]$ -hardness proof for EDGE-EDITING TO ϕ . To show that EDGE-REMOVAL TO ϕ is $W[2]$ -hard when parameterized by k , we use the same reduction. Note that to show that if (U, \mathcal{S}, k) is a yes-instance of SET COVER then (G, k) is a yes-instance of EDGE-EDITING TO ϕ , we constructed $F \subseteq E(G)$, that is, we proved that (G, k) is a yes-instance of EDGE-REMOVAL TO ϕ . \square

Lemma 6 and Observation 1 imply Theorem 3 (ii). To show the claim for VERTEX-REMOVAL TO ϕ we combine Lemma 6 with Lemma 5 and obtain the following lemma that implies Theorem 1 (ii).

Lemma 7. *There is a constant c such that there is an FOL-formula $\phi \in \Pi_3$ without free variables that has at most c variables such that VERTEX-REMOVAL TO ϕ is $W[2]$ -hard.*

4.3 Kernelization lower bounds

In this subsection we obtain the kernelization lower bounds for EDGE-REMOVAL (EDITING) TO ϕ and VERTEX-REMOVAL TO ψ .

First, we show the lower bounds for EDGE-REMOVAL TO ϕ and EDGE-EDITING TO ϕ for Π_1 -formulas. To do it, we use the known results about kernelization lower bounds for the H -FREE EDGE REMOVAL and H -FREE EDITING. Recall that for a graph H , H -FREE EDGE REMOVAL (H -FREE EDITING) asks, given a graph G and a nonnegative integer k , whether there is a set of edges F (a set $F \subseteq \binom{V(G)}{2}$) respectively) of size at most k such that $G - F$ ($G \triangle F$ respectively) does not contain an induced subgraph isomorphic to H . Since the property that a graph G has no induced subgraph isomorphic to H can be expressed by an FOL-formula $\phi_H \in \Pi_1$ that has $|V(H)|$ variables, H -FREE EDGE REMOVAL and H -FREE EDITING can be written as EDGE-REMOVAL TO ϕ_H and EDGE-EDITING TO ϕ_H respectively. The first kernelization lower bounds for H -FREE EDGE REMOVAL and H -FREE EDITING were obtained by Kratsch and Wahlström in [15] who proved that there are graphs H for which these problems do not admit polynomial kernels unless $\text{NP} \subseteq \text{coNP/poly}$. Some further results were obtained by Guillemot et al. [13]. In [5] Cai and Cai completely characterized the cases when the problems have no polynomial kernels if H is a path or cycle or is 3-connected graph up to the conjecture that $\text{NP} \not\subseteq \text{coNP/poly}$. In particular, they proved that H -FREE EDGE REMOVAL and H -FREE EDITING do not have polynomial kernels if $H = C_4$ unless $\text{NP} \subseteq \text{coNP/poly}$. This immediately yields the following lemma.

Lemma 8. *There is an FOL-formula $\phi \in \Sigma_1$ without free variables that has 5 variables such that EDGE-REMOVAL TO ϕ and EDGE-EDITING TO ϕ have no polynomial kernels unless $\text{NP} \subseteq \text{coNP/poly}$.*

Lemma 8 and Observation 1 prove Theorem 4 (ii). Using Lemma 5, we obtain the following lemma for VERTEX-REMOVAL TO ϕ .

Lemma 9. *There is a constant c such that there is an FOL-formula $\phi \in \Pi_2$ without free variables that has at most c variables such that VERTEX-REMOVAL TO ϕ has no polynomial kernel unless $\text{NP} \subseteq \text{coNP/poly}$.*

Our final task is to show that it is unlikely that VERTEX-REMOVAL TO ϕ has a polynomial kernel for Σ_2 -formulas. We do it by using the *cross-composition* technique introduced by Bodlaender, Jansen and Kratsch [3] (see also [6] for the introduction to the technique). Here we only briefly sketch the main notions that we need to apply it.

Let Σ be a finite alphabet. An equivalence relation \mathcal{R} on the set of strings Σ^* is called a *polynomial equivalence relation* if the following two conditions hold:

- i) there is an algorithm that given two strings $x, y \in \Sigma^*$ decides whether x and y belong to the same equivalence class in time polynomial in $|x| + |y|$,
- ii) for any finite set $S \subseteq \Sigma^*$, the equivalence relation \mathcal{R} partitions the elements of S into a number of classes that is polynomially bounded in the size of the largest element of S .

Let $\mathcal{L} \subseteq \Sigma^*$ be a problem, let \mathcal{R} be a polynomial equivalence relation on Σ^* , and let $\mathcal{P} \subseteq \Sigma^* \times \mathbb{N}$ be a parameterized problem. An *OR-cross-composition of \mathcal{L} into \mathcal{P}* (with respect to \mathcal{R}) is an algorithm that, given t instances $I_1, I_2, \dots, I_t \in \Sigma^*$ of \mathcal{L} belonging to the same equivalence class of \mathcal{R} , takes time polynomial in $\sum_{i=1}^t |I_i|$ and outputs an instance $(I, k) \in \Sigma^* \times \mathbb{N}$ such that:

- i) the parameter value k is polynomially bounded in $\max\{|I_1|, \dots, |I_t|\} + \log t$,
- ii) the instance (I, k) is a **yes**-instance of \mathcal{P} if and only there is $i \in \{1, \dots, t\}$ such that I_i is a **yes**-instance of \mathcal{L} .

It is said that \mathcal{L} *OR-cross-composes into \mathcal{P}* if a cross-composition algorithm exists for a suitable relation \mathcal{R} .

Bodlaender, Jansen and Kratsch [3] proved the following theorem.

Theorem 6 ([3]). *If an NP-hard problem \mathcal{L} OR-cross-composes into the parameterized problem \mathcal{P} , then \mathcal{P} does not admit a polynomial kernelization unless $\text{NP} \subseteq \text{coNP/poly}$.*

Lemma 10. *There is $\phi \in \Sigma_2$ without free variables that has 3 variables such that VERTEX-REMOVAL TO ϕ has no polynomial kernel unless $\text{NP} \subseteq \text{coNP/poly}$.*

Proof. We define the formula ϕ as follows:

$$\phi = \exists x \forall y \forall z [((x \sim y) \wedge (x \sim z)) \rightarrow ((y = z) \vee (y \sim z))].$$

In terms of graphs, $G \models \phi$ means that there is a vertex x whose neighborhood is a clique.

We consider the CLIQUE problem:

CLIQUE

Input: A graph G and a positive integer k .

Question: Is there a clique in G with at least k vertices?

and show that CLIQUE OR-cross-composes into VERTEX-REMOVAL TO ϕ .

We say that two instances (G_1, k_1) and (G_2, k_2) of CLIQUE are *equivalent* if $|V(G_1)| = |V(G_2)|$ and $k_1 = k_2$.

Let $(G_1, k), \dots, (G_t, k)$ be equivalent instances of CLIQUE where graphs have n vertices. We construct the instance (G', k') of VERTEX-REMOVAL TO ϕ as follows.

- Construct disjoint copies of G_1, \dots, G_t .
- For every $i \in \{1, \dots, t\}$, construct $n - k + 2$ vertices $u_i^1, \dots, u_i^{n-k+2}$ and make them adjacent to the vertices of G_i .
- Set $k' = n - k$.

We claim that (G', k') is a **yes-instance** of VERTEX-REMOVAL TO ϕ if and only if there is $i \in \{1, \dots, t\}$ such that (G_i, k) is a **yes-instance** of CLIQUE.

Suppose that there is $i \in \{1, \dots, t\}$ such that (G_i, k) is a **yes-instance** of CLIQUE. Then G_i has a clique K of size k . Let $S = V(G) \setminus K$. Note that $|S| = n - k = k'$. Now for $x = u_i^1$, we have that the neighborhood of x in G' is the clique K , that is, (G', k') is a **yes-instance** of VERTEX-REMOVAL TO ϕ .

Assume that (G', k') is a **yes-instance** of VERTEX-REMOVAL TO ϕ . Then there is a set of vertices $S \subseteq V(G')$ of size at most k' such that $(G' - S) \models \phi$. Let $G'' = G' - S$. We have that there is $x \in V(G'')$ such that the neighborhood of x in G'' is a clique. Then there is $i \in \{1, \dots, t\}$ such that $x \in V(G_i)$ or $x \in \{u_i^1, \dots, u_i^{n-k+2}\}$. Suppose that $x \in V(G_i)$. Since $|S| \leq k' = n - k$, x is adjacent in G'' to at least two distinct vertices of $\{u_i^1, \dots, u_i^{n-k+2}\}$ but these two vertices are not adjacent. It implies that $x \in \{u_i^1, \dots, u_i^{n-k+2}\}$ and the neighborhood of x in G'' is $V(G_i) \setminus S$, that is, $K = V(G_i) - S$ is a clique. Because $|S| \leq k'$, we have that $|K| \geq n - k' = k$, that is, (G_i, k) is a **yes-instance** of CLIQUE.

Since $|V(G')| = \mathcal{O}(nt)$ and $k' = \mathcal{O}(n)$, we conclude that VERTEX-REMOVAL TO ϕ has no polynomial kernel unless $\text{NP} \subseteq \text{coNP/poly}$ by Theorem 6. \square

We have that Lemmata 9 and 10 imply Theorem 2 (ii).

5 Conclusion

In this paper we have provided necessary and sufficient conditions (subject to some complexity assumptions) on the fixed-parameter tractability, as well as polynomial kernelization, of graph modification problems to the properties expressible by an FOL-formula from a certain prefix class. While we stated our results for undirected graphs, in fact, all our results could be rewritten for directed graphs. In particular, the FPT and kernelization algorithms work for directed graphs without any changes. For the hardness proofs, we need only a minor modification. Denote by $\text{arc}(x, y)$ the predicate for variables x and y meaning that (x, y) is an arc of a directed graph. Denote by $\vec{\phi}$ the FOL-formula on directed graphs obtained from an FOL-formula ϕ on undirected graphs by replacing every predicate $x \sim y$ with $\text{arc}(x, y) \vee \text{arc}(y, x)$. Then we have the following observation.

Observation 2. *Let G be the underlying undirected graph of a directed graph D and let ϕ be an FOL-formula on undirected graphs without free variables. Then $G \models \phi$ if and only if $D \models \vec{\phi}$.*

Observation 2 immediately implies that whenever VERTEX REMOVAL TO ϕ or EDGE REMOVAL/COMPLETION/EDITING TO ϕ is hard (W[2]-hard or does not have a polynomial kernel unless $\text{NP} \subseteq \text{coNP/poly}$), the same holds for the variant of the problem on directed graphs. The straightforward reduction constructs a directed graph from an undirected graph G by turning its edges to arcs by assigning arbitrary orientations.

Our results are for FOL-formulas. It would be very interesting to obtain a similar type of dichotomies for prefix classes of *Monadic Second Order Logic* (MSOL) formulas on graphs. MSOL is substantially richer and allows to express more interesting graph properties like connectivity that cannot be expressed in FOL. The crucial difference is that while MODEL CHECKING for FOL-formulas can be solved in polynomial time for formulas of bounded size (see Theorem 5), the problem for MSOL is well-known to be NP-complete even for formulas whose size is bounded by a constant.

Acknowledgments. We are grateful to Pål Drange for his very helpful remarks.

References

- [1] F. N. ABU-KHZAM, *A kernelization algorithm for d -hitting set*, Journal of Computer and System Sciences, 76 (2010), pp. 524 – 531.
- [2] H. L. BODLAENDER, R. G. DOWNEY, M. R. FELLOWS, AND D. HERMELIN, *On problems without polynomial kernels*, J. Comput. Syst. Sci., 75 (2009), pp. 423–434.
- [3] H. L. BODLAENDER, B. M. P. JANSEN, AND S. KRATSCH, *Kernelization lower bounds by cross-composition*, SIAM J. Discrete Math., 28 (2014), pp. 277–305.
- [4] E. BÖRGER, E. GRÄDEL, AND Y. GUREVICH, *The classical decision problem*, Springer Science & Business Media, 2001.
- [5] L. CAI AND Y. CAI, *Incompressibility of H -free edge modification problems*, Algorithmica, 71 (2015), pp. 731–757.
- [6] M. CYGAN, F. V. FOMIN, L. KOWALIK, D. LOKSHTANOV, D. MARX, M. PILIPCZUK, M. PILIPCZUK, AND S. SAURABH, *Parameterized Algorithms*, Springer, 2015.
- [7] R. G. DOWNEY AND M. R. FELLOWS, *Fundamentals of Parameterized Complexity*, Texts in Computer Science, Springer, 2013.
- [8] R. FAGIN, *Generalized first-order spectra and polynomial-time recognizable sets*, in Complexity of Computation, vol. 7, AMS, 1974, pp. 43–74.
- [9] J. FLUM AND M. GROHE, *Parameterized Complexity Theory*, Texts in Theoretical Computer Science. An EATCS Series, Springer, 2006.
- [10] M. FRICK AND M. GROHE, *The complexity of first-order and monadic second-order logic revisited*, Ann. Pure Appl. Logic, 130 (2004), pp. 3–31.
- [11] G. GOTTLÖB, P. G. KOLAITIS, AND T. SCHWENTICK, *Existential second-order logic over graphs: Charting the tractability frontier*, J. ACM, 51 (2004), pp. 312–362.
- [12] M. GROHE, *Descriptive complexity, canonisation, and definable graph structure theory*, vol. 47, Cambridge University Press, 2017.
- [13] 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 (2013), pp. 900–926.
- [14] R. IMPAGLIAZZO, R. Paturi, AND F. ZANE, *Which problems have strongly exponential complexity?*, J. Comput. Syst. Sci., 63 (2001), pp. 512–530.
- [15] S. KRATSCH AND M. WAHLSTRÖM, *Two edge modification problems without polynomial kernels*, Discrete Optimization, 10 (2013), pp. 193–199.
- [16] J. M. LEWIS AND M. YANNAKAKIS, *The node-deletion problem for hereditary properties is NP-complete*, J. Comput. Syst. Sci., 20 (1980), pp. 219–230.
- [17] R. NIEDERMEIER, *Invitation to fixed-parameter algorithms*, vol. 31 of Oxford Lecture Series in Mathematics and its Applications, Oxford University Press, 2006.

- [18] C. SMORYNSKI, *The incompleteness theorems*, in Handbook of mathematical logic, vol. 90 of Stud. Logic Found. Math., North-Holland, Amsterdam, 1977, pp. 821–865.
- [19] M. Y. VARDI, *The complexity of relational query languages (extended abstract)*, in Proceedings of the 14th Annual ACM Symposium on Theory of Computing, May 5-7, 1982, San Francisco, California, USA, ACM, 1982, pp. 137–146.
- [20] R. WILLIAMS, *Faster decision of first-order graph properties*, in Joint Meeting of the Twenty-Third EACSL Annual Conference on Computer Science Logic (CSL) and the Twenty-Ninth Annual ACM/IEEE Symposium on Logic in Computer Science (LICS), CSL-LICS '14, Vienna, Austria, July 14 - 18, 2014, ACM, 2014, pp. 80:1–80:6.
- [21] M. YANNAKAKIS, *Edge-deletion problems*, SIAM Journal on Computing, 10 (1981), pp. 297–309.