Algorithmica (2018) 80:3481–3524
https://doi.org/10.1007/s00453-017-0401-6

A Polynomial Kernel for Trivially Perfect Editing

Pål Grønås Drange[1] · Michał Pilipczuk[2]

Received: 21 November 2016 / Accepted: 19 December 2017 / Published online: 28 December 2017

Abstract We give a kernel with $O(k^7)$ vertices for TRIVIALLY PERFECT EDITING, the problem of adding or removing at most k edges in order to make a given graph trivially perfect. This answers in affirmative an open question posed by Nastos and Gao (Soc Netw 35(3):439–450, 2013), and by Liu et al. (Tsinghua Sci Technol 19(4):346–357, 2014). Our general technique implies also the existence of kernels of the same size for related TRIVIALLY PERFECT COMPLETION and TRIVIALLY PERFECT DELETION problems. Whereas for the former an $O(k^3)$ kernel was given by Guo (in: ISAAC 2007, LNCS, vol 4835, Springer, pp 915–926, 2007), for the latter no polynomial kernel was known. We complement our study of TRIVIALLY PERFECT EDITING by proving that, contrary to TRIVIALLY PERFECT COMPLETION, it cannot be solved in time $2^{o(k)} \cdot n^{O(1)}$ unless the exponential time hypothesis fails. In this manner we complete the picture

A preliminary version of this work has appeared at ESA 2015 [13]; this full version contains complete proofs of all the results as well as new combinatorial corollaries about the sizes of minimal obstructions, and a lower bound for COGRAPH EDITING. A presentation of these results can be also found in the Ph.D. thesis of the first author [10]. The research leading to these results has received funding from the European Research Council under the European Union's Seventh Framework Programme (FP/2007-2013)/ERC Grant Agreement No. 267959 (P. Drange and M. Pilipczuk, while the latter was affiliated with the University of Bergen), and from the Polish National Science Centre Grant UMO-2013/11/D/ST6/03073 (M. Pilipczuk, after he moved to the University of Warsaw). Also, this work was partially done while M. Pilipczuk was holding a post-doc position at Warsaw Center of Mathematics and Computer Science.

✉ Michał Pilipczuk
michal.pilipczuk@mimuw.edu.pl

Pål Grønås Drange
pal.drange@ii.uib.no

1 Department of Informatics, University of Bergen, Bergen, Norway

2 Institute of Informatics, University of Warsaw, Warsaw, Poland

of the parameterized and kernelization complexity of the classic edge modification problems for the class of trivially perfect graphs.

Keywords Trivially perfect graph · Graph modification · Polynomial kernel

Mathematics Subject Classification 05C85 · 68W05 · 68R10

1 Introduction

Graph modification problems form an important subclass of discrete computational problems, where the task is to modify a given graph using a constrained number of modifications in order to make it satisfy some property Π, or equivalently belong to some class of graphs $\mathcal{G}$. Well-known examples of graph modification problems include Vertex Cover, Clique, Cluster Editing, Feedback Vertex Set, Odd Cycle Transversal, and Minimum Fill-In. The systematic study of graph modification problems dates back to the early 1980s and the work of Yannakakis [37], who showed that there is a dichotomy for the vertex deletion problems: unless a graph class $\mathcal{G}$ is trivial (finite or co-finite), the problem of deleting the least number of vertices to obtain a graph from $\mathcal{G}$ is NP-hard. However, when, in order to obtain a graph from $\mathcal{G}$, we are to modify the edge set of the graph instead of the vertex set, there are three natural classes of problems: deletion problems (deleting the least number of edges), completion problems (adding the least number of edges) and editing problems (performing the least number of edge additions and deletions). For neither of these is any complexity dichotomy in the spirit of Yannakakis' result known. Indeed, Yannakakis states that it

> *would be nice if the same kind of techniques could be applied to the edge-deletion problems. Unfortunately we suspect that this is not the case—the reductions we found for the properties considered* [...] *do not seem to fall into a pattern.*
> —Mihalis Yannakakis [37]

Even though for edge modification problems there is no general P versus NP classification known, much can be said about their parameterized complexity. Recall that a parameterized problem is called *fixed-parameter tractable* if it can be solved in time $f(k) \cdot n^{O(1)}$ for some computable function f, where n is the size of the input and k is its parameter. In our case, the natural parameter k is the allowed number of modifications. Cai [5] made a simple observation that for all the aforementioned graph modification problems there is a simple branching algorithm running in time $c^k n^{O(1)}$ for some constant c, as long as $\mathcal{G}$ is *characterized by a finite set of forbidden induced subgraphs*: there is a finite list of graphs $H_1, H_2, \ldots, H_p$ such that any graph G belongs to $\mathcal{G}$ if and only if G does not contain any H_i as an induced subgraph. Although many studied graph classes satisfy this property, there are important examples, like chordal or interval graphs, that are outside this regime.

For this reason, the parameterized analysis of modification problems for graph classes characterized by a finite set of forbidden induced subgraphs focused on studying the design of *polynomial kernelization algorithms* (*polynomial kernels*); recall that such an algorithm is required, given an input instance (G, k) of the problem, to preprocess it in polynomial time and obtain an equivalent output instance (G', k'), where $|G'|, k' \leq p(k)$ for some polynomial p. That is, the question is the following: can you, using polynomial-time preprocessing only, bound the size of the tackled instance by a polynomial function depending only on k?

For vertex deletion problems the answer is again quite simple: as long as $\mathcal{G}$ is characterized by a finite set of forbidden induced subgraphs, the task is to hit all the copies of these subgraphs (so-called *obstacles*) that are originally contained in the graph. Hence, one can construct a simple reduction to the d-Hitting Set problem for a constant d depending on $\mathcal{G}$, and use the classic $O(k^d)$ kernel for the latter that is based on the sunflower lemma (see e.g., [16, 19]). For edge modifications problems, however, this approach fails utterly: every edge addition/deletion can create new obstacles, and thus it is not sufficient to hit only the original ones. For this reason, edge modification problems behave counterintuitively w.r.t. polynomial kernelization, and up to recently very little was known about their complexity.

On the positive side, kernelization of edge modification problems for well-studied graph classes was explored by Guo [24], who showed that four problems: Threshold Completion, Split Completion, Chain Completion, and Trivially Perfect Completion, all admit polynomial kernels. However, the study took a turn for the interesting when Kratsch and Wahlström [28] showed that there is a graph H on seven vertices, such that the deletion problem to H-free graphs (the class of graphs not admitting H as an induced subgraph) does not admit a polynomial kernel, unless the polynomial hierarchy collapses. This shows that the subtle differences between edge modification and vertex deletion problems have tremendous impact on the kernelization complexity.

Kratsch and Wahlström conclude by asking whether there is a "simple" graph, like a path or a cycle, for which an edge modification problem does not admit a polynomial kernel under similar assumptions. The question was answered by Guillemot et al. [23] who showed that both for the class of P_ℓ-free graphs (for $\ell \geq 7$) and for the class of C_ℓ-free graphs (for $\ell \geq 4$), the edge deletion problems probably do not have polynomial kernelization algorithms. They simultaneously gave a cubic kernel for the Cograph Editing problem, the problem of editing to a graph without induced paths on four vertices.

These results were later improved by Cai and Cai [6], who tried to obtain a complete dichotomy of the kernelization complexity of edge modification problems for classes of H-free graphs, for every graph H. The project has been almost fully successful—the question remains unresolved only for a finite number of graphs H. In particular, it turns out that the existence of a polynomial kernel for any of H-Free Editing, H-Free Edge Deletion, or H-Free Completion problems is in fact a very rare phenomenon, and basically happens only for specific, constant-size graphs H. In particular, for H being a path or a cycle, the aforementioned three problems admit polynomial kernels if and only if H has at most three edges.

At the same time, there is a growing interest in identifying parameterized problems that are solvable in *subexponential parameterized time*, i.e., in time $2^{o(k)} n^{O(1)}$. Although for many classic parameterized problems already known NP-hardness reductions show that the existence of such an algorithm would contradict the *expo-*

nential time hypothesis of Impagliazzo et al. [25], subexponential parameterized algorithms were known to exist for problems in restricted settings, like planar, or more generally H-minor free graphs [8], or tournaments [1]. See the book of Flum and Grohe [16] for a wider discussion.

Therefore, it was an immense surprise when Fomin and Villanger [20] showed that Chordal Completion (also called Minimum Fill-In) can be solved in time $2^{O(\sqrt{k}\log k)}n^{O(1)}$. Following this discovery, a new line of research was initiated. Ghosh et al. [22] showed that Split Completion is solvable in the same running time. Although Komusiewicz and Uhlmann [27] showed that we cannot expect Cluster Editing to be solvable in subexponential parameterized time, as shown by Fomin et al. [17], when the number of clusters in the target graph is sublinear in the number of allowed edits, this is possible nonetheless.

Following these three positive examples, Drange et al. [12] showed that completion problems for trivially perfect graphs, threshold graphs and pseudosplit graphs all admit subexponential parameterized algorithms. Later, Bliznets et al. showed that both Proper Interval Completion and Interval Completion also admit subexponential parameterized algorithms [2,3].

Let us remark that in almost all these results, the known existence of a polynomial kernelization procedure for the problem played a vital role in designing the subexponential parameterized algorithm. Kernelization is namely used as an opening step that enables us to assume that the size of the considered graph is polynomial in the parameter k, something that turns out to be extremely useful in further reasoning. The only exception is the algorithm for the Interval Completion problem [3], for which the existence of a polynomial kernel remains a notorious open problem. The need of circumventing this issue created severe difficulties in the aforementioned result.

In this paper we study the Trivially Perfect Editing problem. Recall that trivially perfect graphs are exactly graphs that do not contain a P_4 or a C_4 as an induced subgraph; see Sect. 2.2 for a structural characterization of this graph class. Interest in trivially perfect graphs started with the attempts to prove the strong perfect graph theorem. In recent times, new source of motivation has grown, with the realization that trivially perfect graphs are related to the width parameter *treedepth* (called also vertex ranking number, ordered chromatic number, and minimum elimination tree height). Although it had been known that both the completion and the deletion problem for trivially perfect graphs are NP-hard, it was open for a long time whether the editing version is NP-hard as well [4,31].

This question was answered very recently by Nastos and Gao [33], who showed that the problem is indeed NP-hard. The work of Nastos and Gao focuses on exhibiting applications of trivially perfect graphs in social network theory, since this graph class may serve as a model for *familial groups*, communities in social networks showing a hierarchical nature. Specifically, the *editing number* to a trivially perfect graph[1] can be used as a measure of how much a social network resembles a collection of hierarchies. Nastos and Gao also ask whether it is possible to obtain a polynomial kernelization algorithm for this problem. The question about the existence of a polynomial kernel

[1] Nastos and Gao use the term *quasi-threshold graph* instead of a trivially perfect graph.

for TRIVIALLY PERFECT EDITING was then restated in a recent survey by Liu et al. [29], which *nota bene* contains a comprehensive overview of the current status of the research on the kernelization complexity of graph modification problems.

Our contribution. We answer the question of Nastos and Gao [33] and of Liu et al. [29] in affirmative by proving the following theorem.

Theorem 1 *The problem* TRIVIALLY PERFECT EDITING *admits a proper kernel with* $O(k^7)$ *vertices.*

Here, we say that a kernel (kernelization algorithm) is *proper* if it can only decrease the parameter, i.e., the output parameter k' satisfies $k' \leq k$.

To prove Theorem 1, we employ an extensive analysis of the tackled instance, based on the equivalent structural definition of trivially perfect graphs. The main approach is to construct a small *vertex modulator*, a set of vertices whose removal results in obtaining a trivially perfect graph. However, since we are allowed only edge deletions and additions, this modulator just serves as a tool for exposing the structure of the instance. More specifically, we greedily pack disjoint obstructions into a set X, whose size can be guaranteed to be at most $4k$, with the condition that to get rid of each of these obstructions, at least one edge must be edited inside the modulator per obstruction. Having obtained such a modulator, the rest of the graph, $G - X$, is trivially perfect, and we may apply the structural view on trivially perfect graphs to find irrelevant parts that can be reduced.

While the modulator technique is commonly used in kernelization, the new insight in this work is as follows. Since we work with an edge modification problem, we can be less restrictive about when an obstacle can be greedily packed into the modulator. For example, the obstacle does not need to be completely vertex-disjoint with the so far constructed X; sharing just one vertex is still allowed. This observation allows us to reason about the adjacency structure between X and $V(G)\setminus X$, which is of great help when identifying irrelevant parts.

After the announcement of this result at ESA 2015 [13], several results using the same basic technique have appeared: a quadratic vertex kernel for THRESHOLD EDITING and CHAIN EDITING [11], a cubic vertex kernel for DIAMOND-FREE DELETION [34], and a polynomial kernel for CLAW-DIAMOND-FREE DELETION [7]. We hope that this generic methodology will find applications in other edge modification problems as well.

By slight modifications of our kernelization algorithm, we also obtain polynomial kernels for TRIVIALLY PERFECT DELETION and TRIVIALLY PERFECT COMPLETION.

Theorem 2 *The problem* TRIVIALLY PERFECT DELETION *admits a proper kernel with* $O(k^7)$ *vertices.*

Theorem 3 *The problem* TRIVIALLY PERFECT COMPLETION *admits a proper kernel with* $O(k^7)$ *vertices.*

To the best of our knowledge, no polynomial kernel for TRIVIALLY PERFECT DELETION was known so far. For TRIVIALLY PERFECT COMPLETION, a cubic kernel was announced earlier by Guo [24]. Unfortunately, the work of Guo [24] is published only as a conference extended abstract, where it is only sketched how the approach yielding a quartic kernel for Split Deletion could be used to obtain a cubic kernel for Trivially Perfect Completion. The details, and indeed the rules of this kernelization algorithm are deferred to the full version, which, alas, has not appeared. For this reason, we believe that our proof of Theorem 3 fills an important gap in the literature—the polynomial kernel for Trivially Perfect Completion is an important ingredient of the subexponential parameterized algorithm for this problem [12].

We also note that our kernelization procedures can be also used to prove combinatorial upper bounds on the sizes of minimal obstructions to admitting an editing set of size k to a trivially perfect graph, under the induced subgraph order. More precisely, we say that a graph G is *a minimal obstruction for k-editing to a trivially perfect graph* if one cannot modify G by adding or removing at most k edges in order to obtain a trivially perfect graph, but every proper induced subgraph of G already has this property. In other words, (G, k) is a no-instance of Trivially Perfect Editing, but (G', k) is a yes-instance of Trivially Perfect Editing whenever G' is a proper induced subgraph of G. Similarly, we define being a minimal obstruction for k-completion and k-deletion to a trivially perfect graph. With these definitions in mind, the following result appears to be a simple corollary of our main results.

Theorem 4 *Every minimal obstruction for k-editing, k-completion, or k-deletion to a trivially perfect graph, has at most $O(k^7)$ vertices.*

Finally, we show that Trivially Perfect Editing, in addition to being NP-complete, cannot admit a subexponential parameterized algorithm, provided that the exponential time hypothesis holds.

Theorem 5 Trivially Perfect Editing *is NP-complete and, under ETH, cannot be solved in time $2^{o(k)} \operatorname{poly}(n)$ nor $2^{o(n+m)}$, even on graphs with maximum degree* 4.

In other words; the familial group measure cannot be computed in time subexponential in terms of the value of the measure. This stands in contrast with Trivially Perfect Completion and the related Threshold Editing [11] that admit subexponential parameterized algorithms, and shows that Trivially Perfect Editing is more similar to Trivially Perfect Deletion, for which a similar lower bound has been proved earlier by Drange et al. [12]. In fact, our reduction can be used as an alternative proof of hardness of Trivially Perfect Deletion as well.

Let us note that the NP-hardness reduction for Trivially Perfect Editing presented by Nastos and Gao [33] cannot be used to prove the nonexistence of a subexponential parameterized algorithm, since it involves a cubic blow-up of the parameter (see Sect. 6 for details). To prove Theorem 5, we resort to the technique used for similar hardness results by Komusiewicz and Uhlmann [27] and by Drange et al. [12]. Finally, we prove similar lower bounds for Cograph Editing. Even on graphs of degree at most four, Cograph Editing is NP-complete and assuming the exponential time hypothesis, does not admit a subexponential time algorithm.

2 Preliminaries

2.1 Graphs and Complexity

Graphs. In this work we consider only undirected simple finite graphs. For a graph G, by $V(G)$ and $E(G)$ we denote the vertex and edge set of G, respectively. The *size* of a graph G is defined as $|G| = |V(G)| + |E(G)|$.

For a vertex $v \in V(G)$, by $N_G(v)$ we denote the open neighborhood of v, i.e., $N_G(v) = \{u \in V(G) \mid uv \in E(G)\}$. The closed neighborhood of v, denoted by $N_G[v]$, is defined as $N_G(v) \cup \{v\}$. These notions are extended to subsets of vertices as follows: $N_G[X] = \bigcup_{v \in X} N_G[v]$ and $N_G(X) = N_G[X] \setminus X$. We omit the subscript whenever G is clear from context.

When $U \subseteq V(G)$ is a subset of vertices of G, we write $G[U]$ to denote the *induced subgraph* of G, i.e., the graph $G' = (U, E_U)$ where E_U is $E(G)$ restricted to U. The degree of a vertex $v \in V(G)$, denoted $\deg_G(v)$, is the number of vertices it is adjacent to, i.e., $\deg_G(v) = |N_G(v)|$. We denote by $\Delta(G)$ the maximum degree in the graph, i.e., $\Delta(G) = \max_{v \in V(G)} \deg(v)$. For a set A, we write $\binom{A}{2}$ to denote the set of unordered pairs of elements of A; thus $E(G) \subseteq \binom{V(G)}{2}$. By $\overline{G}$ we denote the *complement* of a graph G, i.e., $V(\overline{G}) = V(G)$ and $E(\overline{G}) = \binom{V(G)}{2} \setminus E(G)$.

If v and u are such that $N[v] = N[u]$, then we call v and u *true twins*. Observe that v and u are adjacent if they are true twins. On the other hand, if v and u have $N(v) = N(u)$, then we call v and u *false twins*, and in this case we may observe that v and u are non-adjacent. If X is an inclusion-wise maximal set of vertices such that for every pair of vertices v and u in X they are true (resp. false) twins, then we call X a true (resp. false) twin class.

For a graph G and a set of vertices $X \subseteq V(G)$, we denote by $G - X$ the (induced subgraph) $G[V(G) \setminus X]$. When $F \subseteq \binom{V(G)}{2}$, we write $G - F$ to denote the graph G' on vertex set $V(G)$ and edge set $E(G) \setminus F$. Finally, we let $G \triangle F$ be the graph on vertex set $V(G)$ and edge set $E(G) \triangle F$, where $\triangle$ denotes the *symmetric difference*; For two sets A and B, $A \triangle B = (A \setminus B) \cup (B \setminus A)$. We will also say that two sets A and B are *nested* if $A \subseteq B$ or $B \subseteq A$.

A vertex $v \in V(G)$ is *universal* if it is adjacent to all the other vertices of the graph. Note that the set of universal vertices of a graph forms a clique, which is also a true twin class. This clique will be denoted by $\mathrm{uni}(G)$ and called the *universal clique* of G.

Modules and the modular decomposition. In our kernelization algorithm we will use the notion of a *module* in a graph.

Definition 1 Given a graph G, a set of vertices $M \subseteq V(G)$ is called a *module* if for any two vertices v and u in M, we have that $N(v) \setminus M = N(u) \setminus M$, i.e., all the vertices of M have exactly the same neighborhood outside M.

Observe that for any graph G, any singleton $M = \{v\}$ is a module, and also $V(G)$ itself is a module. However, G can contain a whole hierarchy of modules. This hierarchy can be captured using the following notion of a *modular decomposition*, introduced by Gallai [21]. The following description of a modular decomposition is taken verbatim from the work of Bliznets et al. [3].

A module decomposition of a graph G is a rooted tree T, where each node t is labeled by a module $M^t \subseteq V(G)$, and is one of four types:

Leaf t is a leaf of T, and M^t is a singleton;

Union $G[M^t]$ is disconnected, and the children of t are labeled with different connected components of $G[M^t]$;

Join the complement of $G[M^t]$ is disconnected, and the children of t are labeled with different connected components of the complement of $G[M^t]$;

Prime neither of the above holds, and the children of t are labeled with different modules of G that are proper subsets of M^t, and are inclusion-wise maximal with this property.

Moreover, we require that the root of T is labeled with the module $V(G)$. We need the following properties of the module decomposition.

Theorem 6 (See [32]) *For a graph G, the following holds.*

1. *A module decomposition $(T, (M^t)_{t \in V(T)})$ of G exists, is unique, and computable in linear time.*
2. *At any prime node t of T, the labels of the children form a partition of M^t. In particular, for each vertex v of G there exists exactly one leaf node with label $\{v\}$.*
3. *Each module M of G is either a label of some node of T, or there exists a* ***union*** *or* ***join*** *node t such that M is a union of labels of some children of t.*

Let us remark that since in this work we do not optimize the running time of the kernelization algorithm, we do not need to compute the modular decomposition in linear time. Any simpler polynomial time algorithm would suffice (see the work of McConnell and Spinrad [32] for a literature overview).

Parameterized complexity The running time of an algorithm is usually described as a function of the length of the input. To refine the complexity analysis of computationally hard problems, parameterized complexity introduced the notion of an extra "parameter" that is an additional part of a problem instance responsible for measuring its complexity. To simplify the notation, we will consider inputs to problems of the form (G, k), which is a pair consisting of a graph G and a nonnegative integer k. A problem is then said to be *fixed parameter tractable* if there is an algorithm which solves the problem in time $f(k) \cdot \mathrm{poly}(|G|)$, where f is any function, and $\mathrm{poly} \colon \mathbb{N} \to \mathbb{N}$ any polynomial function. In the case when $f(k) = 2^{o(k)}$ we say that the algorithm is a subexponential parameterized algorithm. When a problem $\Pi \subseteq \mathcal{G} \times \mathbb{N}$ is fixed-parameter tractable, where $\mathcal{G}$ is the class of all graphs, we say that Π belongs to the complexity class FPT. For a more rigorous introduction to parameterized complexity we refer to the books of Downey and Fellows [9] and of Flum and Grohe [16].

A *kernelization algorithm* (or *kernel*) is a polynomial-time algorithm for a parameterized problem Π that takes as input a problem instance (G, k) and returns an equivalent instance (G', k'), i.e., $(G, k) \in \Pi \Leftrightarrow (G', k') \in \Pi$, where both $|G'|$ and k' are bounded by $f(k)$ for some function f. We then say that f is the *size of the kernel*. When $k' \leq k$, we say that the kernel is a *proper kernel*. Specifically, a proper polynomial kernelization algorithm for Π is a polynomial time algorithm which takes as input an instance (G, k) and returns an equivalent instance (G', k') with $k' \leq k$ and $|G'| \leq p(k)$ for some polynomial function p.

Fig. 1 *Trivially perfect graphs are* $\{C_4, P_4\}$-free

(a) P_4 **(b)** C_4

Tools for lower bounds. As evidence that Trivially Perfect Editing cannot be solved in subexponential parameterized time $2^{o(k)}n^{O(1)}$, we will use the Exponential Time Hypothesis (ETH), formulated by Impagliazzo et al. [25]:

Hypothesis 1 (ETH) *There exists a positive real s such that* 3Sat *with n variables and m clauses cannot be solved in time* $2^{sn}(n+m)^{O(1)}$.

Impagliazzo et al. [25] proved a fundamental result called *Sparsification Lemma*, which can serve as a Turing reduction from an arbitrary instance of 3Sat to an instance where the number of clauses is linear in the number of variables. Thus, the following statement is an immediate corollary of the Sparsification Lemma.

Proposition 1 [25] *Unless ETH fails, there exists a positive real number s such that* 3Sat *with n variables and m clauses cannot be solved in time* $2^{s(n+m)}(n+m)^{O(1)}$. *In particular,* 3Sat *does not admit an algorithm with time complexity* $2^{o(n+m)}$.

2.2 Trivially Perfect Graphs

Combinatorial properties. A graph G is trivially perfect if and only if it does not contain a C_4 or a P_4 as an induced subgraph. That is, trivially perfect graphs are defined by the forbidden induced subgraph family $F = \{C_4, P_4\}$ (see Fig. 1). However, we mostly rely on the following recursive characterization of the trivially perfect graphs:

Proposition 2 [26] *The class of trivially perfect graphs can be defined recursively as follows:*

- K_1 *is a trivially perfect graph.*
- *Adding a universal vertex to a trivially perfect graph results in a trivially perfect graph.*
- *The disjoint union of two trivially perfect graphs results in a trivially perfect graph.*

Based on Proposition 2, a superset of the current authors [12] proposed the following notion of a decomposition for trivially perfect graphs. In the following, for a rooted tree T and vertex $t \in V(T)$, by T_t we denote the subtree of T rooted at t.

Definition 2 (*Universal clique decomposition,* [12]) A *universal clique decomposition* (*UCD*) of a connected graph G is a pair $\mathcal{T} = (T = (V_T, E_T), \mathcal{B} = \{B_t\}_{t \in V_T})$, where T is a rooted tree and $\mathcal{B}$ is a partition of the vertex set $V(G)$ into disjoint nonempty subsets, such that

- if $vw \in E(G)$ and $v \in B_t$, $w \in B_s$, then either $t = s$, t is an ancestor of s in T, or s is an ancestor of t in T, and

– for every node $t \in V_T$, the set of vertices B_t is the universal clique of the induced subgraph $G[\bigcup_{s \in V(T_t)} B_s]$.

We call the vertices of T *nodes* and the sets in $\mathcal{B}$ *bags* of the universal clique decomposition $(T, \mathcal{B})$. By slightly abusing notation, we often identify nodes with corresponding bags. Note that by the definition, in a universal clique decomposition every non-leaf node t has at least two children, since otherwise the bag B_t would not comprise *all* the universal vertices of the graph $G[\bigcup_{s \in V(T_t)} B_s]$.

The following lemma explains the connection between trivially perfect graphs and universal clique decompositions.

Lemma 1 [12] *A connected graph G admits a universal clique decomposition if and only if it is trivially perfect. Moreover, such a decomposition is unique up to isomorphisms.*

Note that a universal clique decomposition can trivially be found in polynomial time by repeatedly locating universal vertices and connected components. Moreover, we can extend the notion of a universal clique decomposition also to a disconnected trivially perfect graph G. In this case, the universal clique decomposition of G becomes a rooted forest consisting of universal clique decompositions of the connected components of G. Since a graph is trivially perfect if and only if each of its connected component is, Lemma 1 can be easily generalized to the following statement: Every (possibly disconnected) graph G is trivially perfect if and only if it admits a universal clique decomposition, where the decomposition has the shape of a rooted forest. Moreover, this decomposition is unique up to isomorphism.

The following definition of a quasi-ordering of vertices respecting the UCD will be helpful when arguing the correctness of the kernelization procedure.

Definition 3 Let $(T, \mathcal{B})$ be the universal clique decomposition of a trivially perfect graph G. We impose a quasi-ordering $\preceq$ on vertices of G defined as follows. Suppose vertex u belongs to bag B_t and vertex v belongs to bag B_s. Then $u \preceq v$ if and only if $t = s$ or t is an ancestor of s in the rooted forest T.

Thus, classes of vertices pairwise equivalent with respect to $\preceq$ are exactly formed by the bags of $\mathcal{B}$, and otherwise the ordering respects the rooted structure of T. Note that since the UCD of a trivially perfect graph is unique up to isomorphism, the quasi-ordering $\preceq$ is uniquely defined and can be computed in polynomial time.

Computational problems. In this work we are mainly interested in the Trivially Perfect Editing problem, defined formally as follows:

Trivially Perfect Editing	
Input:	A graph G and a non-negative integer k.
Parameter:	k
Question:	Is there a set $S \subseteq \binom{V(G)}{2}$ of size at most k such that $G \triangle S$ is trivially perfect?

For a graph G, any set $F \subseteq \binom{V(G)}{2}$ for which $G \triangle F$ is trivially perfect will henceforth be referred to as an *editing set*.

In the Trivially Perfect Deletion and Trivially Perfect Completion problems we allow only edge deletions and edge additions, respectively. More formally, we require that the editing set S is contained in, or disjoint from $E(G)$, respectively. In Sect. 3 we prove Theorem 1, that is, we show that Trivially Perfect Editing admits a kernel with $O(k^7)$. Actually, the character of our data reduction rules will be very simple; The kernelization algorithm will start with instance (G, k), and perform only the following operations:

- edit some $e \in \binom{V(G)}{2}$, decrement the budget k by 1, and terminate the algorithm if k becomes negative; or
- remove some vertex u of G and proceed with instance $(G - u, k)$.

Thus, the kernel will essentially be an induced subgraph of G, modulo performing some edits whose safeness and necessity can be deduced. In the proofs of correctness, we will never use any minimality argument that exchanges edge deletions for completions, or vice versa. Therefore, the whole approach can be applied almost verbatim to Trivially Perfect Deletion and Trivially Perfect Completion, yielding proofs for Theorems 2 and 3 after very minor modifications. We hope that the reader will be convinced about this after understanding all the arguments of Sect. 3. However, for the sake of completeness we, in Sect. 4, review the modifications of the argumentation of Sect. 3 that are necessary to prove Theorems 2 and 3.

Weakly laminar set systems. In the kernelization algorithm we will need the following auxiliary definition and result.

Definition 4 (*Weakly laminar set system*) A set system $\mathcal{F} \subseteq 2^U$ over a ground set U is called a *weakly laminar set system* if for every X_1 and X_2 in $\mathcal{F}$ with $x_1 \in X_1 \backslash X_2$ and $x_2 \in X_2 \backslash X_1$, there is no $Y \in \mathcal{F}$ with $\{x_1, x_2\} \subseteq Y$.

We now show that the size of a weakly laminar set system is bounded linearly in the size of the ground set.

Lemma 2 *Let $\mathcal{F}$ be a weakly laminar set system over a finite ground set U. Then the cardinality of $\mathcal{F}$ is at most $|U| + 1$.*

Proof We proceed by induction on $|U|$, with the claim being trivial when $U = \emptyset$. Let $\mathcal{F}$ be a weakly laminar set system over a nonempty ground set U, and suppose $|\mathcal{F}| \geq 2$ for otherwise we are done. Let Y and Z be a pair of different sets from $\mathcal{F}$ for which $|Y \cap Z|$ is maximized. As Y and Z are different, without loss of generality suppose $Y \backslash Z$ is nonempty, and let x be any element of $Y \backslash Z$.

We claim that Y is the only set of $\mathcal{F}$ that contains x. Suppose that, on the contrary, there is some other $W \in \mathcal{F}$ such that $x \in W$. We have that $Y \cap Z \nsubseteq W$, for otherwise we would have $|W \cap Y| > |Y \cap Z|$, a contradiction to the choice of the pair (Y, Z). Hence, there is some element $y \in (Y \cap Z) \backslash W = Y \cap (Z \backslash W)$. Since $x \in Y \cap (W \backslash Z)$, we obtain a contradiction with the definition of a weakly laminar set system for $X_1 = W$, $X_2 = Z$, $x_1 = x$, $x_2 = y$, and Y.

Consequently, indeed Y is the only set from $\mathcal{F}$ that contains x. Consider a set system $\mathcal{F}' = \mathcal{F}\setminus\{Y\}$ over the ground set $U' = U\setminus\{x\}$. Obviously $\mathcal{F}'$ is also a weakly laminar set system, so by induction we have $|\mathcal{F}'| \leq |U'| + 1 = |U|$. Hence $|\mathcal{F}| = |\mathcal{F}'| + 1 \leq |U| + 1$, as claimed. □

The proof given above is due to Peter Novotný, and was suggested after proposing the lemma as a competition problem for the 2015 Czech–Polish–Slovak Mathematical Match. The argument replaced our previous, slightly longer proof. We also remark that Lemma 2 can be directly inferred from the well-known Sauer–Shelah lemma [35, 36], since every weakly laminar family has VC dimension at most 1; this follows immediately from the definition. Since the presented proof of Lemma 2 is very easy, we included it for the sake of being self-contained.

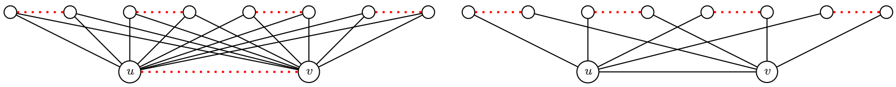
3 A Kernel for Trivially Perfect Editing

This section is devoted to the proof of Theorem 1, stating that Trivially Perfect Editing admits a proper kernel with $O(k^7)$ vertices. As usual, the kernelization algorithm will be given as a sequence of *data reduction rules*: simple preprocessing procedures that, if applicable, simplify the instance at hand. For each rule we shall prove two results: (a) that applicability of the rule can be recognized in polynomial time, and (b) that the rule is *safe*, i.e., the resulting instance is equivalent to the input one. At the end of the proof we will argue that if no rule is applicable, then the size of the instance must be bounded by $O(k^7)$. Some rules will decrement the budget k for edge edits; if this budget drops below zero, we may conclude that we are dealing with a no-instance, so we immediately terminate the algorithm and provide a constant-size trivial no-instance as the obtained kernel, for example the instance $(C_4, 0)$.

Before starting the formal description, let us give a brief overview of the structure of the proof. In Sect. 3.1 we give some preliminary basic rules, which mostly deal with situations where we can find a large number of induced C_4s and P_4s in the graph (henceforth called *obstacles*), which share only one edge or non-edge. We then infer that this edge or non-edge has to be included in any editing set of size at most k, and hence we can perform the necessary edit and decrement the budget.

In Sect. 3.2 we perform a greedy algorithm that iteratively packs disjoint induced C_4s and P_4s in the graph. Note that if we are able to pack more than k of them, then this certifies that the considered instance does not have a solution, and we can terminate the algorithm. Hence, if X is the union of vertex sets of the packed obstacles, then $|X| \leq 4k$ and $G - X$ is a trivially perfect graph. Uncovering such a set X, which we call a TP-modulator, imposes a lot of structure on the considered instance, and is the key for further analysis of irrelevant parts of the input.

Although the applied modulator technique is standard in the area of kernelization for graph modification problems, in this paper we introduce a new twist to it that may have possible further applications. Namely, we observe that since we consider edge editing problems, the packed obstacles do not have to be entirely vertex-disjoint, but the next obstacle can be packed even if it shares one vertex with the union of vertex sets of the previous obstacles; in some limited cases even having two vertices in common is permitted. Thus, the obtained modulator X has the property that not only is there



(a) Rule 1: There are four C_4s sharing only the vertices u and v. Unless the edge uv is added, we must use at least as many edits as the size of the non-matching.

(b) Rule 2: There are four P_4s sharing only the vertices u and v. Unless the edge uv is deleted, we must use at least as many edits as the size of the non-matching.

Fig. 2 Illustrations of Rules 1 and 2. The red dotted edges are non-edges; They form a matching in the complement graph. In each of the cases, the only common vertices are u and v (Color figure online)

no obstacle in the graph G that is vertex-disjoint with X, but even the existence of obstacles sharing one vertex with X is forbidden. This simple observation enables us to reason about the adjacency structure between X and $V(G) \setminus X$. In Sect. 3.3 we analyze this structure in order to prove the most important technical result of the proof: The number of subsets of X that are neighborhoods within X of vertices from $V(G) \setminus X$ is bounded polynomially in k; see Lemma 7.

In Sect. 3.4 we proceed to analyze the trivially perfect graph $G - X$. Having the polynomial bound on the number of neighborhoods within X, we can locate in the UCD of $G - X$ a polynomial (in k) number of *important bags*, where something interesting from the point of view of X-neighborhoods happens. The parts between the important bags have very simple structure. They are either *tassels*: sets of trees hanging below some important bag, where each such tree is a module in the whole graph G; or *combs*: long paths stretched between two important bags where all the vertices of subtrees attached to the path have exactly the same neighborhood in X. Tassels and combs are treated differently: large tassels contain large trivially perfect modules in G that can be reduced quite easily, however for combs we need to devise a quite complicated irrelevant vertex rule that locates a vertex that can be safely discarded in a long comb. The module reduction rules are described in Sect. 3.5, while in Sect. 3.6 we reduce the sizes of tassels and combs and conclude the proof.

3.1 Basic Rules

In this section we introduce the first two basic reduction rules (see Fig. 2). In the argumentation of the next sections, we will assume that none of these rules is applicable. An instance satisfying this property will be called *reduced*.

Rule 1 *For an instance* (G, k) *with* $uv \notin E(G)$*, if there is a matching of size at least* $k+1$ *in* $\overline{G[N(u) \cap N(v)]}$*, then add edge* uv *to* G *and decrease* k *by one, i.e., return the new instance* $(G + uv, k - 1)$.

Rule 2 *For an instance* (G, k) *with* $uv \in E(G)$ *and* $N_1 = N(u) \setminus N[v]$ *and* $N_2 = N(v) \setminus N[u]$*, if there is a matching in* $\overline{G}$ *between* N_1 *and* N_2 *of size at least* $k+1$*, then delete edge* uv *from* G *and decrease* k *by one, i.e., return the new instance* $(G - uv, k - 1)$.

Lemma 3 *Applicability of Rules 1 and 2 can be recognized in polynomial time. Moreover, both these rules are safe, i.e., the input instance (G, k) is a yes-instance if and only if the output instance $(G', k-1)$ is a yes-instance.*

Proof Observe that verifying the applicability of Rule 1 or of Rule 2 to a fixed (non-)edge uv boils down to computing the cardinality of the maximum matching in an auxiliary graph. This problem is well-known to be solvable in polynomial time [14]. Thus, by iterating over all edges and non-edges of G we obtain polynomial time algorithms for recognizing applicability of Rules 1 and 2. We proceed to the proof of the safeness for both rules.

Rule 1 Let $x_0 y_0, x_1 y_1, \ldots, x_k y_k$ be edges of the found matching in $\overline{G[N(u) \cap N(v)]}$. Observe that for each $i, 0 \leq i \leq k$, vertices u, x_i, v, y_i induce a C_4 in G. These induced C_4s share only the non-edge uv, hence any editing set that does not contain uv must contain at least one element of $\binom{\{u,x_i,v,y_i\}}{2} \setminus \{uv\}$, and consequently be of size at least $k+1$. We infer that every editing set for G that has size at most k has to include the edge uv, and the safeness of the rule follows.

Rule 2 We proceed similarly as for Rule 1. Suppose $x_0 y_0, x_1 y_1, \ldots, x_k y_k$ is the found matching in $\overline{G}$, where $x_i \in N_1$ and $y_i \in N_2$ for $0 \leq i \leq k$. Then vertices x_i, u, v, y_i induce a P_4, and all these P_4s for $0 \leq i \leq k$ pairwise share only the edge uv. Similarly as for Rule 1, we conclude that every editing set for G of size at most k has to contain uv, and the safeness of the rule follows. □

We can now use Lemma 3 to apply Rules 1 and 2 exhaustively; note that each application reduces the budget k, hence at most k applications can be performed before discarding the instance as a no-instance. From now on, we assume that the considered instance (G, k) is reduced.

3.2 Modulator Construction

We now move to the construction of a small modulator whose *raison d'être* is to expose structure in the considered graph G. We say that a subset $W \subseteq V(G)$ with $|W| = 4$ is an *obstruction* if $G[W]$ is isomorphic to a C_4 or a P_4. Formally, our modulator will be compliant to the following definition.

Definition 5 (*TP-modulator*) Let (G, k) be an instance of Trivially Perfect Editing. A subset $X \subseteq V(G)$ is a *TP-modulator* if for every obstruction W the following holds (see Fig. 3):

- $|W \cap X| \geq 2$, and
- if $|W \cap X| = 2$, then it cannot happen that $G[W]$ is a C_4 of the form $x_1 - y_1 - y_2 - x_2 - x_1$ or a P_4 of the form $x_1 - y_1 - y_2 - x_2$, where $W \cap X = \{x_1, x_2\}$.

We call a TP-modulator X *small* if $|X| \leq 4k$.

In particular, observe that for a TP-modulator X there is no obstacle disjoint with X, so $G - X$ is trivially perfect. The following result shows that from now we can assume that a small TP-modulator is given to us.

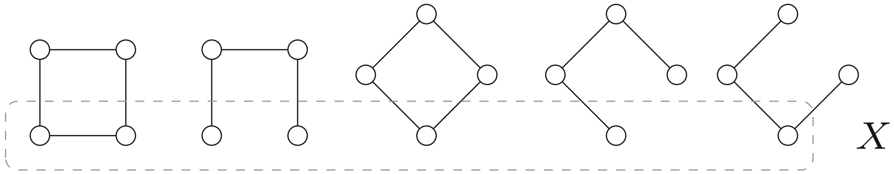


Fig. 3 Forbidden patterns of intersection between an obstruction and a TP-modulator X

Lemma 4 *Given an instance* (G, k) *for* Trivially Perfect Editing*, we can in polynomial time construct a small TP-modulator* $X \subseteq V(G)$*, or correctly conclude that* (G, k) *is a no-instance.*

Proof The algorithm starts with $X_0 = \emptyset$, and iteratively constructs an increasing family of sets $X_0 \subseteq X_1 \subseteq X_2 \subseteq \cdots$. In the ith iteration we look for an obstacle W that contradicts the fact that X_{i-1} is a TP-modulator according to Definition 5, by verifying all the quadruples of vertices in $O(n^4)$ time. If this check verifies that X_{i-1} is a TP-modulator, then we terminate the algorithm and output $X = X_{i-1}$. Otherwise, we set $X_i = X_{i-1} \cup W$ and proceed to the next iteration. Moreover, if we performed $k+1$ iterations, i.e., successfully constructed set X_{k+1}, then we terminate the algorithm concluding that (G, k) is a no-instance. Since in each iteration the next X_i grows by at most 4 vertices, we infer that if we succeed in outputting a TP-modulator X, then it has size at most $4k$.

We are left with proving that if the algorithm successfully constructed X_{k+1}, then (G, k) is a no-instance. To this end, we prove by induction on i that for every $i = 0, 1, \ldots, k+1$ and every editing set F for G, it holds that $|F \cap \binom{X_i}{2}| \geq i$. Indeed, from this statement for $i = k+1$ we can infer that every editing set for G has size at least $k+1$, so (G, k) is a no-instance. The base of the induction is trivial, so for the induction step suppose that $X_i = X_{i-1} \cup W$, where W is an obstacle with $|W \cap X_{i-1}| \leq 1$ or having the form described in the second point of Definition 5.

First, if $|W \cap X_{i-1}| \leq 1$, then $\binom{W}{2}$ is disjoint with $\binom{X_{i-1}}{2}$. Since F is an editing set for G, we have that $F \cap \binom{W}{2} \neq \emptyset$, and hence

$$\left|F \cap \binom{X_i}{2}\right| \geq \left|F \cap \binom{X_{i-1}}{2}\right| + \left|F \cap \binom{W}{2}\right| \geq i - 1 + 1 = i,$$

by the induction hypothesis. Second, if $|W \cap X_{i-1}| = 2$ and W has one of the two forms described in the second point of Definition 5, then it is easy to see that F in fact has to have a nonempty intersection with $\binom{W}{2} \setminus \{x_1x_2\}$: editing only the (non)edge x_1x_2 would turn a C_4 into a P_4 or vice versa. Since $\binom{W}{2} \setminus \{x_1x_2\}$ is disjoint with $\binom{X_{i-1}}{2}$, we analogously obtain that

$$\left|F \cap \binom{X_i}{2}\right| \geq \left|F \cap \binom{X_{i-1}}{2}\right| + \left|F \cap \left(\binom{W}{2} \setminus \{x_1x_2\}\right)\right| \geq i - 1 + 1 = i.$$

□

By applying Lemma 4, from now on we assume that we are given a small TP-modulator X in G.

3.3 Bounding the Number of Neighborhoods in a TP-modulator

Recall that we exposed a small TP-modulator X in the input graph G. In polynomial time we compute the universal clique decomposition $\mathcal{T} = (T, \mathcal{B})$ of the trivially perfect graph $G - X$. The goal of this section is to analyze the structure of neighborhoods within X of vertices residing outside X.

Definition 6 (*X-neighborhood*) Let G be a graph and $X \subseteq V(G)$. For a vertex $v \in V(G) \setminus X$, the *X-neighborhood* of v, denoted $N_G^X(v)$, is the set $N_G(v) \cap X$. The family of X-neighborhoods of G is the set $\{N_G^X(v) \colon v \in V(G) \setminus X\}$.

Again, we shall omit the subscript G whenever this does not lead to any confusion. Recall that the UCD $\mathcal{T}$ gives us a quasi-ordering $\preceq$ on the vertices of $G - X$. We have $u \preceq v$ if the bag to which v belongs is a descendant of the bag which u belongs to, where every bag is considered its own descendant. We shall use the notation $u \prec v$ to denote that $u \preceq v$ and $v \not\preceq u$. The following two lemmas show that the quasi-ordering $\preceq$ is compatible with the inclusion ordering of X-neighborhoods.

Lemma 5 *If $u \prec v$ then $N^X(u) \supseteq N^X(v)$.*

Proof Suppose $u \in B_t$ and $v \in B_s$, where $t \neq s$ and t is an ancestor of s in the forest T. Recall that in a UCD, every non-leaf node has at least two children, which means that there exists some node s' that is a descendant of t, but which is incomparable with s. Let w be any vertex of $B_{s'}$. From the definition of a UCD it follows that $uv, uw \in E(G)$ but $vw \notin E(G)$.

For the sake of contradiction suppose that $N^X(u) \not\supseteq N^X(v)$, which means there exists a vertex $x \in X$ with $xv \in E(G)$ and $xu \notin E(G)$. It follows that $\{x, u, v, w\}$ is an obstacle regardless of whether wx is an edge or a non-edge: it is an induced C_4 if $wx \in E(G)$ and an induced P_4 if $wx \notin E(G)$. Thus we have uncovered an obstacle sharing only one vertex with X, contradicting the fact that X is a TP-modulator. □

Lemma 6 *If $u, v \in B_t$ for some $B_t \in \mathcal{B}$, then*

$$N^X(u) \subseteq N^X(v) \quad or \quad N^X(v) \subseteq N^X(u).$$

Proof Since $u, v \in B_t$, we have that $uv \in E(G)$. For the sake of contradiction, suppose that there exist some $x_u \in N^X(u) \setminus N^X(v)$ and $x_v \in N^X(v) \setminus N^X(u)$. It can be now easily seen that regardless whether $x_u x_v$ belongs to $E(G)$ or not, the quadruple $\{u, v, x_u, x_v\}$ forms one of the obstacles forbidden in the second point of the Definition 5. This is a contradiction with the fact that X is a TP-modulator. □

Lemmas 5 and 6 motivate the following refinement of the quasi-ordering $\preceq$: If u, v belong to different bags of $\mathcal{T}$, then we put $u \preceq_N v$ if and only if $u \preceq v$, and if they are in the same bag, then $u \preceq_N v$ if and only if $N^X(u) \supseteq N^X(v)$. Thus, by Lemma 6 $\preceq_N$ refines $\preceq$ by possibly splitting every bag of $\mathcal{T}$ into a family of linearly ordered equivalence classes. Moreover, by Lemmas 5 and 6 we have the following corollary.

Corollary 1 *If* $u \preceq_N v$ *then* $N^X(u) \supseteq N^X(v)$.

Observe that for a pair of vertices $u, v \in V(G)\backslash X$, the following conditions are equivalent: (a) u and v are comparable w.r.t $\preceq$, (b) u and v are comparable w.r.t. $\preceq_N$, and (c) $uv \in E(G)$. We have now prepared all the tools needed to prove the main lemma from this section.

Lemma 7 *If* (G, k) *is a reduced instance for* Trivially Perfect Editing *and* X *is a small TP-modulator, then the number of different* X*-neighborhoods is at most* $O(k^4)$.

Proof Let $\mathcal{F}$ be the family of X-neighborhoods in G. For every $Z \in \mathcal{F}$, let us choose an arbitrary vertex $v_Z \in V(G)\backslash X$ with $Z = N^X(v_Z)$. We split $\mathcal{F}$ into two subfamilies: The first family $\mathcal{F}_1$ contains all the sets of $\mathcal{F}$ that contain the endpoints of some non-edge in $G[X]$, whereas the second family $\mathcal{F}_2$ contains all the sets of $\mathcal{F}$ that induce complete graphs in $G[X]$. We bound the sizes of $\mathcal{F}_1$ and $\mathcal{F}_2$ separately. *Bounding* $|\mathcal{F}_1|$: Let xy be a non-edge of $G[X]$, and for $2 \leq \kappa \leq |X|$ let $\mathcal{F}_1^{xy,\kappa}$ be the family of those sets of $\mathcal{F}_1$ that contain $\{x, y\}$ and have cardinality exactly κ. Take any distinct $Z_1, Z_2 \in \mathcal{F}_1^{xy,\kappa}$, and observe that they are not nested since both have size κ. By Corollary 1, this means that vertices v_{Z_1} and v_{Z_2} are incomparable w.r.t. $\preceq_N$, so $v_{Z_1}v_{Z_2} \notin E(G)$. Hence, set $\{v_Z : Z \in \mathcal{F}_1^{xy,\kappa}\}$ is independent in G. Observe now that if we had that $|\{v_Z : Z \in \mathcal{F}_1^{xy,\kappa}\}| \geq 2k + 2$, then Rule 1 would be applicable to the non-edge xy. Since we assume that the instance is reduced, we conclude that $|\{v_Z : Z \in \mathcal{F}_1^{xy,\kappa}\}| \leq 2k + 1$, and hence also $|\mathcal{F}_1^{xy,\kappa}| \leq 2k + 1$. By summing through all the κ between 2 and $|X|$ and through all the non-edges of $G[X]$, we infer that

$$|\mathcal{F}_1| \leq \binom{4k}{2} \cdot 4k \cdot (2k + 1) = O(k^4).$$

Bounding $|\mathcal{F}_2|$: Consider any pair of X-neighborhoods $Z_1, Z_2 \in \mathcal{F}_2$ such that they are not nested, and moreover there exist vertices $x_1 \in Z_1\backslash Z_2$ and $x_2 \in Z_2\backslash Z_1$ such that $x_1x_2 \in E(G)$. Since Z_1 and Z_2 are not nested, by Corollary 1 we infer that v_{Z_1} and v_{Z_2} are incomparable w.r.t. $\preceq_N$, and hence $v_{Z_1}v_{Z_2} \notin E(G)$. Observe that then $G[\{v_{Z_1}, v_{Z_2}, x_1, x_2\}]$ is an induced P_4; however, the existence of such an obstacle is not forbidden by the definition of a TP-modulator.

Create an auxiliary graph H with $V(H) = \mathcal{F}_2$, and put $Z_1Z_2 \in E(H)$ if and only if Z_1 and Z_2 satisfy the condition from the previous paragraph, i.e., Z_1 and Z_2 are not nested and there exist $x_1 \in Z_1\backslash Z_2$ and $x_2 \in Z_2\backslash Z_1$ with $x_1x_2 \in E(G)$. Run the classic greedy 2-approximation algorithm for vertex cover in H. This algorithm either finds a matching M in H of size more than $\binom{4k}{2} \cdot k$, or a vertex cover C of H of size at most $2 \cdot \binom{4k}{2} \cdot k$. In the first case, assign each edge Z_1Z_2 of M to the corresponding edge x_1x_2 of $G[X]$ as in the definition of the edges of H. Observe that since $|X| \leq 4k$, then some edge $x_1x_2 \in G[X]$ is assigned at least $k + 1$ times. Then it is easy to see that the sets $\{v_{Z_1}, v_{Z_2}, x_1, x_2\}$ for Z_1Z_2 being edges of M assigned to x_1x_2 induce P_4s that share only the edge x_1x_2, and hence Rule 2 would be applicable to x_1x_2. This is a contradiction with the assumption that (G, k) is reduced. Hence, we can assume that we have successfully constructed a vertex cover C of H of size at most $2 \cdot \binom{4k}{2} \cdot k = O(k^3)$.

Let now $\mathcal{F}_2' = \mathcal{F}_2 \setminus C$. Since $\mathcal{F}_2'$ is independent in H, it follows that for any non-nested $Z_1, Z_2 \in \mathcal{F}_2'$ and any $x_1 \in Z_1 \setminus Z_2$, $x_2 \in Z_2 \setminus Z_1$, we have that $x_1 x_2 \notin E(G)$. Since the sets of $\mathcal{F}_2'$ induce complete graphs in $G[X]$, this means that in particular there is no set $Z_3 \in \mathcal{F}_2'$ that contains both x_1 and x_2. This proves that the family $\mathcal{F}_2'$ is a weakly laminar set system with X as ground set, so by Lemma 2 we infer that $|\mathcal{F}_2'| \leq |X| + 1 \leq 4k + 1$. Concluding,

$$|\mathcal{F}_2| \leq |C| + |\mathcal{F}_2'| \leq O(k^3) + 4k + 1 = O(k^3),$$

and $|\mathcal{F}| \leq |\mathcal{F}_1| + |\mathcal{F}_2| = O(k^4) + O(k^3) = O(k^4)$. □

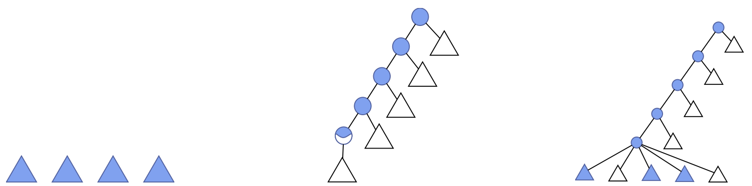
3.4 Locating Important Bags

In the previous section we analyzed the structure of neighborhoods that nodes from $V(G) \setminus X$ have in X. Our goal in this section is to perform the symmetric analysis: to understand, how the neighborhood of a fixed $x \in X$ in $V(G) \setminus X$ looks like. Eventually, we aim to locate a family I of $O(k)$ *important bags*, where some non-trivial behavior w.r.t. the neighborhoods of vertices of X happens. Then, we will perform a lowest common ancestor-closure on the set I, thus increasing its size to at most twice. After performing this step, all the connected components of $T - I$ have very simple structure from the point of view of their neighborhoods in X. As there are only $O(k)$ such components, we will be able to kernelize them separately.

The following definition and lemma explains what are the types of neighborhoods that vertices of X can have in $V(G) \setminus X$. To simplify the notation, in the following we treat $\preceq$ also as a partial order on the vertices of the forest T denoting the ancestor–descendant relation, i.e., $s \preceq t$ if and only if s is an ancestor of t (possibly $s = t$).

Definition 7 (*Type 0, 1, and 2 neighborhoods*) Let $x \in X$ be any vertex and consider $U_x = N(x) \setminus X$. We say that U_x is (see Fig. 4):

– A *neighborhood of Type 0* if U_x is the union of the vertex sets of a collection of connected components of $G - X$.



(a) Type 0: x sees a disjoint union of connected components.

(b) Type 1: x sees all the vertices in bags from a root and to a point in a bag, and nothing else.

(c) Type 2: x has as neighbors all the vertices from a root and down to a bag, and a collection of subtrees below that bag.

Fig. 4 Three types of neighborhoods; simply denoted Type 0, Type 1, and Type 2. The blue parts mark the possible neighborhoods of a vertex $x \in X$ (Color figure online)

- A *neighborhood of Type 1* if there exists a node $t_x \in V(T)$ such that $\bigcup_{s \prec t_x} B_s \subseteq U_x \subseteq \bigcup_{s \preceq t_x} B_s$. In other words, U_x consists of all the vertices contained in bags on the path from t_x to the root of its subtree in T, where some vertices of B_{t_x} itself may be excluded.
- A *neighborhood of Type 2* if there exists a node $t_x \in V(T)$ and a collection $\mathcal{L}_x$ of subtrees of T rooted at children of t_x such that $U_x = \bigcup_{s \preceq t_x} B_s \cup \bigcup_{S \in \mathcal{L}_x} \bigcup_{s \in V(S)} B_s$. In other words, U_x is formed by all the vertices contained in bags on the path from t_x to the root of its subtree in T, plus a selection of subtrees rooted in the children of t_x, where the vertices appearing in the bags of each such subtree are either all included in U_x or all excluded from U_x.

Lemma 8 *Let $x \in X$ be any vertex and consider $U_x = N(x) \setminus X$. Then U_x is of Type 0, 1 or 2.*

Proof From Corollary 1 we infer that U_x is closed downwards w.r.t. the quasi-ordering $\preceq_N$, i.e., if $v \in U_x$ and $u \preceq_N v$, then also $u \in U_x$. Let S_x be the set of nodes of T whose bags contain at least one vertex of U_x. It follows that S_x is closed under taking ancestors in forest T. Moreover if $t \in S_x$, then the bags of all the ancestors of t other than t are fully contained in U_x.

Claim 1 *Suppose $t, t' \in S_x$ are two nodes that are incomparable w.r.t. $\preceq$. Then $U_x \supseteq \bigcup_{s \succeq t} B_s$ and $U_x \supseteq \bigcup_{s \succeq t'} B_s$, i.e., U_x contains all the vertices of all the bags contained in the subtrees of T rooted at t and t'.*

Proof We prove the statement for the subtree rooted at t'; The proof for the subtree rooted at t is symmetric. Let y and y' be arbitrary vertices of $B_t \cap U_x$ and $B_{t'} \cap U_x$, respectively. For the sake of contradiction suppose there exists some $v \in \bigcup_{s \succeq t'} B_s$ such that $vx \notin E(G)$. Since $v \in \bigcup_{s \succeq t'} B_s$ and t, t' are incomparable w.r.t. $\preceq$, by the properties of the universal clique decomposition we have that $yy' \notin E(G)$, $vy \notin E(G)$ and $vy' \in E(G)$. Since $xy, xy' \in E(G)$ by the definition of U_x, we conclude that $\{y, y', x, v\}$ would induce a P_4 in G that has only one vertex in common with X (see Fig. 5), a contradiction to the definition of a TP-modulator.

We now use Claim 1 to perform a case study that recognizes U_x as a neighborhood of Type 0, 1, or 2. Suppose first that U_x contains vertices of at least two distinct connected components of $G - X$. Let C_1, C_2 be any two such components, and let T_1 and T_2 be the trees of the forest T that are UCDs of C_1 and C_2, respectively. Since S_x is closed under taking ancestors in T, it follows that the roots of T_1 and T_2 belong to S_x. Claim 1 implies then that the entire vertex sets of C_1 and C_2 are contained in U_x. Since $\{C_1, C_2\}$ was an arbitrary pair of distinct components containing a vertex of U_x, it follows that U_x must be the union of vertex sets of a selection of connected components of $G - X$, i.e., a neighborhood of Type 0.

Since $U_x = \emptyset$ is also a neighborhood of Type 0, we are left with analyzing the case when $U_x \subseteq V(C_0)$ for C_0 being a connected component of $G - X$; Let T_0 be the UCD of C_0. Observe that if U_x does not contain any pair of vertices incomparable w.r.t. $\preceq$, then S_x must form a path from some node of T_0 to the root of T_0, and hence U_x is a neighborhood of Type 1. Otherwise, there exists some node of S_x such that at least two subtrees rooted at its children contain nodes from S_x. Let t_x be such a node that is

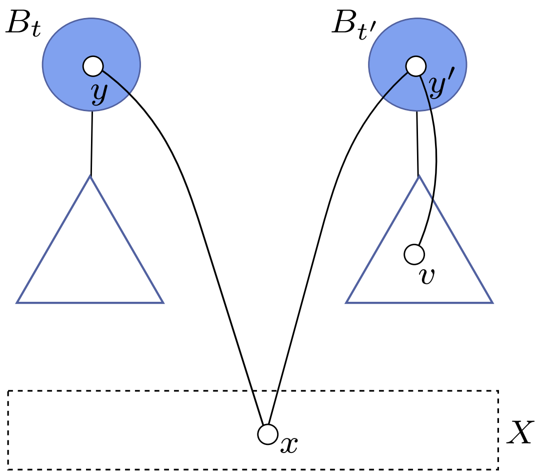


Fig. 5 An induced P_4, $yxy'v$, with only one vertex x in the modulator, appearing in the proof of Claim 1 (Color figure online)

highest in T_0, and let $\mathcal{L}_x$ be the family of subtrees rooted at children of t_x that contain nodes of S_x. Again applying Claim 1, we infer that U_x contains all the vertices of all the bags of every subtree of $\mathcal{L}_x$: for any two distinct subtrees $T_1, T_2 \in \mathcal{L}_x$, S_x contains the roots of T_1 and T_2, and hence by Claim 1 U_x contains all the vertices of all the bags of T_1 and T_2. Since t_x was chosen to be the highest, it follows that U_x is a neighborhood of Type 2 for node t_x and selection of subtrees $\mathcal{L}_x$. □

Clearly, for every $x \in X$ we can in polynomial time analyze U_x and recognize it as a neighborhood of Type 0, 1, or 2. Let I_0 be the set of nodes t_x for vertices $x \in X$ for which U_x is of Type 1 or 2. To simplify the structure of $T - I_0$, we perform the lowest common ancestor-closure operation on I_0. The following variant of this operation is taken verbatim from the work of Fomin et al. [18].

Definition 8 [18] For a rooted tree T and vertex set $M \subseteq V(T)$ the lowest common ancestor-closure (LCA-closure) is obtained by the following process. Initially, set $M' = M$. Then, as long as there are vertices x and y in M' whose least common ancestor w is not in M', add w to M'. When the process terminates, output M' as the LCA-closure of M. The following folklore lemma summarizes two basic properties of LCA-closures.

Lemma 9 [18] *Let T be a tree, $M \subseteq V(T)$ and $M' =$ LCA-closure(M). Then $|M'| \leq 2|M|$ and for every connected component C of $T - M'$, $|N(C)| \leq 2$.*

Construct now the set I by taking LCA-closure(I_0) and adding the root of every connected component of T that contains a bag of I_0 (provided it is not already included). The nodes from I will be called *important nodes*, or *important bags*. From Lemma 9 it follows that $|I| \leq 3|X| \leq 12k$, and by the construction we infer that every connected component C of $T - I$ is of one of the following three forms:

- C is not adjacent to any node of I, and is thus simply a connected component of T that does not contain any important bag.

– C is adjacent to one node a of I, and it is a subtree rooted at a child of a.
– C is adjacent to two nodes a and b of I such that a is an ancestor of b. Then C is formed by the internal nodes of the $a - b$ path in T, plus all the subtrees rooted at the other children of these internal nodes.

3.5 Twin and Module Reductions

In this section we give two new reduction rules: a twin reduction and a module reduction rule. These rules are executed exhaustively by the algorithm as Rules 3 and 4. The reason why we introduce them now is that only after understanding the structural results of Sects. 3.3 and 3.4, the motivation of these rules becomes apparent. Namely, these rules will be our main tools in reducing the sizes of parts of $G - X$ located between the important bags.

3.5.1 Twin Reduction

Rule 3 *If $T \subseteq V(G)$ is a true twin class of size $|T| > 2k + 5$, and $v \in T$ is an arbitrarily picked vertex, then remove v from the graph, i.e., proceed with the instance $(G - v, k)$.*

Lemma 10 *Applicability of Rule 3 can be recognized in polynomial time. Moreover, Rule 3 is safe, i.e., (G, k) is a yes-instance if and only if $(G - v, k)$ is a yes-instance.*

Proof In order to recognize the applicability of Rule 3 we only need to inspect every true twin classes in the graph, which clearly can be done in polynomial time. We proceed to the proof of the safeness of the rule. Let T be a true twin class of size at least $2k + 5$ and let v be the vertex the rule deleted. Since the class of trivially perfect graphs is hereditary, if (G, k) is a yes-instance, it follows that $(G - v, k)$ is a yes-instance. Suppose now that $(G - v, k)$ is a yes-instance. Let F be a set of edges with $|F| \leq k$ such that $(G - v)\triangle F$ is trivially perfect.

We now show that $G\triangle F$ is also trivially perfect, which means that F is also a solution to (G, k). For the sake of contradiction, suppose W is an obstruction in $G\triangle F$. Since $(G - v)\triangle F$ is trivially perfect, W must contain the deleted vertex v. Since F has size at most k, at most $2k$ vertices of T can be incident to an edge of F. Let v_1, v_2, v_3, and v_4 be four vertices of T that are different from v and are not incident to F. Then one of them, say v_1, is not contained in W. Since v and v_1 are true twins both in G and in $G\triangle F$, we can replace v with v_1 in W yielding a new set W' which is an obstruction in $G\triangle F$. However, since v is not a member of W', we have that W' is an obstruction in $(G - v)\triangle F$, contradicting the assumption that $(G - v)\triangle F$ was trivially perfect. □

3.5.2 Module Reduction

Recall that a module is a set of vertices M such that for every vertex v in $V(G)\setminus M$, either $M \subseteq N(v)$ or $M \cap N(v) = \emptyset$; see Definition 1. The following rule enables us to reduce large trivially perfect modules.

Rule 4 *Suppose $M \subseteq V(G)$ is a module such that $G[M]$ is trivially perfect and it contains an independent set of size at least $2k+5$. Then let us take any independent set $I \subseteq M$ of size $2k+4$, and we delete every vertex of M apart from I, i.e., proceed with the instance $(G-(M\backslash I), k)$.*

Observe that Rule 4 always deletes at least one vertex, since $|M| \geq 2k+5$ and $|I| = 2k+4$. Actually, we could define a stronger rule where we only assume that $|M| \geq 2k+5$; however, the current statement will be helpful in recognizing the applicability of Rule 4. We first prove that the rule is indeed safe.

Lemma 11 *Provided that (G, k) is a reduced instance (w.r.t. Rules 1 and 2), then Rule 4 is safe, i.e., (G, k) is a yes-instance if and only if $(G-(M\backslash I), k)$ is a yes-instance.*

Proof Let $A = M\backslash I$, and $G' = G - A$. Since G' is an induced subgraph of G, by heredity, if (G, k) is a yes-instance, then (G', k) is a yes-instance. We proceed to the proof of the other direction. Suppose then that (G', k) is a yes-instance, and let F, $|F| \leq k$, be a minimum-size editing set for G'.

Claim 2 *No vertex of I is incident to any edit of F.*

Proof Since F has minimum possible size, it is inclusion-wise minimal. We show that if $F_I \subseteq F$ is the set of edges of F incident to a vertex of I and $F' = F\backslash F_I$, then $G' \triangle F$ being trivially perfect implies $G' \triangle F'$ being trivially perfect. Since $|I| = 2k+4$, we can find at least four vertices $v_1, \ldots, v_4 \in I$ that are not incident to any edit of F. Suppose that $G' \triangle F'$ is not trivially perfect. Then there is an obstruction W in $G' \triangle F'$ containing at least one of the vertices of I incident to an edge of F. Create W' by replacing every vertex of $(W \cap I)\backslash\{v_1, \ldots, v_4\}$ by a different vertex of $\{v_1, \ldots, v_4\}$ that is not contained in W. Since vertices of I are not incident to the edits of F', they are false twins in $G' \triangle F'$, and hence W' created in this manner induces a graph isomorphic to the one induced by W. Thus, W' is an obstacle in $G' \triangle F'$. However, the vertices $v_1, \ldots, v_4$ are not incident to the edits of F and hence W' induces the same graph in $G' \triangle F'$ as in $G' \triangle F$. Therefore W' would be an obstacle in $G' \triangle F$, a contradiction to $G' \triangle F$ being trivially perfect.

Since we argued that $F' \subseteq F$ is also a solution, by the optimality of F we infer that $F = F'$ and $F_I = \emptyset$. □

We now argue that $G \triangle F$ is trivially perfect, which will imply that (G, k) is a yes-instance. For the sake of contradiction, suppose that there exists an obstacle W in $G \triangle F$; it follows that W shares at least one vertex with $M\backslash I$. From Claim 2 it follows that no edit of F is incident to any vertex of M, so in $G \triangle F$ we still have that M is a module.

If the obstruction W induces a P_4, then it is known that W is fully contained in the module M, or has at most one vertex in M [23, Observation 1]. Since $G[M] = (G \triangle F)[M]$ is trivially perfect, the latter is the case. But since M is a module in $G \triangle F$, then replacing the single vertex of $W \cap A$ with any vertex of I would yield an obstacle in $G' \triangle F$, a contradiction.

Consider then the case when W induces a C_4 in $G \triangle F$. Since $G[M] = (G \triangle F)[M]$ is C_4-free, we have that W is not entirely contained in M. Also, if W had three vertices in M, then the remaining vertex would need to be contained in $N_G(M)$, and hence would be adjacent in $G \triangle F$ to all the other three vertices of W, a contradiction to $(G \triangle F)[W]$ being a C_4. Therefore, at most two vertices of W can be in M.

Suppose exactly two vertices w_1 and w_3 of W are in M, and w_2 and w_4 are outside M. As M is a module both in G and in $G \triangle F$, we must have that $w_2, w_4 \in N_G(M)$ and hence the 4-cycle induced by W in $G \triangle F$ must be $w_1 - w_2 - w_3 - w_4 - w_1$. Take any two vertices $w_1', w_3' \in I$ and obtain W' by replacing w_1 and w_3 with them. It follows that W' induces a C_4 in $G' \triangle F$, a contradiction.

Finally, consider the case when exactly one vertex of W, say w_1, is in M. Again, replacing w_1 with any vertex of I would yield an induced C_4 contained in $G' \triangle F$, a contradiction. Thus, we conclude that $G \triangle F$ is trivially perfect. □

Observe that in order to apply Rule 4, one needs to be given the module M. Given M, finding any independent set $I \subseteq M$ of size $2k + 4$ can then be done easily as follows: We can find an independent set of maximum cardinality in M in polynomial time, since $G[M]$ is trivially perfect and the Independent Set problem is polynomial-time solvable on trivially perfect graphs (it boils down to picking one vertex from every leaf bag of the universal clique decomposition of the considered graph). Then we take any of its subsets of size $2k + 4$ to be I. Hence, to apply Rule 4 exhaustively, we need the following statement.

Lemma 12 *There exists a polynomial-time algorithm that, given an instance (G, k), either finds a module $M \subseteq V(G)$ where Rule 4 can be applied, or correctly concludes that Rule 4 is inapplicable.*

Proof Using Theorem 6 we compute the module decomposition $(T, (M^t)_{t \in V(T)})$ of G. Then we verify applicability of Rule 4 to each module M^t for $t \in V(T)$, by checking whether $G[M]$ is trivially perfect and contains an independent set of size $2k + 5$ (the latter check can be done in polynomial time since $G[M]$ is trivially perfect). Moreover, we perform the same check on all the modules N_t formed as follows: take a union node $t \in V(T)$, and construct a module N_t by taking the union of labels of those children of t that induce trivially perfect graphs.

We now argue that if Rule 4 is applicable to some module M in G, then this algorithm will encounter some (possibly different) module M' to which Rule 4 is applicable as well. By the third point of Theorem 6, either $M = M^t$ for some $t \in V(T)$, or M is the union of a collection of labels of children of some union or join node. In the first case the algorithm verifies M explicitly. In the following, let $\alpha(H)$ denote the size of a maximum independent set in a graph H.

If now M is a union of labels of some children of a union node t, then by heredity $M \subseteq N^t$. Moreover, N^t induces a trivially perfect graph (since trivially perfect graphs are closed under taking disjoint union) and clearly $\alpha(N^t) \geq \alpha(M)$. Hence, Rule 4 is applicable to $M' = N^t$, and this will be discovered by the algorithm.

Finally, suppose M is a union of labels of some children $t_1, t_2, \ldots, t_p$ of a join node t. Observe that since for every $i \neq j$, every vertex of M^{t_i} is adjacent to every vertex of M^{t_j}, it follows that $\alpha(G[M]) = \max_{i=1,2,\ldots,p} \alpha(G[M^{t_i}])$. Without loss of generality suppose that the maximum on the right hand side is attained for the module M^{t_1}. Then by heredity $G[M^{t_1}]$ is trivially perfect, and $\alpha(G[M^{t_1}]) = \alpha(G[M]) \geq 2k + 5$.

Therefore Rule 4 is applicable to $M' = M^{t_1}$, and this will be discovered by the algorithm. □

We remark here that for the kernelization algorithm it is not necessary to be sure that Rule 4 is inapplicable at all. Instead, we could perform it on demand. More precisely, during further analysis of the structure of $G - X$ we argue that some modules have to be small, since otherwise Rule 4 would be applicable. This analysis can be performed by a polynomial-time algorithm that would just apply Rule 4 on any encountered module that needs shrinking. However, we feel that the fact that Rule 4 can be indeed applied exhaustively provides a better insight into the algorithm, and streamlines the presentation.

Having introduced and verified Rules 3 and 4, we can now prove that after applying them exhaustively, all the trivially perfect modules in the graph are small.

Lemma 13 *A (possibly disconnected) trivially perfect graph with maximum true twin class size t and maximum independent set size α has at most $(2\alpha - 1)t$ vertices in total.*

Proof Let $\mathcal{T}$ be the UCD of G, a trivially perfect graph with independent set number α and every true twin class of size at most t. Since any collection comprising one vertex from each leaf bag of $\mathcal{T}$ forms an independent set, there are at most α leaf bags in $\mathcal{T}$. Thus the number of nodes of $\mathcal{T}$ in total is at most $2\alpha - 1$. Since every bag of the decomposition $T \subseteq V(G)$ is a true twin class, we conclude that there are at most $(2\alpha - 1)t$ vertices in G. □

Corollary 2 *Suppose an instance (G, k) is reduced, and moreover Rules 3 and 4 are not applicable to (G, k). Then for every module $M \subseteq V(G)$ such that $G[M]$ is trivially perfect, we have that $|M| = O(k^2)$.*

Proof Suppose M is such a module. Observe that members of every true twin class in $G[M]$ are also true twins in G (since M is a module). Hence twin classes in $G[M]$ have size at most $2k+4$, as otherwise Rule 3 would be applicable. Moreover, if $G[M]$ contained an independent set of size $2k + 5$, then Rule 4 would be applicable. By Lemma 13, we infer that $|M| \leq (4k + 7)(2k + 4) = O(k^2)$. □

From now on we assume that in the considered instance (G, k) we have exhaustively applied Rules 1–4, using the algorithms of Lemmas 3, 10, and 12. Hence Corollary 2 can be used. Observe that to perform this step, we do not need to construct the small modulator X at all. However, we hope that the reader already sees that Rules 1–4 will be useful for shrinking too large parts of $G - X$ between the important bags.

3.6 Kernelizing Non-important Parts (Irrelevant Vertex Deletion)

Recall that we have fixed a small TP-modulator X with $|X| \leq 4k$ such that $G - X$ is a trivially perfect graph with universal clique decomposition $\mathcal{T}$. Moreover, Rules 1–4 are inapplicable to (G, k). By Lemma 7 we have that the number of X-neighborhoods is $O(k^4)$. By the marking procedure, we have marked a set I of $O(k)$ bags of $\mathcal{T}$ as important, in such a manner that every connected component of $\mathcal{T} - I$ is adjacent to at most two vertices of I, and is in fact of one of the three forms described at the end of Sect. 3.4.

Thus, the whole vertex set of $G - X$ can be partitioned into four sets:

V_I: vertices contained in bags from I;
V_0: vertices contained in bags of those components of $\mathcal{T} - I$ that are not adjacent to any bag from I;
V_1: vertices contained in bags of those components of $\mathcal{T} - I$ that are adjacent to exactly one bag from I;
V_2: vertices contained in bags of those components of $\mathcal{T} - I$ that are adjacent to exactly two bags from I.

We are going to establish an upper bound on the cardinality of each of these sets separately. Upper bounds for V_I, V_0, and V_1 follow already from the introduced reduction rules, but for V_2 we shall need a new reduction rule. The upper bounds on the cardinalities of V_I and V_0 are quite straightforward.

Lemma 14 $|V_I| \leq O(k^6)$.

Proof Consider for some $a \in I$ the bag B_a. Note that B_a is a module in $G - X$. By Lemma 7 there are only $O(k^4)$ possible X-neighborhoods among vertices of $G - X$. Hence, vertices of B_a can be partitioned into $O(k^4)$ classes w.r.t. the neighborhoods in X. Each such class is a module in G that is also a clique, and hence it is a true twin class. Since the twin reduction rule (Rule 3) is not applicable, each true twin class has size at most $2k + 5$, which implies that $|B_a| \leq O(k^5)$. As $|I| = O(k)$, we conclude that $|V_I| \leq O(k^6)$. □

We remark that using a more precise analysis of the situation in one bag B_a for $a \in I$, one can see that the X-neighborhoods of elements of B_a are nested, so there is only at most $|X| + 1 \leq 4k + 1$ of them. By plugging in this argument in the proof of Lemma 14, we obtain a sharper upper bound of $O(k^3)$ instead of $O(k^6)$. However, the upper bounds on $|V_0|$ and $|V_1|$ are $O(k^6)$ and $O(k^7)$, respectively, so establishing a better bound here would have no influence on the overall asymptotic kernel size. Hence, we resorted to a simpler proof of a weaker upper bound.

Lemma 15 $|V_0| \leq O(k^6)$.

Proof Observe that V_0 is the union of bags of these connected components of $G - X$, whose universal clique decompositions (being components of $\mathcal{T}$) do not contain any important bag. By the definition of important bags, each such connected component C is a module in G, and clearly its neighborhood is entirely contained in X. Recall that by Lemma 7 there are only $O(k^4)$ possible different X-neighborhoods among vertices of $G - X$. Thus, we can group the connected components of $G[V_0]$ according to their X-neighborhoods into $O(k^4)$ groups, and the union of vertex sets in each such group forms a module in G. Since Rule 4 is not applicable, by Corollary 2 we have that each of these modules has size $O(k^2)$. Thus we infer that $|V_0| \leq O(k^6)$. □

To bound the size of V_1 we need a few more definitions. Suppose that C is a component of $\mathcal{T} - I$ that is adjacent to exactly one important bag $a \in I$. By the construction of I, we have that C is a tree rooted in a child of a. We shall say that C is *attached below* a. The union of bags of all the components of $\mathcal{T} - I$ attached below a will be called the *tassel rooted at* a. Thus, V_1 can be partitioned into $O(k)$ tassels.

Lemma 16 *For every $a \in I$, the tassel rooted at a has size at most $O(k^6)$.*

Proof Let $C_1, C_2, \ldots, C_r$ be the components of $\mathcal{T} - I$ rooted at the children of a, whose union of bags forms the tassel rooted at a. Recall that none of the C_is contains any important bag. Therefore, from Lemma 8 we infer that for any C_i and any $x \in X$, either all the vertices from the bags of C_i are adjacent to x, or none of them. Thus, the union of bags of each C_i forms a module in G: The vertices in this union have the same X-neighborhood, and moreover their neighborhoods in $G - X$ are formed by the vertices from the bags on the path from a to the root of a's connected component in $\mathcal{T}$. Similarly as in the proof of Lemma 15, by Lemma 7 there are only $O(k^4)$ possible X-neighborhoods, so we can partition the components C_i into $O(k^4)$ classes with respect to their neighborhoods in X. The union of bags in each such class forms a module in G; since Rule 4 is not applicable, by Corollary 2 we infer that its size is bounded by $O(k^2)$. Thus, the total number of vertices in all the components C_i is at most $O(k^6)$. □

As $|I| = O(k)$, Lemma 16 immediately implies the following.

Lemma 17 $|V_1| \le O(k^7)$.

We are left with bounding the cardinality of V_2. Let us fix any component C of $\mathcal{T} - I$ which is adjacent in $\mathcal{T}$ to two nodes of I. From the construction of I, it follows that C has the following form:

- C contains a path $P = a_1 - a_2 - \cdots - a_d$ such that in $\mathcal{T}$, node a_d is a child of an important node $b^{\uparrow}$, and a_1 has exactly one important child $b^{\downarrow}$.
- For every $i = 1, 2, \ldots, d$, C contains also all the subtrees of $\mathcal{T}$ rooted in children of a_i that are different from a_{i-1} (where $a_0 = b^{\downarrow}$).

Such a component C will be called a *comb* (see Fig. 6). The path P is called the *shaft* of a comb; the union of the bags of the shaft will be denoted by Q. The union of the bags of the subtrees rooted in children of a_i, apart from a_{i-1}, will be called the *tooth at* i, and denoted by R_i. Note that the subgraph induced by a tooth is not necessarily connected; it is, however, always non-empty by the definition of the universal clique decomposition. We also denote $R = \bigcup_{i=1}^{d} R_i$. By somehow abusing the notation, we will also denote $B_i = B_{a_i}$ for $i = 1, 2, \ldots, d$. The number of teeth d is called the *length* of a comb.

Since the comb C does not contain any important vertices, from Lemma 8 and the construction of I we immediately infer the following observation about the X-neighborhoods of vertices of the shaft and the teeth.

Lemma 18 *There exist two sets Y, Z with $Z \subseteq Y \subseteq X$ such that $N_X(u) = Y$ for every $u \in Q$ and $N_X(v) = Z$ for every $v \in R$.*

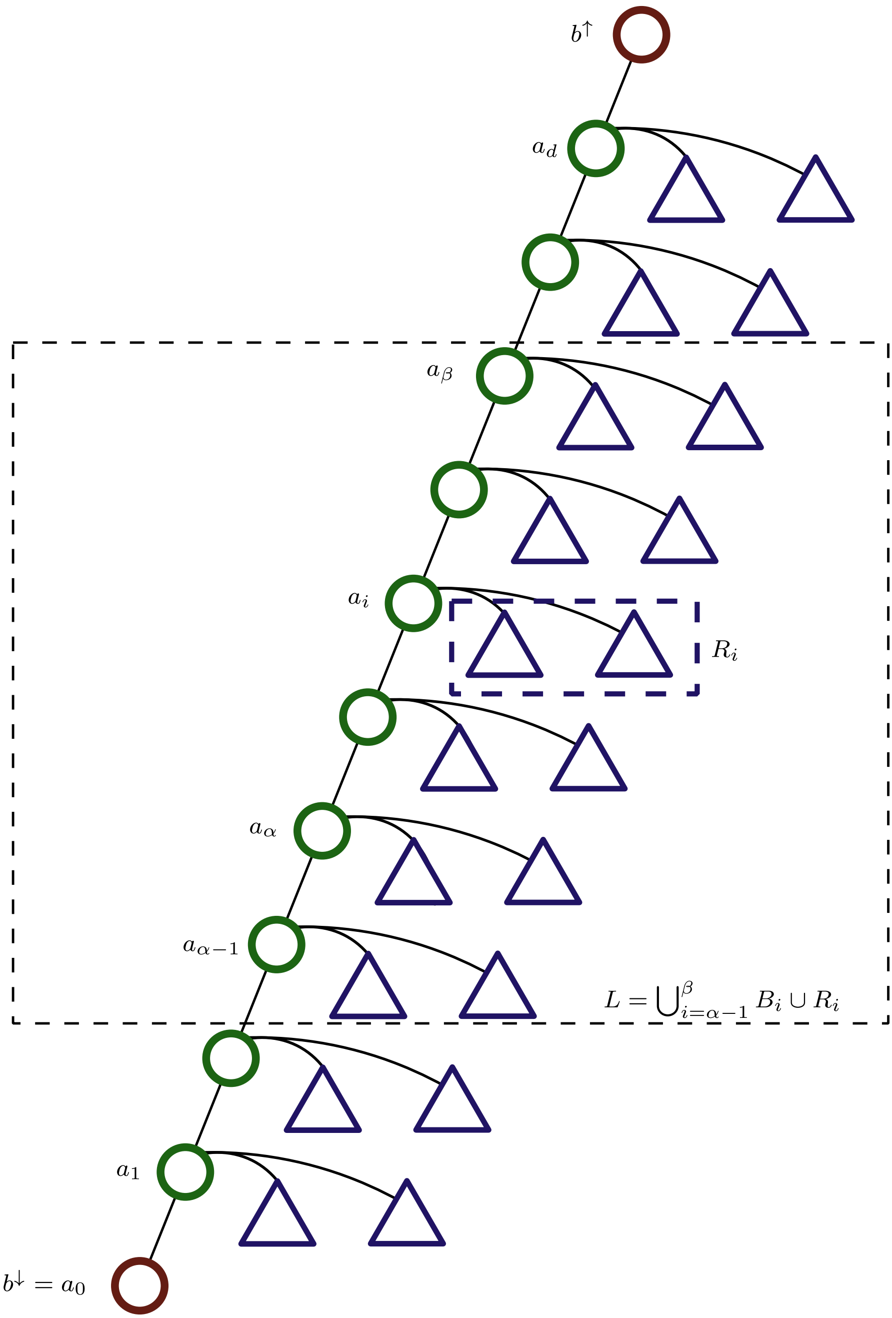


Fig. 6 The anatomy of a *comb*. The top and bottom bags, $b^{\uparrow}$ and $b^{\downarrow}$, are important bags

In particular, Lemma 18 implies that every tooth of a comb is a module. Hence, since Rule 4 is not applicable, we infer that $|R_i| = O(k^2)$ for $i = 1, 2, \ldots, d$. Also, observe that each B_i is a twin class, so by inapplicability of Rule 3 we conclude that $|B_i| \leq 2k + 5$ for each $i = 1, 2, \dots, d$.

Since $\mathcal{T}$ is a forest and $|I| = O(k)$, it follows that in $\mathcal{T} - I$ there are $O(k)$ combs. As we already observed, for each comb the sizes of individual teeth and bags on the shaft are bounded polynomially in k. Hence, the only thing that remains is to show how to reduce combs that are long. In order to do this, we need one more definition: a tooth R_i is called *simple* if $G[R_i]$ is edgeless, and it is called *complicated* otherwise. We can now state the final reduction rule.

Rule 5 *Suppose C is a comb of length at least $(4k + 3)^2$, and adopt the introduced notation for the shaft and the teeth of C. Define an index β as follows:*

(i) If at least $4k + 3$ teeth R_i are complicated, then we let $\beta = d$.
(ii) Otherwise, there is a sequence of $4k + 3$ consecutive teeth $R_i, \dots, R_{i+4k+2}$ that are simple. Let β be the index of the last tooth of this sequence, i.e., $\beta = i+4k+2$.

Having defined β, remove the tooth R_β from the graph and do not modify the budget. That is, proceed with the instance $(G - R_\beta, k)$.

Lemma 19 *Rule 5 is safe.*

Proof Since $G - R_\beta$ is an induced subgraph of G, then we trivially have that the existence of a solution for (G, k) implies the existence of a solution for $(G - R_\beta, k)$. Hence, we now prove the converse. Suppose that F is a solution to $(G - R_\beta, k)$, that is, a set of edits in $G - R_\beta$ such that $(G - R_\beta)\triangle F$ is trivially perfect and $|F| \leq k$.

We will say that a tooth R_i is *spoiled* if any vertex of $R_i \cup B_i$ is incident to an edit from F, and *clean* otherwise. The first goal is to find an index α such that

(a) $1 < \alpha < \beta$,
(b) the teeth $R_{\alpha-1}$ and R_α are clean, and
(c) if any of the teeth $R_{\alpha+1}, R_{\alpha+2}, \dots, R_\beta$ is complicated, then R_α is also complicated.

Suppose first that β was constructed according to case (i), i.e., there are at least $4k+3$ complicated teeth in the comb, and hence $\beta = d$. Out of these teeth R_i, at most one can have index 1, at most one can have index d, at most $2k$ can be spoiled (since $|F| \leq k$) and at most $2k$ can have the preceding tooth R_{i-1} spoiled. This leaves at least one complicated tooth R_i such that $1 < i < d$ and both R_i and R_{i-1} are clean. Then we can take $\alpha = i$; thus, property (c) of α is satisfied since R_α is complicated.

Suppose then that β was constructed according to case (ii), i.e., the following teeth are all simple: $R_{\beta-(4k+2)}, R_{\beta-(4k+1)}, \dots, R_{\beta-1}, R_\beta$. Similarly as before, out of these $4k+3$ teeth, one has index β, one has index $\beta-(4k+2)$, at most $2k$ can be spoiled, and at most $2k$ can have the preceding tooth spoiled. Hence, among them there is a tooth R_i such that $\beta - (4k+2) < i < \beta$ and both R_i and R_{i-1} are clean. Again, we take $\alpha = i$; thus, property (c) is satisfied since all the teeth $R_{\beta-(4k+2)}, R_{\beta-(4k+1)}, \dots R_{\beta-1}, R_\beta$ are simple.

With α defined, we are ready to complete the proof of Lemma 19. To that aim, define $L = \bigcup_{i=\alpha-1}^{\beta} B_i \cup R_i$. Construct F' from F by removing all the edits that are incident to any vertex of L; clearly $|F'| \leq |F| \leq k$. We claim that F' is a solution to the instance (G, k), that is, that $G \triangle F'$ is trivially perfect. For the sake of a contradiction, suppose that $A \subseteq V(G)$ is a vertex set of size 4 such that $G \triangle F'[A]$ is a P_4 or a C_4. Let $A_0 = A \cap L$ and $A_1 = A \setminus A_0$.

Claim 3 $|A_0| = 1$ *or* $|A_0| = 2$.

Proof Suppose first that $A_0 = \emptyset$, so $A \subseteq V(G) \setminus L \subseteq V(G - R_\beta)$. Since $F \cap \binom{V(G)\setminus L}{2} = F' \cap \binom{V(G)\setminus L}{2}$ and $R_\beta \subseteq L$, we have that the induced subgraph $G \triangle F'[A]$ is equal to the induced subgraph $(G - R_\beta) \triangle F[A]$. However, the graph $(G - R_\beta) \triangle F$ is trivially perfect, so it cannot have an induced P_4 or C_4; a contradiction.

Suppose now that $|A_0| \geq 3$. Since $A_0 \subseteq L$ and no edit of F' is incident to any vertex of L, we infer that there is no edit of F' between vertices of A: only at most one vertex of A does not belong to A_0. Therefore $G[A] = G \triangle F'[A]$ and $G[A]$ is an induced C_4 or P_4 in the graph G. However, $A_0 \subseteq L \subseteq V(G) \setminus X$, so $|A \cap X| \leq 1$. Thus, $G[A]$ would be an obstacle in G that has at most one common vertex with TP-modulator X, a contradiction with the definition of a TP-modulator (Definition 5). □

To obtain a contradiction, we shall construct a set A'_0 satisfying the following properties:

(i) $A'_0 \subseteq R_{\alpha-1} \cup B_{\alpha-1} \cup R_\alpha \cup B_\alpha$;
(ii) $|A'_0| = |A_0|$ and $G[A'_0]$ is edgeless if and only if $G[A_0]$ is edgeless;
(iii) $|A_0 \cap Q| = |A'_0 \cap Q|$ and hence $|A_0 \cap R| = |A'_0 \cap R|$.

Let us define $A' = A_1 \cup A'_0$. For now we postpone the exact construction.

Claim 4 *If* A'_0 *satisfies properties (i), (ii), and (iii), then* $G \triangle F'[A]$ *is isomorphic to* $G \triangle F'[A']$.

Proof By property (iii) there exists a bijection η between A_0 and A'_0 that preserves belonging to Q or R between the argument and the image. Extend η to A by defining $\eta(u) = u$ for $u \in A_1$; we claim that η is an isomorphism between $G \triangle F'[A]$ and $G \triangle F'[A']$. To see this, observe that since $A_0, A'_0 \subseteq L$, then we have that no vertex of A_0 or A'_0 is incident to any edit of F'. Moreover, in G, all the vertices of $L \cap R$ have the same neighborhood in $V(G) \setminus L$, and the same holds also for the vertices of $L \cap Q$. As the neighborhoods of these vertices in G and in $G \triangle F'$ are exactly the same, we infer that each vertex $u \in A_0$ is adjacent in $G \triangle F'$ to the same vertices of A_1 as the vertex $\eta(u)$ is.

To conclude the proof, we need to prove that η restricted to A'_0 is also an isomorphism between $G \triangle F'[A_0]$ and $G \triangle F'[A'_0]$. Again, A_0 and A'_0 are not incident to any edit of F', so $G \triangle F'[A_0] = G[A_0]$ and $G \triangle F'[A'_0] = G[A'_0]$. By Claim 3 we have that $|A_0| = 1$ or $|A_0| = 2$, and we conclude by observing that a pair of simple graphs with at most two vertices are isomorphic if and only if both of them are edgeless or both of them contain an edge, and in both cases any bijection between the vertex sets is an isomorphism. □

We now argue that the existence of a set A'_0 satisfying properties (i), (ii), and (iii) leads to a contradiction. Recall that the teeth $R_{\alpha-1}$ and R_α are clean, which means that no vertex of $R_{\alpha-1} \cup B_{\alpha-1} \cup R_\alpha \cup B_\alpha$ is incident to any edit from F. Moreover, as $\beta > \alpha$, we have that $A' \subseteq V(G - R_\beta)$. By the construction of F' and A' we infer that $G \triangle F'[A'] = (G - R_\beta) \triangle F[A']$. By Claim 4 we have that $G \triangle F'[A']$ is a P_4 or a C_4, since $G \triangle F'[A]$ was. This would, however, mean that $(G - R_\beta) \triangle F$ would contain an induced P_4 or an induced C_4, a contradiction to the assumption that $(G - R_\beta) \triangle F$ is trivially perfect.

Therefore, we are left with constructing a set A'_0 satisfying properties (i), (ii), and (iii). We give different constructions depending on the alignment of the vertices of A_0. In each case we just define A'_0; verifying properties (i), (ii), and (iii) in each case is trivial.

Case 1. $|A_0| = 1$.
- Case 1a. $A_0 = \{u\}$ and $u \in Q$. Then $A'_0 = \{u'\}$ for any $u' \in B_{\alpha-1}$.
- Case 1b. $A_0 = \{u\}$ and $u \in R$. Then $A'_0 = \{u'\}$ for any $u' \in R_{\alpha-1}$.

Case 2. $|A_0| = 2$.
- Case 2a. $A_0 = \{u, v\}, u, v \in Q$. As $G[Q]$ is a clique, it follows that $uv \in E(G)$. Then $A'_0 = \{u', v'\}$ for any $u' \in B_{\alpha-1}$ and $v' \in B_\alpha$.
- Case 2b. $A_0 = \{u, v\}, u \in Q, v \in R$, and $uv \notin E(G)$. Then $A'_0 = \{u', v'\}$ for any $u' \in B_{\alpha-1}$ and $v' \in R_\alpha$.
- Case 2c. $A_0 = \{u, v\}, u \in Q, v \in R$, and $uv \in E(G)$. Then $A'_0 = \{u', v'\}$ for any $u' \in B_\alpha$ and $v' \in R_{\alpha-1}$.
- Case 2d. $A_0 = \{u, v\}, u, v \in R$, and $uv \notin E(G)$. Then $A'_0 = \{u', v'\}$ for any $u' \in R_\alpha$ and $v' \in R_{\alpha-1}$.
- Case 2e. $A_0 = \{u, v\}, u, v \in R$, and $uv \in E(G)$. As there are no edges in G between different teeth, we observe that $u, v \in R_i$ for some i such that $R_i \subseteq L$, i.e., $\alpha - 1 \leq i \leq \beta$. In particular, the tooth R_i must be complicated. If $i = \alpha - 1$ or $i = \alpha$, then we can take $A'_0 = A_0$. Otherwise we have that $\alpha < i \leq \beta$ and R_i is complicated, so by property (c) of β we infer that R_α is also complicated. Then we take $A'_0 = \{u', v'\}$ for any $u', v' \in R_\alpha$ such that $u'v' \in E(G)$.

This case study is exhaustive due to Claim 3. □

We can finally gather all the pieces and prove our main theorem.

Theorem 7 *The problem* TRIVIALLY PERFECT EDITING *admits a proper kernel with* $O(k^7)$ *vertices.*

Proof The algorithm first applies Reduction Rules 1—4 exhaustively. As each application of a reduction rule either decreases n and does not change k, or decreases k while not changing n, the number of applications of these rules will be bounded by $O(n + k)$ until k becomes negative and we can conclude that we are working with a no-instance. By Lemmas 3, 10, 11, and 12, these rules are safe, applicability of each rule can be recognized in polynomial time, and applying the rules also takes polynomial time.

After all the rules, Rules 1–4, have been applied exhaustively, we construct a small TP-modulator X using the algorithm of Lemma 4. In case the construction fails, we conclude that we are working with a no-instance. Otherwise, in polynomial time we construct the universal clique decomposition $\mathcal{T}$ of $G - X$, and then we mark the set I of important bags. Both locating the important bags and performing the lowest common ancestor closure can be done in polynomial time. After this, we examine all the combs of $\mathcal{T} - I$. In case there is a comb of length greater than $(4k+3)^2$, we apply Rule 5 on it and restart the whole algorithm. Observe that each application of this rule reduces the vertex count by one while keeping k, so the total number of times the algorithm is restarted is bounded by the vertex count of the original instance.

We are left with analyzing the situation when Reduction Rule 5 is not applicable, i.e., all the combs have length less than $(4k+3)^2$. As we have argued, the inapplicability of Rules 3 and 4 ensures that bags of shafts of combs have sizes $O(k)$ and teeth of combs have sizes $O(k^2)$. Hence, every comb has $O(k^4)$ vertices. Since the number of combs is $O(k)$, we infer that $|V_2| \leq O(k^5)$. Together with the upper bounds on the sizes of V_I, V_0, and V_1 given by Lemmas 14, 15, and 17, we conclude that

$$\begin{aligned} |V(G)| &= |X| + |V_I| + |V_0| + |V_1| + |V_2| \\ &\leq 4k + O(k^6) + O(k^6) + O(k^7) + O(k^5) = O(k^7). \end{aligned}$$

Hence, we can output the current instance as the obtained kernel. □

4 Kernels for Trivially Perfect Completion/Deletion

We now present how the technique applied to Trivially Perfect Editing also yields polynomial kernels for Trivially Perfect Completion and Trivially Perfect Deletion after minor modifications. That is, we prove Theorems 2 and 3.

We show that all the rules given above, with only two minor modifications are correct for both problems. Clearly, the running times of the algorithms recognizing applicability of the rule do not depend on the problem we are solving, so we only need to argue for their safeness.

In the first two rules, Rules 1 and 2, we add and delete an edge, respectively, and the argument is that any editing set of size at most k must necessarily include this edit. However, in the completion and deletion version, we are not allowed both operations. Hence, for the first rule, in the deletion variant we can immediately infer that we are working with a no-instance, and respectively for the second rule in the completion variant.

Thus, the two following rules replace Rule 1 for deletion and Rule 2 for completion, and their safeness is guaranteed by a trivial modification of the proof of Lemma 3:

Rule 6 *For an instance (G, k) with $uv \notin E(G)$, if there is a matching of size at least $k+1$ in $\overline{G[N(u) \cap N(v)]}$, then return a trivial no-instance as the computed kernel.*

Rule 7 *For an instance (G, k) with $uv \in E(G)$ and $N_1 = N(u) \setminus N[v]$ and $N_2 = N(v) \setminus N[u]$, if there is a matching in $\overline{G}$ between N_1 and N_2 of size at least $k+1$, then return a trivial no-instance as the computed kernel.*

Observe that Rules 6 and 7 are applicable in exactly the same instances as their unmodified variants. Hence, exhaustive application of the basic rules with any of these modifications results in exactly the same notion of a reduced instance as the one introduced in Sect. 3.1. We now argue that Rules 3 and 4 are safe for both the deletion and the completion variant, without any modifications.

Lemma 20 *Rules 3 and 4 are safe both for* Trivially Perfect Deletion *and for* Trivially Perfect Completion.

Proof The proof of the safeness of Rule 3 (Lemma 10) in fact argues that every editing set F for $(G - v, k)$ with $|F| \leq k$ is also an editing set for (G, k). This holds also for editing sets that consist only of edge additions/deletions, so the reasoning remains the same for Trivially Perfect Deletion and Trivially Perfect Completion.

The proof of the safeness of Rule 4 (Lemma 11) first argues that any minimum-size editing set F for the reduced instance (G', k) is not incident to any vertex of I. This is done by showing that otherwise F would not be an inclusion-wise minimal editing set (proof of Claim 2), and the argumentation can be in the same manner applied to minimum-size completion/deletion sets. Then it is argued that F is in fact an editing set for the original instance (G, k), and the argumentation is oblivious to whether F is allowed to contain edge additions or deletions. □

We now proceed to the analysis of Rule 5 in the completion and deletion variants. First, let us consider the construction of the modulator. In the completion/deletion variants we can construct the modulator in exactly the same manner as for editing. Indeed, the main argument for the bound $|X| \leq 4k$ states that if the construction was performed for more than k rounds, then we are dealing with a no-instance, since then any editing set for G has size at least $k + 1$. Completion and deletion sets are editing sets in particular, so the same argument holds also for Trivially Perfect Deletion and Trivially Perfect Completion.

Results of Sects. 3.3 and 3.4, i.e., the analysis of the X-neighborhoods and marking of the important bags, work in exactly the same manner, since they are based on the same notions of a reduced instance and of a TP-modulator. Thus, Lemma 7 holds as well, and we have marked the same set I of $O(k)$ important bags, with the same properties. Rules 3 and 4 are not modified, so the bounds on $|V_I|$, $|V_0|$ and $|V_1|$ from Lemmas 14, 15, and 17 also hold.

We are left with analyzing Rule 5, and we claim that this rule is also safe for Trivially Perfect Deletion and Trivially Perfect Completion without any modifications. Indeed, in the proof of the safeness of the rule (Lemma 19), we have argued that for every editing set F ($|F| \leq k$) for the new instance (G', k), there exists some $F' \subseteq F$ which is a solution to the original instance (G, k). In case F consists of edge deletions or edge additions only, so does F'. Hence, (G', k) being a yes-instance of Trivially Perfect Deletion, resp. Trivially Perfect Completion, implies that (G, k) is also a yes-instance of the same problem. Thus Rule 5 is safe without any modifications, and the kernel size analysis contained in the proof of Theorem 7 (end of Sect. 3.6) can be performed in exactly the same manner. This concludes the proof of Theorems 2 and 3.

5 Obstructions for Modifying to Trivially Perfect Graphs

In this section we prove Theorem 4, which establishes a polynomial upper bound on the sizes of minimal obstructions for k-editing, k-completion, and k-deletion to a trivially perfect graph. Recall that a graph G is a minimal obstruction for k-editing to a trivially perfect graph if it does not admit an editing set (to a trivially perfect graph) of size at most k, but every its proper induced subgraph has such an editing set. Minimal obstructions for k-completion and k-deletion are defined analogously. We first prove the theorem for minimal obstructions for k-editing, the proofs for completions and deletions will be analogous.

Let $p(k) \in O(k^7)$ be the polynomial upper bound on the kernel size for TRIVIALLY PERFECT EDITING; we can assume that $p(k)$ is a non-decreasing function. Let G be a minimal obstruction for k-editing to a trivially perfect graph. That is, (G, k) is a no-instance of TRIVIALLY PERFECT EDITING, but (G', k) is a yes-instance of TRIVIALLY PERFECT EDITING whenever G' is a proper induced subgraph of G. Suppose, for the sake of contradiction, that $|V(G)| > 2 \cdot p(k)$.

First, let us exhaustively apply the basic reduction rules (Rules 1 and 2) to the instance (G, k), yielding a new instance (H, ℓ). We first prove that the number of applications is bounded by k.

Claim 5 *The basic reduction rules can not be applied more than k times to the instance* (G, k).

Proof For the sake of contradiction suppose that the basic reduction rules can be applied $k+1$ times, thus resulting in an instance with parameter -1. Fix some sequence of such applications of length $k+1$. Each application is triggered by a structure formed by 2 "central" vertices and at most $2k+2$ "petal" vertices. Let X be the set of vertices involved in any of these structures, for the considered sequence of $k+1$ applications. Then $|X| \leq (2k+4) \cdot (k+1) < p(k)$, which means that X is not equal to the whole vertex set of G, so $G[X]$ is a proper induced subgraph of G. However, the same sequence of $k+1$ basic reduction rules could be applied to the instance $(G[X], k)$; as each of the applications decrements the parameter by 1, this proves that $(G[X], k)$ is also a no-instance of TRIVIALLY PERFECT EDITING. This contradicts the assumption that G is a minimal obstacle for k-editing. □

Therefore, the exhaustive application of basic reduction rules yields an instance (H, ℓ), for some $0 \leq \ell \leq k$. Each application either adds or removes one edge from the graph, hence H has the same vertex set as G, and differs from G by an editing set of size at most $k - \ell$. Similarly as in the proof of Claim 5, let X be the set of all vertices of G that were involved in any of the structures on which the basic reduction rules were applied. Then $|X| \leq (k-\ell) \cdot (2k+4) < p(k)$. By definition, the graph H is reduced w.r.t. Rules 1 and 2.

Let us now apply the remaining reduction rules (Rules 3–5) to the instance (H, ℓ) exhaustively. Observe that these rules only remove some vertices and preserve the parameter intact, so in particular they cannot make the Rules 1 and 2 applicable again. Consequently, the exhaustive application of these rules yields an induced subgraph H' of H with the following properties:

- $|V(H')| \leq p(\ell) \leq p(k)$; and
- H' has an editing set of size ℓ if and only if H has an editing set of size at most ℓ.

However, we assumed that (G, k) was a no-instance, hence (H, ℓ) is also a no-instance, and therefore H' does not have an editing set of size at most ℓ. Since H' is an induced subgraph of H, we have $H' = H[Y]$ where $Y = V(H')$.

Now, consider the graph $G' = G[X \cup Y]$. We have $|X| < p(k)$ and $|Y| \leq p(k)$, hence $|X \cup Y| < 2 \cdot p(k) < |V(G)|$. Consequently, G' is a proper induced subgraph of G. The following claim will give us the sought contradiction with the minimality of G, thereby proving that the number of vertices in G in fact has to be at most $2 \cdot p(k)$.

Claim 6 *The graph G' has no editing set of size at most k, i.e., (G', k) is a no-instance of* Trivially Perfect Editing.

Proof Since $X \subseteq V(G')$, we can apply the same sequence of basic reduction rules to (G', k) as was applied to (G, k). This results in obtaining the instance $(H[X \cup Y], \ell)$ that is a yes-instance if and only if (G', k) is a yes-instance. However, we know that $(H', \ell) = (H[Y], \ell)$ is a no-instance, and hence so is $(H[X \cup Y], \ell)$. Consequently, (G', k) is a no-instance of Trivially Perfect Editing. ◻

Thus, we have completed the proof of Theorem 4 for minimal obstructions for k-editing to a trivially perfect graph. The cases of minimal obstructions for k-completion and k-deletion follow by essentially the same reasoning. The only difference is that in these cases, one of the basic reduction rules may conclude that we are working with a no-instance, instead of applying a reduction. However, it suffices to note that this cannot happen when basic reduction rules are exhaustively applied to (G, k), due to the same argument as in the proof of Claim 5. Namely, in such case, if we denote by X the set of all vertices of G involved in structures on which the basic reduction rules are applied (up to the termination of the kernelization procedure), then the same rules would apply when starting from $G[X]$ instead, which proves that $(G[X], k)$ is also a no-instance. However $|X| < p(k) < |V(G)|$, which contradicts the minimality of G.

6 Hardness Results

In this section we show that Trivially Perfect Editing is NP-hard, and furthermore not solvable in subexponential parameterized time unless the Exponential Time Hypothesis fails. Recall that the NP-hardness of the problem was already established by Nastos and Gao [33]. Their reduction (see the proof of Theorem 3.3 in [33]) starts with an instance of Exact 3-Cover with universe of size n and set family of size m, and constructs an instance (G, k) of Trivially Perfect Editing with $k = \Theta(mn^2)$. Thus, the parameter blow-up is at least cubic, and the reduction cannot be used to establish the non-existence of a subexponential parameterized algorithm under ETH.

Here, we give a direct, linear reduction from 3Sat to Trivially Perfect Editing. Furthermore, the resulting graph in our reduction has maximum degree equal to 4. Thus, we in fact prove that even on input graphs of maximum degree 4, Trivially Perfect Editing remains NP-hard and does not admit a subexponential parameterized algorithm, unless ETH fails. Formally, the following theorem will be proved, where for an input formula φ of 3Sat, by $\mathcal{V}(\varphi)$ and $\mathcal{C}(\varphi)$ we denote the variable and clause sets of φ, respectively:

Theorem 8 *There exists a polynomial-time reduction that, given an instance φ of* 3Sat, *returns an equivalent instance (G_φ, k_φ) of* Trivially Perfect Editing,

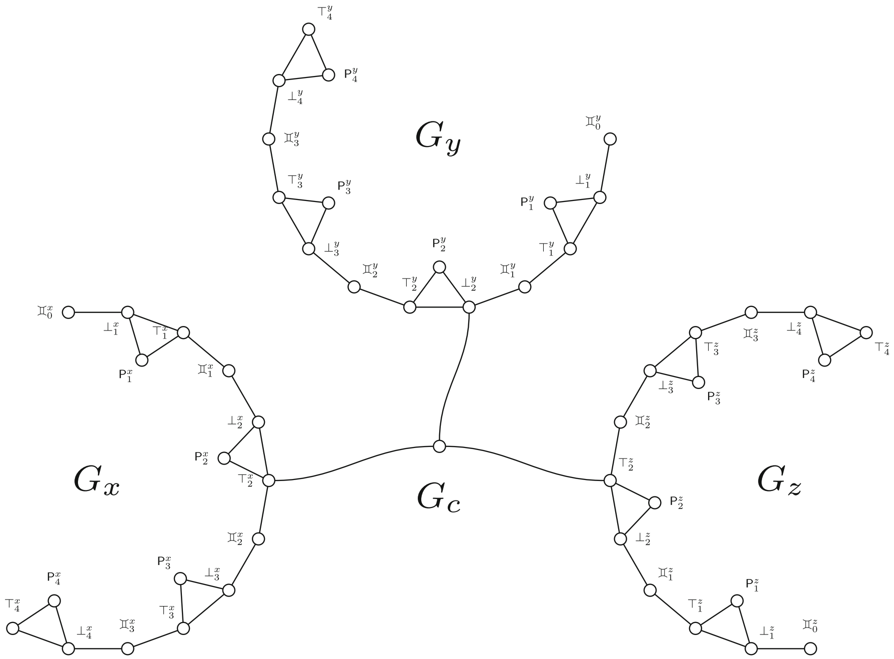


Fig. 7 Gadget $c = x \vee \neg y \vee z$. The clause c is now the second clause all variables x, y, and z appear in, and x and z appears positively whereas y appears negatively

where $|V(G_\varphi)| = 13|\mathcal{C}(\varphi)|, |E(G_\varphi)| = 18|\mathcal{C}(\varphi)|, k_\varphi = 5|\mathcal{C}(\varphi)|$*, and* $\Delta(G_\varphi) = 4$*. Consequently, even on instances with maximum degree* 4*,* Trivially Perfect Editing *remains* **NP***-hard and cannot be solved in time* $2^{o(k)}n^{O(1)}$ *or* $2^{o(n+m)}$*, unless ETH fails.*

Theorem 8 clearly refines Theorem 5, and its conclusion follows from the reduction by an application of Proposition 1. Hence, we are left with constructing the reduction, to which the rest of this section is devoted. Our approach is similar to the technique used by Komusiewicz and Uhlmann to show the hardness of a similar problem, Cluster Editing [27]; However, the gadgets are heavily modified to work for the Trivially Perfect Editing problem.

Let φ be the input instance of 3Sat. By standard modifications of the formula we may assume that every clause contains exactly three literals, all containing different variables, and that every variable appears in at least two clauses. For a variable $x \in \mathcal{V}(\varphi)$, let $p_x > 1$ be the number of occurrences of x in the clauses of φ; Moreover, we order these occurrences arbitrarily. Observe that $\sum_{x \in \mathcal{V}(\varphi)} p_x = 3|\mathcal{C}(\varphi)|$. Now, for every $x \in \mathcal{V}(\varphi)$ we create a *variable gadget*, and for every $c \in \mathcal{C}(\varphi)$ we create a *clause gadget*.

Variable gadgets For $x \in \mathcal{V}(\varphi)$, construct a graph G_x isomorphic to C_{3p_x}, a cycle on $3p_x$ vertices. The vertices of G_x are labeled $\bot_i^x, \top_i^x, \amalg_i^x$ for $i \in [0, p_x - 1]$, in the order of their appearance on the cycle. We then add a vertex P_i^x adjacent to $\top_i^x$ and $\bot_i^x$, for each $i \in [0, p_x - 1]$, see Fig. 7. Formally, the vertices P_i^x do not belong to G_x,

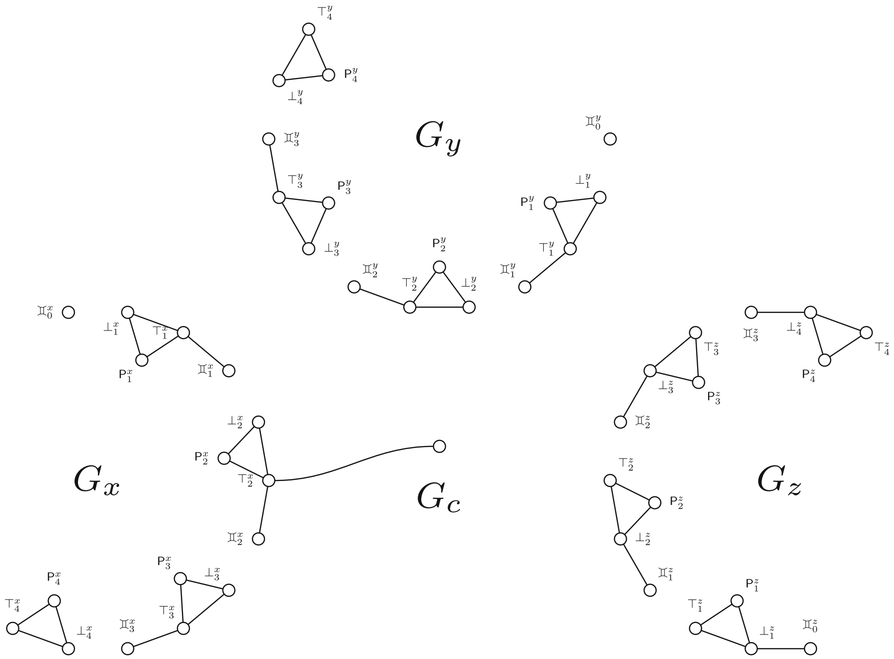


Fig. 8 Edited gadget of $c = x \vee \neg y \vee z$ where $\alpha(x) = \top$, $\alpha(y) = \top$ and $\alpha(z) = \bot$ and x has been chosen (no choice) to satisfy c. Notice the formation of *paws*, except the one incident to c which induces a *cricket*

but they will be used to wire variable gadgets with clause gadgets. This concludes the construction of the variable gadget, and it should be clear that the number of created vertices and edges is bounded linearly in p_x; More precisely, we created $4p_x$ vertices and $5p_x$ edges.

For the sake of later argumentation, we now define the deletion set F_x^α for G_x. If, in an assignment of variables $\alpha : \mathcal{V}(\varphi) \to \{\top, \bot\}$, we have $\alpha(x) = \top$, then we let F_x^α be the set consisting of every edge of the form $\top\!\!\!\!\bot_i^x \bot_{i+1 \bmod p_x}^x$ for $i \in [0, p_x - 1]$. If, on the other hand, $\alpha(x) = \bot$, we define the deletion set F_x^α to be the set comprising the edges $\top_i^x \top\!\!\!\!\bot_i^x$ for $i \in [0, p_x - 1]$, see Fig. 8. We will later show that these are the only relevant editing sets of size at most p_x for G_x.

Clause gadget The clause gadgets are very simple. A clause gadget consists simply of one vertex, i.e., for a clause $c \in \mathcal{C}(\varphi)$ construct the vertex v_c. This vertex will be connected to G_x, G_y and G_z, for x, y, and z being the variables appearing in c, in appropriate places, depending on whether the variable occurs positively or negatively in c. More precisely, if c is the ith clause x appears in, then we make v_c adjacent to $\top_i^x$ provided that x appears positively in c, and to $\bot_i^x$ provided that x appears negatively in c. This concludes the construction of a clause gadget. As every clause gadget contains one vertex and three edges, the construction of all the clause gadgets creates $|\mathcal{C}(\varphi)|$ vertices and $3|\mathcal{C}(\varphi)|$ edges.

The deletion set for a clause gadget will be as follows. Let $\alpha : \mathcal{V}(\varphi) \to \{\top, \bot\}$, be an assignment of the variables that satisfies all the clauses. Suppose $c = \ell_x \vee \ell_y \vee \ell_z$, where the literals ℓ_x, ℓ_y, and ℓ_z contain variables x, y, and z, respectively. Pick any literal satisfying c, say ℓ_x, and delete the two other edges in the connection, i.e., the two edges connecting v_c with vertices of G_y and G_z. Thus v_c remains a vertex of degree 1, adjacent to a vertex of G_x.

Let G_φ be the constructed graph. We set the budget for edits to

$$k_\varphi = \sum_{x \in \mathcal{V}(\varphi)} p_x + 2|\mathcal{C}(\varphi)| = 5|\mathcal{C}(\varphi)|.$$

Observe also that

$$|V(G_\varphi)| = \sum_{x \in \mathcal{V}(\varphi)} 4p_x + |\mathcal{C}(\varphi)| = 13|\mathcal{C}(\varphi)|,$$
$$|E(G_\varphi)| = \sum_{x \in \mathcal{V}(\varphi)} 5p_x + 3|\mathcal{C}(\varphi)| = 18|\mathcal{C}(\varphi)|,$$

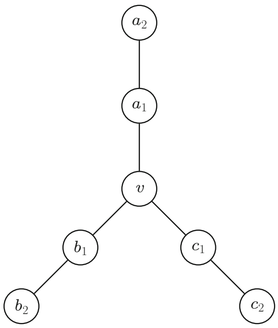
and that $\Delta(G_\varphi) = 4$. Thus, all the technical properties stated in Theorem 8 are satisfied, and we are left with proving that (G_φ, k_φ) is a yes-instance of Trivially Perfect Editing if and only if φ is satisfiable.

Before we state the main lemma, we give two auxiliary observations that settle the tightness of the budget:

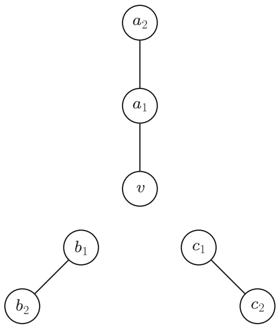
Claim 7 *Suppose that a graph H is a cycle on $3p$ vertices for some $p > 1$, and suppose F is an editing set for H. Then $|F| \geq p$. Moreover, if $|F| = p$ then F consists of deletions of every third edge of the cycle.*

Claim 8 *Suppose a graph H is a subdivided claw, i.e., the star $K_{1,3}$ with every leg subdivided once (see Fig. 9). Furthermore, suppose that F is an editing set for H. Then $|F| \geq 2$. Moreover, if $|F| = 2$ then F consists of deletions of two edges incident to the center of the subdivided claw (see Fig. 9).*

We will prove the two claims in order now. The astute reader should already see that this implies the tightness of the budget: every editing set needs to include exactly p_x edges of every variable gadget G_x (by Claim 7), and exactly two edges incident to every vertex v_c (by Claim 8). The additional vertices P_i^x will form the degree-1 vertices of subdivided claws created by clause gadgets, and all the subgraphs in question pairwise



(a) A subdivided claw.



(b) An optimally edited subdivided claw.

Fig. 9 A subdivided claw and its optimum editing set share at most single vertices, which means that any edit can influence at most one of them. This statement is made formal in the proof of Lemma 21.

Proof of Claim 7 Let $v_0, v_1, \ldots, v_{3p-1}$ be the vertices of H, in their order of appearance on the cycle. For $i = 0, 1, \ldots, p-1$, let $A_i = \{v_{3i}, v_{3i+1}, v_{3i+2}, v_{3i+3}\}$; Here and in the sequel, the indices behave cyclically in a natural manner. Observe that each A_i induces a P_4 in H, hence $F \cap \binom{A_i}{2} \neq \emptyset$. However, the sets $\binom{A_i}{2}$ are pairwise disjoint for $i = 0, 1, \ldots, p-1$, from which it follows that $|F| \geq p$.

Suppose now that $|F| = p$. Hence $|F \cap \binom{A_i}{2}| = 1$ for each $i \in [0, p-1]$, and there are no edits outside the sets $\binom{A_i}{2}$. There are five possible ways for an A_i of how $F \cap \binom{A_i}{2}$ can look like: It is either a deletion of the edge $v_{3i}v_{3i+1}$, $v_{3i+1}v_{3i+2}$, or $v_{3i+2}v_{3i+3}$ (henceforth referred to as types D^-, D^0, and D^+, respectively), or an addition of the edge $v_{3i}v_{3i+2}$ or $v_{3i+1}v_{3i+3}$ (henceforth called types C^- and C^+, respectively)—the sixth possibility, which has been left out, creates an induced C_4. Observe now that if some A_i has type D^-, then A_{i+1} also has type D^-, or otherwise a $P_4 v_{3i+1} - v_{3i+2} - v_{3i+3} - v_{3i+4}$ would remain in the graph. Similarly, if A_i has type D^+ then A_{i-1} also has type D^+. Hence, if type D^+ or D^- appears for any A_i, then all the A_is have the same type. Observe now that if some A_i had type C^- and C^+, then A_{i-1} would have to have type D^+ and A_{i+1} would have to have type D^- or otherwise an unresolved P_4 would appear; This is a contradiction with the previous observations, since types D^- and D^+ cannot appear simultaneously. Hence, we are left with only three possibilities: all the A_is have type D^-, or all have type D^0, or all have type D^+. □

Proof of Claim 8 Denote the vertices of H as in Fig. 9. Consider the following three P_4s in H:

(i) $a_2 - a_1 - v - c_1$,
(ii) $b_2 - b_1 - v - a_1$, and
(iii) $c_2 - c_1 - v - b_1$.

Observe that any edge addition in H can destroy at most one of these P_4s, and a deletion of any of edges a_1a_2, b_1b_2, or c_1c_2 also can destroy at most one of these P_4s. Moreover, a deletion of any of the edges incident to the center v destroys only two of them. We infer that $|F| \geq 2$ since no single edit can destroy all three considered P_4s, and moreover if $|F| = 2$, then F contains at least one deletion of an edge incident to v, say va_1. After deleting this edge we are left with a $P_5 b_2 - b_1 - v - c_1 - c_2$, and it can be readily checked that the only way to edit it to a trivially perfect graph using only one edit is to delete vb_1 or vc_1. Thus, any editing set F with $|F| = 2$ in fact consists of deletions of two edges incident to v. □

Lemma 21 *The input* 3Sat *instance* φ *is satisfiable if and only if* (G_φ, k_φ) *is a yes-instance of* Trivially Perfect Editing.

Proof Suppose φ is satisfiable and let $\alpha : \mathcal{V}(\varphi) \to \{\top, \bot\}$ be a satisfying assignment. Define editing set $F^\alpha = \bigcup_{x \in \mathcal{V}(\varphi)} F_x^\alpha \cup \bigcup_{c \in \mathcal{C}(\varphi)} F_c^\alpha$; Note that F consists of deletions only. Then we have that $|F^\alpha| = k_\varphi$ and it can be easily seen that $G \triangle F$ is a disjoint union of components of constant size, each being a paw or a cricket (see Fig. 10). Both

(a) Paw **(b)** Cricket

Fig. 10 Shapes of components of G after editing deletion sets F_x^α and F_c^α for α being a satisfying assignment. Both of them are trivially perfect, so a disjoint union of any number of their copies is also trivially perfect

these graphs are trivially perfect, so a disjoint union of any number of their copies is also a trivially perfect graph. Thus F^α is a solution to the instance (G_φ, k_φ).

For the other direction, let $F \subseteq \binom{V(G_\varphi)}{2}$ be an editing set such that $G_\varphi \triangle F$ is trivially perfect, and $|F| \leq k_\varphi$. For every $x \in \mathcal{V}(\varphi)$ consider the subgraph G_x. For every $c \in \mathcal{C}(\varphi)$ consider the subgraph G_c induced in G by

- vertex v_c;
- the three neighbors of v_c, say $\square^x_{i_x}$, $\square^y_{i_y}$, and $\square^z_{i_z}$, where x, y, z are variables appearing in c and each symbol $\square$ is replaced by $\perp$ or $\top$ depending whether the variable's occurrence is positive or negative; and
- vertices $\mathsf{P}^x_{i_x}$, $\mathsf{P}^y_{i_y}$, and $\mathsf{P}^z_{i_z}$.

Observe that each G_x is isomorphic to a cycle on $3p_x$ vertices and each G_c is isomorphic to a subdivided claw. Moreover, all these subgraphs pairwise share at most one vertex, which means that sets $\binom{V(G_x)}{2}$ for $x \in \mathcal{V}(\varphi)$ and $\binom{V(G_c)}{2}$ for $c \in \mathcal{C}(\varphi)$ are pairwise disjoint. By Claim 7 we infer that $|F \cap \binom{V(G_x)}{2}| \geq p_x$ for each $x \in \mathcal{V}(\varphi)$, and by Claim 8 we infer that $|F \cap \binom{V(G_c)}{2}| \geq 2$ for each $c \in \mathcal{C}(\varphi)$. Thus

$$|F| \geq \sum_{x \in \mathcal{V}(\varphi)} p_x + 2|\mathcal{C}(\varphi)| = k_\varphi.$$

Hence, in fact $|F| = k_\varphi$ and all the used inequalities are in fact equalities: $|F \cap \binom{V(G_x)}{2}| = p_x$ for each $x \in \mathcal{V}(\varphi)$ and $|F \cap \binom{V(G_c)}{2}| = 2$ for each $c \in \mathcal{C}(\varphi)$. Using Claims 7 and 8 again, we infer that F has the following form: it consists of deletions only, from every cycle G_x it deletes every third edge, and for every vertex v_c it deletes two out of three edges incident to it. In particular, no edit is incident to any of the vertices P^x_i for $x \in \mathcal{V}(\varphi)$ and $i \in [0, p_x - 1]$.

Consider now the cycle G_x; We already know that the solution deletes either all the edges $\perp^x_i \top^x_i$ for $i \in [0, p_x - 1]$, or all the edges $\top^x_i \amalg^x_i$ for $i \in [0, p_x - 1]$, or all the edges $\amalg^x_i \perp^x_{i+1 \bmod p_x}$ for $i \in [0, p_x - 1]$. Observe that the first case cannot happen, since then we would have an induced P_4 $\perp^x_i - \mathsf{P}^x_i - \top^x_i - \amalg^x_i$ remaining in the graph—no other edit can destroy it. Hence, one of the latter two cases happen. Construct an assignment $\alpha : \mathcal{V}(\varphi) \to \{\top, \perp\}$ by, for each $x \in \mathcal{V}(\varphi)$, putting $\alpha(x) = \perp$ if all the edges $\top^x_i \amalg^x_i$ are included in F, and $\alpha(x) = \top$ if all the edges $\amalg^x_i \perp^x_{i+1 \bmod p_x}$ are included in F. We now claim that α satisfies φ.

For the sake of contradiction, suppose that a clause $c = \ell_x \vee \ell_y \vee \ell_z$ is not satisfied by α. Let e be the edge incident to v_c which has not been removed and suppose without loss of generality that this edge connects v_c with G_x. Suppose further that $\ell_x = x$, i.e., x appears positively in c, so $e = v_c \top_i^x$ for some $i \in [0, p_x - 1]$. Since x does not satisfy $c, \alpha(x) = \bot$ and both edges $\amalg_{i-1 \bmod p_x}^x \bot_i^x$ and $\bot_i^x \top_i^x$ are not deleted in F—the deleted edge is $\top_i^x \amalg_i^x$. But then we have the following induced P_4: $v_c - \top_i^x - \bot_i^x - \amalg_{i-1 \bmod p_x}^x$, which contradicts the assumption that $G_\varphi \triangle F$ is trivially perfect. The case when $\ell_x = \neg x$, i.e., x appears negatively in c, is symmetric.

Hence α is indeed a satisfying assignment for φ and we are done. □

Lemma 21 guarantees that the reduction is correct, and hence Theorem 8 follows by a straightforward application of Proposition 1. We can also observe that this reduction works immediately for Trivially Perfect Deletion as well since every optimal edit set consisted purely of deletions (see Claims 7 and 8), however this result is known [12].

6.1 Cographs

Let us recall that since $\overline{P_4} = P_4$, the problems Cograph Deletion and Cograph Completion are polynomial-time equivalent. The NP-hardness of Cograph Editing was first shown by Liu et al. [30], however, their reduction from Exact 3-Cover, adapted from the proof of the NP-hardness of Cograph Deletion by El-Mallah and Colbourn [15] suffers a quadratic blow-up in the parameter, and has $\Omega(|C|^6)$ vertices, where $|C|$ is the number of sets in the input instance. Hence, this reduction is unsuitable for showing the kind of lower bounds we are after.

Instead, we leverage the reduction provided in the previous section to prove the following result.

Theorem 9 Cograph Completion, Cograph Deletion, *and* Cograph Editing *are NP-complete and, under ETH, cannot be solved in time* $2^{o(k)} \operatorname{poly}(n)$ *nor* $2^{o(n+m)}$*, even on graphs with maximum degree* 4.

In fact, the reduction given in the previous section already is sufficient for showing Theorem 9. However, to prove this we need to slightly modify the analysis. This is done in the next lemma.

Lemma 22 *Given an instance* φ *of* 3Sat*,* φ *is satisfiable if and only if* (G_φ, k_φ) *is a yes-instance of* Cograph Editing *if and only if* (G_φ, k_φ) *is a yes-instance of* Cograph Deletion.

Proof Consider the following five statements.

(1) φ is a yes-instance of 3Sat.
(2) (G_φ, k_φ) is a yes-instance of Trivially Perfect Editing.
(3) (G_φ, k_φ) is a yes-instance of Trivially Perfect Deletion.
(4) (G_φ, k_φ) is a yes-instance of Cograph Editing.
(5) (G_φ, k_φ) is a yes-instance of Cograph Deletion.

The proof of Lemma 21, together with the remark given afterwards about the constructed editing set consisting purely of deletions, shows that statements (1), (2), and (3) are equivalent. Clearly, since every trivially perfect graph is also a cograph, we have that statement (2) implies statement (4), and statement (3) implies statement (5). Statement (5) trivially implies statement (4). Therefore, to prove the equivalence of all the above statements it suffices to prove that statement (4) implies statement (1), that is, the existence of a cograph editing set of size at most k_φ implies that φ is satisfiable.

To show this, we examine the argument verifying the analogous implication in the proof of Lemma 21; that is, that the existence of an editing set F of size at most k_φ such that $G_\varphi \triangle F$ is trivially perfect implies that φ is satisfiable. We show that it is easy to modify the argument so that we use only the property that $G_\varphi \triangle F$ is P_4-free, i.e., it is a cograph; then the implication from statement (4) to statement (1) follows. Observe that we relied on the assumption that $G_\varphi \triangle F$ is C_4-free only in two places.

- In the proof of Claim 7, we used C_4-freeness to exclude one of the six ways of how $F \cap \binom{A_i}{2}$ can look like. More precisely, we excluded the possibility of adding the edge $v_{3i} v_{3i+3}$, as then A_i would induce a C_4. However, even if this type, called further C^0, was not excluded a priori, it is easy to see that its occurrence would lead to the same contradiction as the occurrence of types C^- or C^+. Indeed, if type C^0 appeared in some A_i, then we would necessarily have that A_{i-1} has type D^+ and A_{i+1} has type D^-, just as for types C^- and C^+, for otherwise an unresolved P_4 would appear. However, we argued that the simultaneous appearance of types D^- and D^+ would lead to a contradiction.
- In the proof of Claim 8, we argued that the only way to edit a P_5 to a trivially perfect graph using one edit is to delete one of the edges incident to the middle vertex. This claim holds also when we consider editing to a cograph.

Thus, Claims 7 and 8 hold even if we only assume that $G_\varphi \triangle F$ is P_4-free. It is easy to see that the rest of the proof of Lemma 21 relies only on P_4-freeness, and hence we are done. □

Theorem 9 follows by combining Lemma 22 above with Lemma 21, in the same way as Theorem 8 followed from Lemma 21.

7 Conclusion

In this paper we gave the first polynomial kernels for Trivially Perfect Editing and Trivially Perfect Deletion, which answers an open problem by Nastos and Gao [33], and Liu et al. [29]. We also proved that assuming ETH, Trivially Perfect Editing does not have a subexponential parameterized algorithm. Together with the earlier results [12,24], we thus obtain a complete picture of the existence of polynomial kernels and subexponential parameterized algorithms for edge modification problems related to trivially perfect graphs; see Fig. 1 for an overview. In particular, the fact that all three problems Trivially Perfect Editing, Trivially Perfect Completion, and Trivially Perfect Deletion admit polynomial kernels, stands in an interesting contrast with the results of Cai and Cai [6], who showed that this is not the case for any of C_4-Free Editing, C_4- Free Completion and C_4-Free Deletion.

Table 1 Graph modification problems related to trivially perfect graphs and cographs

Problem	Polynomial kernel	Subexp. time
Trivially Perfect Completion	Yes [24]	Yes [12]
Trivially Perfect Deletion	Yes	No [12]
Trivially Perfect Editing	Yes	No
Cograph completion	Yes [23]	No [12]
Cograph deletion	Yes [23]	No [12]
Cograph editing	Yes [23]	No

The main contribution of the paper is the proof that Trivially Perfect Editing admits a polynomial kernel with $O(k^7)$ vertices. We apply the existing technique of constructing a *vertex modulator*, but with a new twist: The fact that we are solving an edge modification problem enables us also to argue about the adjacency structure between the modulator and the rest of the graph, which is helpful in understanding the structure of the instance. This approach is of general nature, as witnessed by the fact that it was successfully applied to other edge modification problems as well [7,11,34].

Finally, we showed that both Trivially Perfect Editing and Cograph Editing, in addition to being NP-complete, are not solvable in subexponential parameterized time unless the exponential time hypothesis fails. The same result was known for Trivially Perfect Deletion, but contrasts the previous result that the completion variant *does admit* a subexponential parameterized algorithm [12] (Table 1).

Let us conclude by stating some open questions. In this paper, we focused purely on constructing a polynomial kernel for Trivially Perfect Editing and related problems, and in multiple places we traded possible savings in the overall kernel size for simpler arguments in the analysis. We expect that a tighter analysis of our approach might yield kernels with $O(k^6)$ or even $O(k^5)$ vertices, but we think that the really challenging question is to match the size of the cubic kernel for Trivially Perfect Completion of Guo [24].

Generally, we find the vertex modulator technique very well-suited for tackling kernelization of edge modification problems, since it is at the same time versatile, and exposes well the structure of a large graph that is close in the edit distance to some graph class. We have high hopes that this generic approach will find applications in other edge modification problems as well, both in improving the sizes of existing kernels and in finding new positive results about the existence of polynomial kernels. For concrete questions where the technique might be applicable, we propose the following:

1. Is it possible to improve the $O(k^3)$ vertex kernels for Cograph Editing and Cograph Completion of Guillemot et al. [23]?
2. Do the Claw-Free Edge Deletion or Line Graph Edge Deletion problems admit polynomial kernels? Here, the task is to remove at most k edges to obtain a graph that is *claw-free*, i.e., does not admit an induced $K_{1,3}$ as an induced subgraph, respectively is a line graph. Recently, Cygan et al. [7] gave a polynomial kernel for the related {Claw,Diamond}- Free Edge Deletion problem.
3. Does Interval Completion admit a polynomial kernel?

Acknowledgements The authors thank Daniel Lokshtanov for pointing out that our kernelization procedure can be also used to give a polynomial upper bound on the sizes of minimal obstructions for k-editing/deletion/completion to a trivially perfect graph; that is, to derive the results given in Sect. 5. The authors are also grateful to Peter Novotný for suggesting a simpler way to prove Lemma 2.

References

1. Alon, N., Lokshtanov, D., Saurabh, S.: Fast fast. In: ICALP 2009, LNCS, vol. 5555, pp. 49–58. Springer (2009)
2. Bliznets, I., Fomin, F.V., Pilipczuk, M., Pilipczuk, M.: A subexponential parameterized algorithm for proper interval completion. SIAM J. Discrete Math. **29**(4), 1961–1987 (2015)
3. Bliznets, I., Fomin, F.V., Pilipczuk, M., Pilipczuk, M.: Subexponential parameterized algorithm for interval completion. In: SODA 2016, pp. 1116–1131. SIAM (2016)
4. Burzyn, P., Bonomo, F., Durán, G.: NP-completeness results for edge modification problems. Discrete Appl. Math. **154**(13), 1824–1844 (2006)
5. Cai, L.: Fixed-parameter tractability of graph modification problems for hereditary properties. Inf. Process. Lett. **58**(4), 171–176 (1996)
6. Cai, L., Cai, Y.: Incompressibility of H-free edge modification problems. Algorithmica **71**(3), 731–757 (2015)
7. Cygan, M., Pilipczuk, M., Pilipczuk, M., van Leeuwen, E.J., Wrochna, M.: Polynomial kernelization for removing induced claws and diamonds. Theory Comput. Syst. **60**(4), 615–636 (2017)
8. Demaine, E.D., Fomin, F.V., Hajiaghayi, M., Thilikos, D.M.: Subexponential parameterized algorithms on graphs of bounded genus and H-minor-free graphs. J. ACM **52**(6), 866–893 (2005)
9. Downey, R.G., Fellows, M.R.: Parameterized Complexity. Springer, Berlin (1999)
10. Drange, P.G.: Parameterized graph modification algorithms. Ph.D. dissertation, University of Bergen, Norway (2015)
11. Drange, P.G., Dregi, M.S., Lokshtanov, D., Sullivan, B.D.: On the threshold of intractability. In: ESA 2015, LNCS, vol. 9294, pp. 411–423. Springer (2015)
12. Drange, P.G., Fomin, F.V., Pilipczuk, M., Villanger, Y.: Exploring the subexponential complexity of completion problems. ACM Trans. Comput. Theory **7**(4), 14:1–14:38 (2015)
13. Drange, P.G., Pilipczuk, M.: A polynomial kernel for trivially perfect editing. In: ESA 2015, LNCS, vol. 9294, pp. 424–436. Springer (2015)
14. Edmonds, J.: Paths, trees, and flowers. Can. J. Math. **17**(3), 449–467 (1965)
15. El-Mallah, E., Colbourn, C.: The complexity of some edge deletion problems. IEEE Trans. Circuits Syst. **35**(3), 354–362 (1988)
16. Flum, J., Grohe, M.: Parameterized Complexity Theory. Springer, New York (2006)
17. Fomin, F.V., Kratsch, S., Pilipczuk, M., Pilipczuk, M., Villanger, Y.: Tight bounds for parameterized complexity of cluster editing with a small number of clusters. J. Comput. Syst. Sci. **80**(7), 1430–1447 (2014)
18. Fomin, F.V., Lokshtanov, D., Misra, N., Saurabh, S.: Planar F-deletion: approximation, kernelization and optimal FPT algorithms. In: FOCS 2012, pp. 470–479. IEEE (2012)
19. Fomin, F.V., Saurabh, S., Villanger, Y.: A polynomial kernel for proper interval vertex deletion. SIAM J. Discrete Math. **27**(4), 1964–1976 (2013)
20. Fomin, F.V., Villanger, Y.: Subexponential parameterized algorithm for minimum fill-in. SIAM J. Comput. **42**(6), 2197–2216 (2013)
21. Gallai, T.: Transitiv orientierbare graphen. Acta Math. Acad. Sci. Hung. **18**(1–2), 25–66 (1967)
22. Ghosh, E., Kolay, S., Kumar, M., Misra, P., Panolan, F., Rai, A., Ramanujan, M.S.: Faster parameterized algorithms for deletion to split graphs. Algorithmica **71**(4), 989–1006 (2015)
23. Guillemot, S., Havet, F., Paul, C., Perez, A.: On the (non-)existence of polynomial kernels for P_l-free edge modification problems. Algorithmica **65**(4), 900–926 (2013)

24. Guo, J.: Problem kernels for NP-complete edge deletion problems: split and related graphs. In: ISAAC 2007, LNCS, vol. 4835, pp. 915–926. Springer (2007)
25. Impagliazzo, R., Paturi, R., Zane, F.: Which problems have strongly exponential complexity? J. Comput. Syst. Sci. **63**(4), 512–530 (2001)
26. Jing-Ho, Y., Jer-Jeong, C., Chang, G.J.: Quasi-threshold graphs. Discrete Appl. Math. **69**(3), 247–255 (1996)
27. Komusiewicz, C., Uhlmann, J.: Cluster editing with locally bounded modifications. Discrete Appl. Math. **160**(15), 2259–2270 (2012)
28. Kratsch, S., Wahlström, M.: Two edge modification problems without polynomial kernels. Discrete Optim. **10**(3), 193–199 (2013)
29. Liu, Y., Wang, J., Guo, J.: An overview of kernelization algorithms for graph modification problems. Tsinghua Sci. Technol. **19**(4), 346–357 (2014)
30. Liu, Y., Wang, J., Guo, J., Chen, J.: Complexity and parameterized algorithms for cograph editing. Theor. Comput. Sci. **461**, 45–54 (2012)
31. Mancini, F.: Graph modification problems related to graph classes. Ph.D. thesis, University of Bergen (2008)
32. McConnell, R.M., Spinrad, J.: Modular decomposition and transitive orientation. Discrete Math. **201**(1–3), 189–241 (1999)
33. Nastos, J., Gao, Y.: Familial groups in social networks. Soc. Netw. **35**(3), 439–450 (2013)
34. Sandeep, R.B., Sivadasan, N.: Parameterized lower bound and improved kernel for diamond-free edge deletion. In: IPEC 2015, LIPIcs, vol. 43, pp. 365–376. Schloss Dagstuhl, Leibniz-Zentrum fuer Informatik (2015)
35. Sauer, N.: On the density of families of sets. J. Comb. Theory Ser. A **13**(1), 145–147 (1972)
36. Shelah, S.: A combinatorial problem; stability and order for models and theories in infinitary languages. Pac. J. Math. **41**(1), 247–261 (1972)
37. Yannakakis, M.: Edge-deletion problems. SIAM J. Comput. **10**(2), 297–309 (1981)