(Sub)linear kernels for edge modification problems toward structured graph classes

Gabriel Bathie, Nicolas Bousquet, Yixin Cao, Yuping Ke, Théo Pierron

(Sub)linear kernels for edge modification problems toward structured graph classes*

Gabriel Bathie
École Normale Supérieure de Lyon, France

Nicolas Bousquet
LIRIS, CNRS, Université Claude Bernard Lyon 1, Université de Lyon, France

Yixin Cao
Department of Computing, Hong Kong Polytechnic University, Hong Kong, China

Yuping Ke
Department of Computing, Hong Kong Polytechnic University, Hong Kong, China

Théo Pierron[1]
LIRIS, CNRS, Université Claude Bernard Lyon 1, Université de Lyon, France

Abstract

In a (parameterized) graph edge modification problem, we are given a graph G, an integer k and a (usually well-structured) class $\mathcal{G}$ of graphs, and asked whether it is possible to transform G into a graph $G' \in \mathcal{G}$ by adding and/or removing at most k edges. Parameterized graph edge modification problems received considerable attention in the last decades.

In this paper, we focus on finding small kernels for edge modification problems. One of the most studied problems is the Cluster Editing problem, in which the goal is to partition the vertex set into a disjoint union of cliques. Even if a $2k$-vertex kernel exists for Cluster Editing, this kernel does not reduce the size of the instance in most cases. Therefore, we explore the question of whether linear kernels are a theoretical limit in edge modification problems, in particular when the target graph class is very structured (such as a partition into cliques for instance). We prove, as far as we know, the first sublinear kernel for an edge modification problem. Namely, we show that Clique + Independent Set Deletion, which is a restriction of Cluster Deletion, admits a kernel of size $O(k/\log k)$.

We also obtain small kernels for several other edge modification problems. We first show that Cluster Deletion admits a $2k$-vertex kernel as Cluster Editing, improving the previous $4k$-vertex kernel. We prove that (Pseudo-)Split Completion (and the equivalent (Pseudo-)Split Deletion) admits a linear kernel, improving the existing quadratic kernel. We also prove that Trivially Perfect Completion admits a quadratic kernel (improving the cubic kernel), and finally prove that its triangle-free version (Starforest Deletion) admits a linear kernel, which is optimal under the Exponential Time Hypothesis.

2012 ACM Subject Classification Theory of computation → Parameterized complexity and exact algorithms

Keywords and phrases kernelization, graph editing, split graphs, (sub)linear kernels

[1] Corresponding author: theo.pierron@univ-lyon1.fr

* This work was supported by ANR project GrR (ANR-18-CE40-0032), RGC grants 15201317 and 15226116, and NSFC grant 61972330.

1 Introduction

A central problem in the context of data transmission, collection or storage, is to recover the original information when the data have been altered. Although it is not possible to know what the original data were in the general setting, it may be possible when we have some knowledge of the structure of the original data. When we know that the alteration is limited, it is reasonable to assume that the original data have the desired structure and is the closest to the altered data.

When the data that we are recovering form a graph, the problem becomes the following: given a graph G (the altered data) and a class $\mathcal{G}$ of graphs (the structure of the data), find a graph in $\mathcal{G}$ that is the "closest" to G (candidate for the original data). There are multiple ways to define the distance between two graphs, but the most widely used is the minimum number of vertex modifications or edge modifications needed to turn one into the other. This type of problem, called graph modification problems, received considerable attention, for instance in computational biology [2], machine learning [1], and image processing [31].

In this work, we focus on edge modification problems, i.e., the distance is the minimum number of edge modifications (see Section 2 for formal definitions of these problems). A line of work, initiated by Yannakakis [33], showed that deciding whether a graph G is at distance at most k from $\mathcal{G}$ is NP-complete for most classes of graphs, even for very restricted classes such as bipartite graphs. See [5, 27, 28] for an overview of the different results.

Therefore, in the last decades, edge modification problems received considerable attention from the point of view of parameterized complexity, which studies the resources required to solve NP-complete problems in a fine-grained way. See for example [3, 4, 6, 11, 15, 19, 29], and see [9] for a recent survey on the topic. In this paper, we will consider problems parameterized by the size k of the solution (that is, the set of edges to add or remove).

In this work, we focus on graph classes that can be characterized by a finite number of forbidden induced subgraphs. In his seminal paper, Cai [6] showed that, for every class $\mathcal{G}$ of graphs that can be characterized by a finite number of forbidden induced subgraphs, the $\mathcal{G}$-edge modifications problems are FPT parameterized by k. In other words, there exists a constant c, a function f (that only depend on $\mathcal{G}$), and an algorithm running in time $f(k) \cdot n^c$ that either finds a solution of size at most k, or returns that there is no such solution. Therefore, most of the subsequent efforts focused on determining for which $\mathcal{G}$ these problems admit a polynomial kernel. Intuitively, a kernel is a polynomial-time preprocessing algorithm that extracts the "hard" part of an instance (G, k): it solves easy parts of the instance and returns an equivalent instance (G', k'), whose size is bounded by $f(k')$, for some function f. A kernel is a polynomial kernel if f is a polynomial. The interested reader is referred to [18] for more details.

One of the most studied edge modification problems is Cluster Editing, in which the goal is to partition the graph into a disjoint union of cliques. This problem is known to admit a kernel with at most $2k$ vertices [7]. While this result seems impressive at first glance (for many parameterized problems, a linear kernel is asymptotically optimal), we can remark that here, we are comparing the number of vertices of the kernel with the number of edges in the solution. It turns out that for most graphs in practice, the number of edges that have to be modified to obtain a cluster graph is larger than the number of vertices. For example, it is the case for most of the public instances of the PACE challenge 2021 on cluster editing[2].

This raises the question of whether linear kernels are optimal, in particular for Cluster

[2] See `https://pacechallenge.org/2021/` for more information.

Editing. We partially answer this question by giving a sublinear kernel for the closely related Clique + IS Deletion problem. It provides a "proof of concept" that linear kernels are not always optimal. As far as we know, it is the first example of a sublinear kernel for a graph edge modification problem. We provide linear kernels for several well-studied edge modification problems, with the only exception of the Trivially Perfect Completion problem, for which we can only obtain a quadratic-vertex kernel.

Our results. In this work, our goal is to understand when it is possible to obtain small, and in particular linear or sublinear kernels for edge modification problems. We focus in particular on graph classes where the vertex set can be partitioned into highly structured classes such as cliques or independent sets. A typical example of such a graph class is the class of split graphs, i.e., graphs that can be partitioned into a clique and an independent set.

Most of our results are based on a high-level technique, that we call Label-And-Reduce, which helps to design efficient kernelization algorithms for edge modification problems. The key idea is to use the strong structure of each of the graphs of $\mathcal{G}$ to find a highly structured partition $X_1, \ldots, X_\ell$ of the graphs of $\mathcal{G}$ (e.g., a partition in cliques or independent sets, complete bipartition between subsets...). We then define rules that label vertices x as belonging to X_i, in such a way that if there is a solution, there is one for which $x \in X_i$. We finally show that 1) when no rule can be applied, the number of unlabeled vertices is $O(\text{poly}(k))$ when (G, k) is a positive instance and 2) the number of labeled vertices in each class X_i can be reduced to $O(\text{poly}(k))$.

(Sub)linear kernels for Cluster deletion and Clique+Independent Set deletion. Since cluster graphs are P_3-free graphs, edge modification problems toward cluster graphs seem to be the simplest among all nontrivial graph modification problems toward H-free classes. Both Cluster Deletion and Cluster Editing have indeed received considerable attention in the last two decades in parameterized complexity, see e.g., [4, 7, 8, 17, 29].

After a sequence of results, Cao and Chen [7] devised a $2k$-vertex kernel for Cluster Editing. Their algorithm actually implies a $2k$-vertex kernel for Cluster Deletion, improving on the $4k$-vertex kernel [20]. We record this simple result for future reference. Less trivially, we show that the same algorithm produces a kernel of the same size for the Strong Triadic Closure problem, which, though originally not posed as an edge modification problem, is closely related to cluster edge deletion [25]. We introduce this problem in Section 3 and prove the following.

▶ **Theorem 1.1.** *Cluster Deletion and Strong Triadic Closure both admit $2k$-vertex kernels.*

As the original results [7], both algorithms work for the weighted versions of the problems as well.

We then focus on a restricted class of cluster graphs, where all clusters but at most one have size 1. It corresponds to graphs that are the disjoint union of a clique and an independent set. In what follows, we will refer to this class as the class of *Clique + IS graphs*. While Clique + IS Completion is trivial, (since the optimal solution is found by changing each connected component into a clique,) both Clique + IS Deletion and Clique + IS Editing are NP-complete (reduction from the Clique problem), and both can be solved in subexponential time ($O^*(1.64^{\sqrt{k \ln k}})$ and $O^*(2^{\sqrt{k \ln k}})$ respectively[3]) [11]. Since Clique +

[3] Recall that O^* denotes the complexity up to polynomial factors.

IS graphs are $(P_3, 2K_2)$-free graphs, Clique + IS Deletion is FPT by [6]. We prove the following result in Section 4.

▶ **Theorem 1.2.** *Clique + IS Deletion admits a kernel of* $2k/\log k + 1$ *vertices.*

Our algorithm uses the structure of clique+IS graphs to remove vertices with small degree and to reduce the instance when the minimum degree of the input graph is large. Theorem 1.2 is, as far as we know, the first sublinear kernel for edge modification problems. We conjecture that the size of this kernel is not optimal, and ask the following:

▶ **Open Problem 1.** Is there an $O(k^{1-\varepsilon})$-vertex kernel for Clique + IS Deletion for some $\varepsilon > 0$?

Moreover, it is plausible that other edge modification problems toward highly structured classes also admit a sublinear kernel. A natural candidate is the closely related Cluster Editing problem, which already admits a $2k$ kernel [7, 8].

▶ **Open Problem 2.** Does Cluster Editing (resp. Cluster Deletion) admit a sublinear kernel?

Let us remark that Komusiewicz and Uhlmann [24] have proved that neither of the two problems can be solved in subexponential time.

A linear kernel for (Pseudo-)Split Completion. *Split graphs* are graphs whose vertex set can be partitioned into a clique K and an independent set I (with no constraint on the set of edges between K and I). Since split graphs are closed under complementation, the Split Completion and Split Deletion problems are equivalent. Natanzon et al. [28] showed that these two problems are NP-complete. Since split graphs are $(2K_2, P_4, C_5)$-free graphs, the latter problems are FPT [6]. Ghosh et al. [19] later showed that these problems can be solved in subexponential $O^*(2^{O(\sqrt{k}\log k)})$ time, and that they admit a quadratic-vertex kernel. Cygan et al. [10] improved the complexity to $O^*(2^{\sqrt{k}})$. A classic result of Hammer and Simeone [22] states that the related Split Editing problem can be decided in polynomial time.

We improve upon the result of Ghosh et al. [19] by showing that the Split Completion (and therefore Split Deletion) problem admits a linear kernel in Section 5.

▶ **Theorem 1.3.** *Split Completion and Split Deletion admit a kernel with at most* $11k + 6\sqrt{2k} + 4$ *vertices.*

This result is the main technical contribution of the paper. From a very high-level perspective, our algorithm works as follows. Let (G, k) be a positive instance. If the clique of the solution is large enough, the neighborhood of many vertices of that clique has not been modified and we show that we can detect some of them and label them as clique vertices. Since the number of unlabeled clique vertices of the solution is bounded by a linear function, we can prove via a tricky and short argument that the number of unlabeled vertices of the independent set can be bounded. While the reduction rules are not very complicated, showing that the answer is negative when the number of unlabeled vertices is too large is the core of the proof. We finally show that we reduce the number of labeled vertices to $O(k)$ vertices.

While we use a tricky argument to bound the number of unlabeled vertices of the independent set of a solution, our bound on the number of unlabeled clique vertices is quite harsh. This leads to the following question.

▶ **Open Problem 3.** Does Split Completion admit a sublinear kernel, or at least with a subquadratic number of edges?

With minor tweaks, our algorithm produces a linear kernel of roughly the same size for the pseudo-split ($(2K_2, C_4)$-free graphs) deletion problem. Equivalently, a pseudo-split graph is either a split graph or a split graph plus a C_5 such that every vertex on the C_5 is adjacent to every vertex in the clique part of the split graph and is non adjacent to any vertex in the independent part of the split graph. Similarly to edge modification problems toward split graphs, Pseudo-split Editing can be decided in polynomial time [12], and only subexponential algorithms are known for the equivalent Pseudo-split Completion and Pseudo-split Deletion [10, 13].

A quadratic kernel for trivially perfect graphs. A *trivially perfect graph* is a graph such that for any pair of adjacent vertices u, v satisfies $N(u) \subseteq N(v)$ or $N(v) \subseteq N(u)$. The class of trivially perfect graphs can equivalently be characterized as the class of (P_4, C_4)-free graphs. Drange et al. [13, 14] showed that, under the Exponential Time Hypothesis (ETH), Trivially Perfect Deletion and Trivially Perfect Editing cannot be solved in subexponential time. Liu et al. [26] gave an FPT algorithm for Trivially Perfect Deletion running in time $O^*(2.42^k)$. On the other hand, the Trivially Perfect Completion problem does not admit such lower bounds. Drange et al. [13] designed a subexponential $O(2^{\sqrt{k \log k}})$ algorithm for the problem, and Bliznets et al. [3] showed that assuming the ETH, this cannot be improved beyond $O(2^{k^{1/4}})$. In 2018, Drange and Pilipczuk [14] showed that the three problems admit a polynomial kernel of size $O(k^7)$, recently improved by Dumas et al. [16] into $O(k^3)$.

In the specific case of Trivially Perfect Completion, a cubic-vertex kernel was already provided by Guo [21]. Our kernel is based on one of its claims, which states that the instance can be reduced to vertices that belong to at least one obstruction (that is, an induced P_4 or C_4). By counting obstructions more precisely, we improve this result in Section 6 by showing the following.

▶ **Theorem 1.4.** *Trivially Perfect Completion admits a kernel with $2k^2 + 2k$ vertices.*

Note moreover that in the extended abstract of [21], the existence of the cubic kernel is based on a lemma whose proof has, to the best of our knowledge, never been published nor made accessible. We give a proof of this statement of Guo in this article.

A linear kernelization of starforests. We finally focus on triangle-free trivially perfect graphs, also known as starforests. A *star* is a tree with at most one internal vertex. A star with n vertices is called an *n-star*. Note that the single vertex graph, as well as a K_2, are stars. The class of *starforests* is the class of graphs that are a disjoint union of stars, that is every connected component is a star.

One can remark that removing an edge from a starforest yields another starforest, hence it is never interesting to add edges to obtain a star forest. Therefore, Starforest Completion is trivial, and Starforest Editing is equivalent to Starforest Deletion. Drange et al. [15] showed that Starforest Deletion is NP-complete and cannot be solved in subexponential time (that is in time $O(2^{o(k)} \operatorname{poly}(n))$), assuming the ETH [23].

In Section 7, we prove the following result.

▶ **Theorem 1.5.** *Starforest Deletion admits a kernel with at most $4k + 2$ vertices.*

We also show that, under the ETH, Starforest Deletion does not admit a sublinear kernel. To the best of our knowledge, this work is the first formally published work on kernelization of Starforest Deletion.

2 Preliminaries

Elementary definitions. In this work, all the graphs are undirected and simple (i.e., with no parallel edges or self-loops). When G is a graph, $V(G)$ denotes the set of vertices of G, and $E(G)$ denotes its set of edges. Throughout the paper, we use n (resp. m) to denote the size of $V(G)$ (resp. $E(G)$). If $uv \in E(G)$, we say that u and v are *adjacent*. Given a vertex $u \in V(G)$, the set $N(u) = \{v \text{ such that } uv \text{ is an edge}\}$ is the *open neighborhood* of u, and $N[u] = N(u) \cup \{u\}$ is the *closed neighborhood* of u. The *degree* of u in G, denoted $d(u)$, is the size of $N(u)$. We use $\delta(G)$ to denote the minimum degree of G. The *complement graph* $\bar{G}$ of G is the graph with vertex set $V(G)$ and edge set $\{uv \mid u \neq v \text{ and } uv \notin E(G)\}$. A *clique* of G is a set of vertices that are pairwise adjacent. An *independent set* of G is a set of vertices of G that are pairwise not adjacent. For two sets X and Y, we use $X \triangle Y = (X \setminus Y) \cup (Y \setminus X)$ to denote their *symmetric difference*.

Kernelization algorithms. A *kernelization algorithm* (in short, a kernel) is a polynomial-time algorithm that takes as input an instance (G, k) of a parameterized problem Π and outputs an instance (G', k') that is positive if and only if (G, k) is positive, the size of G' is at most $f(k')$ for some computable function f. When f is a polynomial, we say that the algorithm is a *polynomial kernel*. When dealing with graph problems, the size of the instance is often measured in terms of the number of vertices of G'. Most kernelization algorithms (including those presented in this work) consist of the iterative application of *reduction rules*. A *reduction rule* is an algorithm that transforms an instance (G, k) of Π into an instance (G', k') of Π in polynomial time. We say that a reduction rule R is *safe* if (G, k) is positive if and only if (G', k') is.

Graph edge modification problems. Given a set F of edges, we use the notation $G + F$, $G - F$, and $G \Delta F$ to denote the graphs with vertex set $V(G)$ and respective set of edges $E(G) \cup F$, $E(G) \setminus F$, and $E(G) \Delta F$. Let $\mathcal{G}$ be a class of graphs. In a (parameterized) $\mathcal{G}$-graph edge modification problem, we are given a graph G, an integer k and a class $\mathcal{G}$ of graphs, and ask whether it is possible to transform G into a graph $G' \in \mathcal{G}$ by modifying (*adding*, *removing*, or doing both, which is called *editing*) at most k edges.

Formally, we will consider the following problems:

$\mathcal{G}$-Completion (resp. Deletion, resp. Editing)

Input:	A graph G and a nonnegative integer k.
Output:	Is there a set F of at most k edges of G such that $G + F$ (resp. $G - F$, resp. $G \Delta F$) is in $\mathcal{G}$?

Given a graph G, we say that a set F of edges is a *solution* when $G + F$ (resp. $G - F$, $G \Delta F$) lies in $\mathcal{G}$. We denote by $\mathrm{opt}(G)$ the minimum size of a solution (the considered problem should be clear from the context). We sometimes abuse the notation to use a solution to denote the resulting graph obtained after applying a solution.

Here, we focus on cases where $\mathcal{G}$ is characterized by a finite set of forbidden induced subgraphs. An example of such graphs is the class of split graphs, which are $(2K_2, P_4, C_5)$-free

graphs. Hence we only consider hereditary graph classes $\mathcal{G}$, that is, if $G \in \mathcal{G}$, then any induced subgraph H of G is also in $\mathcal{G}$. As a result, if (G, k) is a positive instance of a $\mathcal{G}$-edge modification problem, then for any subgraph H of G, (H, k) is also a positive instance.

3 Cluster Deletion and Strong Triadic Closure

The goal of this section is to prove Theorem 1.1, which we recall here.

▶ **Theorem 1.1.** *Cluster Deletion and Strong Triadic Closure both admit $2k$-vertex kernels.*

We first prove the kernel for Cluster Deletion following the ideas of [7]. A trivial but crucial fact is that a solution F is incident to at most $2|F|$ endpoints. If a vertex v is not incident to any edge in F, then v has to be *simplicial* (i.e., $N[v]$ is a clique in G). The first rule allows us to reduce some simplicial vertices. We then show that when this rule cannot be applied, every positive instance is small, that is we can reject every big instance.

For a vertex set $U \subseteq V(G)$, we write $d(U) = |E(U, V(G) \setminus U)|$, i.e., the number of edges between U and $V(G) \setminus U$; note that $d(\{v\})$ is precisely $d(v)$, so we use the shorter form.

▶ **Rule 3.1.** *If there is a simplicial vertex v such that $d(N[v]) \leq d(v)$, then remove $N[v]$ and decrease k by $d(N[v])$.*

The safeness of this rule comes from the following lemma.

▶ **Lemma 3.2.** *If v satisfies the hypothesis of Rule 3.1, then adding all edges between $N[v]$ and $V(G) \setminus N[v]$ to an optimal solution of $G - N[v]$ yields an optimal solution for G.*

Proof. Clearly, the construction yields a solution for G of size $\mathrm{opt}(G - N[v]) + d(N[v])$. To prove that it yields an optimal solution, we show that $\mathrm{opt}(G) \geqslant \mathrm{opt}(G - N[v]) + d(N[v])$.

Let F be an optimal solution to the graph G. Let X be the clique of $G - F$ containing v. Note that $X \subseteq N[v]$. Moreover, if $X = N[v]$, then we are done hence we can assume that $X \subset N[v]$. In other words, neither X nor $N[v] \setminus X$ is empty. Since any induced subgraph of $G - F$ is a cluster graph, the subset of edges in F with both endpoints in $V(G) \setminus N[v]$ is a solution to $G - N[v]$. Noting that this solution is disjoint from $E(X, V(G) \setminus X)$, we have

$$\begin{aligned} \mathrm{opt}(G) = |F| &\geqslant |F \cap E(G - N[v])| + d(X) \\ &\geqslant \mathrm{opt}(G - N[v]) + |X| \cdot |N[v] \setminus X| \\ &\geqslant \mathrm{opt}(G - N[v]) + |X| + |N[v] \setminus X| - 1 \qquad (1) \\ &= \mathrm{opt}(G - N[v]) + d(v) \\ &\geqslant \mathrm{opt}(G - N[v]) + d(N[v]), \end{aligned}$$

where the third inequality holds because both $|X|$ and $|N[v] \setminus X|$ are positive integers. ◀

Let us mention that the condition of Rule 3.1 can be weakened to $d(N[v]) < 2d(v) - 1$. We do not prove the stronger statement because it does not improve the analysis of the kernel size, but let us briefly explain why it is true. The bound $\mathrm{opt}(G) \geq \mathrm{opt}(G - N[v]) + 2d(v) - 1$ holds unless $|X| = 1$ or $|N[v] \setminus X| = 1$; see the third inequality of (1). In the first case, v itself makes a trivial component, and all the vertices in $N(v)$ are in the same component; this can only happen when there exists another vertex u with $N(v) \subseteq N(u)$. In the second case, a vertex $u \in N(v)$ is incident to all the edges between $N(v)$ and $V(G) \setminus N[v]$. If $d(N[v]) < 2d(v) - 1$, then $\mathrm{opt}(G) \geq \mathrm{opt}(G - N[v]) + 2d(v) - 1$ holds in both cases.

We may now show that every positive instance is small.

▶ **Lemma 3.3.** *If (G, k) is a yes-instance where Rule 3.1 is not applicable, then $|V(G)| \leqslant 2k$.*

Proof. Let F be an optimal solution to G, and let $\{v_1, v_2, \ldots, v_r\}$ be the vertices that are not incident to any edge in F; they have to be simplicial in G. For $i = 1, \ldots, r$, the set $N[v_i]$ forms a component of $G - F$, therefore for $i, j \in \{1, \ldots, r\}$, the sets $N[v_i]$ and $N[v_j]$ are either the same or mutually disjoint.

We now double count the number of endpoints of edges in F (with multiplicity). On the one hand, there are $2|F|$ of them. On the other hand, note that for $i = 1, \ldots, r$, the number of such endpoints that lie in $N[v_i]$ is $d(N[v_i])$. Moreover, each of the vertices not in $\bigcup_{i=1}^{r} N[v_i]$ is an end of at least one edge in F. Therefore, we get a lower bound for $2|F|$:

$$2|F| \geqslant \sum_{i=1}^{r} d(N[v_i]) + \left|V(G) \setminus \bigcup_{i=1}^{r} N[v_i]\right| \geqslant \sum_{i=1}^{r} |N[v_i]| + \left|V(G) \setminus \bigcup_{i=1}^{r} N[v_i]\right| \geqslant |V(G)|.$$

The second inequality holds because Rule 3.1 does not apply to v_i for $i = 1, \ldots, r$, hence $d(N[v_i]) \geqslant d(v_i) = |N[v_i]|$. We finally obtain that $|V(G)| \leqslant 2|F| \leqslant 2k$. ◀

Therefore the following rule is safe, which gives a $2k$-vertex kernel for Cluster Deletion.

▶ **Rule 3.4.** *If Rule 3.1 cannot be applied and the size of the instance is at least $2k + 1$, return a trivially negative instance.*

To conclude the proof of Theorem 1.1, we show that the same rules apply to the Strong Triadic Closure problem we introduce hereafter.

In the original definition, which was motivated by applications in social networks, the *strong triadic closure* problem asks for a partition of the edge set of the input graph into strong edges and weak ones, such that for every two vertices that are linked to a common neighbor with strong edges are adjacent. The objective is to maximize the number of strong edges. For our purpose, it is more convenient to define the problem as follows.

Strong Triadic Closure

Input:	A graph G and a nonnegative integer k.
Output:	Is there a set F of at most k edges such that the missing edge of every P_3 of $G - F$ is in $E(G)$?

Thus, we call the set of weak edges the solution to the strong triadic closure problem. For any set $F \subseteq E(G)$, if $G - F$ is a cluster graph, then F is also a solution to the strong triadic closure problem: setting all edges in F weak, and all other edges strong is a feasible partition of $E(G)$. However, as illustrated in Figure 1, a strong triadic closure of a graph can have fewer weak edges than an optimal solution to the Cluster Deletion problem on the same graph.

Surprisingly, as we show below, Rule 3.1 works for Strong Triadic Closure without change. Moreover, a word-for-word copy of the proof of Lemma 3.3 shows that this is also the case for Rule 3.4, which concludes the proof of Theorem 1.1.

▶ **Lemma 3.5.** *Rule 3.1 is safe for Strong Triadic Closure.*

Proof. Similarly to the proof of Lemma 3.2, we show that adding all the edges from $E(N[v], V(G) \setminus N[v])$ to an optimal solution for $G - N[v]$ yields an optimal solution for

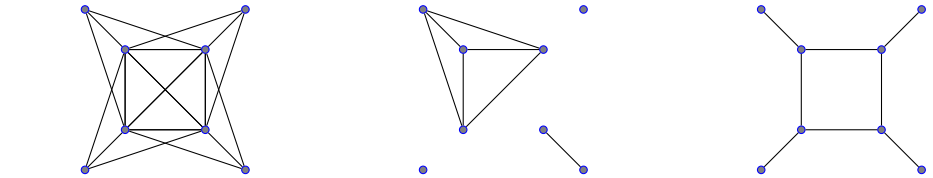

Figure 1 The example given by Konstantinidis et al. [25]: (a) the input graph; (b) a maximum cluster subgraph with seven edges; and (c) a maximum strong triadic closure with eight edges.

G. It is still clear that it is a solution, and we show that it is optimal by proving that $\mathrm{opt}(G) \geqslant \mathrm{opt}(G - N[v]) + d(N[v])$.

Let F be an optimal solution to the graph G. We may assume that $N[v]$ is not a separate component of $G - F$, otherwise the inequality already holds. Let $X = \{w \in V(G) \mid N[w] = N[v]\}$ denote the set of "true twins" of v, and $Y \subseteq N[v]$ the endpoints of edges in $E(N[v], V(G) \setminus N[v]) \setminus F$. Note that X, Y are disjoint; $X \neq \emptyset$ because $v \in X$; and $Y \neq \emptyset$ because $E(N[v], V(G) \setminus N[v]) \not\subseteq F$ (otherwise $N[v]$ would be a component of $G - F$).

We will now lower bound $|F|$ by counting how many edges from F can lie in $E(G - N[v])$, $E(N[v], V(G) \setminus N[v])$, and $E(N[v])$.

- First observe that by definition, the subset of edges in F with both endpoints in $V(G) \setminus N[v]$ is a solution to $G - N[v]$. Therefore $|F \cap E(G - N[v])| \geqslant \mathrm{opt}(G - N[v])$.
- Let $w \in N[v] \setminus (X \cup Y)$. Since v is simplicial and by definition of X, w must be incident to an edge from $E(N[v], V(G) \setminus N[v])$. Moreover, by definition of Y, all such edges must lie in F. Therefore, every vertex in $N[v] \setminus (X \cup Y)$ is incident to at least one edge in $F \cap E(N[v], V(G) \setminus N[v])$. Therefore, $|F \cap E(N[v], V(G) \setminus N[v])| \geqslant |N[v] \setminus (X \cup Y)|$.
- Finally, $|F \cap E(N[v])| \geqslant |X| \cdot |Y|$ since all the edges between X and Y lie in F. Indeed, assume that there is $x \in X$ and $y \in Y$ such that $xy \notin F$. By definition, there exists $z \in V(G) \setminus N[v]$ such that $yz \in E(G) \setminus F$, and since $z \notin N[v]$, z is not adjacent to a vertex in x. Therefore, xyz is a P_3 in $G - F$ but $xz \notin E(G)$, a contradiction.

We may now conclude the proof by wrapping up these three inequalities:

$$\begin{aligned}
\mathrm{opt}(G) = |F| &= |F \cap E(G - N[v])| + |F \cap E(N[v], V(G) \setminus N[v])| + |F \cap E(N[v])| \\
&\geqslant \mathrm{opt}(G - N[v]) + |N[v] \setminus (X \cup Y)| + |X| \cdot |Y| \\
&\geqslant \mathrm{opt}(G - N[v]) + |N[v]| - |X| - |Y| + |X| + |Y| - 1 \\
&\geqslant \mathrm{opt}(G - N[v]) + |N[v]| - 1 \\
&= \mathrm{opt}(G - N[v]) + d(v) \\
&\geqslant \mathrm{opt}(G - N[v]) + d(N[v]),
\end{aligned}$$

where $|X| \cdot |Y| \geq |X| + |Y| - 1$ because both $|X|$ and $|Y|$ are positive integers. ◀

We conclude this section with two remarks. First, for Strong Triadic Closure, we may alternatively state Rule 3.1 as follows.

▶ **Rule 3.6.** *If there is a simplicial vertex v such that $d(N[v]) \leq d(v)$, then set all the edges in $G[N[v]]$ strong, set all the edges between $N[v]$ and $V(G) \setminus N[v]$ weak, and delete $N[v]$.*

Finally, we claim that our kernelization algorithms for Cluster Deletion and Strong Triadic Closure work for the weighted versions as well; see [7].

4 Sublinear-vertex kernel for Clique + Independent Set Deletion

The goal of this section is to prove Theorem 1.2, which we recall here.

▶ **Theorem 1.2.** *Clique + IS Deletion admits a kernel of* $2k/\log k + 1$ *vertices.*

In order to obtain the announced kernel, we apply the Label-And-Reduce technique. For this problem, the labeling rules aim to identify vertices that will be in the independent set of a solution, assuming that a solution exists. We can then delete all the edges incident to these vertices, decrease the parameter accordingly, and remove these vertices from the graph.

We assume that k is smaller than m since otherwise, the instance is trivial: we obtain an independent set (which is a clique+IS graph) by deleting all the edges in G.

▶ **Rule 4.1** (Low degree reduction rule 1)**.** *If there exists* $v \in V(G)$ *with* $d(v) < \sqrt{2(m-k)} - 1$*, delete* v *from* G *and decrease the parameter by* $d(v)$*.*

This rule can be implemented to run in linear time. It is moreover safe. Indeed, since we consider the deletion problem, any vertex v deleted by the rule has degree smaller than $\sqrt{2(m-k)} - 1$ in $G - F$, hence cannot be in the clique of any optimal solution according to the following lemma.

▶ **Lemma 4.2.** *Let* (G, k) *is a positive instance of Clique + IS Deletion. If* F *is a solution of* (G, k)*, then the clique in* $G - F$ *has at least* $\sqrt{2(m-k)}$ *vertices.*

Proof. Since F contains at most k edges, the graph $G-F$ has at least $m-k$ edges. Moreover, $G - F$ is a clique+IS graph, and all its edges are the edges of a unique clique. Therefore, the order c of the clique of $G - F$ satisfies $\binom{c}{2} \geqslant m - k$, hence $c \geqslant \sqrt{2(m-k)}$. ◀

One can prove that this first rule can be extended to obtain a linear kernel. Indeed, when this rule cannot be applied, then all the vertices have degree at least $\sqrt{2(m-k)}$. Assuming $m \geqslant 2k$ (otherwise we are done), we have $\sqrt{2(m-k)} \geqslant \sqrt{m}$. The handshaking lemma then gives that $2m \geqslant n\sqrt{m}$ *i.e.* $n = O(\sqrt{m})$. Therefore, we get a linear kernel when $m = O(k^2)$. Otherwise, $m = \Omega(k^2)$ hence the minimum degree of the graph is $\Omega(\sqrt{2(m-k)}) = \Omega(k)$ and at most $O(1)$ vertices can be removed. In that case, the existence of a solution can be tested in polynomial time.

To further reduce the size of the kernel, we use two more rules to take care of very sparse or very dense instances.

▶ **Rule 4.3** (Low degree reduction rule 2)**.** *Let* v *be a vertex of degree at most* $2\log k - 1$*. If there is no solution* F *of* (G, k) *such that* v *is in the clique of* $G - F$*, remove* v *from* G *and decrease* k *by* $d(v)$*.*

This rule is trivially safe. Moreover, it can be performed in polynomial time. Indeed, since we consider an edge-deletion problem, if v lies in the clique K of $G - F$, then every vertex of K is adjacent to v in G, i.e., $K \subseteq N[v]$. Since the degree of v is at most $2\log k - 1$, there are at most k^2 subsets in $N[v]$. We can therefore try all of them and decide in polynomial time whether there exists a solution F of (G, k) such that v is in the clique in $G - F$.

▶ **Rule 4.4** (High degree reduction rule)**.** *If* G *has minimum degree* $\delta(G) \geq k/(2\log k)$*, solve the instance and output a trivial equivalent instance.*

Again, this rule is clearly safe. The not-so-easy part is to show that Clique + IS Deletion can be decided in polynomial time when $\delta(G) \geq k/(2\log k)$.

▶ **Lemma 4.5.** *Rule 4.4 can be applied in polynomial time.*

Proof. Note that removing an edge from G reduces by one the degree of two vertices. Therefore, if $\delta(G) \geq k/(2\log k)$, then for every solution F of at most k edges, the independent set part in $G-F$ contains at most $4\log k$ vertices. Moreover, Damaschke and Mogren showed in [11, Lemma 3] that a set F is an *optimal solution* of G if and only if I is a minimum vertex cover of $\bar{G}$, where I is the independent set part of $G-F$. In our case, this means that the minimum vertex cover of $\bar{G}$ has at most $4\log k$ vertices. As shown below, we can find such a minimum vertex cover in polynomial time in our case by combining an approximation algorithm with an exhaustive search on the approximate solution.

Using a greedy 2-approximation algorithm for Vertex Cover, we can compute a vertex cover J of $\bar{G}$ such that $I \subseteq J$ and $|J| \leq 2 \cdot |I|$. If $|J| > 8\log k$, then $|I| > 4\log k$, and hence the instance is negative. Otherwise, we can check if one of the k^8 subsets of J is the set of isolated vertices of a solution. Rule 4.4 can therefore be performed in polynomial time. ◀

To finish the proof of Theorem 1.2, it remains to bound the order of the reduced graph. This is the goal of the following lemma.

▶ **Lemma 4.6.** *If (G,k) is a positive instance and none of the rules can be applied, then $|V(G)| \leq 2 \cdot \frac{k}{\log k} + 1$.*

Proof. Observe that due to Rules 4.1 and 4.4, we have $\sqrt{2(m-k)} - 1 \leqslant \delta(G) < \frac{k}{2\log k}$, which implies that $m \leqslant \frac{k^2}{2\log^2 k}$ when $k > 257$. Note that we can assume without loss of generality that this last condition is satisfied since otherwise, we can solve the instance in polynomial time by taking every subset of k edges as a candidate for F; this can be done in time $O(n^k) = O(n^{257})$ (note that this degree can be reduced to the cost of increasing the multiplicative constant of our kernel).

Similarly to Lemma 4.2, we can show that the clique in $G-F$ has at most $\sqrt{2m}+1 \leq k/\log k + 1$ vertices.

Furthermore, Rule 4.3 ensures that $\delta(G) \geqslant 2\log k$. Removing an edge from G reduces by one the degree of two vertices. Therefore, by removing k edges from G, we can make at most $k/\log k$ vertices isolated. Since all the vertices of G are either in the clique or isolated, G has at most $k/\log k + 1 + k/\log k = 2k/\log k + 1$ vertices. ◀

Concluding remarks. The number of vertices in the kernel for a graph edge modification problem usually comes from an upper bound on the number of edges in a positive instance, which translates into a similar upper bound on the number of vertices. However, this is often far from tight. To obtain our sublinear kernel, we use the fact that a linear upper bound on the number of edges yields a sublinear upper bound on the number of vertices.

For example, in clique graphs with few isolated vertices (e.g., $o(n)$ isolated vertices), we have $m = \Theta(n^2)$, hence such a graph with $O(f(k))$ edges has $O(\sqrt{f(k)})$ vertices. Therefore, one could hope that Clique + IS Deletion admits a kernel of size $O(\sqrt{k})$. However, we were not able to obtain such a result. Note that our reduction rules ensure that when the number of edges is far from k and the number of missing edges is far from k, we indeed have an $O(\sqrt{k})$-vertex kernel. Hence, in order to improve the size of the kernel, one only needs to take into account instances where the number of edges (or non-edges) is close to k.

Finally, one can easily show that Rule 4.1 can be adapted for the Clique + IS Editing problem by modifying the constant. On the other hand, it seems that Rules 4.3 and 4.4 do not readily generalize to Clique + IS Editing, therefore we were not able to obtain an $O(k/\log k)$-vertex kernel for this problem. However, it is an easy exercise to show that

we can weaken them in order to obtain a kernel with at most k/c vertices for any possible constant $c > 1$, at the cost of a running time in $O(n^c)$.

5 Linear-vertex kernels for (Pseudo-)Split Completion

The goal of this section is to prove Theorem 1.3, which we recall here.

▶ **Theorem 1.3.** *Split Completion and Split Deletion admit a kernel with at most* $11k + 6\sqrt{2k} + 4$ *vertices.*

Since the class of split graphs is closed under complementation, it is sufficient to prove that Theorem 1.3 holds for Split Completion.

We use the structure of the input graph to detect and label vertices that will be in the clique or the independent set part of a split decomposition of a well-chosen solution. More precisely, we show that, if the instance (G, k) is positive, the labeling constructed by our algorithm satisfies that there exists a solution F of (G, k) and a split decomposition (K^*, I^*) of $G + F$ such that all the vertices labeled as "clique" (resp. "independent set") are in K^* (resp. I^*). We then prove that if (G, k) is a positive instance, then the number of unlabeled vertices at the end of the algorithm is $O(k)$. Moreover, we show that we can reduce the number of labeled vertices to $O(k)$. Combining the above yields a linear kernel.

We present our reduction rules in Section 5.1 and prove their correctness and consequence in subsequent sections.

5.1 Labeling and reduction rules

Our algorithm keeps track of a partition (K, I, D) of $V(G)$, which corresponds to the labels of the vertices of G. The set K (resp. I) stands for the vertices already labeled "clique" (resp. "independent set") while D (for "do not know") contains the vertices that are not yet labeled. Initially, no vertex is labeled, hence $K = \emptyset, I = \emptyset$ and $D = V(G)$.

We will apply the following reduction rules, whose correctness is postponed in Section 5.2.

▶ **Rule 5.1** (I-rules). *Move $v \in D$ to I whenever at least one of the following holds:*

(a) *v has all of its neighbors in K,*

(b) *v is non-adjacent to at least $k + 1$ vertices of K.*

Notice that this rule applies to isolated vertices, since whenever v is isolated, $N(v) = \emptyset \subseteq K$.

We say that a vertex v *dominates* a vertex set X if $X \subseteq N[v]$.

▶ **Rule 5.2** (K-rules). *Move $v \in D$ to K whenever at least one of the following holds:*

(a) *v has a neighbor in I,*

(b) *$N(v)$ contains at least $k + 1$ non-edges,*

(c) *v dominates $K \cup D$.*

The following reduction rule simply ensures that K is a clique and I an independent set.

▶ **Rule 5.3.** *Apply one of the following operations as long as possible:*

(a) *if there is a non-edge e between vertices of K, then add e to $E(G)$ and decrease k by 1.*

(b) *if $k < 0$ then return a trivially negative instance.*

(c) *if there is an edge between vertices of I, then return a trivially negative instance.*

We apply these rules exhaustively, and stop when none can be applied. At each step, we remove a vertex from D or we add an edge to G. Then the algorithm stops after at most n^2 steps. Moreover, one can easily apply the rules in polynomial time. When none of the previous rules can be applied, we apply the following reduction rule.

▶ **Rule 5.4** (Delabeling rule). **(a)** *If K contains at least $k+1$ vertices, replace K by a set $K' = \{v'_1, \dots, v'_k\}$ of k vertices and denote by G' the resulting graph. Moreover, for each vertex $v \in D$, if v is non-adjacent to t vertices of K in G, then connect v to $v'_{t+1}, \dots, v'_k$ and do not connect it to $v_1, \dots, v_t$.*

(b) *Replace I by an independent set I' of $\sqrt{2k}$ vertices and make them completely adjacent to K', and not connected to D.*

With this rule, we can bound the number of vertices of the resulting graph by $|D|$ plus at most k vertices (for K'), plus at most $\sqrt{2k}$ vertices (for I'). Therefore, Theorem 1.3 boils down to the following lemma.

▶ **Lemma 5.5.** *If (G, k) is a positive instance, then $|D| \leq 10k + 5\sqrt{2k} + 4$.*

While it is not very difficult to prove that the reduction rules are safe, the main technical contribution of this section consists in proving Lemma 5.5. We divide the proof into two parts: we separately give a linear upper bound on the size of $D \cap K^*$ and $D \cap I^*$.

First, we prove that the number of vertices of D in the clique K^* of the solution is linear in k. Namely, we show that if too many vertices of K^* are in D, the neighborhood of many of them is not modified. Hence, one of them must be complete to $K \cup D$, a contradiction with Rule 5.2.

Arguing that the number of vertices of D in the independent set I^* of the solution is $O(k)$ is more involved. First note that if a vertex has an independent set of size larger than $O(\sqrt{k})$ in its neighborhood, it is added to K by Rule 5.2-b. Since D only contains $O(k)$ vertices in the clique, the number of vertices of D in the independent set is at most $O(k^{3/2})$. To obtain a better upper bound on the size of D, we carefully distinguish the size of the neighborhood of the vertices of $D \cap K^*$ in I^*. Very roughly, we prove that the number of vertices in $D \cap K^*$ with many neighbors in I^* is bounded by a sublinear function which permits to improve the size of the kernel. The proof of Lemma 5.5 is postponed to Section 5.3.

Lemma 5.5 together with Rule 5.4 ensure that the following reduction rule is correct, which completes the proof of Theorem 1.3:

▶ **Rule 5.6** (Final Rule). *If none of the previous rules can be applied, and the graph contains at least $11k + 6\sqrt{2k} + 5$ vertices, return a trivially negative instance.*

5.2 Correctness of the reduction rules

To analyze our algorithm, we study the evolution of the instance (G, k) with the partition $P = (K, I, D)$ after the application of each rule. We will refer to the tuple (G, k, P) as an *annotated instance* of Split Completion.

The following definition formalizes when a labeling of G is compatible with a solution F.

▶ **Definition 5.7.** *Let H be a graph, let F be a set of edges such that $H + F$ is a split graph, and let $P = (K, I, D)$ be a partition of $V(H)$. We say that P* is compatible with *F, and denote it $P \vDash F$, if there exists a split decomposition (K^*, I^*) of $H + F$ such that $K \subseteq K^*$ and $I \subseteq I^*$. In that case, we say that the decomposition (K^*, I^*)* witnesses *the fact that $P \vDash F$.*

An annotated instance comprises a graph along with a partial labeling of the vertices. Such an instance is positive when there exists a solution that is compatible with the labeling. This leads to the following definition.

▶ **Definition 5.8** (Positive annotated instance)**.** *An annotated instance (G, k, P) is positive if there exists a solution F of (G, k) such that $P \vDash F$.*

This allows us to extend safeness properties to reduction rules operating on annotated instances. We now show that the labeling and reduction rules preserve the existence of a solution.

▶ **Lemma 5.9.** *Rules 5.1 to 5.3 are safe.*

Proof. Let (G', k', P') be the instance obtained by applying one of the rules to an instance (G, k, P). We prove that (G, k, P) is a positive annotated instance if and only if (G', k', P') is. We first prove the direct implication. By hypothesis, there exists a solution F of (G, k) such that $P \vDash F$; and let (K^*, I^*) be a split decomposition of $G + F$ witnessing that P is compatible with F. We write $P' = (K', I', D')$ and $P = (K, I, D)$ and we consider three cases depending on the type of the rule that was applied.

- Rule 5.1. Assume that some vertex $v \in D$ is moved to I'. If $v \in I^*$, then $I' = I \cup \{v\} \subseteq I^*$ and $K' = K \subseteq K^*$, hence $P' \vDash F$. Therefore, we can assume that $v \in K^*$.
  If v is moved to I' in application of Rule 5.1-a, then, for every neighbor w of v in $G + F$, either $w \in K$ or $vw \in F$. Removing from F the edges incident to v provides a smaller solution F' such that $G' + F'$ is a split graph, and all the neighbors of v in $G' + F'$ are in the clique. Therefore, G' has a split decomposition where v is in the independent set, i.e., $P' \vDash F'$.
  If v is moved to I' in application of Rule 5.1-b, then v is non-adjacent to at least $k + 1$ vertices of $K \subseteq K^*$. Since K^* is a clique in $G + F$, all the edges between v and K must be added and then F must contain at least $k + 1$ edges. This is a contradiction, as $|F| \leq k$.
- Rule 5.2. Assume that some vertex $v \in D$ is moved to K'. If $v \in K^*$, then $K' = K \cup \{v\} \subseteq K^*$ and $I' = I \subseteq I^*$, hence $P' \vDash F$. Therefore, we can assume that $v \in I^*$.
  If v is moved to K' in application of Rule 5.2-a, then v has a neighbor u which belongs to $I \subseteq I^*$. Therefore, there is an edge uv between two vertices that must belong to the independent set. As we can only add edges, this is a contradiction.
  If v is moved to K' in application of Rule 5.2-b, then there are at least $k + 1$ non-edges in the graph induced by $N(v)$. Since $v \in I^*$ and I^* is an independent set, $N(v) \subseteq K^*$ must be in K^*. Since K^* is a clique in $G + F$, F must contain the $k + 1$ missing edges of $N(v)$, a contradiction with $|F| \leq k$.
  If v is moved to K' in application of Rule 5.2-c, then v is adjacent to $K \cup D \supseteq K^*$. Observe that $(K^* \cup \{v\}, I^* \setminus \{v\})$ is a split decomposition of $G' + F$. Moreover, since $K \subseteq K^*$ and $K' = K \cup \{v\}$, K' is a subset of $K^* \cup \{v\}$, hence $P' \vDash F$. Therefore, F is a solution of (G', k') and $P' \vDash F$, as desired.
- Rule 5.3. Assume that Rule 5.3-a was applied, i.e., some edge e has been added (the other cases are trivial since the instance (G, k, P) cannot be positive). No vertex has moved, therefore we have $P' = P$, and $k' = k - 1$. Since $K \subset K^*$ and the endpoints of e lie in K, e is an edge in $G + F$ but not in G, hence $e \in F$. Let $F' = F \setminus \{e\}$. By construction, F' is a solution of (G', k'). Since $G + F = G' + F'$, (K^*, I^*) is a split decomposition of $G' + F'$, and therefore $P' \vDash F'$.

We now prove the converse: assume that (G', k', P') is positive. In the case of Rule 5.1 and Rule 5.2, remark that any solution of (G', k') is a solution of (G, k), since the instance is

unchanged. Hence, if P' is compatible with a solution F' of (G', k'), then P is also compatible with F', as $K \subseteq K'$ and $I \subseteq I'$. Therefore, we have a solution F' of (G, k) that P respects, i.e., (G, k, P) is positive.

The case of Rule 5.3 follows by reversing the construction done in the same case of the other direction. ◀

Note that the initial labeling $P = (\emptyset, \emptyset, V(G))$ is compatible with every solution of (G, k) (if any). Therefore, by applying transitively Lemma 5.9, we get that the labeling and reduction process is safe.

We finally show that Rule 5.4 is safe.

▶ **Lemma 5.10.** *Rule 5.4 is safe.*

Proof. Let (G, k, P) be the annotated instance before Rule 5.4 is applied, and let (G', k') be the instance that it returns. Let us prove that there exists a solution F of (G, k) such that $P \vDash F$ if and only if (G', k') has a solution.

First, assume that there exists a solution F of (G, k) such that $P \vDash F$. Let (K^*, I^*) be a split decomposition of $G + F$ that witnesses the fact that $P \vDash F$. Let $K^D = K^* \cap D$ and $I^D = I^* \cap D$. We can extract from F a solution F' of (G, k) that only adds edges between vertices of K^*. Notice that, by construction, $K^* = K \sqcup K^D$.

Recall that (G', k') is obtained by replacing the subset of vertices K in G with a set K' of k vertices. Moreover, by construction, vertices of K^D that are non-adjacent to t vertices of K in G are non-adjacent to t vertices of K' in G'. Rule 5.1-b ensures that $t \leq k$, and Rule 5.3-a ensures that K induces a clique in G. Therefore, since K' induces a clique in G', the number of edges needed to turn $K^D \cup K$ into a clique is the same as the number of edges needed to turn $K^D \cup K'$ into a clique.

For every $u \in K^D$, let $uv_1, \ldots, uv_t$ be all the edges in F that are adjacent to u and some vertex $v_i \in K$. We construct a solution F' of (G', k') by replacing every such edge uv_i of F by uv_i', where $v_i' \in K'$ is defined in Rule 5.3-b. Since $k' = k$, (G', k') is also a positive instance.

Conversely, assume that (G', k') has a solution F'. Since every vertex of K' is adjacent to every vertex of I', for any split decomposition of (K^*, I^*) of $G' + F'$, we have $K' \subseteq K^*$. Otherwise, if some vertex of K' were in I^*, which would imply that F' contains more than k' edges, a contradiction. Indeed, I' is an independent set of size $\sqrt{2k'}$, and therefore, one requires more than k' edge additions to turn it into a clique. As vertices of I' are only adjacent to vertices of K', up to removing some edges from F', we can assume that $I' \subseteq I^*$.

Hence, by performing the same process as in the other direction in reverse, we can construct a solution F of (G, k) such that $G + F$ admits a clique decomposition (A, B) that satisfies $K \subseteq A$ and $I \subseteq B$. In other words, we have $P \vDash F$, which concludes the proof. ◀

5.3 Structure of positive instances

This section is devoted to the proof of Lemma 5.5, restated below.

▶ **Lemma 5.5.** *If (G, k) is a positive instance, then $|D| \leq 10k + 5\sqrt{2k} + 4$.*

In what follows, we assume that the input is a positive instance and the labeling/reduction process stopped and returned a generalized instance (G, k, P). In particular, Rules 5.1 to 5.3 cannot be applied. By Lemma 5.9, we get that there exists a solution F of (G, k) such that $|F| \leq k$ and $P \vDash F$. Unrolling the definition, this means that there exists a split decomposition (K^*, I^*) of $G + F$ such that $K \subseteq K^*$ and $I \subseteq I^*$. Let $K^D = D \cap K^*$ be the

set of unlabeled vertices that belong to the clique, and let $I^D = D \cap I^*$ be the set of the unlabeled vertices that belong to the independent set. For every $v \in D$, let $I_v = N(v) \cap I^D$. We give an upper bound on the cardinality of D by giving separate upper bounds on the respective cardinalities of K^D and I^D.

Before diving into the details of the proof, let us make two observations on the structure of D, that follow from the fact that the labeling rules cannot be applied.

▶ **Observation 5.11.** *For every vertex $v \in K^D$, $|I_v| \leq \sqrt{2k} + 1$.*

Proof. If $|I_v| > \sqrt{2k} + 1$, then $N(v)$ contains more than $\binom{\sqrt{2k}+1}{2} = k + \sqrt{k/2}$ non-edges. Hence, we can apply Rule 5.2-b, a contradiction. ◀

▶ **Observation 5.12.** *Every vertex $v \in I^D$ has a neighbor in K^D.*

Proof. A vertex $v \in I^D$ can only have neighbors in K and K^D. If v does not have neighbors in K^D, it only has neighbors in K, hence we can apply Rule 5.1-a, a contradiction. ◀

We first prove that $|K^D| = O(k)$.

▶ **Lemma 5.13.** *We have $|K^D| \leq 4k$.*

Proof. Let us prove this statement by contradiction: we prove that if $|K^D| \geq 4k + 1$, then there is a vertex in D that dominates $D \cup K$, which contradicts the fact that Rule 5.2-c cannot be applied.

By assumption, K^* is a clique in $G + F$. Since F contains at most k edges, there are at most $2k$ vertices of K^D that are adjacent to edges of F. Since $|K^D| \geq 4k + 1$, there are at least $2k + 1$ vertices in K^D that dominate $K^* = K^D \cup K$. Let X denote the set of such vertices. We will now show that there is a vertex in X that also dominates I^D, that is, a vertex of K^D that dominates $K^D \cup I^D \cup K = D \cup K$. To prove the existence of this vertex, we will prove that for any vertex u in X such that $I_u \neq I^D$, there exists a vertex $v \in X$ such that $|I_v| > |I_u|$. By applying this property repeatedly, we eventually find a vertex v such that $I_v = I^D$.

Let u be a vertex of X such that $I_u \neq I^D$. Since $K^D \subseteq N[u]$ and Rule 5.2-b cannot be applied, there are at most k non-edges between I_u and K^D. Hence, these non-edges are adjacent to at most k vertices of X (X being a clique, every non-edge is incident to at most one vertex of X), and then at least $k + 1$ vertices of X dominate I_u. Let X' be the subset of vertices of X that dominate $K^* \cup I_u$. Let w be a vertex of $I^D \setminus I_u$. As noted in Observation 5.12, w is adjacent to some vertex $v \in K^D$.

Assume that w is anticomplete to X', so that $v \notin X'$. Since $v \in K^*$, every vertex of X' is adjacent to v. Therefore v contains at least $k + 1$ non-edges in its neighborhood, namely the edges between w and X', a contradiction.

Therefore, $v \in X'$ and the conclusion follows since I_v contains I_u and w. ◀

By bounding locally the size of the neighborhood of each vertex in K^D using Observation 5.11, Lemma 5.13 directly provides an $O(k^{\frac{3}{2}})$ kernel. However, as we will show, this is not tight. Using a more global counting argument, we can show that $|I^D| = O(k)$.

▶ **Lemma 5.14.** *We have $|I^D| \leq 6k + 5\sqrt{2k} + 4$.*

Proof. First, notice that Observation 5.12 implies that $I^D \subseteq \bigcup_{v \in K^D} N(v)$. Therefore, if $|K^D| \leq \sqrt{8k}$, Observation 5.11 implies the following upper bound on the cardinality of I^D:

$$|I^D| \leq |K^D| \cdot (\sqrt{2k} + 1) \leq 4k + 2\sqrt{2k} \leq 6k + 5\sqrt{2k} + 4.$$

In what follows, we assume that $|K^D| > \sqrt{8k}$. We partition I^D into two sets: I^+, the set of vertices that have degree least $|K^D|/4$, i.e., vertices that are adjacent to at least $|K^D|/4$ vertices of K^D, and $I^- = I^D \setminus I^+$. We bound their sizes independently.

First, by counting the number n_e of edges between K^D and I^+ from the point of view of K^D, we get $n_e \leq |K^D| \cdot (\sqrt{2k}+1)$. From the point of view of I^+, we get $n_e \geq |K^D| \cdot |I^+|/4$. By combining the two inequalities, we get $|I^+| \leq 4(\sqrt{2k}+1)$.

It remains to show that $|I^-| \leqslant 6k + \sqrt{2k}$. To this end, we consider two types of vertices in K^D: those that are adjacent to more than $\sqrt{2k}$ edges of F in the solution, and the others. We then bound the number of vertices in I^- adjacent to (at least) a vertex of each type.

Since we add at most k edges to G, there are at most $\sqrt{2k}$ vertices in K^D incident to more than $\sqrt{2k}$ edges of F. By Observation 5.11, these vertices of K^D have at most $\sqrt{2k}(\sqrt{2k}+1) \leq 2k + \sqrt{2k}$ neighbors in I^D (and therefore in I^-).

To conclude the proof, it is thus sufficient to show that there are at most $4k$ vertices in I^- that are adjacent to vertices of K^D of the second type.

Let v be a vertex of K^D of the second type. We write $K_v = N(v) \cap K^D$ and $I_v^- = N(v) \cap I^-$. Observe that, by definition, $|K_v| \geqslant |K^D| - \sqrt{2k} \geq |K^D|/2$. Let $\bar{d}$ be the average degree in K_v of vertices in I_v^-. Since Rule 5.2-b cannot be applied, there are at least $|K_v| \cdot |I_v^-| - k$ edges between K_v and I_v^-, hence $\bar{d} \geqslant |K_v| - k/|I_v^-| \geqslant |K^D|/2 - k/|I_v^-|$. However, by definition of I^-, each vertex has degree at most $|K^D|/4$ in K_v hence $\bar{d} \leqslant |K^D|/4$. Combining the above yields $|I_v^-| \leqslant 4k/|K^D|$. Since there are at most $|K^D|$ vertices of the second type, the union of their neighborhoods has size at most $|K^D| \cdot 4k/|K^D| = 4k$, which is the sought result. ◀

5.4 Pseudo-split graphs

The proof of the kernel for Split Deletion can be easily adapted to give a kernel of similar size for Pseudo-split Deletion. Except for Rule 5.2-b where the $k+1$ should be replaced by $k+2$, Rules 5.1 to 5.4 are still safe, and their safeness can be proven with almost identical proofs. Moreover, Lemma 5.5 also extends (with a slightly worse but still linear bound), and we obtain the following result.

▶ **Theorem 5.15.** *Pseudo-split Deletion admits a $11k + 7\sqrt{2k} + 19$-vertex kernel.*

6 A quadratic-vertex kernel for Trivially Perfect Completion

The goal of this section is to prove Theorem 1.4 that we recall there:

▶ **Theorem 1.4.** *Trivially Perfect Completion admits a kernel with $2k^2 + 2k$ vertices.*

Recall that trivially perfect graphs are (C_4, P_4)-free graphs. We also have the following characterization.

▶ **Theorem 6.1** ([30, 32])**.** *H is a trivially perfect graph if and only if every connected induced subgraph of H contains a universal vertex.*

In what follows, we refer to induced P_4 or C_4 of a graph as its *obstructions*. We say that a pair (u, v) of vertices is a *diagonal* if $uv \notin E$ and there exists two vertices a and b such that $uavb$ is a P_4 or a C_4. Given a diagonal (u, v), the number of obstructions containing (u, v) is the number of distinct pairs (a, b) such that $uavb$ is a P_4 or a C_4. Note that every obstruction contains exactly two diagonals, and that any solution must contain at least one of the two diagonals of each obstruction.

We first present a reduction rule that should be applied exhaustively, and then two reduction rules that should be applied once.

▶ **Rule 6.2.** *Let u, v be two non-adjacent vertices. If the number of obstructions containing u, v is at least $k+1$, then add uv to E and decrease k by 1.*

▶ **Lemma 6.3.** *Rule 6.2 is safe.*

Proof. If there is no solution, then indeed, the reduced instance still has no solution. Assume now that (G, k) is positive. Let F be a solution of (G, k). If F contains uv the conclusion follows. Assume by contradiction that uv is not in F. Since every obstruction contains exactly two diagonals, for every obstruction containing u, v as a diagonal, the other diagonal is in F. Since the other diagonal consists of the other pair of vertices of the obstruction and there are at least $k+1$ disjoint pairs, F must a contain at least $k+1$ edges, a contradiction. ◀

Moreover, Rule 6.2 can easily be applied in polynomial time.

The *modulator* $X(G)$ of G is the subset of vertices of G that are in at least one obstruction. Guo [21, Theorem 4] stated without proof that the following rule is safe, i.e., that (G, k) is a positive instance if and only if $(G[X(G)], k)$ is.

▶ **Rule 6.4** (Guo [21])**.** *If $X(G) \neq V(G)$, remove all vertices of G that are not in $X(G)$.*

For completeness, we include a proof of the following lemma, which implies Guo's result.

▶ **Lemma 6.5.** *If $v \notin X(G)$, then* $\mathrm{opt}(G) = \mathrm{opt}(G - v)$.

Proof. It is trivial that $\mathrm{opt}(G - v) \leq \mathrm{opt}(G)$. For the other direction, let F be an optimal solution of $G - v$. We show that it is also a solution for G, i.e., $G + F$ is trivially perfect. Note that we can assume that G is connected (up to considering only the component of v).

Let H be a connected induced subgraph of $G + F$. According to Theorem 6.1, our goal is to show that H has a dominating vertex. If $v \notin V(H)$, then H is a connected subgraph of $G - v + F$, hence has a dominating vertex.

We may thus assume that $v \in V(H)$. If $H - v$ is not connected, then v must dominate H. Indeed, otherwise there is a P_4 $uvwx$ in H. Note that $F \subset E(G - v)$ hence $uv, vw \notin F$. Therefore we must have $wx \in F$. But since G is connected, it contains a shortest path of length at least 2 from v to x, and this yields either a P_4 or a C_4 containing v in G, a contradiction.

Finally, assume that $H - v$ is connected. Since $G - v + F$ is trivially perfect, $H - v$ contains a dominating vertex u. We may assume that $uv \notin E(G) \cup F$, otherwise we are done. Let u' be a neighbor of v in H. By construction $uu'v$ induce a P_3. We claim that u' dominates H, which concludes. Assume that this is not the case and there is $w \in V(H) \setminus N_H[u']$. Now $vu'uw$ is a P_4 in H, hence uu' or uw lie in F (again $u'v \notin F$ since $F \subset E(G - v)$). If $uu' \in F$ then u' and v are at distance at least two from u in G, hence a shortest path (in G) from $\{u', v\}$ to u yields a P_4 containing v in G. Similarly, if $uw \in F$, u, u', v are all at distance at least 2 from w in G and a shortest path from $\{u, u', v\}$ to w yields a P_4 containing v in G, a contradiction. ◀

When the first two rules cannot be applied, we perform the following rule which detects trivially negative instances.

▶ **Rule 6.6.** *If $|V(G)| > 2k^2 + 2k$, output a trivially negative instance.*

In order to complete our proof, we simply have to prove that after applying the first two rules exhaustively, the size of a positive instance is quadratic. The next lemma ensures that Rule 6.6 is safe, which concludes the proof of Theorem 1.4.

▶ **Lemma 6.7.** *If (G, k) is a positive instance and every diagonal belongs to at most k obstructions, then $|X(G)| \leq 2k^2 + 2k$.*

Proof. Since (G, k) is a positive instance, there exists a solution F containing at most k edges. Moreover every edge e of F belongs to at most k obstructions in G. Therefore, the number of vertices that are in an obstruction containing e as a diagonal is at most $2k + 2$ (since all the obstructions contain the endpoints of e).

Let x be a vertex of $X(G)$. Since all the vertices of $X(G)$ belong to at least one obstruction of G and F is a solution, F contains at least one of the two diagonals of some obstruction containing x. Therefore, we can map each vertex of $X(G)$ to an edge of F which is a diagonal of an obstruction containing x. The first part of the proof ensures that the number of vertices mapped to any edge of F is at most $2k + 2$, therefore the total size of $X(G)$ is at most $2k^2 + 2k$, which completes the proof. ◀

7 A linear-vertex kernel for Starforest Deletion

The goal of this section is to prove Theorem 1.5, which we recall here.

▶ **Theorem 1.5.** *Starforest Deletion admits a kernel with at most $4k + 2$ vertices.*

Stars can be divided into two sets: centers and leaves. Let us define the notion of *center set* of a star forest. We say that a set D of vertices of G is a *dominating set* of G if every vertex of G is either in D or adjacent to a vertex of D.

▶ **Definition 7.1** (Center set)**.** *Let $\mathbb{S}$ be a star-forest. A set $C^* \subseteq V(\mathbb{S})$ is a* center set *of $\mathbb{S}$ if C^* is a dominating set of $\mathbb{S}$ such that every star S of $\mathbb{S}$ contains exactly one vertex c of C^*. This vertex is called the center of S.*

Note that a center set is not necessarily unique since in 2-stars, both vertices can be selected as a center. Given a star forest $\mathbb{S}$ with a set of centers C^*, the *leaves* of $\mathbb{S}$ are the vertices outside of C^*. By definition, every leaf has degree 1 and its unique neighbor is in C^*.

In what follows, we show how to use the structure of the input graph to identify and label vertices that are centers of an optimal solution, which leads to a Label-And-Reduce kernelization algorithm.

Let (G, k) be an instance of Starforest Deletion. Our first reduction rule, which is indeed safe, removes trivial connected components.

▶ **Rule 7.2** (Clean-up rule)**.** *Remove from G any connected component with 1 or 2 vertices.*

Assume now that Rule 7.2 cannot be applied anymore.

▶ **Rule 7.3** (Center labeling rule)**.** *Let C be the set of vertices of G that are adjacent to a vertex of degree 1 in G.*

(a) *For every $v \notin C$, if v is adjacent to a vertex u of C, delete all the other edges between v and C, and decrease the parameter accordingly.*

(b) *For every $u, v \in C$, if u and v are adjacent then remove uv from G and decrease k by 1.*

The fact that Rule 7.3 is safe is a consequence of the following lemma:

▶ **Lemma 7.4.** *Let C be the set of vertices of G that are adjacent to a vertex of degree 1. If (G, k) is a positive instance, then there exists a solution F of (G, k) and a center-set C^* of $G - F$ such that $C \subseteq C^*$.*

Proof. Let F be a solution of (G, k) that maximizes the number of vertices of C in a center-set C^* of $G - F$. Let $\mathbb{S} = G - F$. Assume by contradiction that C is not included in C^*. We prove that there exists a solution F' of (G, k) and a center set C' of $G - F'$ containing more vertices from C than C^*, a contradiction.

Let $v \in C \setminus C^*$. Let w be a neighbor of v of degree 1 in G. Observe that v cannot have degree 1 in G, otherwise $\{v, w\}$ induces a 2-star in G. Therefore, v has degree at least 2 in G. However, it has degree 1 in $G - F$ since $v \notin C^*$. Notice that $w \in C^*$, as it is either isolated or adjacent to v in $G - F$, and $v \notin C^*$.

If w is the center of the star containing v, that star is reduced to $\{v, w\}$. Hence, we can replace w by v in C^*, which increases the size of $C \cap C^*$, a contradiction.

Otherwise, let $u \neq w$ be the center of the star of v. We set $F' = (F \setminus vw) \cup uv$ (see Figure 2). We have $|F'| \leq |F|$ and F' is still a solution of (G, k). Moreover, $G - F'$ has a center set C' containing $C^* \cap C$ and v, which contradicts the maximality of F. ◀

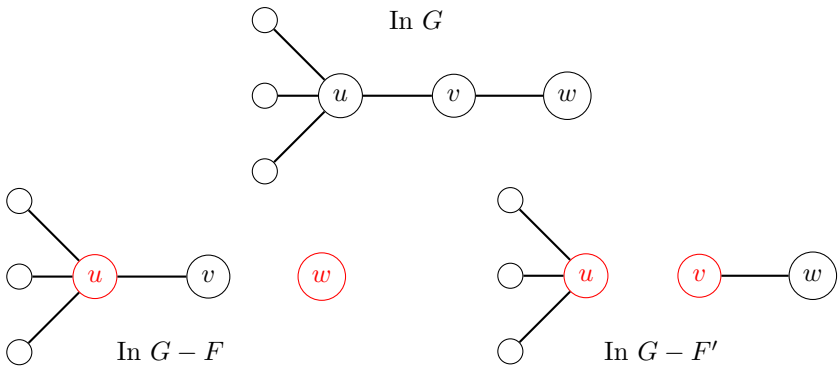

Figure 2 Illustration of the transformation used to make v a center, where v is a vertex adjacent to the degree-1 vertex w. Vertices of the center set (C^* in $G - F$, C' in $G - F'$) are drawn in red.

Lemma 7.4 ensures that Rule 7.3 is safe. Indeed, if there exists a solution, then there is also a solution where C is in the center-set. Hence we can safely remove all the edges between the vertices of C, since each star only contains one vertex of the center-set of $G - F$. Moreover, if an edge between v and a vertex w of C is kept in $G - F$, then we can choose to keep any other edge between v and C instead of (v, w), since all the vertices of C are centers of their stars.

When neither Rule 7.2 nor Rule 7.3 can be applied, we apply the following rule:

▶ **Rule 7.5** (Center reduction rule). *Merge all the vertices of C, and remove all but $k + 2$ vertices of degree 1.*

▶ **Lemma 7.6.** *Rule 7.5 is safe.*

Proof. Assume that Rule 7.3 cannot be applied. Let (G_0, k) be the instance before Rule 7.5 is applied, let (G_1, k) be the instance after vertices of C have been merged to the vertex c, and let (G_2, k) be the instance that Rule 7.5 returns. Since Rule 7.3 cannot be applied, there is no edge between vertices of C, and each vertex of G_0 is adjacent to at most one vertex of C. Therefore there are no loops nor parallel edges in G_1.

The instances (G_0, k) and (G_1, k) are equivalent. Indeed, by Lemma 7.4, if G_0 (resp. G_1) is positive, there exists a solution F (resp. F') such that C (resp. $\{c\}$) is in a center-set of $G_0 - F$ (resp. $G_1 - F'$). For every vertex v of degree 1 in G_0, let c_v be its only neighbor

(which is in C). We can then transform a solution of F of one instance, with a center set that contains C or c, into a solution of the other instance by swapping edges of the form (v, c) for edges (v, c_v). This operation does not change the cardinality of the solution, and the instances have the same parameter, therefore the instances are equivalent.

Let us finally prove that (G_1, k) is positive if and only if (G_2, k) is. Since G_2 is a subgraph of G_1, if (G_1, k) is positive then removing the same set of edges in G_2 also gives a solution.

Now, assume that (G_2, k) is positive. Let F be a solution of (G_2, k). Without loss of generality, c is adjacent to $k+2$ leaves in G_1 (otherwise $G_1 = G_2$, and the equivalence is trivial). As $|F| \leq k$, c is still adjacent to at least 2 leaves in $G_2 - F$, and therefore, c is the center of its star. This implies that the same set of edges F is a solution of G_1, since G_1 is obtained from G_2 by adding pendant vertices adjacent to c, and c is a center of any solution of G_2. ◀

When Rules 7.2 to 7.5 cannot be applied, we apply the following rule once.

▶ **Rule 7.7** (Kernel size rule)**.** *If $|V(G)| > 4k+3$, return a trivial negative instance. Otherwise, return (G, k).*

Rule 7.7 ensures that the returned kernel has at most $4k + 3$ vertices. In the remainder of this section, we study positive instances of Starforest Deletion to prove that Rule 7.7 is safe.

In the two following lemmas, we assume that none of Rules 7.2 to 7.5 can be applied. The following lemma uses the sparsity of starforests (they have many vertices of degree 1) to get information on the structure of positive instances.

▶ **Lemma 7.8.** *If (G, k) is a positive instance of Starforest Deletion with m edges, then G contains at least $m - 3k$ vertices of degree* 1*.*

Proof. Since (G, k) is a positive instance, there exists a set $F \subseteq E$ of size at most k and a starforest $\mathbb{S}$ such that $\mathbb{S} = G - F$. Let t denote the number of stars (i.e., of connected components) in $\mathbb{S}$. In each star of $\mathbb{S}$, there is at most one vertex which does not have degree 1. Therefore, $\mathbb{S}$ has at least $n - t$ vertices of degree 1. Let ℓ be the number of vertices in G that have degree 1. We can obtain G from $\mathbb{S}$ by adding at most k edges, and adding an edge can change the degree of at most two vertices. Therefore, we have $\ell \geq n - t - 2k$. Moreover, $\mathbb{S}$ has $n - t$ edges, hence $m \leq n - t + k$. By combining the above, we obtain that $\ell \geq n - 2k + m - n - k = m - 3k$. ◀

In the last step of Rule 7.3, we remove all but $k + 2$ vertices of degree 1. In the following lemma, we apply Lemma 7.8 to show that the number of remaining vertices must be small.

▶ **Lemma 7.9.** *If (G, k) is a positive instance where no rule can be applied, then $|V(G)| \leq 4k + 3$.*

Proof. By Lemma 7.8, G contains at least $m - 3k$ vertices of degree 1. Since Rule 7.2 cannot be applied, G does not contain 2-stars, and therefore each edge is incident to at most one vertex of degree 1. Hence, removing all vertices of degree 1 from G removes the same number of edges, i.e., we obtain a graph H with at most $3k$ edges. As all the degree 1 vertices of G are adjacent to a single vertex v, all vertices of H but v have degree at least 2. Hence, H contains at most $3k + 1$ vertices. Moreover, by Rule 7.5, G contains at most $k + 2$ vertices of degree 1. Therefore, we have $|V(G)| \leq 3k + 1 + k + 2 = 4k + 3$. ◀

By applying the contrapositive of Lemma 7.9, we get that Rule 7.7 is safe.

Improving the multiplicative constant in the linear bound. In the proof of Lemma 7.9, we use a simple argument based on the minimum degree to show that the $3k$ remaining edges span at most $3k + 1$ vertices. The worst case is when every vertex has degree 2, that is, when every connected component is a cycle. In a cycle, an optimal solution can easily be found in polynomial time, and therefore we can remove cycles. We can also show that long induced paths can be reduced. Combining these results gives a smaller kernel, at the cost of an increased running time and slightly more involved analysis.

However, these improvements do not yield a sublinear kernel. It turns out that, under the ETH, Starforest Deletion does not have a sublinear kernel. Indeed, Drange et al. [15] proved that, under the ETH, Starforest Deletion does not admit a subexponential FPT algorithm, i.e., an algorithm running in time $O^*(2^{o(k)})$. Moreover, there is an $O^*(2^n)$ algorithm for Starforest Deletion: for each subset S of vertices, test whether there exists a solution in which S is the center set. Therefore, a kernel with $o(k)$ vertices would imply an $O^*(2^{o(k)})$ algorithm; a contradiction.

References

1 Nikhil Bansal, Avrim Blum, and Shuchi Chawla. Correlation clustering. *Machine learning*, 56(1):89–113, 2004.

2 Amir Ben-Dor, Ron Shamir, and Zohar Yakhini. Clustering gene expression patterns. *Journal of computational biology*, 6(3-4):281–297, 1999.

3 Ivan Bliznets, Marek Cygan, Pawel Komosa, Lukáš Mach, and Michał Pilipczuk. Lower bounds for the parameterized complexity of minimum fill-in and other completion problems. In *Proceedings of the Twenty-Seventh Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 1132–1151. SIAM, 2016.

4 Sebastian Böcker. A golden ratio parameterized algorithm for cluster editing. *Journal of Discrete Algorithms*, 16:79–89, 2012.

5 Pablo Burzyn, Flavia Bonomo, and Guillermo Durán. NP-completeness results for edge modification problems. *Discrete Applied Mathematics*, 154(13):1824–1844, 2006.

6 Leizhen Cai. Fixed-parameter tractability of graph modification problems for hereditary properties. *Information Processing Letters*, 58(4):171–176, 1996.

7 Yixin Cao and Jianer Chen. Cluster editing: Kernelization based on edge cuts. *Algorithmica*, 64(1):152–169, 2012. doi:10.1007/s00453-011-9595-1.

8 Jianer Chen and Jie Meng. A $2k$ kernel for the cluster editing problem. *Journal of Computer and System Sciences*, 78(1):211–220, 2012.

9 Christophe Crespelle, Pål Grønås Drange, Fedor V. Fomin, and Petr A Golovach. A survey of parameterized algorithms and the complexity of edge modification. *arXiv preprint arXiv:2001.06867*, 2020.

10 Marek Cygan, Fedor V. Fomin, Łukasz Kowalik, Daniel Lokshtanov, Dániel Marx, Marcin Pilipczuk, Michał Pilipczuk, and Saket Saurabh. *Parameterized algorithms*, volume 5. Springer, 2015.

11 Peter Damaschke and Olof Mogren. Editing simple graphs. *Journal of Graph Algorithms and Applications*, 18(4):557–576, 2014. doi:10.7155/jgaa.00337.

12 Pål Grønås Drange. *Parameterized graph modification algorithms*. PhD thesis, University of Bergen, 2015.

13 Pål Grønås Drange, Fedor V. Fomin, Michał Pilipczuk, and Yngve Villanger. Exploring the subexponential complexity of completion problems. *ACM Transactions on Computation Theory (TOCT)*, 7(4):1–38, 2015.

14 Pål Grønås Drange and Michał Pilipczuk. A polynomial kernel for trivially perfect editing. *Algorithmica*, 80(12):3481–3524, 2018.

15 Pål Grønås Drange, Felix Reidl, Fernando Sánchez Villaamil, and Somnath Sikdar. Fast biclustering by dual parameterization. *arXiv preprint arXiv:1507.08158*, 2015.

16 Maël Dumas, Anthony Perez, and Ioan Todinca. A cubic vertex-kernel for trivially perfect editing. In *46th International Symposium on Mathematical Foundations of Computer Science, MFCS 2021, August 23-27, 2021, Tallinn, Estonia*, volume 202 of *LIPIcs*, pages 45:1–45:14. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2021. doi:10.4230/LIPIcs.MFCS.2021.45.

17 Fedor V. Fomin, Stefan Kratsch, Marcin Pilipczuk, Michał Pilipczuk, and Yngve Villanger. Tight bounds for parameterized complexity of cluster editing with a small number of clusters. *Journal of Computer and System Sciences*, 80(7):1430–1447, 2014.

18 Fedor V. Fomin, Daniel Lokshtanov, Saket Saurabh, and Meirav Zehavi. *Kernelization: theory of parameterized preprocessing*. Cambridge University Press, 2019.

19 Esha Ghosh, Sudeshna Kolay, Mrinal Kumar, Pranabendu Misra, Fahad Panolan, Ashutosh Rai, and MS Ramanujan. Faster parameterized algorithms for deletion to split graphs. *Algorithmica*, 71(4):989–1006, 2015.

20 Niels Grüttemeier and Christian Komusiewicz. On the relation of strong triadic closure and cluster deletion. *Algorithmica*, 82(4):853–880, 2020. doi:10.1007/s00453-019-00617-1.

21 Jiong Guo. Problem kernels for NP-complete edge deletion problems: Split and related graphs. In *International Symposium on Algorithms and Computation*, pages 915–926. Springer, 2007.

22 Peter L. Hammer and Bruno Simeone. The splittance of a graph. *Combinatorica*, 1(3):275–284, 1981.

23 Russell Impagliazzo and Ramamohan Paturi. On the complexity of k-SAT. *Journal of Computer and System Sciences*, 62(2):367–375, 2001.

24 Christian Komusiewicz and Johannes Uhlmann. Cluster editing with locally bounded modifications. *Discrete Applied Mathematics*, 160(15):2259–2270, 2012.

25 Athanasios L. Konstantinidis, Stavros D. Nikolopoulos, and Charis Papadopoulos. Strong triadic closure in cographs and graphs of low maximum degree. *Theoretical Computer Science*, 740:76–84, 2018. doi:10.1016/j.tcs.2018.05.012.

26 Yunlong Liu, Jianxin Wang, Jie You, Jianer Chen, and Yixin Cao. Edge deletion problems: Branching facilitated by modular decomposition. *Theoretical Computer Science*, 573:63–70, 2015.

27 Federico Mancini. *Graph modification problems related to graph classes*. PhD thesis, University of Bergen, 2008.

28 Assaf Natanzon, Ron Shamir, and Roded Sharan. Complexity classification of some edge modification problems. *Discrete Applied Mathematics*, 113(1):109–128, 2001.

29 René van Bevern, Vincent Froese, and Christian Komusiewicz. Parameterizing edge modification problems above lower bounds. *Theory of Computing Systems*, 62(3):739–770, 2018.

30 E. S. Wolk. The comparability graph of a tree. *Proceedings of the American Mathematical Society*, 13:789–795, 1962. doi:10.1090/S0002-9939-1962-0172273-0.

31 Zhenyu Wu and Richard Leahy. An optimal graph theoretic approach to data clustering: Theory and its application to image segmentation. *IEEE transactions on pattern analysis and machine intelligence*, 15(11):1101–1113, 1993.

32 Jing-Ho Yan, Jer-Jeong Chen, and Gerard Jennhwa Chang. Quasi-threshold graphs. *Discrete Applied Mathematics*, 69(3):247–255, 1996. doi:10.1016/0166-218X(96)00094-7.

33 Mihalis Yannakakis. Node-and edge-deletion NP-complete problems. In *Proceedings of the tenth annual ACM symposium on Theory of computing*, pages 253–264, 1978.