On the Parameterized Approximability of Contraction to Classes of Chordal Graphs

Spoorthy Gunda

Simon Fraser University, Burnaby, Canada
sgunda@sfu.ca

Pallavi Jain

Indian Institute of Technology Jodhpur, India
pallavi@iitj.ac.in

Daniel Lokshtanov

University of California, Santa Barbara, CA, USA
daniello@ucsb.edu

Saket Saurabh

The Institute of Mathematical Sciences, HBNI, Chennai, India
University of Bergen, Norway
saket@imsc.res.in

Prafullkumar Tale

Max Planck Institute for Informatics, Saarland Informatics Campus, Saarbrücken, Germany
prafullkumar.tale@mpi-inf.mpg.de

Abstract

A graph operation that *contracts edges* is one of the fundamental operations in the theory of graph minors. Parameterized Complexity of editing to a family of graphs by contracting k edges has recently gained substantial scientific attention, and several new results have been obtained. Some important families of graphs, namely the subfamilies of chordal graphs, in the context of edge contractions, have proven to be significantly difficult than one might expect. In this paper, we study the $\mathcal{F}$-CONTRACTION problem, where $\mathcal{F}$ is a subfamily of chordal graphs, in the realm of parameterized approximation. Formally, given a graph G and an integer k, $\mathcal{F}$-CONTRACTION asks whether there exists $X \subseteq E(G)$ such that $G/X \in \mathcal{F}$ and $|X| \leq k$. Here, G/X is the graph obtained from G by contracting edges in X. We obtain the following results for the $\mathcal{F}$-CONTRACTION problem.

- CLIQUE CONTRACTION is known to be FPT. However, unless NP $\subseteq$ coNP/poly, it does not admit a polynomial kernel. We show that it admits a polynomial-size approximate kernelization scheme (PSAKS). That is, it admits a $(1+\epsilon)$-approximate kernel with $\mathcal{O}(k^{f(\epsilon)})$ vertices for every $\epsilon > 0$.
- SPLIT CONTRACTION is known to be W[1]-Hard. We deconstruct this intractability result in two ways. Firstly, we give a $(2+\epsilon)$-approximate polynomial kernel for SPLIT CONTRACTION (which also implies a factor $(2+\epsilon)$-FPT-approximation algorithm for SPLIT CONTRACTION). Furthermore, we show that, assuming Gap-ETH, there is no $\left(\frac{5}{4}-\delta\right)$-FPT-approximation algorithm for SPLIT CONTRACTION. Here, $\epsilon, \delta > 0$ are fixed constants.
- CHORDAL CONTRACTION is known to be W[2]-Hard. We complement this result by observing that the existing W[2]-hardness reduction can be adapted to show that, assuming FPT$\neq$ W[1], there is no $F(k)$-FPT-approximation algorithm for CHORDAL CONTRACTION. Here, $F(k)$ is an arbitrary function depending on k alone.

We say that an algorithm is an $h(k)$-FPT-approximation algorithm for the $\mathcal{F}$-CONTRACTION problem, if it runs in FPT time, and on any input (G, k) such that there exists $X \subseteq E(G)$ satisfying $G/X \in \mathcal{F}$ and $|X| \leq k$, it outputs an edge set Y of size at most $h(k) \cdot k$ for which G/Y is in $\mathcal{F}$. We find it extremely interesting that three closely related problems have different behavior with respect to FPT-approximation.

2012 ACM Subject Classification Theory of computation → Approximation algorithms analysis

Keywords and phrases Graph Contraction, FPT-Approximation, Inapproximability, Lossy Kernels

Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques (APPROX/RANDOM 2020).
Editors: Jarosław Byrka and Raghu Meka; Article No. 51; pp. 51:1–51:19
Leibniz International Proceedings in Informatics
LIPICS Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

Digital Object Identifier 10.4230/LIPIcs.APPROX/RANDOM.2020.51

Category APPROX

Related Version A full version of the paper can be found here `https://arxiv.org/abs/2006.10364`, [31].

Funding *Saket Saurabh*: This project has received funding from the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation programme (grant agreement No 819416), and Swarnajayanti Fellowship (No DST/SJF/MSA01/2017-18).
Prafullkumar Tale: This research is a part of a project that has received funding from the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation programme under grant agreement SYSTEMATICGRAPH (No. 725978). Most part of this project was completed when the author was a Senior Research Fellow at The Institute of Mathematical Sciences, HBNI, Chennai, India.

1 Introduction

Graph modification problems have been extensively studied since the inception of Parameterized Complexity in the early '90s. The input of a typical graph modification problem consists of a graph G and a positive integer k, and the objective is to edit k vertices (or edges) so that the resulting graph belongs to some particular family, $\mathcal{F}$, of graphs. These problems are not only mathematically and structurally challenging, but have also led to the discovery of several important techniques in the field of Parameterized Complexity. It would be completely appropriate to say that solutions to these problems played a central role in the growth of the field. In fact, just in the last few years, parameterized algorithms have been developed for several graph editing problems [12, 10, 11, 9, 5, 6, 26, 24, 17, 18, 19, 27]. The focus of all of these papers and the vast majority of papers on parameterized graph editing problems has so far been limited to edit operations that delete vertices, delete edges or add edges.

In recent years, a different edit operation has begun to attract significant scientific attention. This operation, which is arguably the most natural edit operation apart from deletions/insertions of vertices/edges, is the one that contracts an edge. Here, given an edge uv that exists in the input graph, we remove the edge from the graph and merge its two endpoints. Edge contraction is a fundamental operation in the theory of graph minors. For some particular family of graphs, $\mathcal{F}$, we say that a graph G belongs to $\mathcal{F} + kv$, $\mathcal{F} + ke$ or $\mathcal{F} - ke$ if some graph in $\mathcal{F}$ can be obtained by deleting at most k vertices from G, deleting at most k edges from G or adding at most k edges to G, respectively. Using this terminology, we say that a graph G belongs to $\mathcal{F}|ke$ if some graph in $\mathcal{F}$ can be obtained by contracting at most k edges in G. In this paper, we study the following problem.

> $\mathcal{F}$-CONTRACTION **Parameter:** k
> **Input:** A graph G and an integer k
> **Question:** Does G belong to $\mathcal{F}|ke$?

For several families of graphs $\mathcal{F}$, early papers by Watanabe et al. [46, 47], and Asano and Hirata [3] showed that $\mathcal{F}$-EDGE CONTRACTION is NP-complete.

In the framework of Parameterized Complexity, these problems exhibit properties that are quite different from those problems where we only delete or add vertices and edges. Indeed, a well-known result by Cai [7] states that in case $\mathcal{F}$ is a hereditary family of graphs with a finite set of forbidden induced subgraphs, then the graph modification problems, $\mathcal{F}+kv$, $\mathcal{F}+ke$ or $\mathcal{F}-ke$, defined by $\mathcal{F}$ admits a simple FPT algorithm (an algorithm with running time $f(k)n^{\mathcal{O}(1)}$). However, for $\mathcal{F}$-CONTRACTION, the result by Cai [7] does not hold. In particular, Lokshtanov et al. [38] and Cai and Guo [8] independently showed that if $\mathcal{F}$ is either the family of P_ℓ-free graphs for some $\ell \geq 5$ or the family of C_ℓ-free graphs for some $\ell \geq 4$, then $\mathcal{F}$-CONTRACTION is W[2]-Hard (W[i]-hardness, for $i \geq 1$, is an analogue to NP-hardness in Parameterized Complexity, and is used to rule out FPT-algorithm for the problem) when parameterized by k (the number of edges to be contracted). These results immediately imply that CHORDAL CONTRACTION is W[2]-Hard when parameterized by k. The parameterized hardness result for CHORDAL CONTRACTION led to finding subfamilies of chordal graphs, where the problem could be shown to be FPT. Two subfamilies that have been considered in the literature are families of *split graphs* and *cliques*. Cai and Guo [8] showed that CLIQUE CONTRACTION is FPT, however, it does not admit a polynomial kernel. Later, Cai and Guo [32] also claimed to design an algorithm that solves SPLIT CONTRACTION in time $2^{\mathcal{O}(k^2)} \cdot n^{\mathcal{O}(1)}$, which proves that the problem is FPT. However, Agrawal et al. [2] found an error with the proof and showed that SPLIT CONTRACTION is W[1]-Hard.

Inspired by the intractable results that CHORDAL CONTRACTION, SPLIT CONTRACTION and CLIQUE CONTRACTION are W[2]-Hard, W[1]-Hard, and does not admit polynomial kernel, respectively, we study them from the viewpoint of parameterized approximation.

Our Results and Methods. We start by defining a few basic definitions in parameterized approximation. To formally define these, we need a notion of parameterized optimization problems. We defer formal definitions to Section 2 and give intuitive definitions here. We say that an algorithm is an $h(k)$-FPT-approximation algorithm for the $\mathcal{F}$-CONTRACTION problem, if it runs in FPT time, and on any input (G,k) if there exists $X \subseteq E(G)$ such that $G/X \in \mathcal{F}$ and $|X| \leq k$, it outputs an edge set Y of size at most $h(k) \cdot k$ and $G/Y \in \mathcal{F}$. Let $\alpha \geq 1$ be a real number. We now give an informal definition of α-approximate kernels. The kernelization algorithm takes an instance I with parameter k, runs in polynomial time, and produces a new instance I' with parameter k'. Both k' and the size of I' should be bounded in terms of just the parameter k. That is, there exists a function $g(k)$ such that $|I'| \leq g(k)$ and $k' \leq g(k)$. This function $g(k)$ is called the *size* of the kernel. For minimization problems, we also require the following from α-approximate kernels: For every $c \geq 1$, a c-approximate solution S' to I' can be transformed in polynomial time into a $(c \cdot \alpha)$-approximate solution S to I. However, if the quality of S' is "worse than" k', or $(c \cdot \alpha) \cdot OPT(I) > k$, the algorithm that transforms S' into S is allowed to fail. Here, $OPT(I)$ is the value of the optimum solution of the instance I.

Our first result is about CLIQUE CONTRACTION. It is known to be FPT. However, unless NP $\subseteq$ coNP/poly, it does not admit a polynomial kernel [8]. We show that it admits a PSAKS. That is, it admits a $(1+\epsilon)$-approximate polynomial kernel with $\mathcal{O}(k^{f(\epsilon)})$ vertices for every $\epsilon > 0$. In particular, we obtain the following result.

▶ **Theorem 1.** *For any $\epsilon > 0$, CLIQUE CONTRACTION parameterized by the size of solution k, admits a time efficient $(1+\epsilon)$-approximate polynomial kernel with $\mathcal{O}(k^{d+1})$ vertices, where $d = \lceil \frac{1}{\epsilon} \rceil$.*

Overview of the proof of Theorem 1. Let us fix an input (G, k) and a constant $\epsilon > 0$. Given a graph G, contracting edges of G to get into a graph class $\mathcal{F}$ is same as partitioning the vertex set $V(G)$ into connected sets, $W_1, W_2, \ldots, W_\ell$, and then contracting each connected set to a vertex. These connected sets are called *witness sets*. A witness set W_i is called *non-trivial*, if $|W_i| \geq 2$, and *trivial* otherwise.

Observe that if a graph G can be transformed into a clique by contracting edges in F, then G can also be converted into a clique by deleting all the endpoints of edges in F. This observation implies that if G is *k-contractible* to a clique, then there exists an induced clique of size at least $|V(G)| - 2k$. Let I be a set of vertices in G, which induces this large clique and let $C = V(G) \setminus I$. Observe that C forms a vertex cover in the graph $\overline{G}$ (graph with vertex set $V(G)$ and those edges that are not present in $E(G)$). Using a factor 2-approximation algorithm, we find a vertex cover X of $\overline{G}$. Let $Y = V(\overline{G}) - X$ be an independent set in $\overline{G}$. If $|X| > 4k$, we immediately say No. Now, suppose that we have some solution and let $W_1, W_2, \ldots, W_\ell$ be those witness sets that are either non-trivial or contained in X. Now, let us say that a set W_i is *nice* if it has at least one vertex outside X, and *small* if it contains less than $\mathcal{O}(1/\epsilon)$ vertices. A set that is not *small* is *large*. Observe that there exists a $(1 + \epsilon)$-approximate solution where *the only sets* that are not nice are small. Also, observe that all nice sets are adjacent. Now, we classify all subsets of X of size at most $\mathcal{O}(1/\epsilon)$ as *possible* and *impossible* small witness sets. Notice that if a set $A \subseteq X$ has more than $2k$ non-neighbors, then it can not possibly be a witness set, as one of these non-neighbors will be a trivial witness set. Now for every set, $A \subseteq X$ of size at most $\mathcal{O}(1/\epsilon)$ mark all of its non-neighbors, but if there are more than $2k$, then mark $2k + 1$ of them. Now, look at an unmarked vertex in Y, the only reason it could still be relevant if it is part of some W_i. So its job is (a) connecting the vertices in W_i, or (b) potentially being the vertex in Y that is making some W_i nice, or (c) it is a neighbor to *all* the small (not nice) subsets of X in the solution. Now notice that any vertex in Y that is unmarked does jobs (b) and (c) equally well. So we only need to care about connectivity. Look at some nice and small set W_i; we only need to preserve the neighborhoods of the vertices of Y into W_i. For every subset of size $\mathcal{O}(1/\epsilon)$, we keep one vertex in Y that has that set in its neighborhood. Notice that we do not care that different W_i's use different marked vertices for connectivity because merging two W_i's is more profitable for us. Finally, we delete all unmarked vertices and obtain an $(1 + \epsilon)$-approximate kernel of size roughly $k^{\mathcal{O}(1/\epsilon)}$. We argue that this kernelization algorithm is time efficient i.e. the running time is polynomial in the size of an input and the constant in the exponent is independent of ϵ. This completes the overview of the proof for Theorem 1. Next, we move to Split Contraction.

Split Contraction is known to be W[1]-Hard [2]. We ask ourselves whether Split Contraction is *completely* FPT*-inapproximable* or admits an α-FPT-approximation algorithm, for some fixed constant $\alpha > 0$. We obtain two results towards our goal.

▶ **Theorem 2.** *For every $\epsilon > 0$, Split Contraction admits a factor $(2 + \epsilon)$-FPT-approximation algorithm. In fact, for any $\epsilon > 0$, Split Contraction admits a $(2 + \epsilon)$-approximate kernel with $\mathcal{O}(k^{f(\epsilon)})$ vertices.*

Given, Theorem 2, it is natural to ask whether Split Contraction admits a factor $(1 + \epsilon)$-FPT-approximation algorithm, for every $\epsilon > 0$. We show that this is not true and obtain the following hardness result.

▶ **Theorem 3.** *Assuming Gap-ETH, no FPT time algorithm can approximate Split Contraction within a factor of $\left(\frac{5}{4} - \delta\right)$, for any fixed constant $\delta > 0$.*

Overview of the proofs of Theorems 2 and 3. Our proof for Theorem 2 uses ideas for $(1+\epsilon)$-approximate kernel for Clique Contraction (Theorem 1) and thus we omit its overview. Towards the proof of Theorem 3, we give a gap preserving reduction from a variant of the Densest-k-Subgraph problem (given a graph G and an integer k, find a subset $S \subseteq V(G)$ of k vertices that induces maximum number of edges). Chalermsook et al. [13] showed that, assuming Gap-ETH[1], for any $g = o(1)$, there is no FPT-time algorithm that, given an integer k and any graph G on n vertices that contains at least one k-clique, always output $S \subseteq V(G)$, of size k, such that $\mathsf{Den}(S) \geq k^{-g(k)}$. Here, $\mathsf{Den}(S) = |E(G[S])|/\binom{|S|}{2}$. We need a strengthening of this result that says that assuming Gap-ETH, for any $g = o(1)$ and for any constant $\alpha > 1$, there is no FPT-time algorithm that, given an integer k and any graph G on n vertices that contains at least one k-clique, always outputs $S \subseteq V(G)$, of size αk, such that $\mathsf{Den}(S) \geq k^{-g(k)}$. Starting from this result, we give a gap-preserving reduction to Split Contraction that takes FPT time and obtain Theorem 3. Given an instance (G, k) of Densest-k-Subgraph, we first use color coding to partition the edges into $t = \binom{k}{2}$ color classes such that every color class contains exactly one edge of a "densest subgraph" (or a clique). For each color class we make one *edge selection gadget.* Each edge selection gadget corresponding to the color class j consists of an independent set $\mathtt{ES}_j$ that contains a vertex corresponding to each edge in the color class$j$, and a *cap vertex* g_j that is adjacent to every vertex in $\mathtt{ES}_j$. Next, we add a sufficiently large clique Z of size $\rho \cdot |V(G)|$, where for every vertex $v \in V(G)$, we have ρ vertices. Every vertex in an edge selection gadget is adjacent to every vertex of Z, except those corresponding to the endpoints of the edge the vertex represents. Finally, we add a clique $\mathtt{SV}$ of size t that has one vertex s_j for each edge selection gadget. Make the vertex s_j adjacent to every vertex in $\mathtt{ES}_j$. We also add sufficient guards on vertices everywhere, so that "unwanted" contractions do not happen. The idea of the reduction is to contract edges in a way that the vertices in $\mathtt{SV}$, Z, and g_j, $j \in \{1, \ldots, t\}$, become a giant clique and other vertices become part of an independent set, resulting in a split graph. Towards this we first use $2t$ contractions so that g_j, s_j, and a vertex $a_j \in \mathtt{ES}_j$ are contracted into one. One way to ensure that they form a clique along with Z is to contract each of them to a vertex in Z. However, this will again require t edge contractions. We set our budget in a way that this is not possible. Thus, what we need is to destroy the non-neighbors of a_j. One way to do this again will be to match the vertices obtained after the first round of $2t$ contractions in a way that there are no non-adjacencies left. However, this will also cost $t/2$, and our budget does not allow this. The other option (which we take) is to take the union of all non-neighbors of a_j, say N, and contract each of them to one of the vertex in $Z \setminus N$. Observe that to minimize the contractions to get rid of non-neighbors of a_j, we would like to minimize $|N|$. This will happen when N spans a large number of edges. Thus, it precisely captures the Densest-k-Subgraph problem. The budget is chosen in a way that we get the desired gap-preserving reduction, which enables us to prove Theorem 3.

Our final result concerns Chordal Contraction. Lokshtanov et al. [38] showed that Chordal Contraction is W[2]-Hard. We observe that the existing W[2]-hardness reduction can be adapted to show the following theorem.

▶ **Theorem 4.** *Assuming* FPT$\neq$ *W[1], no* FPT *time algorithm can approximate Chordal Contraction within a factor of $F(k)$. Here, $F(k)$ is a function depending on k alone.*

[1] We refer the readers to [13] for the definition of Gap-ETH and related terms.

Overview of the proof of Theorem 4. Towards proving Theorem 4, we give a 1-approximate polynomial parameter transformation (1-appt) from Set Cover (given a universe U, a family of subsets $\mathcal{S}$, and an integer k, we shall decide the existence of a subfamily of size k that contains all the elements of U) to Chordal Contraction. That is, given any solution of size at most ℓ for Chordal Contraction, we can transform this into a solution for Set Cover of size at most ℓ. Karthik et al. [35] showed that assuming FPT$\neq$ W[1], no FPT time algorithm can approximate Set Cover within a factor of $F(k)$. Pipelining this result with our reduction we get Theorem 4.

Related Work. To the best of our knowledge, Heggernes et al. [33] was the first to explicitly study $\mathcal{F}$-Contraction from the viewpoint of Parameterized Complexity. They showed that in case $\mathcal{F}$ is the family of trees, $\mathcal{F}$-Contraction is FPT but does not admit a polynomial kernel, while in case $\mathcal{F}$ is the family of paths, the corresponding problem admits a faster algorithm and an $\mathcal{O}(k)$-vertex kernel. Golovach et al. [28] proved that if $\mathcal{F}$ is the family of planar graphs, then $\mathcal{F}$-Contraction is again FPT. Moreover, Cai and Guo [8] showed that in case $\mathcal{F}$ is the family of cliques, $\mathcal{F}$-Contraction is solvable in time $2^{\mathcal{O}(k \log k)} \cdot n^{\mathcal{O}(1)}$, while in case $\mathcal{F}$ is the family of chordal graphs, the problem is W[2]-Hard. Heggernes et al. [34] developed an FPT algorithm for the case where $\mathcal{F}$ is the family of bipartite graphs. Later, a faster algorithm was proposed by Guillemot and Marx [30].

Pioneering work of Lokshtanov et al. [40] on the approximate kernel is being followed by a series of papers generalizing/improving results mentioned in this work and establishing lossy kernels for various other problems. Lossy kernels for some variations of Connected Vertex Cover [21, 36], Connected Feedback Vertex Set [43], Steiner Tree [20] and Dominating Set [22, 44] have been established (also see [41, 45]). Krithika et al. [37] were first to study graph contraction problems from the lenses of lossy kernelization. They proved that for any $\alpha > 1$, Tree Contraction admits an α-lossy kernel with $\mathcal{O}(k^d)$ vertices, where $d = \lceil \alpha/(\alpha - 1) \rceil$. Agarwal et al. [1] proved similar result for $\mathcal{F}$-Contraction problems where graph class $\mathcal{F}$ is defined in parametric way from set of trees. Eiben et al. [21] obtained similar result for Connected $\mathcal{H}$-Hitting Set problem.

Organization. Due to space constraints, we omit some of the results from this extended abstract. We present the notations and preliminaries in Section 2 and in Section 3 we give the $(1 + \epsilon)$–approximate polynomial kernel for Clique Contraction.

2 Preliminaries

In this section, we give notations and definitions that we use throughout the paper. Unless specified, we will be using all general graph terminologies from the book of Diestel [15].

2.1 Graph Theoretic Definitions and Notations

For an undirected graph G, sets $V(G)$ and $E(G)$ denote the set of vertices and edges, respectively. Two vertices u, v in $V(G)$ are said to be *adjacent* if there is an edge uv in $E(G)$. The neighborhood of a vertex v, denoted by $N_G(v)$, is the set of vertices adjacent to v in G. For subset S of vertices, we define $N(S) = \bigcup_{v \in S} N(v)) \setminus S$. The subscript in the notation for the neighborhood is omitted if the graph under consideration is clear. For a set of edges F, set $V(F)$ denotes the endpoints of edges in F. For a subset S of $V(G)$, we denote the graph obtained by deleting S from G by $G - S$ and the subgraph of G induced on set S by $G[S]$. For two subsets S_1, S_2 of $V(G)$, we say S_1, S_2 *are adjacent if there exists an edge* with one endpoint in S_1 and other in S_2.

An edge e in G is a *chord* of a cycle C (resp. path P) if (i) both the endpoints of e are in C (resp. in P), and (ii) edge e is not in C (resp. not in P). An *induced cycle* (resp. *path*) is a cycle (resp. path) which has no chord. We denote induced cycle and path on ℓ vertices by C_ℓ and P_ℓ, respectively. A *complete graph* G is an undirected graph in which for every pair of vertices $u, v \in V(G)$, there is an edge uv in $E(G)$. As an immediate consequence of definition we get the following.

▶ **Lemma 5.** *A connected graph G is complete if and only if G does not contain an induced P_3.*

A *clique* is a subset of vertices in the graph that induces a complete graph. A set $I \subseteq V(G)$ of pairwise non-adjacent vertices is called an *independent set.* A graph G is a *split graph* if $V(G)$ can be partitioned into a clique and an independent set. For split graph G, partition (X, Y) is *split partition* if X is a clique and Y is an independent set. In this article, whenever we mention a split partition, we first mention the clique followed by the independent set. We will also use the following well-known characterization of split graphs. Let, $2K_2$ be a graph induced on four vertices, which contains exactly two edges and no isolated vertices.

▶ **Lemma 6** ([29])**.** *A graph G is a split graph if and only if it does not contain C_4, C_5 or $2K_2$ as an induced subgraph.*

A graph G is *chordal* if every induced cycle in G is a triangle; equivalently, if every cycle of length at least four has a chord. A vertex subset $S \subseteq V(G)$ is said to *cover* an edge $uv \in E(G)$ if $S \cap \{u, v\} \neq \emptyset$. A vertex subset $S \subseteq V(G)$ is called a *vertex cover* in G if it covers all the edges in G.

We start with the following observation, which is useful to find a large induced clique in the input graph. The *complement* of G, denoted by $\overline{G}$, is a graph whose vertex set is $V(G)$ and edge set is precisely those edges which are not present in $E(G)$. Note that given a graph G, if S is a set of vertices such that $G - S$ is a clique, then S is a vertex cover in the complement graphs of G, denoted by $\overline{G}$, as $\overline{G} - S$ is edgeless. Using the well-known factor 2-approximation algorithm for Vertex Cover [4], we have following.

▶ **Observation 2.1** ([4])**.** *There is a factor 2-approximation algorithm to compute a set of vertices whose deletion results in a complete graph.*

Using, Lemma 6 one can obtain a simple factor 5-approximation algorithm for deleting vertices to get a split graph.

▶ **Observation 2.2.** *There is a factor 5-approximation algorithm to compute a set of vertices whose deletion results in a split graph.*

Recently, for every $\epsilon > 0$, a factor $(2 + \epsilon)$-approximation algorithm for deleting vertices to get a split graph has been obtained [39]. However, for our purposes Observation 2.2 will suffice.

2.2 Graph Contraction

The *contraction* of edge $e = uv$ in G deletes vertices u and v from G, and adds a new vertex, which is made adjacent to vertices that were adjacent to either u or v. Any parallel edges added in the process are deleted so that the graph remains simple. The resulting graph is denoted by G/e. Formally, for a given graph G and edge $e = uv$, we define G/e in the following way: $V(G/e) = (V(G) \cup \{w\}) \setminus \{u, v\}$ and $E(G/e) = \{xy \mid x, y \in V(G) \setminus \{u, v\}, xy \in E(G)\} \cup \{wx | \; x \in N_G(u) \cup N_G(v)\}$. For a subset of edges F in G, graph G/F denotes the graph obtained from G by repeatedly contracting edges in F until no such edge remains. We say that a graph G is *contractible* to a graph H if there exists an onto function $\psi : V(G) \to V(H)$ such that the following properties hold.

- For any vertex h in $V(H)$, graph $G[W(h)]$ is connected, where set $W(h) := \{v \in V(G) \mid \psi(v) = h\}$.
- For any two vertices h, h' in $V(H)$, edge hh' is present in H if and only if there exists an edge in G with one endpoint in $W(h)$ and another in $W(h')$.

For a vertex h in H, set $W(h)$ is called a *witness set* associated with h. We define H-*witness structure* of G, denoted by $\mathcal{W}$, as collection of all witness sets. Formally, $\mathcal{W} = \{W(h) \mid h \in V(H)\}$. Witness structure $\mathcal{W}$ is a partition of vertices in G, where each witness forms a connected set in G. Recall that if a *witness set* contains more than one vertex, then we call it *non-trivial* witness set, otherwise a *trivial* witness set.

If graph G has a H-witness structure, then graph H can be obtained from G by a series of edge contractions. For a fixed H-witness structure, let F be the union of spanning trees of all witness sets. By convention, the spanning tree of a singleton set is an empty set. Thus, to obtain H from G, it is *sufficient* to contract edges in F. If such witness structure exists, then we say that graph G is contractible to H. We say that graph G is k-*contractible* to H if cardinality of F is at most k. In other words, H can be obtained from G by at most k edge contractions. Following observation is an immediate consequence of definitions.

▶ **Observation 2.3.** *If graph G is k-contractible to graph H, then the following statements are true.*

- *For any witness set W in a H-witness structure of G, the cardinality of W is at most $k+1$.*
- *For a fixed H-witness structure, the number of vertices in G, which are contained in non-trivial witness sets is at most $2k$.*

In the following two observations, we state that if a graph can be transformed into a clique or a split graph by contracting few edges, then it can also be converted into a clique or split graph by deleting few vertices.

▶ **Observation 2.4.** *If a graph G is k-contractible to a clique, then G can be converted into a clique by deleting at most $2k$ vertices.*

Proof. Let F be a set of edges of size at most k such that G/F is a clique. Let $\mathcal{W}$ be a G/F-witness structure of G. Let X be a set of all vertices which are contained in the non-trivial witness sets in $\mathcal{W}$. By Observation 2.3, size of X is at most $2k$. Any two vertices in $V(G) \setminus X$ are adjacent to each other as these vertices form singleton sets, which are adjacent in G/F. Hence, G can be converted into a clique by deleting vertices in X. ◀

▶ **Observation 2.5.** *If a graph G is k-contractible to a split graph then G can be converted into a split graph by deleting at most $2k$ vertices.*

Proof. For graph G, let F be the set of edges such that G/F is a split graph and $|F| \leq k$. Let $V(F)$ be the collection of all endpoints of edges in F. Since cardinality of F is at most k, $|V(F)|$ is at most $2k$. We argue that $G - V(F)$ is a split graph. For the sake of contradiction, assume that $G - V(F)$ is not a split graph. We know that a graph is split if and only if it does not contain induced C_4, C_5 or $2K_2$. This implies that there exists a set of vertices V' in $V(G) \setminus V(F)$ such that $G[V']$ is either C_4, C_5 or $2K_2$. Since no edge in F is incident on any vertices in V', $G/F[V']$ is isomorphic to $G[V']$. Hence, there exists a C_4, C_5 or $2K_2$ in G/F contradicting the fact that G/F is a split graph. Hence, our assumption is wrong and $G - V(F)$ is a split graph. ◀

Consider a connected graph G which is k-contractible to the clique K_ℓ. Let $\mathcal{W}$ be a K_ℓ-witness structure of G. The following observation gives a sufficient condition for obtaining a witness structure of an induced subgraph of G from $\mathcal{W}$.

▶ **Observation 2.6.** *Let $\mathcal{W}$ be a clique witness structure of G. If there exists two different witness sets $W(t_1), W(t_2)$ in $\mathcal{W}$ and a vertex v in $W(t_1)$ such that the set $W(t) = (W(t_1) \cup W(t_2)) \setminus \{v\}$ is a connected set in $G - \{v\}$, then $\mathcal{W}'$ is a clique witness structure of $G - \{v\}$, where $\mathcal{W}'$ is obtained from $\mathcal{W}$ by removing $W(t_1), W(t_2)$ and adding $W(t)$.*

Proof. Let $G' = G - \{v\}$. Note that $\mathcal{W}'$ is a partition of vertices in G'. Any set in $\mathcal{W}' \setminus \{W(t)\}$ is a witness set in $\mathcal{W}$ and does not contain v. Hence, these sets are connected in G'. Since $G'[W(t)]$ is also connected, all the witness sets in $\mathcal{W}'$ are connected in G'.

Consider any two witness sets $W(t'), W(t'')$ in $\mathcal{W}'$. If none of these two is equal to $W(t)$ then both of these sets are present in $\mathcal{W}$. Since none of these witness sets contains vertex v, they are adjacent to each other in G'. Now, consider a case when one of them, say $W(t'')$, is equal to $W(t)$. As witness sets $W(t')$ and $W(t_2)$ are present in $\mathcal{W}$, there exists an edge with one endpoint in $W(t')$ and another in $W(t_2)$. The same edge is present in graph G' as it is not incident on v. Since $W(t_2)$ is subset of $W(t)$, sets $W(t')$ and $W(t)$ are adjacent in G'. Hence any two witness sets in $\mathcal{W}'$ are adjacent to each other. This implies that $\mathcal{W}'$ is a clique witness structure of graph $G - \{v\}$. ◀

In the case of Split Contraction, the following observation guarantees the existence of witness structure with a particular property.

▶ **Observation 2.7.** *For a connected graph G, let F be a set of edges such that G/F is a split graph. Then, there exists a set of edges F' which satisfy the following properties: (i) G/F' is a split graph. (ii) The number of edges in F' is at most $|F|$. (iii) There exists a split partition of G/F' such that all vertices in G/F' which correspond to a non-trivial witness set in G/F'-witness structure of G are in clique side.*

Proof. Let $(C,\ I)$ be a split partition of vertices of G/F such that C is a clique and I is an independent set. If all the vertices corresponding to non-trivial witness sets are in C, then the observation is true. Consider a vertex a in I which corresponds to a non-trivial witness set W_a. Since G is connected, G/F is a connected split graph. This implies that there exists a vertex, say b, in C which is adjacent to a in G/F. We denote witness set corresponding to b by W_b. We construct a new witness structure by shifting all but one vertices in W_a to W_b. Since ab is an edge in G/F, there exists an edge in G with one endpoint in W_a and another in W_b. Let that edge be $u_a u_b$ with vertices u_a and u_b contained in sets W_a and W_b, respectively. Consider a spanning tree T of graph $G[W_a]$ which is rooted at u_a. We can replace edges in F whose both endpoints are in $V(W_a)$ with $E(T)$ to obtain another set of edges F^* such that G/F^* is a split graph. Formally, $F^* = (F \cup E(T)) \setminus (E(G[W_a]) \cap F)$. Note that the number of edges in F^* and F are same. Let v_1 be a leaf vertex in T and v_2 be its unique neighbor. Consider $F' = (F^* \cup \{u_a u_b\}) \setminus \{v_1 v_2\}$. Since edge $v_1 v_2$ is in F^* and $u_a u_b$ is not in F^*, $|F'| = |F^*|$. We now argue that G/F' is also a split graph. Let $\mathcal{W}'$ be the G/F'-witness structure of G. Note that $\mathcal{W}'$ can be obtained from G/F^*-witness structure $\mathcal{W}^*$ of G by replacing W_a by $\{v_1\}$ and W_b by $W_b \cup (W_a \setminus \{v_1\})$. Since all other witness set remains unchanged any witness set which was adjacent to W_b is also adjacent to $W_b \cup (W_a \setminus \{v_1\})$. Similarly, any witness set which was not adjacent to W_a is not adjacent to $\{v_2\}$. In other words, this operation of shifting edges did not remove any vertex from the neighborhood of b (which is in C) nor it added any vertex in the neighborhood of a (which is in I). Hence, G/F' is also a split graph with (C, I) as one of its split partition. Note that there exists a split partition of G/F' such that the number of vertices in the independent side corresponding to non-trivial witness set is one less than the number of vertices in I which corresponds to non-trivial witness sets. Hence, by repeating this process at most $|V(G)|$ times, we get a set of edges that satisfy three properties mentioned in the observation. ◀

2.3 Parameterized Complexity and Lossy Kernelization

An important notion in parameterized complexity is *kernelization*, which captures the efficiency of data reduction techniques. A parameterized problem Π admits a kernel of size $g(k)$ (or $g(k)$-kernel) if there is a polynomial time algorithm (called *kernelization algorithm*) which takes as input (I,k), and returns an instance (I',k') of Π such that: (i) (I,k) is a yes-instance if and only if (I',k') is a yes-instance; and (ii) $|I'|+k' \leq g(k)$, where $g(\cdot)$ is a computable function whose value depends only on k. Depending on whether the function $g(\cdot)$ is *linear, polynomial* or *exponential*, the problem is said to admit a *linear, polynomial* or *exponential kernel*, respectively. We refer to the corresponding chapters in the books [25, 14, 16, 23, 42] for a detailed introduction to the field of kernelization.

In lossy kernelization, we work with the optimization analog of parameterized problem. Along with an instance and a parameter, an optimization analog of the problem also has a string called *solution*. We start with the definition of a *parameterized optimization problem.* It is the parameterized analog of an optimization problem used in the theory of approximation algorithms.

▶ **Definition 7** (Parameterized Optimization Problem)**.** *A parameterized optimization problem is a computable function $\Pi : \Sigma^* \times \mathbb{N} \times \Sigma^* \mapsto \mathbb{R} \cup \{\pm\infty\}$. The instances of Π are pairs $(I,k) \in \Sigma^* \times \mathbb{N}$ and a solution to (I,k) which is simply a string $S \in \Sigma^*$ such that $|S| \leq |I|+k$.*

The *value* of a solution S is $\Pi(I,k,S)$. In this paper, all optimization problems are minimization problems. Therefore, we present the rest of the section only with respect to parameterized *minimization* problem. The *optimum value* of (I,k) is defined as:

$$\mathrm{OPT}_\Pi(I,k) = \min_{S \in \Sigma^*,\, |S| \leq |I|+k} \Pi(I,k,S),$$

and an *optimum solution* for (I,k) is a solution S such that $\Pi(I,k,S) = \mathrm{OPT}_\Pi(I,k)$. For a constant $c > 1$, S is *c-factor approximate* solution for (I,k) if $\frac{\Pi(I,k,S)}{OPT_\Pi(I,k)} \leq c$. We omit the subscript Π in the notation for optimum value if the problem under consideration is clear from the context.

For some parameterized optimization problems we are unable to obtain FPT algorithms, and we are also unable to find satisfactory polynomial time approximation algorithms. In this case one might aim for FPTapproximation algorithms, algorithms that run in time $f(k)n^c$ and provide good approximate solutions to the instance.

▶ **Definition 8.** *Let $\alpha \geq 1$ be constant. A fixed parameter tractable α-approximation algorithm for a parameterized optimization problem Π is an algorithm that takes as input an instance (I,k), runs in time $f(k)|I|^{\mathcal{O}(1)}$, and outputs a solution S such that $\Pi(I,k,S) \leq \alpha \cdot OPT(I,k)$ if Π is a minimization problem, and $\alpha \cdot \Pi(I,k,S) \geq OPT(I,k)$ if Π is a maximization problem.*

Note that Definition 8 only defines constant factor FPT-approximation algorithms. The definition can in a natural way be extended to approximation algorithms whose approximation ratio depends on the parameter k, on the instance I, or on both. Next, we define an α-*approximate polynomial-time preprocessing algorithm* for a parameterized minimization problem Π as follows.

▶ **Definition 9** (α-Approximate Polynomial-time Preprocessing Algorithm)**.** *Let $\alpha \geq 1$ be a real number and Π be a parameterized minimization problem. An α-approximate polynomial-time preprocessing algorithm is defined as a pair of polynomial-time algorithms, called the* reduction algorithm *and the* solution lifting algorithm*, that satisfy the following properties.*

- *Given an instance (I, k) of Π, the reduction algorithm computes an instance (I', k') of Π.*
- *Given instances (I, k) and (I', k') of Π, and a solution S' to (I', k'), the solution lifting algorithm computes a solution S to (I, k) such that $\frac{\Pi(I,k,S)}{OPT(I,k)} \leq \alpha \cdot \frac{\Pi(I',k',S')}{OPT(I',k')}$.*

We sometimes refer α-approximate polynomial-time preprocessing algorithm kernel as *α-lossy rule* or *α-reduction rule.*

3 Lossy Kernel for Clique Contraction

In this section, we present a lossy kernel for Clique Contraction. We first define a natural optimization version of the problem.

$$\text{CLC}(G, k, F) = \begin{cases} \min\{|F|, k+1\} & \text{if } \ G/F \ \text{ is a clique} \\ \infty & \text{otherwise} \end{cases}$$

If the number of vertices in the input graph is at most $k + 3$, then we can return the same instance as a kernel for the given problem. Further, we assume that the input graph is connected; otherwise, it can not be edited into a clique by edge contraction only. Thus, we only consider *connected graphs with at least $k + 3$ vertices.* By the definition of optimization problem, for any set of edges F, if G/F is a clique, then the maximum value of $\text{CLC}(G, k, F)$ is $k+1$. Hence, any spanning tree of G is a solution of cost $k+1$. We call it a *trivial solution* for the given instance. Consider an instance $(P_4, 1)$, where P_4 is a path on four vertices. One needs to contract at least two edges to convert P_4 into a clique. We call $(P_4, 1)$ a *trivial No-instance* for this problem. Finally, we assume that we are given an $\epsilon > 0$.

We start with a reduction rule, which says that if the minimum number of vertices that need to be deleted from an input graph to obtain a clique is large, then we can return a trivial instance as a lossy kernel.

▶ **Reduction Rule 3.1.** *For a given instance (G, k), apply the algorithm mentioned in Observation 2.1 to find a set X such that $G - X$ is a clique. If the size of X is greater than $4k$, then return $(P_4, 1)$.*

▶ **Lemma 10.** *Reduction Rule 3.1 is a 1-reduction rule.*

Proof. Let (G, k) be an instance of Clique Contraction such that the Reduction Rule 3.1 returns $(P_4, 1)$ when applied on it. The solution lifting algorithm returns a spanning tree F of G. Note that for a set of edges F', if P_4/F' is a clique then F' contains at least two edges. This implies $\text{CLC}(P_4, 1, F') = 2$ and $\text{OPT}(P_4, 1) = 2$.

Since a factor 2-approximation algorithm returned a set of size strictly more than $4k$, for any set X' of size at most $2k$, $G - X'$ is not a clique. But by Observation 2.4, if G is k-contractible to a clique then G can be edited into a clique by deleting at most $2k$ vertices. Hence, for any set of edges F^* if G/F^* is a clique, then the size of F^* is at least $k + 1$. This implies that $\text{OPT}(G, k) = k + 1$, and for a spanning tree F of G, $\text{CLC}(G, k, F) = k + 1$.

Combining these values, we get $\frac{\text{CLC}(G,k,F)}{\text{OPT}(G,k)} = \frac{k+1}{k+1} = \frac{2}{2} = \frac{\text{CLC}(P_4,1,F')}{\text{OPT}(P_4,1)}$. This implies that if F' is factor c-approximate solution for $(P_4, 1)$, then F is factor c-approximate solution for (G, k). This concludes the proof. ◀

We now consider an instance (G, k) for which Reduction Rule 3.1 does not return a trivial instance. This implies that for a given graph G, in polynomial time, one can find a partition (X, Y) of $V(G)$ such that $G - X = G[Y]$ is a clique and $|X|$ is at most $4k$. For $\epsilon > 0$, find a smallest integer d, such that $\frac{d+1}{d} \leq 1 + \epsilon$. In other words, fix $d = \lceil \frac{1}{\epsilon} \rceil$. We note that if the number of vertices in the graph is at most $\mathcal{O}(k^{d+1})$, then the algorithm returns this graph as a lossy kernel of the desired size. Hence, without loss of generality, we assume that the number of vertices in the graph is larger than $\mathcal{O}(k^{d+1})$.

Given an instance (G, k), a partition (X, Y) of $V(G)$ with $G[Y]$ being a clique, and an integer d, consider the following two marking schemes.

▶ **Marking Scheme 3.1.** *For a subset A of X, let $M_1(A)$ be the set of vertices in Y whose neighborhood contains A. For every subset A of X which is of size at most d, mark a vertex in $M_1(A)$.*

Formally, $M_1(A) = \{y \in Y | A \subseteq N(y)\}$. If $M_1(A)$ is an empty set, then the marking scheme does not mark any vertex. If it is non-empty, then the marking scheme arbitrarily chooses a vertex and marks it.

▶ **Marking Scheme 3.2.** *For a subset A of X, let $M_2(A)$ be the set of vertices in Y whose neighborhood does not intersect A. For every subset A of X which is of size at most d, mark $2k + 1$ vertices in $M_2(A)$.*

Formally, $M_2(A) = \{y \in Y | N(y) \cap A = \emptyset\}$. If the number of vertices in $M_2(A)$ is at most $2k+1$, then the marking scheme marks all vertices in $M_2(A)$. If it is larger than $2k+1$, then it arbitrarily chooses $2k + 1$ vertices and marks them.

▶ **Reduction Rule 3.2.** *For a given instance (G, k), partition (X, Y) of $V(G)$ with $G[Y]$ being a clique, and an integer d, apply the Marking Schemes 3.1 and 3.2. Let G' be the graph obtained from G by deleting all the unmarked vertices in Y. Return the instance (G', k).*

Above reduction rule can be applied in time $|X|^d \cdot |V(G)|^{\mathcal{O}(1)} = \mathcal{O}(k^{\mathcal{O}(d)}|V(G)|^{\mathcal{O}(1)})$ as $|X|$ is at most $4k$. Note that G' is an induced subgraph of G. We first show that since G is a connected graph, G' is also connected. In the following lemma, we prove a stronger statement.

▶ **Lemma 11.** *Consider instance (G, k) of* Clique Contraction. *Let Y' be the set of vertices marked by Marking Scheme 3.1 or 3.2 for some positive integer d. For any subset Y'' of $Y \setminus Y'$, let G'' be the graph obtained from G by deleting Y''. Then, G'' is connected.*

Proof. Recall that, by our assumption, G is connected and Y is a clique in G. Hence, for every vertex in X, there exists a path from it to some vertex in Y. By the construction of G'', $(X, Y \setminus Y'')$ forms a partition of $V(G'')$ and $Y \setminus Y''$ is a clique in G''. To prove that G'' is connected, it is sufficient to prove that for every vertex in X, there exists a path from it to a vertex in $Y \setminus Y''$ in G.

Fix an arbitrary vertex, say x, in X. Consider a path P from x to y in G, where y is some vertex in Y. Without loss of generality, we can assume that y is the only vertex in $V(P) \cap Y$. We argue that there exists another path, say P_1, from x to a vertex in $Y \setminus Y''$. If y is in $Y \setminus Y''$ then $P_1 = P$ is a desired path. Consider the case when y is in Y''. Let x_0 be the vertex in $V(P)$ which is adjacent with y. Note that x_0 may be same as x. As Marking Scheme 3.1 considers all subsets of size at most d, it considered singleton set $\{x_0\}$. As x_0 is adjacent with y, we have $\{x_0\} \subseteq N(y)$. Since y is in Y'', and hence unmarked, there exists a vertex, say y_1, in Y which has been marked by Marking Scheme 3.1. Consider a path P_1 obtained from P by deleting vertex y (and hence edge x_0y) and adding vertex y_1 with edge x_0y_1. This is a desired path from x to a vertex in $Y \setminus Y''$. As x is an arbitrary vertex in X, this statement is true for any vertex in X and hence G'' is connected. ◀

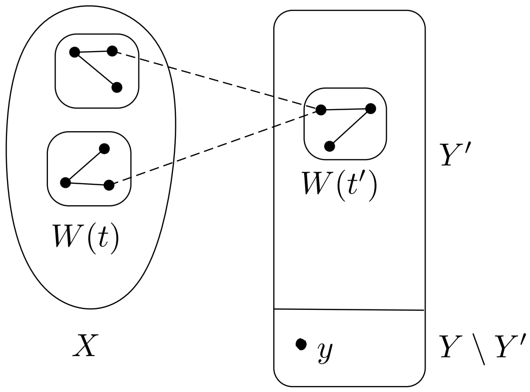

Figure 1 Straight lines (e.g within $W(t)$) represent edges in original solution F. Dashed lines (e.g. across $W(t)$ and $W(t')$) represents extra edges added to solution F. Please refer to the proof of Lemma 12.

Thus, because of Lemma 11, from now onwards, we assume that G' is connected. In fact, in our one of the proof, we will iteratively remove vertices from $Y \setminus Y'$, and Lemma 11 ensures that the graph at each step remains connected. In the following lemma, we argue that given a solution for (G', k), we can construct a solution of almost the same size for (G, k).

▶ **Lemma 12.** *Let (G', k) be the instance returned by Reduction Rule 3.2 when applied on an instance (G, k). If there exists a set of edges of size at most k, say F', such that G'/F' is a clique, then there exists a set of edges F such that G/F is a clique and cardinality of F is at most $(1 + \epsilon) \cdot |F'|$.*

Proof. If no vertex in Y is deleted, then G' and G are identical graphs, and the statement is true. We assume that at least one vertex in Y is deleted. Let Y' be the set of vertices in Y, which are marked. Note that the sets X, Y' forms a partition of $V(G')$ such that Y' is a clique and a proper subset of Y. Let $\mathcal{W}'$ be a G'/F'-witness structure of G'. We construct a clique witness structure $\mathcal{W}$ of G from $\mathcal{W}'$ by adding singleton witness sets $\{y\}$ for every vertex y in $Y \setminus Y'$. Since $G[Y \setminus Y']$ is a clique in G, any two newly added witness sets are adjacent to each other. Moreover, any witness set in $\mathcal{W}'$, which intersects Y' is also adjacent to the newly added witness sets. We now consider witness sets in $\mathcal{W}'$, which do not intersect Y'.

Let $\mathcal{W}^\star$ be a collection of witness sets $W(t)$ in $\mathcal{W}'$ such that $W(t)$ is contained in X and there exists a vertex y in $Y \setminus Y'$ whose neighborhood does not intersect with $W(t)$. See Figure 1. We argue that every witness set in $\mathcal{W}^\star$ has at least $d + 1$ vertices. For the sake of contradiction, assume that there exists a witness set $W(t)$ in $\mathcal{W}^\star$ which contains at most d vertices. Since Marking Scheme 3.2 iterated over all the subsets of X of size at most d, it also considered $W(t)$ while marking. Note that the vertex y belongs to the set $M_2(W(t))$. Since y is unmarked, there are $2k + 1$ vertices in $M_2(W(t))$ which have been marked. All these marked vertices are in G'. Since the cardinality of F' is at most k, the number of vertices in $V(F')$ is at most $2k$. Hence, at least one marked vertex in $M_2(W(t))$ is a singleton witness set in $\mathcal{W}'$. However, there is no edge between this singleton witness set and $W(t)$. This non-existence of an edge contradicts the fact that any two witness sets in $\mathcal{W}'$ are adjacent to each other in G'. Hence, our assumption is wrong, and $W(t)$ has at least $d + 1$ vertices.

Next, we show that there exists a witness set in $\mathcal{W}'$ that intersects Y'. This is ensured by the fact that G' is connected, and we are in the case where at least one vertex in Y is deleted. The last assertion implies that Y' is non-empty, and hence there must be a witness set in $\mathcal{W}'$ that intersects Y'. Let $W(t')$ be a witness set in $\mathcal{W}'$ that intersects Y'. Note that $W(t')$ is adjacent to every vertex in $Y \setminus Y'$. Let $W(t)$ be a witness set in $\mathcal{W}^\star$. Since $W(t')$ and $W(t)$ are two witness sets in the G'/F'-witness structure, there exists an edge with one endpoint in $W(t')$ and another in $W(t)$. Therefore, the set $W(t') \cup W(t)$ is adjacent to every other witness set in $\mathcal{W}$.

We now describe how to obtain F from F'. We initialize $F = F'$. For every witness set $W(t)$ in $\mathcal{W}^\star$ add an edge between $W(t)$ and $W(t')$ to the set F'. Equivalently, we construct a new witness set by taking the union of $W(t')$ and all witness sets $W(t)$ in $\mathcal{W}^\star$. This witness set is adjacent to every vertex in $Y \setminus Y'$, and hence G/F is a clique. We now argue the size bound on F. Note that we have added one extra edge for every witness set $W(t)$ in $\mathcal{W}^\star$. We also know that every such witness set has at least $d + 1$ vertices. Hence, we have added one extra edge for at least d edges in the solution F'. Moreover, since witness sets in $\mathcal{W}^\star$ are vertex disjoint, no edge in F can be part of two witness sets. This implies that the number of edges in F' is at most $({}^{d+1}/{}_{d})|F| \leq (1 + \epsilon) \cdot |F|$. ◀

In the following lemma, we argue that the value of the optimum solution for the reduced instance can be upper bounded by the value of an optimum solution for the original instance.

▶ **Lemma 13.** *Let (G', k) be the instance returned by Reduction Rule 3.2 when applied on an instance (G, k). If $OPT(G, k) \leq k$, then $OPT(G', k) \leq OPT(G, k)$.*

Proof. Let F be a set of at most k edges in G such that $\text{OPT}(G, k) = \textsc{ClC}(G, k, F)$ and $\mathcal{W}$ be a G/F-witness structure of G. Since we are working with a minimization problem, to prove this lemma it is sufficient to find a solution for G' which is of size $|F|$. Recall that (X, Y) is a partition of $V(G)$ such that $G - X = G[Y]$ is a clique. Let Y' be the set of vertices that were marked by either of the marking schemes. In other words, (X, Y') is a partition of G' such that $G' - X = G'[Y]$ is a clique. We proceed as follows. At each step, we construct graph $G^\star$ from G by deleting one or more vertices of $Y \setminus Y'$. Simultaneously, we also construct a set of edges $F^\star$ from F by either replacing the existing edges by new ones or by simply adding extra edges to F. At any intermediate state, we ensure that $G^\star/F^\star$ is a clique, and the number of edges in $F^\star$ is at most $|F|$. Let $F^\circ = F$ be an optimum solution for the input instance (G, k). *For notational convenience, we rename $G^\star$ to G and $F^\star$ to F at regular intervals but do not change F°.*

To obtain $G^\star$ and $F^\star$, we delete witness sets which are subsets of $Y \setminus Y'$ (Condition 3.1) and modify the ones which intersect with $Y \setminus Y'$. Every witness set of latter type intersects with Y' or X or both. We partition these non-trivial witness sets in $\mathcal{W}$ into two groups depending on whether the intersection with X is empty (Condition 3.2) or not (Condition 3.3). We first modify the witness sets that satisfy the least indexed condition. If there does not exist a witness set which satisfies either of these three conditions, then $Y \setminus Y'$ is an empty set, and the lemma is vacuously true.

▶ **Condition 3.1.** *There exists a witness set $W(t)$ in $\mathcal{W}$ which is a subset of $Y \setminus Y'$.*

Construct $G^\star$ from G by deleting the witness sets $W(t)$ in $\mathcal{W}$. Let $F^\star$ be obtained from F by deleting those edges whose both the endpoints are in $W(t)$. Since the class of cliques is closed under vertex deletion, $G^\star/F^\star$ is a clique, and as we only deleted edges from F, we have $|F^*| \leq |F|$. We repeat this process until there exists a witness set that satisfies Condition 3.1.

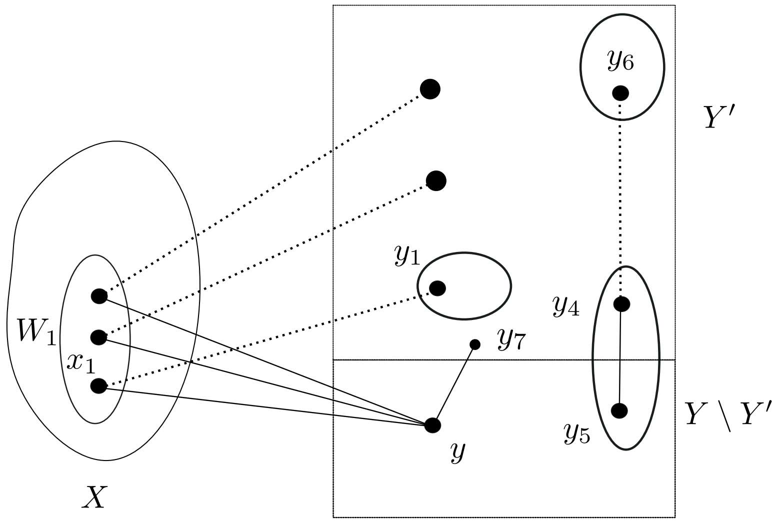

Figure 2 Straight lines (e.g. y_4y_5) represent edges in original solution F. Dotted lines (e.g. y_4y_6) represents edges which are replaced for some edges in F. Please refer to the proof of Lemma 13.

> At this stage we rename $G^\star$ to G and $F^\star$ to F.

▶ **Condition 3.2.** *There exists a witness set $W(t)$ in $\mathcal{W}$ which contains vertices from $Y \setminus Y'$ but does not intersect X.*

Since $W(t)$ is not contained in $Y \setminus Y'$ and $W(t) \cap X$ is empty it must intersect with Y'. See Figure 2. Let y_4 and y_5 be vertices in $W(t) \cap Y'$ and $W(t) \cap (Y \setminus Y')$, respectively. Let $W(t_1)$, different from $W(t)$, be a witness set which intersects Y'. Since Y' is large and non-empty, such a witness set exists. Let y_6 be a vertex in the set $W(t_1) \cap Y'$. Consider the witness sets $W(t), W(t_1)$ and vertex y_5 in $W(t)$ in graph G. Lemma 11 implies that these witness sets satisfy the premise of Observation 2.6. This implies $\mathcal{W}^\star$ is a clique witness structure of $G - \{y_5\}$, where $\mathcal{W}^\star$ is obtained from $\mathcal{W}$ by removing $W(t), W(t_1)$ and adding $(W(t) \cup W(t_1)) \setminus \{y_5\}$. This corresponds to replacing an edge in F which was incident to y_5 with the one across $W(t)$ and $W(t_1)$. For example, in Figure 2, we replace edge y_4y_5 in the set F with an edge y_4y_6 to obtain a solution for $G - \{y_5\}$. An edge in F has been replaced with another edge and one vertex in $Y \setminus Y'$ is deleted. The size of $F^\star$ is same as that of F and $G^\star/F^\star$ is a clique. We repeat this process until there exist a witness set which satisfies Condition 3.2.

> At this stage we rename $G^\star$ to G and $F^\star$ to F.

▶ **Condition 3.3.** *There exists a witness set $W(t)$ in $\mathcal{W}$ which contains vertices from $Y \setminus Y'$ and intersects X.*

Let y be a vertex in $W(t) \cap (Y \setminus Y')$, X_t be the set of vertices in $W(t) \cap X$ which are adjacent to y via edges in F, and Q_t be the set of vertices in $W(t) \cap Y$ which are adjacent to y via edges in F. We find a *substitute* for y in Y'. If the set X_t is empty then the vertex y is adjacent only with the vertices of Y, in this case the edges incident to y can be replaced as mentioned in the Condition 3.2. Assume that X_t is non-empty. For every vertex x in X_t the set $\{x\}$ is considered by Marking Scheme 3.1. Since y is adjacent to every vertex x in X_t, the set $M_1(\{x\})$ is non-empty. As y is in $Y \setminus Y'$, and hence unmarked, for every x in X_t, there is a vertex in $M_1(\{x\})$, say y_x, different from y which has been marked. We construct $F^\star$ from F by the following operation: For every vertex x in X_t, replace the edge xy in F by xy_x. Fix a vertex x_o in X_t, and for every vertex u in Q_t, replace the edge uy in F with uy_{x_o}. Since we are replacing a set of edges in F with another set of edges of same size we have $|F^\star| \leq |F|$. (For example, in Figure 2, $X_t = W_1$ and $Q_t = \{y_7\}$. Edges yx_1, yy_7 are replaced by x_1y_1, y_1y_7 resp.) We argue that if $G^\star$ is obtained from G by removing y, then $G^\star/F^\star$ is a clique.

We argue that contracting edges in $F^\star$ partitions $W(t)$ into $|X_t| + |Q_t|$ many parts and merges each part with some witness set in $\mathcal{W} \setminus \{W(t)\}$. Recall that F contains a spanning tree of graph $G[W(t)]$. Let T be a spanning tree of $G[W(t)]$ such that $E(T) \subseteq F$ and T contains all edges in F that are incident on y. It is easy to see that such a spanning tree exists. Let y be the root of tree T. For every z in $X_t \cup Q_t$, let $W'(z)$ be the set of vertices in the subtree of T rooted at z. As $V(T) = W(t)$, set $\{W'(z) |\ z \in X_t \cup Q_t\}$ is a partition of $W(t) \setminus \{y\}$. For every x in X_t, let $W(y_x)$ be the witness set in $\mathcal{W}$ containing the vertex y_x. For every x in $X_t \setminus \{x_o\}$, let $W^\star(y_x)$ be the set $W(y_x) \cup W'(x)$. Let $W^\star(y_{x_o})$ be the set $W(y_{x_o}) \cup W'(x_o) \cup \bigcup_{y'} W'(y')$ for all y' in Q_t. We obtain $\mathcal{W}^\star$ from $\mathcal{W}$ by removing $W(t)$ and $W(y_x)$ for every x in X_t, and adding the sets $W^\star(y_x)$ for every x in X_t. Since $W^\star(y_x)$ contains the set $W(y_x)$ which was adjacent to every witness set in $\mathcal{W}$, $W^\star(y_x)$ will be adjacent with every witness set in $\mathcal{W}^\star$. We repeat this process until there exists a witness set that satisfies this condition.

Any vertex in $Y \setminus Y'$ must be a part of some witness set in $\mathcal{W}$, and any witness set in $\mathcal{W}$ satisfies at least one of the above conditions. If there are no witness sets that satisfy these conditions, then $Y \setminus Y'$ is empty. This implies $G^\star = G'$ and there exists a solution $F^\star$ of size at most $|F^\circ|$. This concludes the proof of the lemma. ◀

We are now in a position to prove the following lemma.

▶ **Lemma 14.** *Reduction Rule 3.2, along with a solution lifting algorithm, is an* $(1+\epsilon)$*-reduction rule.*

Proof. Let (G', k) be the instance returned by Reduction Rule 3.2 when applied on an instance (G, k). We present a solution lifting algorithm as follows. For a solution F' for (G, k) if $\textsc{Clc}(G', k, F') = k+1$, then the solution lifting algorithm returns a spanning tree F of G (a trivial solution) as solution for (G, k). In this case, $\textsc{Clc}(G, k, F) = \textsc{Clc}(G', k, F')$. If $\textsc{Clc}(G', k, F') \leq k$, then size of F' is at most k and G'/F' is a clique. Solution lifting algorithm uses Lemma 12 to construct a solution F for (G, k) such that cardinality of F is at most $(1+\epsilon) \cdot |F'|$. In this case, $\textsc{Clc}(G, k, F) \leq (1+\epsilon) \cdot \textsc{Clc}(G', k, F')$. Hence, there exists a solution lifting algorithm which given a solution F' for (G', k') returns a solution F for (G, k) such that $\textsc{Clc}(G, k, F) \leq (1+\epsilon) \cdot \textsc{Clc}(G', k, F')$.

If $\mathrm{OPT}(G, k) \leq k$, then by Lemma 13, $\mathrm{OPT}(G', k) \leq \mathrm{OPT}(G, k)$. If $\mathrm{OPT}(G, k) = k+1$ then $\mathrm{OPT}(G', k) \leq k+1 = \mathrm{OPT}(G, k)$. Hence in either case, $\mathrm{OPT}(G', k) \leq \mathrm{OPT}(G, k)$.

Combining the two inequalities, we get $\frac{\textsc{Clc}(G,k,F)}{\mathrm{OPT}(G,k)} \leq \frac{(1+\epsilon)\cdot\textsc{Clc}(G',k,F')}{\mathrm{OPT}(G',k)}$. This implies that if F' is a factor c-approximate solution for (G', k) then F is a factor $(c \cdot (1+\epsilon))$-approximate solution for (G, k). This concludes the proof. ◀

We are now in a position to present the main result of this section.

Proof. (of Theorem 1) For a given instance (G, k) with $|V(G)| \geq k + 3$, a kernelization algorithm applies the Reduction Rule 3.1. If it returns a trivial instance, then the statement is vacuously true. If it does not return a trivial instance, then the algorithm partitions $V(G)$ into two sets (X, Y) such that $G - X = G[Y]$ is a clique and size of X is at most $4k$. Then the algorithm applies the Reduction Rule 3.2 on the instance (G, k) with the partition (X, Y) and the integer $d = \lceil \frac{1}{\epsilon} \rceil$. The algorithm returns the reduced instance as $(1 + \epsilon)$-lossy kernel for (G, k).

The correctness of the algorithm follows from Lemma 10 and Lemma 14 combined with the fact that Reduction Rule 3.2 is applied at most once. By Observation 2.1, Reduction Rule 3.1 can be applied in polynomial time. The size of the instance returned by Reduction Rule 3.2 is at most $\mathcal{O}((4k)^d \cdot (2k+1) + 4k) = \mathcal{O}(k^{d+1})$. Reduction Rule 3.2 can be applied in time $n^{\mathcal{O}(1)}$ if the number of vertices in (G, k) is more than $\mathcal{O}(k^{d+1})$. ◀

References

1 Akanksha Agarwal, Saket Saurabh, and Prafullkumar Tale. On the parameterized complexity of contraction to generalization of trees. *Theory of Computing Systems*, pages 1–28, 2017.

2 Akanksha Agrawal, Daniel Lokshtanov, Saket Saurabh, and Meirav Zehavi. Split contraction: The untold story. In *STACS*, pages 5:1–5:14, 2017. doi:10.4230/LIPIcs.STACS.2017.5.

3 Takao Asano and Tomio Hirata. Edge-contraction problems. *Journal of Computer and System Sciences*, 26(2):197–208, 1983.

4 Reuven Bar-Yehuda and Shimon Even. A linear-time approximation algorithm for the weighted vertex cover problem. *Journal of Algorithms*, 2(2):198–203, 1981.

5 Ivan Bliznets, Fedor V. Fomin, Marcin Pilipczuk, and Michal Pilipczuk. A subexponential parameterized algorithm for proper interval completion. *SIAM Journal on Discrete Mathematics*, 29(4):1961–1987, 2015.

6 Ivan Bliznets, Fedor V. Fomin, Marcin Pilipczuk, and Michal Pilipczuk. Subexponential parameterized algorithm for interval completion. *ACM Transactions on Algorithms*, 14(3):35:1–35:62, 2018.

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

8 Leizhen Cai and Chengwei Guo. Contracting few edges to remove forbidden induced subgraphs. In *IPEC*, pages 97–109, 2013.

9 Yixin Cao. Linear recognition of almost interval graphs. In *Proceedings of the 27th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 1096–1115, Arlington, Virginia, 2016. ACM-SIAM.

10 Yixin Cao. Unit interval editing is fixed-parameter tractable. *Information and Computation*, 253:109–126, 2017.

11 Yixin Cao and Dániel Marx. Interval deletion is fixed-parameter tractable. *ACM Transactions on Algorithms*, 11(3):21:1–21:35, 2015.

12 Yixin Cao and Dániel Marx. Chordal editing is fixed-parameter tractable. *Algorithmica*, 75(1):118–137, 2016.

13 Parinya Chalermsook, Marek Cygan, Guy Kortsarz, Bundit Laekhanukit, Pasin Manurangsi, Danupon Nanongkai, and Luca Trevisan. From Gap-ETH to FPT-inapproximability: Clique, dominating set, and more. In *FOCS*, pages 743–754, 2017. doi:10.1109/FOCS.2017.74.

14 Marek Cygan, Fedor V. Fomin, Łukasz Kowalik, Daniel Lokshtanov, Dániel Marx, Marcin Pilipczuk, Michal Pilipczuk, and Saket Saurabh. *Parameterized Algorithms*. Springer International Publishing, Switzerland, 2015.

15 Reinhard Diestel. *Graph Theory, 4th Edition*, volume 173 of *Graduate texts in mathematics*. Springer-Verlag Berlin Heidelberg, Germany, 2012.

16 Rod G. Downey and Michael R. Fellows. *Fundamentals of Parameterized complexity*. Springer-Verlag, London, 2013.

17 Pål Grønås Drange, Markus Sortland Dregi, Daniel Lokshtanov, and Blair D. Sullivan. On the threshold of intractability. In *Algorithms - 23rd Annual European Symposium (ESA)*, pages 411–423, Germany, 2015. Springer-Verlag Berlin Heidelberg.

18 Pål Grønås Drange, Fedor V. Fomin, Michal Pilipczuk, and Yngve Villanger. Exploring the subexponential complexity of completion problems. *ACM Transactions on Computation Theory*, 7(4):14:1–14:38, 2015.

19 Pål Grønås Drange and Michal Pilipczuk. A polynomial kernel for trivially perfect editing. *Algorithmica*, 80:3481–3524, 2018.

20 Pavel Dvorák, Andreas Emil Feldmann, Dusan Knop, Tomás Masarík, Tomas Toufar, and Pavel Veselý. Parameterized approximation schemes for steiner trees with small number of steiner vertices. In *STACS*, pages 26:1–26:15, 2018.

21 Eduard Eiben, Danny Hermelin, and M. S. Ramanujan. Lossy kernels for hitting subgraphs. In *MFCS*, pages 67:1–67:14, 2017. doi:10.4230/LIPIcs.MFCS.2017.67.

22 Eduard Eiben, Mithilesh Kumar, Amer E Mouawad, Fahad Panolan, and Sebastian Siebertz. Lossy kernels for connected dominating set on sparse graphs. *SIAM Journal on Discrete Mathematics*, 33(3):1743–1771, 2019.

23 Jörg Flum and Martin Grohe. *Parameterized Complexity Theory*. Texts in Theoretical Computer Science. An EATCS Series. Springer-Verlag Berlin Heidelberg, Germany, 2006.

24 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.

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

26 Fedor V Fomin and Yngve Villanger. Subexponential parameterized algorithm for minimum fill-in. *SIAM Journal on Computing*, 42(6):2197–2216, 2013.

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

28 Petr A. Golovach, Pim van 't Hof, and Daniel Paulusma. Obtaining planarity by contracting few edges. *Theoretical Computer Science*, 476:38–46, 2013.

29 Martin Charles Golumbic. *Algorithmic graph theory and perfect graphs*, volume 57. Elsevier, Academic Press, 2004.

30 Sylvain Guillemot and Dániel Marx. A faster FPT algorithm for bipartite contraction. *Information Processing Letters*, 113(22–24):906–912, 2013.

31 Spoorthy Gunda, Pallavi Jain, Daniel Lokshtanov, Saket Saurabh, and Prafullkumar Tale. On the parameterized approximability of contraction to classes of chordal graphs, 2020. arXiv:2006.10364.

32 Chengwei Guo and Leizhen Cai. Obtaining split graphs by edge contraction. *Theoretical Computer Science*, 607:60–67, 2015.

33 Pinar Heggernes, Pim van 't Hof, Benjamin Lévêque, Daniel Lokshtanov, and Christophe Paul. Contracting graphs to paths and trees. *Algorithmica*, 68(1):109–132, 2014.

34 Pinar Heggernes, Pim van 't Hof, Daniel Lokshtanov, and Christophe Paul. Obtaining a bipartite graph by contracting few edges. *SIAM Journal on Discrete Mathematics*, 27(4):2143–2156, 2013.

35 Karthik C. S., Bundit Laekhanukit, and Pasin Manurangsi. On the parameterized complexity of approximating dominating set. In *STOC*, pages 1283–1296. ACM, 2018. doi:10.1145/3188745.3188896.

36 R Krithika, Diptapriyo Majumdar, and Venkatesh Raman. Revisiting connected vertex cover: Fpt algorithms and lossy kernels. *Theory of Computing Systems*, 62(8):1690–1714, 2018.

37 R Krithika, Pranabendu Misra, Ashutosh Rai, and Prafullkumar Tale. Lossy kernels for graph contraction problems. In *FSTTCS*, pages 23:1–23:14, 2016.

38 Daniel Lokshtanov, Neeldhara Misra, and Saket Saurabh. On the hardness of eliminating small induced subgraphs by contracting edges. In *IPEC*, pages 243–254, 2013.

39 Daniel Lokshtanov, Pranabendu Misra, Fahad Panolan, Geevarghese Philip, and Saket Saurabh. A $(2+\epsilon)$-factor approximation algorithm for split vertex deletion. In *47th International Colloquium on Automata, Languages and Programming, ICALP 2020*, volume 168 of *LIPIcs*, page to appear, 2020.

40 Daniel Lokshtanov, Fahad Panolan, M. S. Ramanujan, and Saket Saurabh. Lossy kernelization. In *STOC*, pages 224–237, 2017.

41 Pasin Manurangsi. A note on max k-vertex cover: Faster FPT-AS, smaller approximate kernel and improved approximation. In *SOSA*, 2019.

42 Rolf Niedermeier. *Invitation to fixed-parameter algorithms.* Oxford Lecture Series in Mathematics and Its Applications. Oxford University Press, Oxford, 2006.

43 M. S. Ramanujan. An approximate kernel for connected feedback vertex set. In *27th Annual European Symposium on Algorithms, ESA 2019, September 9-11, 2019, Munich/Garching, Germany*, volume 144 of *LIPIcs*, pages 77:1–77:14, 2019.

44 Sebastian Siebertz. Lossy kernels for connected distance-r domination on nowhere dense graph classes. *arXiv preprint*, 2017. `arXiv:1707.09819`.

45 René van Bevern, Till Fluschnik, and Oxana Yu Tsidulko. On $(1+\epsilon)$-approximate problem kernels for the rural postman problem. *arXiv preprint*, 2018. `arXiv:1812.10131`.

46 Toshimasa Watanabe, Tadashi Ae, and Akira Nakamura. On the removal of forbidden graphs by edge-deletion or by edge-contraction. *Discrete Applied Mathematics*, 3(2):151–153, 1981.

47 Toshimasa Watanabe, Tadashi Ae, and Akira Nakamura. On the NP-hardness of edge-deletion and-contraction problems. *Discrete Applied Mathematics*, 6(1):63–78, 1983.