Quick Separation in Chordal and Split Graphs

Pranabendu Misra
Max Planck Institute for Informatics, Saarland Informatics Campus, Saarbrücken, Germany
pmisra@mpi-inf.mpg.de

Fahad Panolan
Department of Computer Science and Engineering, IIT Hyderabad, India
fahad@iith.ac.in

Ashutosh Rai
Depaertment of Applied Mathematics, Charles University, Prague, Czech Republic
ashutosh@kam.mff.cuni.cz

Saket Saurabh
Institute of Mathematical Sciences, HBNI, India
UMI ReLax, Chennai, India
University of Bergen, Norway
saket@imsc.res.in

Roohani Sharma
Institute of Mathematical Sciences, HBNI, India
roohani@imsc.res.in

Abstract

In this paper we study two classical cut problems, namely Multicut and Multiway Cut on chordal graphs and split graphs. In the Multicut problem, the input is a graph G, a collection of ℓ vertex pairs $(s_i, t_i), i \in [\ell]$, and a positive integer k and the goal is to decide if there exists a vertex subset $S \subseteq V(G) \setminus \{s_i, t_i : i \in [\ell]\}$ of size at most k such that for every vertex pair (s_i, t_i), s_i and t_i are in two different connected components of $G - S$. In Unrestricted Multicut, the solution S can possibly pick the vertices in the vertex pairs $\{(s_i, t_i) : i \in [\ell]\}$. An important special case of the Multicut problem is the Multiway Cut problem, where instead of vertex pairs, we are given a set T of terminal vertices, and the goal is to separate every pair of distinct vertices in $T \times T$. The fixed parameter tractability (FPT) of these problems was a long-standing open problem and has been resolved fairly recently. Multicut and Multiway Cut now admit algorithms with running times $2^{\mathcal{O}(k^3)}n^{\mathcal{O}(1)}$ and $2^k n^{\mathcal{O}(1)}$, respectively. However, the kernelization complexity of both these problems is not fully resolved: while Multicut cannot admit a polynomial kernel under reasonable complexity assumptions, it is a well known open problem to construct a polynomial kernel for Multiway Cut. Towards designing faster FPT algorithms and polynomial kernels for the above mentioned problems, we study them on *chordal* and *split* graphs. In particular we obtain the following results.

1. Multicut on chordal graphs admits a polynomial kernel with $\mathcal{O}(k^3\ell^7)$ vertices. Multiway Cut on chordal graphs admits a polynomial kernel with $\mathcal{O}(k^{13})$ vertices.
2. Multicut on chordal graphs can be solved in time $\min\{\mathcal{O}(2^k \cdot (k^3 + \ell) \cdot (n + m)), 2^{\mathcal{O}(\ell \log k)} \cdot (n + m) + \ell(n + m)\}$. Hence Multicut on chordal graphs parameterized by the number of terminals is in XP.
3. Multicut on split graphs can be solved in time $\min\{\mathcal{O}(1.2738^k + kn + \ell(n+m), \mathcal{O}(2^\ell \cdot \ell \cdot (n+m))\}$. Unrestricted Multicut on split graphs can be solved in time $\mathcal{O}(4^\ell \cdot \ell \cdot (n + m))$.

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

Keywords and phrases chordal graphs, multicut, multiway cut, FPT, kernel

Digital Object Identifier 10.4230/LIPIcs.MFCS.2020.70

Related Version A full version of the paper is available at `https://kam.mff.cuni.cz/~ashutosh/Papers/Conference/Multicut_Chordal.pdf`.

Funding *Ashutosh Rai*: Supported by Center for Foundations of Modern Computer Science (Charles University projectUNCE/SCI/004).

Saket Saurabh: European Research Council (ERC) under the European Union's Horizon 2020 research and innovation programme (grant no. 819416), and Swarnajayanti Fellowship grant DST/SJF/MSA-01/2017-18.

1 Introduction

Graph cuts and flows are a central topic in computer science and combinatorial optimization. A fundamental problem in this setting is Multicut, where the input is a graph G and a collection of ℓ terminal vertex pairs $(s_i, t_i), i \in [\ell]$, and the goal is to output a minimum sized vertex subset $S \subseteq V(G) \setminus \{s_i, t_i : i \in [\ell]\}$ such that for every vertex pair (s_i, t_i), s_i and t_i are in two different connected components of $G - S$. Another variant of the problem where a solution can possibly contain vertices from terminal pairs is called Unrestricted Multicut. Note that Unrestricted Multicut can be easily reduced to Multicut by adding a new terminal of degree one for each existing terminal and making it adjacent to the existing terminal. An important special case of Multicut is Multiway Cut, where we are given a set T of terminal vertices, and the goal is to separate every pair of distinct vertices in $T \times T$. One can similarly define Unrestricted Multiway Cut, where the solution can possibly pick terminal vertices. When $|T| = 2$ this is the famous Min (s, t)-Cut problem which admits a classical polynomial time algorithm. However, Multiway Cut becomes NP-hard even for a set of three terminals [7].

These problems appear in a number of applications, and they have been intensively studied over the past few decades, and several algorithmic tools and hardness results on them have been obtained in the field of approximation algorithms. These problems have played an important role in the development of the field of parameterized complexity. The first FPT algorithm for Multiway Cut parameterized by solution size k was given by Marx [17], who introduced the notion of *important separators*. This notion has since become an important algorithmic tool in the design of parameterized algorithms. Then, an algorithm of running time $4^k n^{\mathcal{O}(1)}$ was designed by Chen, Liu, and Lu [3], and later this was improved to $2^k n^{\mathcal{O}(1)}$ by Cygan et al. [5]. In fact, the algorithm of Cygan et al. [5] also gives a $4^{k-LP} n^{\mathcal{O}(1)}$ time algorithm for Multiway Cut, where LP is the optimal value of the LP relaxation of the natural ILP formulation for the problem. Marx and Razgon [18] and Bousquet et.al [1] independently proved that Multicut is FPT when parameterized by the solution size k. In particular, the algorithm of Marx and Razgon [18] developed the technique of *randomized sampling of important separators* which results in the best known algorithm for Multicut with running time $2^{\mathcal{O}(k^3)} n^{\mathcal{O}(1)}$.

These problems are very well studied from the perspective of kernelization as well. For Unrestricted Multiway Cut, a randomized polynomial kernel with $\mathcal{O}(k^3)$ vertices was obtained by Kratsch and Wahlström using the technique of representative families on gammoids [16]. They also designed a randomized kernel for Multiway Cut and Multicut with $\mathcal{O}(k^{\ell+1})$ and $\mathcal{O}(k^{\lceil\sqrt{2\ell}\rceil+1})$ vertices, respectively.[1] Obtaining a polynomial kernel for Multiway Cut when the parameter is k alone remains a long standing open problem in the field of kernelization. This is also posed as an open problem in the recent book on kernelization [10]. However, for Multicut, it is known that if there exists a polynomial

[1] This work is an extension of their work of randomized polynomial kernel for Odd Cycle Transversal which was awarded the EATCS-IPEC Nerode Prize 2018.

kernel with parameter k, then co-NP $\subseteq$ NP/poly and the polynomial hierarchy collapses to the third level [4]. This effectively rules out a polynomial kernel for this problem when parameterized by k alone, while a polynomial kernel when parameterized by both k and ℓ is not ruled out for general graphs.

Obtaining an algorithm faster than $2^{\mathcal{O}(k^3)}n^{\mathcal{O}(1)}$ time for Multicut is an outstanding open problem, and one way forward towards this is to understand the complexity of Multicut on special classes of graphs. Let us recall some results obtained in this direction. Calinescu et al. proved that Multicut is NP-hard even on bounded degree trees, whereas Unrestricted Multicut can be solved in polynomial time [6]. They also showed that even Unrestricted Multicut becomes NP-complete on bounded degree graphs of tree-width two. On the other hand, Guo et al. proved that Unrestricted Multicut is NP-complete on interval graphs, while Multicut is polynomial time solvable on interval graphs [13]. Papadopoulos proved that Multicut is polynomial time solvable on permutation graphs and co-bipartite graphs, but NP-complete on split graphs (which is subclass of chordal graphs) [19]. For planar graphs, Dahlhaus et al. showed that Multiway Cut can be solved in time $n^{\mathcal{O}(\ell)}$ [7]. This running time is improved to $2^{\mathcal{O}(\ell)}n^{\mathcal{O}(\sqrt{\ell})}$ and a matching lower bound under ETH is provided by Klein and Marx [15]. Note that this is not true in general graphs, as Multiway Cut is NP-hard even when $\ell = 3$. Very recently, a polynomial kernel for Multiway Cut on planar graphs parameterized by k is obtained by Jansen et al. [14].

In this paper, we study Multicut, Unrestricted Multicut, and Multiway Cut on chordal graphs and split graphs, and obtain new fast FPT algorithms and polynomial kernels for them. These problems are formally defined as follows.

Multicut (MC)	
Input:	A graph $G = (V, E)$, a set of pairs of vertices $T = \{(s_i, t_i) \mid i \in [\ell]\}$ and an integer k.
Parameter:	k, ℓ
Question:	Does there exist $S \subseteq V(G) \setminus \cup_{i \in \ell}\{s_i, t_i\}$ such that $\lvert S\rvert \leq k$ and there is no path from s_i to t_i for all $i \in [\ell]$ in $G - S$?

Unrestricted Multicut (UMC)	
Input:	A graph $G = (V, E)$, a set of pairs of vertices $T = \{(s_i, t_i) \mid i \in [\ell]\}$ and an integer k.
Parameter:	k, ℓ
Question:	Does there exist $S \subseteq V(G)$ such that $\lvert S\rvert \leq k$ and there is no path from s_i to t_i for all $i \in [\ell]$ in $G - S$?

Multiway Cut (MWC)	
Input:	A graph $G = (V, E)$, $T \subseteq V(G)$ and an integer k.
Parameter:	k
Question:	Does there exist $S \subseteq V(G) \setminus T$ such that $\lvert S\rvert \leq k$ and there is no path from t_i to t_j for all $t_i, t_j \in T$, $i \neq j$, in $G - S$?

Our results and methods

Chordal graphs are a well studied subclass of perfect graphs that contains several other graph classes such as split graphs, interval graphs, threshold graphs and block graphs. They are characterized by the property that every cycle of length 4 or more has a chord in it. Alternatively, they are the class of intersection graphs of a collection of sub-trees of a tree; and graphs that have a forest-decomposition where every bag induces a clique.

Our first result is a polynomial kernel for MULTICUT on chordal graphs when parameterized by k and ℓ. This is obtained by a sequence of reduction rules that are based on the structure of the clique-forest of the input chordal graph. First, we get rid of non-terminal simplicial vertices, which helps us bound the number of leaves and higher degree nodes in the clique-forest of G. One key step here is to ensure that each terminal vertex occurs in exactly one bag of the clique-forest decomposition. Then a marking procedure marks a bounded number of vertices in high degree bags, and deletes unmarked vertices to bound the size of high degree bags. After this, we only need to bound the lengths of degree 2 paths in the clique-forest and the number of vertices occurring in them. For bounding the first, we look at the end nodes of the degree 2 path which are either a bag containing a terminal or a high degree node in the forest, and since their sizes are bounded, it helps us mark bounded number of "interesting" degree 2 nodes on the path. Then we apply a reduction rule which deletes the "uninteresting" bags from the path while preserving the size of the min-cut between the interesting nodes. Finally we do a similar marking procedure for vertices in the bags of the degree 2, as we did for high degree nodes. Then, deleting unmarked vertices gives us a polynomial kernel for MC parameterized by k and ℓ.

▶ **Theorem 1.** *MC admits a kernel with $\mathcal{O}(k^3\ell^7)$ vertices on chordal graphs.*

We extend this result to a polynomial kernel for MWC on chordal graphs, parameterized by k alone. One of the key reduction rules here, first described by [5], shows that the number of terminals in T can be reduced to $2k$. Then, combined with a tighter analysis of our previous kernelization result, we prove the following.

▶ **Theorem 2.** *MWC admits a kernel with $\mathcal{O}(k^{13})$ vertices on chordal graphs.*

Next we present FPT algorithms for these problems on chordal graphs. These algorithms are based on two crucial ingredients. The first ingredient is a fact that there is a *unique* important $(\{v\}, X)$-separator in a chordal graph G of a fixed size k, for any $v \in V(G)$ and $X \subset V(G) \setminus \{v\}$ such that X induces a clique in G. This result uses the fact that minimal separators in a chordal graph are cliques and a lemma from [18] that bounds the number of important separators inducing cliques. Our second ingredient is the design of a *Pushing Lemma* for MC on chordal graphs based on the clique-forest decomposition of the chordal graph. These two ingredients are combined with the structure of the graph, to yield fast FPT algorithms for MC on chordal graphs parameterized by k or $k + \ell$. This is formalized in the theorems below.

▶ **Theorem 3.** *MC on chordal graphs can be solved in $\mathcal{O}(2^k \cdot (k^3 + \ell) \cdot (n + m))$ time.*

▶ **Theorem 4.** *MC on chordal graphs can be solved in $2^{\mathcal{O}(\ell \log k)} \cdot (n + m) + \ell(n + m)$ time.*

Finally, we turn to MC on split graphs. Split graphs are a subclass of chordal graphs, where the vertex set can be partitioned into an independent set and a clique. It is known that the problem remains NP-hard on split graphs [19]. We design fast FPT algorithms for it parameterized by k or ℓ. We also consider UMC on split graphs and design an FPT algorithm for it parameterized by ℓ. Let us note that, while UMC can be easily reduced to MC, the reduction does not produce a split graph. Hence, we need a slightly different algorithm for it.

▶ **Theorem 5.** *MC on split graphs can be solved in $\mathcal{O}(1.2738^k + kn + \ell(n + m))$ time.*

▶ **Theorem 6.** *MC on split graphs can be solved in $\mathcal{O}(2^\ell \cdot \ell \cdot (n + m))$ time.*

▶ **Theorem 7.** *UMC on split graphs can be solved in $\mathcal{O}(4^\ell \cdot \ell \cdot (n + m))$ time.*

2 Preliminaries

We use $[n]$ to denote the set of first n positive integers $\{1, 2, 3, \ldots n\}$. For a graph G, we denote the set of vertices of the graph by $V(G)$ and the set of edges of the graph by $E(G)$. We denote $|V(G)|$ and $|E(G)|$ by n and m respectively, where the graph is clear from context. We abbreviate an edge $\{u, v\}$ as uv sometimes. For a set $S \subseteq V(G)$, the *subgraph of G induced by S* is denoted by $G[S]$ and it is defined as the subgraph of G with vertex set S and edge set $\{\{u, v\} \in E(G) : u, v \in S\}$ and the subgraph obtained after deleting S (and the edges incident to the vertices in S) is denoted by $G - S$. For $v \in V(G)$, we will use $G - v$ to denote $G - \{v\}$ for ease of notation. All vertices adjacent to a vertex v are called neighbours of v and the set of all such vertices is called the *open* neighbourhood of v, denoted by $N_G(v)$. For a set of vertices $S \subseteq V(G)$, we define $N_G(S) = (\cup_{v \in S} N(v)) \setminus S$. We define the *closed* neighbourhood of a vertex v in the graph G to be $N_G[v] := N_G(v) \cup \{v\}$ and closed neighbourhood of a set of vertices $S \subseteq V(G)$ to be $N_G[S] := N_G(S) \cup S$. We drop the subscript G when the graph is clear from the context. We say a vertex v is *simplicial* in G if $N(v)$ forms a clique in G.

Let P be a path in the graph G on at least three vertices. We say that $\{u, v\} \in E(G)$ is a *chord* of P if $u, v \in V(P)$ but $\{u, v\} \notin E(P)$. Similarly, for a cycle C on at least four vertices, $\{u, v\} \in E(G)$ is a chord of C if $u, v \in V(C)$ but $\{u, v\} \notin E(C)$. A path P or cycle C is *chordless* if it has no chords. The *length* of a path or a cycle is the number of vertices in it. We also use P and C to denote the set of vertices or edges of the path P or cycle C respectively, when it is clear from the context.

A set $S \subseteq V(G) \setminus \{u, v\}$ is called a (u, v)-*separator* for $u, v \in V(G)$, if there is no path from u to v in $G - S$. For $X, Y \subseteq V(G)$, an (X, Y)-*separator* in G is a set $S \subseteq V(G)$ such that there is no path from x to y in $G - S$ for all $x \in X$, $y \in Y$. A set $S \subseteq V(G)$ is a *minimal separator* of a graph G, if there exist $u, v \in V(G)$ such that S is an inclusion-wise minimal (u, v)-separator.

▶ **Definition 8.** *A* forest-decomposition *of a graph G is a pair (F, β), where F is a forest and $\beta : V(F) \to 2^{V(G)}$ such that* (**i**) $\cup_{x \in V(F)} \beta(x) = V(G)$, (**ii**) *for every edge $uv \in E(G)$ there exists $x \in V(F)$ such that $\{u, v\} \subseteq \beta(x)$, and* (**iii**) *for every vertex $v \in V(G)$ the subgraph of F induced by the set $\beta^{-1}(v) := \{x \mid v \in \beta(x)\}$ is connected.*

For $x \in V(F)$, we call $\beta(x)$ the *bag* of x, and for the sake of clarity of presentation, we sometimes use x and $\beta(x)$ interchangeably. We refer to the vertices in $V(F)$ as nodes. A tree-decomposition is a forest-decomposition where F is a tree. For two adjacent nodes x_1 and x_2, $\beta(x_1) \cap \beta(x_2)$ is called *adhesion* of x_1 and x_2. For a path $P = x_1 x_2 \ldots x_{p-1} x_p$ in F, the set of adhesions on the path P refers to the set $\{\beta(x_i) \cap \beta(x_{i+1}) \colon i \in [p-1]\}$. We will state a simple property of adhesions which we will use repeatedly.

▶ **Lemma 9** (folklore). *Let (F, β) be a forest-decomposition of G, and let $P = x_0 x_1 x_2 \ldots x_p$ be a path in F such that $u \in \beta(x_0)$, $v \in \beta(x_p)$ and $\{u, v\} \cap \beta(x_i) = \emptyset$ for all $i \in \{1, \ldots, p-1\}$. Then for all $A_i := \beta(x_{i-1}) \cap \beta(x_i)$, there is no path from u to v in $G - A_i$.*

Chordal Graphs: A graph G is called *chordal* if it does not contain any chordless cycle of length at least four. It is well known that the set of chordal graphs is closed under the operation of taking induced subgraphs and contracting edges [12]. A *clique-forest* of G is a forest-decomposition of G where every bag is a maximal clique. We further insist that every bag of the clique-forest is distinct. The following lemma shows that the class of chordal graphs is exactly the class of graphs that have a clique-forest.

▶ **Lemma 10** ([12]). *A graph G is a chordal graph if and only if G has a clique-forest.*

It is also known that if G is chordal, then its clique-forest can be computed in $\mathcal{O}(m+n)$ time [11]. Observe that since every bag is a maximal clique, not only the bags are distinct in the clique-forest (F,β) of G, but also for any $x,y \in V(F)$, we have that none of $\beta(x)$ and $\beta(y)$ is a subset of the other, i.e., $\beta(x) \nsubseteq \beta(y)$ and $\beta(y) \nsubseteq \beta(x)$. Also, given a forest F and a surjective function $\beta : V(F) \to \mathcal{S}$ where $\mathcal{S} \subseteq 2^V$, such that it satisfies property 3 of Definition 8, we can associate a graph G with $V(G) = \cup_{S\in\mathcal{S}} S$ and $E(G)$ defined by $uv \in E(G)$ if and only if there exists $x \in V(F)$ such that $\{u,v\} \subseteq \beta(x)$. It is easy to see that in this case the graph G is chordal and that the bags of (F,β) correspond to the maximal cliques of G and we say that G is the chordal graph associated with the clique-forest (F,β).

We need another property of clique-forests of chordal graphs, which says that deleting some adhesion is necessary to disconnect vertices that do not occur in the same bag.

▶ **Lemma 11** ($\star$[2]). *Let (F,β) be the clique-forest of a chordal graph G, and let $P = x_0x_1x_2\ldots x_p$ be a path in F such that $u \in \beta(x_0)$, $v \in \beta(x_p)$ and $\{u,v\} \cap \beta(x_i) = \emptyset$ for all $i \in \{1,\ldots,p-1\}$. Let $A_i = \beta(x_i) \cap \beta(x_{i-1})$ be the adhesions on P for $i \in [p]$. Then for any (u,v)-separator S in G, there exists $i \in [p]$ such that $A_i \subseteq S$.*

3 A Polynomial Kernel for Multicut on Chordal graphs

In this section we will show that Multicut (MC) admits a polynomial kernel on chordal graphs parameterized by the solution size and the number of terminal pairs, that is, we will prove Theorem 1. Throughout this section, we will assume that the input graph G in an MC instance (G,T,k) is chordal, unless otherwise stated. We will use T^* to denote the set of all terminals, i.e., $T^* := \bigcup_{i\in\ell}\{s_i,t_i\}$. We also associate a measure τ with the instance (G,T,k), where $\tau := |T^*|$. We present a series of reduction rules, which will be applied in order, assuming that while applying a reduction rule, none of the previous reduction rules apply to the current instance.

▶ **Reduction Rule 1.** *Let (G,T,k) be an instance of MC. If there exist $(s_i,t_i) \in T$ such that $s_it_i \in E(G)$, say No. Let $S_i := N(s_i) \cap N(t_i)$ for all $i \in [\ell]$ and let $S := \cup_{i\in[\ell]} S_i$. Output $(G-S,T,k-|S|)$.*

The correctness of the reduction rule follows from the fact that any solution must delete S_i for all $i \in [\ell]$. Now we present a rule which deletes simplicial vertices from the graph which are not terminals.

▶ **Reduction Rule 2.** *Let (G,T,k) be an instance of MC. If there exists $v \in V(G) \setminus T^*$ such that v is a simplicial vertex in G, delete v.*

▶ **Lemma 12** ($\star$). *Reduction Rule 2 is correct.*

Now we can show that each leaf bag in the clique-forest of G must contain a terminal.

▶ **Lemma 13** ($\star$). *Let (G,T,k) be an instance of MC after applying Reduction Rule 2, and let (F,β) be a clique-forest of G. Then for each leaf node $x \in V(F)$, $\beta(x) \cap T^* \neq \emptyset$.*

[2] Proofs of the results marked with ($\star$) and full proofs of most of the theorems in Section 1 have been omitted due to space constraints. We give proof ideas for all the theorems in Section 1. Detailed proofs for all the results can be found in the appended full version of the paper.

Now we apply another reduction rule to make sure that all the terminals are part of exactly one bag in the clique-forest of G.

▶ **Reduction Rule 3.** *Let (G, T, k) be an instance of MC. Let G' be obtained from G by making $k+1$ copies of each $t \in T^*$, deleting t, and introducing a new terminal vertex t' to G' which forms a clique with the copies of t. More formally, $V(G') = (V(G) \setminus T^*) \bigcup (\cup_{t \in T^*} C_t) \bigcup T'^*$, where $C_t = \{v_t^1, \ldots, v_t^{k+1}\}$, $T'^* = \cup_{t \in T^*}\{t'\}$, $V(G) \cap (T'^* \cup (\cup_{t \in T^*} C_t)) = \emptyset$, $N_{G'}[v_t^j] = N_G(t) \cup C_t \cup \{t'\}$, and $N_{G'}(t') = C_t$ for all $t \in T'^*$ and $j \in [k+1]$. Then (G', T', k) is the new instance, where $T' = \cup_{(s_i,t_i) \in T}\{(s_i', t_i')\}$.*

▶ **Lemma 14** (⋆). *Reduction Rule 3 is correct.*

Observe that if reduction rules 1 and 2 do not apply before the application of Reduction Rule 3, then they do not apply after the application of Reduction Rule 3 as well. This happens because we do not add any simplicial vertices, and also do not introduce any edge between terminal pairs or vertex in the common neighbourhood of terminal pairs.

▶ **Lemma 15** (⋆). *Let (G, T, k) be an instance of MC obtained after applying Reduction Rule 3. For each $t \in T^*$, t belongs to exactly one bag of the clique-forest of G, and $|N(t)| = k+1$.*

▶ **Lemma 16** (⋆). *Let (G, T, k) be an instance obtained after applying Reduction Rule 3 and let (F, β) be the clique forest of G. Let $V(F) = F_1 \cup F_2 \cup F_{\geq 3}$, where F_1 is the set of leaves of F, F_2 is the set of nodes of F with degree exactly 2 and $F_{\geq 3}$ is the set of nodes of degree at least 3. Let F_T be the set of nodes that contain terminals. Then $F_1 \subseteq F_T$; $|F_T|, |F_{\geq 3}| \leq \tau$; and $|\bigcup_{x \in F_T} \beta(x)| \leq (k+2)\tau$.*

So far, we have bounded $|F_1|$, $|F_{\geq 3}|$, and $|F_T|$. We have also bounded the number of vertices appearing in the bags that contain terminals (which includes leaf bags). Next we will bound the number of vertices appearing in the bags in $F_{\geq 3}$. For that, we first define some notions.

We have established that after applying the aforementioned reduction rules, every terminal appears in exactly one bag of the clique-forest (F, β) of G. This enables us to define the notion of clique-paths between terminals. For $(s_i, t_i) \in T$, let x_{s_i} and x_{t_i} be the unique and distinct nodes in $V(F)$ that contain s_i and t_i respectively. Now, we look at the unique path between x_{s_i} and x_{t_i} in F and call it $\Pi_G(s_i, t_i)$. We will drop the subscript G if the graph is clear from the context.

Now, for each pair $(s_i, t_i) \in T$ such that $\Pi(s_i, t_i)$ is non-empty, for each bag $\beta(x)$ for $x \in \Pi(s_i, t_i)$ of degree at least 3 that does not contain a terminal, we want to mark at most $2k+2$ vertices in $\beta(x)$. Let $\Pi(s_i, t_i) := x_1 x_2 \ldots x_d$ be a nonempty path where $x_{s_i} = x_1$ and $x_{t_i} = x_d$. Let $x_p \in \Pi(s_i, t_i), p \in \{2, \ldots, d-1\}$ be an internal node of $\Pi(s_i, t_i)$ with degree at least 3 that does not contain a terminal. We define two orderings $\leq_{(s_i,t_i)}$ and $\leq_{(t_i,s_i)}$ on vertices of $\beta(x_p)$ as follows. For $u, v \in \beta(x_p)$, $u \leq_{(s_i,t_i)} v$ if and only if, for all $q \geq p$, $p, q \in [d]$, if $u \in \beta(x_q)$ then $v \in \beta(x_q)$. Similarly, for defining $\leq_{(t_i,s_i)}$, we say that $u \leq_{(t_i,s_i)} v$ if and only if, for all $q \leq p$, $p, q \in [d]$, if $u \in \beta(x_q)$ then $v \in \beta(x_q)$. In other words, the ordering represents how far along $\Pi(s_i, t_i)$ the vertices of $\beta(x_p)$ go, ranking the ones that go the farthest on either side as the largest.

Now we describe the *marking procedure*. For each bag $x \in F_{\geq 3} \setminus F_T$, for each $(s_i, t_i) \in T$ for which x is an internal vertex of $\Pi(s_i, t_i)$, we mark $k+1$ vertices which are largest in the ordering $\leq_{(s_i,t_i)}$ and call the set $M_x(s_i, t_i)$. We also mark $k+1$ vertices which are largest in the ordering $\leq_{(t_i,s_i)}$ and call that set $M_x(t_i, s_i)$. Let the set of all marked vertices inside

a bag $\beta(x)$, such that $x \in F_{\geq 3} \setminus F_T$ be $M(x) := \bigcup_{(s_i,t_i)\in T}(M_x(s_i,t_i) \cup M_x(t_i,s_i))$ and let $M := \big(\cup_{x\in F_{\geq 3}\setminus F_T} M(x)\big) \cup \big(\cup_{t\in T^*} \beta(x_t)\big)$.

Now we give the next reduction rule which will help us bound the size of bags in $F_{\geq 3}$.

▶ **Reduction Rule 4.** *Let (G,T,k) be an instance of MC where G is chordal graph and let (F,β) be the clique-forest of G. If there exists a node $x \in F_{\geq 3} \setminus F_T$ such that $\beta(x) \setminus M$ is nonempty, then delete an arbitrary vertex $v \in \beta(x) \setminus M$ from G.*

▶ **Lemma 17** (⋆). *Reduction Rule 4 is correct.*

The key idea behind the proof of this lemma is that, given any path between s_i and t_i, we can replace the deleted unmarked vertex by a pair of marked vertices to obtain another path that is present in the reduced graph.

▶ **Lemma 18** (⋆). *Let (G,T,k) be an instance of MC after applying Reduction Rule 4 exhaustively, and let (F,β) be the clique-forest of G. Let $F_{\geq 3}$ be set of nodes of F with degree at least 3. Then, $|\beta(x)| = \mathcal{O}(k\ell\tau)$ for all $x \in F_{\geq 3}$ and $|\bigcup_{x\in F_{\geq 3}} \beta(x)| = \mathcal{O}(k\ell\tau^2)$.*

Now we have bounded the number of vertices in the graph which are part of any bags in $F_1, F_{\geq 3}$, and F_T. What remains to be bounded is the number of degree 2 nodes and the number of vertices of G that appears in the bags corresponding to the degree 2 nodes. For that, first we will bound the length of a path which consists of only degree 2 nodes. To that end, we first describe a marking procedure, that marks a bounded number of degree 2 nodes.

Let $Q := x_1x_2\ldots x_q$ be a path in F such that $x_1, x_q \in F_{\geq 3} \cup F_T$ and $x_i \notin F_{\geq 3} \cup F_T$ for all $i \in \{2,3,\ldots,q-1\}$. That is, Q is a path in F with all internal nodes having degree 2 and not containing any terminal, while the first and the last nodes either have degree at least 3 or they contain a terminal. Now, we mark some nodes in Q as $D(Q) \subseteq V(Q)$. For that, let $B_1 := \beta(x_1)$ and let $B_q = \beta(x_q)$. Suppose x_i and x_{i+1}, $i \in [q-1]$ are such that $B_1 \cap (\beta(x_i) \setminus \beta(x_{i+1})) \neq \emptyset$. In such a case, we add x_i and x_{i+1} to $D(Q)$. Similarly, if x_j and x_{j-1}, $j \in \{2,3,\ldots,q\}$ are such that $B_q \cap (\beta(x_j) \setminus \beta(x_{j-1})) \neq \emptyset$, then we add x_j and x_{j-1} to $D(Q)$. Observe that for any pair of marked nodes x_i and x_{i+1} (or x_j and x_{j-1}), we can find a vertex $v_i \in x_1$ (or $v_j \in x_q$), such that the nodes x_i and x_{i+1} (or x_j and x_{j-1}) differ on v_i (or on v_j). No other two consecutive nodes of Q differ on v_i or v_j due to property **(iii)** of Definition 8. This shows that for every vertex in $\beta(x_1) \cup \beta(x_q)$, at most two nodes are marked by the procedure.

▶ **Observation 1.** *Let $Q := x_1x_2\ldots x_q$ be a path in F such that $x_1, x_q \in F_{\geq 3} \cup F_T$ and $x_i \notin F_{\geq 3} \cup F_T$ for all $i \in \{2,3,\ldots,q-1\}$. Let $B_1 := \beta(x_1)$ and $B_q := \beta(x_q)$. Let $Q' := y_1y_2\ldots y_r$ be a subpath of Q such that $y_1, y_r \in D(Q)$ but $y_i \notin D(Q)$ for all $i \in \{2,3,\ldots,r-1\}$. Then $B_1 \cap \beta(y_1) = B_1 \cap \beta(y_2) = \ldots = B_1 \cap \beta(y_r)$ and $B_q \cap \beta(y_1) = B_q \cap \beta(y_2) = \ldots = B_q \cap \beta(y_r)$.*

We next state a lemma, which shows that it is necessary and sufficient to pick one adhesion in the solution for every terminal pair, and that any minimal solution can be looked at as a collection of adhesions of the clique-forest, at most one of which comes from any degree 2 path in the clique-forest.

▶ **Lemma 19** (⋆). *Let S be a minimal solution to an instance (G,T,k) of MC and let (F,β) be the clique-forest of G. Then, there exists $S_i \subseteq S$ for all $(s_i,t_i) \in T$, such that*

1. *S_i is an adhesion on $\Pi(s_i,t_i)$,*
2. *$S = \bigcup_{(s_i,t_i)\in T} S_i$, and*
3. *if $Q := x_1x_2\ldots x_q$ is a path in F such that $x_z \notin F_{\geq 3} \cup F_T$ for all $z \in \{2,3,\ldots,q-1\}$, then at most one adhesion from the path Q is picked as S_i for some pair(s) $(s_i,t_i) \in T$.*

Observe that the adhesions picked by Lemma 19 for a minimal solution for different pairs might not be distinct or disjoint. Now we are ready to give the reduction rule which decreases the number of degree 2 nodes.

▶ **Reduction Rule 5.** *Let (G, T, k) be an instance of MC and (F, β) be the clique-forest of G. Let $Q := x_1x_2 \dots x_q$ be a path in F such that $x_1, x_q \in F_{\geq 3} \cup F_T$ and $x_\gamma \notin F_{\geq 3} \cup F_T$ for all $\gamma \in \{2, 3, \dots, q-1\}$. Let $Q' := y_1, \dots y_r$ be a subpath of Q of length at least 3 such that $y_1, y_r \in D(Q)$ but $y_\alpha \notin D(Q)$ for all $\alpha \in \{2, 3, \dots, r-1\}$. Let $W = \cup_{x \in Q'} \beta(x)$. Consider an auxiliary graph G^* with vertex set $W \cup \{s, t\}$ where s and t are new vertices such that $N_{G^*}(s) = \beta(y_1)$, $N_{G^*}(t) = \beta(y_r)$ and $G^*[W] = G[W]$. Let the size of a minimum vertex cut between s and t in G^* be c. Let $z := c - |\beta(y_1) \cap \beta(y_r)|$ and let $U = \{u_1, \dots, u_z\}$ be a set of new vertices such that $U \cap V(G) = \emptyset$. To get a new clique-forest (F', β'), delete y_α for each $\alpha \in \{2, 3, \dots, q-1\}$, make y_1 and y_r adjacent in F' while preserving all other adjacencies of F, and put $\beta'(y_1) = \beta(y_1) \cup U$, $\beta'(y_r) = \beta(y_r) \cup U$ and $\beta'(x) = \beta(x)$ for all $x \notin V(Q')$. Let G' be the chordal graph corresponding to (F', β'). Output (G', T, k).*

▶ **Lemma 20** (⋆). *Reduction Rule 5 is well defined. That is, $z \geq 0$, F' is a forest, and (F', β') satisfies property* (**iii**) *of Definition 8.*

Now we prove a lemma that relates the adhesions of the input and output instances of the reduction rule, and then prove the correctness of the reduction rule.

▶ **Lemma 21** (⋆). *Let (G', T, k) be obtained by applying Reduction Rule 5 on (G, T, k). Let $A^* := \beta'(y_1) \cap \beta'(y_r)$, and let $\mathcal{A}$ be set of adhesions on the path Q'. Let $(s_i, t_i) \in T$. Then for any adhesion $A = \beta(z_1) \cap \beta(z_2)$ on $\Pi_G(s_i, t_i)$ such that $A \notin \mathcal{A}$, $A = \beta'(z_1) \cap \beta'(z_2)$ is also an adhesion on $\Pi_{G'}(s_i, t_i)$. Similarly, for any adhesion $A' = \beta'(z_1) \cap \beta'(z_2)$ on $\Pi_{G'}(s_i, t_i)$ such that $A' \neq A^*$, $A' = \beta(z_1) \cap \beta(z_2)$ is also an adhesion on $\Pi_G(s_i, t_i)$.*

▶ **Lemma 22** (⋆). *Reduction Rule 5 is correct.*

▶ **Lemma 23** (⋆). *Let (G, T, k) be an instance obtained after exhaustively applying Reduction Rule 5 and (F, β) be the clique-forest of G. Then $|V(F)| = \mathcal{O}(k\ell\tau^2)$.*

Now, we are ready to give the final reduction rule which would bound the size of the graph. For that, we make use of the marking procedure defined for Reduction Rule 4 once again. Let $F_2 = V(F) \setminus (F_{\geq 3} \cup F_T)$ where (F, β) is the clique-forest of G.

For each pair $(s_i, t_i) \in T$ such that $\Pi(s_i, t_i)$ is non-empty, for each bag $x \in F_2$ that is an internal node of $\Pi(s_i, t_i)$, we would mark at most $2k + 2$ vertices in $\beta(x)$. Let $\Pi(s_i, t_i) := x_1x_2 \dots x_d$ be a nonempty path where $x_{s_i} = x_1$ and $x_{t_i} = x_d$. Let $x_p \in \Pi(s_i, t_i), p \in \{2, 3, d-1\}$ be an internal node of $\Pi(s_i, t_i)$ such that $x_p \in F_2$. We define two orderings $\leq_{(s_i,t_i)}$ and $\leq_{(t_i,s_i)}$ on $\beta(x)$ for all $x \in F_2$ as before. That is, for $u, v \in \beta(x)$, $u \leq_{(s_i,t_i)} v$ if and only if, for all $q \geq p$, $p, q \in [d]$, if $u \in \beta(x_q)$ then $v \in \beta(x_q)$. Similarly, for $\leq_{(t_i,s_i)}$, we say that $u \leq_{(t_i,s_i)} v$ if and only if, for all $q \leq p$, $p, q \in [d]$, if $u \in \beta(x_q)$ then $v \in \beta(x_q)$. We then mark $k + 1$ vertices which are highest in each of these orderings exactly as before and call them $Z_x(s_i, t_i)$ and $Z_x(t_i, s_i)$ respectively. Let $Z(x) := \bigcup_{(s_i,t_i) \in T} (Z_x(s_i, t_i) \cup Z_x(t_i, s_i))$ and let $Z := \big(\cup_{x \in F_2} Z(x)\big) \cup \big(\cup_{x \in F_T \cup F_{\geq 3}} \beta(x)\big)$.

▶ **Reduction Rule 6.** *Let (G, T, k) be an instance of MC and let (F, β) be the clique-forest of G. If there exists a node $x \in F_2$ such that $\beta(x) \setminus Z$ is nonempty, then delete an arbitrary vertex $v \in \beta(x) \setminus Z$ from G.*

The proof of correctness of Reduction Rule 6 is exactly the same as proof of Lemma 17. Now we are ready to prove the final lemma that bounds the size of the instance.

▶ **Lemma 24.** *Let (G, T, k) be an instance of MC after exhaustive application of Reduction Rule 6. Then $|V(G)| = \mathcal{O}(k^3\ell^3\tau^4)$.*

Proof. Let (F, β) be the clique-forest of G. We already know due to Lemmas 16, 18, and 23 that $|V(F)| = \mathcal{O}(k\ell\tau^2)$ and $|\beta(x)| = \mathcal{O}(k\ell\tau)$ for all $x \in F_T \cup F_{\geq 3}$. We also know that $|F_2| = \mathcal{O}(k\ell\tau^2)$. So if we can show $|\beta(x)| = \mathcal{O}(k^2\ell^2\tau^2)$ for all $x \in F_2$, this would prove the lemma. For that, we want to show that $|Z| = \mathcal{O}(k^2\ell^2\tau^2)$. This would mean that $|\beta(x)| = \mathcal{O}(k^2\ell^2\tau^2)$ for all $x \in F_2$, as otherwise Reduction Rule 6 would apply.

Now, for a bag $\beta(x)$ such that $x \in F_2$, we want to give a bound for Z_x. We mark at most $2k+2$ vertices for each pair of terminals $(s_i, t_i) \in T$. That gives us $|Z_x| \leq (2k+2)\ell$. Now, combining that with $|F_2| = \mathcal{O}(k\ell\tau^2)$, we get that $|\cup_{x\in F_2} Z(x)| = \mathcal{O}(k^2\ell^2\tau^2)$. We already know that $|\cup_{x\in F_T\cup F_{\geq 3}} \beta(x)| = \mathcal{O}(k\ell\tau^2)$, so this gives us $|Z| = \mathcal{O}(k^2\ell^2\tau^2)$ as desired. ◀

The proof of Theorem 1 follows from Lemma 24, since the reduction rules can be applied in polynomial time, and we know that $\tau \leq 2\ell$.

4 Kernel for Multiway Cut on Chordal Graphs

In this section we will give a polynomial kernel for Multiway Cut (MWC) on chordal graphs parameterized by k alone, that is, we will prove Theorem 2. The following preprocessing decreases the number of terminals to a linear function of the solution size.

▶ **Lemma 25** ([5]). *Given an instance (G', T', k') of Multiway Cut, in polynomial time we can arrive at an equivalent instance (G, T, k) such that $T \subseteq T'$, $k \leq k'$, $|T| \leq 2k$ and G is obtained from G' by performing one of the following two operations iteratively.*

1. *Taking an induced subgraph, and*
2. *contracting an edge.*

▶ **Reduction Rule 7.** *Apply Lemma 25 to get an instance (G, T, k) such that $|T| \leq 2k$.*

The correctness follows from Lemma 25 and the fact that chordal graphs are closed under taking induced subgraphs and contracting edges.

Now, for proving Theorem 2, we first apply Reduction Rule 7 and then reduce the MWC instance obtained to an MC instance. Then we kernelize the MC instance and get a size bound using Lemma 24. Finally, we reduce the kernelized MC instance back to a MWC instance. We can do that safely because the kernelization procedure for MC preserves one-to-one correspondence of terminals with respect to the initial instance.

5 FPT algorithms for Multicut on Chordal Graphs

In this section, we design FPT algorithms for Multicut (MC) on chordal graphs. In particular, we prove Theorems 3 and 4. The most crucial ingredient for the proofs in this section is the fact that in any graph (not necessarily chordal), amongst all the important $(\{v\}, W)$-separators that induce a clique, where $W \subset V(G)$ and $v \in V(G) \setminus W$, there is at most one important separator of a fixed size k [18]. Below we state a classical result concerning chordal graphs that we use to design our algorithms.

▶ **Proposition 26** ([9]). *Every minimal separator in a chordal graph is a clique.*

▶ **Corollary 27** ($\star$). *Let G be a chordal graph, and let $X \subseteq V(G)$ such that $G[X]$ is a clique. Then every minimal $(\{v\}, X)$-separator in G is a clique.*

Let G be a graph and $X, Y \subseteq V(G)$. Let $S \subseteq V(G)$ be an (X,Y)-separator in G and let R denote the set of vertices reachable from $X \setminus S$ in $G - S$ (if X (resp. Y) is a singleton set then $S \cap X = \emptyset$ (resp. $S \cap Y = \emptyset$)). Then S is called an *important* (X,Y)*-separator* if it is inclusion-wise minimal and there is no (X,Y)-separator S' such that $|S'| \leq |S|$ and $R \subset R'$, where R' is the set of vertices reachable from X in $G - S'$.

▶ **Lemma 28** ([18], Lemma 3.16)**.** *For any graph G (not necessarily chordal), $W \subset V(G)$, and $v \in V(G) \setminus W$, there is at most one important $(\{v\}, W)$-separator of size exactly k inducing a clique.*

While the lemma in [18] is stated as there are at most k important $(\{v\}, W)$-separators of size at most k that induce a clique, the proof follows by proving the statement in Lemma 28.

▶ **Lemma 29.** *For a positive integer k, a chordal graph G, $v \in V(G)$, and $X \subseteq V(G)$ such that $G[X]$ is a clique, there is at most one $(\{v\}, X)$-important separator of size k in G.*

The proof of Lemma 29 follows from Corollary 27 and Lemma 28.

▶ **Proposition 30** ([3, 17])**.** *Given a graph G and $X, Y \subseteq V(G)$ and a positive integer k, the set $\mathcal{S}_k$ of all important (X,Y)-separators of G of size at most k can be computed in time $\mathcal{O}(|\mathcal{S}_k| \cdot k^2 \cdot (n+m))$.*

We now step towards stating and proving our pushing lemma that, together with Lemma 29, leads to the design of a branching algorithm for MC on chordal graphs. Let (G, T, k) be an instance of MC where G is a chordal graph. Consider the clique forest, say (F, β), of G, defined in Lemma 10. Without loss of generality, let G be a connected graph. Thus, it will be safe to assume that F is a tree. Root the tree F at an arbitrary node. Also, we will assume (G, T, k) to be reduced with respect to Reduction Rules 1 and 3 for the rest of this section. Note that Reduction Rule 1 can be applied in $\mathcal{O}(\ell \cdot (n+m))$ time and Reduction Rule 3 can be applied in $\mathcal{O}(k \cdot (n+m))$ time. Also both these rules are applied only once in the course of the algorithms. Thus the application of these rules exhaustively contribute a factor of $\mathcal{O}((k+\ell)(n+m))$ to the running times of our algorithms. From Lemma 15, for each $(s_i, t_i) \in T$, s_i and t_i belong to exactly one bag of the clique-forest (F, β). We denote the unique bag of F containing s_i (resp. t_i), as x_{s_i} (resp. x_{t_i}). Let x_i^{lca} denote the bag which is the unique least common ancestor of x_{s_i} and x_{t_i} in the rooted tree F.

▷ Claim 31. Let (G, T, k) be an instance of MC where G is a chordal graph. Let S be any solution to the instance (G, T, k). Let (F, β) be a rooted clique-forest of G. For each pair $(s_i, t_i) \in T$, every (s_i, t_i)-separator contains an adhesion on the unique x_{s_i} to x_{t_i} path in F.

The proof of the above claim follows from Lemma 11.

▶ **Lemma 32** (⋆, Pushing Lemma for MC on Chordal Graphs)**.** *Let (G, T, k) be an instance of MC where G is a connected chordal graph, and let (F, β) be a rooted clique tree of G as defined above. Let y denote the root bag of F. Let $(s_p, t_p) \in T$ be such that x_p^{lca} is deepest in the rooted tree F, that is, x_p^{lca} is such that $dist_F(y, x_p^{lca}) = \max\{dist_F(y, x_i^{lca}) : i \in [\ell]\}$, where $dist_F(y, x_i^{lca})$ denote the distance between y and x_i^{lca} in F. Then there is a solution to (G, T, k) that contains either an important $(\{s_p\}, \beta(x_p^{lca}))$-separator or an important $(\{t_p\}, \beta(x_p^{lca}))$-separator of size at most k.*

Observe that none of important $(\{s_p\}, \beta(x_p^{lca}))$-separator and important $(\{t_p\}, \beta(x_p^{lca}))$-separator can contain a terminal, as these are minimal separators in G and each of the

terminals occur in only one bag due to Reduction Rule 3, and hence do not belong to any adhesions in G, which are also minimal separators of G.

The algorithms of Theorem 3 and 4 are based on a branching algorithm that branches on important separators described in Lemma 32.

Description of the algorithm for MC on chordal graphs (Algorithm 1): Let (G, T, k) be an instance of MC where G is a connected chordal graph. Let (F, β) be a rooted tree-decomposition of G. Let $\mathcal{I}_s$ (resp. $\mathcal{I}_t$) be the collection of all important $(\{s_p\}, \beta(x_p^{lca}))$-separator (resp. $(\{t_p\}, \beta(x_p^{lca}))$-separator) of size at most k. The algorithm branches on the sets in $\mathcal{I}_s \cup \mathcal{I}_t$. That is, it reduces the instance (G, T, k) to the set of instances $(G-I, T', k-|I|)$, where $I \in \mathcal{I}_s \cup \mathcal{I}_t$ and T' denotes the set of pairs of terminals with are connected in $G - I$.

Proof of Theorem 3. The correctness of Algorithm 1 follows from Lemma 32. To prove the theorem, we show that it runs in $\mathcal{O}(2^k \cdot (k^3 + \ell) \cdot (n + m))$ time. Let $T(k)$ denote the number of leaves in the branching tree rooted at an instance where the budget parameter is k. Since, from Lemma 29, there is a unique important $(\{s_p\}, \beta(x_p^{lca}))$-separator (and $(\{t_p\}, \beta(x_p^{lca}))$-separator) of a fixed size, from the description of the algorithm we get the following recurrence: $T(k) \leq 2\sum_{i \in [k]} T(k-i)$, $T(1) = 1$. Using induction one can show that $T(k) \leq 2^{k+1}$. Thus, the number of nodes in the branching tree are at most $2 \cdot 2^{k+1}$. Also the time spent at each node is equal to the time taken by Reduction Rules 1 and 3, which is $\mathcal{O}((k + \ell)(n + m))$, plus time taken by the algorithm of Proposition 30, which is $\mathcal{O}(k^3 \cdot (n+m))$ because $\mathcal{S}_k$ in the proposition has size at most k from Lemma 29. The desired running time thus follows. ◀

We give the following reduction rule, which helps in proving Theorem 4.

▶ **Reduction Rule 8.** *Let (G, T, k) be an instance of MC. If there exists $(s_i, t_i) \in T$ such that there is no path from s_i to t_i in G, then delete (s_i, t_i) from T, that is, the reduced instance is $(G, T \setminus \{(s_i, t_i)\}, k)$.*

The correctness of Reduction Rule 8 is easy to see and it can be applied in $\mathcal{O}(\ell(n + m))$ time. Theorem 4 is obtained by applying Reduction Rule 8 after each branching step of the algorithm, hence decreasing the size of the terminal set, and solving a recurrence reflecting this. That is, $T(k, \ell) \leq \sum_{i \in [2k]} T(k, \ell - 1)$, $T(k, 0) = 1$, which solves to $T(k, \ell) \leq (2k)^\ell$.

6 FPT algorithms for Multicut on Split Graphs

In this section, we design FPT algorithms for Multicut on split graphs. In particular, we prove Theorems 5, 6, and 7. Recall that a graph G is a split graph if and only if $V(G)$ can be partitioned into two parts: C and I, such that the set $G[C]$ is a clique and $G[I]$ is an independent set. It is known that given a split graph G, such a partition can be obtained in time $\mathcal{O}(n + m)$ [8]. Henceforth, we assume that such a partition of the input split graph is given to us. In what follows, we denote an instance of Multicut or Unrestricted Multicut on split graphs by $(G = (C, I), T, k)$. Note that split graphs are also chordal graphs, hence the reduction rules designed for chordal graphs can also be applied on split graphs. We assume that the input instance $(G = (C, I), T, k)$ is reduced with respect to Reduction Rule 1 and Reduction Rule 8. Note that the exhaustive application of these reduction rules contribute $\mathcal{O}(\ell \cdot (n + m))$ to the running time of our algorithms.

▶ **Lemma 33 ($\star$).** *Let $(G = (C, I), T, k)$ be an instance of MC which is reduced with respect to Reduction Rule 1 and 8. Then, for each $(s_i, t_i) \in T$, $s_i, t_i \in I$.*

▶ **Lemma 34** (⋆). *If (G, T, k) is an instance of MC on split graphs that is reduced with respect to Reduction Rules 1 and 8, then for each $(s_i, t_i) \in T$, the length of any shortest path from s_i to t_i in G is 4. Also, the internal vertices of this shortest path belong to C.*

Let $(G = (C, I), T, k)$ be an instance of MC where G is a split graph. For each $(s_i, t_i) \in T$, we associate a set of pairs of the vertices in C as follows. For each $i \in [\ell]$, we now define $\mathcal{P}_i \subseteq C \times C$. A pair $(u, v) \in C \times C$, $u \neq v$, belongs to $\mathcal{P}_i$, if $s_i u v t_i$ is a path in G.

▶ **Lemma 35** (⋆). *Let $(G = (C, I), T, k)$ be an instance of MC on split graphs. For each $(s_i, t_i) \in T$, let $\mathcal{P}_i$ be as defined above. Then S is a multicut for the instance (G, T, k) if and only if S contains a vertex from each of the pairs in $\mathcal{P}_i$, for each $i \in [\ell]$.*

The proof of Theorem 5 follows by Lemma 35 and reducing the MC instance to a Vertex Cover instance using Lemma 35 and then solving it using the algorithm in [2].

▶ **Lemma 36** (⋆). *Let $(G = (C, I), T, k)$ be an instance of MC that is reduced with respect to Reduction Rules 1 and 8. Let S be a multicut for $(G = (C, I), T, k)$. For any $(s_i, t_i) \in T$, either $N(s_i) \subseteq S$ or $N(t_i) \subseteq S$.*

Proof of Theorem 6. We design a branching algorithm for MC on split graphs. Let $(G = (C, I), T, k)$ be an instance of MC that is reduced with respect to Reduction Rules 1 and 8. Pick a pair $(s_i, t_i) \in T$. From Lemma 36, we know at least one of the following definitely hold: either $N(s_i)$ belongs to the solution or $N(t_i)$ belongs to the solution. Thus, we branch on the following two instances: $(G - N(s_i), T_1 = T - s_i, k - |N(s_i)|)$ and $(G - N(t_i), T_2 = T - t_i, k - |N(t_i)|)$, where $T - s_i$ (similarly $T - t_i$) denote the set of terminal pairs in T that do not contain s_i (or t_i). Since the deletion of the neighbours of s_i (resp. t_i) isolates s_i (resp. t_i) in the resulting graph, the correctness of the algorithm follows from Lemma 36. Also, $|T_1|, |T_2| < \ell$, as $(s_i, t_i) \in T$ but, $(s_i, t_i) \notin T_1$ and $(s_i, t_i) \notin T_2$.

Since we stop when $T = \emptyset$, the depth of the branching tree of this branching algorithm is at most ℓ. Since at each time, we branch in two branches, the number of leaves in this branching tree is at most 2^ℓ. Also, the time taken at each node (which is equal to checking if T is empty and if it is not empty, then computing the two instances to recurse on) is $\mathcal{O}(n+m)$, and the reduction rules can be applied in $\mathcal{O}(\ell \cdot (n+m))$ time, we get an algorithm with running time $\mathcal{O}(2^\ell \cdot \ell \cdot (n+m))$. ◀

The algorithm of Theorem 7 is similar to the algorithm of Theorem 6, except in this case we also branch on the possibilities of including s_i or t_i in the solution.

References

1. Nicolas Bousquet, Jean Daligault, and Stéphan Thomassé. Multicut is FPT. *SIAM J. Comput.*, 47(1):166–207, 2018.
2. Jianer Chen, Iyad A Kanj, and Ge Xia. Improved upper bounds for vertex cover. *Theoretical Computer Science*, 411(40-42):3736–3756, 2010.
3. Jianer Chen, Yang Liu, and Songjian Lu. An improved parameterized algorithm for the minimum node multiway cut problem. *Algorithmica*, 55(1):1–13, 2009.
4. Marek Cygan, Stefan Kratsch, Marcin Pilipczuk, Michał Pilipczuk, and Magnus Wahlström. Clique cover and graph separation: New incompressibility results. *TOCT*, 6(2):6:1–6:19, 2014.
5. Marek Cygan, Marcin Pilipczuk, Michał Pilipczuk, and Jakub Onufry Wojtaszczyk. On multiway cut parameterized above lower bounds. *TOCT*, 5(1):3:1–3:11, 2013.
6. Gruia Călinescu, Cristina G. Fernandes, and Bruce Reed. Multicuts in unweighted graphs and digraphs with bounded degree and bounded tree-width. *Journal of Algorithms*, 48(2):333–359, 2003.

7 E. Dahlhaus, D. Johnson, C. Papadimitriou, P. Seymour, and M. Yannakakis. The complexity of multiterminal cuts. *SIAM Journal on Computing*, 23(4):864–894, 1994.

8 Elias Dahlhaus. Parallel algorithms for hierarchical clustering and applications to split decomposition and parity graph recognition. *J. Algorithms*, 36(2):205–240, 2000.

9 Reinhard Diestel. Graph theory. 2005. *Grad. Texts in Math*, 101, 2005.

10 Fedor V. Fomin, Daniel Lokshtanov, Saket Saurabh, and Meirav Zehavi. *Kernelization: Theory of Parameterized Preprocessing*. Cambridge University Press, 2019.

11 Philippe Galinier, Michel Habib, and Christophe Paul. Chordal graphs and their clique graphs. In Manfred Nagl, editor, *Graph-Theoretic Concepts in Computer Science, 21st International Workshop, WG '95, Aachen, Germany, June 20-22, 1995, Proceedings*, volume 1017 of *Lecture Notes in Computer Science*, pages 358–371. Springer, 1995.

12 M. C. Golumbic. *Algorithmic Graph Theory and Perfect Graphs*. Academic Press, New York, 1980.

13 Jiong Guo, Falk Hüffner, Erhan Kenar, Rolf Niedermeier, and Johannes Uhlmann. Complexity and exact algorithms for vertex multicut in interval and bounded treewidth graphs. *European Journal of Operational Research*, 186(2):542–553, 2008.

14 Bart M. P. Jansen, Marcin Pilipczuk, and Erik Jan van Leeuwen. A Deterministic Polynomial Kernel for Odd Cycle Transversal and Vertex Multiway Cut in Planar Graphs. In Rolf Niedermeier and Christophe Paul, editors, *36th International Symposium on Theoretical Aspects of Computer Science (STACS 2019)*, volume 126 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 39:1–39:18, Dagstuhl, Germany, 2019. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik.

15 Philip N. Klein and Dániel Marx. Solving planar k -terminal cut in $o(n^{c\sqrt{k}})$ time. In *Automata, Languages, and Programming - 39th International Colloquium, ICALP 2012, Warwick, UK, July 9-13, 2012, Proceedings, Part I*, volume 7391 of *Lecture Notes in Computer Science*, pages 569–580. Springer, 2012.

16 Stefan Kratsch and Magnus Wahlström. Representative sets and irrelevant vertices: New tools for kernelization. In *53rd Annual IEEE Symposium on Foundations of Computer Science, FOCS 2012, New Brunswick, NJ, USA, October 20-23, 2012*, pages 450–459. IEEE Computer Society, 2012.

17 Dániel Marx. Parameterized graph separation problems. *Theor. Comput. Sci.*, 351(3):394–406, 2006.

18 Dániel Marx and Igor Razgon. Fixed-parameter tractability of multicut parameterized by the size of the cutset. *SIAM J. Comput.*, 43(2):355–388, 2014.

19 Charis Papadopoulos. Restricted vertex multicut on permutation graphs. *Discrete Applied Mathematics*, 160(12):1791–1797, 2012.