Quick Separation in Chordal and Split Graphs

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

Fahad Panolan
Indian Institute of Technology Hyderabad, Hyderabad, India. fahad@iith.ac.in

Ashutosh Rai
Faculty of Mathematics and Physics, Charles University, Prague, Czech Republic.
ashutosh@kam.mff.cuni.cz

Saket Saurabh
Faculty of Mathematics and Physics, Charles University, Prague, Czech Republic.
ashutosh@kam.mff.cuni.cz

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 $\ell$ 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 $\mathcal{O}(k^3 n^{\omega(1)})$ and $\mathcal{O}(n^{\omega(1)})$, 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 $\mathcal{O}(k^3 \ell^2)$ vertices. MULTIWAY CUT on chordal graphs admits a polynomial kernel with $\mathcal{O}(k^{12})$ vertices.

2. MULTICUT on chordal graphs can be solved in time $\min\{\mathcal{O}(2^k \cdot (k^3 + \ell) \cdot (n + m)),$ $\mathcal{O}(2^{(\ell + k) \log k} \cdot (n + m) + \ell (n + m))\}$. Hence MULTICUT on chordal graphs parameterized by the number of terminals is in XP.

3. UNRESTRICTED MULTICUT on split graphs can be solved in time $\min\{\mathcal{O}(1.2738 \cdot k n + \ell (n + m),\mathcal{O}(2^{\ell} \cdot (n + m)))\}$.

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.CVIT.2016.23

Full version of the MFCS 2020 submission.

© John Q. Public and Joan R. Public;
Licensed under Creative Commons License CC-BY
42nd Conference on Very Important Topics (CVIT 2016).
Editors: John Q. Open and Joan R. Access; Article No. 23; pp. 23:1–23:21
Leibniz International Proceedings in Informatics
Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany
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 \( \ell \) terminal vertex pairs \( (s_i, t_i), i \in \{1, \ldots, \ell\} \), and the goal is to output a minimum sized vertex subset \( S \subseteq V(G) \setminus \{s_i, t_i : i \in \{1, \ldots, \ell\}\} \) 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 \( \text{NP} \)-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 \( 4^k n^{O(1)} \) was designed by Chen, Liu, and Lu [3], and later this was improved to \( 2^{k \sqrt{n}} n^{O(1)} \) by Cygan et al. [5]. In fact, the algorithm of Cygan et al. [5] also gives a \( 4^k \times LP n^{O(1)} \) time algorithm for **Multiway Cut**, where \( LP \) is the optimum LP solution 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 \( 2^{O(k^3)} n^{O(1)} \).

These problems are very well studied from the perspective of kernelization as well. For **Unrestricted Multiway Cut**, a randomized polynomial kernel with \( O(k^3) \) 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 \( O(k^{\sqrt{\ell}}) \) and \( O(k^{\sqrt{|T|} + 1}) \) 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 kernel with parameter \( k \), then \( \text{co-NP} \subseteq \text{NP/poly} \) 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 \( \ell \) is not ruled out for general graphs.

Obtaining an algorithm faster than \( 2^{O(k^3)} n^{O(1)} \) 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.

---

\(^1\) 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.
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 $n^{O(\ell)}$ [7].
This running time is improved to $2^{O(\ell \sqrt{\log \ell})}$ and a matching lower bound under ETH is
provided by Klein and Marx [15]. Note that this is not true in general graphs, as MULTICUT
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 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:* $k, \ell$

*Question:* Does there exist $S \subseteq V(G) \setminus \cup_{i \in \ell} \{s_i, t_i\}$ such that $|S| \leq k$ 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 $|S| \leq k$ 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 sub-class 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 $\ell$. 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 degree 2, as we did for high degree nodes. Then, deleting unmarked vertices gives us a polynomial kernel for MC parameterized by \( k \) and \( \ell \).

**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 \( X \subseteq V(G) \setminus \{v\} \) 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 \( ([v], W) \)-separators inducing cliques, where \( W \subseteq V(G) \) and \( v \in V(G) \setminus W \). 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 \( 2^{\mathcal{O}(\ell \log k)} \cdot \ell \cdot (n + m) + \ell(n + m) \) time.

Finally, we turn to MC on split graphs. Split graphs are a sub-class 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 \( \ell \). We also consider UMC on split graphs and design FPT algorithm for it parameterized by \( \ell \). 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 \( N_G(S) = \bigcup \{N_G(v) : v \in S\} \). 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 $G$. For $C \subseteq V(G)$, if $G[C]$ is connected and $N(C) = \emptyset$, then we say that $G[C]$ is a connected component of $G$.

A path $P$ in a graph $G$ is a subgraph of $G$ where $V(P) = \{x_1, x_2, \ldots, x_{\ell}\} \subseteq V(G)$ and $E(P) = \{(x_1, x_2), (x_2, x_3), \ldots, (x_{\ell-1}, x_{\ell})\} \subseteq E(G)$ for some $\ell \in [n]$. We denote it by $P := x_1x_2 \ldots x_{\ell}$. The vertices $x_1$ and $x_{\ell}$ are called endpoints of the path $P$ and the remaining vertices in $V(P)$ are called internal vertices of $P$. The length of a path is the number of vertices in it. We also say that $P$ is a path between $x_1$ and $x_{\ell}$. A cycle $C$ in $G$ is a subgraph of $G$ where $V(C) = \{x_1, x_2, \ldots, x_{\ell}\} \subseteq V(G)$ and $E(C) = \{(x_1, x_2), (x_2, x_3), \ldots, (x_{\ell-1}, x_{\ell}), (x_{\ell}, x_1)\} \subseteq E(G)$, i.e., it is a path with an additional edge between $x_1$ and $x_{\ell}$. 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.

Let us note that any chordless cycle has length at least 4. 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 walk from $u$ to $v$ in $G$ is a sequence $W := v_1v_2\ldots v_m$ of vertices of $G$ such that $v_1 = u$, $v_m = v$, and for all $i \in [m-1]$, either $v_i = v_{i+1}$, or $v_iv_{i+1} \in E(G)$. Observe that if there exists a walk from $u$ to $v$ in $G$, this implies that there is also a path from $u$ to $v$ in $G$.

A set $S \subseteq V(G)$ \{u, v\} is called a $(u, v)$-separators 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 inclusion-wise minimal $(u, v)$-separator.

**Definition 8.** A forest-decomposition of a graph $G$ is a pair $(F, \beta)$, where $F$ is a forest and $\beta : V(F) \to 2^{V(G)}$ such that

- $\bigcup_{x \in V(F)} \beta(x) = V(G)$,
- for every edge $uv \in E(G)$ there exists $x \in V(F)$ such that $\{u, v\} \subseteq \beta(x)$, and
- 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 $P = x_1x_2 \ldots x_{p-1}x_p$ in $F$, the set of adhesions on the path $P$ refers to the set $\{\beta(x_i) \cap \beta(x_{i+1}) \mid i \in [p-1]\}$. We will state a simple property of adhesions which we will use repeatedly.

**Lemma 9 (folklore).** Let $(F, \beta)$ be a forest-decomposition of $G$, 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 $i \in \{1, \ldots, p-1\}$. 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 $O(m + n)$ time [11]. Observe that since every bag is a maximal clique, not only the bags are distinct in the clique-forest $(F, \beta)$ 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 $\beta : V(F) \rightarrow S$ where $S \subseteq 2^V$, such that it satisfies property 3 of Definition 8, we can associate a graph $G$ with $V(G) = \cup_{S \subseteq S} S$ 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 $(F, \beta)$ correspond to the maximal cliques of $G$ and we say that $G$ is the chordal graph associated with the clique-forest $(F, \beta)$.

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.

\textbf{Lemma 11.} Let $(F, \beta)$ 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 $A_i \subseteq S$.

\textbf{Proof.} For the sake of contradiction, suppose that $S$ does not contain any adhesion on $P$. This gives that $A_i \setminus S$ is nonempty for all $i \in [p]$. Let $v_i$ be an arbitrary vertex in $A_i \setminus S$. Since $A_i \cap A_{i+1} \subseteq \beta(x_i)$, either $v_i = v_{i+1}$ or $v_iv_{i+1} \in E(G)$ for all $i \in [p-1]$. This means that $W := uv_1v_2 \ldots v_pv$ is a walk in $G - S$, and this would imply there is a path from $u$ to $v$ in $G - S$, a contradiction. \hfill $\blacksquare$

\section{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 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 $T^\ast$ to denote the set of all terminals, i.e., $T^\ast := \bigcup_{i \in \ell} \{s_i, t_i\}$. We also associate a measure $\tau$ with the instance $(G, T, k)$, where $\tau := |T^\ast|$. 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. In this section we will say that a reduction rule is correct, if in addition to the input and output instances being equivalent, the graph in the output instance is chordal. We first give a reduction rule which takes care of trivial instances and vertices which are in common neighborhood of some terminal pair.

\textbf{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 $i \in \ell$. After application of the reduction rule, we see that no two terminals are adjacent, so they do not appear together in any of the bags of the clique-forest of $G$. Now we present a rule which deletes simplicial vertices from the graph which are not terminals.

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

\textbf{Lemma 12.} Reduction Rule 2 is correct.
**Proof.** Let \( G' = G - v \). As \( G' \) is an induced subgraph of \( G \), it is chordal, and any solution to \( (G, T, k) \) remains a solution to \( (G', T, k) \). For the converse, let \( S \subseteq V(G') \) be a solution to \( (G', T, k) \). Suppose, for the sake of contradiction that \( S \) is not a solution to \( G \). Then there exist \((s, t_1, t_2) \in T\) such that \( G \) has a path between \( s \) and \( t_1 \) in \( G - S \). Let us look at a shortest (hence chordless) path \( P \) between \( s \) and \( t_1 \) in \( G - S \). As there is no such path between \( s \) and \( t_1 \) in \( G' - S \), \( P \) must pass through \( v \). Let \( u_1 \) and \( u_2 \) be the neighbours of \( v \) on \( P \). Then \( u_1 u_2 \in E(G) \) as \( v \) is simplicial, and so \( u_1 u_2 \) is a chord of \( P \), a contradiction. This shows that \( S \) is a solution to \( (G, T, k) \) and finishes the proof of the lemma. ▶

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

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

**Proof.** Let us suppose, for the sake of contradiction, that there exists a leaf \( x \in V(F) \), such that \( \beta(x) \cap T^* = \emptyset \). Let \( N_F(x) = \{x_1\} \). Since all the bags are distinct and also maximal cliques, we must have a vertex \( v \in V(G) \setminus T^* \) such that \( v \in \beta(x) \setminus \beta(x_1) \). But then \( N[v] = \beta(x) \) which is a clique and hence \( v \) is simplicial in \( G \). It is a contradiction to that fact that Reduction Rule 2 does not apply and proves the statement of the lemma. ▶

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 \( t \in T^* \) and introducing a new terminal vertex \( t' \) to \( G' \) which forms a clique with the copies of \( t \). More formally, \( V(G') = (V(G) \setminus T^*) \cup \bigcup_{t \in T^*} \{t'\} \cup T'^* \), where \( C_t = \{v_{t_1}, \ldots , v_{t_k}\}, T'^* = \bigcup_{t' \in T^*} \{t'\}, V(G) \cap (T'^* \cup \bigcup_{t \in T^*} C_t) = \emptyset \), \( N_{G'}(v_{t'}) = N_{G}(t) \cup C_t \cup \{t'\} \), and \( N_{G'}(t') = C_t \) for all \( t \in T'^* \) and \( j \in [k + 1] \). \((G', T', k)\) is the new instance, where \( T' = \cup_{(s_i, t_1) \in T} \{s_i', t_1'\} \).

**Lemma 14.** Reduction Rule 3 is correct.

**Proof.** We first show that \( G' \) is chordal. Let us assume it is not, then it contains a chordless cycle \( C := v_1 v_2 \ldots v_q v_1 \) where \( q \geq 4 \). Since \( N_{G'}(t') \) is a clique for all \( t \in T'^* \), \( C \) does not contain \( t' \) for all \( t' \in T'^* \). Now we look at some \( C_t \). Since for all \( v_{t_1}, v_{t_2} \in C_t \), \( N_{G'}[v_{t_1}] = N_{G'}[v_{t_2}] = N_{G}(t) \cup C_t \cup \{t'\} \), if there are two vertices in \( C_t \), then that would give rise to a chord in \( C \). So we have that \( C \) contains at most one vertex from every \( C_t \). For a fixed \( t \), let this vertex be \( v_{t_1} \). Now, since \( N_{G'}(v_{t_1}) = N_{G}(t) \cup C_t \cup \{t'\} \), and \( t' \notin C \) and \( C \cap C_t = \{v_{t_1}\} \), we have that the neighbours of \( v_{t_1} \) on \( C \) are also neighbours of \( t \) in \( G \), and the non-neighbours of \( v_{t_1} \) on \( C \) in \( G' \) are either new vertices or they are also non-neighbours of \( t \) in \( G \). This enables us to replace each occurrence of a unique vertex from some \( C_t \) with \( t \) and get a chordless cycle in \( G \), which is a contradiction.

Let \( S \) be a solution for \((G, T, k)\). We show that \( S \) is also a solution for \((G', T', k)\). Suppose that is not the case, then there exist \((s_i', t'_1) \in T'^* \) such that there is a path from \( s_i' \) to \( t'_1 \) in \( G' - S \). Let \( P \) be one shortest path from \( s_i' \) to \( t'_1 \) in \( G' - S \). Since \( C_t \) forms a clique and all vertices from \( C_t \) have the same closed neighbourhood, we have that exactly one vertex from \( C_{t_1} \) and \( C_{t_2} \) respectively appears on \( P \). Let \( P := s_i v_{t_1} \ldots v_p t_i \). Since \( N_{G'}(v_{t_1}) = N_{G}(s_i) \cup C_t \cup \{s_i'\} \), we have that \( v_{t_1} \in C_t \). Similarly, we can argue that \( v_p \in N_{G}(t_i) \). This means that the path \( P^* := s_i v_{t_1} \ldots v_p t_i \) exists in \( G - S \) between \( s_i \) and \( t_i \), which is a contradiction.

For the converse, let \( S' \) be a minimal solution for \((G', T', k)\). We claim that \( S' \setminus \bigcup_{t \in T^*} C_t \) is a solution to \((G, T, k)\). Suppose not, and let \( P := s_i v_{t_1} \ldots v_p t_i \) be a path from \( s_i \) to \( t_i \) in
Quick Separation in Chordal and Split Graphs

\[ G - S' \] for some \((s_i, t_i) \in T\). Since \(|C_{s_i}| = |C_{t_i}| = k + 1\), we have that \(C_{s_i}, C_{t_i} \not\subseteq S'\). So we have some \(x, y \in [k + 1]\) such that \(v^x_s, v^y_t \not\in S'\). Since \(N_{G'}(v^x_s) = N_G(v^y_t) \cup C_{s_i} \cup \{s'_j\}\) for all \(j \in [k + 1]\), it is easy to see that \(P' := s'_i v^x_s v_1 \ldots v_p t'_j t_j\) is a path in \(G' - S'\), which is a contradiction.

Observe that Reduction Rule 3 preserves the number of terminals. Also, 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 \(|N(t)| = k + 1\).

**Proof.** Let \((G, T, k)\) be an instance obtained after applying Reduction Rule 3. Clearly, for each \(t \in T^*, |N(t)| = k + 1\) due to Reduction Rule 3. To see that a terminal \(t \in T^*\) belongs to exactly one bag of the clique-forest \((F, \beta)\) of \(G\), let us suppose that it is not the case and let \(x_1, x_2 \in V(F)\) be two adjacent nodes such that \(t \in \beta(x_1) \cap \beta(x_2)\). We know that all the bags in the clique-forest are distinct, and none is a subset of the other. This means that there exist \(u_1 \in \beta(x_1) \setminus \beta(x_2)\) and \(u_2 \in \beta(x_2) \setminus \beta(x_1)\) and \(u_1 u_2 \notin E(G)\), as no bag contains both \(u_1\) and \(u_2\) because of the connectivity property of the clique-forest \((F, \beta)\). But we also know that \(\beta(x_1)\) and \(\beta(x_2)\) form a clique in \(G\) and hence \(u_1, u_2 \in N_G(t)\). This is a contradiction to the fact that the neighbourhood of a terminal forms a clique after applying Reduction Rule 3, and proves the lemma.

**Lemma 16.** Let \((G, T, k)\) be an instance obtained after applying Reduction Rule 3 and let \((F, \beta)\) 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 \(|\bigcup_{x \in F_T} \beta(x)| \leq (k + 2)\tau\).

**Proof.** We know from Lemma 13 that after applying Reduction Rule 2, each leaf bag contains a terminal. This gives us \(F_1 \subseteq F_T\). We also know that because of Reduction Rule 3 and Lemma 15, each terminal occurs in exactly one bag of \((F, \beta)\). This gives us \(|F_T| \leq \tau\). As the number of vertices with degree at least three in a forest is at most the number of leaves, we also get \(|F_{\geq 3}| \leq \tau\) as desired. We know that each bag in \(F_T\) contains a terminal and is a clique in \(G\), and hence \(\bigcup_{x \in F_T} \beta(x) \subseteq N_G(T)\). From Lemma 15, we get that \(|N(t)| = k + 1\) for all \(t \in T^*\). This gives \(|\bigcup_{x \in F_T} \beta(x)| \leq (k + 2)\tau\).

So far, we have bounded the number of leaf bags, the number of bags with degree at least three in the clique-forest of \(G\), and also 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 with degree at least three. For that, we need to define some notions first.

We have established that after applying the aforementioned reduction rules, every terminal appears in exactly one bag of the clique-forest \((F, \beta)\) 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 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 \(\Pi(s_i, t_i) := x_1x_2 \ldots x_d\) 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 \(P\) 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 \(u, v \in \beta(x)\), \(u \leq_{(s_i, t_i)} v\) 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 \(v \leq_{(t_i, s_i)} v\) 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 the path \((s_i, t_i)\) the vertices of \(\beta(x_p)\) go, ranking the ones that go the farthest on either side as the highest.

Now we describe the marking procedure. For each bag \(x \in F_{G_3} \setminus F_T\) 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 \(x \in F_{G_3} \setminus F_T\) 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 \(M := (\bigcup_{x \in F_{G_3} \setminus F_T} M(x)) \cup (\bigcup_{x \in T} \beta(x))\).

Now we are ready to give the next reduction rule which will help us bound the size of bags in \(F_{G_3}\).

\begin{itemize}
  \item \textbf{Reduction Rule 4.} Let \((G, T, k)\) be an instance of MC where \(G\) is chordal graph and let \((F, \beta)\) be the clique-forest of \(G\). If there exists a node \(x \in F_{G_3} \setminus F_T\) such that \(\beta(x) \setminus M\) is nonempty, then delete an arbitrary vertex \(v \in \beta(x) \setminus M\) from \(G\).
  \item \textbf{Lemma 17.} Reduction Rule 4 is correct.
\end{itemize}

\textbf{Proof.} Let \(G' := G - v\). As \(G'\) is an induced subgraph of \(G\), \(G'\) is chordal and any solution for \((G, T, k)\) is a solution for \((G', T, k)\). For the other direction, let \(S\) be a solution for \((G', T, k)\). We claim that \(S\) is also a solution for \((G, T, k)\). Suppose that it is not the case, then there exists \((s_i, t_i) \in T\) such that there is a path between \(s_i\) and \(t_i\) in \(G - S\). Also, this path must pass through \(v\), as otherwise \(v\) would also exist in \(G' - S\). Let the shortest path between \(s_i\) and \(t_i\) in \(G - S\) be \(P := s_i v_1 v_2 \ldots v_{q-1} v_q t_i\), which has length \(q + 1\) and let \(v_p := v\) for some \(p \in [q]\). We claim that a path \(P' := s_i v_1 v_2 \ldots v_{p-1} v_p v_{p+1} \ldots v_q t_i\) exists in \(G' - S\) between \(s_i\) and \(t_i\), where \(v_p, v_p'' \in M\), which will be a contradiction to \(S\) being a solution for \((G', T, k)\).

To see that, let us first observe that since \(P\) is a shortest path between \(s_i\) and \(t_i\) in \(G - S\), \(V(P) \subseteq \bigcup_{x \in \Pi_G(s_i, t_i)} \beta(x)\). If \(\Pi_G(s_i, t_i)\) does not contain any node of degree at least 3, then \(v \notin \bigcup_{x \in \Pi_G(s_i, t_i)} \beta(x)\), and hence \(v \notin P\) which is a contradiction. Clearly, \(v\) is also not in the bags containing \(s_i\) or \(t_i\) because then by definition of \(M\), \(v\) would not be deleted. So \(v\) occurs in a bag of degree at least 3, that is an internal vertex of \(\Pi_G(s_i, t_i)\) and does not contain a terminal. Let \(x_v \in F_{G_3} \setminus F_T\) be a node such that \(v \in \beta(x_v)\). Clearly, \(v \notin M_{s_i}\), as otherwise \(v\) would not be deleted. We know that the marking procedure marked \(k + 1\) vertices in \(\beta(x_v)\) as \(M_x(s_i, t_i)\). Let \(v_p''\) be an arbitrary vertex in \(M_x(s_i, t_i) \setminus S\). Similarly, let \(v_p'\) be an arbitrary vertex in \(M_x(t_i, s_i) \setminus S\). Now, we want to show that the path \(P' := s_i v_1 v_2 \ldots v_{p-1} v_p v_{p+1} \ldots v_q t_i\) exists in \(G' - S\). We first observe that \(V(P') \subseteq V(G')\). Now, all we need to show is that the edges \(v_{p-1} v_p', v_p v_p'', v_{p+1} \in E(G)\). As \(G'\) is an induced subgraph of \(G\), this would mean that these edges also exist in \(G'\), proving the claim.

Clearly, \(v_p', v_p'' \in E(G)\) as \(v_p, v_p'' \in \beta(x_v)\) which induces a clique in \(G\). Let \(\Pi_G(s_i, t_i) := x_1 x_2 \ldots x_d\), where \(x_r = x_{s_i}\) for some \(r \in [s]\). Since \(v \notin x_{s_i}\), we have that there exists \(\alpha \in \{r, r + 1, \ldots, s\}\) such that \(v, v_{\alpha+1} \in \beta(x_{\alpha})\). Now, since \(v_p' \in M_x(s_i, t_i)\) and \(v_p'' \in M_x(s_i, t_i)\), we have that \(v_p' \geq (s_i, t_i) v\). Which means that for all \(x_{\alpha}'\), \(\alpha' \in \{r, r + 1, \ldots, s\}\), if \(v \in \beta(x_{\alpha}')\)
then \(v^p_i \in \beta(x_i)\). This would mean that \(v^p_i \in \beta(x_a)\) and \(v^p_i v_{p+1} \in E(G)\) as \(\beta(x_a)\) induces a clique in \(G\). Similarly we can show that \(v_p v'_p \in E(G)\), which finishes the proof of the lemma.

\[\blacktriangleleft\]

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

**Proof.** We know from Lemma 16 that \(|F_{\geq 3}| \leq \tau\). So all we need to show that for all \(x \in F_{\geq 3}\), \(|\beta(x)| = O(k\ell r)\). For a bag \(\beta(x_t)\) containing a terminal \(t\), we know from Lemma 15 that \(|\beta(x_t)| \leq k + 2\). We already know due to Lemma 16 that \(|\bigcup_{t \in T} \beta(x_t)| \leq (k + 2)\tau\). Now, for a bag \(\beta(x)\) such that \(x \in F_{\geq 3} \setminus F_T\), we want to give a bound for \(M_x\). We mark at most \(2k + 2\) vertices for each \((s_i,t_i)\) \(\in T\). That gives us \(|M_x| \leq \ell(2k + 2)\). Now, combining that with \(|F_{\geq 3}| \leq \tau\), we get that \(|\bigcup_{t \in T} M(x)\) \(\leq \ell \tau(2k + 2)\). Now, if for some \(x \in F_{\geq 3} \setminus F_T\), \(|\beta(x)| > (k + 2)\ell r + (k + 2)\ell r\), then \(\beta(x)\) \(\subseteq M\) is non-empty and Reduction Rule 4 applies. This gives us \(|\beta(x)| \leq (2k + 2)\ell r + (k + 2)\ell r = O(k\ell r)\) for all \(x \in F_{\geq 3}\) and proves the lemma.

\[\blacktriangleleft\]

Now we have bounded the number of vertices in the graph which are part of degree 1 and degree at least 3 bags of the clique-forest of \(G\), and also the number of vertices which appear in any bag that contains a terminal. 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 \(Q := x_1 x_2 ... x_q\) 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 \(i \in \{2, 3, ..., q - 1\}\). 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 \(B_i \cap (\beta(x_i) \setminus \beta(x_{i+1})) \neq \emptyset\). In such a case, we add \(x_i\) and \(x_{i+1}\) to \(D(Q)\). Similarly, if \(x_j\) and \(x_{j-1}\), \(j \in \{2, 3, ..., q\}\) 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 \(D(Q)\).

\[\blacktriangleleft\]

**Observation 1.** Let \(Q := x_1 x_2 ... x_q\) 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 \(i \in \{2, 3, ..., q - 1\}\). Let \(B_1 := \beta(x_1)\) and \(B_q := \beta(x_q)\). Let \(Q' := y_1 y_2 ... y_r\) be a subpath of \(Q\) such that \(y_i, y_r \in D(Q)\) but \(y_r \notin D(Q)\) for all \(i \in \{2, 3, ..., r - 1\}\). Then \(B_1 \cap \beta(y_1) = B_1 \cap \beta(y_2) = ... = B_1 \cap \beta(y_r)\) and \(B_q \cap \beta(y_1) = B_q \cap \beta(y_2) = ... = B_q \cap \beta(y_r)\).

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 collections 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 where \(G\) is a chordal graph and let \((F,\beta)\) be the clique-forest of \(G\). Then, there exist \(S_i \subseteq S\) for all \((s_i,t_i) \in T\), such that

1. \(S_i\) is an adhesion on \(E(s_i,t_i)\),
2. \(S = \bigcup_{(s_i,t_i) \in T} S_i\), and
3. if \(Q := x_1 x_2 ... x_q\) is a path in \(F\) such that \(x_z \notin F_{\geq 3} \cup F_T\) for all \(z \in \{2, 3, ..., q - 1\}\), then at most one adhesion from the path \(Q\) is picked as \(S_i\) for some pair \((s_i,t_i) \in T\).
Proof. The first part of the lemma is obvious from Lemma 11, as it is necessary to pick one adhesion on \( \Pi(s_i, t_i) \) to disconnect the pair \((s_i, t_i)\). For the second part, we observe that it suffices to delete one adhesion from \( \Pi(s_i, t_i) \) to disconnect \( s_i \) from \( t_i \) due to Lemma 9.

For the third part, suppose that an adhesion \( S_Q \) is picked from \( Q \) in \( S \). Then observe that, by Lemma 9, \( S_Q \) separates every path in \( G \) that contains a vertex in \( \beta(x_1) \) and a vertex in \( \beta(x_q) \). Hence, for the purpose of separating such paths in \( G \), it is sufficient to include at most one adhesion in \( Q \). Finally observe that, as every internal vertex \( x_z \) of \( Q \) has degree 2 in \( F_1 \) and \( \beta(x_z) \) doesn’t contain any terminals and every bag of the forest decomposition (\( F, \beta \)) induces a clique in \( G \), any induced path in \( G \) between two terminals \( s_i \) and \( t_i \) that are separated by an adhesion from \( Q \) must contain a vertex in \( \beta(x_1) \) and a vertex in \( \beta(x_q) \), i.e. \( Q \) is a sub-path of \( \Pi(s_i, t_i) \). Therefore, any adhesion from \( Q \) (in \( S \)) is sufficient to separate \( s_i \) and \( t_i \) in \( G \). Hence, if \( S \) contains more than one adhesion from \( Q \), then it is safe to drop all but one of them from \( S \).

Observe that the adhesions picked by Lemma 19 for a minimal solution for different pairs might not be distinct or disjoint. Also that the collection of adhesions might not be unique for a given minimal solution. Now we are ready to give the reduction rule which decreases the number of degree 2 nodes. We give the reduction rule in terms of the clique-forest of \( G \), but as clique-forests are in one-to-one correspondence with chordal graphs, it can also be viewed as a well defined operation on the graph \( G \).

**Reduction Rule 5.** Let \((G, T, k)\) be an instance of MC where \( G \) is a chordal graph and \((F, \beta)\) be the clique-forest of \( G \). Let \( Q := x_1x_2 \ldots x_q \) be a path in \( F \) such that \( x_1, x_q \in F_{y \geq 3} \cup F_T \) and \( x_\gamma \notin F_{y \geq 3} \cup F_T \) for all \( \gamma \in \{2, 3, \ldots, q - 1\} \). Let \( Q' := y_1, \ldots, y_r \) 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 \( \alpha \in \{2, 3, \ldots, r - 1\} \). 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 \( U = \{y_1, \ldots, y_z\} \) be a set of new vertices such that \( U \cap V(G) = \emptyset \). To get a new clique-forest \( (F', \beta') \), delete \( y_\alpha \) for each \( \alpha \in \{2, 3, \ldots, q - 1\} \), 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 \( (F', \beta') \).

Output \((G', T, k)\).

We first show that the reduction rule is well-defined.

**Lemma 20.** Reduction Rule 5 is well defined. That is, \( z \geq 0 \), \( F' \) is a forest, and \( (F', \beta') \) satisfies property 3 of Definition 8.

**Proof.** First, we show that \( z \geq 0 \). For that, we observe that \( \beta(y_1) \cap \beta(y_r) \subseteq N(s) \cap N(t) \) and hence \( c \geq |\beta(y_1) \cap \beta(y_r)| \). It is easy to see that \( F' \) is a forest, as it can be obtained by contracting all but one edges of the path \( Q' \) in \( F \). To show that \( (F', \beta') \) satisfies property 3 of Definition 8, we first observe that \( F'[\beta'y_1)] \) is connected for all \( v \in V(G') \setminus V(G) \), as they occur only in adjacent bags \( \beta'(y_1) \) and \( \beta'(y_2) \). For \( v \in V(G') \) such that \( v \in \beta(y_1) \cap \beta(y_r) \), \( F'[\beta'y_1)] \) remains connected because \( F[\beta'y_1)] \) was connected, and \( y_1y_2 \in E(F') \). For any other \( v \in V(G') \), if \( F[\beta'y_1)] \) did not intersect \( Q' \), then \( F'[\beta'y_1)] \) remains connected, as it is identical to \( F[\beta'y_1)] \). The only remaining case is \( F[\beta'y_1)] \cap Q \neq \emptyset \) and \( v \notin \beta(y_1) \cup \beta(y_r) \). In this case, \( F[\beta'y_1)] \) contains either \( y_1 \) or \( y_r \), but not both, and a proper subpath of \( Q' \).

Without loss of generality, let \( y_1 \in F[\beta'y_1)] \) and let the subpath of \( Q' \) in \( F[\beta'y_1)] \) be \( Q'' \). Then we have that \( F'[\beta'y_1)] = (F[\beta'y_1)] \setminus Q'') \cup \{y_1\} \), which can be obtained from
contracting the edges of \( Q'' \) in \( F[β^{−1}(v)] \) and is connected. Hence, \( (F', β') \) satisfies property 3 of definition 8.

This shows that \( G' \) is a chordal graph and \((G', T, k)\) is a valid instance of MC. Now we prove a lemma that relates the adhesions of the input and output instances of the reduction rule.

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

**Proof.** From the construction of \((F', β')\), we can look at \( F' \) as the forest obtained from \( F \) by contracting all the edges of \( Q' \) except one. This means that for any \((s_i, t_i) \in T \) and \( x \in V(F') \setminus \{y_2, \ldots, y_{r−1}\} \), \( x \in Π_G(s_i, t_i) \) if and only if \( x \in Π_{G'}(s_i, t_i) \). Moreover, if \( x \in V(F') \setminus \{y_1, y_r\} \) then \( β(x) = β'(x) \), else if \( x \in \{y_1, y_r\} \) and \( β'(x) = β(x) \cup U \). Hence, for any pair \( z_1, z_2 \in V(F') \) such that at least one of them is not in \( \{y_1, y_r\} \), we have \( β(z_1) \cap β(z_2) = β'(z_1) \cap β'(z_2) \).

Now, to prove the first part of the lemma, let \( A = β(z_1) \cap β(z_2) \) be an adhesion on \( Π_{G'}(s_i, t_i) \) (for the graph \( G \)) such that \( A \notin A \). Since we have that \( β(x) = β'(x) \) for all \( x \in V(F') \setminus Q' \), if \( \{z_1, z_2\} \cap Q' = \emptyset \), \( A = β'(z_1) \cap β'(z_2) \) remains an adhesion on \( Π_{G'}(s_i, t_i) \) (for the graph \( G' \)). Now we look at the case when \( z_1 \in V(F) \setminus Q' \) and \( z_2 = y_i \). Since \( β(y_1) \subseteq β'(y_1) \) and \( U \cap β(z_1) = \emptyset \) (\( U \) is the new set of vertices added, i.e., \( U = V(G') \setminus V(G) \)), we have that \( β'(z_1) \cap β'(z_2) = β(z_1) \cap β(z_2) \). The case when \( z_2 \in V(F) \setminus Q' \) and \( z_1 = y_r \) is similar. A similar argument holds for the second part of the lemma, where we consider adhesions on \( Π_{G'}(s_i, t_i) \) (for the graph \( G' \)) excluding the adhesion \( A^* = β'(y_1) \cap β'(y_r) \). We obtain that each of them is an adhesion on \( Π_G(s_i, t_i) \) (for the graph \( G \)).

**Lemma 22.** Reduction Rule 5 is correct.

**Proof.** We have already shown that \( G' \) is chordal. Now we will show that there exists a solution for \((G', T, k)\) if and only if there exists a solution for \((G, T, k)\). Let us recall that we applied the reduction rule on the subpath \( Q' = (y_1, y_2, \ldots, y_r) \) of the path \( Q = (x_1, \ldots, x_d) \) in the forest \( F \). In the forward direction, let \( S \) be a minimal solution for \((G, T, k)\). Let \( A := \{A_a\} \) be the set of adhesions of the path \( Q' \) where \( A_a := β(y_a) \cap β(y_{a+1}) \) for \( a \in [r−1] \) and let \( B_1 := β(x_1) \) and \( B_q := β(x_q) \). We want to show that there exists a solution for \((G', T, k)\). For that, we look at a collection of adhesions \( S_{i,j} \) as described in Lemma 19 such that \( S = \bigcup_{(t_i, t_j) \in E(S)} S_{i,j} \). Now we look at two possible cases.

**Case 1.** The first case is when there does not exist any \( A_a \in A \) such that \( A_a = S_i \) for some \((t_i, t_j) \in T \). In this case we claim that \( S \) is also a solution for \((G', T, k)\). First we observe that none of the vertices of \( β(y_1) \) and \( β(y_r) \) are deleted by the reduction rule. So, \( S_i \subseteq V(G') \) for all \((s_i, t_i) \in T \) and we have that \( S \subseteq V(G') \). Also, for any \((s_i, t_i) \in T \), we know from Lemma 21 that \( S_i \) remains an adhesion on \( Π_{G'}(s_i, t_i) \), so deleting \( S \) disconnects \( s_i \) from \( t_i \), because \( S_i \subseteq S \). Hence \( S \) is a solution to \((G', T, k)\).

**Case 2.** The second case is when there exists some \( A_a \in A \) such that \( A_a = S_i \) for some \((t_i, t_j) \in T \). Because of Lemma 19, we know that there exists unique such \( A_a \). Let \( A^* := β'(y_1) \cap β'(y_r) \). We claim that \( S' := (S \setminus A_a) \cup A^* \) is a solution to \((G', T, k)\). We see that \( |S'| \leq |S| \), because \( |A_a| = c = |A^*| \). Suppose that \( A_a \) separates some pair \((s_i, t_i) \in T \) in \( G \), i.e. \( A_a \) is an adhesion on \( Π_G(s_i, t_i) \). This implies that \( Q' \subseteq Π_G(s_i, t_i) \) and \( A_a \) is an
adhesion on \(Q'\). In \(G'\), we have that \(y_1, y_r \in \Pi_G((s_i, t_i))\), and hence \(A^*\) is an adhesion which will disconnect \(s_i\) from \(t_i\) in \(G'\). Therefore deleting \(S'\) in \(G'\) disconnects \(s_i\) from \(t_i\).

Next consider a pair \((s_i, t_i)\) in \(T\) such that \(A_\omega \neq S_i\). By Lemma 19, we know that \(S_i\) is not an adhesion on the path \(Q'\) (for the graph \(G\)). Then, by Lemma 21, we know that \(S_i\) is still an adhesion on \(\Pi_G((s_i, t_i))\) (for the graph \(G'\)). So to show that deleting \(S'\) in \(G'\) disconnects \(s_i\) from \(t_i\), it is sufficient to show that \(S_i \subseteq S'.\) Let \(v \in S_i\) be an arbitrary vertex.

We will show that \(v \in S'\), which will finish the proof of forward direction.

We first look at the case when \(v \in B_1 \cup B_q\). If \(v \notin A_\omega\), then we have that \(v \in S'\) and we are done. If \(v \in A_\omega\), then \(v \in \beta(y_1)\) from the connectivity property of the clique-forest. We also know from Observation 1 that \(B_1 \cap \beta(y_1) = B_1 \cap \beta(y_r) = \ldots = B_1 \cap \beta(y_r)\).

That is, \(v \in \beta(y_1) \cap \beta(y_r)\). By definition of \(S'\), \(\beta(y_1) \cap \beta(y_r) \subseteq A^* \subseteq S',\) and hence \(v \in S'\).

The case when \(v \in B_q\) is similar. Next we look at the case when \(v \notin B_1 \cup B_q\). Then we claim that \(v \notin A_\omega\) and hence \(v \in S'\). Suppose this is not the case and \(v \in A_\omega\). We know that \(v \in S_{i,j}\) and \(S_{i,j}\) is not an adhesion on \(Q\). Hence \(v\) occurs in the bag \(\beta(x)\) for some \(x \in V(Q)\). However, as \(v\) is in an adhesion \(A_\omega\) on \(Q'\), it must be that \(v \in B_1 \cup B_q\), to satisfy the connectivity property of the clique-forest (which is a tree-decomposition of \(G\)). This is a contradiction to our assumption that \(v \notin B_1 \cup B_q\).

In the reverse direction, let \(S'\) be a minimal solution to \((G', T, k)\). We consider a collection of adhesions \(S'_i\) such that \(S' = \bigcup_{(s_i, t_i) \in T} S'_i\) as described in Lemma 19. Recall that \(A^* = \beta(y_1) \cap \beta(y_r)\). We have two cases. The first case is when \(S'_i = A^*\) for some \((t_i, t_i) \in T\).

Due to Lemma 9 and Lemma 11, it is sufficient to delete an adhesion on \(Q'\) to disconnect \(s_i\) from \(t_i\). Let \(C\) be a min-cut of size \(c\) in the auxiliary graph \(G^*\), as described in the Reduction Rule 5. Observe that, by construction of \(G^*\) and Lemma 11, \(C\) is a minimum-size adhesion on the path \(Q'\) (for the graph \(G\)). We claim that \(S := (S' \setminus A^*) \cup C\) is a solution to \((G, T, k)\). To see that, let us first observe that \(|S| = |S'|\) since \(A^* \subseteq S'\), \(A^* \cap \beta(y_1) \cup \beta(y_r) = C \cap (\beta(y_1) \cup \beta(y_r))\) and \(|C| = |A^*| = |\beta(y_1) \cap \beta(y_r)| = |\beta(y_1) \cap \beta(y_r)| + |U|\). Now, if for a pair of terminals \((s_i, t_i) \in T\), if \(S'_i = A^*\), then observe that, in \(G, Q' \subseteq \Pi_G(s_i, t_i)\), and hence deleting \(C\) in \(G\) disconnects \(s_i\) and \(t_i\). If \(S'_i \neq A\), then \(S'_i \cap U = \emptyset, S'_i \subseteq S\) and \(S'_i\) is still an adhesion on \(\Pi_G(s_i, t_i)\) by Lemma 21. Hence deleting \(S'_i\) would disconnect \(s_i\) from \(t_i\) in \(G\). Therefore, in this case, \(S\) is a solution to \((G, T, k)\).

The other case is when \(A^* \neq S'_i\) for any \((s_i, t_i) \in T\). We claim that \(S'_i\) itself is a solution to \((G, T, k)\). To see that, let us first observe that \(U \cap A = \emptyset\) for all adhesions \(A\) in \(G'\) such that \(A \neq \beta(y_1) \cap \beta(y_r)\). So we have that \(S' \subseteq V(G)\). Also, \(S'_i\) is also an adhesion on \(\Pi_G(s_i, t_i)\) for all \(\{s_i, t_i\} \in \binom{T}{2}\) due to Lemma 21, and deleting it would disconnect \(s_i\) from \(t_i\) in \(G\). This finishes the proof of the lemma.

**Lemma 23.** Let \((G, T, k)\) be an instance obtained after exhaustively applying Reduction Rule 5 and \((F, \beta)\) be the clique-forest of \(G\). Then \(|V(F)| = O(k + \tau^2)|. \]

**Proof.** From Lemma 16 and Lemma 18, we know that for all \(x \in V_T \cup V_{\geq 3}\), \(|\beta(x)| = O(k + \tau)|. \]

Let \(Q := x_1 x_2 \ldots x_q\) 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 \(i \in \{2, 3, \ldots, q - 1\}\). Since the for every vertex in \(\beta(x_1) \cup \beta(x_2)\), at most two nodes in \(Q\) are marked, and we have that \(|\beta(x_1) \cup \beta(x_2)| = O(k + \tau)|\) due to Lemma 18, we mark at most \(O(k + \tau)|\) nodes on \(Q\). If there are any unmarked nodes on \(Q\), then Reduction Rule 5 would apply, so we have that after the exhaustive application of the rule, \(|V(Q)| = O(k + \tau)|. \]

But since \(|F_T \cup F_{\geq 3}| = O(|\tau|)|, there can be \(O(|\tau|)|\) many such paths, and hence the total number of nodes on such paths is \(O(k + \tau^2)|. \)

This, combined with \(|F_T \cup F_{\geq 3}| = O(|\tau|)|\) proves the lemma.
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 $F_2 = V(F) \setminus (F_{G3} \cup F_T)$ where $(F, \beta)$ 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 want to mark at most $2k + 2$ vertices in $\beta(x)$. Let $\Pi(s_i, t_i) := x_1x_2 \ldots x_d$ be a nonempty path where $x_i = x_1$ and $x_j = x_d$. 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 $v \in \beta(x_q)$.

Now we describe the marking procedure. For each bag $x \in F_2$, for each pair of terminals $(s_i, t_i) \in T$ 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 $Z_x(s_i, t_i)$. We also mark $k + 1$ vertices which are largest in the ordering $\leq_{(t_i, s_i)}$ and call that set $Z_x(t_i, s_i)$. Let the set of all marked vertices inside a bag $\beta(x)$, such that $x \in F_2$ be $Z(x) := \bigcup_{(s_i, t_i) \in T}(Z_x(s_i, t_i) \cup Z_x(t_i, s_i))$ and let $Z := \bigcup_{x \in F_2} Z(x)$.

**Reduction Rule 6.** Let $(G, T, k)$ be an instance of MC and let $(F, \beta)$ 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 $|V(G)| = O(k^3 \ell^4 \tau^4)$.

**Proof.** Let $(F, \beta)$ be the clique-forest of $G$. We already know due to Lemmas 16, 18, and 23 that $|V(F)| = O(k^4 \ell^2 \tau^2)$ and $|\beta(x)| = O(k \ell \tau)$ for all $x \in F_T \cup F_{G3}$. We also know that $|F_2| = O(k \ell \tau^2)$. So if we can show $|\beta(x)| = O(k^2 \ell^2 \tau^2)$ for all $x \in F_2$, this would prove the lemma. For that, we want to show that $|Z| = O(k^2 \ell^2 \tau^2)$. This would mean that $|\beta(x)| = O(k^2 \ell^2 \tau^2)$ 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 $|F_2| = O(k \ell \tau^2)$, we get that $|\bigcup_{x \in F_2} Z(x)| = O(k^2 \ell^2 \tau^2)$. We already know that $|\bigcup_{x \in F_2 \cup F_{G3}} \beta(x)| = O(k \ell \tau^2)$, so this gives us $|Z| = O(k^2 \ell^2 \tau^2)$ as desired. □

**Proof of Theorem 1.** Given an instance $(G, T, k)$, we first find the clique-forest $(F, \beta)$ of $G$ and then apply reduction rules 1-6 in order exhaustively. It is easy to see that the reduction rules can be applied in polynomial time and we have argued that they are correct. From Lemma 24, we know that if none of the reduction rules apply, then the number of vertices in $G$ is $O(k^3 \ell^7)$, since $\tau = O(\ell)$.

### 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 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.

Proof of Theorem 2. Due to Reduction Rule 7, we can assume that for the MWC instance \((G, T, k)\), \(|T| \leq 2k\). Now, given an instance \((G, T, k)\) of MWC, we can obtain an equivalent instance \((G', T', k')\) of MC, by putting \(G' := G, k' := k\), and \(T' := \{ (t_i, t_j) \mid t_i, t_j \in T, i < j \}\).

Clearly, since \(|T| \leq 2k\) and \(\bigcup_{(s_i, t_i) \in T} \{ s_i, t_i \} = T\), for the MC instance \((G', T', k)\), we have \(\ell = O(k^2)\) and \(\tau \leq 2k\). Now, we apply reduction rules 1-6 on the MC instance \((G', T', k')\).

Let the output instance, on which none of the reduction rules apply, be \((G'', T'', k'')\). Due to Lemma 24, we have that \(|V(G'')| = O(k^3\ell^3\tau^4)\). Let \(T' := \bigcup_{(s_i, t_i) \in T} \{ s_i, t_i \}\). Observe that none of the reduction rules alter the set of terminals, except Reduction Rule 3, which also preserves one-to-one correspondence between the terminals (and between pairs of terminals). Because of this, there is a one-to-one correspondence between the pairs of terminals of the starting instance \((G', T', k')\) and the kernelized instance \((G'', T'', k'')\). This means that for all \{(t_i, t_j) \in T'\}, we have that \((t_i, t_j) \in T''\). This implies that \((G'', T', k'')\) is an equivalent instance to \((G', T', k')\) and hence to \((G, T, k)\). Since \(\tau = O(k)\) and \(\ell = O(k^2)\), this gives a kernel for MWC on chordal graphs with \(O(k^{13})\) vertices.

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. 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.

Proof. Consider a chordal graph \(G'\) obtained from \(G\) by introducing a new vertex \(x\) that is adjacent to all the vertices of \(X\). Then by applying Proposition 26 to the minimal \((v, x)\)-separators in \(G'\), we obtain the corollary.

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), \(W \subseteq V(G)\), 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 \( S_k \) of all important \( (X, Y) \)-separators of \( G \) of size at most \( k \) can be computed in time \( O(|S_k| \cdot k^2 \cdot (n + m)) \).

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 \((F, \beta)\), 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 \( O(\ell \cdot (n + m)) \) time and Reduction Rule 3 can be applied in \( O(k \cdot (n + m)) \) time. Also both these rules are applied only once in the course of the algorithms. Thus, the application of these rules incur an additive running time of \( O((k + \ell) \cdot (n + m)) \) to 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 \((F, \beta)\). We denote the unique bag of \( F \) containing \( s_i \) (resp. \( t_i \)), as \( x_{s_i} \) (resp. \( x_{t_i} \)). Let \( x_{s_i}^{\text{loc}} \) 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 \((F, \beta)\) 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 chordal graph, and let \((F, \beta)\) be a rooted clique forest of \( G \) as defined above. Let \( y \) denote the root bag of \( F \). Let \((s_p, t_p)\) \( \in T \) be such that \( x_{s_p}^{\text{loc}} \) is deepest in the rooted forest \( F \), that is, \( x_{s_p}^{\text{loc}} \) is such that \( \text{dist}_F(y, x_{s_p}^{\text{loc}}) = \max \{ \text{dist}_F(y, x_{s_i}^{\text{loc}}) : i \in [\ell] \} \), where \( \text{dist}_F(y, x_{s_i}^{\text{loc}}) \) denote the distance between \( y \) and \( x_{s_i}^{\text{loc}} \) in \( F \). Then there is a solution to \((G, T, k)\) that contains either an important \( (\{s_p\}, \beta(x_{s_p}^{\text{loc}})) \)-separator or an important \( (\{t_p\}, \beta(x_{t_p}^{\text{loc}})) \)-separator of size at most \( k \).

**Proof.** Let \( S \) be a solution to the instance \((G, T, k)\). From Claim 31, \( S \) contains an adhesion on the unique \( x_{s_p} \) to \( x_{t_p}^{\text{loc}} \) path or on the unique \( x_{t_p} \) to \( x_{s_p}^{\text{loc}} \). Without loss of generality, let this adhesion, say \( A \), be on the \( x_{s_p} \) to \( x_{t_p}^{\text{loc}} \) path. Then \( A \) is an \( (\{s_p\}, \beta(x_{s_p}^{\text{loc}})) \)-separator. If \( A \) is an important \( (\{s_p\}, \beta(x_{s_p}^{\text{loc}})) \)-separator, we are done. Otherwise, let \( A^* \) be the unique important separator of size \(|A|\) (from Lemma 29). Here we rely on the fact that \( \beta(x_{s_p}^{\text{loc}}) \) induces a clique in \( G \). We claim that \( S^* = (S \setminus A) \cup A^* \) is also a solution to \((G, T, k)\).

Observe that \(|S^*| = |S|\). For the sake of contradiction, suppose that there exists \((s_i, t_i)\) \( \in T \), \( i \neq p \), such that there exists an \( s_i \) to \( t_i \) path in \( G - S^* \). From Claim 31, since \( S \) was an \( (s_i, t_i)-\)separator, \( S \) contains some adhesion, say \( A' \), on the unique \( (x_{s_i}, x_{t_i}^{\text{loc}}) \)-path or on the unique \( (x_{t_i}, x_{s_i}^{\text{loc}}) \)-path. Without loss of generality, let this path be the \( (x_{s_i}, x_{t_i}^{\text{loc}}) \)-path. If \( A' \cap A = \emptyset \), then we are done. We now show that \( A' \cap A \subseteq A^* \). Note that, if we prove this we are done.
From Claim 31, $A^*$ contains an adhesion on the $(x_{s_p}, x_{t_p}^{lca})$-path. Also, from Lemma 9, we conclude that $A^*$ is indeed an adhesion on the $(x_{s_p}, x_{t_p}^{lca})$-path. Let $A$ be the adhesion between bag $x_i$ and $x_{j+1}$, that is, let $A = \beta(x_i) \cap \beta(x_{j+1})$. Let $A' = \beta(x_i) \cap \beta(x_{t+1})$ and let $A^* = \beta(x_i) \cap \beta(x_{t+1})$. Let us define a $< \!$ relation on the bags of $F$. We say $x_j < x_{t+1}$ if $x_j$ is in the subtree rooted at $x_{t+1}$ in $F$. We say $x_j \leq x_{t+1}$ if either $x_j = x_{t+1}$, or $x_j < x_{t+1}$. Without loss of generality, let $x_j < x_{j+1}$, $x_{t+1} < x_{t+1}$ and $x_{t+1} < x_{t+2}$. Now we make the following claim.

Claim: $x_j < x_{t+1}$.

Proof. Suppose not. If $x_{t+1} = x_j$, then $A^* = A$. Otherwise, if $x_{t+1} < x_j$, then set of vertices reachable from $s_i$ in $G - A^*$ is a strict subset of the set of vertices reachable from $s_i$ in $G - A$, thereby violating that $A^*$ is an important separator.

We divide the remaining proof of Lemma 32 in the following three cases.

Case 1: $x_{t+1} < x_j$: From the claim above, we have that $x_j \leq x_{t+1} \leq x_{t+2}$. Since $A \cap A' \subset \beta(x_j)$ and $A \cap A' \subset \beta(x_{t+1})$, and $F$ is a clique forest decomposition (which is a tree decomposition), therefore, $A \cap A' \in x_j$ and $A \cap A' \in C(x_{t+1})$, we conclude that $A \cap A' \subset A^*$ and hence $A \cap A' \subset S^*$.

Case 2: $x_{t+1} < x_j$: In this case, since $x_{j+1}^{lca} \neq x_{t+1}^{lca}$ (because of the choice of the pair $(s_p, t_p)$), the unique $x_{j+1}^{lca}$ path passes through $x_{t+1}$ and $x_{t+2}$. Therefore, $A^*$ is also an adhesion on this path. Since $A^* \subset S^*$, from Claim 31, we conclude that $S$ is an $(s_i, t_i)$-separator.

Case 3: $x_j$ and $x_{t+1}$ are incomparable: In this case, since by the claim above, we have that $x_j < x_{t+1}$, $x_{t+1} < x_{t+2}$ and $x_{t+2}$ are also incomparable. Now, either $A \cap A' = \emptyset$ or $A \cap A' \subset A^*$ by the connectivity property of forest-decompositions, and hence we are done.

This finishes the proof of the lemma.

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 chordal graph. Let $(F, \beta)$ be a rooted forest decomposition of $G$. From Lemma 32, there exists a pair $(s_p, t_p) \in T$ such that there exists a solution $S$ which contains either an important $\{(s_p), \beta(x_{p}^{lca})\}$-separator of size at most $k$, or an important $\{(t_p), \beta(x_{t}^{lca})\}$-separator of size at most $k$. Let $I_s$ (resp. $I_t$) be the collection of all important $\{(s_p), \beta(x_{p}^{lca})\}$-separator (resp. $\{(t_p), \beta(x_{t}^{lca})\}$-separator) of size at most $k$. The algorithm branches the sets in $I_s \cup I_t$. That is, it reduces the instance $(G, T, k)$ to the set of instances $(G - I, T - I, k - |I|)$, where $I \in I_s \cup I_t$ and $T - I$ denotes the set of pairs of terminals with no vertex in $I$.

Proof of Theorem 3. The correctness of Algorithm 1 follows from Lemma 32. To prove the theorem, we show that it runs in $O(2^k \cdot (k^3 + \ell) \cdot (n + m))$ 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 $\{(t_p), \beta(x_{t}^{lca})\}$-separator) of a fixed size, from the description of the algorithm, we get the following recurrence: $T(k) \leq 2 \sum_{i \in [k]} T(k - i)$. 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 $O((k + \ell)(n + m))$, plus time taken by the algorithm of Proposition 30, which is $O(k^3 \cdot (n + m))$ because $S_k$ in the proposition has size at most $k$ from Lemma 29. The desired running time thus follows.

Before we prove Theorem 4, we give the following reduction rule.
Quick Separation in Chordal and Split Graphs

1. 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 exhaustively in \(O((k + \ell)(n + m))\) time.

2. Proof of Theorem 4. For the proof of this theorem, observe that, after each branching step of Algorithm 1, there is no path between \(s_p\) to \(t_p\). After each branching step of the algorithm, one can then apply Reduction Rule 8. Thus, in the resulting instance the size of the terminal set decreases by at least 1. Thus, if \(T(k, \ell)\) denotes the number of leaves in the branching tree rooted at an instance with \(\ell\) terminal pairs and budget \(k\), we get the following recurrence:

\[
T(k, \ell) \leq \sum_{i \in [2k]} T(k, \ell - 1), T(k, 0) = 1.
\]

This solves to \(T(k, \ell) \leq (2k)^\ell\). Thus, the running time of \(2O((\log k) \cdot (n + m) + \ell(n + m))\) follows from Proposition 30 and the fact that the reduction rules can be applied in \(O((k + \ell)(n + m))\) time.

6. FPT algorithms for Multicut on Split Graphs

In this section, we design faster FPT algorithms for Multicut on split graphs. In particular, we prove Theorem 5, 6 and 7. Recall that a graph is a split graph if and only if its vertex set can be partitioned into two parts: \(C\) and \(I\), such that the set \(C\) induces a clique and \(I\) is an independent set. It is known that given a split graph \(G\), such a partition can be obtained in time \(O(n + m)\) [8]. Henceforth, we assume that the partition of the input split graph into a clique \(C\) and an independent set \(I\) 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. This exhaustive application of these reduction rules contribute \(O(\ell \cdot (n + m))\) to the running time of our algorithms.

Lemma 33. Let \((G = (C,I), T, k)\) be an instance of MC on split graphs which is reduced with respect to Reduction Rules 1 and 8. Then, for each \((s_i, t_i) \in T\), \(s_i, t_i \in I\).

Proof. Since Reduction Rule 1 is not applicable, we remain to prove that it is not the case that there exists \((s_i, t_i) \in T\), such that \(s_i \in C\) and \(t_i \in I\) (or vice-versa; this case is symmetric). Suppose this happens. Then consider a shortest \(s_i\) to \(t_i\) path, say \(P = s_i v_1 \ldots v_r t_i\), in \(G\).

Such a path exists because Reduction Rule 8 is not applicable. Also, since Reduction Rule 1 is not applicable, we have that \(r > 1\). Since \(t_i \in I\) and \(v_r \in N(t_i), v_r \in C\). Since \(s_i, v_r \in C\) and \(C\) is a clique, we have that \((s_i, v_r) \in E\). Since \(P\) is a shortest path from \(s_i\) to \(t_i\), we conclude that \(r = 1\), thereby contradicting the applicability of Reduction Rule 1.

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 \(C\).

Proof. Let \((s_i, t_i) \in T\). From Lemma 33, \(s_i, t_i \in I\). Let \(P\) be a shortest \(s_i\) to \(t_i\) path in \(G\). Since Reduction Rules 1 and 8 are not applicable, length of \(P\) is at least 4. Let \(P = s_i v_1 \ldots v_r t_i\). For the sake of contradiction, let \(r \geq 3\). From Lemma 33, since \(s_i, t_i \in I\), and \(v_1, v_r \in N(s_i), v_r \in N(t_i)\), we have that \(v_1, v_r \in C\). Since \(C\) is a clique, we have that \((v_1, v_r) \in E\), thereby contradicting that \(P\) is a shortest \(s_i\) to \(t_i\) path in \(G\).
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 \(i \in [t]\), 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 \(G\).

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

\[\textbf{Proof.} \text{For the forward direction, let } S \subseteq V(G) \setminus \bigcup_{i \in [t]} \{s_i, t_i\} \text{ be a multicut for the instance } (G, T, k). \text{ For a } \mathcal{P}_i, \text{ consider any pair } (u, v) \in \mathcal{P}_i. \text{ If } S \cap \{u, v\} = \emptyset, \text{ then there exists the path } s_i u v t_i \text{ in } G - S, \text{ contradicting that } S \text{ is a multicut for } (G, T, k). \]

\[\text{For the backward direction, let } S \text{ be a set that hits all the pairs in each } \mathcal{P}_i. \text{ From the } S \text{ of contradiction, suppose that there exists a path from } s_i \text{ to } t_i \text{ in } G - S. \text{ Then, from Lemma 34, there is an } s_i \text{ to } t_i \text{ path } s_i u v t_i \text{ such that } u, v \in C. \text{ From the definition of } \mathcal{P}_i, (u, v) \in \mathcal{P}_i. \text{ This contradicts that } S \text{ hits all pairs of vertices in } \mathcal{P}_i. \]

\[\textbf{Proof of Theorem 5.} \text{Let } (G = (C, I), T, k) \text{ be an instance of MC where } G \text{ is a split graph. Construct a graph } G' \text{ with vertex set } V(G') = C \text{ and, } (u, v) \in E(G') \text{ if } (u, v) \in \mathcal{P}_i, \text{ for some } i \in [t]. \text{ Observe that } (u, v) \in \mathcal{P}_i \text{ if and only if } u, v \in N(s_i) \cap N(t_i) \text{ which can be checked in linear time for a pair } (s_i, t_i). \text{ So the total time for constructing the graph } G' \text{ is } \mathcal{O}(|\ell| \cdot (n + m)). \]

\[\text{Then from Lemma 35, } S \text{ is a multicut for } (G, T, k) \text{ if and only if } S \text{ is a vertex cover for } (G', k). \text{ Since VERTEX COVER can be solved in time } \mathcal{O}(1.2739^k + kn) \text{ [2] and the reduction rules can be applied in } \mathcal{O}(\ell \cdot (m + n)) \text{ time, the theorem follows.} \]

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

\[\textbf{Proof.} \text{For the sake of contradiction, suppose that there exists } u \in N(s_i) \text{ and } v \in N(t_i), \text{ such that } u, v \not\in S. \text{ From Lemma 33, since } s_i, t_i \in I, u, v \in C. \text{ Since } C \text{ is a clique, } (u, v) \in E(G) \text{ and } s_i, t_i \not\in S. \text{ Therefore, } s_i u v t_i \text{ is a path in } G - S, \text{ contradicting that } S \text{ is a multicut for } (G = (C, I), T, k). \]

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

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

**Proof of Theorem 7.** Let $(G, T, k)$ be an instance of UMC. If $T = \emptyset$, then return that $(G, T, k)$ is a Yes instance. Otherwise, pick a pair $(s_i, t_i) \in T$. We branch into the following four cases: either $s_i$ belongs to the solution, or $t_i$ belongs to the solution, or when neither $s_i$ nor $t_i$ belongs to the solution. Reduction Rules 1 and 8 can be applied for the pair $(s_i, t_i)$. Then in this case, from Lemma 36, either $N(s_i)$ belongs to the solution or $N(t_i)$ belongs to the solution. Thus, correspondingly to the above four cases we recurse on the following four instances: $(G - \{s_i\}, T_1 = T - s_i, k - 1)$, $(G - \{t_i\}, T_2 = T - t_i, k - 1)$, $(G - N(s_i), T_1 = T - s_i, k - |N(s_i)|)$ and $(G - N(t_i), T_2 = T - t_i, k - |N(t_i)|)$. The correctness of this algorithm again follows from the exhaustiveness of the branching cases and Lemma 36.

Again, $|T_1|, |T_2| < \ell$.

Since we stop when $T = \emptyset$, the depth of the branching tree of this branching algorithm is at most $\ell$. Since at each time, we branch in four branches, the number of leaves in this branching tree is at most $4^\ell$. Also, the time taken at each node is $O(n + m)$ and reduction rules are applicable in time $O(\ell \cdot (n + m))$. Thus, we get an algorithm with running time $O(4^\ell \cdot \ell \cdot (n + m))$. □

**References**


