Subset Feedback Vertex Set in Chordal and Split Graphs

Geevarghese Philip[1], Varun Rajan[2], Saket Saurabh[3,4], and Prafullkumar Tale[5]

[1] Chennai Mathematical Institute, Chennai, India. gphilip@cmi.ac.in
[2] Chennai Mathematical Institute, Chennai, India. varunrajan09@gmail.com
[3] The Institute of Mathematical Sciences, HBNI, Chennai, India. saket@imsc.res.in
[4] Department of Informatics, University of Bergen, Bergen, Norway.
[5] The Institute of Mathematical Sciences, HBNI, Chennai, India. pptale@imsc.res.in

Abstract. In the Subset Feedback Vertex Set (Subset-FVS) problem the input is a graph G, a subset T of vertices of G called the "terminal" vertices, and an integer k. The task is to determine whether there exists a subset of vertices of cardinality at most k which together intersect all cycles which pass through the terminals. Subset-FVS generalizes several well studied problems including Feedback Vertex Set and Multiway Cut. This problem is known to be NP-Complete even in split graphs. Cygan et al. proved that Subset-FVS is fixed parameter tractable (FPT) in general graphs when parameterized by k [SIAM J. Discrete Math (2013)]. In split graphs a simple observation reduces the problem to an equivalent instance of the 3-Hitting Set problem with same solution size. This directly implies, for Subset-FVS *restricted to split graphs*, (i) an FPTalgorithm which solves the problem in $\mathcal{O}^\star(2.076^k)$ time [6][Wahlström, Ph.D. Thesis], and (ii) a kernel of size $\mathcal{O}(k^3)$. We improve both these results for Subset-FVS on split graphs; we derive (i) a kernel of size $\mathcal{O}(k^2)$ which is the best possible unless NP $\subseteq$ coNP/poly, and (ii) an algorithm which solves the problem in time $\mathcal{O}^*(2^k)$. Our algorithm, in fact, solves Subset-FVS on the more general class of *chordal graphs*, also in $\mathcal{O}^*(2^k)$ time.

1 Introduction

In a *covering* or *transversal problem* we are given a universe of elements U, a family $\mathcal{F}$ ($\mathcal{F}$ could be given implicitly) and an integer k and the objective is to check whether there exists a subset of U of size at most k which intersects all the elements of $\mathcal{F}$. Several natural problems on graphs can be framed in the form of such a problem. For instance, consider the classic Feedback Vertex Set (FVS) problem. Here, given a graph G and a positive integer k, the objective is to decide whether there exists a vertex subset X (also called a feedback vertex set) of size at most k which intersects all cycles, that is, $G - X$ is a forest. Other examples include Odd Cycle Transversal, Directed Feedback Vertex Set and Vertex Cover (VC). These problems have been particularly well studied in parameterized complexity [6,4,19,21,24,22].

[6] The $\mathcal{O}^\star()$ notation hides polynomial factors.

Recently, a natural generalization of covering problems has attracted a lot of attention from the point of view of parameterized complexity. In this generalization, apart from U, $\mathcal{F}$ and k, we are also given a subset T of U and the objective is to decide whether there is a subset of U of size at most k that intersects all the sets in $\mathcal{F}$ that contain an element in T. This leads to the *subset variant* of classic covering problems; typical examples include SUBSET FEEDBACK VERTEX SET (SUBSET-FVS), SUBSET DIRECTED FEEDBACK VERTEX SET or SUBSET ODD CYCLE TRANSVERSAL. All these problems have received considerable attention and they have been shown to be *fixed-parameter tractable* (FPT) [6,4,19].

In this paper we study SUBSET-FVS when the input belongs to a more structured families of graphs. This problem was first introduced by Even et al.[9]. The problem generalizes several other well-studied problems like FVS, VC, MULTIWAY CUT [10]. The question whether the SUBSET-FVS problem is fixed parameter tractable (FPT) when parameterized by the solution size was posed independently by Kawarabayashi and the third author in 2009. Cygan et al. [6] and Kawarabayashi and Kobayashi [19] independently answered this question positively in 2011. Wahlström [24] gave the first parameterized algorithm where the dependence on k is $2^{\mathcal{O}(k)}$. Lokshtanov et al. [21] presented a different FPT algorithm which has linear dependence on the input size. On the other hand, Fomin et al. presented a parameter preserving reduction from VC to SUBSET-FVS ([10, Theorem 2.1]) which rules out possibility of an algorithm with subexponential dependence on k under the Exponential-Time Hypothesis. Finally, Hols and Kratsch [18] showed that SUBSET-FVS problem has a randomized polynomial kernelization of size $\mathcal{O}(k^9)$ vertices using matroid-based tools.

While all the results we mentioned above holds for an arbitrary input graphs. There has also been interest in studying SUBSET-FVS on structured families of graphs, such as chordal graphs and split graphs. The problem remains NP-Complete on split graphs which is more restricted subclass of chordal graphs[10]. The stark difference in difficulty between SUBSET-FVS and FVS in this graph class is also witnessed by the fact that known upper bounds on the number of minimal feedback vertex sets and minimal subset feedback vertex sets in split graphs are n^2 and $3^{n/3}$ respectively [10]. It goes without saying that FVS is polynomial time solvable on chordal graphs while SUBSET-FVS is NP-Complete on split graphs. Golovach et al. [14] initiated algorithmic study of SUBSET-FVS on chordal graphs and presented an exact exponential time algorithm to solve SUBSET-FVS on chordal graphs. This algorithm was later improved by Chitnis et al. [3].

In this article, we studyf SUBSET-FVS on chordal and split graphs in the realm of parameterized complexity. For a given set of vertices T, *T-cycle* is a cycle which contains at least one vertex from T. Formally, the problem we study is as follows.

SUBSET FVS IN CHORDAL GRAPHS **Parameter:** k
Input: A chordal graph $G = (V, E)$, a set of *terminal vertices* $T \subseteq V$, and an integer k
Question: Does there exist a set $S \subseteq V$ of at most k vertices of G such that the subgraph $G[V \setminus S]$ contains no T-cycle?

If the input graph in Subset FVS in Chordal Graphs is restricted to split graphs, then the problem will be called Subset FVS in Split Graphs.

A simple observation (Lemma 1) states that in chordal graphs it is enough to intersect all T-triangles to hit all cycles passing through them. This provides a parameter preserving reduction from Subset FVS in Split Graphs to 3-Hitting Set (3-HS). On a positive side, this immediately implies a polynomial compression (in fact, a polynomial kernel) for Subset FVS in Split Graphs of $\mathcal{O}(k^3)$ size [1] and an FPT algorithm running in time $2.076^k n^{\mathcal{O}(1)}$ [23]. On the negative side, when we formulate the problem in terms of 3-HS, we lose structural properties of the input graph. These structural properties can be exploited to obtain better algorithms and smaller kernels for original problems. This idea was recently demonstrated by Le et al. [20] by providing better kernels for several implicit 3-HS problems. This article is written in the same spirit of obtaining better results for implicit 3-hitting set problems by exploiting structural properties of the input graph.

Our results and methods: Our first main result is a non-trivial quadratic kernel for Subset FVS in Split Graphs; improving over the kernel obtained using ideas in designing $\mathcal{O}(k^3)$ kernel for 3-HS [1]. More precisely, we obtain the following.

Theorem 1. Subset FVS in Split Graphs *has a quadratic-size kernel.*

We design the kernel for Subset FVS in Split Graphs using non-trivial uses of expansion lemma – a combinatorial tool central in designing quadratic kernel for Undirected FVS [22]. Our kernelization algorithm works in phases. Given an input graph (G, T, k) and a split partition (K, I) of $V(G)$, where K is a clique and I is an independent set, our algorithm works as follows. We first reduce the input to an instance $(G; T; k)$ where the terminal set T is exactly the independent set I from a split partition (K, I) of G. Then we show that if a (non-terminal) vertex $v \in K$ has at least $k + 1$ neighbours in I then we can include v in a solution or safely delete one edge incident with v; this leads to an instance where each $v \in K$ has at most k neighbours in I. We apply the expansion lemma to this instance to bound the number of vertices in K by $10k$; this gives the bound of $\mathcal{O}(k^2)$ on the number of vertices in I. We also point out that to identify an *irrelevant edge* incident to $v \in K$, we use expansion lemma. In short, we can say that we design a kernel with smaller size by discovering various matching structure and use it to reduce the input. Note that the parameter preserving reduction from VC to Subset FVS in Split Graphs implies that the this kernel can not be improved to kernel of size $\mathcal{O}(k^{2-\epsilon})$ under the assumption that $\mathsf{NP} \not\subseteq \mathsf{coNP/poly}$ [7].

Next, we present an FPT algorithm for Subset FVS in Chordal Graphs that improves over the running time of 3-HS.

Theorem 2. Subset FVS in Chordal Graphs *admits an algorithm with running time $\mathcal{O}(2^k(n + m))$. Here n, m are the number of vertices and the edges of the input graph G, respectively.*

To design our FPT algorithm we look at *clique-tree* defined for the input chordal graph and find an useful vertex to branch on. The structure provided by clique-tree is essential for our improvement. We also note that in running time, dependency on the number of edges and the number of vertices of input graph is linear.

2 Preliminaries

2.1 Graphs

All our graphs are finite, undirected, and simple. We mostly conform to the graph-theoretic notation and terminology from the book of Diestel [8]. We describe the few differences from Diestel and some notation and terms of our own, and reproduce some definitions from Diestel for ease of reference. Let G be a graph. For a vertex v in $V(G)$ we use $N_G(v)$ to denote the *open neighbourhood* $\{u \in V(G) \mid vu \in E(G)\}$ of v, and $N_G[v]$ to denote its closed neighbourhood $N(v) \cup \{v\}$. We drop the subscript G when there is no ambiguity. The *length* of a path or cycle is the number of *edges* in the path (or cycle). An edge uv is a *bridge* if it is not contained in any cycle of G. An edge e in G is a *chord* of a cycle C if (i) both the endvertices of e are in C, and (ii) edge e is not in C. An *induced cycle* is a cycle which has no chord. A vertex v of degree exactly one in a tree T is a *leaf* of the tree, *unless* v is the designated *root* vertex (if one exists) of T. A *matching* M in G is a set of edges no two of which share an endvertex. The endvertices of the edges in M are *saturated* by M. M is *between* vertex sets X, Y if $X \cap Y = \emptyset$ and each edge in M has one end each in X and Y, respectively. A *maximum matching* of G is a matching of the largest size in G.

Let $S \subseteq V(G)$ and $F \subseteq E(G)$ be a vertex subset and an edge subset of G, respectively. We use (i) $G[S]$ to denote the subgraph of G *induced* by S, (ii) $G - S$ to denote the graph $G[V \setminus S]$, and (ii) $G - F$ to denote the graph $(V(G), (E(G) \setminus F))$. A *triangle* is a cycle of length three. Set S is a *feedback vertex set* (FVS) of G if $G - S$ is a forest. A path P (or cycle C) *passes through* S if P (or C) contains a vertex from S. Let $T \subseteq V(G)$ be a specified set of vertices called *terminal* vertices (or *terminals*). A *T-cycle* (T-triangle) is a cycle (triangle) which passes through T. Graph G is a *T-forest* if it contains no T-cycle. Vertex set S is a *subset feedback vertex set* (subset-FVS) of G with respect to terminal set T if the graph $G - S$ is a T-forest. Note that S may contain vertices from T, and that $G - S$ need not be a forest. Set S is a *subset triangle hitting set* (subset-THS) of G with respect to terminal set T if $G - S$ contains no T-triangle. More generally, we say that a vertex v *hits* a cycle C if C contains v. Vertex set S *hits* a set $\mathcal{C}$ of cycles if for each cycle $C \in \mathcal{C}$ there is a vertex $v \in S$ which hits C. We elide the phrase "with respect to T" when there is no ambiguity.

K_n is the complete graph on n vertices. A subset $S \subseteq V(G)$ of vertices of graph G is a *clique* if its vertices are all pairwise adjacent, and is an *independent set* if they are all pairwise non-adjacent. A clique C in G is a *maximal clique* if C is not a *proper* subset of some clique in G. A vertex v of G is a *simplicial vertex* (or is simplicial) in G if $N[v]$ is a clique. In this case we say that $N[v]$ is a *simplicial clique* in G and that v is a simplicial vertex *of* $N[v]$.

Fact 1 ([2], Lemma 3). *Vertex v is simplicial in graph G if and only v belongs to precisely one maximal clique of G, namely the set $N[v]$.*

2.2 Chordal Graphs and Clique Trees

A graph G *chordal* (or *triangulated*) if every induced cycle in G is a triangle; equivalently, if every cycle of length at least four has a chord. If G is a chordal graph then [15]: (i) every induced subgraph of G is chordal; (ii) G has a simplicial vertex, and if G is not a complete graph then G has two non-adjacent simplicial vertices. Whether a graph H is chordal or not can be found in time $\mathcal{O}(|V(H)| + |E(H)|)$, and if H is chordal then a simplicial vertex of H can be found within the same time bound [15].

The number of maximal cliques in a chordal graph G is at most $|V(G)|$ and they can all be enumerated in time $\mathcal{O}(|V(G)| + |E(G)|)$ ([15, Theorem 4.17]). Let $\mathcal{C}(G)$ be the set of maximal cliques of a chordal graph G. A *clique tree*[7] of G is a graph $\mathcal{T}_G$ with the following properties:

1. The vertex set of $\mathcal{T}_G$ is the set $\mathcal{C}(G)$.
2. $\mathcal{T}_G$ is a tree.
3. (Clique Intersection Property) Let C_1, C_2 be any two maximal cliques of G, and let $C' = C_1 \cap C_2$. If C is a maximal clique of G which lies on the path from C_1 to C_2 on $\mathcal{T}_G$, then it is the case that $C' \subseteq C$.

Fact 2.

1. *A connected graph G is chordal if and only if it has a clique tree [2, Theorem 3.1].*
2. *A clique tree $\mathcal{T}_G$ of a chordal graph G can be computed in $\mathcal{O}(|V(G)| + |E(G)|)$ time [13, Theorem 12].*
3. *Let G be a connected chordal graph and $\mathcal{T}_G$ a clique tree of G. For each vertex v of G, the set of all nodes of $\mathcal{T}_G$ which contain v form a connected subgraph (a subtree) of $\mathcal{T}_G$ [2, Theorem 3.2].*

We also need

Observation 1. *Let G be a connected graph with at least two vertices and let $\mathcal{T}_G$ be a clique tree of G. If C is a leaf node of $\mathcal{T}_G$ then C is a simplicial clique in G.*

Proof. Let C be a leaf node of $\mathcal{T}_G$ and let C' be the unique neighbour of C in $\mathcal{T}_G$. Since C is a maximal clique we have that $C \setminus C' \neq \emptyset$. Pick a vertex $v \in C \setminus C'$. Since C is a clique we have that $C \subseteq N[v]$.

If there is a vertex $u \in (N[v] \setminus C)$ then let C'' be a maximal clique which contains the set $\{u, v\}$. Then C'' is distinct from C and C'. Now (i) $v \in (C \cap C'')$, (ii) $v \notin C'$, and (iii) C' is a maximal clique which lies on the path from C to C'' in $\mathcal{T}_G$. This contradicts the Clique Intersection Property of $\mathcal{T}_G$. Hence we get that $N[v] \subseteq C$ as well. Thus $C = N[v]$ is a simplicial clique in G.

[7] We refer readers to monograph of J. Blair and B. Peyton for an introduction to clique trees [2].

2.3 Split Graphs

A graph G is a *split graph* if its vertex set can be partitioned into a clique and an independent set in G. Such a partition is called a *split partition* of G. We use (K, I) to denote the split partitions of the graphs. We refer to vertex sets K and I as the *clique side* and *independent side*, respectively, of graph G. We say that an edge uv in $G[K]$ is *highlighted* if there is a vertex x in I such that the vertices $\{x, u, v\}$ induce a triangle in G. Every split graph is a chordal graph as well [16]. Given a graph G as its adjacency list we can (i) check if G is a split graph, and if it is, (ii) find a partition $V(G) = K \uplus I$ into a clique K and independent set I, both in time $\mathcal{O}(|V(G)| + |E(G)|)$ [17].

If a chordal graph G contains a cycle then it contains an induced cycle, and every induced cycle in G is—by definition—a triangle. Conversely, if G contains a triangle then it trivially contains a cycle. Thus a chordal graph contains a cycle if and only if it contains a triangle. This carries over to subset feedback vertex sets in chordal (and therefore split) graphs in a natural way.

Lemma 1. *Let G be a chordal graph and let $T \subseteq V(G)$ be a specified set of terminal vertices. A vertex subset $S \subseteq V(G)$ is a subset-FVS of G with respect to T if and only if the graph $G - S$ contains no T-triangles.*

Proof. If S is a subset-FVS of G with respect to T then—by definition—the graph $G - S$ contains no T-triangles. For the reverse direction, let S be a subset-THS of G with respect to T, and let $H = G - S$. Since H is an induced subgraph of G we get that H is chordal. Assume for the sake of contradiction that there is a T-cycle in H. Let C be a T-cycle in H of the smallest length ℓ. Then $\ell \geq 4$ and we get that C has a chord. Now this chord is part of two cycles C', C'' such that (i) each of $\{C', C''\}$ has length strictly smaller than C, and (ii) the union of the vertex sets of C' and C'' is the vertex set of C. Thus at least one of $\{C', C''\}$, say C', is a T-cycle H of length strictly smaller than ℓ, a contradiction.

We use $(G; T; k)$ to denote an instance of SUBSET FVS IN CHORDAL GRAPHS or SUBSET FVS IN SPLIT GRAPHS where G is the input graph, T is the specified set of terminals, and k is the parameter.

Corollary 1. *An instance $(G; T; k)$ of* SUBSET FVS IN CHORDAL GRAPHS *(or of* SUBSET FVS IN SPLIT GRAPHS*) is a YES instance if and only if there is a vertex subset $S \subseteq V(G)$ of size at most k such that S is a T-THS of G.*

Lemma 2. *If G is a chordal graph and uv is a bridge in G then the graph $G - \{uv\}$ is also chordal. If H is a* split *graph with at least three vertices on the clique side and uv is an edge with exactly one end in the clique side of H then the graph $H - \{uv\}$ is also split. If uv is a* bridge *in such a split graph H then the graph $H - \{uv\}$ is also split.*

Proof. Let G be a chordal graph. If an edge xy is a chord of some cycle C then xy lies in a cycle C' whose vertex set is a strict subset of the vertex set of C. So we get, since uv is a bridge in G, that uv is not a chord of any cycle in G.

Suppose $G - \{uv\}$ is not chordal. Then it contains an induced cycle C of length four or more. Since edge uv is not present in graph $G - \{uv\}$ it is not present in cycle C either. Thus cycle C is present in graph G. Since uv is not a chord of cycle C we get that C is an induced cycle of length at least four in G, which contradicts the chordality of G.

Now let H be a split graph with at least three vertices on its clique side, and let (K, I) be a split partition of H. Since every edge in $H[K]$ is part of a cycle, any bridge uv of H has one end in each of K, I. The sets K and I remain a clique and an independent set, respectively, once we delete any such edge uv. Hence $H - \{uv\}$ is a split graph.

Since chordal graphs and split graphs are (respectively) closed under taking induced subgraphs [15] we get

Corollary 2. *Let H be a graph obtained from a graph G by repeatedly deleting vertices and/or bridges of the remaining graph at each stage.*

1. *If G is a chordal graph then so is H.*
2. *If G is a split graph and at least three of the clique vertices of G remain in H then H is a split graph.*

Let $(G; T; k)$ be an instance of SUBSET FVS IN CHORDAL GRAPHS. A subset $S \subseteq V(G)$ of vertices of G is a *solution* of this instance if S is a T-FVS (equivalently, a T-THS) of G.

2.4 Parameterized Algorithms and Kernelization

A problem is called *fixed parameter tractable* (FPT) if there exists an algorithm $\mathcal{A}$, a computable function f, and a constant c such that, given any instance $(I; k)$ of the parameterized problem, the algorithm $\mathcal{A}$, correctly decides whether $(I; k)$ is an YES instance or not in time $\mathcal{O}(f(k)|I|^c)$. The algorithm $\mathcal{A}$ is called an FPT algorithm. If constant c is equal to one then we call it as *linear* FPT algorithm. Instances $(I; k)$ and $(I'; k')$ of parameterized problems are *equivalent* if $(I; k)$ is a YES instance if and only if $(I'; k')$ is a YES instance. A *kernel* for a parameterized problem is an algorithm $\mathcal{B}$ that, given an instance (I, k) of the problem, works in time $\mathcal{O}((|I| + k)^c)$ and returns an equivalent instance (I', k') of the same problem. We require that k' is upper bounded by some computable function of k. If there exists a computable function g such that size of an output obtained by algorithm $\mathcal{B}$ for (I, k) is at most $g(k)$, we say that problem admits kernel of size $g(k)$. We refer interested readers to [5] for detailed introduction of the subject.

Let H be a triangle on the vertex set $\{x, y, z\}$. Then $I_{\text{YES}} = (H; \{x\}; 1)$ and $I_{\text{NO}} = (H; \{x\}; 0)$ are constant-size trivial YES and NO instances, respectively, of both SUBSET FVS IN SPLIT GRAPHS and SUBSET FVS IN CHORDAL GRAPHS. Our algorithms make use of *reduction rules* which transform one instance of a problem to another instance of the same problem. We use $(G; T; k)$ to represent the instance given as input to each reduction rule, and $(G'; T'; k')$ to represent the (modified) instance output by the rule. We say that a reduction rule is *safe* if for every input instance $(G; T; k)$ the rule outputs an *equivalent* instance $(G'; T'; k')$. We update $G \leftarrow G', T \leftarrow T', k \leftarrow k'$ to get the input instance $(G; T; k)$ for further processing. We say that an instance $(G; T; k)$ is *reduced* with respect to a reduction rule `RR` if none of the conditions of rule `RR` apply to $(G; T; k)$. Equivalently: instance $(G; T; k)$ is *reduced* with respect to reduction rule `RR` if, when given the instance $(G; T; k)$ as input, rule `RR` produces as output an instance $(G'; T'; k')$ which is identical to $(G; T; k)$. We say that a reduction rule `RR` *applies* to an instance $(G; T; k)$ if $(G; T; k)$ is *not* reduced with respect to `RR`.

2.5 Expansion Lemmas

Let t be a positive integer and G a bipartite graph with vertex bipartition (P, Q). A set of edges $M \subseteq E(G)$ is called a *t-expansion of P into Q* if (i) every vertex of P is incident with exactly t edges of M, and (ii) the number of vertices in Q which are incident with at least one edge in M is exactly $t|P|$. We say that M *saturates* the endvertices of its edges. Note that the set Q may contain vertices which are *not* saturated by M. We need the following generalizations of Hall's Matching Theorem known as *expansion lemmas*:

Lemma 3 ([5] Lemma 2.18). *Let t be a positive integer and G be a bipartite graph with vertex bipartition (P, Q) such that $|Q| \geq t|P|$ and there are no isolated vertices in Q. Then there exist nonempty vertex sets $X \subseteq P$ and $Y \subseteq Q$ such that (i) X has a t-expansion into Y, and (ii) no vertex in Y has a neighbour outside X. Furthermore the sets X and Y can be found in time polynomial in the size of G.*

Lemma 4 ([11]). *Let t be a positive integer and G be a bipartite graph with vertex bipartition (P, Q) such that $|Q| > \ell t$, where ℓ is the size of a maximum matching in G, and there are no isolated vertices in Q. Then there exist nonempty vertex sets $X \subseteq P$ and $Y \subseteq Q$ such that (i) X has a t-expansion into Y, and (ii) no vertex in Y has a neighbour outside X. Furthermore the sets X and Y can be found in time polynomial in the size of G.*

We need sets X, Y of Lemma 4 with an additional property:

Lemma 5. *If the premises of Lemma 4 are satisfied then we can find, in polynomial time, sets X, Y of the kind described in Lemma 4 and a vertex $w \in Y$ such that there exists a t-expansion M from X into Y which does not saturate w.*

Proof. Let X, Y be sets of the kind guaranteed to exist by Lemma 4, and let M be a t-expansion from X into Y. If $|Y| > t|X|$ then there must exist a vertex which is not saturated by M. We can find such a vertex $w \in Y$ by, for instance, deleting each vertex $y \in Y$ in turn and testing if the resulting graph has t-expansion from X into $Y \setminus y$. Thus it is enough to show that we can find, in polynomial time, such a pair X, Y for which $|Y| > t|X|$ holds. We give a proof by algorithm. We start by setting $X = Y = \emptyset$. It holds vacuously that (i) there is a t-expansion from X into Y, and (ii) no vertex in Y has a neighbour outside X, and trivially that $|Y| = t|X|$.

1. Find sets $X' \subseteq P$ and $Y' \subseteq Q$ as guaranteed to exist by Lemma 4. Let M' be a t-expansion from X' into Y'. If $|Y'| > t|X'|$ then return $((X \cup X)', (Y \cup Y'))$. Otherwise, if there is a vertex $w \in Q \setminus Y'$ which has no neighbour in $P \setminus X'$ then return $((X \cup X'), (Y \cup Y' \cup \{w\}))$.
2. At this point we have $|X'| < |P|$ and $|Y'| = t|X'|$. From above we get that there is t-expansion, say M, from X into Y. Since $X \cap X' = \emptyset = Y \cap Y'$ we get that $M \cup M'$ is a t-expansion from $X \cup X'$ into $Y \cup Y'$. Set $\hat{X} \leftarrow X \cup X', \hat{Y} \leftarrow Y \cup Y'$. Then (i) there is a t-expansion from $\hat{X}$ into $\hat{Y}$, (ii) no vertex in $\hat{Y}$ has a neighbour outside $\hat{X}$, and (iii) $|\hat{Y}| = t|\hat{X}|$.
3. Let $\hat{P} = (P - \hat{X}), \hat{Q} = (Q - \hat{Y})$. Consider the subgraph $\hat{G} = G[\hat{P} \cup \hat{Q}]$ and its vertex bipartition $(\hat{P}, \hat{Q})$. Since $t \geq 1$ and the vertices in $\hat{X} \cup \hat{Y}$ are saturated by a t-expansion from $\hat{X}$ into $\hat{Y}$ we get that the subgraph $G[\hat{X} \cup \hat{Y}]$ contains a *matching* of size $|\hat{X}|$. Since the subgraph $\hat{G}$ of G contains none of the vertices saturated by this matching we get that the size $\hat{\ell}$ of a maximum matching in $\hat{G}$ satisfies $\hat{\ell} \leq \ell - |\hat{X}|$.
   Since every vertex in $\hat{Q}$ has at least one neighbour in $\hat{P}$ (otherwise we would have returned in step (1)) we get that there are no isolated vertices in the set $\hat{Q}$ in graph $\hat{G}$. Since $|\hat{Y}| = t|\hat{X}|$ and $|Q| > \ell t$ we have that $|\hat{Q}| = |Q| - t|\hat{X}| > \ell t - t|\hat{X}| = t(\ell - |\hat{X}|) \geq t\hat{\ell}$. Thus graph $\hat{G}$ and its vertex bipartition $(\hat{P}, \hat{Q})$ satisfy the premises of Lemma 4. Set $G \leftarrow \hat{G}, P \leftarrow \hat{P}, Q \leftarrow \hat{Q}, X \leftarrow \hat{X}, Y \leftarrow \hat{Y}$ and go to step (1).

Correctness. Note that before step (1) is executed it is always the case that (i) there is a t-expansion from X into Y, (ii) no vertex in Y has a neighbour outside X, and (iii) $|Y| = t|X|$. So we get that *if* the algorithm terminates (which it does only at step (1)) it returns a correct pair of vertex subsets.

The graph G from the premise of Lemma 4 has a vertex bipartition (P, Q) with $(|P| > 0, |Q| > 0, |Q| > \ell t)$, and the sets $\hat{X}, \hat{Y}$ in steps (2) and (3) satisfy $0 < |\hat{X}| < |P|$ and $|\hat{Y}| = t|\hat{X}|$. So the sets $\hat{P}, \hat{Q}$ of step (3) satisfy $|\hat{P}| > 0, |\hat{Q}| > 0, |\hat{Q}| > \hat{\ell}t$. Thus the graph $\hat{G}$ computed in step (3) has strictly fewer vertices than the graph G passed in to the previous step (1). Since we update $G \leftarrow \hat{G}$ before looping back to step (1), we get that the algorithm terminates in polnomially many steps.

2.6 Branching Rules and Algorithms.

Branching algorithm recursively apply *reduction rules* and *branching rules* to a given instance. To analysis the running time of an algorithm we assign a measure to input instance. Branching rules on instance creates at least two instances which have strictly smaller measures than original instance and branching algorithm solves these instances recursively. Consider an application of branching rule BR to any instance of measure k. Let integer r is strictly greater than one and t_i is a positive real number for every i in $\{1, 2, \ldots, r\}$. Suppose rule BR branches the current instance I into r many instances $I_1, I_2, \ldots, I_r$ of measure $k - t_1, k - t_2, \ldots, k - t_r$ respectively. We say branching rule is *exhaustive* if I is a YES instance if and only if at least one of $I_1, I_2, \ldots, I_r$ is a YES instance. We call $(t_1, t_2, \ldots, t_r)$

as branching vector of rule BR. The running time of the branching algorithm using only branching rule BR is $\mathcal{O}^*(\alpha^k)$, where α is the unique positive real root of $x^k - x^{k-t_1} - x^{k-t_2} \cdots - x^{k-t_r} = 0$ [12, Theorem 2.1]. We use $\mathcal{O}^*$ notation to suppress functions which are polynomial in input size. If algorithm uses more than one branching algorithm then the running time of the algorithm is dominated by running time of branching rule for which this α value is highest.

3 Kernel Bounds for Subset FVS in Split Graphs

In this section we show that Subset FVS in Split Graphs has a quadratic-size kernel with a linear number of vertices on the clique side. We show also that the kernel size is tight unless coNP $\subseteq$ NP/poly.

Theorem 3. Subset FVS in Split Graphs *has a quadratic-size kernel. More precisely: There is a polynomial-time algorithm which, given an instance $(G; T; k)$ of* Subset FVS in Split Graphs, *returns an instance $(G'; T'; k')$ of* Subset FVS in Split Graphs *such that (i) $(G; T; k)$ is YES instance if and only if $(G'; T'; k')$ is YES instance, and (ii) $|V(G')| = \mathcal{O}(k^2)$, $|E(G')| = \mathcal{O}(k^2)$, and $k' \leq k$. Moreover if (K', I') is the split partition of split graph G' then $|K'| \leq 10k$.*

Our algorithm works as follows. We first reduce the input to an instance $(G; T; k)$ where the terminal set T is exactly the independent set I from a split partition (K, I) of G. Then we show that if a (non-terminal) vertex $v \in K$ has at least $k + 1$ neighbours in I then we can include v in a solution or safely delete one edge incident with v; this leads to an instance where each $v \in K$ has at most k neighbours in I. We apply the expansion lemma (Lemma 4) to this instance to bound the number of vertices in K by $10k$; this gives the bound of $\mathcal{O}(k^2)$ on the number of vertices in I.

We now describe the reduction rules. Recall that we use $(G; T; k)$ and $(G'; T'; k')$ to represent the input and output instances of a reduction rule, respectively. We always apply the *first* rule—in the order in which they are described below—which applies to an instance. Thus we apply a rule to an instance *only if* the instance is reduced with respect to all previously specified reduction rules.

Recall that a split graph may have more than one split partition. To keep our presentation short we need to be able to refer to one split partition which "survives" throughout the application of these rules. Towards this we fix an arbitrary split partition $(K^\star, I^\star)$ of the original input graph. Whenever we say "the split partition (K, I) of graph G" we mean the ordered pair $((K^\star \cap V(G)), (I^\star \cap V(G)))$. The only ways in which our reduction rules modify the graph are: (i) delete a vertex, or (ii) delete an edge of the form $uv \; ; \; u \in K^\star, v \in I^\star$. So $((K^\star \cap V(G)), (I^\star \cap V(G)))$ remains a split partition of the "current" graph G at each stage during the algorithm.

Our first reduction rule deals with some easy instances.

Reduction Rule 1. *Recall that (K, I) is the split partition of graph G. Apply the first condition which matches $(G; T; k)$:*

1. *If $T = \emptyset$ then output $I_{\textsf{YES}}$ and stop.*
2. *If $k < 0$, or if $k = 0$* and *there is a T-triangle in G, then output $I_{\textsf{NO}}$ and stop.*
3. *If there is no T-triangle in G then output $I_{\textsf{YES}}$ and stop.*
4. *If $|K| \leq k + 1$ then output $I_{\textsf{YES}}$ and stop.*
5. *If $|K| = k + 2$ and there is an edge uv in $G[K]$ which is* not *highlighted then output $I_{\textsf{YES}}$ and stop.*

Observation 2. *If $(G; T; k)$ is reduced with respect to Reduction Rule 1 then the clique side K of G has size at least three.*

Proof. The second and third parts of the rule ensure that $k \geq 1$. The fourth part now implies $|K| \geq 3$.

Lemma 6. *Reduction Rule 1 is safe.*

Proof. We analyze each part separately.

1. If $T = \emptyset$ then there are no T-cycles in G. So $(G; T; k)$ is (vacuously) a YES instance, as is $I_{\textsf{YES}}$.
2. If $k < 0$ then—since there does not exist a vertex subset S of negative size—$(G; T; k)$ is a NO instance. If the second part of this condition holds then $(G; T; k)$ is clearly a NO instance. Thus in both cases $(G; T; k)$ is a NO instance, as is $I_{\textsf{NO}}$.
3. This condition applies to $(G; T; k)$ only if the previous one does *not* apply. Thus $k \geq 0$ and so $(G; T; k)$ is a YES instance, as is $I_{\textsf{YES}}$.
4. Let $S \subseteq K$ be an arbitrary k-sized set of vertices on the clique side. Deleting S from G gives a graph H with at most one vertex on the clique side. Since there are no edges among vertices on the independent side in H we get that H has no triangles; S is a solution of $(G; T; k)$ of size at most k. Thus $(G; T; k)$ is a YES instance, as is $I_{\textsf{YES}}$.
5. Let $S = K \setminus \{u, v\}$. Then $|S| = k$. Since vertices $\{u, v\}$ have no common neighbour on the independent side of G, we get that $G - S$ contains no triangles. Thus $(G; T; k)$ is a YES instance, as is $I_{\textsf{YES}}$. □

Each remaining rule deletes a vertex or edge from the graph. We use the next two observations in our proofs of safeness.

Observation 3. *Let $(G; T; k)$ be a* YES *instance of* Subset FVS in Split Graphs *which is reduced with respect to Reduction Rule 1.*

1. *Let G' be a graph obtained from G by deleting a vertex $v \in V(G)$, and let $T' = T \setminus \{v\}$. Then $(G'; T'; k' = k)$ is a* YES *instance. If $(G; T; k)$ has a solution S of size at most k with $v \in S$ then $(G'; T'; k' = k - 1)$ is a* YES *instance.*
2. *Let G' be a graph obtained from G by deleting an edge which has exactly one of its endvertices in the clique side of G. Then $(G'; T' = T; k' = k)$ is a* YES *instance.*

Proof. First we consider the case $G' = G - \{v\}$. From Corollary 2 we get that graph G' is a split graph. Since $T' = T \setminus \{v\} \subseteq V(G')$ we get that both $(G'; T'; k)$ and $(G'; T'; k-1)$ are instances of SUBSET FVS IN SPLIT GRAPHS.

Suppose $(G; T; k)$ is a YES instance, and let S be a solution of $(G; T; k)$ of size at most k. Then the graph $G - S$ is a split graph with no T-triangles. We consider two cases:

1. If $v \in S$ then $G' - (S \setminus \{v\}) = (G - \{v\}) - (S \setminus \{v\}) = G - S$. Hence $S \setminus \{v\}$ is a T-THS of the split graph G', of size at most $|S| - 1 = k - 1$. Thus $(G'; T' = T \setminus \{v\}; k' = k-1)$ is a YES instance, and so is $(G'; T' = T \setminus \{v\}; k' = k)$.
2. If $v \notin S$ then the graph $G' - S = (G - \{v\}) - S = G - (S \cup \{v\})$ is obtained by deleting vertex v from $G - S$. Thus $G' - S$ is a split graph. Since deleting a vertex cannot create a new T-triangle, we get that $G' - S$ has no T-triangles. Thus S is an T-THS of G' of size at most k, and $(G'; T'; k' = k)$ is a YES instance.

Now we consider the case $G' = G - \{xy\}$ where edge xy has one end, say x, in the clique side K of G and the other end y in the independent side I. From Observation 2 we get $|K| \geq 3$, and then from Lemma 2 we get that G' is a split graph. Once again, since $T' = T \setminus \{v\} \subseteq V(G')$ we get that both $(G'; T'; k)$ and $(G'; T'; k-1)$ are instances of SUBSET FVS IN SPLIT GRAPHS.

Suppose $(G; T; k)$ is a YES instance, and let $S \subseteq V(G)$ be a solution of $(G; T; k)$ of size at most k. Then graph $G - S$ has no T-cycle. Since deleting an edge cannot introduce a new cycle, we get that the graph $G' - S = (G - S) - \{e\}$ has no T-cycle either. Thus S is a solution of $(G'; T; k)$ of size at most k, and $(G'; T; k)$ is a YES instance.

Observation 4. *Let $(G'; T'; k')$ be a YES instance of* SUBSET FVS IN SPLIT GRAPHS *and let S' be a T'-THS of G' of size at most k'. Let G be a* split *graph which can be constructed from graph G' by adding a vertex v and zero or more edges each incident with the new vertex v. Then both $(G; T_1 = T'; k = k'+1)$ and $(G; T_2 = T' \cup \{v\}; k = k'+1)$ are YES instances of* SUBSET FVS IN SPLIT GRAPHS, *and the set $S' \cup \{v\}$ is a solution of size at most k for both these instances.*

Proof. Since G is a split graph both $(G; T_1; k)$ and $(G; T_2; k)$ are instances of SUBSET FVS IN SPLIT GRAPHS. Since S' is a T'-FVS of G' of size at most k' we have that graph $G' - S'$ has no T'-triangle. Let $S = S' \cup \{v\}$. Then $|S| = |S'| + 1 \leq (k'+1)$ and $G - S = G' - S'$. Thus graph $G - S$ has no T'-triangle. Since $v \notin V(G - S)$ we get that $G - S$ has no triangle which contains vertex v. Thus $G - S$ has no $T' \cup \{v\}$-triangle either. Hence both $(G; T_1; k)$ and $(G; T_2; k)$ are YES instances of SUBSET FVS IN SPLIT GRAPHS and the set $S' \cup \{v\}$ is a solution of size at most k for both these instances.

Reduction Rule 2. *If there is a vertex v of degree zero in G then delete v from G to get graph G'. Set $T' \leftarrow T \setminus \{v\}, k' \leftarrow k$. The reduced instance is $(G'; T'; k')$.*

Since adding or deleting vertices of degree zero does not create or destroy cycles of any kind, we have

Observation 5. *Reduction Rule 2 is safe.*

Reduction Rule 3. *If there is a* non-terminal *vertex v in G which is* not *adjacent to a terminal vertex, then delete v from G to get graph G'. Set $T' \leftarrow T, k' \leftarrow k$. The reduced instance is $(G'; T'; k')$.*

Lemma 7. *Reduction Rule 3 is safe.*

Proof. Let $(G; T; k)$ be an instance given as input to Reduction Rule 3 and let $(G'; T'; k')$ be the corresponding instance output by the rule. Then $G' = G - \{v\}$ where vertex v is as defined by the rule, and $T' = T, k' = k$. From Observation 3 we get that if $(G; T; k)$ is a YES instance then so is $(G'; T'; k')$.

Now suppose $(G'; T'; k')$ is a YES instance, and let $S' \subseteq V(G')$ be a solution of $(G'; T'; k')$ of size at most k'. We claim that S' is a solution of $(G; T; k)$ as well. Suppose not; then graph $G - S'$ has a T-cycle. Since $T' = T$ and $G' - S' = G - (S' \cup \{v\})$ does not contain any T'-cycle we get that every T-cycle in $G - S'$ must contain vertex v.

Let C be a *shortest* T-cycle of $G - S'$, and let t be a terminal vertex in C. Since—by assumption—vertices v and t are not adjacent we get that C contains at least four vertices. Since C is a cycle of length at least four in the chordal graph $G - S'$ we get that C has a chord. This chord is part of two cycles C', C'' such that (i) each of $\{C', C''\}$ has length strictly smaller than C, and (ii) the union of the vertex sets of C' and C'' is the vertex set of C. At least one of the two cycles C', C'' contains the terminal t; assume that t is in C'. If vertex v is also in C' then C' is a T-cycle of $G - S'$ which is *shorter* than C; this contradicts our assumption about C. If v is *not* in C' then C' is a T-cycle—and hence T'-cycle—in $G - (S' \cup \{v\}) = G' - S'$, which contradicts our assumption that S' is a solution of $(G'; T'; k')$.

Thus S' is a solution of $(G; T; k)$ of size at most $k' = k$, and $(G; T; k)$ is a YES instance.

Reduction Rule 4. *If there is a* bridge *e in G then delete edge e (*not *its endvertices) to get graph G'. Set $T' \leftarrow T, k' \leftarrow k$. The reduced instance is $(G'; T'; k')$.*

Lemma 8. *Reduction Rule 4 is safe.*

Proof. Let $(G; T; k)$ be an instance given as input to Reduction Rule 4 and let $(G'; T'; k')$ be the corresponding instance output by the rule. Then $G' = G - \{e\}$ where e is a bridge in G, $T' = T, k' = k$.

From Observation 3 we get that if $(G; T; k)$ is a YES instance then so is $(G'; T'; k')$.

Now suppose $(G'; T'; k')$ is a YES instance, and let $S' \subseteq V(G')$ be a solution of $(G'; T'; k')$ of size at most k'. Observe that since (i) e is a bridge in G, and (ii) deleting vertices does not introduce new cycles, edge e, if it exists in graph $G - S'$, is a bridge in $G - S'$ as well. So e cannot be in *any* cycle in $G - S'$. Hence if graph $G - S'$ has a T-cycle C then C does not contain edge e, which implies that C is present in the graph $G' - S' = (G - S') - \{e\}$ as well. But this contradicts our assumption that S' is a solution of $(G'; T'; k')$. Thus there cannot be a T-cycle in $G - S'$. So S' is a solution of $(G; T; k)$ of size at most $k' = k$, and $(G; T; k)$ is a YES instance.

Lemma 9. *Let $(G; T; k)$ be an instance of* SUBSET FVS IN SPLIT GRAPHS *which is reduced with respect to Reduction Rules 1, 2, 3, and 4. Then*

1. *Each vertex in G has degree at least two.*
2. *Every vertex in G is part of some T-triangle.*
3. *If $(G; T; k)$ is a YES instance then every terminal vertex on the clique side of G is present in* every *solution of $(G; T; k)$ of size at most k.*

Proof. We prove each claim in turn. Let (K, I) be the split partition of G.

1. Since $(G; T; k)$ is reduced with respect to Reduction Rule 2 we get that every vertex in G has degree at least one. If vertex v has degree exactly one then the only edge incident on v is a bridge, which cannot exist since $(G; T; k)$ is reduced with respect to Reduction Rule 4. Thus every vertex in G has degree at least two.
2. From Observation 2 we get $|K| \geq 3$. Consider a vertex $v \in K$. Since $(G; T; k)$ is reduced with respect to Reduction Rule 3 we get that v is adjacent to at least one terminal vertex t. If $t \in K$ then v is part of a T-triangle which contains t. If $t \in I$ then let $u \in K$; $u \neq v$ be another neighbour of t in K. Such a neighbour exists because t has degree at least two and every neighbour of t is in K. Since uv is an edge in $G[K]$ we get that $\{t, u, v\}$ is a T-triangle which contains vertex v.

   Now suppose v is a vertex in I. Then v has at least two neighbours $x, y \in K$ for which xy is an edge. If v is a terminal then it belongs to the T-triangle $\{v, x, y\}$. If v is *not* a terminal then—since $(G; T; k)$ is reduced with respect to Reduction Rule 3—we get that v is adjacent to at least one terminal vertex, which has to be in K. Set x to be such a terminal neighbour of v. Then v belongs to the T-triangle $\{v, x, y\}$.
3. Suppose not. Let S be a solution of $(G; T; k)$ of size at most k, and let $t \in (K \cap T) \setminus S$ be a terminal vertex on the clique side of G which is not in S. If there are two other vertices x, y on the clique side which are also not in S then $\{t, x, y\}$ is a T-triangle in $G - S$, a contradiction. So we have that $|K \setminus S| \leq 2$. Now since $(G; T; k)$ is reduced with respect to Reduction Rule 1 we have—part (4) of the rule—that $|K| \geq k + 2 = |S| + 2$, from which we get $|K \setminus S| \geq 2$. Thus $|K \setminus S| = 2$. Substituting these in the identity $|K| = |K \setminus S| + |K \cap S|$ we get $|S| \leq |K \cap S|$ which implies $|S| = |K \cap S|$ and $S = K \cap S$.

   Thus we get that K is of the form $K = S \cup \{t, x\}$ for some vertex x. Now from part (5) of Reduction Rule 1 we get that vertices t and x have a common neighbour, say y, in set I. So the T-triangle $\{t, x, y\}$ is present in graph $G - S$, a contradiction. Hence t must be in S. □

It is thus safe to pick a terminal vertex from the clique side into the solution.

Reduction Rule 5. *If there is a terminal vertex t on the clique side then delete t to get graph G'. Set $T' \leftarrow T \setminus \{t\}, k' \leftarrow k - 1$. The reduced instance is $G'; T'; k')$.*

Lemma 10. *Reduction Rule 5 is safe.*

Proof. Suppose $(G; T; k)$ is a YES instance. From Lemma 9 we get that vertex t is present in *every* solution of $(G; T; k)$ of size at most k, and so from Observation 3 we get that $(G'; T'; k')$ is a YES instance.

If $(G'; T'; k')$ is a YES instance then we get from Observation 4 that $(G; T; k)$ is a YES instance as well.

Observation 6. *Let $(G; T; k)$ be reduced with respect to Reduction Rules 1, 2, 3, and 5. Let (K, I) be the split partition of G. Then $T = I$ and every vertex in K has a neighbour in I.*

Proof. Since $(G; T; k)$ is reduced with respect to Reduction Rule 5 we have that no vertex on the clique side of G is a terminal. Thus $T \subseteq I$. Suppose there is a non-terminal vertex $v \in I$. From part (2) of Lemma 9 we get that v must be adjacent to some terminal vertex. This cannot happen because every neighbour of v is in K and none of them is a terminal. So every vertex in I is a terminal. Thus $I \subseteq T$, and hence $T = I$.

Every vertex in K is a non-terminal, and every non-terminal is adjacent to some terminal vertex. So every vertex in K must have a neighbour in I.

Our kernelization algorithm can be thought of having two main parts: (i) bounding the number of vertices on the clique side by $\mathcal{O}(k)$, and (ii) bounding the number of independent set vertices in the neighbourhood of each clique-side vertex by k. We now describe the second part. We need some more notation. For a vertex $v \in K$ on the clique side of graph G we use (i) $N_1(v)$ for the set of neighbours $N(v) \cap I$ of v on the independent side I, and (ii) $N_2(v)$ to denote the set of all *other* clique vertices—than v—which are adjacent to some vertex in $N_1(v)$; that is, $N_2(v) = N(N_1(v)) \setminus \{v\}$. Informally, $N_2(v)$ is the second neighbourhood of v "going via I". We use $B(v)$ to denote the bipartite graph obtained from $G[N_1(v) \cup N_2(v)]$ by deleting every edge with both its endvertices in $N_2(v)$. Equivalently: Let H be the (bipartite) graph obtained by deleting, from G, every edge which has both its ends on the clique side of G. Then $B(v) = H[N_1(v) \cup N_2(v)]$. We call $B(v)$ the bipartite graph *corresponding to* vertex $v \in K$.

Bounding the Independent-side Neighbourhood of a Vertex on the Clique Side The first reduction rule of this part applies when there is a vertex $v \in K$ which is part of more than k T-triangles and these T-triangles are pairwise vertex-disjoint apart from the one common vertex v. In this case any solution of size at most k must contain v, so we delete v and reduce k.

Lemma 11. *Let $v \in K$ be a vertex on the clique side of graph G such that the bipartite graph $B(v)$ contains a* matching *of size at least $k + 1$. Then* every *T-FVS of G of size at most k contains v.*

Proof. Suppose not; let $S \subseteq V(G)$; $|S| \leq k$ be a T-FVS of G of size at most k such that $v \notin S$. Let $M = \{x_1y_1, \ldots, x_{k+1}y_{k+1}\}$ be a matching in graph $B(v)$. Note that every edge in M is present in graph G. Further, each edge $x_iy_i \in M$ (i) has one end in the clique side K of G and the other end in the independent side I, and (ii) forms a triangle $\{v, x_i, y_i\}$ in G together with vertex v. Since every vertex in I is a terminal we get that each triangle of the form $\{v, x_i, y_i\}$ is a T-triangle.

Now since S is a T-FVS of G it has a non-empty intersection with every triangle in the set $\{\{v, x_i, y_i\} \ ; \ 1 \leq i \leq (k+1)\}$. Since $v \notin S$ we get that S has a non-empty intersection with every set in the collection $\{\{x_i, y_i\} \ ; \ 1 \leq i \leq (k+1)\}$. Since the vertex pairs $\{x_i, y_i\} \ ; \ 1 \leq i \leq (k+1)$ are pairwise disjoint we get that S contains at least $k+1$ distinct vertices from the set $\cup_{i=1}^{k+1}\{x_i, y_i\}$. This contradicts the assumption $|S| \leq k$.

Reduction Rule 6. *If there is a vertex v on the clique side K of graph G such that the bipartite graph $B(v)$ has a* matching *of size at least $k+1$ then delete vertex v from G to get graph G'. Set $T' \leftarrow T, k' \leftarrow k-1$. The reduced instance is $(G'; T'; k')$.*

Lemma 12. *Reduction Rule 6 is safe.*

Proof. Since—Observation 6—vertex v is not a terminal vertex in G we get that $T' \subseteq V(G')$. Suppose $(G; T; k)$ is a YES instance. Then since v is in present in *every* solution of $(G; T; k)$ of size at most k—Lemma 11—we get from Observation 3 that $(G'; T'; k')$ is a YES instance.

If $(G'; T'; k')$ is a YES instance then from Observation 4 we get that $(G; T; k)$ is a YES instance as well.

Let $(G; T; k)$ be an instance which is reduced with respect to Reduction Rule 6. We show that if there is a vertex $v \in K$ on the clique side of G which has more than k neighbours in the independent side I, then we can find an *edge* of the form $vw \ ; \ w \in I$ which can safely be deleted from the graph. We get this by a careful application of the "matching" version (Lemma 4) of the Expansion Lemma together with Lemma 5. Let v be such a vertex and let $P = N_2(v), Q = N_1(v), t = 1$. Then (P, Q) is a bipartition of the graph $B(v)$ corresponding to vertex v. Let $\ell \leq k$ be the size of a maximum matching of $B(v)$. Note that $|Q| \geq (k+1) > \ell t$ and that—by part (1) of Lemma 9—there are no isolated vertices in set $|Q|$. Thus Lemma 5 applies to graph $B(v)$ together with $P, Q, t = 1$. Since a 1-expansion from X into Y contains a matching between X and Y which saturates X we get

Corollary 3. *Let $(G; T; k)$ be an instance which is reduced with respect to Reduction Rule 6. Suppose there is a vertex $v \in K$ on the clique side of G which has more than k neighbours in the independent side I. Then we can find, in polynomial time, non-empty vertex sets $X \subseteq N_2(v) \subseteq K, Y \subseteq N_1(v) \subseteq I$ and a vertex $w \in Y$ such that (i) there is a matching M between X and Y which saturates every vertex of X and does* not *saturate w, and (ii) $N_G(Y) = X \cup \{v\}$.*

Lemma 13. *Let $(G; T; k)$ be an instance which is reduced with respect to Reduction Rule 6, and let $v \in K$ be a vertex on the clique side which has more than k*

neighbours in the independent side I. Let $X \subseteq K, w \in Y \subseteq I, M \subseteq E(G[X \cup Y])$ be as guaranteed to exist by Corollary 3. Let $G' = G - \{vw\}$, and let S' be a T-THS of G' of size at most k. If $v \notin S'$ then $(S' \setminus Y) \cup X$ is a T-THS of G' of size at most k.

Proof. Let $M = \{x_1y_1, \ldots, x_{|X|}y_{|X|}\}$ be a matching in G between X and Y which saturates all of X and does not saturate w. Since $vw \notin M$ we get that matching M is present in graph G' as well. Thus $\{\{v, x_1, y_1\}, \ldots, \{v, x_{|X|}, y_{|X|}\}\}$ is a set of $|X|$-many T-triangles in G' which pairwise intersect exactly in $\{v\}$. Let S' be a T-THS of G' of size at most k which does *not* contain v. Then S' contains at least one vertex from each of the sets $\{x_i, y_i\}$; $1 \leq i \leq |X|$. Let $\hat{S} = (S' \setminus Y) \cup X$. Then we can get $\hat{S}$ from S' as follows.

- For each edge $x_iy_i \in M$,
  - if $S' \cap \{x_i, y_i\} = \{y_i\}$ then delete y_i from S' and add x_i, and,
  - if $S' \cap \{x_i, y_i\} = \{x_i, y_i\}$ then delete y_i from S' (and don't add anything).
- Delete all of $Y \setminus \{y_1, \ldots, y_{|X|}\}$ from S'.

Thus to get $\hat{S}$ from S' we add at most as many vertices as we delete, and so it is the case that $|\hat{S}| \leq |S'| \leq k$.

Consider the induced subgraphs $H' = G' - S'$ and $\hat{H} = G' - \hat{S}$ of G'. Of these H' has no T-triangles. Every vertex of $\hat{H}$ which is *not* present in H' belongs to the set Y. So if $\hat{H}$ contains a T-triangle then each such T-triangle must contain a vertex from Y. Now from Corollary 3 we get that $N(Y) \subseteq X \cup \{v\}$ and by definition we have that no vertex in X is present in $\hat{H}$. Thus every vertex in Y has degree at most one in $\hat{H}$. So no vertex in Y is in any T-triangle in $\hat{H}$. Hence there are no T-triangles in $\hat{H}$. Thus $\hat{S} = (S' \setminus Y) \cup X$ is a T-THS of G' of size at most k.

Reduction Rule 7. *If there is a vertex v on the clique side K of graph G such v has more than k neighbours in the independent side I, then find a vertex $w \in I$ as described by Corollary 3 and delete the edge vw to get graph G'. Set $T' \leftarrow T, k' \leftarrow k$. The reduced instance is $(G'; T'; k')$.*

Lemma 14. *Reduction Rule 7 is safe.*

Proof. If $(G; T; k)$ is a YES instance then we get from Observation 3 that $(G' = G - \{vw\}; T'; k')$ is a YES instance.

Now suppose $(G'; T'; k')$ is a YES instance. Let S' be a T-THS of G' of size at most k. If $v \in S'$ then we have $G - S' = G' - S'$ and in this case S' is a T-THS of G as well, of size at most k. If $v \notin S'$ then from Lemma 13 we get that $\hat{S} = (S' \setminus Y) \cup X$ is a T-THS of G' of size at most k. Thus the graph $H' = G' - \hat{S}$ has no T-triangles. The only difference between graphs H' and $H = G - \hat{S}$ is that the latter graph has the extra edge vw. So if H contains a T-triangle then each such T-triangle must contain both the vertices $\{v, w\}$.

From Corollary 3 we know that $N(w \in Y) \subseteq X \cup \{v\}$. Thus vertex w has no neighbours in the graph $H' = (G - \{vw\}) - \hat{S}$, and has exactly one neighbour—namely, vertex v—in the graph $H = G - \hat{S}$. So w is not part of any triangle in graph H. Thus $\hat{S} = (S' \setminus Y) \cup X$ is a T-THS of graph G of size at most k.

We now show how to bound the number of vertices on the clique side K of an instance $(G; T; k)$ which is reduced with respect to Reduction Rule 7.

Bounding the Size of the Clique Side We partition the clique side K into three parts and bound the size of each part separately. To do this we first find a 3-approximate solution $\tilde{S}$ to $(G; T; k)$. For this we initialize $\tilde{S} \leftarrow \emptyset$ and iterate as follows: If there is a vertex v in the independent side I such that v is part of a triangle $\{v, x, y\}$ in the graph $G - \tilde{S}$—note that in this case $\{x, y\} \subseteq K$—then we set $\tilde{S} \leftarrow \tilde{S} \cup \{v, x, y\}$. We repeat this till there is no such vertex $v \in I$ or till $|\tilde{S}|$ becomes larger than $3k$, whichever happens first.

Reduction Rule 8. *Let $(G; T; k)$ be an instance which is reduced with respect to Reduction Rule 7 and let $\tilde{S}$ be the set constructed as described above. If $|\tilde{S}| > 3k$ then return I_{NO}.*

Lemma 15. *Reduction Rule 8 is safe.*

Proof. By Observation 9 we have that the terminal set T of graph G is exactly its independent side I. Hence each triangle whose vertex set is added to $\tilde{S}$ by the construction is a T-triangle, and these vertex sets are pairwise disjoint. If $|\tilde{S}| > 3k$ then graph G contains more than k pairwise vertex-disjoint T-triangle and therefore is a NO instance, as is I_{NO}.

At this point we have that the cardinality of the approximate solution $\tilde{S}$ is at most $3k$. We now partition the sets K, I into three parts each and bound each part separately (See Figure 1.):

- $K_{\tilde{S}}$ is the set of clique-side vertices included in $\tilde{S}$: $K_{\tilde{S}} = K \cap \tilde{S}$.
- $I_{\tilde{S}}$ is the set of independent-side vertices included in $\tilde{S}$: $I_{\tilde{S}} = I \cap \tilde{S}$.
- K_0 is the set of clique-side vertices not in $\tilde{S}$ whose neighbourhoods in the independent side I are all contained in $I_{\tilde{S}}$: $K_0 = \{u \in (K \setminus K_{\tilde{S}}) \; ; \; (N(u) \cap I \subseteq I_{\tilde{S}}\}$;
- I_0 is the set of independent-side vertices not in $\tilde{S}$ whose neighbourhoods are all contained in $K_{\tilde{S}}$: $I_0 = \{v \in I \setminus I_{\tilde{S}} \; ; \; N(v) \subseteq K_{\tilde{S}}\}$
- K_1, I_1 are the remaining vertices in each set: $K_1 = K \setminus (K_{\tilde{S}} \cup K_0)$ and $I_1 = I \setminus (I_{\tilde{S}} \cup I_0)$.
- K_1 is the set of clique-side vertices not in $\tilde{S}$ which have at least one neighbour in I outside of $I_{\tilde{S}} \cup I_0$. Equivalently, it is the set of clique-side vertices not in $K_{\tilde{S}} \cup K_0$: $K_1 = K \setminus (K_{\tilde{S}} \cup K_0)$.
- I_1 is the set of independent-side vertices which are not in $I_{\tilde{S}} \cup I_0$:$I_1 = I \setminus (I_{\tilde{S}} \cup I_0)$. Since $\tilde{S}$ is a solution each vertex in I_1—being a terminal—can have exactly one neighbour in K_1.

We list some simple properties of this partition which we need later in our proofs.

Observation 7. *$|K_{\tilde{S}}| \leq 2k$ and $|I_{\tilde{S}}| \leq k$. Each vertex in K_1 has (i) no neighbour in I_0 and (ii) at least one neighbour in I_1. Each vertex in I_1 has exactly one neighbour in K_1. The bipartite graph obtained from $G[K_1 \cup I_1]$ by deleting all the edges in $G[K_1]$ is a forest where each connected component is a star.*

Proof. It follows directly from the construction that $|K_{\tilde{S}}| \leq 2k$ and $|I_{\tilde{S}}| \leq k$.

Let v be a vertex in K_1. Then $v \notin K_{\tilde{S}}$ by construction. If v has a neighbour $w \in I_0$ then $w \in I_0$ has a neighbour outside of $K_{\tilde{S}}$, a contradiction.

Since I is the set of terminals we get—Observation 3—that vertex v has at least one neighbour in the set I. Since the vertices in I_0 do not have neighbours outside the set $K_{\tilde{S}}$ we get that $v \notin K_{\tilde{S}}$ does not have a neighbour in the set I_0. So if v has no neighbour in I_0 then its neighbourhood on the independent side is contained in the set $I_{\tilde{S}}$, which implies that v is in K_0 and not in $K_{\tilde{S}}$, a contradiction.

If a vertex $v \in I_1$ has two neighbours x, y in the set K_1 then—since $v \in I$ is a terminal—the vertices $\{v, x, y\}$ form a T-triangle which does not intersect the T-THS $\tilde{S} = K_{\tilde{S}} \cup I_{\tilde{S}}$, a contradiction. So each vertex in I_1 has exactly one neighbour in K_1, which implies that the bipartite graph obtained from $G[K_1 \cup I_1]$ by deleting all the edges in $G[K_1]$ is a forest where each connected component is a star.

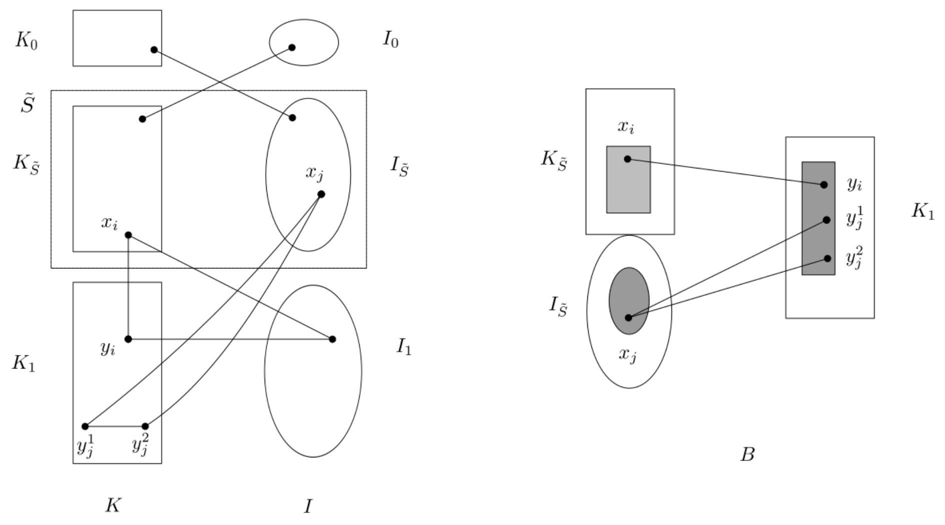


Fig. 1. Left side figure shows the partition of $V(G)$ as described after Lemma 15. On the right side, we have graph B as described above Corollary 5. Shaded regions in $\tilde{S}$ and K_1 represents sets X, Y, respectively, as mentioned in the same corollary.

Let H be the bipartite graph obtained from $G[I_{\tilde{S}} \cup K_0]$ by deleting all the edges in $G[K_0]$. Since—Observation 6—every vertex in the set K_0 has at least one neighbour in the set I and since $(N(K_0) \cap I) \subseteq I_{\tilde{S}}$ by construction, we get that there are no isolated vertices in graph H. So if $|K_0| \geq 2|I_{\tilde{S}}|$ then Lemma 3 applies to graph H with $P \leftarrow I_{\tilde{S}}, Q \leftarrow K_0, t \leftarrow 2$ and we get

Corollary 4. *Let $(G; T; k)$ be an instance which is reduced with respect to Reduction Rule 8, and let the sets $K_{\tilde{S}}, K_0, K_1, I_{\tilde{S}}, I_0, I_1$ be as described above. If $|K_0| \geq 2|I_S|$ then we can find, in polynomial time, non-empty vertex sets $X \subseteq I_{\tilde{S}} \subseteq I, Y \subseteq K_0 \subseteq K$ such that (i) X has a 2-expansion M into Y, and (ii) $N_G(Y) = X$.*

Lemma 16. *Let $(G; T; k)$ be an instance which is reduced with respect to Reduction Rule 8, and let the sets $K_{\tilde{S}}, K_0, K_1, I_{\tilde{S}}, I_0, I_1$ be as described above. Suppose $|K_0| \geq 2|I_{\tilde{S}}|$, and let $X \subseteq I_{\tilde{S}} \subseteq I, Y \subseteq K_0 \subseteq K, M \subseteq E(G[X \cup Y])$ be as guaranteed to exist by Corollary 4. If S is a T-THS of graph G of size at most k then $(S \setminus Y) \cup X$ is also a T-THS of G of size at most k.*

Proof. Let $X = \{x_1, \ldots, x_{|X|}\}$. For each vertex $x_i \in X$ let $x_i y_i^1, x_i y_i^2$ be the two edges in M which are incident with x_i. Then the set $\{y_1^1, y_1^2, \ldots, y_{|X|}^1, y_{|X|}^2\} \subseteq Y$ has size exactly $2|X|$, and the $|X|$-many sets $\{\{x_1, y_1^1, y_1^2\}, \ldots, \{x_{|X|}, y_{|X|}^1, y_{|X|}^2\}\}$ form pairwise vertex-disjoint T-triangles in G.

The T-THS S of G of size at most k contains at least one vertex from each of the sets $\{\{x_i, y_i^1, y_i^2\} \; ; \; 1 \leq i \leq |X|\}$. Let $\hat{S} = (S \setminus Y) \cup X$. Then we can get $\hat{S}$ from S as follows.

- For each triangle $\{\{x_i, y_i^1, y_i^2\} \; ; \; 1 \leq i \leq |X|\}$,
  - if $x_i \in S$ then set $S \leftarrow S \setminus \{y_i^1, y_i^2\}$;
  - if $x_i \notin S$ then set $S \leftarrow (S \setminus \{y_i^1, y_i^2\}) \cup \{x_i\}$.
- Delete all of $Y \setminus \{y_1^1, y_1^2, \ldots, y_{|X|}^1, y_{|X|}^2\}$ from S.

Thus to get $\hat{S}$ from S we add at most as many vertices as we delete, and so it is the case that $|\hat{S}| \leq |S| \leq k$.

Consider the induced subgraphs $H = G - S$ and $\hat{H} = G - \hat{S}$ of G. Of these H has no T-triangles. Every vertex of $\hat{H}$ which is *not* present in H belongs to the set Y. So if $\hat{H}$ contains a T-triangle then each such T-triangle must contain a vertex from Y. Now from Corollary 4 we get that $N(Y) = X$ and by definition we have that no vertex in X is present in $\hat{H}$. Thus each vertex in Y has degree zero in $\hat{H}$, and so is not in any T-triangle in $\hat{H}$. Hence there are no T-triangles in $\hat{H}$. Thus $\hat{S} = (S \setminus Y) \cup X$ is a T-THS of G' of size at most k.

Reduction Rule 9. *If $|K_0| \geq 2|I_{\tilde{S}}|$ then find sets $X \subseteq I_{\tilde{S}}$ and $Y \subseteq K_0$ as described by Corollary 4. Set $G' \leftarrow G - X, T' \leftarrow T \setminus X, k' \leftarrow k - |X|$. The reduced instance is $(G'; T'; k')$.*

Lemma 17. *Reduction Rule 9 is safe.*

Proof. If $(G; T; k)$ is a YES instance then we get from Lemma 16 that G has a T-THS S of size at most k which contains all of X. By applying part (1) of Observation 3 $|X|$ times we get that $G' = G - X, T' = T \setminus X, k' = k - |X|$ is a YES instance.

Now suppose $(G';T';k')$ is a YES instance. Observe that we can get graph G from G' by adding, in turn, each vertex $x \in X$ and some edges incident on x, and also that every vertex that we add in this process is a terminal vertex in graph G. Hence by applying Observation 4 $|X|$ times we get that $(G;T = T' \cup X; k = k' + |X|)$ is a YES instance.

At this point we have the bounds $|K_{\tilde{S}}| \leq 2k$ and $|K_0| < 2|I_{\tilde{S}}| = 2k$. We now use a more involved application of the Expansion Lemma to bound the size of the remaining part K_1 of the clique side. The general idea is that if K_1 is at least twice as large as the approximate solution $\tilde{S}$ then the 2-expansion which exists between subsets of these two sets will yield a non-empty set of "redundant" vertices in K_1.

Consider the bipartite graph B obtained from the induced subgraph $G[\tilde{S} \cup K_1]$ of G by (i) deleting all the edges in the two induced subgraphs $G[\tilde{S}]$ and $G[K_1]$, respectively, and (ii) deleting every edge $uv\ ;\ u \in K_{\tilde{S}}, v \in K_1$ *if and only if* there is *no* vertex $w \in I_1$ such that $\{u, v, w\}$ form a triangle in G. Consider a vertex $v \in K_1$. If v has a neighbour $w \in I_{\tilde{S}}$ then the edge vw is present in graph B and so v is not isolated in B. Now suppose v has no neighbour in $I_{\tilde{S}}$. From the construction we know that v has no neighbour in I_0 either. Then from Lemma 9 and Observation 6 we get that there is a triangle $\{v, x, y\}$ in G where $x \in (I \setminus (I_0 \cup I_{\tilde{S}})) = I_1$ and $y \in K$. Now by construction vertex $x \in I_1$ has no neighbour in the set K_0, and from Observation 7 we get that x has no neighbour other than v in the set K_1. Thus we get that $y \in K_{\tilde{S}}$, and hence that the edge vy survives in graph B. Hence v is not isolated in B in this case either. So if $|K_1| \geq 2|\tilde{S}|$ then Lemma 3 applies to the bipartite graph B with $P \leftarrow \tilde{S}, Q \leftarrow K_1, t \leftarrow 2$ and we get

Corollary 5. *Let $(G;T;k)$ be an instance which is reduced with respect to Reduction Rule 9, and let the sets $K_1, \tilde{S}, K_{\tilde{S}}, I_{\tilde{S}}, K$ and the bipartite graph B be as described above. If $|K_1| \geq 2|\tilde{S}|$ then we can find, in polynomial time, non-empty vertex sets $X \subseteq \tilde{S} = (K_{\tilde{S}} \cup I_{\tilde{S}}), Y \subseteq K_1 \subseteq K$ such that (i) X has a 2-expansion M into Y, and (ii) $N_B(Y) = X$.*

Lemma 18. *Let $(G;T;k)$ be an instance which is reduced with respect to Reduction Rule 8, and let the sets $K_1, \tilde{S}, K_{\tilde{S}}, I_{\tilde{S}}, K$ and the bipartite graph B be as described above. Suppose $|K_1| \geq 2|\tilde{S}|$, and let $X \subseteq \tilde{S} = (K_{\tilde{S}} \cup I_{\tilde{S}}), Y \subseteq K_1 \subseteq K, M \subseteq E(G[X \cup Y])$ be as guaranteed to exist by Corollary 4. If S is a T-THS of graph G of size at most k then $(S \setminus Y) \cup X$ is also a T-THS of G of size at most k.*

Proof. Let $X = \{x_1, \ldots, x_{|X|}\}$. Without loss of generality, let $X \cap K_{\tilde{S}} = \{x_1, \ldots, x_\ell\}$ and $X \cap I_{\tilde{S}} = \{x_{\ell+1}, \ldots, x_{|X|}\}$. Note that at most one of these two sets could be the empty set; this does not affect the remaining arguments.

For each vertex $x_i \in (X \cap K_{\tilde{S}})$ let $x_i y_i$ be an edge in M which is incident with x_i. Then we know by the construction that there is a vertex $z_i \in I_1$ such that $\{x_i, y_i, z_i\}$ is a T-triangle in G. From Observation 7 we get that the vertices $z_i \in I_1\ ;\ 1 \leq i \leq \ell$ are pairwise disjoint. Now for each vertex $x_j \in (X \cap I_{\tilde{S}})$ let $x_j y_j^1, x_j y_j^2$ be the two edges in M which are incident with x_j. Then we get that the vertices $\{x_j, y_j^1, y_j^2\}$ form a T-triangle in G. Putting these together we get that the $|X|$-many sets $\{\{x_1, y_1, z_1\}, \ldots, \{x_\ell, y_\ell, z_\ell\}, \{x_\ell, y_\ell^1, y_\ell^2\}, \ldots, \{x_{|X|}, y_{|X|}^1, y_{|X|}^2\}\}$ form pairwise vertex-disjoint T-triangles in G. The T-THS S of G of size at most k contains at least one vertex from each of these $|X|$ sets. Let $\hat{S} = (S \setminus Y) \cup X$. Then we can get $\hat{S}$ from S as follows.

- For each triangle $\{\{x_i, y_i, z_i\}\ ;\ 1 \leq i \leq \ell\}$,
  - if $x_i \in S$ then set $S \leftarrow S \setminus \{y_i\}$;
  - if $x_i \notin S$ and $y_i \in S$ then set $S \leftarrow (S \setminus \{y_i\}) \cup \{x_i\}$.
- For each triangle $\{\{x_j, y_j^1, y_j^2\}\ ;\ \ell + 1 \leq j \leq |X|\}$,
  - if $x_j \in S$ then set $S \leftarrow S \setminus \{y_j^1, y_j^2\}$;
  - if $x_j \notin S$ then set $S \leftarrow (S \setminus \{y_j^1, y_j^2\}) \cup \{x_j\}$.
- Delete all of $Y \setminus \{y_1 \ldots, y_\ell, y_{\ell+1}^1, y_{\ell+1}^2, \ldots, y_{|X|}^1, y_{|X|}^2\}$ from S.

Thus to get $\hat{S}$ from S we add at most as many vertices as we delete, and so it is the case that $|\hat{S}| \leq |S| \leq k$.

Consider the induced subgraphs $H = G - S$ and $\hat{H} = G - \hat{S}$ of G. Of these H has no T-triangles. Every vertex of $\hat{H}$ which is *not* present in H belongs to the set Y. So if $\hat{H}$ contains a T-triangle then each such T-triangle must contain a vertex from Y.

Assume that there exists a T-triangle $\{y, z, u\}$ in $\hat{H}$, where vertices y, z, u are in sets Y, I and K, respectively. As Y is a subset of K_1, any vertex in Y is not adjacent with a vertex in I_0. This implies that z belongs to the set $I_{\tilde{S}} \cup I_1$. Now from Corollary 5 we get that $N_B(Y) = X$ and since $\hat{S}$ contains X, no vertex in X is present in $\hat{H}$. In other words, no vertex in set $N_G(y) \cap I_{\tilde{S}}$ is present in graph $\hat{H}$. This implies that z is in I_1. Now, consider the vertex u which is adjacent with z and hence can not be in set K_0. Vertex z, which is in I_1, has exactly one neighbor in K_1 (Observation 7). Since both y, u are adjacent with z and y is contained in $Y \subseteq K_1$, vertex u is contained in set $K_{\tilde{S}}$. We now argue that vertex u is adjacent with y even in graph B. This follows from the fact that while constructing graph B, edge yu is not deleted as there exists z in I_1 which is adjacent with both y and u. Hence vertex u is contained in set X. By definition we have that no vertex in X is present in $\hat{H}$. Thus our assumption is wrong and no vertex in Y is in any T-triangle in $\hat{H}$. Hence there are no T-triangles in $\hat{H}$, and $\hat{S} = (S \setminus Y) \cup X$ is a T-THS of G' of size at most k.

Reduction Rule 10. *If $|K_1| \geq 2|\tilde{S}|$ then find sets $X \subseteq \tilde{S}$ and $Y \subseteq K_1$ as described by Corollary 5. Set $G' \leftarrow G - X, T' \leftarrow T \setminus X, k' \leftarrow k - |X|$. The reduced instance is $(G'; T'; k')$.*

Lemma 19. *Reduction Rule 10 is safe.*

Proof. If $(G; T; k)$ is a YES instance then we get from Lemma 18 that G has a T-THS S of size at most k which contains all of X. By applying part (1) of Observation 3 $|X|$ times we get that $G' = G - X, T' = T \setminus X, k' = k - |X|$ is a YES instance.

Now suppose $(G'; T'; k')$ is a YES instance. Observe that we can get graph G from G' by adding, in turn, each vertex $x \in X$ and some edges incident on x, and also that every vertex that we add in this process is a terminal vertex in graph G. Hence by applying Observation 4 $|X|$ times we get that $(G; T = T' \cup X; k = k' + |X|)$ is a YES instance.

Proof. (of Theorem 3) Consider an algorithm which apply reduction rules described in this section on a given instance $(G; T; k)$ of SUBSET FVS IN SPLIT GRAPHS. The algorithm applies least indexed reduction rule exhaustively before moving on to next reduction rule. Let $(G'; T'; k')$ be the instance obtained by applying all the reduction rules exhaustively. If $|V(G')|$ and $|E(G')|$ is at most $\mathcal{O}(k^2)$ and clique partition of G' contains at most $10k$ vertices then the algorithm outputs $(G'; T'; k)$ otherwise it outputs I_{NO}.

Let $(G'; T'; k)$ be output of the algorithm on input $(G; T; k)$. First part of the proof i.e. $(G; T; k)$ is a YES instance if and only if $(G'; T'; k)$ is a YES instance follows from the proof of safeness of each reduction rule. We now the argue size bound mentioned in second part of the Theorem. If $(G'; T'; k)$ is I_{YES} or I_{NO} then this size bounds are vacuously true. Hence we assume that no reduction rule returned trivially YES or NO instances. Since Reduction Rule 2 is not applicable, there are no isolated vertices graph. Let (K', I') be the split partition of split graph G'. Reduction Rules 6 and 7 are not applicable, every vertex in K' adjacent with at most k vertices in I'. Since there are no isolated vertices in graph, this implies $|I'| \leq k \cdot |K'|$. We are under the assumption that Reduction Rule 8 did not return I_{NO} and hence approximate solution, $\tilde{S}$, is of size at most $3k$. By Observation 7, $|K' \cap \tilde{S}| = |K'_{\tilde{S}}| \leq 2k$ and $|I' \cap \tilde{S}| = |I'_{\tilde{S}}| \leq k$. Let K'_0, K'_1 be the partition as defined after Lemma 15. Since Reduction Rule 9 is not applicable, this implies $|K_0| < 2|I'_{\tilde{S}}| < 2k$. Similarly, since Reduction Rule 10 is not applicable implies that $|K_1| < 2|\tilde{S}| < 6k$. Since $K'_{\tilde{S}}, K_0$ and K_1 is tri-partition of K, we can conclude that the cardinality of K is upper bounded by $10k$.

We conclude this proof by stating that the running time of entire algorithm is polynomial in size of input. Every Reduction rule either finds some structure in graph or apply Lemma 3, 4 or 5. Either of these process can be done in polynomial time. In case of Reduction Rule 8, algorithm finds an approximate solution in polynomial time as mentioned above the statement.

4 Kernel Lower bound

Let $\Pi \subseteq \Sigma^*$ be any language. A *polynomial compression* for a parameterized problem $Q \subseteq \Sigma^* \times \mathbb{N}$ is an algorithm $\mathcal{C}$ that, given an instance $(I; k)$ of Q, works in time $\mathcal{O}((|I| + k)^c)$ and returns a string y such that (1) $|y| \leq p(k)$ for some polynomial $p(\cdot)$, and (2) $y \in \Pi$ if and only if $(I; k) \in Q$. Here, c is a constant. If $|\Sigma| = 2$, the polynomial $p(\cdot)$ will be called the bitsize of the compression. Dell and van Melkebeek [7] established following breakthorugh result.

Proposition 1. *For any $\epsilon > 0$, the* VERTEX COVER *problem parameterized by the solution size does not admit a polynomial compression with bitsize $\mathcal{O}(k^{2-\epsilon})$, unless* NP $\subseteq$ coNP$/poly$.

We prove similar result for SUBSET FVS IN SPLIT GRAPHS.

Theorem 4. *For any $\epsilon > 0$, the* SUBSET FVS IN SPLIT GRAPHS *problem parameterized by the solution size does not admit a polynomial compression with bitsize $\mathcal{O}(k^{2-\epsilon})$, unless* $\mathsf{NP} \subseteq \mathsf{coNP}/poly$.

Proof. We prove the above statement using method of contradiction. Fix a language $\Pi \subseteq \Sigma^*$. We assume that there exists a algorithm $\mathcal{C}_1$ and a constant $\delta > 0$ such that given any instance $(I;k)$ of SUBSET FVS IN SPLIT GRAPHS as input, algorithm $\mathcal{C}_1$ outputs string $y \in \Sigma$ of size $\mathcal{O}(k^{2-\delta})$ such that such that $(I;k)$ is a YES instance of SUBSET FVS IN SPLIT GRAPHS if and only if y is a YES instance of Π. Using this assumption, we prove polynomial compression for VERTEX COVER which contradicts Proposition 1.

Fomin et al. proved that decision version of SUBSET FVS IN SPLIT GRAPHS is NP-complete [10, Theorem 2.1] by presenting a reduction from VERTEX COVER problem. We present the reduction they used in the theorem.

Reduction: Let (G,k) be an instance of VERTEX COVER, where G is an arbitrary graph with n vertices and m edges. We construct a split graph H with split partition (K,I) as follows. $V(H) = K \dot\cup I$ contains $n+m$ vertices: for each vertex $u \in V(G)$, there is a vertex $u \in K$, and for each edge $\{v,w\} \in E(G)$, there is a vertex $u_{vw} \in I$. $E(H)$ is defined so that vertices in K are pairwise adjacent, and each vertex u_{vw} of I has exactly two neighbors: vertices v and w in K. Consequently, K is a clique and I is an independent set.

In [10, Theorem 2.1], authors proved that $(G;k)$ is a YES instance of VERTEX COVER if and only if $(H;k)$ is a YES instance of SUBSET FVS IN SPLIT GRAPHS.

We can construct a compression algorithm $\mathcal{C}_2$ for VERTEX COVER, which uses the above reduction to covert an instance $(G;k)$ of VERTEX COVER to an instance $(H;k)$ of SUBSET FVS IN SPLIT GRAPHS. On this instance, $\mathcal{C}_2$ runs an compression algorithm $\mathcal{C}_1$ to produce a string y. By the property of compression algorithm, $(H;k)$ is a YES instance of SUBSET FVS IN SPLIT GRAPHS if and only if y is a yes instance of Π. By [10, Theorem 2.1], $(H;k)$ is a YES instance of SUBSET FVS IN SPLIT GRAPHS if and only if $(G;k)$ is a yes instance of Π. Hence $(G;k)$ is a YES instance of VERTEX COVER if and only if y is a YES instance of Σ^* and size of y is at most $\mathcal{O}(k^{2-\delta})$. Since this entire process can be completed in polynimal in size of input $(G;k)$, this contradicts Proposition 1.

5 An FPT Algorithm For SUBSET FVS IN CHORDAL GRAPHS

In this section we prove Theorem 2; we show that the SUBSET FVS IN CHORDAL GRAPHS problem can be solved in $\mathcal{O}(2^k(n+m))$ time where n,m are the number of vertices and edges of G, respectively.

We first give an informal description of the algorithm. At a high level, the algorithm proceeds by branching on the vertices of a clique in the input graph. We use properties of chordal graphs to ensure that we can always find a clique to branch on, and that this can be done with branching vectors[8] which keep the running time within $\mathcal{O}^{\star}(2^k)$.

We apply reduction rules to ensure that every simplicial clique in the graph has size at least three, and that every vertex in the graph is part of a T-triangle. The latter condition implies that every simplicial clique contains at least one terminal vertex. Let $Q = \{s, a, b\}$ be a simplicial clique in the graph, where s is a simplicial vertex. The algorithm branches by picking either of the two vertices a, b into the solution. This is safe[9] since s, a, b is the only T-triangle containing s. The branching vector is $(1, 1)$ and resolves to a running time of $\mathcal{O}^{\star}(2^k)$. This branching rule is applied whenever there is a simplicial clique of size three in the graph.

Let $Q = \{t, a, b, c, d\} \subseteq V(G)$ be a—not necessarily simplicial—clique of size five in G which contains a terminal t. If a solution does not contain t, then it must contain at least three of the vertices $\{a, b, c, d\}$. This is because each pair of these four vertices forms a T-triangle with t. This implies that the three-way branch $\{\{t\}, \{a, b\}, \{c, d\}\}$ is correct and complete.[10] This branching has the branching vector $(1, 2, 2)$ which resolves to a running time of $\mathcal{O}^{\star}(2^k)$. Note that if clique Q contains more than five vertices (including a terminal t) then we can apply this branching to a sub-clique $\{t, a, b, c, d\} \subsetneq Q$. If there is a simplicial clique of size *at least* five in the graph then the algorithm applies this branching.

We now come to the case where every simplicial clique in the graph has size exactly four. Simple branching rules now fail to give a running time within the $\mathcal{O}^{\star}(2^k)$ bound. We argue if the graph has more than eight vertices at this point then we can find three simplicial cliques with certain properties. We design a branching rule which applies to this structure and gives a total running time of $\mathcal{O}^{\star}(2^k)$.

5.1 Reduction Rules

The first two rules apply at the leaves of the branching tree.

Reduction Rule 11. *Let $(G; T; k)$ be an instance of* SUBSET FVS IN CHORDAL GRAPHS*. If $k \leq 0$ and there is a T-triangle in G then return a trivial NO instance.*

Reduction Rule 12. *Let $(G; T; k)$ be an instance of* SUBSET FVS IN CHORDAL GRAPHS*. If $k \geq 0$ and there is* no *T-triangle in G then return a trivial YES instance.*

We can get rid of connected components which are cliques.

Reduction Rule 13. *Let $(G = (V, E); T; k)$ be an instance of* SUBSET FVS IN CHORDAL GRAPHS*, and let Q be a connected component of G which is a clique. Delete Q from G to obtain the graph $G' = G - Q$, and let $T' = T \setminus Q$. The reduced instance is computed as follows:*

[8] See [12] for definition of Branching Vectors.

[9] Precise arguments follow.

[10] For a given instance (G, T, k) each branch create new (smaller) instance and solves the problem recursively. In this case, three branches would be $(G - \{t\}, T \setminus \{t\}, k - 1)$, $(G - \{a, b\}, T \setminus \{a, b\}, k - 2)$ and $(G - \{a, b\}, T \setminus \{c, d\}, k - 2)$.

1. *If $|Q| \in \{1,2\}$,* or *if Q contains no terminal, then the reduced instance is $(G';T';k)$.*
2. *Else: If Q contains at most two non-terminals, then decrement k by $|Q|-2$: the reduced instance is $(G';T';k-(|Q|-2))$.*
3. *In the remaining case Q contains three or more non-terminals. Let $t=|T\cap Q|$ be the number of* terminals *in Q. Decrement k by t: the reduced instance is $(G';T';k-t)$.*

Lemma 20. *Reduction Rule 13 is safe.*

Proof. Since G' is induced subgraph of G, any subset-FVS of G is also a subset-FVS of G'. This implies forward direction in Case 1. For reverse direction in the same case, notice that no vertices in Q is part of any T-triangle in G and vertices in clique Q are disjoint from vertices of G'. Hence if S is subset-FVS of G' then is also a subset-FVS of G.

Consider clique Q and an optimal subset-FVS S in graph G. In Case 2, clique Q contains at most two non-terminal vertices. If cardinality of $Q\setminus S$ is three or more then vertices in $Q\setminus S$ forms a T-triangle which contradicts the fact that S is a subset-FVS. Hence $Q\setminus S$ contains at most two vertices. By deleting any $|Q|-2$ vertices we obtains a cycle-less graph in Q and hence it is irrelevant which $|Q|-2$ vertices are deleted. This proves the forward direction in Case 2. For reverse direction, note that adding an isolated edge in graph G' does not change its subset-FVS. This observation along with the fact that for any subset X of $V(G)$ if S is a subset-FVS of $G-X$ of size $k-|X|$ then $S\cup X$ is a subset-FVS of G of size k.

In Case 3, clique Q contains at least three non-terminal vertices, say y_1,y_2,y_3. Any subset-FVS can omit exclude at most two terminals in clique Q. If S does not contains a terminal, say t, then it must include all but one vertices in Q. Let another vertex excluded by S in clique Q be x. If x is a terminal then S includes y_1,y_2,y_3. In this case, $(S\setminus\{y_1,y_2,y_3\})\cup\{t,x\}$ is a subset-FVS of strictly lesser cardinality then S. This contradicts the optimality of S. If x is not a terminal then S contains at least two vertices from set $\{y_1,y_2,y_3\}$. Without loss of generality, let those be y_1 and y_2. Again in this case, $(S\setminus\{y_1,y_2\})\cup\{t\}$ is a subset-FVS of strictly lesser cardinality then S which contradicts the optimality of S. Hence S contains all the terminal vertices in clique Q. Since including all the terminals in Q kills all T-cycles contained in it and S is optimum, S contains does not contain any non-terminal vertex in Q. Reverse direction in Case 3 is implied by correctness proof for Case 1 and the fact that for any subset X of $V(G)$ if S is a subset-FVS of $G-X$ of size $k-|X|$ then $S\cup X$ is a subset-FVS of G of size k.

Next reduction rule deletes any vertex which is not part of a T-triangle.

Reduction Rule 14. *Let $(G=(V,E);T;k)$ be an instance of* SUBSET FVS IN CHORDAL GRAPHS*, and let N be the set of all non terminal vertices in G which do not have any terminal from T as a neighbour. Delete N from G to obtain the graph $G'=G-N$. The reduced instance is $(G';T;k)$.*

Lemma 21. *Reduction Rule 14 is safe.*

Proof. Since G' is induced subgraph of G, any subset-FVS of G is also a subset-FVS of G'. This implies the correctness of forward direction. In reverse direction, consider subset-FVS S' of G'. If S' is not a subset-FVS of G then there exists a T-cycle in $G \setminus S$. Since this T-cycle is not contains in G', it must contain a vertex, say v, from set N. By Lemma 1, if there exists a T-cycle contains vertex v then there is a T-triangle containing v. This implies v has a neighbor in terminal set contradicting the fact that v is in N.

We can safely delete *some* edges which are not part of any T-triangle, to get a graph with no "tiny" maximal cliques.

Reduction Rule 15. *Let $(G = (V, E); T; k)$ be an instance of* SUBSET FVS IN CHORDAL GRAPHS*, and let $e = \{u, v\}$ be a bridge in G. Delete the edge e to get the graph $G' = (V, E \setminus \{e\})$. The reduced instance is $(G'; T; k)$.*

Lemma 22. *Reduction Rule 15 is safe.*

Proof. By Lemma 2, G' is a chordal graph. Since G' is a subgraph of G, any subset-FVS of G is also a subset-FVS of G'. This proves the safeness of forward direction of reduction rule. In graph G', vertices u, v are different connected components. If S' is a subset-FVS of G' which does not contain either of u or v then these two vertices are in different connected components of $G' - S'$. Hence adding an edge $e = uv$ in graph $G' - S'$ does not add more cycle. If S' contains either u or v then graphs $G' - S'$ and $G - S'$ are identical. Hence in either case, S' is a subset-FVS of G.

Lemma 23. *Let $(G; T; k)$ be an instance of* SUBSET FVS IN CHORDAL GRAPHS *which is reduced with respect to Reduction Rules 13 and 15. Then every maximal clique in G is of size at least three.*

Proof. Reduction Rule 13 ensures that for any maximal clique Q in the reduced graph G, some vertex in Q has a neighbour which is not in Q. Thus every maximal clique in Q has size at least two. Let $Q = \{u, v\}$ be a maximal clique of size two in G. Then there is a third vertex w in G such that either $\{u, w\}$ or $\{v, w\}$ is an edge in G. If *both* these edges are present in G then $\{u, v, w\}$ is a clique, contradicting the maximality of Q. So let $\{u, w\}$ be a non-edge in G.

Reduction Rule 13 ensures that G has at least three vertices, and hence Reduction Rule 15 ensures that G has no cut vertex[11]. G is thus 2-vertex-connected. In particular, there is a path from u to w which avoids the vertex v; let P be a shortest such path. Since G is chordal and since $\{u, v, w\}$ is a path in G, path P cannot be of length two or more. Thus P is a single edge $\{u, w\}$, a contradiction.

At this point, every vertex in the graph is part of at least one T-triangle.

Lemma 24. *Let $(G; T; k)$ be an instance of* SUBSET FVS IN CHORDAL GRAPHS *which is reduced with respect to Reduction Rules 13, 15, and 14. Then every vertex in G is part of a T-triangle.*

[11] Exercise 2.3.1 in "Graph Theory with Applications" by Bondy and Murty

Proof. Let v be a vertex in G and let t be a terminal which is adjacent to v. Let Q be a maximal clique in G which contains the edge $\{v, t\}$; such a clique exists, because the edge $\{v, t\}$ is a clique by itself. Clique Q contains at least one other vertex u, and $\{u, v, t\}$ form a T-triangle which contains v.

Consider a simplicial vertex v in reduced graph G. Either vertex v is a terminal or by Lemma 24, it is adjacent with some terminal. In either case, simplicial clique which containing v contains a terminal. Combining this fact with Lemma 23, we get following Corollary.

Corollary 6. *Let $(G; T; k)$ be an instance of* Subset FVS in Chordal Graphs *which is reduced with respect to Reduction Rules 13, 14, and 15, and let Q be a simplicial clique in G. Then Q has size at least three, and contains a terminal vertex.*

It remains to argue that these reduction rules can be applied in polynomial time. In the following Lemma we prove a stronger statement. We prove that not only each of this reduction rules can be applied in linear time, but we can apply all these rules exhaustively in linear time.

Lemma 25. *Let $(G; T; k)$ be an instance of* Subset FVS in Chordal Graphs*. We can exhaustively apply Reduction Rules 13, 14, 15 on this instance in time $\mathcal{O}(n+m)$. Here n, m are number of vertices and edges in input graph G.*

Proof. Let $\mathcal{C}$ be the set of all maximal clique of input graph G. The cardinality of set $\mathcal{C}$ is at most n and we can compute this set in time $\mathcal{O}(n + m)$ ([15, Theorem 4.17]). While constructing set $\mathcal{C}$, we can mark all cliques in C which contains terminal in it. We define *strong neighbors* of T as set of non-terminal vertices which are present in maximal clique containing a terminal and is of size at least three. Given a chordal graph G and set $\mathcal{C}$, one can mark all strong neighbors of T in time $\mathcal{O}(n + m)$. Moreover, all maximal cliques in $\mathcal{C}$ of size two are either bridges or isolated cliques. Given graph G, algorithm computes set $\mathcal{C}$ and performs following steps. Step 1 : Mark all strong neighbors of T and delete remaining vertices. Step 2 : Find and delete all bridges in graph G. Step 3 : Delete isolated cliques in graph according to Reduction Rule 13. One iteration of three steps can be performed in time $\mathcal{O}(n + m)$. Next, we argue that only one iteration is sufficient to get a non-reducible instance.

Consider a graph G' obtained by above process. We first argue that every non-terminal vertex in G' is strong neighbor (and hence neighbor) of T. Suppose not. Consider a non-terminal vertex v in G' which is not a strong neighbor of T. Since vertex v was not deleted in Step 1, v was a strong vertex in G. This implies, v is part of maximal clique Q of size at least 3. Since we only delete bridge edges in Step 2, no edge in Q has been deleted. As v is not deleted in Step 3, v is not a part of isolated clique. This implies Q is not an isolated clique and hence no vertex in Q has been deleted in Step 3. Hence Q is present in graph G'. Since Q contains a terminal, vertex v and is of size at least there, this contradicts the fact that v is not a strong neighborhood of T. All bridges in graph G has been deleted in Step 2. Step 3 of the algorithm deletes an isolated clique which satisfies certain criteria. Hence if there is a bridge in G' then the same bridge is present in G at the end of Step 2. Hence G' contains no bridge. Graph G' also does not contain any isolated cliques as all such cliques were deleted in Step 3.

5.2 Simple Branching Rules

Our first branching rule ensures that every non-terminal vertex has at least two terminals in its neighbourhood. So let v be a non-terminal vertex which has exactly one terminal t in its neighbourhood. Let x be a vertex—as guaranteed to exist by Lemma 24—such that $\{v, t, x\}$ is a T-triangle containing v. By our assumption *every* T-triangle which contains v also contains t. A solution S as called as *target solution* if $|S| \leq k$. It follows that if S is a target solution then $(S \setminus \{v\}) \cup \{t\}$ is also a target solution. This means that we may assume that if there exists a target solution then there exists one which does not contain v. Hence it is safe to branch on any two neighbours of v which form a T-triangle containing v.

Branching Rule 1. *Let $(G; T; k)$ be an instance of* Subset FVS in Chordal Graphs*, let v be a non-terminal vertex which has exactly one terminal neighbour t, and let $\{v, t, x\}$ be a T-triangle containing v. Let $G_1 = G - \{t\}, G_2 = G - \{x\}$ and $T_1 = T \setminus \{t\}, T_2 = T \setminus \{x\}$. The new instances are: $(G_1; T_1; k-1), (G_2; T_2; k-1)$.*

Lemma 26. *Branching Rule 1 is exhaustive and it can be executed in time $\mathcal{O}(n+m)$.*

Proof. Consider an instance $(G; T; k)$ of Subset FVS in Chordal Graphs and let $(G_1; T_1; k-1)$ and $(G_2; T_2; k-1)$ be two instances produced by Branching Rule 1 when applied on $(G; T; k)$. We argue that $(G; T; k)$ is a YES instance if and only if either $(G_1; T_1; k-1)$ or $(G_2; T_2; k-1)$ is a YES instance. Let v, t, x be vertices in G as specified in the statement of branching rule.

($\Rightarrow$) Since $\{v, t, x\}$ is a T-triangle in G, any target solution of $(G; T; k)$ contains at least one vertex among them. Consider a target solution S which contains t. Set $S \setminus \{t\}$ is a solution of $(G - \{t\}; T_1; k-1)$. By applying similar argument in the case when S contains x implies that $(G - \{x\}; T_2; k-1)$ is an YES instance. We now consider the case when S contains v. Note that any T-cycle passing through v contains terminal t. Hence if S is a target solution contains v then $(S \setminus \{v\}) \cup \{t\}$ is also a target solution. This implies if $(G; T; k)$ is a YES instance then either $(G_1; T_1; k-1)$ or $(G_2; T_2; k-1)$ is also a YES instance.

($\Leftarrow$) The proof of reverse direction follows from the fact that for any $U \subseteq V(G)$ if S' is a target solution of $(G - U; T \setminus U; k - |U|)$ then $S' \cup U$ is a target solution of $(G; T; k)$.

To apply this reduction rule, we need to find a vertex which is adjacent with exactly one terminal. If such vertex exists then it can be found in time $\mathcal{O}(n+m)$.

The rest of our branching rules apply to simplicial cliques. As noted above, this is more or less straightforward when we have a simplicial clique of size either three, or at least five. Let v be a simplicial vertex in graph G and $\{v, a, b\}$ its simplicial clique. By Corollary 6 at least one of $\{v, a, b\}$ is a terminal, and so every solution must contain at least one of $\{v, a, b\}$. Vertex v is not part of any triangle in the graphs $G - \{a\}$ and $G - \{b\}$. It follows that if a solution S contains v then both $(S \setminus \{v\}) \cup \{a\}$ and $(S \setminus \{v\}) \cup \{b\}$ are solutions of the same size or smaller. Thus we get that there is an optimal solution which does not contain v, and contains at least one of $\{a, b\}$. This gives us a two-way branching rule for such cliques.

Branching Rule 2. *Let $(G; T; k)$ be an instance of* Subset FVS in Chordal Graphs*, and let $\{v, a, b\}$ be a simplicial clique with v being the simplicial vertex. Let $G_1 = G - \{v, a\}, G_2 = G - \{v, b\}$ and $T_1 = T \setminus \{v, a\}, T_2 = T \setminus \{v, b\}$. The new instances are: $(G_1; T_1; k - 1), (G_2; T_2; k - 1)$.*

Lemma 27. *Branching Rule 2 is exhaustive and it can be executed in time $\mathcal{O}(n+m)$.*

Proof. Consider an instance $(G; T; k)$ of Subset FVS in Chordal Graphs and let $(G_1; T_1; k - 1)$ and $(G_2; T_2; k - 1)$ be two instance produced by Branching Rule 2 when applied on $(G; T; k)$. Let v be a simplicial vertex of degree two and a, b be its neighbors.

($\Rightarrow$) By Corollary 6 at least one of $\{v, a, b\}$ is a terminal. Since $\{v, a, b\}$ is a T-triangle in G, any target solution of $(G; T; k)$ contains at least one vertex among them. Consider a target solution S which contains a. Set $S \setminus \{a\}$ is a solution of $(G - \{a\}; T_1; k - 1)$. In graph $G - \{a\}$, vertex v is adjacent with only b. Reduction Rules 15, 13 applied to $G - a$ will delete the edge incident of v and then vertex v. Hence $(G - \{a\}; T_1; k - 1)$ is a YES instance if and only if $(G - \{a, v\}; T_1; k - 1)$ is a YES instance. By applying similar argument in the case when S contains b implies that if a target solution S contains b then $(G - \{b, v\}; T_1; k - 1)$ is a YES instance. We now consider the case when S contains v. We claim that we can construct another solution which excludes v and is of cardinality at most S. Since v is part of exactly one T-triangles, namely, $\{v, a, b\}$, every T-triangle passing through v can be hit by picking either a or b. Hence, if a target solution S contains v then $(S \setminus \{v\}) \cup \{a\}$ is also a target solution. This implies if $(G; T; k)$ is a YES instance then either $(G_1; T_1; k - 1)$ or $(G_2; T_2; k - 1)$ is also a YES instance.

($\Leftarrow$) If S' is a target solution of $(G \setminus \{v, a\}; T \setminus \{v, a\}; k - 1)$ then $S' \cup \{a\}$ is a solution of $G - v$ of size at most k. Since vertex v is adjacent with only b in graph $G - (S' \cup \{a\})$, set $S' \cup \{a\}$ is a solution of G as well. This implies that $(G; T; k)$ is a YES instance. By applying similar argument in case of $(G \setminus \{v, b\}; T \setminus \{v, b\}; k - 1)$, we complete the proof of the reverse direction.

To apply this reduction rule, algorithm needs to find simplicial vertex of degree two if one exists. It is easy to check all vertices of degree two whether or not their neighborhood is a clique. Hence one can find a simplicial clique to branch on or conclude that no such clique exits in time $\mathcal{O}(n + m)$.

Now let Q be a clique in G of size at least five, and let v be its simplicial vertex. By Corollary 6 Q contains a terminal, say t. We can thus—as argued at the beginning of this section—safely branch in the following manner.

Branching Rule 3. *Let $(G; T; k)$ be an instance of* Subset FVS in Chordal Graphs*, Q be a clique of size at least five in G, and t a terminal in Q. Let $\{a, b, c, d\}$*

be four other vertices of Q. Let $G_1 = G - \{t\}, G_2 = G - \{a, b\}, G_3 = G - \{c, d\}$ and $T_1 = T \setminus \{t\}, T_3 = T \setminus \{a, b\}, T_3 = T \setminus \{c, d\}$. The new instances are: $(G_1; T_1; k-1), (G_2; T_2; k-2), (G_3; T_3; k-2)$.

Lemma 28. *Branching Rule 3 is exhaustive and it can be executed in time $\mathcal{O}(n+m)$.*

Proof. Consider an instance $(G; T; k)$ of Subset FVS in Chordal Graphs and let $(G_1; T_1; k-1)$, $(G_2; T_2; k-2)$, and $(G_2; T_2; k-2)$ be three instances produced by Branching Rule 3 when applied on $(G; T; k)$. Let t be a terminal in maximal clique of size at least five and vertices a, b, c, d be any of its four neighbors.

($\Rightarrow$) Consider a target solution S for (G, T, k). We consider following cases: S includes the terminal t; S excludes t but contains *both* a, b and third case is when S excludes t and dose not contains *both* a, b. Since $\{t, a, b\}$ is a T-triangle in G, solution S can not exclude all of a, b, t. Hence in third case, S includes at least one of a, b. If target solution S contains t then set $S \setminus \{t\}$ is a solution of $(G - \{t\}, T_1, k-1)$. Similarly, if target solution S contains a, b then set $S \setminus \{a, b\}$ is a solution of $(G - \{a, b\}, T_2, k-2)$. Consider third case mentioned above. Without loss of generality, assume that S excludes b. This implies S excludes both t and b. Since both $\{t, b, c\}$ and $\{t, b, d\}$ are T-triangles, S must include both c, d. By applying similar argument as in previous case, set $S \setminus \{c, d\}$ is a solution of $(G - \{c, d\}, T_2, k-2)$. This implies if (G, T, k) is a YES instance then either $(G_1; T_1; k-1)$ or $(G_2; T_2; k-2)$ or $(G_3; T_3; k-2)$ is also a YES instance.

($\Leftarrow$) The proof of reverse direction follows from the fact that for any $U \subseteq V(G)$ if S' is a target solution of $(G - U; T \setminus U; k - |U|)$ then $S' \cup U$ is a target solution of $(G; T; k)$.

To apply the branching rule, we need to find a maximal clique of size at least five which contains a terminal if one exists. One can enumerate all maximal cliques in given chordal graph in time $\mathcal{O}(n+m)$ and hence can find desired clique or conclude that no such clique exists.

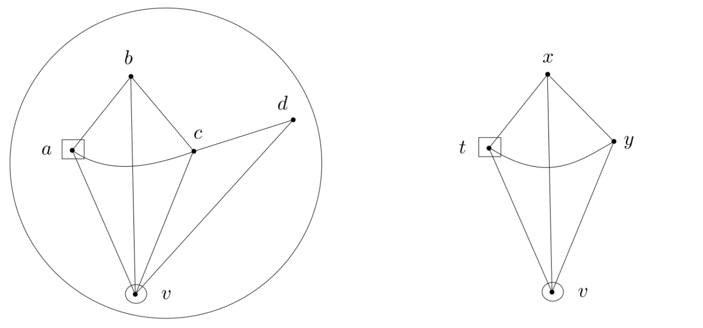
Observation 8. *Let $(G; T; k)$ be an instance which is reduced with respect to all the reduction rules in this section, and to which neither of the above branching rules applies. Then every simplicial clique in G is of size exactly four.*

We now show that we can ensure, within the $\mathcal{O}^\star(2^k)$ time bound, that every simplicial vertex in G is a terminal as well.[12] For this we need a structural result.

Lemma 29. *Let $(G; T; k)$ be an instance of* Subset FVS in Chordal Graphs, *and let S be a* minimal *solution of $(G; T; k)$. Let v be a* non-terminal *vertex in S for which there exists a clique Q in G such that every T-triangle which contains v is also contained in Q. Then there is a* unique *T-triangle $\triangle_v$ such that v is the only vertex in S which is present in $\triangle_v$.*

Proof. If every T-triangle in which v is present also contains another vertex from S then $S \setminus \{v\}$ is a solution as well, contradicting the minimality of S. So there is

[12] The converse may not hold.



(a) Refer to Lemma 29 (b) Refer to Lemma 30

Fig. 2. In both figures, square around the vertex denotes that it is a terminal while circle denotes that the vertex is included in solution S. In 2a, outer circle represents the maximal clique Q. Not all the vertices and edges in Q are shown in the figure.

at least one T-triangle $\triangle_v$ such that v is the only vertex in S which is present in $\triangle_v$.

Now suppose there are two T-triangles $\triangle_1 = \{v, a, b\}, \triangle_1 = \{v, c, d\}$ such that v is the only vertex in S which is present in each of $\triangle_1, \triangle_2$ (See Figure 2a). Now triangles $\triangle_1, \triangle_2$ are distinct, each contains a terminal vertex, and v is not a terminal vertex. So the collection (v, a, b, c, d) contains at least four distinct vertices, say $\{v, a, b, c\}$, of which at least one vertex, say a, is a terminal vertex. Since both these triangles are part of the clique Q, the four vertices $\{v, a, b, c\}$ form a clique. Deleting v from this clique leaves the triangle $\{a, b, c\}$ which (i) contains the terminal a, and (ii) contains no vertex from the set S. Thus the purported solution S does not intersect the T-triangle $\{a, b, c\}$, a contradiction.

Since every neighbour of a simplicial vertex is contained in its simplicial clique, we get

Corollary 7. *Let $(G; T; k)$ be an instance of* Subset FVS in Chordal Graphs, *and let S be a* minimal *solution of $(G; T; k)$. Let v be a* non-terminal, simplicial *vertex in S. Then there is a* unique *T-triangle $\triangle_v$ such that v is the only vertex in S which is present in $\triangle_v$.*

This implies that we can safely get rid of *non-terminal, simplicial* vertices.

Lemma 30. *Let $(G; T; k)$ be an instance of* Subset FVS in Chordal Graphs *which is reduced with respect to the reduction and branching rules described so far. Let v be non-terminal vertex which is simplicial in G. If $(G; T; k)$ has a target solution then it has a target solution which does not contain v.*

Proof. By Observation 8, every simplicial clique in G has size exactly four. Let S be a target solution of $(G; T; k)$ which contains v, and let $Q = \{v, t, x, y\}$ be the simplicial clique of v where t is a terminal vertex.

From Corollary 7 we get that there is a unique T-triangle $\triangle_v$ such that v is the only vertex from S which is in $\triangle_v$. Without loss of generality let $\triangle_v$ be $\{v, t, x\}$ (See Figure 2b). Then neither t nor x is in S. Now $\{t, x, y\}$ is a T-triangle as well, so we get that vertex y is in S. If we delete the vertex set $S \setminus \{v\}$ from G, then every remaining T-triangle must intersect v. But the only such triangle is $\triangle_v = \{v, t, x\}$, and so we get that $(S \setminus \{v\}) \cup \{t\}$ is a solution as well.

Let v be a non-terminal vertex which is simplicial, and let $Q = \{v, t, x, y\}$ be the simplicial clique of v where t is a terminal vertex. If $(G; T; k)$ is a YES instance then we get from Lemma 30 that it has a target solution which (i) does not contain v, and (ii) contains either t or both of x, y. This implies that we can safely delete v and branch on $t, \{x, y\}$.

Branching Rule 4. *Let $(G; T; k)$ be an instance of* Subset FVS in Chordal Graphs*, and let $\{v, t, x, y\}$ be a simplicial clique whose simplicial vertex v is a* non-terminal*, and t is a terminal. Let $G_1 = G - \{t\}, G_2 = G - \{v, x, y\}$ and $T_1 = T \setminus \{t\}, T_2 = T \setminus \{x, y\}$. The new instances are: $(G_1; T_1; k-1), (G_2; T_2; k-2)$.*

Lemma 31. *Branching Rule 4 is exhaustive and it can be executed in time $\mathcal{O}(n+m)$.*

Proof. Consider an instance $(G; T; k)$ of Subset FVS in Chordal Graphs and let $(G_1; T_1; k-1)$ and $(G_2; T_2; k-2)$ be two instances produced by Branching Rule 4 when applied on $(G; T; k)$. Let v be a non-terminal simplicial vertex with neighbors t, x, y, where t is a terminal.

$(\Rightarrow)$ Since $\{v, t, x, y\}$ is a clique in G, any target solution of (G, T, k) contains at least one vertex among them. By Lemma 30, there exists a target solution which does not contain v and contains either t or both of x, y. If target solution S contains t then set $S \setminus \{t\}$ is a solution of $(G - \{t\}; T_1; k-1)$. If solution S excludes t then it includes both x, y. In this case, $S \setminus \{x, y\}$ is a solution of $(G - \{x, y\}, T_2, k-2)$. Notice that in graph $G - \{x, y\}$, vertex v is adjacent with exactly one vertex. Reduction Rules 15,13 applied to $G - \{x, y\}$ will delete the edge incident on v and then vertex v. Hence $(G - \{v, x, y\}; T_1; k-1)$ is a YES instance if and only if $(G_2; T_2; k-2)$ is a YES instance. This implies if (G, T, k) is a YES instance then either $(G_1; T_1; k-1)$ or $(G_2; T_2; k-2)$ is also a YES instance.

$(\Leftarrow)$ We use the fact that for all $U \subseteq V(G)$ if S' is a target solution of $(G - U; T \setminus U; k - |U|)$ then $S' \cup U$ is a target solution of $(G; T; k)$. In case $(G_1; T_1; k-1)$ is a YES instance, reverse direction follows immediately. If $(G_2; T_2; k-2)$ is a YES instance then we first note that adding a new vertex of degree one in G_2 does not change its subset-FVS. This completes the proof of the reverse direction of the lemma.

To apply this reduction rule, algorithm needs to find simplicial vertex of degree four if one exists. It is easy to check all vertices of degree four whether or not their neighborhood is a clique. Hence one can find a simplicial clique to branch on or conclude that no such clique exits in time $\mathcal{O}(n+m)$.

At this point we have that every simplicial vertex is a terminal. We now ensure that there is exactly one terminal in any simplicial clique. We show first that if t is a simplicial terminal vertex with exactly three neighbours x, y, z, then we can safely assume that a target solution which contains t does not contain any of $\{x, y, z\}$, and vice versa.

Lemma 32. *Let $(G; T; k)$ be an instance of* Subset FVS in Chordal Graphs. *Let t be a terminal vertex which is simplicial, and let $C = \{t, x, y, z\}$ be its simplicial clique. If $(G; T; k)$ has a target solution which contains two vertices from the clique C, then it has a target solution which does not contain t.*

Proof. Let S be a target solution which contains two vertices from C. If S does not contain t then there is nothing more to prove. So let $t \in S$. We assume, without loss of generality, that x is another vertex from C which is in S, so that $\{t, x\} \subseteq S$. Let H be the graph obtained from G by deleting all of S. Then H contains no T-triangle.

Now consider the set $S' = (S \setminus \{t\}) \cup \{y\}$, which is not larger than S. Let H' be the graph obtained by deleting S' from G. Thus H' is the graph obtained from the graph H by (i) adding back t and all the edges $\{\{t, s\} \in E(G) \ ; \ s \in V(H)\}$, and (ii) deleting y from the resulting graph. Since t is the only vertex *added* to H in this process, we get that every T-triangle in H' must contain vertex t.

Now since S' contains both of $\{x, y\}$, we get that vertex t has degree at most one in graph H'. Thus t it is not part of *any* triangle in H', which implies that H' contains no T-triangle. Thus S' is a target solution which does not contain t.

Corollary 8. *Let $(G; T; k)$ be an instance of* Subset FVS in Chordal Graphs. *Let t be a terminal vertex which is simplicial, and let $C = \{t, x, y, z\}$ be its simplicial clique. Then the following are all equivalent:*

1. *$(G; T; k)$ has a target solution.*
2. *$(G; T; k)$ has a target solution which*
   - *does not contain t, OR*
   - *contains t, and excludes all of $\{x, y, z\}$.*
3. *$(G; T; k)$ has a target solution which, for each $w \in \{x, y, z\}$,*
   - *does not contain w, OR*
   - *contains w, and excludes t.*

Proof. We show that (**1**) and (**2**) are equivalent, and that (**1**) and (**3**) are equivalent. Observe that the directions (**2**) $\implies$ (**1**) and (**3**) $\implies$ (**1**) are trivially true; we now argue that the forward directions hold in each case.

(**1**) $\implies$ (**2**) . If $(G; T; k)$ has a target solution which does not contain t then there is nothing more to prove. So let it be the case that *every* such solution contains t. If there is such a solution which excludes all of $\{x, y, z\}$ then there is nothing more to prove. So let it be the case that every such solution contains at least one of $\{x, y, z\}$, as well. Thus every target solution contains two vertices from clique C. Now using (**1**) we get that $(G; T; k)$ has a target solution which contains two vertices from clique C. Lemma 8 implies that $(G; T; k)$ has a target solution which does not contain t, which contradicts our assumption.

$(\mathbf{1}) \implies (\mathbf{3})$. We present the case for $w = x$; the other two cases follow by symmetric arguments. If $(G;T;k)$ has a target solution which does not contain x then there is nothing more to prove. So let it be the case that *every* such solution contains x. If there is such a solution which excludes t then there is nothing more to prove. So let it be the case that every such solution contains t. Thus every target solution of $(G;T;k)$ contains both of $\{t, x\}$. Now using **(1)** we get that $(G;T;k)$ has a target solution which contains two vertices—t and x—from clique C. Lemma 8 implies that $(G;T;k)$ has a target solution which does not contain t, which contradicts our assumption. □

From this we get

Corollary 9. *If $C = \{t, x, y, z\}$ is a simplicial clique of the given type then it is safe to assume the following:*

- *if there is a target solution which contains t, then there is such a solution which contains none of the vertices $\{x, y, z\}$.*
- *if there is a target solution which contains at least one of $\{x, y, z\}$, then there is such a solution which does not contain t.*

Definition 1. *Let $(G;T;k)$ be an instance of* Subset FVS in Chordal Graphs *to which none of the previous reduction or branching rules applies. Let $C = \{t, x, y, z\}$ be a simplicial clique in G with t being its unique terminal and simplicial vertex. A subset $S \subseteq V(G)$ of vertices of G is a* selective solution *of the pair $((G;T;k), C)$ if the following hold:*

- *$G - S$ has no T-triangles,*
- *$|S| \leq k$,*
- *If S contains t then S contains none of $\{x, y, z\}$, and,*
- *If S contains one of $\{x, y, z\}$ then S does not contain t.*

A selective solution is thus a target solution which satisfies the conditions of Corollary 9. From Corollary 9 we get that there exists a target solution if and only if there exists a selective solution. From now on our algorithm will search *only* for selective solutions.

Now let $\{t, x, y, z\}$ be a simplicial clique with two terminals, say t, x, where t is a simplicial vertex. If vertex t is present in a selective solution S then S does not contain any of $\{x, y, z\}$. But then $\{x, y, z\}$ forms a T-triangle, which contradicts the fact that S is a solution. Thus vertex t is not present in *any* selective solution. Now since t is a terminal vertex, every selective solution S must pick at least two vertices from $\{x, y, z\}$: if S does not contain y, z, for instance, then $\{t, y, z\}$ forms a T-triangle in the graph obtained after deleting S, a contradiction. These considerations lead to the next branching rule.

Branching Rule 5. *Let $(G;T;k)$ be an instance of* Subset FVS in Chordal Graphs*, and let $\{t, x, y, z\}$ be a simplicial clique where t and x are terminal vertices, and t is a simplicial vertex. Let $G_1 = G - \{x, y\}, G_2 = G - \{y, z\}, G_3 = G - \{x, z\}$ and $T_1 = T \setminus \{x, y\}, T_2 = T \setminus \{y, z\}, T_3 = T \setminus \{x, z\}$. The new instances are: $(G_1;T_1;k-2), (G_2;T_2;k-2), (G_3;T_3;k-2)$.*

Lemma 33. *Branching Rule 5 is exhaustive and it can be executed in time $\mathcal{O}(n+m)$.*

Proof. Consider an instance $(G; T; k)$ of Subset FVS in Chordal Graphs and let $(G_1; T_1; k-2), (G_2; T_2; k-2)$, and $(G_3; T_3; k-2)$ be three instances produced by Branching Rule 5 when applied on $(G; T; k)$. Let t be a terminal simplicial vertex with neighbors x, y, and z, where x is a terminal.

($\Rightarrow$) From Corollary 9, there exists a selective solution, say S. By definition of selective solution, if S contains t then S contains none of x, y, z. Since x is a terminal, x, y, z forms a T triangle which is not intersected by S which is a contradiction. Hence S must contain at least one of x, y, z which implies S does not contain t. But t is a terminal vertex and hence S must contain at least two vertices from x, y, z. If S contains $\{x, y\}$ then set $S \setminus \{x, y\}$ is a solution for $(G - \{x, y\}; T \setminus \{x, y\}; k-2)$. By applying similar argument when S contains $\{x, z\}$ and $\{y, z\}$, derive that if $(G; T; k)$ is a YES instance then at least one of $(G_1; T_1; k-2), (G_2; T_2; k-2)$ or $(G_3; T_3; k-2)$ is a YES instance.

($\Leftarrow$) We use the fact that for any $U \subseteq V(G)$ if S' is a target solution of $(G - U; T \setminus U; k - |U|)$ then $S' \cup U$ is a target solution of $(G; T; k)$ to prove the reverse direction.

To apply this reduction rule, algorithm needs to find terminal simplicial vertex of degree four which is adjacent with a terminal if one exists. It is easy to check all vertices of degree four whether or not their neighborhood is a clique and contains a terminal. Hence one can find a simplicial clique to branch on or conclude that no such clique exits in time $\mathcal{O}(n+m)$.

Once Branching Rule 5 has been applied exhaustively, no simplicial clique in the graph contains two or more terminal vertices. Let $\{t, x, y, z\}$ be a simplicial clique with t as its unique terminal (and simplicial) vertex, and let $t' \neq t$ be a terminal which shares two common neighbours—say, x and y—with t. Then we can branch on one of these common neighbours, say x, as follows. If x is in a selective solution S then t is not in S. At least one of $\{y, z\}$ must be in S, or else the T-triangle $\{t, y, z\}$ will remain after deleting S. If x is not picked in S then we branch on the vertex t: if t is picked in S then x, y, z are not in S. This forces the terminal t' to be in S, since otherwise the T-triangle $\{t', x, y\}$ will remain after deleting S. In the remaining case neither of x, t is picked in S, and this forces both of y, z into S. Thus we get

Branching Rule 6. *Let $(G; T; k)$ be an instance of* Subset FVS in Chordal Graphs *to which none of the previous reduction or branching rules applies, let $\{t, x, y, z\}$ be a simplicial clique of G with terminal t, and let $t' \neq t$ be a terminal which has x, y as neighbours. Let $G_1 = G - \{x, y\}, G_2 = G - \{y, z\}, G_3 = G - \{x, z\}, G_4 = G - \{t, t'\}$ and $T_1 = T_2 = T_3 = T, T_4 = T \setminus \{t, t'\}$. The new instances are: $(G_1; T_1; k-2), (G_2; T_2; k-2), (G_3; T_3; k-2), (G_4; T_4; k-2)$.*

Lemma 34. *Branching Rule 6 is exhaustive and it can be executed in time $\mathcal{O}(n+m)$ on instance which is not reducible by any of previously mentioned reduction or branching rule.*

Proof. Consider an instance $(G; T; k)$ of SUBSET FVS IN CHORDAL GRAPHS and let $(G_1; T_1; k-2), (G_2; T_2; k-2), (G_3; T_3; k-2)$, and $(G_4; T_4; k-2)$ be four instances produced by Branching Rule 6 when applied on $(G; T; k)$. Let t be a terminal simplicial vertex with neighbors $x, y,$ and z. Let t' be another terminal which is adjacent with both x and y.

($\Rightarrow$) From Corollary 9, there exists a selective solution, say S. Since $\{t, x, y, z\}$ is a clique containing terminal, S contains at least one vertex of this clique. By definition of selective solution, if S contains t then S contains none of x, y, z. Since $\{t', x, y\}$ is a T-cycle, if S excludes both $\{x, y\}$ then it must include t' to hit this cycle. This implies any selective solution which contains terminal t also contains terminal t'. Consider the case when selective solution does not contain t. Since t is a terminal, S must contain at least two vertices from $\{x, y, z\}$. If S contains $\{x, y\}$ then set $S \setminus \{x, y\}$ is a solution for $(G - \{x, y\}, T \setminus \{x, y\}, k-2)$. By applying similar argument when S contains (x, z), (y, z), and (t, t'), we derive that if $(G; T; k)$ is a YES instance then at least one of $(G_1; T_1; k-2), (G_2; T_2; k-2), (G_3; T_3; k-2)$, or $(G_4; T_4; k-2)$ is a YES instance.

($\Leftarrow$) We use the fact that for any $U \subseteq V(G)$ if S' is a target solution of $(G - U; T \setminus U; k - |U|)$ then $S' \cup U$ is a target solution of $(G; T; k)$ to prove the reverse direction.

To apply this reduction rule, we run following pre-processing: For a given graph G, construct an array of size m. Each entry in the array corresponds to an edge and can store two terminals. Given a chordal graph G, compute set of all maximal cliques in graph. For every maximal clique Q, if it contains terminal t' then for every edge in Q which is not incident on t', add terminal t' to the entry in an array corresponding to that edge. If there are already two terminals in an array corresponding to some particular edge then we do not add new terminals. Since the input instance $(G; T; k)$ is not reducible by Branching Rule 3, every maximal clique which contains a terminal is of size at most four. Hence algorithm spend constant amount of time at each maximal clique which contains a terminal. Since there are at most n all maximal cliques all of which can be computed in $\mathcal{O}(n+m)$ time, the overall time required for this pre-processing is $\mathcal{O}(n+m)$. Notice that if terminal t' is adjacent with x, y and xy is an edge then t', x, y are contained in a maximal clique containing terminal t'. Hence for every edge whose end-points are adjacent with at least two terminals, two terminals will be stored in the array. Once this pre-process is complete, algorithm check for every terminal of degree four whether or not it is simplicial. Suppose terminal t is terminal and is adjacent with x, y, z. For every edge in set $\{xy, yz, zx\}$, algorithm checks whether there exists another terminal t' which is adjacent with end-point of the edge using the array constructed in the pre-processing step. This step can be completed in constant time. At most one of terminals stored for xy can be t and hence at least one another terminal, if exists, will be stored in the entry corresponding to edge xy. Hence one can find a simplicial clique to branch on or conclude that no such clique exits in time $\mathcal{O}(n+m)$.

At this point we have

Observation 9. *Let $(G; T; k)$ be an instance which is reduced with respect to reduction rules 11, 12, 13, 14, and 15, and branching rules 1, 2, 3, 4, 5, and 6. Then G has the following properties:*

1. *Every maximal clique is of size at least three.*
2. *Every non-terminal vertex has at least two terminals as neighbours.*
3. *Every simplicial vertex is a terminal.*
4. *Every simplicial clique C has size exactly four and contains exactly one simplicial vertex, which is also the only terminal vertex in C.*
5. *Each pair of non-terminals in a simplicial clique C has exactly one common neighbour (namely, the simplicial vertex in C) from among the terminal vertices.*

5.3 Dealing With Simplicial Cliques of Size Four

We now derive some structural properties of a "reduced" instance $(G; T; k)$ of SUBSET FVS IN CHORDAL GRAPHS as described in Observation 9, and use these to handle simplicial cliques of size exactly four within the required time bound of $\mathcal{O}^\star(2^k)$. Let $\mathcal{T}_G$ be a clique tree of graph G. Then from Fact 2 we get that every leaf of $\mathcal{T}_G$ is a simplicial clique of size exactly four. If $\mathcal{T}_G$ contains at most two nodes then both of them are leaves, and so they are simplicial cliques of size four each. In this case graph G has at most eight vertices and we can solve this instance in constant time. So we assume, without loss of generality, that $\mathcal{T}_G$ has at least three nodes. Then $\mathcal{T}_G$ has at least one node which is not a leaf; let C_r be such a node. We root the tree $\mathcal{T}_G$ at node C_r.

From now on we assume that $C_\ell = \{t, x, y, z\}$ is a leaf node of $\mathcal{T}_G$ which is at the maximum distance from the root C_r with t being the unique terminal in C_ℓ, and that C_p is the unique neighbour ("parent") of C_ℓ in $\mathcal{T}_G$.

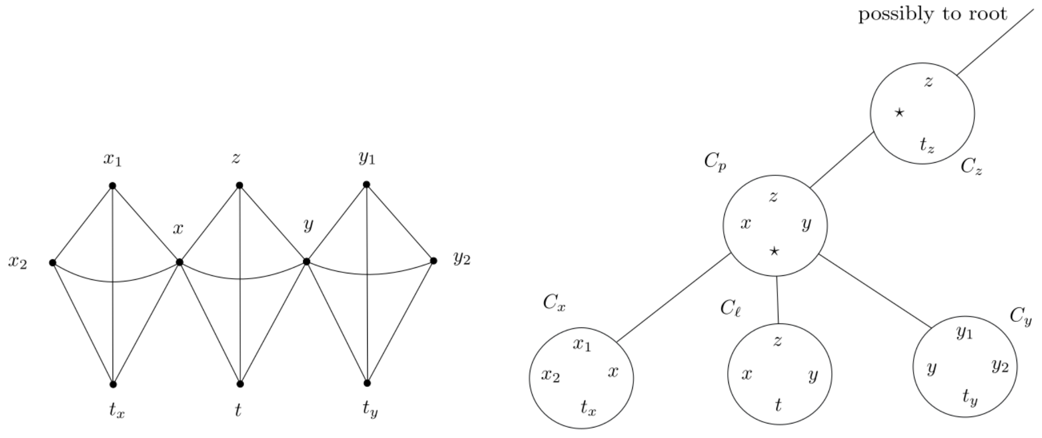
Claim. $\{x, y, z\} \subseteq C_p$, and C_p does not contain a terminal vertex.

Proof. From Observation 9 we know that t is the only simplicial vertex in C_ℓ. For the sake of contradiction, assume that x is not in C_p. Then from the intersection property of clique trees we get that x is not present in any maximal clique apart from C_ℓ. Thus (Fact 1) x is a simplicial vertex, a contradiction. Repeating this argument, we get $\{x, y, z\} \subseteq C_p$.

If C_p contains a terminal vertex t' then t' has all of $\{x, y, z\}$ as neighbours, and since t is simplicial we get that $t' \neq t$. This contradicts part 5 of Observation 9.

Parts 2 and 5 of Observation 9 together imply that each of x, y, z must have at least one terminal other than t as a neighbour, and that these terminals must be pairwise distinct: each such terminal is adjacent to exactly one of x, y, z. Let t_x, t_y, t_z be three such terminal vertices which are adjacent to x, y, z, respectively. From Claim 5.3 we know that none of t_x, t_y, t_z is in C_p, and it is trivially the case that none of these vertices is in C_ℓ either.

Let C'_x, C'_y, C'_z be three maximal cliques that contain the edges $\{x, t_x\}, \{y, t_y\}, \{z, t_z\}$, respectively. Then we have that clique C'_x does not contain vertex y, and similarly for the other pairs. Thus the maximal cliques C'_x, C'_y, C'_z intersect with C_ℓ exactly



(a) (b)

Fig. 3. Left side figure shows vertices and edges in the input graph. Right side shows the clique tree of given graph. Every circle is a maximal clique in input graph. ($\star$) denotes that all vertices in maximal clique are not shown in the figure.

at $\{x\}, \{y\}, \{z\}$, respectively. Now let C_x, C_y, C_z be maximal cliques *closest to* C_p *in the clique tree* $\mathcal{T}_G$ such that $t_i \in C_i$ and $C_i \cap C_\ell = \{i\}$ for $i \in \{x, y, z\}$.

Claim. At least two among the cliques $\{C_x, C_y, C_z\}$ are leaf nodes of the clique tree $\mathcal{T}_G$.

Proof. From item 3 of Fact 2 and from our assumption about distances, we get that all of C_x, C_y, C_z must be adjacent to C_p in the clique tree $\mathcal{T}_G$. Since C_p can have at most one parent node in $\mathcal{T}_G$, we get that at least two among C_x, C_y, C_z, say C_x and C_y, are child nodes of C_p. The child node C_ℓ of C_p is, by assumption, a leaf which is farthest from the root node C_r. It follows that neither C_x nor C_y can be internal nodes in $\mathcal{T}_G$, or else there would be a leaf which is farther from C_r than is C_ℓ. Thus both C_x and C_y are leaf nodes of $\mathcal{T}_G$.

From now on we assume that C_x and C_y are leaf nodes of $\mathcal{T}_G$. By Observation 1 we have that C_x and C_y are simplicial, and by part 4 of Observation 9 we get that these two cliques contain exactly four vertices each, and that t_x, t_y are the only terminals in C_x, C_y, respectively. Recall that $C_\ell = \{t, x, y, z\}$. Let $C_x = \{t_x, x, x_1, x_2\}$ and $C_y = \{t_y, y, y_1, y_2\}$.

We now branch on vertex t (See Figure 4 and Table 1). If t is in the solution then none of $\{x, y, z\}$ is in the solution. Since x is not in the solution, we have that either t_x is in the solution, or both of $\{x_1, x_2\}$ are in the solution. Similarly we get that either t_y is in the solution, or both of $\{y_1, y_2\}$ are in the solution. If t is *not* in the solution then either (i) x is in the solution or (ii) x is not in the

Branch	Vertices picked	New graph G_i	New terminal set T_i	New parameter k_i
B_1	$\{t, t_x, t_y\}$	$G - \{t, t_x, t_y\}$	$T \setminus \{t, t_x, t_y\}$	$k - 3$
B_2	$\{t, t_x, y_1, y_2\}$	$G - \{t, t_x, y_1, y_2\}$	$T \setminus \{t, t_x\}$	$k - 4$
B_3	$\{t, x_1, x_2, t_y\}$	$G - \{t, , x_1, x_2, t_y\}$	$T \setminus \{t, t_y\}$	$k - 4$
$B_4(i)$	$\{t, x_1, x_2, y_1, y_2\}$	$G - \{t, x_1, x_2, y_1, y_2\}$	$T \setminus \{t\}$	$(k - 5)$
$B_4(ii)$	$\{t, x_1, x_2, y_1\}$	$G - \{t, x_1, x_2, y_1\}$	$T \setminus \{t\}$	$(k - 4)$
B_5	$\{x\}$	$G - \{x\}$	T	$k - 1$
B_6	$\{y, z, t_x\}$	$G - \{y, z, t_x\}$	$T \setminus \{t_x\}$	$k - 3$
B_7	$\{y, z, x_1, x_2\}$	$G - \{y, z, x_1, x_2\}$	T	$k - 4$

Table 1. Overview of Branching Rule 7 showing the vertices picked in the solution and the resulting graph, terminal set, and parameter in each branch. Exactly *one* of the two versions of rule B_4 are applicable to any one instance, depending on whether the sets C_x and C_y share a vertex. See also the branching tree in Figure 4.

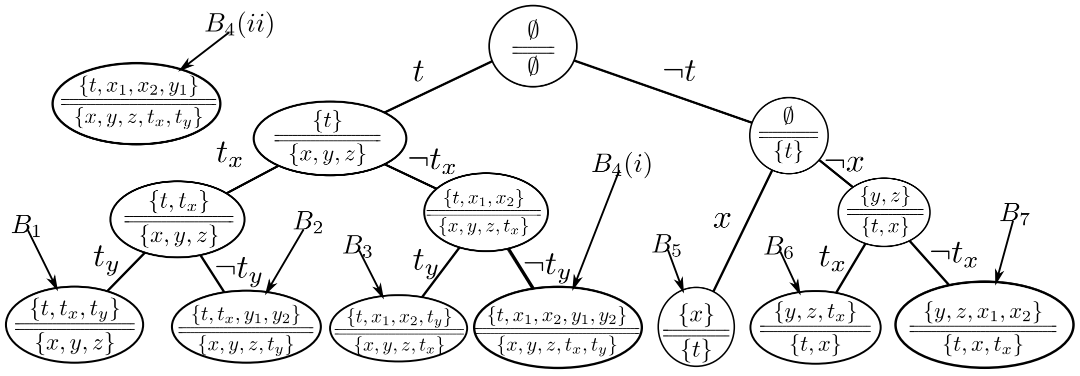

$\emptyset$ / $\emptyset$
t
$\neg t$
$\{t\}$ / $\{x,y,z\}$
$\emptyset$ / $\{t\}$
t_x
$\neg t_x$
x
$\neg x$
$\{t,t_x\}$ / $\{x,y,z\}$
$\{t,x_1,x_2\}$ / $\{x,y,z,t_x\}$
$\{y,z\}$ / $\{t,x\}$
t_y
$\neg t_y$
t_y
$\neg t_y$
t_x
$\neg t_x$
$B_4(ii)$
$\{t,x_1,x_2,y_1\}$ / $\{x,y,z,t_x,t_y\}$
B_1
B_2
B_3
$B_4(i)$
B_5
B_6
B_7
$\{t,t_x,t_y\}$ / $\{x,y,z\}$
$\{t,t_x,y_1,y_2\}$ / $\{x,y,z,t_y\}$
$\{t,x_1,x_2,t_y\}$ / $\{x,y,z,t_x\}$
$\{t,x_1,x_2,y_1,y_2\}$ / $\{x,y,z,t_x,t_y\}$
$\{x\}$ / $\{t\}$
$\{y,z,t_x\}$ / $\{t,x\}$
$\{y,z,x_1,x_2\}$ / $\{t,x,t_x\}$

Fig. 4. The seven branches of Branching Rule 7. An edge label denotes the choice we make of picking (e.g: "t") or excluding (e.g: "$\neg t$") a vertex in/from the solution. The set of vertices that the rule has picked in the solution thus far appears above the double line "$=$" in each node, and the set *excluded* from the solution appears below $=$. The seven leaf nodes correspond to the seven branches $B_1, \cdots, B_7$ of the branching rule. The branch B_4 takes *one* of two forms depending on the instance. If $C_x \cap C_y = \emptyset$ then option $B_4(i)$ applies, and if $C_x \cap C_y \neq \emptyset$ then option $B_4(ii)$ applies.

solution and both of $\{y, z\}$ are in the solution. In the second case, since x is not in the solution, we get that either t_x is in the solution, or both of $\{x_1, x_2\}$ are in the solution. We summarize these seven branches in Table 1. Note that C_x and C_y could share—at most—one vertex, in which case the set $\{x_1, x_2, y_1, y_2\}$ will contain three vertices, not four. Rule $B_4(i)$ applies when $|C_x \cap C_y| = 0$ and rule $B_4(ii)$ applies when $|C_x \cap C_y| = 1$. In stating rule $B_4(ii)$ we have assumed that vertex x_1 is common to both cliques; $C_x = \{t_x, x, x_1, x_2\}$ and $C_y = \{t_y, y, x_1, y_1\}$. On every instance we apply *one* of the two variants $B_4(i)$ and $B_4(ii)$ of rule B_4, as appropriate.

Branching Rule 7. *Let $(G; T; k)$ be an instance of* SUBSET FVS IN CHORDAL GRAPHS *and let $C_\ell = \{t, x, y, z\}, C_x = \{x, t_x, x_1, x_2\}, C_y = \{y, t_y, y_1, y_2\}$ be leaf nodes of the clique tree $\mathcal{T}_G$ of G. For each i ; $1 \leq i \leq 7$, let graph G_i, terminal set T_i, and parameter k_i be as described in Table 1, with the proviso that rule $B_4(i)$ applies if and only if $|C_x \cap C_y| = 0$ and rule $B_4(ii)$ applies if and only if $|C_x \cap C_y| = 1$. The new instances are: $\{(G_i; T_i; k_i)\}$; $1 \leq i \leq 7$.*

Lemma 35. *Branching Rule 7 is exhaustive and it can be executed in time $\mathcal{O}(n+m)$ on instance which is not reducible by any of previously mentioned reduction or branching rule.*

Proof. Consider an instance $(G; T; k)$ of SUBSET FVS IN CHORDAL GRAPHS and cliques C_ℓ, C_x, C_y as described in the statement of branching rule. Let $(G_i; T_i; k_i)$ for each i ; $1 \leq i \leq 7$, be seven instances produced by Branching Rule 7 when applied on $(G; T; k)$ as described in Table 1.

($\Rightarrow$) From Corollary 9, there exists a selective solution, say S. Since $\{t, x, y, z\}$ is a clique containing terminal, S contains at least one vertex of this clique. By definition of selective solution, if S contains t then S contains none of $\{x, y, z\}$. We first see the implication of the fact that S does not contain x in clique C_x. Since x is not a part of solution, $S \cap C_x$ either contains t_x or both x_1, x_2. Hence if S contains t then it either contains t_x or both x_1, x_2. We apply similar argument with respect to y. This implies S either contains t_y or both y_1, y_2. Hence we can conclude that if S contains t then at least one of sets $\{t_x, t_y\}, \{t_x, y_1, y_2\}, \{x_1, x_2, t_y\}, \{x_1, x_2, y_1, y_2\}$ is contained in S. Note that set $\{x_1, x_2, y_1, y_2\}$ may not contain four distinct elements.

Consider the case when selective solution does not contain t. Since t is a terminal, S must contain at least two vertices from x, y, z. Instead of analyzing three cases, viz whether $\{x, y\}, \{y, z\}$ or $\{z, x\}$, we analyze two cases depending on whether or not x is contained in S. We do this to minimize the complexity in stating this branching rule. If x is not contained in S then it contains $\{y, z\}$. Moreover, S either contains t_x or both x_1, x_2. This implies if S does not contain t then S contains at least one of sets $\{x\}, \{y, z, t_x\}, \{y, z, t_x\}$. We have established that if G is a YES instance then by Corollary 9, there exits a selective solution. This selective solution contains at least one of the seven sets mentioned above. Hence if $(G; T; k)$ is a YES instance then at least one of the seven instances specified in Table 1 is a YES instance.

($\Leftarrow$) We use the fact that for every $U \subseteq$ if S' is a target solution of $(G - U; T \setminus U; k - |U|)$ then $S' \cup U$ is a target solution of $(G; T; k)$ to prove reverse direction.

To execute branching rule, we need an algorithm which finds cliques C_ℓ, C_x, C_y which satisfies the mentioned property or conclude that no such cliques exists. We assume graph G is a connected graph. If G is not connected, we can process each of its connected component separately. We first compute clique tree for of graph G in time $\mathcal{O}(n + m)$ (Fact 2). Once clique tree is computed, we can arbitrary root it at any internal node. If clique tree does not have an internal node than it has at most two maximal clique each of which contains at most four vertices. In this case, problem can be solved in constant time. Fix any node which is at farthest distance from the root as C_ℓ. Let $C_\ell = \{t, x, y, z\}$. We now argue that there exists cliques C_x, C_y as desired by branching rule assuming the input instance is not reducible by any reduction and branching rules mentioned before this branching rule.

Let C_p be the parent of C_ℓ in this rooted tree. By Claim 5.3, $\{x, y, z\}$ is present in C_p. Parts 2 and 5 of Observation 9 together imply that each of x, y, z must have at least one terminal other than t as a neighbour, and that these terminals must be pairwise distinct. This implies that C_p has at least three neighbors apart from C_ℓ in the clique tree. Let C_x, C_y be any two neighbors of C_p which are not in the path from C_p to the root. This implies that distance between root and C_ℓ is same as distance between the root and C_x or C_y. Since C_ℓ is the leaf which is farthest from the root, C_x, C_y both are leaves in the clique tree. By part 4 of Observation 9 we get that all simplicial cliques contain exactly four vertices including one terminal each. This proves that any leaf which is farthest from the root can be used as simplicial clique C_ℓ in branching process and concludes the proof of lemma.

We have mentioned all branching rules and now are in a position to conclude main result of this section.

Theorem 5. *There exists an algorithm which given an instance (G, T, k) of* SUBSET FVS IN CHORDAL GRAPHS *runs in time $\mathcal{O}(2^k(n + m))$ and decides whether input is YES of NO instance. Here n, m are number of vertices and edges in input graph G.*

Proof. The algorithm applies Reduction Rules 13, 14 and 15 exhaustively (i) to the input graph, and (ii) after every application of a branching rule. By Lemma 25, this can be done in time $\mathcal{O}(n + m)$. We assume that graph G is reduced with respect to these rules at the start of any branching rule.

The algorithm applies least indexed applicable Branching Rule mentioned in this section. The correctness of branching steps and running time to execute branching rule follows from the arguments given for each case. We now claim that we have taken care of all possible cases. Application of reduction rules implies that every maximal clique is of size at least tree and every non-terminal vertex is adjacent with some terminal. If there exists a non-terminal vertex which is adjacent with exactly one terminal vertex then Branching Rule 1 is applicable. If there exists a maximal clique of size of five or more which contains a terminal then Branching Rule 3 is applicable. Hence we can safely consider chordal graphs in which every maximal clique which contains a terminal is of size at most four. Every chordal graph has a simplicial vertex and hence simplicial clique. If simplicial clique is of size three then Branching Rule 2 is applicable. We are now in case that every simplicial clique which contains a terminal is of size four. Notice that since Reduction Rule 14 is not applicable, every simplicial clique contains a terminal. If this terminal is not a simplicial vertex then Branching Rule 4 is applicable. We are left with the case when every simplicial vertex is terminal. If there exists another terminal in simplicial clique then Branching Rule 5 is applicable. If not then every simplicial clique contains exactly one terminal which is also a simplicial vertex. In a simplicial clique if there exists a pair of non-terminals which has more than one terminals as neighbors then Branching Rule 6 is applicable. Now consider a graph which is not reducible by any of the branching rule. This graph satisfies all the properties mentioned in Observation 9. Hence Branching Rule 7 is applicable on this instance. This completes all the possible cases.

We analyze the branching factor for each branching rule. Branching vectors for Branching rules 1, 2, 3, 4, 5, and 6 are $(1, 1)$; $(1, 1)$; $(1, 2, 2)$; $(1, 2)$; $(2, 2, 2)$; and $(2, 2, 2, 2)$, respectively. Branching vector for Branching Rule 7 is $(3, 4, 4, 5, 4, 1, 3, 4)$ or $(3, 4, 4, 4, 4, 1, 3, 4)$ depending on whether branch $B_4(i)$ or $B_4(ii)$ is being used. For all these branching vectors, branching factor is at most 2. Hence the entire branching algorithm can be executed in time $\mathcal{O}(2^k(n + m))$.

6 Conclusion

In this article, we studied SUBSET FVS IN SPLIT GRAPHS and presented a kernel of size $\mathcal{O}(k^2)$ which contains $\mathcal{O}(k)$ and $\mathcal{O}(k^2)$ vertices on clique sides and independent set side respectively. Even though size of kernel size is optimal under standard complexity assumptions, it is intersting if we can bound the number of vertices on independet set by $\mathcal{O}(k^{2-\epsilon})$ for some positive constant ϵ. Another natual question is to obtain a kernel for SUBSET FVS IN CHORDAL GRAPHS problem. We present an FPT algorithm running in time $\mathcal{O}^*(2^k)$ which solves SUBSET FVS when input graph is chordal. Under ETH, sub-exponential FPT algorithms for this problem are ruled out. Is it possible to obtain an algorithm with smaller base of exponent in running time? It is also interesting to identify other implicit hitting set problems originated in graph theory and obtain better kernel and FPT results than the onces guranteed by HITTING SET.

References

1. Faisal N. Abu-Khzam. A kernelization algorithm for d-hitting set. *J. Comput. Syst. Sci.*, 76(7):524–531, 2010.
2. Jean RS Blair and Barry Peyton. An introduction to chordal graphs and clique trees. In *Graph theory and sparse matrix computation*, pages 1–29. Springer, 1993.
3. Rajesh Chitnis, Fedor V Fomin, Daniel Lokshtanov, Pranabendu Misra, MS Ramanujan, and Saket Saurabh. Faster exact algorithms for some terminal set problems. In

International Symposium on Parameterized and Exact Computation, pages 150–162. Springer, 2013.

4. Rajesh Hemant Chitnis, Marek Cygan, Mohammad Taghi Hajiaghayi, and Dániel Marx. Directed subset feedback vertex set is fixed-parameter tractable. *ACM Trans. Algorithms*, 11(4):28:1–28:28, 2015.
5. Marek Cygan, Fedor V. Fomin, Lukasz Kowalik, Daniel Lokshtanov, Dániel Marx, Marcin Pilipczuk, Michal Pilipczuk, and Saket Saurabh. *Parameterized Algorithms*. Springer, 2015.
6. Marek Cygan, Marcin Pilipczuk, Michał Pilipczuk, and Jakub Onufry Wojtaszczyk. Subset feedback vertex set is fixed-parameter tractable. *SIAM Journal on Discrete Mathematics*, 27(1):290–309, 2013.
7. Holger Dell and Dieter Van Melkebeek. Satisfiability allows no nontrivial sparsification unless the polynomial-time hierarchy collapses. *J. ACM*, 61(4):23:1–23:27, July 2014.
8. Reinhard Diestel. *Graph Theory, 5th Edition*. Springer, 2016.
9. Guy Even, Joseph Naor, and Leonid Zosin. An 8-approximation algorithm for the subset feedback vertex set problem. *SIAM Journal on Computing*, 30(4):1231–1252, 2000.
10. Fedor V Fomin, Pinar Heggernes, Dieter Kratsch, Charis Papadopoulos, and Yngve Villanger. Enumerating minimal subset feedback vertex sets. *Algorithmica*, 69(1):216–231, 2014.
11. Fedor V Fomin, Daniel Lokshtanov, Neeldhara Misra, Geevarghese Philip, and Saket Saurabh. Hitting forbidden minors: Approximation and kernelization. *SIAM Journal on Discrete Mathematics*, 30(1):383–410, 2016.
12. F.V. Fomin and D. Kratsch. *Exact Exponential Algorithms*. Texts in Theoretical Computer Science. An EATCS Series. Springer Berlin Heidelberg, 2010.
13. Philippe Galinier, Michel Habib, and Christophe Paul. Chordal graphs and their clique graphs. In *International Workshop on Graph-Theoretic Concepts in Computer Science*, pages 358–371. Springer, 1995.
14. Petr A Golovach, Pinar Heggernes, Dieter Kratsch, and Reza Saei. Subset feedback vertex sets in chordal graphs. *Journal of Discrete Algorithms*, 26:7–15, 2014.
15. Martin Charles Golumbic. *Algorithmic graph theory and perfect graphs*, volume 57. Elsevier, 2004.
16. Peter L Hammer and Stéphane Földes. Split graphs. *Congressus Numerantium*, 19:311–315, 1977.
17. Peter L Hammer and Bruno Simeone. The splittance of a graph. *Combinatorica*, 1(3):275–284, 1981.
18. Eva-Maria C Hols and Stefan Kratsch. A randomized polynomial kernel for subset feedback vertex set. *Theory of Computing Systems*, 62(1):63–92, 2018.
19. Ken-ichi Kawarabayashi and Yusuke Kobayashi. Fixed-parameter tractability for the subset feedback set problem and the s-cycle packing problem. *Journal of Combinatorial Theory, Series B*, 102(4):1020–1034, 2012.
20. Tien-Nam Le, Daniel Lokshtanov, Saket Saurabh, Stéphan Thomassé, and Meirav Zehavi. Subquadratic kernels for implicit 3-hitting set and 3-set packing problems. In *Proceedings of the Twenty-Ninth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 331–342. SIAM, 2018.
21. Daniel Lokshtanov, MS Ramanujan, and Saket Saurabh. Linear time parameterized algorithms for subset feedback vertex set. *ACM Transactions on Algorithms (TALG)*, 14(1):7, 2018.
22. Stéphan Thomassé. A $4k^2$ kernel for feedback vertex set. *ACM Trans. Algorithms*, 6(2):32:1–32:8, 2010.

23. Magnus Wahlström. *Algorithms, measures and upper bounds for satisfiability and related problems*. PhD thesis, Department of Computer and Information Science, Linköpings universitet, 2007.
24. Magnus Wahlström. Half-integrality, LP-branching and FPT algorithms. In *Proceedings of the twenty-fifth annual ACM-SIAM symposium on Discrete algorithms*, pages 1762–1781. SIAM, 2014.