

Faster Exact Algorithms for Some Terminal Set Problems

Rajesh Chitnis¹, Fedor V. Fomin², Daniel Lokshtanov², Pranabendu Misra³,
M. S. Ramanujan³, and Saket Saurabh^{2,3}

¹ University of Maryland, USA rchitnis@cs.umd.edu

² University of Bergen, Norway. {fedor.fomin|daniello}@ii.uib.no

³ Institute of Mathematical Sciences, India.

{pranabendu|msramanujan|saket}@imsc.res.in

Abstract. Many problems on graphs can be expressed in the following language: given a graph $G = (V, E)$ and a terminal set $T \subseteq V$, find a minimum size set $S \subseteq V$ which intersects all “structures” (such as cycles or paths) passing through the vertices in T . We call this class of problems as *terminal set problems*. In this paper we introduce a general method to obtain faster exact exponential time algorithms for many terminal set problems. More precisely, we show that

- NODE MULTIWAY CUT can be solved in time $O(1.4766^n)$.
- DIRECTED UNRESTRICTED NODE MULTIWAY CUT can be solved in time $O(1.6181^n)$.
- There exists a deterministic algorithm for SUBSET FEEDBACK VERTEX SET running in time $O(1.8980^n)$ and a randomized algorithm with expected running time $O(1.8826^n)$. Furthermore, SUBSET FEEDBACK VERTEX SET on chordal graphs can be solved in time $O(1.6181^n)$.
- DIRECTED SUBSET FEEDBACK VERTEX SET can be solved in time $O(1.9993^n)$.

A key feature of our method is that, it uses the existing best polynomial time, fixed parameter tractable and exact exponential time algorithms for the non-terminal version of the same problem (i.e. when $T = V$), as subroutines. Therefore faster algorithms for these special cases will imply further improvements in the running times of our algorithms. Our algorithms for NODE MULTIWAY CUT, and SUBSET FEEDBACK VERTEX SET on chordal graphs improve the current best algorithms for these problems and answers an open question posed in [15]. Furthermore, our algorithms for DIRECTED UNRESTRICTED NODE MULTIWAY CUT and DIRECTED SUBSET FEEDBACK VERTEX SET are the first exact algorithms improving upon the brute force $O^*(2^n)$ -algorithms.

1 Introduction

The goal of the design of moderately exponential time algorithms for NP-complete problems is to establish algorithms for which the worst-case running time is provably faster than the one of enumerating all prospective solutions, or loosely

speaking, algorithms better than brute-force enumeration. For example, for NP-complete problems on graphs on n vertices and m edges whose solutions are either subsets of vertices or edges, the brute-force or trivial algorithms basically enumerate all subsets of vertices or edges. This mostly leads to algorithms of time complexity $O^*(2^n)$ or $O^*(2^m)$, based on whether we are enumerating vertices or edges⁴. Thus the goal of exact algorithms for graph problems is to improve upon the algorithms running in time $O^*(2^n)$ or $O^*(2^m)$. See the book [11] for an introduction to exact exponential algorithms.

One of the most well studied direction in exact algorithms is to delete vertices of the input graph such that the resulting graph satisfies some interesting properties. More precisely, a natural optimization problem associated with a graph class \mathcal{G} is the following: given a graph G , what is the minimum number of vertices to be deleted from G to obtain a graph in \mathcal{G} ? For example, when \mathcal{G} is the class of empty graphs, forests or bipartite graphs, the corresponding problems are VERTEX COVER (VC), FEEDBACK VERTEX SET (FVS) and ODD CYCLE TRANSVERSAL (OCT), respectively. The best known algorithms for VC, FVS and OCT run in time $O^*(1.2108^n)$ [26], $O^*(1.7347^n)$ [12] and $O^*(1.4661^n)$ [20, 26] respectively. The other problems in this class for which non-trivial exact algorithms are known include finding an induced r -regular subgraph [16], induced subgraph of bounded degeneracy [23] and induced subgraph of bounded treewidth [12].

In this paper we study another class of graph problems which we call as *terminal set problems*. In these problems, the input is a graph $G = (V, E)$ and a terminal set $T \subseteq V$, and the goal is to find a minimum size set $S \subseteq V$ that intersects certain structures such as cycles or paths passing through the vertices in T . In this paper we introduce a general method to obtain faster exact exponential time algorithms for many terminal set problems. The general algorithmic technique is the following. Let the size of the terminal set T be k . We first observe that the size of the optimum solution is at most k (or a function of k). Let S be an optimum solution to the problem and let $X = S \cap (V \setminus T)$. We guess X and delete it from G . Since $S \setminus X \subseteq T$, we create an auxiliary graph on T and determine the rest of the solution using either a known polynomial time algorithm, or a fixed parameter tractable algorithm, or a non-trivial exact algorithm for the non-terminal version (when $T = V$) of the same problem. We now provide a list of problems for which we give improved or new algorithms using our method together with a short overview of previous work on each application.

NODE MULTIWAY CUT and DIRECTED UNRESTRICTED NODE MULTIWAY CUT: A fundamental min-max theorem about connectivity in graphs is Menger's Theorem, which states that the maximum number of vertex disjoint paths between two vertices s and t , is equal to the minimum number of vertices whose removal separates these two vertices. Indeed, a maximum set of vertex disjoint paths between s, t and a minimum size set of vertices separating these two vertices

⁴ Throughout this paper we use the O^* notation which suppresses polynomial factors.

can be computed in polynomial time. A known generalization of this theorem, commonly known as Mader’s T -path Theorem [18] states that, given a graph G and a subset T of vertices, there are either k vertex disjoint paths with only the end points in T (such paths are called T -paths), or there is a vertex set of size at most $2k$ which intersects every T -path. Although computing a maximum set of vertex disjoint T -paths can be done in polynomial time by using matching techniques, the decision version of the dual problem of finding a minimum set of vertices that intersects every T -path is NP-complete for $|T| > 2$. Formally, this problem is the following classical NODE MULTIWAY CUT problem.

NODE MULTIWAY CUT (NMC)

Input: An undirected graph $G = (V, E)$ and a set of terminals $T = \{t_1, t_2, \dots, t_k\}$.

Question: Find a set $S \subseteq V(G) \setminus T$ of minimum size such that $G \setminus S$ has no path between a t_i, t_j pair for any $i \neq j$.

This is a very well studied problem in terms of approximation, as well as parameterized algorithms [5, 8, 14, 19]. A variant of this problem where S is allowed to intersect the set T , is known as UNRESTRICTED NODE MULTIWAY CUT (UNMC). The best known parameterized algorithm for NODE MULTIWAY CUT decides in time $O^*(2^\ell)$ whether there is a solution of size at most ℓ or not. Fomin et al. [10] designed an exact algorithm for UNMC running in time $O^*(1.8638^n)$. In this paper we design an algorithm with running time $O(1.4766^n)$ for both NMC and UNMC.

Next we consider the directed variant of NODE MULTIWAY CUT, namely DIRECTED NODE MULTIWAY CUT (DNMC) where the input is a directed graph and a set $T = \{t_1, \dots, t_k\}$ of terminals and the objective is to find a set of minimum size which intersects every $t_i \rightarrow t_j$ path for every $t_i, t_j \in T$ with $i \neq j$. For the *unrestricted* version of this problem, namely DIRECTED UNRESTRICTED NODE MULTIWAY CUT (DUNMC), we design an exact algorithm with running time $O(1.6181^n)$.

SUBSET FEEDBACK VERTEX SET *and* DIRECTED SUBSET FEEDBACK VERTEX SET: An exact algorithm for FEEDBACK VERTEX SET (FVS) – finding a minimum sized vertex subset such that its removal results in an acyclic graph – remained elusive for several years. In a breakthrough paper Razgon [24] designed an exact algorithm for this problem running in time $O^*(1.8899^n)$. Later, Fomin et al. [9] building upon the work in [24] designed an exact algorithm for FVS running in time $O^*(1.7548^n)$. The current best known algorithm for this problem uses potential maximal clique machinery and runs in time $O^*(1.7347^n)$ [13]. Razgon studied the directed version of FVS and obtained an exact algorithm running in time $O^*(1.9977^n)$ [25]. This is the only known non-trivial exact algorithm for DIRECTED FEEDBACK VERTEX SET (DFVS). A natural generalization of the FEEDBACK VERTEX SET problem is when we only want to hit all the cycles passing through a specified set of terminals. This leads to the following problem.

SUBSET FEEDBACK VERTEX SET (SFVS)

Input: An undirected graph $G = (V, E)$, a set of terminals $T \subseteq V$ of size k

Question: Find a minimum set of vertices which hits every cycle passing through T

Fomin et al. [10] designed an algorithm for SFVS on general graphs which runs in time $O^*(1.8638^n)$. It is important to note that their algorithm not only finds a minimum sized solution, but also enumerates all minimal solutions in the same time. Using our methodology we design an algorithm for SFVS which runs in time $O^*(1.8980^n)$. However, if we are allow randomization then we can design an algorithm with an expected running time of $O^*(1.8826^n)$. Golovach et al. [15] initiated the study of exact algorithms for SFVS on special graph classes by giving an enumeration algorithm for SFVS on chordal graphs which runs in time $O^*(1.6708^n)$. They left it as an open question whether there exists algorithms for SFVS on chordal graphs (even on split graphs) which are faster than $O^*(1.6708^n)$. Though our algorithm using the described methodology for SFVS in general does not improve on the best known algorithm, it answers this question in the affirmative for SFVS on chordal graphs and we obtain an algorithm with running time $O(1.6181^n)$. More generally, our algorithm for SFVS runs in $O(1.6181^n)$ on any graph class \mathcal{G} which is closed under vertex deletions and edge contractions, and where the weighted FVS problem can be solved in polynomial time. Finally, we also consider the directed variant of the SFVS problem, namely DIRECTED SUBSET FEEDBACK VERTEX SET (DSFVS), and obtain an algorithm with running time $O(1.9993^n)$.

2 Preliminaries

Let C be a cycle in a graph G . A *chord* of C is an edge $e \notin C$ which connects two vertices of C . A graph G is called a *chordal graph* if every cycle on four or more vertices has a *chord*.

Now we define the *contraction* of an edge or a subgraph in G . Let G be an undirected graph and let (u, v) be an edge in G . Let G' be the graph obtained from G in the following manner. We add a new vertex w . For every edge (u, z) where $z \neq v$ we add an edge (w, z) , and for every edge (y, v) where $y \neq u$ we add an edge (y, w) . Finally we delete the vertices u and v , and any parallel edges from the graph. We say that G' is obtained from G by *contracting* the edge (u, v) . Let H be a subgraph of G . Consider the graph G' obtained from G by contracting every edge of H in a arbitrary order. We say that G' is obtained from G by contracting the subgraph H .

Now we define the *torso graph* of a subset of vertices in G . Let $G = (V, E)$ be an undirected graph and T and V' be subsets of V . We denote by $\text{torso}(T, V')$ the graph defined in the following manner. The vertex set of this graph is T and the edge set comprises of all pairs (t_i, t_j) such that there is a $t_i - t_j$ path in G whose internal vertices lie in $V' \setminus T$ or there is an edge $(t_i, t_j) \in E$.

We also define an analogous notion of a *torso graph* in directed graphs. Let $D = (V, A)$ be an directed graph and T and V' be subsets of V . We denote by

$\text{torso}(T, V')$ the digraph defined in the following manner. The vertex set of this graph is T and the edge set comprises of all ordered pairs (t_i, t_j) such that there is a directed $t_i - t_j$ path in D whose internal vertices lie in $V' \setminus T$ or there is an edge $(t_i, t_j) \in A$.

Finally, we define a generalization of the *torso graphs*. Let $G = (V, E)$ be an undirected graph and T_1, T_2, \dots, T_ℓ and V' be subsets of V . We denote by $\ell\text{-torso}(T_1, \dots, T_\ell, V')$ the graph defined as follows. It has vertex set $T = T_1 \cup T_2 \cup \dots \cup T_\ell$ and the edge set comprises of all pairs (t_i, t_j) such that $t_i \in T_{i'}$ and $t_j \in T_{j'}$ for some $i' \neq j'$ and there is a $t_i - t_j$ path in G whose internal vertices lie in $V' \setminus T$.

3 Node Multiway Cut

In this section we design an exact algorithm for the NODE MULTIWAY CUT problem. We begin by giving an algorithm for unrestricted version of this problem.

3.1 Unrestricted Node Multiway Cut

The following observation follows from the fact that the set of terminals in an instance of UNRESTRICTED NODE MULTIWAY CUT itself is a potential solution.

Observation 1 *Let (G, T) be an instance of UNRESTRICTED NODE MULTIWAY CUT and S be an optimum solution to this instance. Then $|S| \leq |T|$.*

Now we design an algorithm for UNRESTRICTED NODE MULTIWAY CUT using the FPT algorithm for NODE MULTIWAY CUT and Observation 1. This algorithm uses the FPT algorithm for multiway cut of Cygan et al.[8]. We will use this algorithm for the instances where k is “small”.

Lemma 1. *Let (G, T) be an instance of UNRESTRICTED NODE MULTIWAY CUT where $|T| = k$. Then we can find an optimum solution to this instance in time $O^*(2^k)$.*

Proof. We begin by performing the following transformation on the given instance. For each terminal $t_i \in T$, we add a new vertex t'_i and make it adjacent to t_i alone. Let T' be the set of these new vertices and G' be the graph thus constructed. It is easy to see that the set of optimum solutions to the instance (G', T') of NODE MULTIWAY CUT and the set of optimum solutions to the instance (G, T) of Unrestricted NODE MULTIWAY CUT are in one to one correspondence. Therefore, we may utilise the algorithm of [8] which solves NODE MULTIWAY CUT in time $O^*(2^\ell)$ where ℓ is the size of the optimum solution. By Observation 1, we have that $\ell \leq k$ and therefore, the statement of the lemma follows. \square

Next, we design another algorithm for UNRESTRICTED NODE MULTIWAY CUT which will be used for the instances where k is “large”.

Lemma 2. *Let (G, T) be an instance of UNRESTRICTED NODE MULTIWAY CUT where $G = (V, E)$ and let S be an optimum solution to this instance. Let $X = S \cap (V \setminus T)$ and $Y = S \setminus X$. Then Y is a vertex cover of the graph $\text{torso}(T, V \setminus X)$. Conversely, if Y' is any vertex cover for the graph $\text{torso}(T, V \setminus X)$, then the set $X \cup Y'$ is a solution to this instance.*

Proof. We first show that Y is indeed a vertex cover of $G' = \text{torso}(T, V \setminus X)$. Let E' be the edge set of G' . Suppose that Y is not a vertex cover of G' and there is an edge $(t_i, t_j) \in E'$ which is not covered by Y . Observe that $(t_i, t_j) \notin E$, since this would contradict the assumption of S being a solution. Therefore, it must be the case that there is a path P between t_i and t_j in $G[V \setminus X]$ whose internal vertices are disjoint from T . Since this path is disjoint from both X and Y , it is also present in the graph $G \setminus S$, a contradiction. Hence, we conclude that Y is indeed a vertex cover of $\text{torso}(T, V \setminus X)$.

We now show that for any vertex cover Y' of G' , the set $X \cup Y'$ is a solution to the instance (G, T) of UNRESTRICTED NODE MULTIWAY CUT. Suppose to the contrary that there is a vertex cover Y' of G' such that the set $S' = X \cup Y'$ is not a solution to the instance (G, T) . That is, there is a t_i - t_j path in $G \setminus S'$ for some $t_i, t_j \in T$. Observe that this implies the existence of a $t_{i'}$ - $t_{j'}$ path P for some $t_{i'}, t_{j'} \in T$ such that the internal vertices of P are disjoint from $T \cup S'$. Therefore the edge $(t_{i'}, t_{j'})$ is not covered by Y' in G' , a contradiction. This completes the proof of this lemma. \square

Using the above lemma and the FPT algorithm for Vertex Cover of Chen et al.[4], we are able to show the following lemma.

Lemma 3. *There is an algorithm that, given an instance $(G = (V, E), T)$ of UNRESTRICTED NODE MULTIWAY CUT, runs in time $O\left(1.7850^n \left(\frac{1.2738}{1.7850}\right)^k\right)$ and returns an optimum solution where $k = |T|$ and $n = |V|$.*

Proof. The description of the algorithm is as follows. For every $X \subseteq (V \setminus T)$ such that $|X| \leq k$, we construct the graph $G_X = \text{torso}(T, V \setminus X)$ and compute a minimum vertex cover Y_X for G_X . We compute the minimum vertex cover by using the FPT algorithm of [4], which runs in time $O^*(1.2738^\ell)$ where ℓ is the size of an optimum vertex cover. Finally, we return the set $X \cup Y_X$ which is a smallest solution over all choices of X . The correctness of this algorithm follows from Lemma 2.

In order to bound the running time of this algorithm, first observe that for each X , the set Y_X has size at most $k - |X|$. Therefore, the FPT algorithm we use to compute a minimum vertex cover of $\text{torso}(T, V \setminus X)$ runs in time $O^*(1.2738^{k-|X|})$. Summing over all choices of X , the time taken by our algorithm

is upper bounded by

$$\begin{aligned}
& \sum_{x=0}^k \binom{n-k}{x} O^*(1.2738^{k-x}) \\
&= O^*(1.2738^k) \sum_{x=0}^k \binom{n-k}{x} \left(\frac{1}{1.2738}\right)^x \\
&= O^*(1.2738^k) \left(1 + \frac{1}{1.2738}\right)^{n-k} \\
&= O\left(1.7850^n \left(\frac{1.2738}{1.7850}\right)^k\right)
\end{aligned}$$

□

Now we are ready to prove the main theorem of this section.

Theorem 2. *There is an algorithm that, given an instance $(G = (V, E), T)$ of UNRESTRICTED NODE MULTIWAY CUT, runs in time $O(1.4766^n)$ and returns an optimum solution where $n = |V|$.*

Proof. Let (G, T) be the given instance of UNRESTRICTED NODE MULTIWAY CUT and $|T| = k$. Recall that we have described two different algorithms for UNRESTRICTED NODE MULTIWAY CUT. We now choose either of these algorithms based on the values of k and n . If $k \leq 0.5622n$, then we use the algorithm described in Lemma 1. In this case, the running time is upper bounded by $O^*(2^k) \leq O^*(2^{0.5622n}) \leq O(1.4766^n)$. If $k > 0.5622n$, then we use the algorithm described in Lemma 3. This algorithm runs in time $O\left(1.7850^n \left(\frac{1.2738}{1.7850}\right)^k\right)$ which is a decreasing function of k . Substituting $k = 0.5622n$, we get an upper bound on the running time as $O(1.4766^n)$. This completes the proof of the theorem. □

3.2 Node Multiway Cut

In this subsection, we give an exact algorithm for the NODE MULTIWAY CUT problem. We start with the following observation which follows from the fact that any solution to an instance of NODE MULTIWAY CUT is disjoint from the set of terminals in the instance.

Observation 3 *Let (G, T) be an instance of NODE MULTIWAY CUT. If T is not an independent set in G , then there is no solution to the instance (G, T) . Furthermore, if two terminals t_1 and t_2 have a common neighbor v , then v must be in every solution for the given instance.*

Due to Observation 3, we may assume that the terminal set is independent and the neighborhoods of the terminals in G are pairwise disjoint. This reduces

the restricted NODE MULTIWAY CUT to the following generalization of the UNRESTRICTED NODE MULTIWAY CUT, also known as the GROUP MULTIWAY CUT problem.

GROUP MULTIWAY CUT

Input: An undirected graph $G = (V, E)$ and pairwise disjoint sets of terminals $\{T_1, T_2, \dots, T_\ell\}$.

Question: Find a set $S \subseteq V(G)$ of minimum size such that $G \setminus S$ has no $u - v$ path for any $u \in T_i, v \in T_j$ and $i \neq j$.

In the following, we describe an exact algorithm for GROUP MULTIWAY CUT. The ideas and arguments that are similar to those used in the proof of Theorem 2. First we give a structural lemma about the vertices in T_i .

Lemma 4. *Let (G, T_1, \dots, T_ℓ) be an instance of GROUP MULTIWAY CUT. Let G' be the graph obtained by removing from G , the edges in $G[T_i]$ for every $i \in \{1, \dots, \ell\}$. Then, $S \subseteq V$ is a solution to the instance (G, T) if and only if it is also a solution to the instance (G', T') .*

Proof. Since G' is a subgraph of G , any solution for (G, T) is also a solution for (G', T') . We now consider the reverse direction. Let S be a solution for (G', T') . We claim that S is also a solution for (G, T) . Suppose that this is not the case, that is, there is a path P in G from $u_i \in T_i$ to $u_j \in T_j$ disjoint from S . Furthermore, we can assume without loss of generality that the internal vertices of P are disjoint from T_r for every $1 \leq r \leq \ell$. Observe that P is also present in the graph G' , and therefore S intersects P by definition, a contradiction. \square

Due to Lemma 4, henceforth we can assume that each T_i is an independent set in G . We have the following observation similar to Observation 1.

Observation 4 *Let (G, T_1, \dots, T_ℓ) be an instance of GROUP MULTIWAY CUT. Let S be an optimum solution to this instance and $T = \cup_{i=1}^{\ell} T_i$. Then, $|S| \leq |T|$.*

We have the following lemma which gives us an algorithm for the instances where k is “small”. The proof of this lemma is very similar to the proof of Lemma 1.

Lemma 5. *Let (G, T_1, \dots, T_ℓ) be an instance of GROUP MULTIWAY CUT, let $T = \cup_{i=1}^{\ell} T_i$ and let $|T| = k$. Then we can find an optimum solution to the given instance of GROUP MULTIWAY CUT in time $O^*(2^k)$.*

Proof. We first reduce this instance to an instance of NODE MULTIWAY CUT, by creating a new terminal t_i , for each set T_i , which is then made adjacent to all vertices in T_i . We then apply the FPT algorithm of Cygan et al [8] for NODE MULTIWAY CUT to obtain the required solution. This takes time $O^*(2^k)$. \square

Next, we describe another algorithm for GROUP MULTIWAY CUT. We will use this algorithm for those instances where k is “large”. We have the following lemma, whose proof is very similar to the proof of Lemma 2.

Lemma 6. *Let (G, T_1, \dots, T_ℓ) be an instance of GROUP MULTIWAY CUT where $G = (V, E)$ and let S be an optimum solution to this instance. Let $X = S \cap (V \setminus T)$ and $Y = S \setminus X$ where $T = \cup_{i=1}^\ell T_i$. Then, Y is a vertex cover of ℓ -torso($T_1, \dots, T_\ell, V \setminus X$) and conversely, for any vertex cover Y' for the graph ℓ -torso($T_1, \dots, T_\ell, V \setminus X$), the set $X \cup Y'$ is a solution to this instance.*

Proof. We first show that Y is indeed a vertex cover of $G' = \ell$ -torso($T_1, \dots, T_\ell, V \setminus X$). Let E' be the edge set of G' . Suppose that Y is not a vertex cover of G' and there is an edge $(t_i, t_j) \in E'$ which is not covered by Y . Observe that if $t_i \in T_{i'}$ and $t_j \in T_{j'}$ for $i' \neq j'$ then $(t_i, t_j) \notin E$, since this would contradict the assumption of S being a solution. Furthermore, if $i' = j'$, then $(t_i, t_j) \notin E$ since it contradicts our assumption that $T_{i'}$ is an independent set. Therefore, it must be the case that there is a path P between t_i and t_j in $G[V \setminus X]$ whose internal vertices are disjoint from T . Since this path is disjoint from both X and Y , it is also present in the graph $G \setminus S$, a contradiction. Hence, we conclude that Y is indeed a vertex cover of G' .

We now show that for any vertex cover Y' of G' , the set $X \cup Y'$ is a solution to the instance (G, T_1, \dots, T_ℓ) of GROUP MULTIWAY CUT. Suppose to the contrary that there is a vertex cover Y' of G' such that the set $S' = X \cup Y'$ is not a solution to the instance (G, T_1, \dots, T_ℓ) . That is, there is a t_i - t_j path in $G \setminus S'$ for some $t_i \in T_{i'}$ and $t_j \in T_{j'}$ where $i' \neq j'$. Observe that this implies the existence of a t_p - t_q path in $G \setminus S'$ for some $t_p \in T_{p'}$ and $t_q \in T_{q'}$ where $p' \neq q'$ such that the internal vertices of P are disjoint from $T \cup S'$. Therefore the edge (t_p, t_q) is not covered by Y' in G' , a contradiction. This completes the proof of the lemma. \square

Next, using Lemma 6 we can show the following lemma.

Lemma 7. *There is an algorithm that, given an instance $(G = (V, E), T_1, \dots, T_\ell)$ of GROUP MULTIWAY CUT, runs in time $O\left(1.7850^n \left(\frac{1.2738}{1.7850}\right)^k\right)$ and returns an optimum solution where $k = |\cup_{i=1}^\ell T_i|$ and $n = |V|$.*

Proof. Let $T = \cup_{i=1}^\ell T_i$. The description of the algorithm is as follows. For every $X \subseteq (V \setminus T)$ such that $|X| \leq k$, we construct the graph $G_X = \ell$ -torso($T, V \setminus X$) and compute a minimum vertex cover Y_X for G_X . We compute the minimum vertex cover by using the FPT algorithm of [4], which runs in time $O^*(1.2738^\ell)$ where ℓ is the size of an optimum vertex cover. Finally, we return the set $X \cup Y_X$ which is a smallest solution over all choices of X . The correctness of this algorithm follows from Lemma 6.

In order to bound the running time of this algorithm, first observe that for each X , the set Y_X has size at most $k - |X|$. Therefore, the FPT algorithm we use to compute a minimum vertex cover of torso($T, V \setminus X$) runs in time $O^*(1.2738^{k-|X|})$.

Summing over all choices of X , the time taken by our algorithm is,

$$\begin{aligned}
& \sum_{x=0}^k \binom{n-k}{x} O^*(1.2738^{k-x}) \\
&= O^*(1.2738^k) \sum_{x=0}^k \binom{n-k}{x} \left(\frac{1}{1.2738}\right)^x \\
&= O^*(1.2738^k) \left(1 + \frac{1}{1.2738}\right)^{n-k} \\
&= O\left(1.7850^n \left(\frac{1.2738}{1.7850}\right)^k\right)
\end{aligned}$$

This completes the proof of the lemma. \square

Combining the algorithms from Lemma 5 and Lemma 7, we can show the next theorem.

Theorem 5. *There is an algorithm that, given an instance $(G = (V, E), T_1, \dots, T_\ell)$ of GROUP MULTIWAY CUT, runs in time $O(1.4766^n)$ and returns an optimum solution, where $n = |V|$.*

Proof. Let (G, T_1, \dots, T_ℓ) be the given instance of GROUP MULTIWAY CUT and $T = \cup_{i=1}^\ell T_i$ and $|T| = k$. Recall that we have described two different algorithms for GROUP MULTIWAY CUT. We now choose either of these algorithms based on the values of k and n . If $k \leq 0.5622n$, then we use the algorithm described in Lemma 5. In this case, the running time is upper bounded by $O^*(2^k) \leq O(1.4766^n)$. If $k > 0.5622n$, then we use the algorithm described in Lemma 7. This algorithm runs in time $O\left(1.7850^n \left(\frac{1.2738}{1.7850}\right)^k\right)$ which is a decreasing function of k . Substituting $k = 0.5622n$, we get an upper bound on the running time as $O(1.4766^n)$. This completes the proof of the theorem. \square

The following theorem follows from Theorem 5 and Observation 3.

Theorem 6. *There is an algorithm that, given an instance $(G = (V, E), T)$ of NODE MULTIWAY CUT, runs in time $O(1.4766^n)$ and returns an optimum solution, where $n = |V|$.*

4 Directed Unrestricted Node Multiway Cut

In this section, we consider the DIRECTED UNRESTRICTED NODE MULTIWAY CUT problem.

DIRECTED UNRESTRICTED NODE MULTIWAY CUT

Input: A directed graph $D = (V, A)$ and a set of terminals $T = \{t_1, t_2, \dots, t_k\}$.

Question: Find a set $S \subseteq V$ of minimum size such that $G \setminus S$ has no $t_i \rightarrow t_j$ path for any $i \neq j$.

Since we consider the version where the terminals can be deleted, we have the following observation.

Observation 7 *Let (G, T) be an instance of DIRECTED UNRESTRICTED NODE MULTIWAY CUT and S be an optimum solution to this instance. Then, $|S| \leq |T|$.*

The proof of the next lemma is identical to the proof of Lemma 2 and therefore, we do not repeat it.

Lemma 8. *Let (D, T) be an instance of DIRECTED UNRESTRICTED NODE MULTIWAY CUT where $D = (V, A)$ and let S be an optimum solution to this instance. Let $X = S \cap (V \setminus T)$ and $Y = S \cap X$. Then Y is a vertex cover of the graph $\text{torso}(T, V \setminus X)$. Conversely if Y' is any vertex cover of the graph $\text{torso}(T, V \setminus X)$, then the set $X \cup Y'$ is a solution to this instance.*

Now we describe our algorithm for DIRECTED UNRESTRICTED NODE MULTIWAY CUT.

Theorem 8. DIRECTED UNRESTRICTED NODE MULTIWAY CUT *can be solved in time $O^*(1.6181^n)$.*

Proof. The description of the algorithm is as follows. For every $X \subseteq (V \setminus T)$ such that $|X| \leq k$, we construct the graph $D_X = \text{torso}(T, V \setminus X)$ and compute a minimum vertex cover Y_X for D_X . We compute the minimum vertex cover by using the FPT algorithm of Chen et al. [4], which runs in time $O^*(1.2738^\ell)$ where ℓ is the size of an optimum vertex cover. Finally, we return the set $X \cup Y_X$ which is a smallest solution over all choices of X . The correctness of this algorithm follows from Lemma 8.

Let \mathcal{T} be the running time of our algorithm. We have the following claim.

Claim 1. $\mathcal{T} = O(1.6181^n)$.

For every choice of X we run the FPT algorithm for vertex cover, which takes time $O^*(1.2738^{k-|X|})$. Therefore we have,

Proof.

$$\begin{aligned}
\mathcal{T} &= \sum_{x=0}^k \binom{n-k}{x} O^*(1.2738^{k-x}) \\
&= \sum_{x=0}^k \binom{n-k}{x} O^*(1.618^{k-x}) \\
&= O^*(1.618^k) \sum_{x=0}^k \binom{n-k}{x} \left(\frac{1}{1.618}\right)^x \\
&= O^*(1.618^k) \left(\frac{1}{1.618} + 1\right)^{n-k} \\
&= O^*(1.618^k) \times (1.6181)^{n-k} \\
&= O(1.6181^n)
\end{aligned}$$

□

This completes the proof of the theorem. \square

5 Subset Feedback Vertex Set

In this section we design an exact algorithm for SUBSET FEEDBACK VERTEX SET. We actually design two different algorithms for the problem, and then use these two algorithms to construct our final exact algorithm.

Let (G, T) be the given instance of SUBSET FEEDBACK VERTEX SET. Recall that we are allowed to pick terminal vertices into a solution. The following observation follows from the fact that the set of terminals itself is a solution.

Observation 9 *Let (G, T) be an instance of SUBSET FEEDBACK VERTEX SET, and let S be an optimum solution to this instance. Then $|S| \leq |T|$.*

- Lemma 9.** *1. There is an algorithm that, given an instance $(G = (V, E), T)$ of SUBSET FEEDBACK VERTEX SET, runs in time $O\left(1.2^n \left(\frac{5}{12}\right)^k\right)$ and returns an optimum solution where $k = |T|$ and $n = |V|$.*
- 2. There is an algorithm that, given an instance $(G = (V, E), T)$ of SUBSET FEEDBACK VERTEX SET, runs in time $O\left(2^n \left(\frac{1.7548}{2}\right)^k\right)$ and returns an optimum solution where $k = |T|$ and $n = |V|$.*

Proof. For every $X \subseteq (V \setminus T)$ such that $|X| \leq k$, let T_X be the set of terminals t such that, $G \setminus X$ contains a cycle passing through t which contains no other terminal vertex. Let G_X be the graph obtained from $G \setminus (X \cup T_X)$ by contracting every connected component of $G \setminus (T \cup X)$ ⁵. Let Y_X be a minimum feedback vertex set for G_X containing only vertices of T .

For the first algorithm, we compute Y_X in the following manner. We assign a weight of $k + 1$ to the vertices not in T and 1 to the vertices in T . We then use a FPT algorithm to compute a minimum feedback vertex set of G_X of weight at most k . We use the FPT algorithm of Chen et. al. [3] which runs in time $O^*(5^p)$, where p is the minimum weight of an feedback vertex set.

For the second algorithm we compute Y_X by computing a maximum induced forest of G_X which contains all the non-terminal vertices and taking its complement. Let F be the set of all non-terminal vertices in G_X and let $q = n - |F|$ be the number of terminal vertices in G_X . We use the algorithm of Fomin et al. [9](Section 3) on (G_X, F) and compute a maximum induced forest of G_X containing F . The arguments of Fomin et al. imply that the algorithm runs in time $O^*(1.7548^q)$.

Let $S_X = X \cup T_X \cup Y_X$. We compute S_X for every X and output the one with the smallest number of vertices as our solution.

⁵ We can compute both T_X and G_X in polynomial time

Correctness. The correctness of both the algorithm follow from the following claims.

Claim 2. Let S be an optimum solution to the given instance of SUBSET FEEDBACK VERTEX SET and let $X = S \cap (V \setminus T)$. Let T_X be the set of terminals t such that, $G \setminus X$ contains a cycle passing through t which contains no other terminal vertices. Then $T_X \subseteq S$.

Proof. Let $t \in T_X$ and C_t is a cycle passing through T in $G \setminus X$ which doesn't contain any other terminal vertex. If $t \notin S$, then S doesn't intersect C_t . This is a contradiction. Thus $T_X \subseteq S$. This completes the proof of this claim. \square

The above claim shows the correctness of adding T_X to the solution. The following lemma shows that once T_X is added to the solution, it suffices to compute a minimum feedback vertex set for the graph G_X .

Claim 3. Let S be an optimum solution to the given instance of SUBSET FEEDBACK VERTEX SET and let $X = S \setminus T$ and $Y = S \setminus X$. Furthermore, suppose that there are no cycles in G containing a unique vertex of T . Let G_X be obtained from $G \setminus X$ by contracting every connected component of $G \setminus (T \cup X)$. Then Y is a minimum feedback vertex set of G_X . Conversely if Y' is any feedback vertex set of G_X , then the set $X \cup Y'$ is a solution for the given instance of SUBSET FEEDBACK VERTEX SET.

Proof. We first show that for any $W \subseteq T$, there is a cycle in $G_X \setminus W$ if and only if there is a cycle in $G \setminus (X \cup W)$ which passes through $T \setminus W$. The claim then follows by setting $W = Y$ and $W = Y'$. First, observe that there is a unique connected component H_u in $(G \setminus (T \cup X))$ corresponding to each non-terminal vertex $u \in G_X$.

Now consider a cycle C in $G_X \setminus W$ and observe that all the terminal vertices in C lie in $T \setminus W$. We can replace each non-terminal vertex u in C with a path P_u in H_u to obtain a closed walk in G . However this closed walk is actually a cycle, since the paths $\{P_u\}$ are pairwise vertex disjoint. Thus we obtain a cycle in $G \setminus (X \cup W)$.

Conversely let C be a cycle in $G \setminus (X \cup W)$ that passes through a terminal $t \in T$. We assume that C visits each connected component of $G \setminus (T \cup X)$ at most once, as otherwise we can find a cycle C' which contains a subset of the terminals contained in C , which satisfies this property. By our assumption, we know that every terminal in $T \setminus (T_X \cup W)$ has at most one edge to a connected component in $G \setminus (T \cup X)$. Let P_u be a maximal subpath of C which is contained in the connected component H_u of $G \setminus (T \cup X)$. Consider the closed walk C' obtained from C by contracting the maximal subpath P_u to the vertex u , for every u . Observe that C' is actually a cycle since no vertex is repeated in C' , and C' is present in $G_X \setminus W$. This completes the proof of this claim. \square

Running Time. Let \mathcal{T}_1 be the running time of the first algorithm and, \mathcal{T}_2 be the running time of the second algorithms. The following two claims establish the running times of both the algorithms.

Claim 4. $\mathcal{T}_1 = O\left(1.2^n \times \left(\frac{5}{1.2}\right)^k\right)$.

Proof. Note that $|Y_X| \leq |S| - |X| - |T_X| \leq k - |X|$. Since for every choice of the set X , we look for an FVS of weight at most $k - |X|$ in G_X , we have that,

$$\begin{aligned} \mathcal{T}_1 &= \sum_{x=0}^k \binom{n-k}{x} O^*(5^{k-x}) \\ &= O^*(5^k) \sum_{x=0}^k \binom{n-k}{x} \left(\frac{1}{5}\right)^x \\ &= O^*(5^k) \times \left(1 + \frac{1}{5}\right)^{n-k} \\ &= O^*\left(5^k \times (1.2)^{n-k}\right) \\ &= O\left(1.2^n \times \left(\frac{5}{1.2}\right)^k\right) \end{aligned}$$

□

Claim 5. $\mathcal{T}_2 = O\left(2^n \times \left(\frac{1.7548}{2}\right)^k\right)$.

Proof. Observe the number of terminals is k , $\ell \leq k$. Therefore the exact algorithm runs in time $O^*(1.7548^k)$. Therefore we have,

$$\begin{aligned} \mathcal{T}_2 &= \sum_{x=0}^k \binom{n-k}{x} O^*(1.7548^k) \\ &= O^*(1.7548^k) \sum_{x=0}^k \binom{n-k}{x} \\ &= O^*(1.7548^k) \times 2^{n-k} \\ &= O\left(2^n \times \left(\frac{1.7548}{2}\right)^k\right) \end{aligned}$$

□

This completes the proof of the lemma. □

Theorem 10. *There is an algorithm that, given an instance $(G = (V, E), T)$ of SUBSET FEEDBACK VERTEX SET, runs in time $O(1.9161^n)$ and returns an optimum solution where $n = |V|$.*

Proof. Let (G, T) be the given instance of SUBSET FEEDBACK VERTEX SET, where G is a graph on n vertices and $|T| = k$. Based on the values of n and k we run one of the two algorithms described above.

If $k \leq 0.32789n$, then we run the first algorithm described in Lemma 9. The running time is upper bounded by $O\left(1.2^n \times \left(\frac{5}{1.2}\right)^{0.32789n}\right) = O(1.9161^n)$. Otherwise if $k > 0.32789n$, then we run the second algorithm described in Lemma 9. This algorithm runs in time $O\left(2^n \times \left(\frac{1.7548}{2}\right)^k\right)$ which is a decreasing function of k . Substituting $k = 0.32789n$, we get an upper bound of $O(1.9161^n)$ on the running time in this case as well. \square

We remark that, there are faster FPT [2, 7] and Exact algorithms [12] known for FVS on undirected graphs. If we use the fastest known deterministic and randomized algorithms, then we obtain the following theorem.

Theorem 11. *SUBSET FEEDBACK VERTEX SET can be solved in $O^*(1.8980^n)$ deterministic time, or in $O^*(1.8826^n)$ randomized time.*

5.1 Subset Feedback Vertex Set on Chordal Graphs

In this section we give an algorithm for SUBSET FEEDBACK VERTEX SET on chordal graphs which improves upon the previous best algorithm of Golovach et al. [15], and is much simpler. The main difference between this algorithm and the algorithm for SUBSET FEEDBACK VERTEX SET described earlier is that we use a polynomial time algorithm to solve weighted FEEDBACK VERTEX SET on chordal graphs ([6, 27]), instead of a FPT or an exact algorithm. It is well known (see also [1]) that chordal graphs are closed under vertex deletions and edge contractions.

We are now ready to prove the main theorem of this section:

Theorem 12. *There is an algorithm that, given an instance $(G = (V, E), T)$ of SUBSET FEEDBACK VERTEX SET on Chordal Graphs, returns an optimum solution in $O(1.6181^n)$ time, where $n = |V|$.*

Proof. The algorithm is the same as the two algorithms described in Lemma 9 except that we use the polynomial time algorithm for FEEDBACK VERTEX SET on chordal graphs instead of the FPT or the exact exponential algorithm. For every choice of X , we compute T_X and G_X in polynomial time. Observe that the graph G_X is obtained from G by vertex deletions and edge contractions, implying that G_X is also a chordal graph. Assign weight 1 to each terminal vertex and weight $k+1$ to each non-terminal vertex, and compute in polynomial time a minimum weight feedback vertex set Y_X of G_X using the result of Corneil and Fonlupt [6]. We now analyze the running time of our algorithm.

Let S be any optimum solution and let $X = S \setminus T$. Observe that $|X| \leq |S| \leq |T|$. The next lemma 10 shows that the number of choices of X is at most $O(1.6181^n)$.

Lemma 10. *Let V be a set of n elements and T is a subset of V . Then the number of distinct sets $X \subseteq V$ such that $S \cap T = \emptyset$ and $|X| \leq |T|$ is $O(1.6181^n)$.*

Proof. We look at the selection of the set X as a branching algorithm where in each branch we decide if an element in $V \setminus T$ is in X or not. The leaves (or equivalently the root to leaf paths) of the tree of this branching algorithm correspond to the choices of X . Let $x = |X|$ and $b = |V \setminus (T \cup X)|$. Since $x \leq |T|$ and $b + x = |V \setminus T| = n - |T|$, we have $2x + b \leq n$. We use $\mu = n - (2x + b)$ as the measure to analyze the branching algorithm. Initially $x = 0$ and $b = 0$, and so $\mu = n$. For an element $v \in (V \setminus T)$ we have two branches, either $v \in X$ or $v \notin X$. In the first branch x increases by 1 and b remains the same which implies μ decreases by 2. In the second branch b increases by 1 and x remains the same which implies μ decreases by 1. This corresponds to a branching vector [21] of $(2, 1)$, and this gives us a branching-tree with $O(1.6181^n)$ leaves, which completes the proof of the lemma. \square

Since after choosing X we do only a polynomial time computation, the running time of our algorithm is $O(1.6181^n)$. \square

We remark that we can use the above method to obtain faster exact algorithm for SUBSET FEEDBACK VERTEX SET on other graph classes, such as AT-free graphs [17], which are closed under vertex deletions and edge contractions, and FEEDBACK VERTEX SET is solvable in polynomial time on them.

6 Directed Subset Feedback Vertex Set

In this section we give an exact algorithm for the DIRECTED SUBSET FEEDBACK VERTEX SET problem running in time $O^*(1.9993^n)$. The problem is defined as follows.

DIRECTED SUBSET FEEDBACK VERTEX SET

Input: A directed graph $D = (V, A)$ and a set of terminal vertices T of size k .

Question: Find a minimum set of vertices in D which intersects every cycle in D which contains at least one vertex of T .

Next we observe the following property of directed graphs.

Observation 13 *Let $D = (V, A)$ be a directed graph. For any vertex $v \in V$, the following holds: v belongs to a closed walk in D if and only if v belongs to a cycle in D .*

Lemma 11. *Let $(D = (V, A), T)$ be an instance of DIRECTED SUBSET FEEDBACK VERTEX SET. Let S be an optimum solution to this instance and $X = S \setminus T$, $Y = S \setminus X$. Furthermore, suppose that every cycle in $D \setminus X$ that intersects T , contains at least two vertices of T . Then Y is a feedback vertex set in the graph torso($T, V \setminus X$) if and only if $X \cup Y$ is a subset feedback vertex set for the instance (D, T) .*

Proof. Suppose $X \cup Y$ is a solution in D where $Y \subseteq T$. If Y is not a feedback vertex set in $D_X = \text{torso}(T, V \setminus X)$, then there is a cycle C_X in $D_X \setminus Y$. From C_X in D_X we can obtain a closed walk C' in D in the following manner. We replace every edge (t_i, t_j) of C_X which is not present in A , with a path P_{ij} from t_i to t_j in $D \setminus X$ whose internal vertices lie in $V \setminus (T \cup X)$. Therefore we get a closed walk C' in $D \setminus (X \cup Y)$ which contains a terminal. By Observation 13, there is a cycle in D which passes through a terminal in D , which is not covered by $X \cup Y$. This is a contradiction.

Conversely, let Y be a feedback vertex set in D_X , but $X \cup Y$ is not a solution in D . Then there is a cycle C in $D \setminus (X \cup Y)$ and note that this cycle contains at least two vertices of T . Further assume that C is the shortest such cycle. Observe that every minimal subpath P_{ij} of C from terminals t_i to t_j whose internal vertices lie in $V \setminus T$, implies an edge (t_i, t_j) in D_X . Therefore we can obtain a cycle C' in D_X from C by replacing the subpath P_{ij} with the edge (t_i, t_j) , for every pair of terminals t_i, t_j in C . Observe that this cycle is not covered by Y . This is a contradiction.

This completes the proof of the lemma. \square

The following observation is immediate since the set T forms a potential solution.

Observation 14 *Let (D, T) be an instance of DIRECTED SUBSET FEEDBACK VERTEX SET and let S be an optimum solution for this instance. Then, $|S| \leq |T|$.*

We are now ready to prove the main theorem of this section.

Theorem 15. *There is an algorithm that, given an instance $(D = (V, A), T)$ of DIRECTED SUBSET FEEDBACK VERTEX SET, runs in time $O(1.9993^n)$ and returns an optimum solution where $n = |V|$.*

Proof. The description of the algorithm is as follows. For every $X \subseteq (V \setminus T)$ such that $|X| \leq k$, we first compute (in polynomial time) the set T_X which is the set of terminals $t \in T$ such that there is a directed cycle in the graph $D[(V \setminus (T \cup X)) \cup \{t\}]$. In polynomial time, construct the graph $D_X = \text{torso}(T, V \setminus (X \cup T_X))$. Then we compute a minimum feedback vertex set Y_X for D_X by using the exact exponential algorithm by Razgon [25] for DIRECTED FEEDBACK VERTEX SET, which runs in time $O(1.9977^\ell)$ where ℓ is the number of vertices in the input graph. In this case we have $\ell \leq k$. Finally, we return the set $X \cup T_X \cup Y_X$ which is a smallest solution over all choices of X . The correctness of the algorithm follows from Lemma 11.

Running Time. Let \mathcal{T} be the running time of our algorithm. Therefore $\mathcal{T} = \sum_{x=0}^k \binom{n-k}{x} O^*(1.9977^k)$. To compute an upper bound on \mathcal{T} , we need to examine the values of n and k . The following claim analyzes the running time of our algorithm, and completes the proof of the theorem.

Claim 6. $\mathcal{T} = O(1.9993^n)$.

Proof. Based on the values taken by n and k we consider the following three cases,

1. $k \leq \frac{n-k}{2.1}$.

To address this case, we require the following lemma from [22](Lemma 2.1).

Lemma 12. *Let $0 < \alpha < 1$. Then*

$$\binom{n}{\alpha n} = O^*\left(\left(\frac{1}{\alpha^\alpha(1-\alpha)^{1-\alpha}}\right)^n\right).$$

By Lemma 12 for every $0 \leq x \leq k$ we have

$$\binom{n-k}{x} \leq \binom{n-k}{k} \leq \binom{n-k}{\frac{n-k}{2.1}} = O^*(\gamma^{n-k}), \text{ where } \gamma = 1.9978, \text{ implying}$$

that $\mathcal{T} = O^*(1.9978^n)$.

2. $\frac{n-k}{2.1} < k \leq n-k$.

Note that $k \geq \frac{n-k}{2.1}$ which implies $k \geq \frac{n}{3.1}$. And since $k \leq n-k$, $\sum_{x=0}^k \binom{n-k}{x} \leq 2^{n-k}$.

Hence $\mathcal{T} \leq O^*(1.9977^k)2^{n-k} = O\left(2^n \left(\frac{1.9977}{2}\right)^k\right) \leq O\left(2^n \left(\frac{1.9977}{2}\right)^{\frac{n}{3.1}}\right)$ since $k \geq \frac{n}{3.1}$.

This simplifies to $\mathcal{T} \leq O(1.9993^n)$.

3. $k > n-k$.

We have $k > n-k$ which gives $k > \frac{n}{2}$.

Similar to the previous case, we have $\mathcal{T} \leq O^*(1.9977^k)2^{n-k} = O\left(2^n \left(\frac{1.9977}{2}\right)^k\right) \leq O\left(2^n \left(\frac{1.9977}{2}\right)^{\frac{n}{2}}\right) = O(1.9989^n)$.

Combining all the above three cases, we see that $\mathcal{T} \leq O(1.9993^n)$ for every $0 \leq k \leq n$. □

This completes the proof of the theorem. □

7 Conclusion

We introduced a methodology of obtaining non-trivial exact exponential algorithms for several *terminal set problems*. We conclude with open problems which seems to be evasive to our approach. Designing an algorithm faster than $O^*(2^n)$ for DIRECTED NODE MULTIWAY CUT remains an interesting question. Another interesting problem is SUBSET ODD CYCLE TRANSVERSAL, where the task is to find a vertex subset of minimum size hitting all cycles of odd length containing at least one terminal. Again, the problem is trivially solvable in $O^*(2^n)$ but no faster algorithm for this problem is known. We conclude by remarking that an approach based on our methodology might result in such an algorithm since ODD CYCLE TRANSVERSAL is solvable in time $O^*(1.4661^n)$ [20, 26]. Finally designing an algorithm for MULTICUT on both undirected and directed graphs, faster than the trivial $O^*(2^n)$ algorithm, remains an interesting open problem.

References

1. Belmonte, R., Golovach, P.A., Heggernes, P., vant Hof, P., Kamiński, M., Paulusma, D.: Finding contractions and induced minors in chordal graphs via disjoint paths. In: Algorithms and Computation, pp. 110–119. Springer (2011)
2. Cao, Y., Chen, J., Liu, Y.: On feedback vertex set new measure and new structures. In: Algorithm Theory-SWAT 2010, pp. 93–104. Springer (2010)
3. Chen, J., Fomin, F.V., Liu, Y., Lu, S., Villanger, Y.: Improved algorithms for feedback vertex set problems. *J. Comput. Syst. Sci.* 74(7), 1188–1198 (2008)
4. Chen, J., Kanj, I.A., Xia, G.: Improved upper bounds for vertex cover. *Theoretical Computer Science* 411(40), 3736–3756 (2010)
5. Chen, J., Liu, Y., Lu, S.: An Improved Parameterized Algorithm for the Minimum Node Multiway Cut Problem. *Algorithmica* 55(1), 1–13 (2009)
6. Corneil, D., Fonlupt, J.: The complexity of generalized clique covering. *Discrete Applied Mathematics* 22(2), 109 – 118 (1988–1989)
7. Cygan, M., Nederlof, J., Pilipczuk, M., Pilipczuk, M., van Rooij, J., Wojtaszczyk, J.: Solving connectivity problems parameterized by treewidth in single exponential time. In: Foundations of Computer Science (FOCS), 2011 IEEE 52nd Annual Symposium on. pp. 150–159. IEEE (2011)
8. Cygan, M., Pilipczuk, M., Pilipczuk, M., Wojtaszczyk, J.O.: On multiway cut parameterized above lower bounds. In: Proceedings of the 6th International conference on Parameterized and Exact Computation. pp. 1–12. Springer-Verlag (2011)
9. Fomin, F.V., Gaspers, S., Pyatkin, A.V., Razgon, I.: On the minimum feedback vertex set problem: Exact and enumeration algorithms. *Algorithmica* 52(2), 293–307 (2008)
10. Fomin, F.V., Heggernes, P., Kratsch, D., Papadopoulos, C., Villanger, Y.: Enumerating minimal subset feedback vertex sets. In: WADS. vol. 6844, pp. 399–410 (2011)
11. Fomin, F.V., Kratsch, D.: Exact Exponential Algorithms. Springer-Verlag New York, Inc., New York, NY, USA, 1st edn. (2010)
12. Fomin, F.V., Villanger, Y.: Finding induced subgraphs via minimal triangulations. In: 27th International Symposium on Theoretical Aspects of Computer Science (STACS). vol. 5, pp. 383–394. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik (2010)
13. Fomin, F.V., Villanger, Y.: Finding induced subgraphs via minimal triangulations. In: STACS. vol. 5, pp. 383–394 (2010)
14. Garg, N., Vazirani, V., Yannakakis, M.: Multiway Cuts in Directed and Node Weighted Graphs. In: ICALP (1994), pages 487–498
15. Golovach, P.A., Heggernes, P., Kratsch, D., Saei, R.: An exact algorithm for subset feedback vertex set on chordal graphs. In: Parameterized and Exact Computation, pp. 85–96. Springer (2012)
16. Gupta, S., Raman, V., Saurabh, S.: Maximum r -regular induced subgraph problem: Fast exponential algorithms and combinatorial bounds. *SIAM J. Discrete Math.* 26(4), 1758–1780 (2012)
17. Kratsch, D., Müller, H., Todinca, I.: Feedback vertex set on at -free graphs. *Discrete Applied Mathematics* 156(10), 1936–1947 (2008)
18. Mader, W.: Über die Maximalzahl kreuzungsfreier H -Wege. *Arch. Math. (Basel)* 31(4), 387–402 (1978/79), <http://dx.doi.org/10.1007/BF01226465>
19. Marx, D.: Parameterized Graph Separation Problems. *Theor. Comput. Sci.* 351(3), 394–406 (2006)

20. Mishra, S., Raman, V., Saurabh, S., Sikdar, S.: König deletion sets and vertex covers above the matching size. In: ISAAC. vol. 5369, pp. 836–847 (2008)
21. Niedermeier, R.: Invitation to fixed-parameter algorithms (2002)
22. Pilipczuk, M.: New techniques applicable to selected NP-hard problems. PhD dissertation, University of Warsaw (2011)
23. Pilipczuk, M., Pilipczuk, M.: Finding a maximum induced degenerate subgraph faster than 2^n . In: IPEC. vol. 7535, pp. 3–12 (2012)
24. Razgon, I.: Exact computation of maximum induced forest. In: SWAT. vol. 4059, pp. 160–171 (2006)
25. Razgon, I.: Computing Minimum Directed Feedback Vertex Set in $O^*(1.9977^n)$. In: ICTCS. pp. 70–81 (2007)
26. Robson, J.M.: Algorithms for maximum independent sets. *J. Algorithms* 7(3), 425–440 (1986)
27. Yannakakis, M., Gavril, F.: The maximum k -colorable subgraph problem for chordal graphs. *Information Processing Letters* 24(2), 133–137 (1987)