Breaking the Barrier 2^k for Subset Feedback Vertex Set in Chordal Graphs

Tian Bai
School of Computer Science and Engineering, University of Electronic Science and Technology of China, Chengdu, China

Mingyu Xiao
School of Computer Science and Engineering, University of Electronic Science and Technology of China, Chengdu, China

Abstract

The Subset Feedback Vertex Set problem (SFVS) is to delete k vertices from a given graph such that in the remaining graph, any vertex in a subset T of vertices (called a terminal set) is not in a cycle. The famous Feedback Vertex Set problem is the special case of SFVS with T being the whole set of vertices. In this paper, we study exact algorithms for SFVS in Split Graphs (SFVS-S) and SFVS in Chordal Graphs (SFVS-C). SFVS-S generalizes the minimum vertex cover problem and the prize-collecting version of the maximum independent set problem in hypergraphs (PCMIS), and SFVS-C further generalizes SFVS-S. Both SFVS-S and SFVS-C are implicit 3-Hitting Set problems. However, it is not easy to solve them faster than 3-Hitting Set. In 2019, Philip, Rajan, Saurabh, and Tale (Algorithmica 2019) proved that SFVS-C can be solved in $\mathcal{O}^*(2^k)$ time, slightly improving the best result $\mathcal{O}^*(2.0755^k)$ for 3-Hitting Set. In this paper, we break the "2^k-barrier" for SFVS-S and SFVS-C by introducing an $\mathcal{O}^*(1.8192^k)$-time algorithm. This achievement also indicates that PCMIS can be solved in $\mathcal{O}^*(1.8192^n)$ time, marking the first exact algorithm for PCMIS that outperforms the trivial $\mathcal{O}^*(2^n)$ threshold. Our algorithm uses reduction and branching rules based on the Dulmage-Mendelsohn decomposition and a divide-and-conquer method.

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

Keywords and phrases Subset Feedback Vertex Set, Prize-Collecting Maximum Independent Set, Parameterized Algorithms, Split Graphs, Chordal Graphs, Dulmage-Mendelsohn Decomposition

Digital Object Identifier 10.4230/LIPIcs.MFCS.2024.15

Related Version *Full Version*: `https://arxiv.org/abs/2212.04726`

Funding *Mingyu Xiao:* Mingyu Xiao acknowledges the National Science Foundation of China under Grant No. 62372095.

1 Introduction

The Feedback Vertex Set problem (FVS), one of Karp's 21 NP-complete problems [32], is a fundamental problem in graph algorithms. Given a graph G with n vertices and a parameter k, FVS is to decide whether there is a subset of vertices of size at most k whose deletion makes the remaining graph acyclic. FVS arises in a variety of applications in various fields such as circuit testing, network communications, deadlock resolution, artificial intelligence, and computational biology [6, 11, 29]. Because of the importance of FVS, different variants and generalizations have been extensively studied in the literature. The Subset Feedback Vertex Set problem (SFVS), introduced by Even et al. [17] in 2000, is a famous case. In SFVS, we are further given a vertex subset $T \subseteq V$ called *terminal set*, and we are asked to determine whether there is a set of vertices of size at most k whose removal makes each terminal in T not contained in any cycle in the remaining graph. When the terminal set is the whole vertex set of the graph, SFVS becomes FVS. SFVS also generalizes another famous

49th International Symposium on Mathematical Foundations of Computer Science (MFCS 2024).
Editors: Rastislav Královič and Antonín Kučera; Article No. 15; pp. 15:1–15:18
Leibniz International Proceedings in Informatics
LIPICS Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

problem, i.e., Node Multiway Cut. Whether SFVS is FPT had been once a well-known open problem [12]. Until 2013, Cygan et al. [11] proved the fixed-parameter tractability of SFVS by giving an algorithm with running time $\mathcal{O}^*(2^{\mathcal{O}(k \log k)})$. Recently, Iwata et al. [30, 31] showed the first single-exponential algorithm with running time $\mathcal{O}^*(4^k)$ for SFVS. In 2018, Hols and Kratsch showed that SFVS has a randomized polynomial kernelization with $\mathcal{O}(k^9)$ vertices [25]. Besides, FVS admits a quadratic kernel [28, 40], whereas whether there is a deterministic polynomial kernel for SFVS is still unknown.

SFVS has also been studied in several graph classes [35, 37, 2, 3, 4], such as interval graphs, permutation graphs, chordal graphs, and split graphs. SFVS remains NP-complete even in split graphs [18], while FVS in split and chordal graphs are polynomial-time solvable [44]. It turns out that both SFVS in Split Graphs (SFVS-S) and SFVS in Chordal Graphs (SFVS-C) can be regarded as implicit 3-Hitting Set. Its importance stems from the fact that 3-Hitting Set can be used to recast a wide range of problems, and now it can be solved in time $\mathcal{O}^*(2.0755^k)$ [42]. On the other hand, when we formulate SFVS-S or SFVS-C in terms of 3-Hitting Set, the structural properties of the input graph are lost. We believe these structural properties can potentially be exploited to obtain faster algorithms for the original problems. However, designing a faster algorithm for SFVS-S and SFVS-C seems challenging. Only recently did Philip et al. [37] improve the running bound to $\mathcal{O}^*(2^k)$, where they needed to consider many cases of the clique-tree structures of the chordal graphs. In some cases, they needed to branch into seven branches. Note that 2^k is another barrier frequently considered in algorithm design and analysis. Some preliminary brute force algorithms, dynamic programming, and advanced techniques, such as inclusion-exclusion, iterative compression, and subset convolution, always lead to the bound 2^k. Breaking the "2^k-barrier" becomes an interesting question for many problems.

We highlight that SFVS-S and SFVS-C are important since they generalize a natural variation of the maximum independent set problem called Prize-Collecting Maximum Independent Set in hypergraphs (PCMIS). In PCMIS, we are given a hypergraph H with n vertices. The object is to find a vertex subset S maximizing the size of S minus the number of hyperedges in H that contain at least two vertices from S. In other words, we may balance the size of the vertex subset against the number of hyperedges on which S violates the independent constraints. The prize-collecting version of many important fundamental problems has drawn certain attention recently, such as Prize-Collecting Steiner Tree [36], Prize-Collecting Network Activation [21], and Prize-Collecting Travelling Salesman Problem [5]. To the best of our knowledge, no exact algorithm for PCMIS faster than $\mathcal{O}^*(2^n)$ is known before.

Fomin et al. [19] showed that Cluster Vertex Deletion and Directed FVS in Tournaments admit subquadratic kernels with $\mathcal{O}(k^{5/3})$ vertices and $\mathcal{O}(k^{3/2})$ vertices, respectively; while the size of the best kernel for SFVS-C is still quadratic, which can be easily obtained from the kernelization for 3-Hitting Set [1]. As for parameterized algorithms, Dom et al. [14] first designed an $\mathcal{O}^*(2^k)$-time algorithm for Directed FVS in Tournaments, breaking the barrier of 3-Hitting Set, and the running time bound of which was later improved to $\mathcal{O}^*(1.6191^k)$ by Kumar and Lokshtanov [33]. For Cluster Vertex Deletion, in 2010, Hüffner et al. [27] first broke the barrier of 3-Hitting Set by obtaining an $\mathcal{O}^*(2^k)$-time algorithm. Now it can be solved in $\mathcal{O}^*(1.7549^k)$ time [41].

Contributions and Techniques

In this paper, we contribute to parameterized algorithms for SFVS-S and SFVS-C. Our main contributions are summarized as follows.

1. We firstly break the "2^k-barrier" for SFVS-S and SFVS-C by giving an $\mathcal{O}^*(1.8192^k)$-time algorithm, which significantly improves previous algorithms.
2. We show that an $\mathcal{O}^*(\alpha^k)$-time algorithm ($\alpha > 1$) for SFVS-S leads to an $\mathcal{O}^*(\alpha^n)$-algorithm for PCMIS. Thus, we can solve PCMIS in time $\mathcal{O}^*(1.8192^n)$, also breaking the "2^n-barrier" for this problem for the first time.
3. We make use of the Dulmage-Mendelsohn decomposition of bipartite graphs to catch structural properties, and then we are able to use a new measure μ to analyze the running time bound. This is the most crucial technique for us to obtain a significant improvement. Note that direct analysis based on the original measure k has encountered bottlenecks. Any tiny improvement may need complicated case-analysis.
4. The technique based on Dulmage-Mendelsohn decomposition can only solve SFVS-S. We also propose a divide-and-conquer method by dividing the instance of SFVS-C into instances of SFVS-S. We show that SFVS-C can be solved in time $\mathcal{O}^*(\alpha^k + 1.6191^k)$ if SFVS-S can be solved in time $\mathcal{O}^*(\alpha^k)$.

2 Preliminaries

2.1 Graphs

Let $G = (V, E)$ stand for an undirected graph with a set V of vertices and a set E of edges. We adopt the convention that $n = |V|$ and $m = |E|$. When a graph G' is mentioned without specifying its vertex and edge sets, we use $V(G')$ and $E(G')$ to denote these sets, respectively. For a subset $X \subseteq V$ of vertices, we define the following notations. The *neighbour set* of X, denoted by $N_G(X)$, is the set of all vertices in $V \backslash X$ that are adjacent to a vertex in X, and the *closed neighbour set* of X is expressed as $N_G[X] \coloneqq N_G(X) \cup X$. The subgraph of G induced by X is denoted by $G[X]$. We simply write $G - X \coloneqq G[V \backslash X]$ as the subgraph obtained from G removing X together with edges incident on any vertex in X. For ease of notation, we may denote a singleton set $\{ v \}$ by v.

The *degree* of v in G is defined by $\deg_G(v) \coloneqq |N_G(v)|$. An edge e is a *bridge* if it is not contained in any cycle of G. A *separator* of a graph is a vertex set such that its deletion increases the number of connected components of the graph. The shorthand $[r]$ is expressed as the set $\{ 1, 2, \ldots, r \}$ for $r \in \mathbb{N}^+$.

In an undirected graph $G = (V, E)$, a set $X \subseteq V$ is a *clique* if every pair of distinct vertices u and v in X are connected by an edge $uv \in E$; X is an *independent set* if $uv \notin E$ for every pair of vertices u and v in X; X is a *vertex cover* if for any edge $uv \in E$ at least one of u and v is in X. A subset $S \subseteq V$ is a vertex cover of G if and only if $V \backslash S$ is an independent set. A vertex v is called *simplicial* in G if $N_G[v]$ is a clique [13]. A clique in G is *simplicial* if it is maximal and contains at least one simplicial vertex. A *matching* is a set of edges without common vertices.

2.2 Chordal Graphs and Split Graphs

A *chord* of a cycle is an edge that connects two non-consecutive vertices of the cycle. A graph G is said to be *chordal* if every cycle of length at least 4 contains a chord. A chordal graph G holds the following properties that will be used in the paper: Every induced subgraph of a chordal graph G is chordal, and every minimal separator of G is a clique [13].

Consider a connected chordal graph G, and let $\mathcal{Q}_G$ denote the set of all maximal cliques in G. A *clique graph* of G is an undirected graph $(\mathcal{Q}_G, \mathcal{E}_G, \sigma)$ with the edge-weighted function $\sigma \colon \mathcal{E}_G \to \mathbb{N}$ satisfying that an edge $Q_1Q_2 \in \mathcal{E}_G$ if $Q_1 \cap Q_2$ is a minimal separator and

$\sigma(Q_1Q_2) \coloneqq |Q_1 \cap Q_2|$. A *clique tree* $\mathcal{T}_G$ of G is a maximum spanning tree of the clique graph of G, and the following facts hold [7, 23, 43]: (1) Each leaf node of a clique tree $\mathcal{T}_G$ is a simplicial clique in G; (2) For a pair of maximal cliques Q_1 and Q_2 such that $Q_1Q_2 \in \mathcal{E}_G$, $Q_1 \cap Q_2$ separates each pair of vertices $v_1 \in Q_1 \backslash Q_2$ and $v_2 \in Q_2 \backslash Q_1$.

Whether a graph is chordal can be checked in linear time $\mathcal{O}(n + m)$ [38]. The number of maximal cliques in a chordal graph G is at most n [22], and all of them can be listed in linear time $\mathcal{O}(n + m)$ [23]. These properties will be used in our algorithm.

A graph is a *split graph* if its vertex set can be partitioned into a clique K and an independent set I [39]. Such a partition (I, K) is called a *split partition*. It is worth noting that every split graph is chordal, and whether a graph is a split graph can also be checked in linear time $\mathcal{O}(n + m)$ by definition.

2.3 Subset Feedback Vertex Set in Split and Chordal Graphs

Given a terminal set $T \subseteq V$ of an undirected graph $G = (V, E)$, a cycle in G is a T*-cycle* if it contains a terminal from T, and a T*-triangle* is specifically a T-cycle of length three. A *subset feedback vertex set* of a graph G with a terminal set T is a subset of V whose removal makes G contain no T-cycle.

In this study, we focus on SFVS in split and chordal graphs. The problem takes as input a chordal graph $G = (V, E)$, a terminal set $T \subseteq V$, and an integer k. The task is to determine whether there is a subset feedback vertex set S of size at most k. Moreover, the following lemma shows that the problem can be transformed into the problem of finding a subset of vertices intersecting all T-triangles instead of all T-cycles.

▶ **Lemma 1** ([37]). *Let $G = (V, E)$ be a chordal graph and $T \subseteq V$ be the terminal set. A vertex set $S \subseteq V$ is a subset feedback vertex set of G if and only if $G - S$ contains no T-triangles.*

For the sake of presentation, this paper considers a slight generalization of SFVS-C. In this generalized version, a set of marked edges $M \subseteq E$ is further given, and we are asked to decide whether there is a subset feedback vertex set of size at most k, which also covers all marked edges, i.e., each marked edge must have at least one of its endpoints included in the set. This set is called a *solution* to the given instance. Among all solutions, a *minimum solution* is the one with the smallest size. The size of a minimum solution to an instance $\mathcal{I}$ is denoted by $s(\mathcal{I})$. Formally, the generalization of SFVS-C is defined as follows.

(Generalized) SFVS-C
Input: A chordal graph $G = (V, E)$, a terminal set $T \subseteq V$, a marked edge set $M \subseteq E$, and an integer k.
Output: Determine whether there is a subset of vertices $S \subseteq V$ of size at most k, such that neither edges in M nor T-cycles exist in $G - S$.

We have the following simple observations. Let abc be a T-triangle with a degree-2 vertex b in the graph. Any solution must contain at least one of the vertices a, b, and c. If vertex b is included in the solution, we can replace it with either a or c without affecting the solution's feasibility. Consequently, we can simplify the graph by removing b and marking edge ac. This observation motivates the consideration of the generalized version.

We will simply use SFVS-C to denote the generalized version. When the input graphs are restricted to split graphs, the problem becomes SFVS-S. An instance of our problem is denoted by $\mathcal{I} = (G, T, M, k)$. During our algorithm, it may be necessary to consider some sub-instances where the graph is a subgraph of G. We define the *instance induced by $X \subseteq V$ or $G[X]$* as $(G[X], T \cap X, M \cap E(G[X]), k)$.

In this paper, our algorithms follow a standard branch-and-reduce paradigm. An operation on the input instance, such as the reduction rule, is *safe* if the input instance is a Yes-instance if and only if the output instance is a Yes-instance. A branching operation is *safe* if the input instance is a Yes-instance if and only if at least one of the resulting sub-instances is a Yes-instance. Additionally, we use branching vectors and branching factors in our analysis. The definitions of these standard concepts can be found in [10].

3 The Dulmage-Mendelsohn Decomposition and Reduction

This section introduces the Dulmage-Mendelsohn decomposition of a bipartite graph [15, 16]. The Dulmage-Mendelsohn decomposition will play a crucial role in our algorithm for SFVS-S.

▶ **Definition 2** (Dulmage-Mendelsohn Decomposition [34, 9]). *Let F be a bipartite graph with bipartition $V(F) = A \cup B$. The Dulmage-Mendelsohn decomposition (cf. Fig. 1) of F is a partitioning of $V(F)$ into three disjoint parts C, H and R, such that*

1. *C is an independent set and $H = N_F(C)$;*
2. *$F[R]$ has a perfect matching;*
3. *H is the intersection of all minimum vertex covers of F; and*
4. *any maximum matching in F includes all vertices in $R \cup H$.*

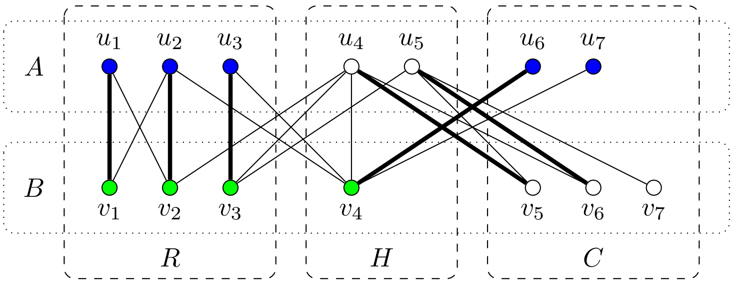

Figure 1 A bipartite graph F with bipartition $V(F) = A \cup B$, where $A = \{u_i\}_{i=1}^{7}$ and $B = \{v_i\}_{i=1}^{7}$. The thick edges form a maximum matching of F. The Dulmage-Mendelsohn decomposition of F is (C, H, R) with $C = \{u_6, u_7, v_5, v_6, v_7\}$, $H = \{u_4, u_5, v_4\}$, and $R = \{u_1, u_2, u_3, v_1, v_2, v_3\}$. If F is an auxiliary subgraph of an instance of SFVS-S, then $\hat{A} = \{u_1, u_2, u_3, u_6, u_7\}$ (denoted by blue vertices) and $\hat{B} = \{v_1, v_2, v_3, v_4\}$ (denoted by green vertices).

The Dulmage-Mendelsohn decomposition always exists and is unique [34], which can be computed in time $\mathcal{O}(m\sqrt{n})$ by finding the maximum matching of the graph F [26]. Leveraging this decomposition, we propose a crucial reduction rule for SFVS-S.

Consider an instance $\mathcal{I} = (G = (V, E), T, M, k)$ of SFVS-S. Let (I, K) be a split partition of G, where I is an independent set and K is a clique. Based on the split partition (I, K) of G, we can uniquely construct an auxiliary bipartite subgraph F with bipartition $V(F) = A \cup B$. In subgraph F, partition A is the subset of the vertices in I that are only incident to marked edges and $B = N_G(A)$. In addition, $E(F)$ is the set of all edges between A and B, i.e., $E(F) \coloneqq \{ab \in E : a \in A,\ b \in B\}$. Notice that F contains no isolated vertex, and all edges in F are marked by the definitions of A and B.

Let (R, H, C) denote the Dulmage-Mendelsohn decomposition of the auxiliary subgraph F. Define $\hat{A} \coloneqq A \cap (R \cup C)$ and $\hat{B} \coloneqq B \cap (R \cup H)$ (see Fig. 1). We have $\hat{B} = N_G(\hat{A})$, and there exists a matching saturating all vertices in $\hat{B}$. This indicates that every solution contains at least $|\hat{B}|$ vertices in $\hat{A} \cup \hat{B}$. On the other hand, $\hat{B}$ is a minimum vertex cover of the subgraph induced by $\hat{A} \cup \hat{B}$. Consequently, there exists a minimum solution to $\mathcal{I}$ containing $\hat{B}$. Next, we introduce the reduction rule, which is called the *DM Reduction*.

▶ **Reduction Rule** (DM Reduction). *Let F be the auxiliary subgraph with bipartition $V(F) = A \cup B$, and let (R, H, C) denote the Dulmage-Mendelsohn decomposition of F. If $\hat{A}$ and $\hat{B}$ are non-empty, delete $\hat{A}$ and $\hat{B}$ from the graph G and decrease k by $|\hat{B}| = |R|/2 + |H \cap B|$.*

▶ **Lemma 3.** *The DM Reduction is safe.*

Proof. Recall that $\hat{A} \coloneqq A \cap (R \cup C)$ and $\hat{B} \coloneqq B \cap (R \cup H)$. According to the definition of the Dulmage-Mendelsohn decomposition, we know that $N_F(\hat{A}) = \hat{B}$, and $\hat{B}$ is a minimum vertex cover of the subgraph induced by $\hat{A} \cup \hat{B}$. For a solution S to the input instance $\mathcal{I}$, the size of $S \cap (\hat{A} \cup \hat{B})$ is no less than $|\hat{B}|$ since S covers every edge in M. Let $S' = (S \backslash \hat{A}) \cup \hat{B}$. Observe that $|\hat{A}| > |\hat{B}|$; otherwise, A would be a minimum vertex cover of F, contradicting that H is a subset of any minimum vertex cover. Consequently, we derive that $|S'| \leqslant |S|$. In addition, we can see that S' is also a solution, leading to the safeness of the DM Reduction. ◀

▶ **Lemma 4.** *Given an instance $\mathcal{I} = (G, T, M, k)$ of SFVS-S, let F be the auxiliary subgraph of G with bipartition $V(F) = A \cup B$. If the DM Reduction cannot be applied, for any non-empty subset $A' \subseteq A$, it holds that $|A'| < |N_G(A')|$.*

Proof. If the DM Reduction cannot be applied, the Dulmage-Mendelsohn decomposition of F must be $(R, H, C) = (\varnothing, A, B)$. According to the definition of the Dulmage-Mendelsohn decomposition, A is a vertex cover, and $H = A$ is the intersection of all minimum vertex covers of F. As a result, we know that A is the unique minimum vertex cover of the auxiliary subgraph F. We assume to the contrary that there exists a subset $A' \subseteq A$ such that $|A'| \geqslant |N_G(A')|$. Then we immediately know that $(A \backslash A') \cup N_G(A')$ is a minimum vertex cover distinct from A, leading to a contradiction. ◀

▶ **Lemma 5.** *Given an instance $\mathcal{I} = (G, T, M, k)$ of SFVS-S, let F be the auxiliary subgraph of G with bipartition $V(F) = A \cup B$. If the DM Reduction cannot be applied and $k < |A|$, the instance $\mathcal{I}$ is a No-instance.*

Proof. The size of the solution to $\mathcal{I} = (G, T, M, k)$ is no less than the size of the minimum vertex cover of F since all marked edges need to be covered. If the DM Reduction cannot be applied, Lemma 4 implies that A is the minimum vertex cover of F. Consequently, the size of the minimum solution to $\mathcal{I}$ must be no less than $|A|$, which means that an instance $\mathcal{I}$ is a No-instance if $k < |A|$. ◀

4 Algorithms for SFVS in Split and Chordal Graphs

This section mainly presents an algorithm for SFVS-S. This algorithm plays a critical role in the algorithm for SFVS-C.

4.1 Good Instances

We begin by introducing a special instance of SFVS-S, which we refer to as a *good instance*. We show that solving good instances is as hard as solving normal instances of SFVS-S in some sense.

▶ **Definition 6** (Good Instances). *An instance $\mathcal{I} = (G = (V, E), T, M, k)$ of SFVS-S is called* good *if it satisfies the following properties:*

(i) *$(T, V \backslash T)$ is the split partition, where terminal set T is the independent set and $V \backslash T$ forms the clique;*
(ii) *every marked edge connects one terminal and one non-terminal; and*
(iii) *the DM reduction cannot be applied on the auxiliary subgraph determined by $(T, V \backslash T)$.*

▶ **Lemma 7.** *For any constant $\alpha > 1$, SFVS-S can be solved in time $\mathcal{O}^*(\alpha^k)$ if and only if SFVS-S on good instances can be solved in time $\mathcal{O}^*(\alpha^k)$.*

Proof. We only need to show that if there exists an algorithm `GoodAlg` solving good instances in time $\mathcal{O}^*(\alpha^k)$, there also exists an algorithm for SFVS-S running in the same time bound $\mathcal{O}^*(\alpha^k)$. The other direction is trivial.

Let $\mathcal{I} = (G = (V, E), T, M, k)$ be an instance of SFVS-S. Notice that α is a constant. We select a sufficiently large constant C such that the branching factor of the branching vector $(1, C, C)$ does not exceed the constant α. Our algorithm for SFVS-S is constructed below.

First, we find the split partition (I, K) of G in polynomial time. If $|K| \leqslant 2C$, we solve the instance directly in polynomial time by brute-force enumerating subsets of K in the solution. Otherwise, we assume that the size of K is at least $2C + 1$. We consider two cases.

Case 1. There is a terminal $t \in K$. In this case, we partition $K \setminus \{t\}$ into two parts K' and K'' such that $|K'| \geqslant C$ and $|K''| \geqslant C$. If t is not included in the solution, at most one vertex in the clique $K \setminus \{t\}$ is not contained in the solution. Consequently, either K' or K'' must be part of the solution. We can branch into three instances by either

- removing t, and decreasing k by 1;
- removing K', and decreasing k by $|K'|$; or
- removing K'', and decreasing k by $|K''|$.

This branching rule yields a branching vector $(1, |K'|, |K''|)$ (w.r.t. the measure k) with the branching factor not greater than α since $|K'| \geqslant C$ and $|K''| \geqslant C$.

Case 2. No terminal is in K. For this case, each non-terminal $v \in I$ is not contained in any T-triangle. However, we cannot directly remove v since it may be incident to marked edges. We can add an edge between v and each vertex $u \in K$ not adjacent to v without creating any new T-triangle. This operation will change v from a vertex in I to a vertex in K, preserving the graph as a split graph. After handling all non-terminal $v \in I$, we know that the terminal set and non-terminal set form a split partition. Subsequently, for each marked edge between two non-terminals v and u, we add a new degree-2 terminal t_{uv} adjacent to u and v and unmark the edge uv. We then apply the DM Reduction and obtain a good instance. Finally, we call the algorithm `GoodAlg` to solve the good instance in time $\mathcal{O}^*(\alpha^k)$.

By either branching with a branching factor not greater than α or solving the instance directly in $\mathcal{O}^*(\alpha^k)$ time, our algorithm runs in time $\mathcal{O}^*(\alpha^k)$. ◀

In the rest of this section, we only need to focus on the algorithm, denoted as `GoodAlg`, for good instances of SFVS-S.

4.2 The Measure and Its Properties

With the help of the auxiliary subgraph and DM Reduction (defined in Section 3), we use the following specific measure to analyze our algorithm.

▶ **Definition 8** (The Measure of Good Instances). *Given a good instance $\mathcal{I} = (G, T, M, k)$ of SFVS-S, let F be the auxiliary subgraph of G with bipartition $V(F) = A \cup B$. We define the measure of the instance $\mathcal{I}$ as*

$$\mu(\mathcal{I}) \coloneqq k - \frac{2}{3}|A|.$$

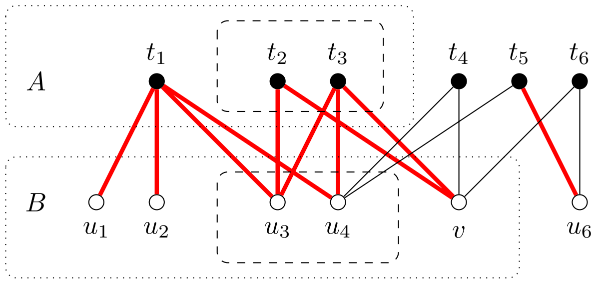

Figure 2 The graph G, where black vertices are terminals, white vertices are non-terminals, thick and red edges are marked edges, and edges between two non-terminals are not presented in the graph; the auxiliary subgraph is F with bipartition $V(F) = A \cup B$, where $A = \{t_1, t_2, t_3\}$ and $B = \{u_1, u_2, u_3, u_4, v\}$ (denoted by dotted boxes). After deleting v, the DM Reduction can be applied. When doing the DM Reduction, $\hat{A} = \{t_2, t_3\}$ and $\hat{B} = \{u_3, u_4\}$ (denoted by dashed boxes) are deleted.

In our algorithm `GoodAlg`, the DM Reduction will be applied as much as possible once the graph changes to keep the instance always good. Additionally, according to Lemma 5, an instance $\mathcal{I}$ can be solved in polynomial time when $\mu(\mathcal{I}) \leqslant 0$. Thus, we can use $\mu(\cdot)$, defined in Definition 8, as our measure to analyze the algorithm.

We may branch on a vertex by including it in the solution or excluding it from the solution in our algorithm. In the first branch, we delete the vertex from the graph and decrease the parameter k by 1. In the second branch, we execute a basic operation of *hiding* the vertex, which is defined according to whether the vertex is a terminal.

Hiding a terminal t: delete every vertex in $N_M(t)$ and decrease k by $|N_M(t)|$.

Hiding a non-terminal v: delete every terminal in $N_M(v)$ and decrease k by $|N_M(v)|$; for each T-triangle vtu containing v, mark edge tu; and last, delete v from the graph.

Here, the notation $N_M(v)$ represents the set of the vertices adjacent to v via a marked edge.

▶ **Lemma 9.** *If there exists a solution containing a vertex v, then it is safe to delete v, decrease k by 1, and do the DM Reduction. If there exists a solution not containing a vertex v, then it is safe to hide v and do the DM Reduction. Moreover, the resulting instance is good after applying either of the above two operations.*

Proof. Assuming a solution S contains v, it is trivial that $S \backslash \{v\}$ is also a solution to the instance $(G - v, T \backslash \{v\}, k - 1)$. Moreover, since the DM Reduction is safe by Lemma 3, the first operation in the lemma is safe.

Now, we assume that a solution S does not contain a vertex v. Since S must cover all edges in M, we know that S contains all neighbours of v in M. This shows that hiding v is safe if v is a terminal. Suppose that v is a non-terminal, for every T-triangle vut containing v, we have that $S \cap \{u, t\} \neq \varnothing$. Consequently, it is safe to mark the edge ut further. Moreover, since the DM Reduction is safe by Lemma 3, the second operation in the lemma is safe.

Finally, either operation only deletes some vertices and marks some edges between terminals and non-terminals. Hence, the terminal set and the non-terminal set still form an independent set and a clique, repetitively. Additionally, the DM Reduction cannot be applied on the resulting instances. Therefore, the resulting instance is good after applying either of the above two operations. ◀

▶ **Lemma 10.** *Given the good instance $\mathcal{I} = (G = (V, E), T, M, k)$, let F be the auxiliary subgraph of G with bipartition $V(F) = A \cup B$, and v be a vertex in $V \backslash A$. Let $\mathcal{I}_1$ be the instance obtained from $\mathcal{I}$ by first deleting v and then doing the DM Reduction (cf. Fig. 2). Then $\mathcal{I}_1$ is a good instance such that $\mu(\mathcal{I}) - \mu(\mathcal{I}_1) \geqslant 0$.*

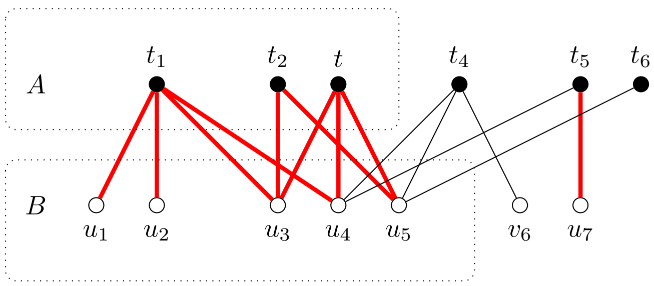

Figure 3 The graph G, where black vertices are terminals, white vertices are non-terminals, thick and red edges are marked edges, and edges between two non-terminals are not presented in the graph; the auxiliary subgraph is F with bipartition $V(F) = A \cup B$, where $A = \{t_1, t_2, t_3\}$ and $B = \{u_1, u_2, u_3, u_4, u_5\}$ (denoted by dotted boxes). After hiding t, the terminal t_2 becomes isolated, and the DM Reduction cannot be applied.

Proof. Let $\mathcal{I}_0 = (G_0, T_0, M_0, k_0)$ be the instance after deleting v from $\mathcal{I}$. Then, $\mathcal{I}_1 = (G_1, T_1, M_1, k_1)$ is the instance after doing the DM Reduction from $\mathcal{I}_0$. Let F_i with bipartition $V(F_i) = A_i \cup B_i$ be the auxiliary subgraph of G_i, where $i \in \{0, 1\}$.

It is clear that $\mu(\mathcal{I}_0) = \mu(\mathcal{I})$ since no edge is newly marked and $k_0 = k$ holds. Assume that $\hat{A}_0 \subseteq A_0$ and $\hat{B}_0 \subseteq B_0$ are deleted. We note that the DM Reduction cannot be applied if and only if $\hat{A}_0 = \hat{B}_0 = \varnothing$. Observe that $A_1 = A_0 \backslash \hat{A}_0$, $B_1 = B_0 \backslash \hat{B}_0$ and $k_1 = k_0 - |\hat{B}_0|$. Thus, we have

$$\mu(\mathcal{I}) - \mu(\mathcal{I}_1) = \mu(\mathcal{I}_0) - \mu(\mathcal{I}_1) = (k_0 - 2/3 \cdot |A_0|) - (k_1 - 2/3 \cdot |A_1|) = |\hat{B}_0| - 2/3 \cdot |\hat{A}_0|.$$

If the DM Reduction cannot be applied, then $\hat{A}_0 = \hat{B}_0 = \varnothing$, which already implies that $\mu(\mathcal{I}) - \mu(\mathcal{I}_1) = 0$. Otherwise, the DM Reduction can be applied after deleting v. In this case, we have $v \in B$. Additionally, according to Lemma 4, $\mathcal{I}$ is a good instance implying that $N_G(\hat{A}_0) = \hat{B}_0 \cup \{v\}$ and $|\hat{B}_0 \cup \{v\}| > |\hat{A}_0|$. Therefore, we obtain that $|\hat{A}_0| = |\hat{B}_0|$. Thus we get $\mu(\mathcal{I}) - \mu(\mathcal{I}_1) \geqslant 0$. The lemma holds. ◀

▶ **Lemma 11.** *Given a good instance $\mathcal{I} = (G = (V, E), T, M, k)$, let F be the auxiliary subgraph of G with bipartition $V(F) = A \cup B$, and t be a terminal in T. Let $\mathcal{I}_1$ be the instance obtained from $\mathcal{I}$ by first hiding t and then doing the DM Reduction (cf. Fig. 3). Then $\mathcal{I}_1$ is a good instance such that*

- *If $t \in A$, it holds $\mu(\mathcal{I}) - \mu(\mathcal{I}_1) \geqslant 4/3$; and*
- *If $t \notin A$, it holds $\mu(\mathcal{I}) - \mu(\mathcal{I}_1) \geqslant 0$.*

Proof. Let instance $\mathcal{I}_0 = (G_0, T_0, M_0, k_0)$ denote the instance after hiding t from $\mathcal{I}$. Then, $\mathcal{I}_1 = (G_1, T_1, M_1, k_1)$ is the instance after doing the DM Reduction from $\mathcal{I}_0$. Let F_i with bipartition $V(F_i) = A_i \cup B_i$ be the auxiliary subgraph of G_i, where $i \in \{0, 1\}$.

After hiding terminal t, a non-terminal is removed if and only if it is adjacent to t via a marked edge. Thus, $B_0 = B \backslash N_M(t)$ and $k_0 = k - |N_M(t)|$ hold. It follows that

$$\mu(\mathcal{I}) - \mu(\mathcal{I}_0) = (k - 2/3 \cdot |A|) - (k_0 - 2/3 \cdot |A_0|) = |N_M(t)| - 2/3 \cdot (|A| - |A_0|).$$

Notice that after hiding t the only deleted terminal is t. Besides, a terminal $t' \neq t$ is removed from A if and only if it becomes an isolated vertex. It follows that $N_G(t') \subseteq N_M(t)$ if $t' \in A \backslash A_0$.

Assume that $\hat{A}_0 \subseteq A_0$ and $\hat{B}_0 \subseteq B_0$ are deleted after doing the DM Reduction. We note that the DM Reduction cannot be applied if and only if $\hat{A}_0 = \hat{B}_0 = \varnothing$. Observe that $k_1 = k_0 - |\hat{B}_0|$ and $|A_1| = |A_0| - |\hat{A}_0|$. Thus, we derive that

$$\begin{aligned}\mu(\mathcal{I}) - \mu(\mathcal{I}_1) &= |N_M(t)| - 2/3 \cdot (|A| - |A_0|) + |\hat{B}_0| - 2/3 \cdot |\hat{A}_0| \\ &\geqslant |N_M(t) \cup \hat{B}_0| - 2/3 \cdot |A \backslash A_1|.\end{aligned}$$

Consider a terminal $t' \in A\backslash A_1$. If $t' \in A\backslash A_0$, we know $N_G(t') \subseteq N_M(t)$. Otherwise, we have $t' \in \hat{A}_0$, and all neighbours of t' in G are deleted, which indicates that $N_G(t') \subseteq \hat{B}_0 \cup N_M(t)$. It follows that $N_G(A\backslash A_1) \subseteq \hat{B}_0 \cup N_M(t)$. Since $\mathcal{I}$ is good, by Lemma 4, we have $A\backslash A_1 = \varnothing$ or $|N_G(A\backslash A_1)| > |A\backslash A_1|$.

If $A\backslash A_1 = \varnothing$ holds, every terminal in A is not deleted, which implies that $t \notin A$. In this case, we have

$$\mu(\mathcal{I}) - \mu(\mathcal{I}_1) \geqslant |N_M(t) \cup \hat{B}_0| \geqslant 0.$$

If $|N_G(A\backslash A_1)| > |A\backslash A_1|$ holds, we have

$$\mu(\mathcal{I}) - \mu(\mathcal{I}_1) \geqslant |N_G(A\backslash A_1)| - 2/3 \cdot |A\backslash A_1| \geqslant 4/3.$$

Therefore, we complete our proof. ◀

▶ **Lemma 12.** *Given a good instance $\mathcal{I} = (G = (V, E), T, M, k)$, let F be the auxiliary subgraph of G with bipartition $V(F) = A \cup B$, and v be a non-terminal in $V\backslash T$. Let $\mathcal{I}_1$ be the instance obtained from $\mathcal{I}$ by first hiding v and then doing the DM Reduction (cf. Fig. 4). Then $\mathcal{I}_1$ is a good instance such that $\mu(\mathcal{I}) - \mu(\mathcal{I}_1) \geqslant 0$.*

Furthermore, if every terminal (resp. non-terminal) is adjacent to at least two non-terminals (resp. terminals) and no two 2-degree terminals have identical neighbours in G, it satisfies that

- *If $v \in B$, it holds*

$$\mu(\mathcal{I}) - \mu(\mathcal{I}_1) \geqslant \min\left\{ \frac{2}{3}|N_G(v) \cap T| - \frac{1}{3}|N_M(v) \cap A|, \frac{4}{3} \right\}.$$

- *If $v \notin B$, it holds*

$$\mu(\mathcal{I}) - \mu(\mathcal{I}_1) \geqslant \min\left\{ \frac{2}{3}|N_G(v) \cap T| + \frac{1}{3}|N_M(v) \cap A|, \frac{4}{3} \right\} = \frac{4}{3}.$$

Proof. Let instance $\mathcal{I}_0 = (G_0, T_0, M_0, k_0)$ denote the instance after hiding v from $\mathcal{I}$. Then, $\mathcal{I}_1 = (G_1, T_1, M_1, k_1)$ is the instance after doing the DM Reduction from $\mathcal{I}_0$. Let F_i with bipartition $V(F_i) = A_i \cup B_i$ be the auxiliary subgraph of G_i, where $i \in \{0, 1\}$.

After hiding v, a vertex is removed if and only if it is a terminal adjacent to v via a marked edge. Thus, $k_0 = k - |N_M(v)|$ and $A\backslash A_0 = A \cap N_M(v)$. It follows that

$$\begin{aligned}\mu(\mathcal{I}) - \mu(\mathcal{I}_0) &= (k - \frac{2}{3}|A|) - (k_0 - \frac{2}{3}|A_0|) \\ &= (k - k_0) + \frac{2}{3}|A_0\backslash A| - \frac{2}{3}|A\backslash A_0| \\ &= |N_M(v)| + \frac{2}{3}|A_0\backslash A| - \frac{2}{3}|A \cap N_M(v)|.\end{aligned}$$

It is easy to see that $\mu(\mathcal{I}) - \mu(\mathcal{I}_0) \geqslant 0$ since $|A \cap N_M(v)| \leqslant |N_M(v)|$. Thus, if the DM Reduction cannot be applied, we have $\mu(\mathcal{I}_0) = \mu(\mathcal{I}_1)$, leading that $\mu(\mathcal{I}) - \mu(\mathcal{I}_1) \geqslant 0$.

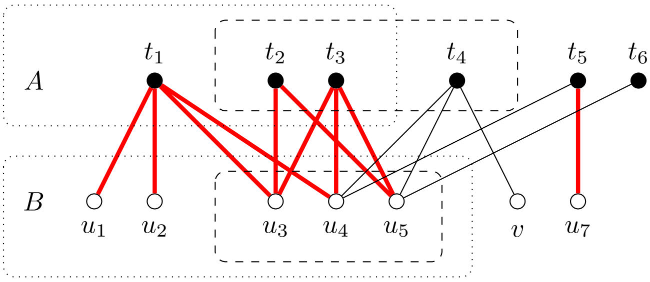

Figure 4 The graph G, where black vertices are terminals, white vertices are non-terminals, thick and red edges are marked edges, and edges between two non-terminals are not presented in the graph; the auxiliary subgraph is F with bipartition $V(F) = A \cup B$, where $A = \{t_1, t_2, t_3\}$ and $B = \{u_1, u_2, u_3, u_4, u_5\}$ (denoted by dotted boxes). After hiding v, the DM Reduction can be applied. When doing the DM Reduction, $\hat{A} = \{t_2, t_3, t_4\}$ and $\hat{B} = \{u_3, u_4, u_5\}$ (denoted by dashed boxes) are deleted.

Next, we consider what terminals belong to set $A_0 \backslash A$. On the one hand, a terminal $t \in A_0 \backslash A$ must be adjacent to v via an unmarked edge. On the other hand, t should not be an isolated vertex after hiding v, which implies that the terminal t is adjacent to at least one vertex distinct from v. Thus, if every terminal is adjacent to at least two non-terminals, we derive that $A_0 \backslash A = (N_G(v) \cap T) \backslash N_M(v)$. It follows that

$$\begin{aligned}\mu(\mathcal{I}) - \mu(\mathcal{I}_0) &= |N_M(v)| + \frac{2}{3}|A_0 \backslash A| - \frac{2}{3}|A \cap N_M(v)| \\ &= |N_M(v)| + \frac{2}{3}|(N_G(v) \cap T) \backslash N_M(v)| - \frac{2}{3}|A \cap N_M(v)| \\ &= |N_M(v)| + \frac{2}{3}(|N_G(v) \cap T| - |N_M(v)|) - \frac{2}{3}|A \cap N_M(v)| \\ &\geqslant \frac{2}{3}|N_G(v) \cap T| - \frac{1}{3}|A \cap N_M(v)|.\end{aligned}$$

If the DM Reduction cannot be applied on $\mathcal{I}_0$, we can derive that

$$\mu(\mathcal{I}) - \mu(\mathcal{I}_1) = \mu(\mathcal{I}) - \mu(\mathcal{I}_0) \geqslant \frac{2}{3}|N_G(v) \cap T| - \frac{1}{3}|A \cap N_M(v)|.$$

Furthermore, if $v \notin B$, then v is not adjacent to any terminal in A, leading that $|A \cap N_M(v)| = 0$. In the case that v belongs to B and it is adjacent to at least two terminals, we further have

$$\mu(\mathcal{I}) - \mu(\mathcal{I}_1) = \mu(\mathcal{I}) - \mu(\mathcal{I}_0) \geqslant \frac{2}{3}|N_G(v) \cap T| \geqslant \frac{4}{3}. \tag{1}$$

Now, we assume that the DM Reduction can be applied on $\mathcal{I}_0$. Suppose $\hat{A}_0 \subseteq A_0$ and $\hat{B}_0 \subseteq B_0$ are deleted. We know that $\hat{A}_0$ and $\hat{B}_0$ are non-empty, and $k_1 = k_0 - |\hat{B}_0| = k - |N_M(v)| - |\hat{B}_0|$ holds. Consider a terminal $t' \in A \backslash A_1$. If $t' \in A \cap N_M(v)$ it is deleted when hiding v; otherwise, it is deleted when doing the DM Reduction which means that $t' \in A \cap \hat{A}_0$. It follows that

$$\begin{aligned}\mu(\mathcal{I}) - \mu(\mathcal{I}_1) =& (k - \frac{2}{3}|A|) - (k_1 - \frac{2}{3}|A_1|) \\ =& |N_M(v)| + |\hat{B}_0| + \frac{2}{3}|A_1 \backslash A| - \frac{2}{3}|A \backslash A_1| \\ \geqslant& |N_M(v)| + |\hat{B}_0| + \frac{2}{3}|A_1 \backslash A| - \frac{2}{3}(|A \cap \hat{A}_0| + |A \cap N_M(v)|) \\ \geqslant& \frac{1}{3}|N_M(v)| + |\hat{B}_0| + \frac{2}{3}|A_1 \backslash A| - \frac{2}{3}|A \cap \hat{A}_0|.\end{aligned}$$

Now, we analyze the lower bound of $\mu(\mathcal{I}) - \mu(\mathcal{I}_1)$ and there are two cases.

Case 1.1. $A \cap \hat{A}_0$ is non-empty. We observe that in graph G, v is not adjacent to any terminal in $\hat{A}_0$. This is because all the terminals adjacent to v via a marked edge are deleted after hiding v and they do not appear in the graph G_0. Hence we know $\hat{B}_0 = N_G(\hat{A}_0)$. Besides, we have $B = N_G(A)$, and thus $B \cap \hat{B}_0 = N_G(A \cap \hat{A}_0)$ holds. since $\mathcal{I}$ is good and $A \cap \hat{A}_0$ is non-empty, we get $|B \cap \hat{B}_0| > |A \cap \hat{A}_0|$. Then we derive that the decrease of the measure is

$$\mu(\mathcal{I}) - \mu(\mathcal{I}_1) \geqslant |\hat{B}_0| - \frac{2}{3}|A \cap \hat{A}_0| \geqslant 1 + \frac{1}{3}|A \cap \hat{A}_0| \geqslant \frac{4}{3}.$$

Case 1.2. $A \cap \hat{A}_0$ is empty. In this case, we can directly obtain that

$$\mu(\mathcal{I}) - \mu(\mathcal{I}_1) \geqslant \frac{1}{3}|N_M(v)| + |\hat{B}_0| + \frac{2}{3}|A_1 \backslash A| \geqslant \frac{1}{3}|N_M(v)| + 1 \geqslant 1. \tag{2}$$

Thus, we have proven the measure does not increase.

Finally, we show that $\mu(\mathcal{I}) - \mu(\mathcal{I}_1) \geqslant 4/3$ always holds when the input graph satisfies the condition in the lemma. We consider two subcases.

Case 2.1. $v \in B$. For this subcase, v is adjacent to at least one terminal in A via a marked edge. Hence, set $N_M(v)$ is non-empty, and we get $\mu(\mathcal{I}) - \mu(\mathcal{I}_1) \geqslant 1/3 + 1 = 4/3$. This completes that the measure is decreased by at least $4/3$ for $v \in B$.

Case 2.2. $v \notin B$. We assume to the contrary that if the DM Reduction can be applied and $\mu(\mathcal{I}) - \mu(\mathcal{I}_1) < 4/3$. According to (2), we can obtain that $|N_M(v)| = 0$, $|\hat{B}_0| = 1$ and $|A_1 \backslash A| = 0$. It means that for every terminal t adjacent to v, it satisfies that

1. $\deg_G(t) \geqslant 2$ holds according to the condition in the lemma;
2. tv is unmarked and $t \notin A$ since $N_M(v)$ is empty;
3. t is in A_0 since $\deg_G(t) \geqslant 2$ and v is the unique non-terminal deleted after hiding v;
4. t is deleted when doing the DM Reduction (i.e., $t \in \hat{A}_0$) since $A_1 \subseteq A$ but $t \notin A$; and
5. t has exactly two neighbours in G which are v and the unique vertex in $\hat{B}_0$ (i.e., $N_G(t) = \{v\} \cup \hat{B}_0$) since $|\hat{B}_0| = 1$.

Above all, we derive that all terminals in $N_G(v)$ are 2-degree vertices and have the same neighbours. By the condition in the lemma, vertex v is adjacent to at least two terminals, contradicting the condition that no two 2-degree terminals have identical neighbours in G. Combine with (1), we conclude that for any vertex $v \notin B$, it holds

$$\mu(\mathcal{I}) - \mu(\mathcal{I}_1) \geqslant \min\left\{\frac{2}{3}|N_G(v) \cap T|, \frac{4}{3}\right\} = \frac{4}{3}.$$ ◀

4.3 An Algorithm for Good Instances

We now give the algorithm `GoodAlg` that solves the good instances. When introducing a step, we assume all previous steps cannot be applied.

▷ Step 1. If $|A| > k$ or even $\mu(\mathcal{I}) < 0$, return No and quit. If $|T| \leqslant k$, return Yes and quit.

▷ Step 2. Delete any vertex in G that is not contained in any T-triangle or marked edge, and then do the DM Reduction on the instance.

▷ Step 3. If there exists a non-terminal v such that $|N_G(v) \cap T| = 1$, hide v and then do the DM Reduction on the instance.

Note that the input instance is good, and every terminal is in some T-triangle after Step 3. Thus, every terminal (resp. non-terminal) is adjacent to at least two non-terminals (resp. terminals).

▷ Step 4. This step deals with some degree-2 terminals in $T \backslash A$, and there are two cases.

1. Let t be a degree-2 terminal in $T \backslash A$. If t is adjacent to exactly one marked edge, hide t and do the DM Reduction.
2. Let t and t' be two degree-2 terminals in $T \backslash A$. If t and t' have the same neighbours and none of them is adjacent to a marked edge, delete one of them and do the DM reduction.

After this step, one can easily find that the condition in Lemma 12 holds.

▷ Step 5. If there exists a non-terminal $v \in B$ adjacent to exactly one terminal t via a marked edge and exactly one terminal t' via an unmarked edge, we branch into two instances by either

- hiding the vertex v and doing the DM Reduction; or
- hiding the vertex t and doing the DM Reduction.

▷ Step 6. If there exists a non-terminal $v \in V \backslash T$ incident to at least one unmarked edge, we branch into two instances by either

- deleting the vertex v, decreasing k by 1, and doing the DM Reduction; or
- hiding v and doing the DM Reduction.

Based on Lemmas 9-12, we can show that Steps 1-4 are safe and they do not increase the measure. After applying any one of Steps 1-6, the resulting instance (or each resulting instance of the branching rule) is good. The complete proofs can be found in the full version.

One can easily find that every edge between a terminal and a non-terminal is marked if Step 6 cannot be applied. Thus, we have $T = A$, and Step 1 will be applied and return the answer. Therefore, we obtain the following result.

▶ **Lemma 13.** *SFVS-S can be solved in time $\mathcal{O}^*(1.8192^k)$.*

Proof. `GoodAlg` contains only two branching operations in Steps 5 and 6. By Lemmas 10, 11, and 12, their branching vectors are not worse than $(1, 4/3)$ whose branching factor is 1.81918. Thus, we conclude that `GoodAlg` solves good instances of SFVS-S in time $\mathcal{O}^*(1.81918^k)$. According to Lemma 7, SFVS-S can be solved in time $\mathcal{O}^*(1.81918^k) \leqslant \mathcal{O}^*(1.8192^k)$. ◀

4.4 SFVS in Chordal Graphs

Our result for SFVS-S (i.e., the $\mathcal{O}^*(1.8192^k)$-time parameterized algorithm) can also be effectively adapted to develop fast parameterized algorithms for SFVS-C.

Our algorithm for SFVS-C is divided into two parts. In the first part, we introduce some reduction rules and branching rules to deal with several easy cases and simplify the instance. If none of the steps in the first part can be applied, we call the instance a "thin" instance. In a thin instance, if all terminals are simplicial, we can easily reduce it to a good instance of SFVS-S and solve it by calling `GoodAlg`. However, if there are "inner" terminals (terminals not being simplicial), we employ a divide-and-conquer approach based on the clique-tree decomposition of chordal graphs in the second part. This technique involves branching on a minimal separator containing inner terminals. In each branch, we will obtain a good instance of SFVS-S for each sub-instance and call `GoodAlg` to solve it. We finally obtain Theorem 14. The details of the algorithm and analysis can be found in the full version.

▶ **Theorem 14.** *For any constant $\alpha > 1$, SFVS-C can be solved in time $\mathcal{O}^*(\alpha^k + 1.6191^k)$ if SFVS-S can be solved in time $\mathcal{O}^*(\alpha^k)$.*

▶ **Corollary 15.** *SFVS-C can be solved in time $\mathcal{O}^*(1.8192^k)$.*

5 Prize-Collecting Maximum Independent Set in Hypergraphs

Although we study the subset feedback vertex set problem in graph subclasses, SFVS-S already generalizes other interesting problems.

Several graph connectivity problems [24, 8, 20] can be modeled as natural problems in hypergraphs. In hypergraphs, an edge can connect any number of vertices, whereas in an ordinary graph, an edge connects exactly two vertices. Given a hypergraph H, the set of vertices and hyperedges are denoted by $V(H)$ and $E(H)$, respectively. The maximum independent set problem in hypergraphs aims to find a maximum vertex subset $I \subseteq V(H)$ such that every hyperedge contains at most one vertex from I. The maximum independent set problem in hypergraphs can be easily reduced to the maximum independent set problem in ordinary graphs: we only need to replace each hyperedge $e \in E(H)$ with a clique formed by the vertices in e to get an ordinary graph. In terms of exact algorithms, we may not need to distinguish this problem in hypergraphs and ordinary graphs. However, the prize-collecting version in hypergraphs becomes interesting, which allows us to violate the independent constraint with penalty. As mentioned above, the prize-collecting version of many central NP-hard problems has drawn certain attention recently.

Prize-Collecting Maximum Independent Set in hypergraphs (PCMIS) **Input**: A hypergraph H and an integer p. **Output**: Determine whether there is a subset of vertices $I \subseteq V(H)$ of the prize at least p, where the prize of I is the size of I minus the number of hyperedges that contain at least two vertices from I.

▶ **Lemma 16.** *PCMIS is polynomially solvable for $p \leqslant 1$, and PCMIS is NP-hard for each constant $p \geqslant 2$.*

Proof. By definition, any singleton set has a prize of 1. Therefore, PCMIS is polynomially solvable when $p \leqslant 1$.

We will prove the NP-hardness of PCMIS with $p \geqslant 2$ by reducing from the maximum independent set problem in ordinary undirected graphs. Let (G, k) be an instance of the maximum independent set problem. We construct an instance (H, p) of PCMIS, where $p \geqslant 2$ is a constant. Since p is a constant, we can assume that $k \geqslant p$.

Suppose $|V(G)| = n$ and $|E(G)| = m$. We now construct a hypergraph H with n vertices and $nm + k - p$ hyperedges. Specifically, for each vertex $v \in V(G)$, we introduce a vertex v', and thus we obtain $V(H) = \{v' : v \in V(G)\}$. Next, for each edge $uv \in E(G)$, we introduce n identical hyperedges $e_i^{uv} = \{u, v\}$ $(i \in [n])$; we also add $k - p$ identical hyperedges $e'_i = V(H)$ $(i \in [k-p])$. Hence, H contains $nm + (k-p)$ hyperedges: $E(H) = E'_1 \cup E'_2$, where $E'_1 = \{e_i^{uv} = \{u, v\} : uv \in E(G),\ i \in [n]\}$ and $E'_2 = \{e'_i = V(H) : i \in [k-p]\}$.

Finally, we show (G, k) is a Yes-instance if and only if (H, p) is a Yes-instance. On the one hand, let $I \subseteq V(G)$ be an independent set of G with the size k. Let $I' = \{v' : v \in I\}$, and we have that every hyperedge in E'_1 contains at most one vertex from I'. Additionally, each hyperedge in E'_2 contains exactly k vertices from I'. Since $k \geqslant p \geqslant 2$, we derive that the prize of I' is $k - |E'_2| = k - (k-p) = p$, which means that (H, p) is a Yes-instance.

On the other hand, let $I' \subseteq V(H)$ be a vertex subset of H with the prize p. Let X be the set of hyperedges containing at least two vertices from I'. We have that $p = |I'| - |X|$. Note that if one hyperedge $e_i^{vu} \in E'_1$ $(i \in [n])$ is in X, then all the n hyperedges identical with e_i^{vu} should be in X, which will make $p \leqslant |I'| - n \leqslant 0$, a contradiction. Therefore, we derive that $I = \{v \in V(G) : v' \in I'\}$ is an independent set in G and $X \cap E'_1 = \varnothing$. Since $X \subseteq E'_2$, we have $|I| = |I'| = p + |X| \geqslant p + (k - p) \geqslant k$. We conclude that I is an independent set of G with size at least k, leading that (G, k) is a Yes-instance. ◀

Previously, no exact algorithm for PCMIS faster than $\mathcal{O}^*(2^n)$ is known. We show that PCMIS can be solved by reducing it to SFVS-S, and then we break the "2^n-barrier" for PCMIS. For an instance (H, p) of PCMIS, we construct an instance (G, T, M, k) of SFVS-S. Suppose that H contains n vertices and m hyperedges. We construct a split graph G as follows. We first introduce a clique with vertices $\{v' : v \in V(H)\}$; then for each hyperedge $e \in E(H)$, introduce a new terminal t'_e whose neighbors are exactly the vertices in the clique corresponding to the vertices in hyperedge e. The terminal set is set as $T = \{t'_e : e \in E(H)\}$, the marked edge set is set as $M = \varnothing$, and let $k = n - p$.

▶ **Lemma 17.** *For any constant $\alpha > 1$, an $\mathcal{O}^*(\alpha^k)$-time algorithm for SFVS-S leads to an $\mathcal{O}^*(\alpha^{n-p})$-algorithm for PCMIS.*

Proof. For an instance (H, p) of PCMIS, we construct an instance (G, T, M, k) of SFVS-S. Suppose that H contains n vertices and m hyperedges. We construct a split graph G as follows. We first introduce a clique with vertices $\{v' : v \in V(H)\}$; then for each hyperedge $e \in E(H)$, introduce a new terminal t'_e whose neighbors are exactly the vertices in the clique corresponding to the vertices in hyperedge e. The terminal set is set as $T = \{t'_e : e \in E(H)\}$, the marked edge set is set as $M = \varnothing$, and let $k = n - p$.

We have the key idea: every hyperedge in a hypergraph contains at most one vertex if and only if the corresponding split graph of the hypergraph contains no T-triangle. Let S be a solution to (G, T, M, k) containing m' terminals and n' non-terminals. We can see that $I = \{v : v' \in V(G) \backslash (T \cup S)\}$ is a vertex set with the prize at least $(n - n') - m' = n - k = p$ in (H, p). As for the opposite direction, suppose I is a solution to (H, p) of size n', and the prize of I is p. We can derive that there are at most $m' = n' - p$ hyperedges containing at least two vertices from I. This means that in G, we can remove m' terminals and $n - n'$ non-terminals to obtain a subgraph without any T-triangle, leading that (G, T, M, k) has a solution of size $(n - n') + m' = n - p = k$. Therefore, (G, T, M, k) is a Yes-instance if and only if (H, p) is a Yes-instance. We finish the proof of Lemma 17. ◀

Based on Lemma 13, we obtain an exact algorithm for PCMIS breaking the 2^n barrier.

▶ **Corollary 18.** *PCMIS can be solved in time $\mathcal{O}^*(1.8192^n)$.*

6 Conclusion

In this paper, we broke the "2^k-barrier" for SFVS in Chordal Graphs. As a corollary, we obtained an exact algorithm faster than $\mathcal{O}^*(2^n)$ for Prize-Collecting Maximum Independent Set in hypergraphs. To achieve this breakthrough, we introduced a new measure based on the Dulmage-Mendelsohn decomposition. This measure served as the basis for designing and analyzing an algorithm that addresses a crucial sub-case. Furthermore, we analyzed the whole algorithm using the traditional measure k, employing various techniques such as a divide-and-conquer approach and reductions based on small separators. The bottleneck of our algorithm occurs when dealing with SFVS in Split Graphs.

We think it is interesting to break the "2^k-barrier" or "2^n-barrier" for more important problems, say the Steiner Tree problem and TSP. For SFVS in general graphs, the best result is $\mathcal{O}^*(4^k)$ [30, 31]. It will also be interesting to reduce the gap between the results in general graphs and chordal graphs.

References

1 Faisal N. Abu-Khzam. A kernelization algorithm for d-hitting set. *J. Comput. Syst. Sci.*, 76(7):524–531, 2010.

2 Tian Bai and Mingyu Xiao. Exact and parameterized algorithms for restricted subset feedback vertex set in chordal graphs. In *Theory and Applications of Models of Computation - 17th Annual Conference, TAMC*, volume 13571 of *LNCS*, pages 249–261. Springer, 2022.

3 Tian Bai and Mingyu Xiao. A parameterized algorithm for subset feedback vertex set in tournaments. *Theor. Comput. Sci.*, 975:114139, 2023.

4 Tian Bai and Mingyu Xiao. Exact algorithms for restricted subset feedback vertex set in chordal and split graphs. *Theor. Comput. Sci.*, 984:114326, 2024. doi:10.1016/J.TCS.2023.114326.

5 Jannis Blauth and Martin Nägele. An improved approximation guarantee for prize-collecting TSP. In *Proceedings of the 55th Annual ACM Symposium on Theory of Computing, (STOC)*, pages 1848–1861. ACM, 2023.

6 Hans L. Bodlaender. On disjoint cycles. *Int. J. Found. Comput. Sci.*, 5(1):59–68, 1994.

7 Peter Buneman. A characterisation of rigid circuit graphs. *Discrete Mathematics*, 9(3):205–212, 1974.

8 Chandra Chekuri and Chao Xu. Computing minimum cuts in hypergraphs. In *Proceedings of the 28th Annual ACM-SIAM Symposium on Discrete Algorithms, (SODA)*, pages 1085–1100. SIAM, 2017.

9 Jianer Chen and Iyad A. Kanj. Constrained minimum vertex cover in bipartite graphs: complexity and parameterized algorithms. *J. Comput. Syst. Sci.*, 67(4):833–847, 2003.

10 Marek Cygan, Fedor V. Fomin, Lukasz Kowalik, Daniel Lokshtanov, Dániel Marx, Marcin Pilipczuk, Michal Pilipczuk, and Saket Saurabh. *Parameterized Algorithms*. Springer, Berlin, 2015.

11 Marek Cygan, Marcin Pilipczuk, Michal Pilipczuk, and Jakub Onufry Wojtaszczyk. Subset feedback vertex set is fixed-parameter tractable. *SIAM J. Discret. Math.*, 27(1):290–309, 2013.

12 Erik D. Demaine, MohammadTaghi Hajiaghayi, and Dániel Marx. 09511 open problems – parameterized complexity and approximation algorithms. In *Parameterized complexity and approximation algorithms*, volume 9511 of *Dagstuhl Seminar Proceedings (DagSemProc)*, pages 1–10, Dagstuhl, Germany, 2010.

13 Gabriel Andrew Dirac. On rigid circuit graphs. *Abh. Math. Sem. Univ. Hamburg*, 25(1):71–76, 1961.

14 Michael Dom, Jiong Guo, Falk Hüffner, Rolf Niedermeier, and Anke Truß. Fixed-parameter tractability results for feedback set problems in tournaments. *J. Discrete Algorithms*, 8(1):76–86, 2010.

15 A. L. Dulmage and N. S. Mendelsohn. Coverings of bipartite graphs. *Canadian Journal of Mathematics*, 10:517–534, 1958.

16 Andrew L Dulmage. A structure theory of bipartite graphs of finite exterior dimension. *The Transactions of the Royal Society of Canada, Section III*, 53:1–13, 1959.

17 Guy Even, Joseph Naor, and Leonid Zosin. An 8-approximation algorithm for the subset feedback vertex set problem. *SIAM J. Comput.*, 30(4):1231–1252, 2000.

18 Fedor V. Fomin, Pinar Heggernes, Dieter Kratsch, Charis Papadopoulos, and Yngve Villanger. Enumerating minimal subset feedback vertex sets. *Algorithmica*, 69(1):216–231, 2014.

19 Fedor V. Fomin, Tien-Nam Le, Daniel Lokshtanov, Saket Saurabh, Stéphan Thomassé, and Meirav Zehavi. Subquadratic kernels for implicit 3-hitting set and 3-set packing problems. *ACM Trans. Algorithms*, 15(1):13:1–13:44, 2019.

20 Kyle Fox, Debmalya Panigrahi, and Fred Zhang. Minimum cut and minimum k-cut in hypergraphs via branching contractions. *ACM Trans. Algorithms*, 19(2):13:1–13:22, 2023.

21 Takuro Fukunaga. Spider covers for prize-collecting network activation problem. *ACM Trans. Algorithms*, 13(4):49:1–49:31, 2017.

22 Delbert Fulkerson and Oliver Gross. Incidence matrices and interval graphs. *Pacific J. Math.*, 15(3):835–855, 1965.

23 Fănică Gavril. The intersection graphs of subtrees in trees are exactly the chordal graphs. *Journal of Combinatorial Theory, Series B*, 16(1):47–56, 1974.

24 Magnús M. Halldórsson and Elena Losievskaja. Independent sets in bounded-degree hypergraphs. *Discret. Appl. Math.*, 157(8):1773–1786, 2009.

25 Eva-Maria C. Hols and Stefan Kratsch. A randomized polynomial kernel for subset feedback vertex set. *Theory Comput. Syst.*, 62(1):63–92, 2018.

26 John E. Hopcroft and Richard M. Karp. An $n^{5/2}$ algorithm for maximum matchings in bipartite graphs. *SIAM J. Comput.*, 2(4):225–231, 1973.

27 Falk Hüffner, Christian Komusiewicz, Hannes Moser, and Rolf Niedermeier. Fixed-parameter algorithms for cluster vertex deletion. *Theory Comput. Syst.*, 47(1):196–217, 2010.

28 Yoichi Iwata. Linear-time kernelization for feedback vertex set. In *44th International Colloquium on Automata, Languages, and Programming, ICALP*, pages 68:1–68:14, 2017.

29 Yoichi Iwata and Yusuke Kobayashi. Improved analysis of highest-degree branching for feedback vertex set. *Algorithmica*, 83(8):2503–2520, 2021.

30 Yoichi Iwata, Magnus Wahlström, and Yuichi Yoshida. Half-integrality, LP-branching, and FPT algorithms. *SIAM J. Comput.*, 45(4):1377–1411, 2016.

31 Yoichi Iwata, Yutaro Yamaguchi, and Yuichi Yoshida. 0/1/all CSPs, half-integral A-path packing, and linear-time FPT algorithms. In *59th IEEE Annual Symposium on Foundations of Computer Science, (FOCS)*, pages 462–473, 2018.

32 Richard M. Karp. Reducibility among combinatorial problems. In *Proceedings of a Symposium on the Complexity of Computer Computations*, The IBM Research Symposia Series, pages 85–103, 1972.

33 Mithilesh Kumar and Daniel Lokshtanov. Faster exact and parameterized algorithm for feedback vertex set in tournaments. In *33rd Symposium on Theoretical Aspects of Computer Science, (STACS)*, volume 47 of *LIPIcs*, pages 49:1–49:13, 2016.

34 László Lovász and Michael D. Plummer. *Matching theory*, volume 121 of *North-Holland Mathematics Studies*. Elsevier Science Ltd., London, 1 edition, 1986.

35 Charis Papadopoulos and Spyridon Tzimas. Polynomial-time algorithms for the subset feedback vertex set problem on interval graphs and permutation graphs. *Discret. Appl. Math.*, 258:204–221, 2019.

36 Lehilton Lelis Chaves Pedrosa and Hugo Kooki Kasuya Rosado. A 2-approximation for the k-prize-collecting steiner tree problem. *Algorithmica*, 84(12):3522–3558, 2022.

37 Geevarghese Philip, Varun Rajan, Saket Saurabh, and Prafullkumar Tale. Subset feedback vertex set in chordal and split graphs. *Algorithmica*, 81(9):3586–3629, 2019.

38 Donald J. Rose, Robert Endre Tarjan, and George S. Lueker. Algorithmic aspects of vertex elimination on graphs. *SIAM J. Comput.*, 5(2):266–283, 1976.

39 Földes Stephane and Peter Hammer. Split graphs. In *Proceedings of the 8th south-east Combinatorics, Graph Theory, and Computing*, volume 9, pages 311–315, 1977.

40 Stéphan Thomassé. A $4k^2$ kernel for feedback vertex set. *ACM Trans. Algorithms*, 6(2):32:1–32:8, 2010.

41 Kangyi Tian, Mingyu Xiao, and Boting Yang. Parameterized algorithms for cluster vertex deletion on degree-4 graphs and general graphs. In *Computing and Combinatorics - 29th International Conference, (COCOON)*, volume 14422 of *LNCS*, pages 182–194. Springer, 2023.

42 Magnus Wahlström. *Algorithms, measures and upper bounds for satisfiability and related problems*. PhD thesis, Linköping University, Sweden, 2007.

43 James Richard Walter. *Representations of rigid cycle graphs.* PhD thesis, Wayne State University, 1972.
44 Mihalis Yannakakis and Fanica Gavril. The maximum k-colorable subgraph problem for chordal graphs. *Inf. Process. Lett.*, 24(2):133–137, 1987.