Mim-Width II. The Feedback Vertex Set Problem*

Lars Jaffke[†1], O-joung Kwon[‡2], and Jan Arne Telle[1]

[1]Department of Informatics, University of Bergen, Norway.
{lars.jaffke, jan.arne.telle}@uib.no
[2]Department of Mathematics, Incheon National University, South Korea.
ojoungkwon@gmail.com

July 17, 2019

Abstract

We give a first polynomial-time algorithm for (Weighted) Feedback Vertex Set on graphs of bounded *maximum induced matching width* (mim-width). Explicitly, given a branch decomposition of mim-width w, we give an $n^{\mathcal{O}(w)}$-time algorithm that solves Feedback Vertex Set. This provides a unified polynomial-time algorithm for many well-known classes, such as Interval graphs, Permutation graphs, and Leaf power graphs (given a leaf root), and furthermore, it gives the first polynomial-time algorithms for other classes of bounded mim-width, such as Circular Permutation and Circular k-Trapezoid graphs (given a circular k-trapezoid model) for fixed k. We complement our result by showing that Feedback Vertex Set is W[1]-hard when parameterized by w and the hardness holds even when a linear branch decomposition of mim-width w is given.

Keywords. Graph Width Parameters; Mim-Width; Graph Classes; Feedback Vertex Set.

1 Introduction

A feedback vertex set in a graph is a subset of its vertex set whose removal results in an acyclic graph. The problem of finding a smallest such set is one of Karp's 21 famous NP-complete problems [26] and many algorithmic techniques have been developed to attack this problem, see e.g. the survey [14]. The study of Feedback Vertex Set through the lens of parameterized algorithmics dates back to the earliest days of the field [9] and throughout the years numerous efforts have been

*The work was partially done while the authors were at Polytechnic University of Valencia, Spain. Based on an extended abstract that appeared at STACS 2018 [23] and the note [20]. Published in *Algorithmica*. The first part of this series, titled "Mim-Width I. Induced Path Problems" [22], is based on an extended abstract that appeared at IPEC 2017 [21].

†Supported by the Bergen Research Foundation (BFS).

‡Supported by the National Research Foundation of Korea (NRF) grant funded by the Ministry of Education (No. NRF-2018R1D1A1B07050294), and the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation programme (ERC consolidator grant DISTRUCT, agreement No. 648527). Part of the research took place while Kwon was at Logic and Semantics, Technische Universität Berlin, Berlin, Germany.

made to obtain faster algorithms for this problem [2, 3, 6, 8, 9, 10, 18, 25, 31, 32]. In terms of parameterizations by structural properties of the graph, FEEDBACK VERTEX SET is known to be FPT parameterized by tree-width [3] and clique-width [5], and W[1]-hard but in XP parameterized by the size of an independent set and the size of a maximum induced matching [24].

In this paper, we study FEEDBACK VERTEX SET parameterized by the *maximum induced matching width* (mim-width for short), a graph parameter defined in 2012 by Vatshelle [34] which measures how easy it is to decompose a graph along cuts[1] with bounded maximum induced matching size on the bipartite graph induced by edges crossing the cut. One interesting aspect of this width-measure is that its modeling power is much stronger than tree-width and clique-width, and many well-known and deeply studied graph classes such as INTERVAL graphs and PERMUTATION graphs have (linear) mim-width 1, with decompositions that can be found in polynomial time [1, 34], while their clique-width can be proportional to the square root of the number of vertices [17]. Hence, designing an algorithm for a problem Π that runs in XP time parameterized by mim-width yields polynomial-time algorithms for Π on several interesting graph classes at once.

We give an XP-time algorithm for FEEDBACK VERTEX SET parameterized by mim-width, assuming that a branch decomposition of bounded mim-width is given. This problem was mentioned as an 'interesting topic for further research' in [24]. Since such a decomposition can be computed in polynomial time [1, 34] for the following classes, this provides a unified polynomial-time algorithm for FEEDBACK VERTEX SET on all of them: INTERVAL and BI-INTERVAL graphs, CIRCULAR ARC, PERMUTATION and CIRCULAR PERMUTATION graphs, CONVEX graphs, k-TRAPEZOID, CIRCULAR k-TRAPEZOID,[2] k-POLYGON, DILWORTH-k and CO-k-DEGENERATE graphs for fixed k. Recently, a superset of the authors proved that taking an (arbitrary) power of a graph increases its mim-width by at most a factor of 2 [19], thereby strictly enhancing the previous list by e.g. powers of PERMUTATION graphs.[3] Furthermore, the authors showed that LEAF POWER graphs also have bounded mim-width[4] [23]. Our algorithm can be applied to WEIGHTED FEEDBACK VERTEX SET as well, which on several of these classes was not known to be solvable in polynomial time.

Theorem 1. *Given an n-vertex graph and one of its branch decompositions of mim-width w, we can solve* (WEIGHTED) FEEDBACK VERTEX SET *in time $n^{\mathcal{O}(w)}$.*

Let us explain some of the essential ingredients of our dynamic programming algorithm which solves the dual MAXIMUM (WEIGHT) INDUCED FOREST problem. Note that the two problems are equivalent in the mim-width parameterization. A crucial observation is that if a forest contains no induced matching of size $w+1$, then the number of internal vertices of the forest is bounded by $6w$ (Lemma 8). Motivated by this observation, given a forest, we define the forest obtained by removing its isolated vertices and leaves to be its *reduced forest*. Let (A, B) be a cut of a graph G and denote by $G_{A,B}$ the bipartite graph induced by this cut. The observation implies that if there is no induced matching of size $w+1$ in $G_{A,B}$, then there are at most $\mathcal{O}(n^{6w})$ possible reduced forests of some induced forests consisting of edges crossing this cut. We enumerate all of them, and use them as indices of the table of our algorithm.

[1] A *cut* of a graph is a bipartition of its vertex set.

[2] Given a (circular) k-trapezoid model.

[3] It is known that powers of permutation graphs are not necessarily permutation graphs [4, 15].

[4] Note however that in contrast to the previously mentioned classes, for LEAF POWER graphs it is currently not known whether the corresponding decomposition can be computed in polynomial time. The construction in the proof presented in [23] uses a given leaf root of the input graph and it is still not known whether a leaf root of a leaf power graph can be computed in polynomial time.

Following the given branch decomposition, we want to recursively ask whether for a forest R in $G_{A,B}$, there is a forest in the graph on the union of A and the boundary[5] of B, such that its restriction to $G_{A,B}$ has R as a reduced forest. When we decide to add some other vertex from B to our forest at a later stage of the algorithm, we do not want to have an edge from A to B not intersecting the vertices of R. One way to avoid these additional edges is to take a vertex cover M of $G_{A,B} - V(R)$, and then ask whether there is a forest F on the union of A and the boundary of B such that it avoids M and $G_{A,B} \cap F$ has R as a reduced forest. We observe that for any such forest F, there is always a vertex cover M that satisfies this condition. This suggests that we add all possible minimal vertex covers of $G_{A,B} - V(R)$ as a second component of the table indices.

To argue that the number of table entries stays bounded by $n^{\mathcal{O}(w)}$, we use the known result that every n-vertex bipartite graph with maximum induced matching size w has at most n^w minimal vertex covers, and that we can enumerate them within the same time bound [21]. Remark that in the companion paper [21], we use minimal vertex covers of a bipartite graph in a similar way. However, in the algorithms described in [21], the full intersection of a solution with a cut could be used as a part of the table indices, whereas in the present paper, we can only store reduced forests (as opposed to the full forests), resulting in a more technical exposition.

Additionally, we observe that our algorithm can also be applied to the connected variant of the problem, i.e. it can be used to solve the Maximum (Weight) Induced Tree problem in the same parameterization and time bound as well.

A natural next question about the complexity of Feedback Vertex Set parameterized by mim-width is whether the problem is fixed-parameter tractable. Under the standard assumption that $\mathsf{FPT} \neq \mathsf{W}[1]$, we rule out this possibility by showing that it is $\mathsf{W}[1]$-hard in the even more restrictive parameterization by *linear* mim-width.

Theorem 2 (See Corollary 24). Feedback Vertex Set *is* $\mathsf{W}[1]$*-hard parameterized by linear mim-width, even if a linear branch decomposition of bounded mim-width is given.*

More precisely, we show that the dual Maximum Induced Forest problem is $\mathsf{W}[1]$-hard parameterized by solution size plus the mim-width of a given linear branch decomposition of the input graph which implies the previous theorem. Moreover, our reduction shows hardness for the Maximum Induced Tree problem in the same parameterization as well. To obtain this result, we build on a reduction that was recently given by Fomin, Golovach, and Raymond [16].

The rest of the paper is organized as follows: After giving some preliminary definitions and tools in Section 2, we give necessary lemmas regarding reduced forests in Section 3. We obtain our algorithm in Section 4 and present the hardness results in Section 5. We conclude with some remarks in Section 6.

2 Preliminaries

For integers a and b with $a \le b$, we let $[a..b] \coloneqq \{a, a+1, \ldots, b\}$ and if a is positive, we define $[a] \coloneqq [1..a]$. Every graph in this paper is finite, undirected and simple. For a graph G we denote by $V(G)$ and $E(G) \subseteq \binom{V(G)}{2}$ its vertex and edge set, respectively. For graphs G and H we say that G is a *subgraph* of H, if $V(G) \subseteq V(H)$ and $E(G) \subseteq E(H)$. Let G be a graph. For a vertex set $X \subseteq V(G)$, we denote by $G[X]$ the subgraph *induced* by X, i.e. $G[X] \coloneqq (X, E(G) \cap \binom{X}{2})$. We use

[5] I.e. the vertices in B that have neighbors in A.

the shorthand $G-X$ for $G[V(G)\setminus X]$. For a vertex $v \in V(G)$, let $G-v := G-\{v\}$, and for an edge $e \in E(G)$, let $G-e = (V(G), E(G)\setminus\{e\})$. For a vertex $v \in V(G)$, we denote by $N_G(v)$ the set of *neighbors* of v in G, i.e. $N_G(v) := \{w \in V(G) \mid \{v,w\} \in E(G)\}$, and the number of neighbors of v is called its *degree*, denoted by $\deg_G(v) := |N_G(v)|$. For $A \subseteq V(G)$, let $N_G(A)$ be the set of vertices in $V(G)\setminus A$ having a neighbor in A. We drop G as a subscript if it is clear from the context.

We denote by $\mathcal{C}(G)$ the set of connected components of G.

For two disjoint vertex sets $X, Y \subseteq V(G)$, we denote by $G[X,Y]$ the bipartite subgraph of G with bipartition (X,Y) such that for $x \in X, y \in Y$, x and y are adjacent in $G[X,Y]$ if and only if they are adjacent in G. A *cut* of G is a bipartition (A,B) of its vertex set. A set M of edges is a *matching* if no two edges in M share an endpoint, and a matching $\{a_1b_1,\ldots,a_kb_k\}$ is *induced* if there are no other edges in the subgraph induced by $\{a_1,b_1,\ldots,a_k,b_k\}$. A vertex set $S \subseteq V(G)$ is a *vertex cover* of G if every edge in G is incident with a vertex in S.

For two graphs G_1 and G_2, $G_1 \cup G_2$ is the graph with the vertex set $V(G_1)\cup V(G_2)$ and the edge set $E(G_1)\cup E(G_2)$, and $G_1 \cap G_2$ is the graph with the vertex set $V(G_1)\cap V(G_2)$ and the edge set $E(G_1)\cap E(G_2)$.

A connected graph all of whose vertices have degree 2 is called a *cycle*. A graph that does not contain a cycle as a subgraph is called a *forest* and a connected forest is a *tree*. A tree of maximum degree 2 is called a *path* and we refer to the *length* of a path as the number of its edges.

A *star* is a tree on at least three vertices containing a special vertex, called its *central* vertex, adjacent to all other vertices. We require a star to have at least three vertices to emphasize the distinction between a star and a graph consisting of a single edge, as they require different treatment in our algorithm.

2.1 Parameterized Complexity

We now give the basic definitions in parameterized complexity and refer to [7, 11] for an introduction.

Definition 3 (Parameterized Problem, FPT, XP). Let Σ be an alphabet. A *parameterized problem* is a set $\Pi \subseteq \Sigma^* \times \mathbb{N}$, the second component being the *parameter* which usually expresses a structural measure of the input.

1. A parameterized problem Π is *fixed-parameter tractable* (FPT) if there exists an algorithm that for any $\langle x,k\rangle \in \Sigma^*\times\mathbb{N}$ decides whether $\langle x,k\rangle \in \Pi$ in time $f(k)\cdot|x|^{\mathcal{O}(1)}$, for some computable function f.

2. A parameterized problem Π is in XP if there exists an algorithm that for any $\langle x,k\rangle \in \Sigma^*\times\mathbb{N}$ decides whether $\langle x,k\rangle \in \Pi$ in time $f(k)\cdot|x|^{g(k)}$, for some computable functions f and g.

2.2 Branch Decompositions and Mim-Width

For a graph G and a vertex subset A of G, we define $\mathrm{mim}_G(A)$ to be the maximum size of an induced matching in $G[A, V(G)\setminus A]$.

A tree is called *subcubic* if all internal vertices have degree 3. A pair $(T,\mathcal{L})$ of a subcubic tree T on at least 2 vertices and a bijection $\mathcal{L}$ from $V(G)$ to the set of leaves of T is called a *branch decomposition*. For each edge e of T, let T_1^e and T_2^e be the two connected components of $T-e$, and let (A_1^e, A_2^e) be the vertex bipartition of G such that for each $i \in \{1,2\}$, A_i^e is the set of all vertices in G mapped to leaves contained in T_i^e by $\mathcal{L}$. The *mim-width of* $(T, \mathcal{L})$, denoted by $\mathrm{mimw}(T, \mathcal{L})$, is defined as $\max_{e \in E(T)} \mathrm{mim}_G(A_1^e)$. The minimum mim-width over all branch decompositions of G is called the *mim-width of* G, and the *linear mim-width of* G if T is restricted to a tree where each internal node is adjacent to at least one leaf. If $|V(G)| \leq 1$, then G does not admit a branch decomposition, and the mim-width and linear mim-width of G are defined to be 0.

To avoid confusion, we refer to elements in $V(T)$ as *nodes* and elements in $V(G)$ as *vertices* throughout the rest of the paper. Given a branch decomposition, one can subdivide an arbitrary edge and let the newly created vertex be the root of T, in the following denoted by r. Throughout the following we assume that each branch decomposition has a root node of degree two. For two nodes $t, t' \in V(T)$, we say that t' is a *descendant* of t if t lies on the path from r to t' in T. For $t \in V(T)$, we denote by G_t the subgraph induced by all vertices that are mapped to a leaf that is a descendant of t, i.e. $G_t = G[X_t]$, where $X_t = \{v \in V(G) \mid \mathcal{L}^{-1}(t') = v \text{ where } t' \text{ is a descendant of } t \text{ in } T\}$. We use the shorthand V_t for $V(G_t)$ and let $\overline{V_t} \coloneqq V(G) \setminus V_t$.

The following definitions which we relate to branch decompositions of graphs will play a central role in the design of the algorithms in Section 4.

Definition 4 (Boundary). Let G be a graph and $A, B \subseteq V(G)$ such that $A \cap B = \emptyset$. We let $\mathrm{bd}_B(A)$ be the set of vertices in A that have a neighbor in B, i.e. $\mathrm{bd}_B(A) \coloneqq \{v \in A \mid N(v) \cap B \neq \emptyset\}$. We define $\mathrm{bd}(A) \coloneqq \mathrm{bd}_{V(G) \setminus A}(A)$ and call $\mathrm{bd}(A)$ the *boundary* of A in G.

Definition 5 (Crossing Graph). Let G be a graph and $A, B \subseteq V(G)$. If $A \cap B = \emptyset$, we define the graph $G_{A,B} \coloneqq G[\mathrm{bd}_B(A), \mathrm{bd}_A(B)]$ to be the *crossing graph from A to B*.

If $(T, \mathcal{L})$ is a branch decomposition of G and $t_1, t_2 \in V(T)$ such that $V_{t_1} \cap V_{t_2} = \emptyset$, we use the shorthand $G_{t_1,t_2} \coloneqq G_{V_{t_1},V_{t_2}}$. We use the analogous shorthand notations $G_{t_1,\overline{t_2}} \coloneqq G_{V_{t_1},\overline{V_{t_2}}}$ and $G_{\overline{t_1},t_2} \coloneqq G_{\overline{V_{t_1}},V_{t_2}}$ (whenever these graphs are defined). For the frequently arising case when we consider $G_{t,\bar{t}}$ for some $t \in V(T)$, we refer to this graph as the *crossing graph w.r.t. t*.

2.3 The Minimal Vertex Covers Lemma

We recall the minimal vertex covers lemma from the first volume of this series of papers. It shows that given a vertex set A of G, the number of minimal vertex covers in $G_{A,V(G)\setminus A}$ is bounded by $n^{\mathrm{mim}_G(A)}$, and furthermore, the set of all minimal vertex covers can be enumerated in time $n^{\mathcal{O}(\mathrm{mim}_G(A))}$. This observation is crucial to argue that in our dynamic programming algorithm, there are at most $n^{\mathcal{O}(w)}$ table entries to consider at each node of the given branch decomposition $(T, \mathcal{L})$, where w denotes the mim-width of $(T, \mathcal{L})$.

Corollary 6 (Minimal Vertex Covers Lemma; [22]). *Let H be a bipartite graph on n vertices with a bipartition (A, B). The number of minimal vertex covers of H is at most $n^{\mathrm{mim}_H(A)}$, and the set of all minimal vertex covers of H can be enumerated in time $n^{\mathcal{O}(\mathrm{mim}_H(A))}$.*

3 Lemmas on reduced forests and vertex covers

In this section, we introduce some technical concepts and prove some technical lemmas that will be used to devise and analyze the Feedback Vertex Set algorithm given in Section 4. As alluded to in the introduction, we would like to store subgraphs of the intersection of induced forests with the edges crossing a cut. We call these subgraphs *reduced forests* and we begin by defining them formally.

Definition 7 (Reduced Forest). Let F be a forest. A *reduced forest* of F is an induced subforest of F obtained as follows.

1. Remove all isolated vertices of F.
2. For each component C of F with $|V(C)| = 2$, remove one of its vertices.
3. For each component C of F with $|V(C)| \geq 3$, remove all leaves of C.

Note that if F has no single-edge component, then the reduced forest is uniquely defined. We give an upper bound on the size of a reduced forest of a forest F by a function of the size of a maximum induced matching in F.

Lemma 8. *Let p be a positive integer. If F is a forest whose maximum induced matching has size at most p and $\mathfrak{R}$ is a reduced forest of F, then $|V(\mathfrak{R})| \leq 6p$.*

Proof. For a forest F, we denote by $m(F)$ the size of a maximum induced matching in F. We prove the lemma by induction on $m(F)$. If $m(F) = 0$, then F contains no edges, and $|V(\mathfrak{R})| = 0$. If $m(F) = 1$, then F consists of one component that contains no path of length 4 and (possibly) some isolated vertices which implies that $\mathfrak{R}$ contains at most 2 vertices. We may assume that $m(F) = p > 1$. We may further assume that F contains no isolated vertices, as they will be removed in the reduced forest.

Suppose F contains a connected component C containing no path of length 4. As observed, C contains no induced matching of size larger than one. Since C contains an edge, we have $m(F - V(C)) = m(F) - 1$. Let $\mathfrak{R}_{F-V(C)}$ be a reduced forest of $F - V(C)$ that is a restriction of $\mathfrak{R}$. By the induction hypothesis, $\mathfrak{R}_{F-V(C)}$ contains at most $6(p-1)$ vertices, and we have that $\mathfrak{R}$ contains at most $6(p-1)+2 \leq 6p$ vertices. Thus, we may assume that every connected component C of F contains a path of length 4, implying that its reduced forest contains at least 3 vertices. It also implies that every connected component of F has a unique reduced forest.

Now, suppose F contains a path $v_1v_2v_3v_4v_5$ such that v_1 and v_5 are not leaves of F, and v_2, v_3, v_4 have degree 2 in $\mathfrak{R}$. Let F' be the forest obtained from F by removing v_2, v_3, v_4 and adding an edge v_1v_5. Let $\mathfrak{R}'$ be the reduced forest of F'.

We claim that $m(F') \leq m(F) - 1$. Let M be a maximum induced matching of F'. If M contains the edge v_1v_5, then we can obtain an induced matching for F by removing v_1v_5 and adding v_1v_2 and v_4v_5. If M does not contain v_1v_5, then one of v_1 and v_5 is not matched by M. Then for F, we can select one of v_2v_3 and v_3v_4 to increase the size of an induced matching. Thus, we have $m(F') \leq m(F) - 1$. By the induction hypothesis, $\mathfrak{R}'$ contains at most $6(p-1)$ vertices, and thus $\mathfrak{R}$ contains at most $6(p-1) + 3 = 6p - 3$ vertices. We may assume that there is no such path.

Let C be a connected component of F, and $\mathfrak{R}_C$ be the reduced forest of C. As $\mathfrak{R}_C$ contains at least 3 vertices, the leaves of $\mathfrak{R}_C$ form an independent set. Let t be the number of leaves in $\mathfrak{R}_C$. Since each leaf of $\mathfrak{R}_C$ is adjacent to a leaf of C, C contains an induced matching of size at least t. Thus, $m(F - V(C)) \leq m(F) - t$. Note that $\mathfrak{R}_C$ contains at most t vertices of degree at least 3. Also, by the previous argument, there are at most 2 vertices between two vertices of degree other than 2 in $\mathfrak{R}_C$. Thus, $\mathfrak{R}_C$ contains at most $t + t + 2(2t-1) \leq 6t$ vertices. Therefore, the result follows by the induction hypothesis. □

Let (A, B) be a vertex partition of a graph G, R be some induced forest in $G_{A,B}$, and M a minimal vertex cover of $G_{A,B} - V(R)$. In the algorithm, we want to ask if there exists an induced forest F in $G[A \cup \mathrm{bd}(B)]$ such that R is a reduced forest of $F \cap G_{A,B}$ and F avoids the vertices in M. However, it turns out that in this direct formulation it is difficult to account for edges between vertices in $\mathrm{bd}(B)$. We therefore define the following notion on an induced forest in $G[A \cup \mathrm{bd}(B)] - E(G[\mathrm{bd}(B)])$, instead of $G[A \cup \mathrm{bd}(B)]$.

Definition 9 (Forest respecting a given forest and a minimal vertex cover). Let (A, B) be a vertex partition of a graph G. Let R be an induced forest in $G_{A,B}$ and M be a minimal vertex cover of $G_{A,B} - V(R)$. An induced forest F in $G[A \cup \mathrm{bd}(B)] - E(G[\mathrm{bd}(B)])$ *respects* (R, M) if it satisfies the following.

1. R is a reduced forest of $G_{A,B} \cap F$.
2. $V(F) \cap M = \emptyset$.

Suppose R is an induced forest in $G_{A,B}$. For an induced forest F of G containing $V(R)$, there are two necessary conditions for R to be a reduced forest of $F \cap G_{A,B}$. First, in $F \cap G_{A,B}$, every vertex in $V(F \cap G_{A,B}) \setminus V(R)$ has at most one neighbor in R; otherwise, when we take a reduced forest of $F \cap G_{A,B}$, this vertex should remain. Second, in $F \cap G_{A,B}$, every leaf x of R should have a neighbor y in $V(F \cap G_{A,B}) \setminus V(R)$ (and the only neighbor of y in R is x); otherwise, we would have removed x when taking a reduced forest.

Motivated by this observation we define the notion of potential leaves, which is a possible leaf neighbor of some vertex in $V(R)$. See Figure 1 for an illustration.

Definition 10 (Potential Leaves). Let (A, B) be a vertex partition of a graph G. Let R be an induced forest in $H := G_{A,B}$ and M be a minimal vertex cover of $H - V(R)$. For each vertex $x \in V(R)$, we define its set of *potential leaves* with respect to R and M, denoted by $PL_{R,M}(x)$, as

$$PL_{R,M}(x) := N_H(x) \setminus (N_H(V(R) \setminus \{x\}) \cup (M \cup V(R))).$$

We can observe the following.

Observation 11. Every forest F respecting (R, M) contains at least one vertex in $PL_{R,M}(x)$ for each leaf x of R.

For a subset A' of A, we consider a pair of an induced forest R' and a minimal vertex cover M' of $G_{A',V(G)\setminus A'} - V(R')$ and we say that this pair is a *restriction* of a pair of R and M for A, if they satisfy certain natural properties. In the dynamic programming algorithm, we use this notion to study the structure of partial solutions w.r.t. cuts corresponding to a node t and the children of t.

Definition 12 (Restriction of a reduced forest and a minimal vertex cover). Let (A_1, A_2, B) be a vertex partition of a graph G. Let R be an induced forest in $G_{A_1 \cup A_2, B}$ and M be a minimal vertex cover of $G_{A_1 \cup A_2, B} - V(R)$. An induced forest R_1 in $G_{A_1, A_2 \cup B}$ and a minimal vertex cover M_1 of $G_{A_1, A_2 \cup B} - V(R_1)$ are *restrictions* of R and M to $G_{A_1, A_2 \cup B}$ if they satisfy the following:

1. $V(R) \cap A_1 \subseteq V(R_1)$ and $V(R_1) \cap B \subseteq V(R)$.
2. Every vertex in $(V(R_1) \setminus V(R)) \cap A_1$ has at most one neighbor in $V(R) \cap B$.

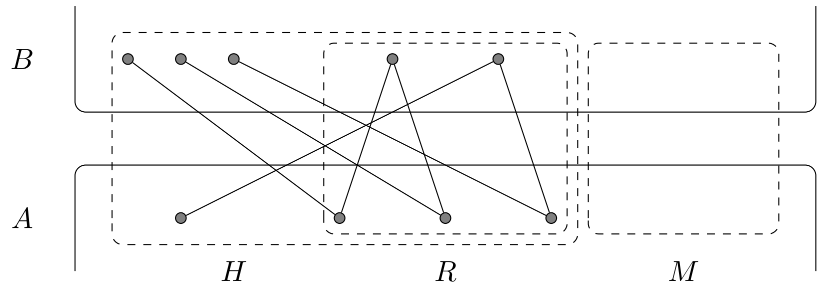

Figure 1: The graph R is an induced forest of $G_{A,B}$ and M is a minimal vertex cover of $G_{A,B} - V(R)$. Observe that R is a reduced forest of H. The four vertices in $V(H) \setminus V(R)$ are potential leaves with respect to R and M.

3. $M \cap A_1 \subseteq M_1$ and $M_1 \cap B \subseteq M$.

Lastly, we define a notion of compatibility of two partial solutions for the algorithm. To clarify, in the following definition, the partitions of the connected components of R_i represent connectivity information about induced forests in $G[A_i \cup \mathrm{bd}(A_{3-i} \cup B)] - E(G[\mathrm{bd}(A_{3-i} \cup B)])$ respecting R_i.

Definition 13 (Compatibility). Let (A_1, A_2, B) be a vertex partition of a graph G. Let R be an induced forest in $G_{A_1 \cup A_2, B}$, and for each $i \in \{1, 2\}$, let R_i be an induced forest in $G_{A_i, A_{3-i} \cup B}$, and P_i be a partition of $\mathcal{C}(R_i)$. We construct an auxiliary graph Q with respect to (R, R_1, R_2, P_1, P_2) in G as follows. Let Q be the graph on the vertex set $\mathcal{C}(R) \cup \mathcal{C}(R_1) \cup \mathcal{C}(R_2)$ such that

– for H_1 and H_2 in distinct sets of $\mathcal{C}(R), \mathcal{C}(R_1), \mathcal{C}(R_2)$, H_1 is adjacent to H_2 in Q if and only if $V(H_1) \cap V(H_2) \neq \emptyset$,

– for $i \in \{1, 2\}$ and the set of connected components contained in the same part of P_i, we add a path on the parts of P_i,

– $\mathcal{C}(R)$ is an independent set.

We say that the tuple (R, R_1, R_2, P_1, P_2) is *compatible* in G if Q has no cycles. We define $\mathcal{U}(R, R_1, R_2, P_1, P_2)$ to be the partition of $\mathcal{C}(R)$ such that for $H_1, H_2 \in \mathcal{C}(R)$, H_1 and H_2 are contained in the same part if and only if they are contained in the same connected component of Q.

The remainder of this section is devoted to proving three technical propositions related to the notions introduced above that will be important to establish the correctness of the algorithm proposed in Section 4. Let $t \in V(T)$ be an internal node in the given branch decomposition of G with children a and b. In Section 3.1 we show that given any forest F_t in $G[V_t \cup \mathrm{bd}(\overline{V_t})]$ respecting a pair (R_t, M_t), we can find restrictions (R_a, M_a) and (R_b, M_b) to $G_{a,\overline{a}}$ and $G_{b,\overline{b}}$, respectively, such that a forest F_a in $G[V_a \cup \mathrm{bd}(\overline{V_a})]$ respecting (R_a, M_a) and a forest F_b in $G[V_b \cup \mathrm{bd}(\overline{V_b})]$ respecting (R_b, M_b) can be combined to the forest F_t, i.e. we have that $F_t = F_a \cup F_b$. In Section 3.2 we prove the converse direction. For the sake of generality, we will state the results in terms of a 3-partition (A_1, A_2, B) rather than $(V_a, V_b, \overline{V_t})$ (i.e., independently of a branch decomposition of a graph).

3.1 Top to bottom

Proposition 14. *Let (A_1, A_2, B) be a vertex partition of a graph G. Let R be an induced forest in $G_{A_1 \cup A_2, B}$ and M be a minimal vertex cover of $G_{A_1 \cup A_2, B} - V(R)$. Let H be an induced forest in $G[A_1 \cup A_2 \cup \mathrm{bd}(B)] - E(G[\mathrm{bd}(B)])$ respecting (R, M).*

Then there are restrictions (R_1, M_1) and (R_2, M_2) of (R, M) to $G_{A_1, A_2 \cup B}$ and $G_{A_2, A_1 \cup B}$, respectively, such that

1. *for each $i \in \{1, 2\}$, $H \cap G[A_i \cup \mathrm{bd}(A_{3-i} \cup B)] - E(G[\mathrm{bd}(A_{3-i} \cup B)])$ respects (R_i, M_i),*
2. *every vertex in $(V(R) \setminus (V(R_1) \cup V(R_2))) \cap B$ has at least two neighbors in $(V(R_1) \cap A_1) \cup (V(R_2) \cap A_2)$, and*
3. *for each $i \in \{1, 2\}$, $V(R_i) \cap A_{3-i} \subseteq V(R_{3-i})$.*

Proof. Let $A = A_1 \cup A_2$ and $H_{A,B} = H \cap G_{A,B}$. For each $i \in \{1, 2\}$, let $F_i^* := H \cap G[A_i \cup \mathrm{bd}(A_{3-i} \cup B)] - E(G[\mathrm{bd}(A_{3-i} \cup B)])$, and let $F_i := F_i^* \cap G_{A_i, A_{3-i} \cup B}$, and let R_i be a reduced forest of F_i such that the following holds.

(Single-edge Rule I.) For a single-edge component vw of F_i with $v \in V(R)$ and $w \notin V(R)$, we select v as a vertex of R_i.

(Single-edge Rule II.) For an edge vw with $v \in A_1$, $w \in A_2$, and $v, w \notin V(R)$ that is a single-edge component in both F_1 and F_2, we select the same vertex as a vertex of R_i in both F_1 and F_2.

We first prove (2).

Claim 14.1. Every vertex in $(V(R) \setminus (V(R_1) \cup V(R_2))) \cap B$ has at least two neighbors in $(V(R_1) \cap A_1) \cup (V(R_2) \cap A_2)$.

Proof. Suppose there exists a vertex v in $(V(R) \setminus (V(R_1) \cup V(R_2))) \cap B$ having at most one neighbor in $(V(R_1) \cap A_1) \cup (V(R_2) \cap A_2)$. If $N_H(v)$ contains exactly one vertex w, then vw was a single-edge component of $H_{A,B}$; otherwise, v would have been removed while taking the reduced forest of $H_{A,B}$. But then $w \notin V(R)$ because $v \in V(R)$, and Single-edge rule I forces $v \in V(R_1) \cup V(R_2)$, a contradiction with the assumption. So v has at least two neighbors in $V(H) \cap (A_1 \cup A_2)$. Thus, v has a neighbor not contained in $(V(R_1) \cap A_1) \cup (V(R_2) \cap A_2)$. Let w be such a vertex, and without loss of generality, we assume that $w \in A_1$.

If v has a neighbor other than w in F_1, then v is contained in R_1. So, in F_1, w is the unique neighbor of v in $V(H) \cap A_1$. Also, since $w \notin V(R_1)$, v is the unique neighbor of w in F_1. Then vw is a single-edge component of F_1, and by Single-edge Rule I, we selected v as a vertex of R_1. This contradicts $v \notin V(R_1)$.

We conclude that every vertex in $(V(R) \setminus (V(R_1) \cup V(R_2))) \cap B$ has at least two neighbors in $(V(R_1) \cap A_1) \cup (V(R_2) \cap A_2)$. ◇

We prove (3).

Claim 14.2. For each $i \in \{1, 2\}$, $V(R_i) \cap A_{3-i} \subseteq V(R_{3-i})$.

Proof. Let $v \in V(R_i) \cap A_{3-i}$. As $v \in V(R_i)$, v has a neighbor w in F_i. Note that either v has at least two neighbors in F_i or vw is a single-edge component of F_i such that v is selected as a vertex of R_i.

Assume that v has at least two neighbors in F_i. By construction, edges between these two vertices and v are in H, and therefore, these two edges are also contained in F_{3-i} as well. Then since v has degree at least 2 in F_{3-i}, v is in R_{3-i}, as required.

Thus, we may assume that vw is a single-edge component of F_i. If $w \in V(R)$, then it should have a neighbor in B, which contradicts the fact that vw is a single-edge component of F_i. So, $w \notin V(R)$.

Note that vw may not be a single-edge component of F_{3-i} because of edges between A_2 and B. If $N_{F_{3-i}}(v)$ contains a vertex other than w, then v is selected as a vertex of R_{3-i} as w is a leaf of F_{3-i}. We may assume that w is the unique neighbor of v in F_{3-i}. In particular, $v \notin V(R)$. Since v is selected as a vertex of R_i, by Single-edge Rule II, v is also selected as a vertex of R_{3-i}. Thus, $v \in V(R_{3-i})$, as required. $\diamond$

In the remainder of this proof we show (1), i.e. that for each $i \in \{1, 2\}$, R_i is a restriction of R. We will construct a minimal vertex cover M_i later, after confirming first two conditions of Definition 12. We give the proof for $i = 1$; an analogous proof holds for $i = 2$.

Claim 14.3. $V(R) \cap A_1 \subseteq V(R_1)$.

Proof. Let $v \in V(R) \cap A_1$. Since $v \in V(R)$, v has at least one neighbor in $H_{A,B}$, and thus, v has at least one neighbor in F_1 on B as well. So, either v has degree at least 2 in F_1 or the unique neighbor of v in F_1 is its potential leaf with respect to (R, M) in $H_{A,B}$. In the former case, clearly v is contained in R_1, and in the latter case, v was chosen as a vertex of R_1 by Single-edge Rule I. $\diamond$

Claim 14.4. $V(R_1) \cap B \subseteq V(R)$.

Proof. It is sufficient to prove that every vertex in $(V(F_1) \setminus V(R)) \cap B$ is not contained in R_1. Suppose v is a vertex in $(V(F_1) \setminus V(R)) \cap B$. If v has degree at least 2 in $H_{A,B}$, then $v \in V(R)$, so we can assume that v has degree at most 1 in $H_{A,B}$. If v is isolated in F_1, then $v \notin V(R_1)$, so v has degree 1 in F_1. Let w be the neighbor of v in F_1. If w has degree at least 2 in F_1, then v is removed by definition of a reduced forest. So, vw is a single-edge component of F_1, and since $v \notin V(R)$, we have $w \in V(R)$. Thus, by Single-edge Rule I, we have that $v \notin V(R_1)$ and $w \in V(R_1)$. We conclude that $V(R_1) \cap B \subseteq V(R)$. $\diamond$

Claim 14.5. Every vertex in $(V(R_1) \setminus V(R)) \cap A_1$ has at most one neighbor in $V(R) \cap B$.

Proof. Suppose not and let $v \in (V(R_1) \setminus V(R)) \cap A_1$ such that v has two neighbors x and y in $V(R) \cap B$. Clearly, $\{v, x, y\} \subseteq V(H)$. But then, $v \in V(R)$ by the definition of reduced forests, a contradiction. $\diamond$

We now construct a minimal vertex cover M_1 of $G_{A_1,A_2 \cup B} - V(R_1)$ which avoids F_1, and verify the third condition of Definition 12. See Figure 2 for an illustration of Y and Z that are constructed below.

Note that there may be an edge between $(V(R) \setminus V(R_1)) \cap B$ and $A_1 \setminus V(F_1) \setminus (M \cap A_1)$, which is not hit by M. For example, it is possible that a vertex $a \in A_1 \setminus V(F_1) \setminus (M \cap A_1)$ and a vertex

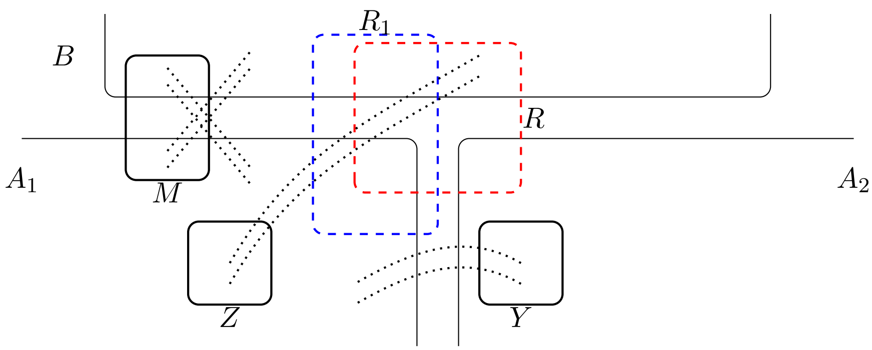

Figure 2: An illustration of Y and Z.

$b \in (V(R) \setminus V(R_1)) \cap B$ are adjacent in G (but not in H) and a was a potential leaf of b with respect to R and M, but b has only neighbors on A_2 in $H_{A,B}$, so that $b \in V(R)$. In this case, when we look at $G_{A_1,A_2 \cup B} - V(R_1)$, a and b are not contained in $V(R_1)$ and a is not contained in $M \cap A_1$. To hit such edges, we define Z as the set of all vertices in $A_1 \setminus V(F_1) \setminus (M \cap A_1)$ having a neighbor in $(V(R) \setminus V(R_1)) \cap B$.

We also need to hit all edges between A_1 and A_2 in $G_{A_1,A_2 \cup B} - V(R_1)$. We use vertices in A_2 to hit these edges. We define Y to be the set of all vertices in $A_2 \setminus V(F_1)$ having a neighbor in $A_1 \setminus V(R_1) \setminus (M \cap A_1)$.

Let $M' := M \cup Y \cup Z$. We first show that M' is a vertex cover of $G_{A_1,A_2 \cup B} - V(R_1)$.

Claim 14.6. The set M' is a vertex cover of $G_{A_1,A_2 \cup B} - V(R_1)$.

Proof. Suppose there is an edge yz in $G_{A_1,A_2 \cup B} - V(R_1)$ not covered by M'. As Y hits all edges between A_1 and A_2 in $G_{A_1,A_2 \cup B} - V(R_1)$, this edge is an edge between A_1 and B. Assume that $y \in A_1$ and $z \in B$.

As $V(R) \cap A_1 \subseteq V(R_1)$, z cannot be in $B \setminus (V(R) \cup M)$, and thus, $z \in (V(R) \setminus V(R_1)) \cap B$. However, since Z covers all edges between $A_1 \setminus V(R_1) \setminus (M \cap A_1)$ and $(V(R) \setminus V(R_1)) \cap B$, y should be contained in Z, a contradiction. Therefore, M' is a vertex cover of $G_{A_1,A_2 \cup B} - V(R_1)$. $\diamond$

Now, we take a minimal vertex cover M_1 of $G_{A_1,A_2 \cup B} - V(R_1)$ contained in M'. Clearly, the set M_1 is a minimal vertex cover of $G_{A_1,A_2 \cup B} - V(R_1)$ satisfying that $M \cap A_1 \subseteq M_1$ and $M_1 \cap B \subseteq M$ by construction. So, M_1 satisfies the third condition of Definition 12 and (R_1, M_1) is a restriction of (R, M).

It remains to show that F_1^* respects (R_1, M_1). By construction, R_1 is a reduced forest of F_1 so we only have to show that that $V(F_1^*) \cap M_1 = \emptyset$, and in particular, by the construction, it suffices to prove that $Z \cap V(F_1^*) = \emptyset$.

Claim 14.7. $Z \cap V(F_1^*) = \emptyset$.

Proof. Suppose not; let $x \in Z \cap V(F_1^*)$. Because $x \notin V(F_1)$, x has no neighbor in B in $G[A_1 \cup \mathrm{bd}(B)]$. Therefore, $x \notin Z$, by definition. $\diamond$

We conclude that F_1^* respects (R_1, M_1). $\square$

Proposition 15. *Let (A_1, A_2, B) be a vertex partition of a graph G. Let R be an induced forest in $G_{A_1 \cup A_2, B}$ and M be a minimal vertex cover of $G_{A_1 \cup A_2, B} - V(R)$. Let H be an induced forest in $G[A_1 \cup A_2 \cup \mathrm{bd}(B)] - E(G[\mathrm{bd}(B)])$ respecting (R, M) and for each $i \in \{1, 2\}$,*

- *let (R_i, M_i) be a restriction of (R, M) that $H \cap G[A_i \cup \mathrm{bd}(A_{3-i} \cup B)] - E(G[\mathrm{bd}(A_{3-i} \cup B)])$ respects (guaranteed by Proposition 14), and*
- *let P_i be the partition of $\mathcal{C}(R_i)$ such that for $C, C' \in \mathcal{C}(R_i)$, C and C' are in the same part if and only if they are contained in the same connected component of $H \cap G[A_i \cup \mathrm{bd}(A_{3-i} \cup B)] - E(G[\mathrm{bd}(A_{3-i} \cup B)])$.*

Then (R, R_1, R_2, P_1, P_2) is compatible.

Proof. Let Q be the auxiliary graph of (R, R_1, R_2, P_1, P_2). It is not difficult to see that if Q contains a cycle, then H also contains a cycle, which leads to a contradiction. Thus, Q has no cycles. □

3.2 Bottom to top

Proposition 16. *Let (A_1, A_2, B) be a vertex partition of a graph G. Let R be an induced forest in $G_{A_1 \cup A_2, B}$ and M be a minimal vertex cover of $G_{A_1 \cup A_2, B} - V(R)$ such that for every vertex x of degree at most 1 in R, $PL_{R,M}(x) \neq \emptyset$. For each $i \in \{1, 2\}$,*

- *let R_i be an induced forest in $G_{A_i, A_{3-i} \cup B}$ and M_i be a minimal vertex cover of $G_{A_i, A_{3-i} \cup B} - V(R_i)$, and H_i be an induced forest in $G[A_i \cup \mathrm{bd}(A_{3-i} \cup B)] - E(G[\mathrm{bd}(A_{3-i} \cup B)])$ respecting (R_i, M_i),*
- *let P_i be the partition of $\mathcal{C}(R_i)$ such that for $C, C' \in \mathcal{C}(R_i)$, C and C' are in the same part if and only if they are contained in the same connected component of H_i,*
- *(R_i, M_i) is a restriction of (R, M).*

Furthermore,

- *for each $i \in \{1, 2\}$, $V(R_i) \cap A_{3-i} \subseteq V(R_{3-i})$,*
- *every vertex in $(V(R) \setminus (V(R_1) \cup V(R_2))) \cap B$ has at least two neighbors in $(V(R_1) \cap A_1) \cup (V(R_2) \cap A_2)$,*
- *(R, R_1, R_2, P_1, P_2) is compatible.*

Then there is an induced forest H in $G[A_1 \cup A_2 \cup \mathrm{bd}(B)] - E(G[\mathrm{bd}(B)])$ respecting (R, M) such that

- *$V(H) \cap (A_1 \cup A_2) = (V(H_1) \cap A_1) \cup (V(H_2) \cap A_2)$.*

Proof. For each $i \in \{1, 2\}$, we obtain H_i' from H_i by removing all vertices that are contained in $(A_{3-i} \cup B) \setminus V(R_i)$. This procedure achieves that $V(H_i') \cap V(G_{A_i, A_{3-i} \cup B}) = V(R_i) \cap V(G_{A_i, A_{3-i} \cup B})$. We take a new graph

$$H^* := G[V(H_1') \cup V(H_2') \cup V(R)].$$

As (R, R_1, R_2, P_1, P_2) is compatible, we can verify that H^* is a forest. Let H be the graph obtained from $H^* - (B \setminus V(R))$ by adding a potential leaf to each vertex in $V(R) \cap (A_1 \cup A_2)$ of degree at most 1 in R and then removing newly created edges between vertices contained in B. We show that H is a forest.

Claim 16.1. H is a forest such that $V(H) \cap (A_1 \cup A_2) = (V(H_1) \cap A_1) \cup (V(H_2) \cap A_2)$.

Proof. Since H^* is a forest, $H^* - (B \setminus V(R))$ is also a forest. Adding a potential leaf of a vertex in $V(R) \cap (A_1 \cup A_2)$ preserves the property of being a forest, as we removed all edges in B. When we take H from H^*, we only change the vertices in B. Also, for each $i \in \{1, 2\}$, we have that

- $V(R) \cap A_i \subseteq V(R_i) \subseteq V(H_i)$ by the first condition of Definition 12, and
- $V(R_i) \cap A_{3-i} \subseteq V(R_{3-i})$ by the precondition of this proposition.

Therefore, we have $V(H) \cap (A_1 \cup A_2) = (V(H_1) \cap A_1) \cup (V(H_2) \cap A_2)$. $\diamond$

In the remainder, we prove that H respects (R, M). We need to verify that

1. R is a reduced forest of $G_{A_1 \cup A_2, B} \cap H$.
2. $V(H) \cap M = \emptyset$.

Condition (2) is easy to verify. Since we remove all vertices in $M \cap B \subseteq B \setminus V(R)$ when we construct H from H^* and then add only potential leaves with respect to R and M, we have $V(H) \cap (M \cap B) = \emptyset$. Furthermore, $V(H) \cap (M \cap (A_1 \cup A_2)) = \emptyset$ because

- $V(H) \cap (A_1 \cup A_2) = (V(H_1) \cap A_1) \cup (V(H_2) \cap A_2)$,
- for each $i \in \{1, 2\}$, $M \cap A_i \subseteq M_i$ by the third condition of Definition 12.

We now verify condition (1). Let $H_{A,B} := H \cap G_{A_1 \cup A_2, B}$. We first verify the following.

Claim 16.2. Every vertex of $V(H_{A,B}) \setminus V(R)$ has degree at most 1 in $H_{A,B}$.

Proof. Let $v \in V(H_{A,B}) \setminus V(R)$. First assume that $v \in A_1 \cup A_2$. Without loss of generality, we assume that $v \in A_1$. Since M is a vertex cover of $G_{A_1 \cup A_2, B} - V(R)$, the neighborhood of v in $H_{A,B}$ is contained in $V(R) \cap B$.

Suppose for contradiction that in $H_{A,B}$, v has at least two neighbors in $V(R) \cap B$. Since (R_1, M_1) is a restriction of (R, M), by the second condition of Definition 12, v is not contained in R_1. If v has at least two neighbors in $V(R_1) \cap B$, then v should be contained in R_1, a contradiction. Therefore, v has at least one neighbor in $(V(R) \setminus V(R_1)) \cap B$, say w. Then vw is an edge of $H_1 \cap G_{A_1, A_2 \cup B} - V(R_1)$, which contradicts the assumption that R_1 is a reduced forest of $H_1 \cap G_{A_1, A_2 \cup B}$. Therefore, v has at most one neighbor in $V(R) \cap B$, as required.

Now we assume $v \in B$. By construction, v is a potential leaf of some vertex in R. Thus v has degree 1 in $H_{A,B}$, as required. $\diamond$

We argue that R is a reduced forest of $H_{A,B}$. Let $v \in V(R)$. If v has degree at least 2 in $H_{A,B}$, then v is contained in any reduced forest of $H_{A,B}$. Suppose v has degree at most 1 in $H_{A,B}$.

Suppose $v \in A_1 \cup A_2$. In this case, by construction, v is incident with its potential leaf in $H_{A,B}$, say w. This means that vw is a single-edge component in $H_{A,B}$, and we can take v as a vertex in R.

Now, suppose $v \in B$. First assume that $v \in V(R_i)$ for some $i \in \{1, 2\}$. If v has degree 1 in R_i, then it also has at least one potential leaf in $H_i \cap G_{A_i, A_{3-i} \cup B}$, and thus v has degree 2 in $H_{A,B}$, a contradiction. Thus, v has no neighbor in R_i, and has exactly one potential leaf, say w. By

Claim 16.2, v is the unique neighbor of w in R, and thus vw is a single-edge component of $H_{A,B}$. Thus, we can take v as a vertex in R. Suppose $v \in (V(R) \setminus (V(R_1) \cup V(R_2))) \cap B$. Then by the precondition, it has at least two neighbors in $(V(R_1) \cap A_1) \cup (V(R_2) \cap A_2) \subseteq (V(H_1) \cap A_1) \cup (V(H_2) \cap A_2)$. Therefore, it is contained in any reduced forest of $H_{A,B}$. With Claim 16.1, it shows that R is a reduced forest of $H_{A,B}$. □

4 Feedback Vertex Set on graphs of bounded mim-width

In this section we give an algorithm that solves the Feedback Vertex Set problem on graphs on n vertices together with one of its branch decomposition of mim-width w in time $n^{\mathcal{O}(w)}$. We first give an algorithm for the unweighted version of the problem and then argue how to modify it for the weighted version.

First, we observe that given a graph G, a subset of its vertex set $S \subseteq V(G)$ is by definition a feedback vertex set if and only if $G - S$ is an induced forest; that is, $V(G) \setminus S$ induces a forest. It is therefore readily seen that computing the minimum size of a feedback vertex set is equivalent to computing the maximum size of an induced forest, so in the remainder of this section we solve the following problem which is more convenient for our exposition.

Maximum Induced Forest
Input: A graph G on n vertices, a branch decomposition $(T, \mathcal{L})$ of G and an integer k.
Parameter: $w := \text{mimw}(T, \mathcal{L})$.
Question: Does G contain an induced forest having at least k vertices?

We furthermore assume that G is connected; otherwise, we can solve it for each connected component. Also, we assume that G contains at least 2 vertices.

We solve the Maximum Induced Forest problem by bottom-up dynamic programming over $(T, \mathcal{L})$, the given branch decomposition of G, starting at the leaves of T. Let $t \in V(T)$ be a node of T. To motivate the table indices of the dynamic programming table, we now observe how a solution to Maximum Induced Forest, an induced forest $\mathcal{F}$, interacts with the graph $G_{t+\text{bd}} := G[V_t \cup \text{bd}(\overline{V_t})] - E(G[\text{bd}(\overline{V_t})])$. The intersection of $\mathcal{F}$ with $G_{t+\text{bd}}$ is an induced forest which throughout the following we denote by $\mathcal{F}_{t+\text{bd}} := \mathcal{F} \cap G_{t+\text{bd}}$. Since we want to bound the number of table entries by $n^{\mathcal{O}(w)}$, we have to focus in particular on the interaction of $\mathcal{F}$ with the crossing graph $G_{t,\bar{t}}$ which is an induced forest in $G_{t,\bar{t}}$, denoted by $\mathcal{F}_{t,\bar{t}} := \mathcal{F}[V(G_{t,\bar{t}})]$.

However, it is not possible to enumerate all induced forests in a crossing graph as potential table indices: Consider for example a star on n vertices and the cut consisting of the central vertex on one side and the remaining vertices on the other side. This cut has mim-value 1 but it contains 2^n induced forests, since each vertex subset of the star induces a forest on the cut. The remedy for this issue are *reduced* (induced) forests, introduced in Section 3.

In order to avoid having exponentially (in n) many table entries at each node $t \in V(T)$, we use all reduced forests of $G_{t,\bar{t}}$ to represent the ways in which induced forests can intersect with $G_{t,\bar{t}}$ as parts of the table entries. By Lemma 8, the number of reduced forests in each cut of mim-value w is bounded by $\mathcal{O}(n^{6w})$. However, reduced forests alone do not provide enough information about induced forests in $G_{t,\bar{t}}$. We now analyze the structure of $\mathcal{F}_{t,\bar{t}}$ to motivate the additional objects that can be used to represent $\mathcal{F}_{t,\bar{t}}$ in such a way that the number of all possible table entries remains bounded by $n^{\mathcal{O}(w)}$.

Let $\mathfrak{R}$ be a reduced forest of $\mathcal{F}_{t,\bar{t}}$. The induced forest $\mathcal{F}_{t,\bar{t}}$ has three types of vertices in $G_{t,\bar{t}}$:

– The vertices of the reduced forest $\mathfrak{R}$.

– The leaves of the induced forest $\mathcal{F}_{t,\bar{t}}$ that are not contained in $\mathfrak{R}$, denoted by $\mathrm{L}(\mathcal{F}_{t,\bar{t}})$.

– Vertices in $\mathcal{F}_{t,\bar{t}}$ that do not have a neighbor in $\mathcal{F}_{t,\bar{t}}$ on the opposite side of the boundary, in the following called *non-crossing* vertices and denoted by $\mathrm{NC}(\mathcal{F}_{t,\bar{t}})$.

As outlined above, the only type of vertices in $\mathcal{F}_{t,\bar{t}}$ that will be used as part of the table indices are the vertices of a reduced forest of $\mathcal{F}_{t,\bar{t}}$, since otherwise, the number of possible indices might be exponential in n. Hence, we neither know about the leaves of $\mathcal{F}_{t,\bar{t}}$ (apart from components that are single edges) nor its non-crossing vertices upon inspecting this part of the index. Suppose we have a vertex $v \in (\mathrm{L}(\mathcal{F}_{t,\bar{t}}) \cup \mathrm{NC}(\mathcal{F}_{t,\bar{t}})) \cap V_t$ and consider $N^*_{\bar{t}}(v) := (N_G(v) \cap \overline{V_t}) \setminus V(\mathfrak{R})$. Then, $\mathcal{F}_{t,\bar{t}}$ does not use any vertex x in $N^*_{\bar{t}}(v)$: If v is a leaf in $\mathcal{F}_{t,\bar{t}}$, then the presence of the edge $\{v, x\}$ would make it a non-leaf vertex and if v is a non-crossing vertex, the presence of $\{v, x\}$ would make v a vertex incident with an edge of the forest crossing the cut. An analogous point can be made for a vertex in $(\mathrm{L}(\mathcal{F}_{t,\bar{t}}) \cup \mathrm{NC}(\mathcal{F}_{t,\bar{t}})) \cap \overline{V_t}$. In the table indices, we capture this property of $\mathcal{F}_{t,\bar{t}}$ by considering a minimal vertex cover of $G_{t,\bar{t}} - V(\mathfrak{R})$ that avoids all leaves and non-crossing vertices of $\mathcal{F}_{t,\bar{t}}$. We observe that such a minimal vertex cover always exists. (Note that $\mathrm{L}(\mathcal{F}_{t,\bar{t}}) \cup \mathrm{NC}(\mathcal{F}_{t,\bar{t}})$ is an independent set in $G_{t,\bar{t}}$.)

Observation 17. Let G be a graph and $X \subseteq V(G)$ an independent set in G. Then, there exists a minimal vertex cover M of G such that $X \cap M = \emptyset$.

Lastly, we have to keep track of how the connected components of $\mathcal{F}_{t,\bar{t}}$ (respectively, $\mathfrak{R}$) are joined together via the forest $\mathcal{F}_{t+\mathrm{bd}}$. This forest induces a partition of $\mathcal{C}(\mathfrak{R})$ in the following way: Two components $C_1, C_2 \in \mathcal{C}(\mathfrak{R})$ are in the same part of the partition if and only if C_1 and C_2 are contained in the same connected component of $\mathcal{F}_{t+\mathrm{bd}}$.

We are now ready to define the indices of the dynamic programming table $\mathcal{T}$ to keep track of sufficiently much information about the partial solutions in the graph $G_{t+\mathrm{bd}}$. Throughout the following, we denote by $\mathcal{R}_t$ the set of all induced forests of $G_{t,\bar{t}}$ on at most $6w$ vertices (which by Lemma 8 contains all reduced forests in $G_{t,\bar{t}}$). For $R \in \mathcal{R}_t$, we let $\mathcal{M}_{t,R}$ be the set of all minimal vertex covers of $G_{t,\bar{t}} - V(R)$ and $\mathcal{P}_{t,R}$ the set of all partitions of the connected components of R.

For an illustration of the above discussion and also the definition of the table indices, which we start on now, see Figure 3. For $(R, M, P) \in \mathcal{R}_t \times \mathcal{M}_{t,R} \times \mathcal{P}_{t,R}$ and $i \in \{0, \ldots, n\}$, we set $\mathcal{T}[t, (R, M, P), i] := 1$ (and to 0 otherwise), if and only if the following conditions are satisfied.

1. There is an induced forest F in $G[V_t \cup \mathrm{bd}(\overline{V_t})] - E(G[\mathrm{bd}(\overline{V_t})])$, such that $V(F) \cap V_t$ has size i.

2. Let $F_{t,\bar{t}} = F \cap G_{t,\bar{t}}$, i.e. $F_{t,\bar{t}}$ is the subforest of F induced by the vertices of the crossing graph $G_{t,\bar{t}}$. Then, R is a reduced forest of $F_{t,\bar{t}}$.

3. M is a minimal vertex cover of $G_{t,\bar{t}} - V(R)$ such that $V(F) \cap M = \emptyset$.

4. P is a partition of $\mathcal{C}(R)$ such that two components $C_1, C_2 \in \mathcal{C}(R)$ are in the same part of the partition if and only if C_1 and C_2 are contained in the same connected component of F.

For a node $t \in V(T)$, we let $\mathcal{T}_t$ be the subtable of $\mathcal{T}$ with respect to t as the table entries that have t as a first index. I.e. for $(R, M, P) \in \mathcal{R}_t \times \mathcal{M}_{t,R} \times \mathcal{P}_{t,R}$ and $i \in \{0, \ldots, n\}$, we let $\mathcal{T}_t[(R, M, P), i] := \mathcal{T}[t, (R, M, P), i]$. Note that (2) and (3) express that F respects (R, M). Regarding (3), recall that

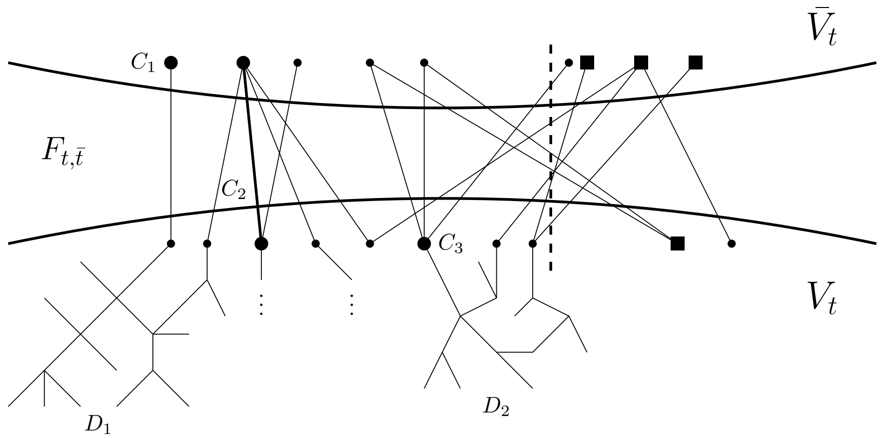

Figure 3: An example of a crossing graph $G_{t,\bar{t}}$ together with an induced forest $\mathcal{F}$ and their interaction. The forest $\mathcal{F}_{t,\bar{t}} = \mathcal{F}[V(G_{t,\bar{t}})]$ is displayed to the left of the dividing line in the drawing and the 4 vertices and 1 edge in bold form a reduced forest R of $\mathcal{F}_{t,\bar{t}}$. The square vertices form a minimal vertex cover of $G_{t,\bar{t}} - V(R)$ satisfying (3). Furthermore, C_i ($i \in [3]$) are the connected components of R and D_i ($i \in [2]$) are the connected components of $\mathcal{F}$.

even though the leaves and non-crossing vertices of $F_{t,\bar{t}}$ are still contained in $G_{t,\bar{t}} - V(R)$, a minimal vertex cover that avoids the leaves and non-crossing vertices of $F_{t,\bar{t}}$ always exists by Observation 17.

Recall that $r \in V(T)$ denotes the root of T, the tree of the given branch decomposition of G. From Property (1) we immediately observe that the table entries store enough information to obtain a solution to Maximum Induced Forest after all table entries have been filled. In particular, we make

Observation 18. The graph G contains an induced forest on i vertices if and only if $\mathcal{T}[r, (\emptyset, \emptyset, \emptyset), i] = 1$.

Before we proceed with the description of the algorithm, we first show that the number of table entries is bounded by a polynomial whose degree is linear in the mim-width w of the given branch decomposition.

Proposition 19. *There are at most $n^{\mathcal{O}(w)}$ table entries in $\mathcal{T}$.*

Proof. Let $t \in V(T)$. We show that the number of table entries in $\mathcal{T}_t$ is bounded by $n^{\mathcal{O}(w)}$ which together with the observation that $|V(T)| = \mathcal{O}(n)$ yields the proposition. By definition, $|\mathcal{R}_t| = \mathcal{O}(n^{6w})$ and by the Minimal Vertex Covers Lemma we have for each $R \in \mathcal{R}_t$ that $|\mathcal{M}_{t,R}| = n^{\mathcal{O}(w)}$. The size of $\mathcal{P}_{t,R}$ is at most the number of partitions of a set of size $6w$, and hence at most $B_{6w} < (w/\log(w))^{\mathcal{O}(w)}$ by standard upper bounds on the Bell number B_{6w}. Finally, there are $n+1$ choices for the integer i. To summarize, there are at most

$$\mathcal{O}\left(n^{6w}\right) \cdot n^{\mathcal{O}(w)} \cdot (w/\log(w))^{\mathcal{O}(w)} \cdot (n+1) = n^{\mathcal{O}(w)}$$

table entries in $\mathcal{T}_t$ and the proposition follows. □

We now show how to compute the table entries in $\mathcal{T}$. First, we explain how to compute the entries in $\mathcal{T}_\ell$ for the leaves ℓ of T and then how to compute the entries in the internal nodes of T from the entries stored in the tables corresponding to their children.

Leaves of T. Let $t \in V(T)$ be a leaf of T and $v = \mathcal{L}^{-1}(t)$. Clearly, the crossing graph $G_{t,\bar{t}}$ is a star S with central vertex v or a single edge. Hence, any induced forest F in $G[\{v\} \cup N(v)] - E(G[N(v)])$ satisfies that either $V(F) = \{v\}$ or $V(F) \subseteq N(v)$ or F contains an edge in $G_{t,\bar{t}}$. In the last case, either F is a single edge or a star with central vertex v. Let R be a reduced forest of F. The cases we have to consider to fill the table entries are the following.

If $F = \emptyset$, then both $\{v\}$ and $N(v)$ are feasible minimal vertex covers and clearly, $P = \emptyset$. If $V(F) = \{v\}$, then $R = \emptyset$, $M = N(v)$, $P = \emptyset$, and $i = 1$. If $V(F) \subseteq N(v)$, then $R = \emptyset$, $M = \{v\}$, $P = \emptyset$, and $i = 0$. Throughout the following, we assume that F contains an edge in $G_{t,\bar{t}}$.

Suppose F is a single edge $\{v, w\}$. Then, R is either the vertex v or the vertex w. If $V(R) = \{v\}$, then $G_{t,\bar{t}} - V(R)$ does not contain any edges and hence $\mathcal{M}_{t,R} = \{\emptyset\}$. Furthermore, F has size one in $G[V_t] = G[\{v\}]$. If $V(R) = \{w\}$, then v is a leaf in F and hence the only minimal vertex cover satisfying (3) is the set of neighbors of v without w, i.e. the set $N(v) \setminus \{w\}$. The size of F in $G[V_t]$ is 1. In both cases, F only has one component, so $\mathcal{P}_{t,R} = \{\{R\}\}$.

Now suppose that F has at least three vertices. Then, F is a star with central vertex v and hence, the reduced forest of any such F is the single vertex v. Since the vertices of F in $\overline{V_t}$ are not counted in the table entry by (1), we only have to consider one index where the reduced forest is v, the minimal vertex cover is empty (again since $G_{t,\bar{t}} - \{v\}$ does not have any edges), the partition of R is the singleton partition and $i = 1$, since F has size one in $G[V_t] = G[\{v\}]$. To summarize, the table entries for the leaf t are set as follows.

$$\mathcal{T}[t, (R, M, P), i] := \begin{cases} 1, & \text{if } R = \emptyset, M \in \{\{v\}, N(v)\}, P = \emptyset, i = 0 \\ 1, & \text{if } R = \emptyset, M = N(v), P = \emptyset, i = 1 \\ 1, & \text{if } R = G[\{v\}], M = \emptyset, P = \{R\}, i = 1 \\ 1, & \text{if } R = G[\{w\}] \text{ where } w \in N(v), M = N(v) \setminus \{w\}, \\ & P = \{R\}, i = 1 \\ 0, & \text{otherwise} \end{cases}$$

Internal Nodes of T. Let $t \in V(T)$ be an internal node with children a and b. Using Propositions 14, 15 and 16, we can show the following.

Proposition 20. *Let $\mathfrak{I} = [(R, M, P), i] \in (\mathcal{R}_t \times \mathcal{M}_{t,R_t} \times \mathcal{P}_{t,R_t}) \times \{0, \ldots, n\}$ such that for every vertex x of degree at most 1 in R, $PL_{R,M}(x) \neq \emptyset$. Then $\mathcal{T}[t, (R, M, P), i] = 1$ if and only if there are restrictions (R_a, M_a) and (R_b, M_b) of (R, M) to $G_{a,\bar{a}}$ and $G_{b,\bar{b}}$, respectively, and partitions P_a and P_b of $\mathcal{C}(R_a)$ and $\mathcal{C}(R_b)$, respectively, and integers i_a and i_b such that*

- $\mathcal{T}[t_a, (R_a, M_a, P_a), i_a] = 1$ *and* $\mathcal{T}[t_b, (R_b, M_b, P_b), i_b] = 1$,
- (R, R_a, R_b, P_a, P_b) *is compatible and* $P = \mathcal{U}(R, R_a, R_b, P_a, P_b)$,
- *every vertex in* $(V(R) \setminus (V(R_a) \cup V(R_b))) \cap B$ *has at least two neighbors in* $(V(R_a) \cap V_a) \cup (V(R_b) \cap V_b)$,
- $V(R_a) \cap V_b \subseteq V(R_b)$ *and* $V(R_b) \cap V_a \subseteq V(R_a)$,
- $i_a + i_b = i$.

Proof. Suppose $\mathcal{T}[t, (R, M, P), i] = 1$. Let H be an induced forest of $G[V_t \cup \mathrm{bd}(\overline{V_t})] - E(G[\mathrm{bd}(\overline{V_t})])$ that is a partial solution with respect to (R, M, P) and i. For each $x \in \{a, b\}$, let $H_x := H \cap (G[V_x \cup \mathrm{bd}(\overline{V_x})] - E(G[\mathrm{bd}(\overline{V_x})]))$. By Proposition 14, there are restrictions (R_a, M_a) and (R_b, M_b) of (R, M) to V_a and V_b, respectively, such that

– H_a respects (R_a, M_a), and H_b respects (R_b, M_b), and

– every vertex in $(V(R) \setminus (V(R_a) \cup V(R_b))) \cap B$ has at least two neighbors in $(V(R_a) \cap V_a) \cup (V(R_b) \cap V_b)$,

– $V(R_a) \cap V_b \subseteq V(R_b)$ and $V(R_b) \cap V_a \subseteq V(R_a)$.

For each $x \in \{a, b\}$, let P_x be the partition of $\mathcal{C}(R_x)$ such that two graphs in $\mathcal{C}(R_x)$ are contained in the same part if and only if they are contained in the same connected component of H_x. Then by Proposition 15, the tuple (R, R_a, R_b, P_a, P_b) is compatible and it is not difficult to verify that $P = \mathcal{U}(R, R_a, R_b, P_a, P_b)$. Let $i_x := |V(H) \cap V(G[V_x])|$. Then, $i_a + i_b = i$ as V_a and V_b are disjoint. This concludes the forward direction.

To verify the converse direction, suppose the latter conditions hold. For each $x \in \{a, b\}$, let H_x be an induced forest in $G[V_x \cup \mathrm{bd}(\overline{V_x})] - E(G[\mathrm{bd}(\overline{V_x})])$ that is a partial solution with respect to (R_x, M_x, P_x) and i_x. By the second, third, and fourth condition, we can apply Proposition 16 to conclude that there is an induced forest H in $G[V_t \cup \mathrm{bd}(\overline{V_t})] - E(G[\mathrm{bd}(\overline{V_t})])$ respecting (R, M) such that

$$H \cap G[V_t] = (H_a \cap G[V_a]) \cup (H_b \cap G[V_b]).$$

Therefore, we have $|V(H) \cap V_t| = |V(H_a) \cap V_a| + |V(H_b) \cap V_b| = i_a + i_b = i$, so $\mathcal{T}[t, (R, M, P), i] = 1$, as required. $\square$

Based on Proposition 20, we can proceed with the computation of the table at an internal node t with children a and b. Let $\mathfrak{I} = [(R, M, P), i] \in (\mathcal{R}_t \times \mathcal{M}_{t,R_t} \times \mathcal{P}_{t,R_t}) \times \{0, \ldots, n\}$.

Step 1 (Valid Index). We verify whether $\mathfrak{I}$ is valid, i.e. whether it can represent a valid partial solution in the sense of the definition of the table entries. That is, each vertex of degree at most 1 in R has to have at least one potential leaf.

Step 2 (Reduced Forests). We consider all pairs of indices for $\mathcal{T}_a$ and $\mathcal{T}_b$ denoted by

– $\mathfrak{I}_a = [(R_a, M_a, P_a), i_a] \in (\mathcal{R}_a \times \mathcal{M}_{a,R_a} \times \mathcal{P}_{a,R_a}) \times \{0, \ldots, n\}$ and
– $\mathfrak{I}_b = [(R_b, M_b, P_b), i_b] \in (\mathcal{R}_b \times \mathcal{M}_{b,R_b} \times \mathcal{P}_{b,R_b}) \times \{0, \ldots, n\}$.

We check

– (R_a, M_a) and (R_b, M_b) are restrictions of (R, M) to $G_{a,\overline{a}}$ and $G_{b,\overline{b}}$ respectively,
– $\mathcal{T}[t_a, (R_a, M_a, P_a), i_a] = 1$ and $\mathcal{T}[t_b, (R_b, M_b, P_b), i_b] = 1$,
– (R, R_a, R_b, P_a, P_b) is compatible and $P = \mathcal{U}(R, R_a, R_b, P_a, P_b)$,
– every vertex in $(V(R) \setminus (V(R_a) \cup V(R_b))) \cap B$ has at least two neighbors in $(V(R_a) \cap V_a) \cup (V(R_b) \cap V_b)$,
– $V(R_a) \cap V_b \subseteq V(R_b)$ and $V(R_b) \cap V_a \subseteq V(R_a)$,
– $i_a + i_b = i$.

If there are $\mathfrak{I}_a$ and $\mathfrak{I}_b$ satisfying all of the above conditions, then we assign $\mathcal{T}[t, (R, M, P), i] = 1$ and otherwise, we assign $\mathcal{T}[t, (R, M, P), i] = 0$. Correctness follows from Proposition 20.

We finish by analyzing the running time of the algorithm. At each node $t \in V(T)$, we can enumerate all table indices in time $n^{\mathcal{O}(w)}$ by Corollary 6 and Proposition 19. Let $\mathfrak{I} = [(R, M, P), i] \in (\mathcal{R}_t \times \mathcal{M}_{t,R_t} \times \mathcal{P}_{t,R_t}) \times \{0, \ldots, n\}$. If t is a leaf node, then $\mathcal{T}[t, (R, M, P), i]$ can be computed in linear time. Assume that t is an internal node. We can check in linear time whether $\mathfrak{I}$ is valid or not. Next, for all pairs of $\mathfrak{I}_a = [(R_a, M_a, P_a), i_a] \in (\mathcal{R}_a \times \mathcal{M}_{a,R_a} \times \mathcal{P}_{a,R_a}) \times \{0, \ldots, n\}$ and $\mathfrak{I}_b = [(R_b, M_b, P_b), i_b] \in (\mathcal{R}_b \times \mathcal{M}_{b,R_b} \times \mathcal{P}_{b,R_b}) \times \{0, \ldots, n\}$ we verify that the conditions of Step 2 hold, which can be done in time $\mathcal{O}(n^2)$. Therefore, by Proposition 19, we can decide whether $\mathcal{T}[t, (R, M, P), i] = 1$ or not in time $n^{\mathcal{O}(w)}$. As T contains $\mathcal{O}(n)$ nodes, we can solve Maximum Induced Forest, and by duality Feedback Vertex Set in time $n^{\mathcal{O}(w)}$.

We can easily modify our algorithm into an algorithm solving the weighted version of the problem. In Weighted Feedback Vertex Set, we are given a graph and a weight function $\omega : V(G) \to \mathbb{R}$, we want to find a set S with minimum $\omega(S)$ such that $G - S$ has no cycles. Similar to Feedback Vertex Set, we can instead solve the problem of finding an induced forest F with maximum $\omega(V(F))$. Instead of specifying i in the table index $[t, (R, M, P), i]$, we store at $\mathcal{T}[t, (R, M, P)]$ the maximum value $\omega(V(F) \cap V_t)$ over all induced forests F that respect (R, M) and whose connectivity partition is P. The procedure for leaf nodes is analogous. In the internal node, we compare all pairs (R_a, M_a, P_a) and (R_b, M_b, P_b) for children t_a and t_b, and take the maximum among all sums $\mathcal{T}[t_a, (R_a, M_a, P_a)] + \mathcal{T}[t_b, (R_b, M_b, P_b)]$. Therefore, we can solve Weighted Feedback Vertex Set (and Maximum Weight Induced Forest) in time $n^{\mathcal{O}(w)}$ as well. We have proved Theorem 1.

Our algorithm can furthermore be used to solve the connected variant of the Maximum (Weight) Induced Forest problem, namely Maximum (Weight) Induced Tree. To see this, note that one part of the table indices is the connectivity partition of all forests that correspond to a given index. Each part of this partition represents a connected component of a corresponding forest. Hence, we can solve Maximum (Weight) Induced Tree as follows. First, we compute all the table entries as when solving Maximum (Weight) Induced Forest. Then, when reading off the solution value to the problem in the table entries corresponding to the root of the branch decomposition, we simply restrict our search to table indices whose connectivity partitions consist of a single part: these entries are precisely the ones that correspond to solutions that form a tree.

Corollary 21. *Given an n-vertex graph and one of its branch decompositions of mim-width w, we can solve* Maximum (Weight) Induced Forest *and* Maximum (Weight) Induced Tree *in time $n^{\mathcal{O}(w)}$.*

5 W[1]-hardness results

We now prove that Feedback Vertex Set is W[1]-hard parameterized by mim-width, ruling out the possibility of FPT-algorithms for this parameterized problem under the standard assumption that FPT $\neq$ W[1]. Again we will prove our results by considering the Maximum Induced Forest problem, the dual to Feedback Vertex Set. Before we proceed, we will introduce some more preliminaries and notation. In particular, we introduce H-graphs which are crucially used in the reduction.

Throughout this section, for a graph G, we let $|G| := |V(G)|$ and $||G|| := |E(G)|$. Let $uv \in E(G)$. We call the operation of adding a new vertex x to $V(G)$ and replacing uv by the path uxv the *edge subdivision* of uv. We call a graph G' a *subdivision of* G if it can be obtained from G by a series of edge subdivisions.

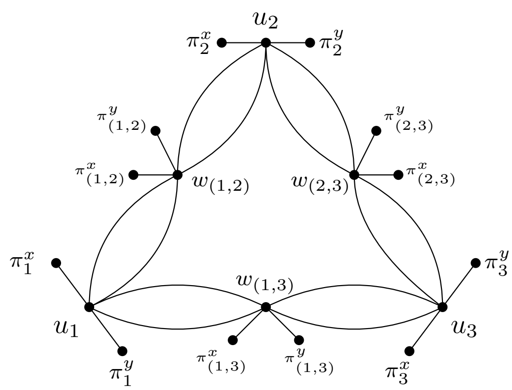

Figure 4: Illustration of the graph H for $k = 3$.

H-Graphs. Let X be a set and $\mathcal{S}$ be a family of subsets of X. The *intersection graph* of $\mathcal{S}$ is a graph with vertex set $\mathcal{S}$ such that $S, T \in \mathcal{S}$ are adjacent if and only if $S \cap T \neq \emptyset$. Let H be a (multi-) graph. We say that G is an *H-graph* if there are a subdivision H' of H and a family of subsets $\mathcal{M} := \{M_v\}_{v \in V(G)}$ (called an *H-representation*) of $V(H')$ where $H'[M_v]$ is connected for all $v \in V(G)$, such that G is isomorphic to the intersection graph of $\mathcal{M}$.

Fomin et al. [16] showed that H-graphs have linear mim-width at most $2 \cdot ||H||$ [16, Thm. 2] and that Independent Set is $\mathsf{W}[1]$-hard parameterized by $k + ||H||$, where k denotes the solution size [16, Thm. 17]. This implies that Independent Set is $\mathsf{W}[1]$-hard for the combined parameter solution size plus linear mim-width [16, Cor. 19]. We will modify their reduction to show that Maximum Induced Forest parameterized by the mim-width of a given linear branch decomposition plus the solution size remains $\mathsf{W}[1]$-hard.

The reduction is from Multicolored Clique where given a graph G and a partition $V_1, \ldots, V_k$ of $V(G)$, the question is whether G contains a clique of size k using precisely one vertex from each V_i ($i \in \{1, \ldots, k\}$). This problem is known to be $\mathsf{W}[1]$-complete parameterized by k [13, 30].

Theorem 22. Maximum Induced Forest *on H-graphs is* $\mathsf{W}[1]$*-hard parameterized by* $k + ||H||$*, where k denotes the solution size, and the hardness holds even when an H-representation of the input graph is given.*

Proof. Let $(G, V_1, \ldots, V_k)$ be an instance of Multicolored Clique. We can assume that $k \geq 2$ and that $|V_i| = p$ for $i \in [k]$. If the second assumption does not hold, let $p := \max_{i \in [k]} |V_i|$ and add $p - |V_i|$ isolated vertices to V_i, for each $i \in [k]$; we denote by $v_1^i, \ldots, v_p^i$ the vertices of V_i.

We obtain an H-graph G' from an adapted version of the construction due to Fomin et al. [16, Proof of Thm. 17]. The graph H is obtained as follows, see Figure 4 for an illustration.[6]

1. Construct k nodes $u_1, \ldots, u_k$.

2. For every $1 \leq i < j \leq k$, construct a node $w_{i,j}$ and two pairs of parallel edges $u_i w_{i,j}$ and $u_j w_{i,j}$.

[6]We would like to stress that the reduction given here is closely inspired by the one due to Fomin, Golovach and Raymond [16]. The main difference in the construction of H and the resulting H-graph G' revolves around introducing the new vertices to H in Steps 3 and 4 below which are key to fit the reduction for Maximum Induced Forest. Note also that the subdivisions described below are the same as in [16].

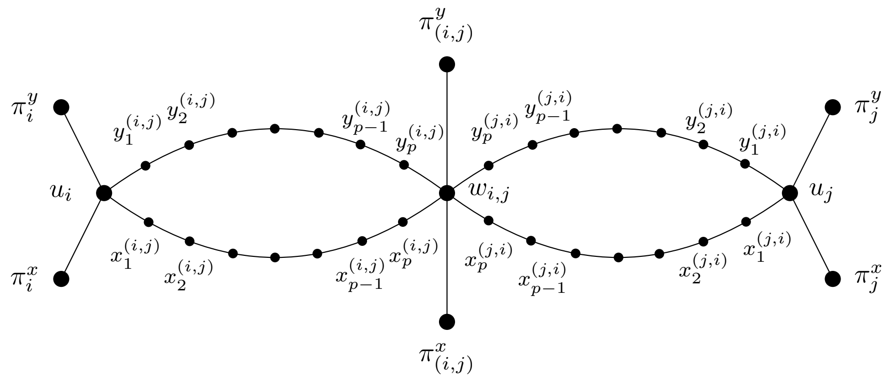

Figure 5: A part of the subdivision H' of H, where $1 \leq i < j \leq k$.

3. For each $i \in [k]$, add to H two neighbors π_i^x and π_i^y of u_i.

4. For each $1 \leq i < j \leq k$, add to H two neighbors $\pi^x_{(i,j)}$ and $\pi^y_{(i,j)}$ of $w_{(i,j)}$.

We let $\Pi \coloneqq \bigcup_{i\in[k]}\{\pi_i^x, \pi_i^y\} \cup \bigcup_{1\leq i<j\leq k}\{\pi^x_{(i,j)}, \pi^y_{(i,j)}\}$. Note that $|H| = (3/2)k(k+1)$ and

$$||H|| = k(3k-1). \tag{1}$$

We obtain a subdivision H' of H by subdividing each edge in $E(G-\Pi)$ p times. We denote the subdivision nodes obtained from subdividing the edges added in Step 2 as follows. Let $1 \leq i < j \leq k$ and consider the pair of edges between u_i and $w_{i,j}$. We denote the subdivision nodes corresponding to the first edge in that pair by $x_1^{(i,j)}, \ldots, x_p^{(i,j)}$, and the subdivision nodes corresponding to the second edge in that pair by $y_1^{(i,j)}, \ldots, y_p^{(i,j)}$. Similarly, for the pair of edges between u_j and $w_{i,j}$, we denote the subdivision nodes corresponding to the first edge in that pair by $x_1^{(j,i)}, \ldots, x_p^{(j,i)}$, and the subdivision nodes corresponding to the second edge in that pair by $y_1^{(j,i)}, \ldots, y_p^{(j,i)}$. To simplify notation, we assume that $u_i = x_0^{(i,j)} = y_0^{(i,j)}$, $u_j = x_0^{(j,i)} = y_0^{(j,i)}$ and $w_{i,j} = x_{p+1}^{(i,j)} = y_{p+1}^{(i,j)} = x_{p+1}^{(j,i)} = y_{p+1}^{(j,i)}$. We illustrate this subdivision process in Figure 5.

We now construct the H-graph G' by defining its H-representation $\mathcal{M} = \{M_v\}_{v\in V(G')}$ where each M_v is a connected subset of $V(H')$. (Recall that G denotes the graph of the Multicolored Clique instance.)

1. For each $i \in [k]$ and $s \in [p]$, we add a vertex z_s^i (representing vertex v_s^i from G) whose model is

$$M_{z_s^i} \coloneqq \{\pi_i^x, \pi_i^y\} \cup \bigcup\nolimits_{j\in[k], j\neq i} \left(\left\{ x_0^{(i,j)}, \ldots, x_{s-1}^{(i,j)} \right\} \cup \left\{ y_0^{(i,j)}, \ldots, y_{p-s}^{(i,j)} \right\} \right).$$

2. For each $i \in [k]$, construct vertices α_i^x with model $M_{\alpha_i^x} \coloneqq \{\pi_i^x\}$ and α_i^y with model $M_{\alpha_i^y} \coloneqq \{\pi_i^y\}$.

3. For each edge $v_s^i v_t^j \in E(G)$ for $s,t \in [p]$ and $1 \leq i < j \leq k$, construct a vertex $r_{s,t}^{(i,j)}$ with

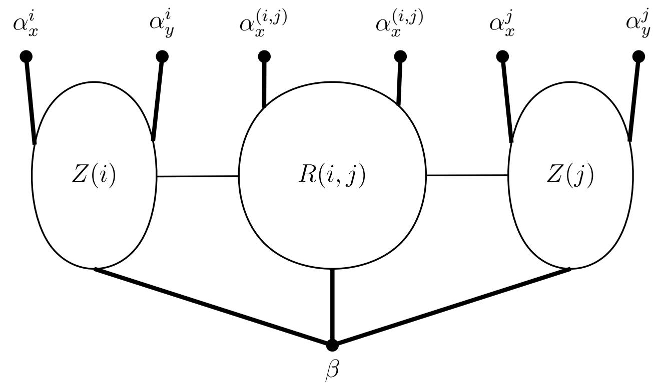

Figure 6: Illustration of a part of G', where $1 \leq i < j \leq k$. Bold edges imply that all possible edges between the corresponding (sets of) vertices are present. Non-bold edges mean that *some* of the edges between the two sets of vertices are present, depending on the construction.

model

$$M_{r_{s,t}^{(i,j)}} := \{\pi_i^x, \pi_i^y\} \cup \left\{x_s^{(i,j)}, \ldots, x_{p+1}^{(i,j)}\right\} \cup \left\{y_{p-s+1}^{(i,j)}, \ldots, y_{p+1}^{(i,j)}\right\}$$
$$\cup \left\{x_t^{(j,i)}, \ldots, x_{p+1}^{(j,i)}\right\} \cup \left\{y_{p-t+1}^{(j,i)}, \ldots, y_{p+1}^{(j,i)}\right\}.$$

4. For each $1 \leq i < j \leq k$, construct vertices $\alpha_x^{(i,j)}$ with model $M_{\alpha_x^{(i,j)}} := \{\pi_{(i,j)}^x\}$ and $\alpha_y^{(i,j)}$ with model $M_{\alpha_y^{(i,j)}} := \{\pi_{(i,j)}^y\}$.
5. Construct a vertex β with model $M_\beta := V(H) \setminus \Pi$.

Throughout the following, for $i \in [k]$ and $1 \leq i < j \leq k$, respectively, we use the notation

$$Z(i) := \bigcup_{s \in [p]} \left\{z_s^i\right\} \text{ and } R(i,j) := \bigcup_{\substack{v_s^i v_t^j \in E(G),\\ s,t \in [p]}} \left\{r_{s,t}^{(i,j)}\right\}$$

and we let $Z_{+\alpha}(i) := Z(i) \cup \{\alpha_x^i, \alpha_y^i\}$ and $R_{+\alpha}(i,j) := R(i,j) \cup \{\alpha_x^{(i,j)}, \alpha_y^{(i,j)}\}$. We furthermore define

$$A := \bigcup_{i \in [k]} \left\{\alpha_x^i, \alpha_y^i\right\} \cup \bigcup_{1 \leq i < j \leq k} \left\{\alpha_x^{(i,j)}, \alpha_y^{(i,j)}\right\}.$$

We continue with some observations about the global structure of G'.

Observation 22.1. Let $1 \leq i < j \leq k$ (wherever required).

1. *$N(\alpha_x^i) = Z(i) = N(\alpha_y^i)$, $N(\alpha_x^{(i,j)}) = R(i,j) = N(\alpha_y^{(i,j)})$, and $N(\beta) = V(G') \setminus A$.*
2. *$Z(i)$ induces a clique in G' and $R(i,j)$ induces a clique in G'.*
3. *A is an independent set in G' of size $2k + 2 \cdot \binom{k}{2}$.*

By Observation 22.1, the structure of the graph G' can be illustrated as shown in Figure 6. The following observation about edges between $Z(i)$ (respectively, $Z(j)$) and $R(i,j)$ (for $1 \leq i < j \leq k$) is crucial for this reduction.

Observation 22.2 (Claim 18 in [16]). For every $1 \le i < j \le k$, a vertex $z_h^i \in V(G')$ (a vertex $z_h^j \in V(G')$) is not *adjacent to a vertex $r_{s,t}^{(i,j)}$ corresponding to the edge $v_s^i v_t^j \in E(G)$ if and only if $h = s$ ($h = t$, respectively).*

We are now ready to prove the correctness of the reduction. In particular we will show that G has a multicolored clique if and only if G' has an induced forest of size $k' := 3k + 3\binom{k}{2} + 1$.

Claim 22.3. If G has a multicolored clique on vertex set $\left\{v_{h_1}^1, \ldots, v_{h_k}^k\right\}$, then G' has an induced forest of size $k' = 3k + 3 \cdot \binom{k}{2} + 1$.

Proof. Using Observation 22.2, one can easily verify that the set

$$I := \left\{z_{h_1}^1, \ldots, z_{h_k}^k\right\} \cup \left\{r_{h_i,h_j}^{(i,j)} \mid 1 \le i < j \le k\right\} \tag{2}$$

is an independent set in G'. By Observation 22.1(3) and the construction given above, we can conclude that $F := I \cup A \cup \{\beta\}$ induces a forest in G': I and A are both independent sets and $A \cup I$ induces a disjoint union of paths on three vertices, the middle vertices of which are contained in I. The only additional edges that are introduced are between β and vertices in I, so F induces a tree. Clearly, $|F| = |I| + |A| + |\{\beta\}| = k + \binom{k}{2} + 2k + 2 \cdot \binom{k}{2} + 1 = k'$, proving the claim. $\diamond$

We now prove the backward direction of the correctness of the reduction. This will be done by a series of claims and observations narrowing down the shape of any induced forest on k' vertices in G'. Eventually, we will be able conclude that any such induced forest contains an independent set of size $k + \binom{k}{2}$ of the shape (2). We can then conclude that G contains a multicolored clique by Observation 22.2.

The following is a direct consequence of Observation 22.1(2).

Observation 22.4. Let F be an induced forest in G'. Then, $V(F)$ contains

1. *at most 2 vertices from $Z(i)$, where $i \in [k]$ and*
2. *at most 2 vertices from $R(i,j)$, where $1 \le i < j \le k$.*

Next, we investigate the interaction of any induced forest with the sets $Z_{+\alpha}(i)$ and $R_{+\alpha}(i,j)$.

Claim 22.5. Let F be an induced forest in G'. If $V(F)$ contains two vertices from $Z(i)$, where $i \in [k]$ (from $R(i,j)$, where $1 \le i < j \le k$), then $V(F)$ cannot contain a vertex from $\{\alpha_x^i, \alpha_y^i\}$ (from $\{\alpha_x^{(i,j)}, \alpha_y^{(i,j)}\}$, respectively).

Proof. Suppose $V(F)$ contains two vertices $a, b \in Z(i)$. We prove the claim for α_x^i and note that the same holds for α_y^i. By Observation 22.1(2), a and b are adjacent and α_x^i is adjacent to both a and b by Observation 22.1(1). Hence, $\{\alpha_x^i, a, b\}$ induces a 3-cycle in G'.

An analogous argument can be given for the second statement. $\diamond$

In the light of Observation 22.4 and Claim 22.5, we make

Observation 22.6. Let F be an induced forest in G'. If $V(F)$ contains three vertices from $Z_{+\alpha}(i)$ for some $i \in [k]$ (three vertices from $R_{+\alpha}(i,j)$, respectively), then this set of three vertices must include α_x^i and α_y^i (resp., $\alpha_x^{(i,j)}$ and $\alpha_y^{(i,j)}$).

The previous observation implies that in G', any induced forest on $k' = 3k + 3 \cdot \binom{k}{2} + 1$ has the following form.

(I). For each $i \in [k]$, $V(F) \cap Z_{+\alpha}(i) = \{\alpha_x^i, \alpha_y^i, z_s^i\}$, for some $s \in [p]$.

(II). For each $1 \leq i < j \leq k$, $V(F) \cap R_{+\alpha}(i,j) = \{\alpha_x^{(i,j)}, \alpha_y^{(i,j)}, r_{t,t'}^{(i,j)}\}$, for some $t, t' \in [p]$.

(III). $\beta \in V(F)$.

To conclude the proof, we argue that any such induced forest F includes an independent set of size $k + \binom{k}{2}$ of the form (2). In particular, we use the following claim to establish the correctness of the reduction.

Claim 22.7. Let F be an induced forest in G' on k' vertices, $1 \leq i < j \leq k$ and $s_i, s_j, t_i, t_j \in [p]$. If $z_{s_i}^i, r_{t_i,t_j}^{(i,j)}, z_{s_j}^j \in V(F)$, then $s_i = t_i$ and $s_j = t_j$.

Proof. Suppose not and assume wlog. that $s_i \neq t_i$. Recall that by ((III)), we can assume that $\beta \in V(F)$, and by construction, β is adjacent to all vertices in $Z(i)$ and $R(i,j)$, so in particular β is adjacent to $z_{s_i}^i$ and $r_{t_i,t_j}^{(i,j)}$. However, by Observation 22.2 and the assumption that $s_i \neq t_i$, we have that $z_{s_i}^i r_{t_i,t_j}^{(i,j)} \in E(G')$, hence $\left\{\beta, z_{s_i}^i, r_{t_i,t_j}^{(i,j)}\right\}$ induces a cycle in F, a contradiction. $\diamond$

Since by ((I)) and ((II)), any induced forest on k' vertices contains precisely one vertex from each $Z(i)$ (for $i \in [k]$) and $R(i,j)$ (for $1 \leq i < j \leq k$), we can conclude together with Claim 22.7 that $V(F)$ contains an independent set

$$\left\{z_{s_1}^1, \ldots, z_{s_k}^k\right\} \cup \left\{r_{s_i,s_j}^{(i,j)} \mid 1 \leq i < j \leq k\right\}$$

which by Observation 22.2 implies that G has a clique on vertex set $\left\{v_{s_1}^1, \ldots, v_{s_k}^k\right\}$ which proves the correctness of the reduction.

Finally, since the size of G' is polynomial in the size of G, $k' = 3k + 3 \cdot \binom{k}{2} + 1$, and $||H|| = k(3k-1)$ (see Eq. 1), we can conclude that Maximum Induced Forest on H-graphs is W[1]-hard parameterized by $k + ||H||$. $\square$

As in both directions of the correctness proof in the above reduction, the solution to Maximum Induced Forest is connected, it shows hardness for the Maximum Induced Tree problem in the same parameterization as well. Furthermore, since a graph on n vertices has an induced forest of size k if and only if it has a feedback vertex set of size $n - k$, we have the following consequence of Theorem 22.

Corollary 23. Maximum Induced Tree *on H-graphs is* W[1]*-hard parameterized by $k + ||H||$, where k denotes the solution size, and* Feedback Vertex Set *on H-graphs is* W[1]*-hard parameterized by $||H||$, and in both cases the hardness holds even if an H-representation of the input graph is given.*

By [16, Thm. 2] we know that the linear mim-width of an H-graph is at most $2 \cdot ||H||$ and a linear branch decomposition achieving this bound can be computed in polynomial time from a given H-representation of the graph in question. Theorem 22 and Corollary 23 therefore imply the following.

Corollary 24. Maximum Induced Forest *and* Maximum Induced Tree *are* W[1]*-hard parameterized by* $k + w$*, and* Feedback Vertex Set *is* W[1]*-hard parameterized by* w*, where* k *denotes the solution size and* w *the linear mim-width of the input graph. In both cases, the hardness holds even if a linear branch decomposition of mim-width* w *is given.*

6 Conclusion

We have shown that (Weighted) Feedback Vertex Set admits an $n^{\mathcal{O}(w)}$-time algorithm when given a branch decomposition of mim-width w. This provides a unified polynomial-time algorithm for Feedback Vertex Set on known classes of bounded mim-width, and gives the first polynomial-time algorithms for Circular Permutation and Circular k-Trapezoid graphs for fixed k.

We note that some of the above mentioned graph classes of bounded mim-width also have bounded asteroidal number, and a polynomial-time algorithm for Feedback Vertex Set on graphs of bounded asteroidal number was previously known due to Kratsch et al. [27]. However, our algorithm improves this result. For instance, k-Polygon graphs have mim-width at most $2k$ [1] and asteroidal number k [33]. The algorithm of Kratsch et al. [27] implies that Feedback Vertex Set on k-Polygon graphs can be solved in time $n^{\mathcal{O}(k^2)}$ while our result improves this bound to $n^{\mathcal{O}(k)}$ time. It is not difficult to see that in general, mim-width and asteroidal number are incomparable.

We conclude with mentioning an open problem regarding a generalization of the Feedback Vertex Set problem, the Subset Feedback Vertex Set problem which was introduced by Even et al. [12]. Here, we are given a graph G, a subset S of its vertices and an integer k and the question is whether there is a set of at most k vertices that intersects all cycles containing a vertex from S. It would be interesting to see whether Subset Feedback Vertex Set is XP-time solvable parameterized by mim-width, possibly by extending the approach given in this paper.

Open Question. Is there an XP-time algorithm that solves Subset Feedback Vertex Set parameterized by the mim-width of a given branch decomposition of the input graph?

This question was also posed recently by Papadopoulos and Tzimas who gave an XP-time algorithm for Subset Feedback Vertex Set parameterized by the size of an independent set in the input graph [29]. Moreover, they also showed in earlier work that Subset Feedback Vertex Set is polynomial-time solvable on Permutation and Interval graphs [28], both classes of linear mim-width 1.

References

[1] Belmonte, R., Vatshelle, M.: Graph classes with structured neighborhoods and algorithmic applications. Theor. Comput. Sci. **511**, 54–65 (2013).

[2] Bodlaender, H.L.: On disjoint cycles. Int. J. Found. Comput. Sci. **5**(1), 59–68 (1994)

[3] Bodlaender, H.L., Cygan, M., Kratsch, S., Nederlof, J.: Deterministic single exponential time algorithms for connectivity problems parameterized by treewidth. Information and Computation **243**, 86–111 (2015).

[4] Brandstädt, A., Le, V.B., Spinrad, J.P.: Graph classes: a survey. SIAM (1999)

[5] Bui-Xuan, B.M., Suchỳ, O., Telle, J.A., Vatshelle, M.: Feedback vertex set on graphs of low clique-width. Eur. J. Combin. **34**(3), 666–679 (2013)

[6] 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)

[7] Cygan, M., Fomin, F.V., Kowalik, Ł., Lokshtanov, D., Marx, D., Pilipczuk, M., Pilipczuk, M., Saurabh, S.: Parameterized Algorithms. Springer (2015).

[8] Dehne, F.K.H.A., Fellows, M.R., Langston, M.A., Rosamond, F.A., Stevens, K.: An $\mathcal{O}(2^{O(k)}n^3)$ fpt algorithm for the undirected feedback vertex set problem. In: Proc. 11th COCOON, *LNCS*, vol. 3595, pp. 859–869. Springer (2005)

[9] Downey, R.G., Fellows, M.R.: Fixed-parameter tractability and completeness I: Basic results. SIAM J. Comput. **24**(4), 873–921 (1995)

[10] Downey, R.G., Fellows, M.R.: Parameterized Complexity. Springer (1999)

[11] Downey, R.G., Fellows, M.R.: Fundamentals of Parameterized Complexity. Texts in Computer Science. Springer (2013)

[12] Even, G., Naor, J., Zosin, L.: An 8-approximation algorithm for the subset feedback vertex set problem. SIAM Journal on Computing **30**(4), 1231–1252 (2000)

[13] Fellows, M.R., Hermelin, D., Rosamond, F.A., Vialette, S.: On the parameterized complexity of multiple-interval graph problems. Theoretical Computer Science **410**(1), 53–61 (2009)

[14] Festa, P., Pardalos, P.M., Resende, M.G.C.: Feedback set problems. In: Handbook of combinatorial optimization, pp. 209–258. Springer (1999)

[15] Flotow, C.: Potenzen von Graphen. Ph.D. thesis, Universität Hamburg (1995)

[16] Fomin, F.V., Golovach, P.A., Raymond, J.F.: On the tractability of optimization problems on H-graphs. In: Proc. 26th ESA, *LIPIcs*, vol. 112, pp. 30:1–30:14. Schloss Dagstuhl (2018). ArXiv:1709.09737

[17] Golumbic, M.C., Rotics, U.: On the clique-width of some perfect graph classes. Int. J. Found. Comput. Sci. **11**(03), 423–443 (2000)

[18] Guo, J., Gramm, J., Hüffner, F., Niedermeier, R., Wernicke, S.: Compression-based fixed-parameter algorithms for feedback vertex set and edge bipartization. J. Comput. Syst. Sci. **72**(8), 1386–1396 (2006)

[19] Jaffke, L., Kwon, O., Strømme, T.J.F., Telle, J.A.: Generalized distance domination problems and their complexity on graphs of bounded mim-width. In: Proc. 13th IPEC, *LIPIcs*, vol. 115, pp. 6:1–6:14 (2018)

[20] Jaffke, L., Kwon, O., Telle, J.A.: A note on the complexity of feedback vertex set parameterized by mim-width. arxiv preprint (2017). ArXiv:1711.05157

[21] Jaffke, L., Kwon, O., Telle, J.A.: Polynomial-time algorithms for the longest induced path and induced disjoint paths problems on graphs of bounded mim-width. In: Proc. 12th IPEC, *LIPIcs*, vol. 89, pp. 21:1–21:13. Schloss Dagstuhl (2017)

[22] Jaffke, L., Kwon, O., Telle, J.A.: Mim-width I. Induced path problems (2019). To appear in *Discrete Applied Mathematics*

[23] Jaffke, L., Kwon, O., Telle, J.A.: A unified polynomial-time algorithm for feedback vertex set on graphs of bounded mim-width. In: Proc. 35th STACS, *LIPIcs*, vol. 96, pp. 42:1–42:14. Schloss Dagstuhl (2018)

[24] Jansen, B.M.P., Raman, V., Vatshelle, M.: Parameter ecology for feedback vertex set. Tsinghua Sci. Technol. **19**(4), 387–409 (2014)

[25] Kanj, I., Pelsmajer, M., Schaefer, M.: Parameterized algorithms for feedback vertex set. In: Proc. 1st IWPEC, *LNCS*, vol. 3162, pp. 235–247. Springer (2004)

[26] Karp, R.M.: Reducibility among combinatorial problems. In: Complexity of Computer Computations, pp. 85–103. Springer (1972)

[27] Kratsch, D., Müller, H., Todinca, I.: Feedback vertex set on AT-free graphs. Discrete Applied Mathematics **156**(10), 1936–1947 (2008)

[28] Papadopoulos, C., Tzimas, S.: Polynomial-time algorithms for the subset feedback vertex set problem on interval graphs and permutation graphs. In: Proc. 21st FCT, *LNCS*, vol. 10472, pp. 381–394. Springer (2017)

[29] Papadopoulos, C., Tzimas, S.: Subset feedback vertex set on graphs of bounded independent set size. In: Proc. 13th IPEC, *LIPIcs*, vol. 115, pp. 20:1–20:14 (2018)

[30] Pietrzak, K.: On the parameterized complexity of the fixed alphabet shortest common supersequence and longest common subsequence problems. Journal of Computer and System Sciences **67**(4), 757–771 (2003)

[31] Raman, V., Saurabh, S., Subramanian, C.R.: Faster fixed parameter tractable algorithms for undirected feedback vertex set. In: Proc. 13th ISAAC, *LNCS*, vol. 2518, pp. 241–248. Springer (2002)

[32] Raman, V., Saurabh, S., Subramanian, C.R.: Faster fixed parameter tractable algorithms for finding feedback vertex sets. ACM Trans. Algorithms **2**(3), 403–415 (2006)

[33] Stewart, L., Valenzano, R.: On polygon numbers of circle graphs and distance hereditary graphs. Discrete Applied Mathematics **248**, 3–17 (2018)

[34] Vatshelle, M.: New width parameters of graphs. Ph.D. thesis, Univ. Bergen (2012)