A Unified Polynomial-Time Algorithm for Feedback Vertex Set on Graphs of Bounded Mim-Width

Lars Jaffke
Department of Informatics, University of Bergen, Norway
lars.jaffke@uib.no

O-joung Kwon
Logic and Semantics, Technische Universität Berlin, Berlin, Germany
ojoungkwon@gmail.com

Jan Arne Telle
Department of Informatics, University of Bergen, Norway
jan.arne.telle@uib.no

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 algorithm for many well-known classes, such as Interval graphs and Permutation graphs, 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 for fixed k. In all these classes the decomposition is computable in polynomial time, as shown by Belmonte and Vatshelle [Theor. Comput. Sci. 2013].

We show that powers of graphs of tree-width $w-1$ or path-width w and powers of graphs of clique-width w have mim-width at most w. These results extensively provide new classes of bounded mim-width. We prove a slight strengthening of the first statement which implies that, surprisingly, Leaf Power graphs which are of importance in the field of phylogenetic studies have mim-width at most 1. Given a tree decomposition of width $w-1$, a path decomposition of width w, or a clique-width w-expression of a graph G, one can for any value of k find a mim-width decomposition of its k-power in polynomial time, and apply our algorithm to solve Feedback Vertex Set on the k-power in time $n^{\mathcal{O}(w)}$.

In contrast to Feedback Vertex Set, we show that Hamiltonian Cycle is NP-complete even on graphs of linear mim-width 1, which further hints at the expressive power of the mim-width parameter.

2012 ACM Subject Classification Mathematics of computing → Graph algorithms, Theory of computation → Parameterized complexity and exact algorithms

Keywords and phrases Graph Width Parameters, Graph Classes, Feedback Vertex Set, Leaf Powers

Digital Object Identifier 10.4230/LIPIcs.STACS.2018.42

Related Version A full version of the paper is available at `https://arxiv.org/abs/1710.07148v2`, [23]

35th Symposium on Theoretical Aspects of Computer Science (STACS 2018).
Editors: Rolf Niedermeier and Brigitte Vallée; Article No. 42; pp. 42:1–42:14
Leibniz International Proceedings in Informatics
LIPICS Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany
SYMPOSIUM ON THEORETICAL ASPECTS OF COMPUTER SCIENCE

1 Introduction

A feedback vertex set in a graph is a subset of its vertices 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 [11] and throughout the years numerous efforts have been made to obtain faster algorithms for this problem [4, 8, 9, 10, 11, 12, 18, 25, 32, 33]. In terms of parameterizations by structural properties of the graph, Feedback Vertex Set is e.g. known to be FPT parameterized by tree-width [9] and clique-width [6], and W[1]-hard but in XP parameterized by 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 [36] which measures how easy it is to decompose a graph along vertex cuts 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, 36], 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.[1] Since such a decomposition can be computed in polynomial time [1, 36] 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, k-Polygon, Dilworth-k and Co-k-Degenerate graphs for fixed k. Furthermore, our algorithm can be applied to Weighted Feedback Vertex Set as well, which for several of these classes was not known to be solvable in polynomial time.

► **Theorem 1.** *Given an n-vertex graph and a branch decomposition of mim-width w, we can solve* (Weighted) Feedback Vertex Set *in time $n^{\mathcal{O}(w)}$.*

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 [35]. 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 the 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 give results that expand our knowledge of the expressive power of mim-width. The k-power of a graph is the graph obtained by adding an edge vw for two vertices v, w with distance at most k. We show that powers of graphs of tree-width $w-1$ or path-width w and powers of graphs of clique-width w have mim-width at most w.

[1] This problem was mentioned as an 'interesting topic for further research' in [24]. Furthermore, the authors recently proved it to be W[1]-hard [21].

▶ **Theorem 2.** *Given a nice tree decomposition of width w, all of whose join bags have size at most w, or a clique-width w-expression of a graph, one can output a branch decomposition of mim-width w of its k-power in polynomial time.*

Theorem 2 implies that LEAF POWER graphs, of importance in the field of phylogenetic studies, have mim-width 1. These graphs are known to be STRONGLY CHORDAL and there has recently been interest in delineating the difference between these two graph classes, on the assumption that this difference was not very big [28, 30]. Our result actually implies a large difference, as it was recently shown by Mengel that there are STRONGLY CHORDAL SPLIT graphs of mim-width linear in the number of vertices [29].

We contrast our positive result with a proof that HAMILTONIAN CYCLE is NP-complete on graphs of linear mim-width 1, even when given a decomposition. Panda and Pradhan [31] showed that HAMILTONIAN CYCLE is NP-complete on ROOTED DIRECTED PATH graphs and we show that the graphs constructed in their reduction have linear mim-width 1. This provides evidence that the class of graphs of linear mim-width 1 is larger than one might have previously expected. Up until now, on all graph classes of linear mim-width 1, HAMILTONIAN CYCLE was known to be polynomial time (PERMUTATION), or even linear time (INTERVAL) solvable. This can be compared with the fact that parameterized by clique-width, FEEDBACK VERTEX SET is FPT [6] and HAMILTONIAN CYCLE only admits an XP algorithm [3, 13] but is W[1]-hard [15] (see also [16]).

Let us explain some of the essential ingredients of our dynamic programming algorithm. 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 7). Motivated by this observation, given a forest, we define the forest obtained by removing its isolated vertices and leaves to be its *reduced forest*. The observation implies that in a cut (A, B) of a graph G, there are at most n^{6w} possible reduced forests of some forests consisting of edges crossing this cut. We enumerate all of them, and use these as indices of the table of our algorithm.

However, the interaction of an induced forest F in G with the edges of the bipartite graph crossing the cut (A, B), denote this graph by $G_{A,B}$, is not completely described by its reduced forest R. Observe that there might still be edges in the graph $G_{A,B}$ after removing the vertices of R; however, these edges are not contained in the forest F. We capture this property of F by considering a minimal vertex cover of $G_{A,B} - V(R)$ that avoids all vertices in F. Hence, as a second component of the table indices, we enumerate all minimal vertex covers of $G_{A,B} - V(R)$, for any possible reduced forest R.

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 n^w minimal vertex covers. Remark that in the companion paper [22], we use minimal vertex covers of a bipartite graph in a similar way. The usage here is more complicated because we cannot index the table by the full intersection forest, but only index by its reduced forest.

Throughout the paper, proofs of statements marked with '★' are deferred to the full version [23].

2 Preliminaries

For a graph G we denote by $V(G)$ and $E(G)$ its vertex and edge set, respectively. For a vertex set $X \subseteq V(G)$, we denote by $G[X]$ the subgraph *induced* by X. We use the shorthand $G - X$ for $G[V(G) \setminus X]$. The union and intersection of two graphs G_1 and G_2 are denoted by $G_1 \cup G_2$ and $G_1 \cap G_2$, respectively. For a vertex $v \in V(G)$, we denote by $N_G(v)$ the set of *neighbors* of v in G. For $A \subseteq V(G)$, let $N_G(A)$ be the set of vertices in $V(G) \setminus A$ having a neighbor in A. When G is clear from the context, we allow to remove it from the subscript. We denote by $\mathcal{C}(G)$ the set of connected components of G.

For 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 a graph G and a vertex subset A of G, we define $\text{mim}_G(A)$ to be the maximum size of an induced matching in $G[A, V(G) \setminus A]$. A pair $(T, \mathcal{L})$ of a subcubic tree T 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* $(T, \mathcal{L})$ is defined as $\text{mimw}(T, \mathcal{L}) := \max_{e \in E(T)} \text{mim}_G(A_1^e)$. 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 path with a pendant leaf at each node.

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. We use the shorthand V_t for $V(G_t)$ and let $\bar{V_t} := V(G) \setminus V_t$.

▶ **Definition 3** (Boundary)**.** Let G be a graph and $A, B \subseteq V(G)$ such that $A \cap B = \emptyset$. We let $\text{bd}_B(A)$ be the set of vertices in A that have a neighbor in B, i.e. $\text{bd}_B(A) := \{v \in V(A) \mid N(v) \cap B \neq \emptyset\}$. We define $\text{bd}(A) := \text{bd}_{V(G)\setminus A}(A)$ and call $\text{bd}(A)$ the *boundary* of A in G.

▶ **Definition 4** (Crossing Graph)**.** Let G be a graph and $A, B \subseteq V(G)$. If $A \cap B = \emptyset$, we define the graph $G_{A,B} := G[\text{bd}_B(A), \text{bd}_A(B)]$ to be the *crossing graph from* A *to* B.

For a node t in a branch decomposition, we define $G_{t,\bar{t}} := G_{V_t, \bar{V_t}}$.

We prove that given a set $A \subseteq V(G)$, the number of minimal vertex covers in $G_{A,V(G)\setminus A}$ is bounded by $n^{\text{mim}_G(A)}$, and furthermore, the set of all minimal vertex covers can be enumerated in time $n^{\mathcal{O}(\text{mim}_G(A))}$. This observation is crucial to argue that we only need to store $n^{\mathcal{O}(w)}$ entries at each node in the branch decomposition in our algorithm.

▶ **Corollary 5** (Minimal Vertex Covers Lemma, ★)**.** *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 $n^{\text{mim}_H(A)}$, and the set of all minimal vertex covers of H can be enumerated in time $n^{\mathcal{O}(\text{mim}_H(A))}$.*

3 Reduced forests

We formally introduce the notion of a *reduced forest* which will be crucial to obtain the desired runtime bound of the algorithm for Feedback Vertex Set.

▶ **Definition 6** (Reduced Forest)**.** Let F be a forest. A *reduced forest* of F, denoted by $\mathfrak{R}(F)$, is an induced subforest of F obtained as follows. (i) Remove all isolated vertices of F. (ii) For each component C of F with $|V(C)| = 2$, remove one of its vertices. (iii) For each component C of F with $|V(C)| \geq 3$, remove all leaves of C.

Note that if F has no component that is a single edge then the reduced forest is uniquely defined. We give an upper bound on the size of a reduced forest $\mathfrak{R}(F)$ by a function of the size of a maximum induced matching in the forest F. This is crucial in our algorithm.

▶ **Lemma 7.** *Let p be a positive integer. If F is a forest whose maximum induced matching has size at most p and F' is a reduced forest of F, then $|V(F')| \leq 6p$.*

Proof. We sketch the proof, and provide the details in the full version [23]. For a forest F, we denote by $m(F)$ the size of the maximum induced matching in F. We prove by induction on $m(F)$. We may assume F contains no isolated vertices. If $m(F) \leq 1$, then F consists of one component containing no path of length 4, and $\mathfrak{R}(F)$ contains at most 2 nodes. We may assume $m(F) = p > 1$.

If F contains a connected component C containing no path of length 4, then it contains at most 2 internal nodes, and $m(F - V(C)) = m(F) - 1$. We may assume every component C of F contains a path of length 4. Assume 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 $\mathfrak{R}(F)$. In this case, we define F' as the forest obtained from F by removing v_2, v_3, v_4 and adding an edge v_1v_5. Then we have $m(F') \leq m(F) - 1$. By induction hypothesis, $\mathfrak{R}(F')$ contains at most $6(p-1)$ nodes, and thus $\mathfrak{R}(F)$ contains at most $6(p-1)+3 \leq 6p$ nodes. We may assume there is no such a path.

Let C be a component of F. As $\mathfrak{R}(C)$ contains at least 3 nodes, the leaves of $\mathfrak{R}(C)$ form an independent set. Suppose $\mathfrak{R}(C)$ contains t leaves. Since each leaf of $\mathfrak{R}(C)$ is incident with a leaf of C, $\mathfrak{R}(C)$ 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 nodes of degree at least 3. By the previous argument, there are at most 2 nodes between two nodes of degree other than 2 in $\mathfrak{R}(C)$. Thus, $\mathfrak{R}(C)$ contains at most $t + t + 2(2t-1) \leq 6t$ nodes. The result follows by induction hypothesis. ◀

Let (A, B) be a vertex partition of a graph G, and R be some forest in $G_{A,B}$. In the algorithm, we will be asking if there exists an induced forest F in $G[A \cup \mathrm{bd}(B)]$ such that $F \cap G_{A,B}$ has R as a reduced forest. However, this formulation turns out to be technical, as we need to significantly consider some edges in B when we merge two partial solutions. To ease this task, we define the following notion on an induced forest in $G[A \cup \mathrm{bd}(B)] - E(G[\mathrm{bd}(B)])$.

▶ **Definition 8** (Forest respecting a 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: (i) R is a reduced forest of $F \cap G_{A,B}$ and (ii) $V(F) \cap M = \emptyset$.

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, if $F \cap G_{A,B}$ contains a vertex v in $G_{A,B} - V(R)$ having at least two neighbors in R, then v should be contained in the reduced forest. Therefore, in $F \cap G_{A,B}$, every vertex in $V(F \cap G_{A,B}) \setminus V(R)$ should have at most one neighbor in R. Second, every leaf x of R should have a neighbor y in $G_{A,B} - V(R)$ such that 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 $V(R)$.

▶ **Definition 9** (Potential Leaves)**.** 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 $G_{A,B} - V(R)$. Let $H := G_{A,B}$. For each vertex $x \in V(R)$, we define its set of *potential leaves* as $PL_{R,M}(x) := N_H(x) \setminus N_H(V(R) \setminus \{x\}) \setminus (M \cup V(R))$.

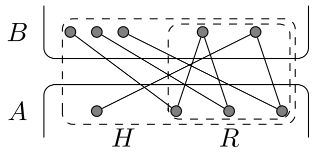

Figure 1 The graph R is a reduced forest of H.

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, this will be necessary when considering cuts corresponding to some node and its child.

▶ **Definition 10** (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 for every $v \in V(R) \cap B$ having at least two neighbors in $V(R) \cap A_1$, $v \in V(R_1)$.
2. $(V(R_1) \setminus V(R)) \cap B = \emptyset$ and $V(R_1) \cap M = \emptyset$.
3. Every vertex in $(V(R_1) \setminus V(R)) \cap A_1$ has at most one neighbor in $V(R) \cap B$.
4. $V(R) \cap M_1 = \emptyset$ and $M \cap A_1 \subseteq M_1$.
5. Let v be a vertex in $M \cap B$ incident with an edge vw in $G_{A_1,B} - V(R)$ for some $w \notin V(R_1)$ that is not covered by any vertices in $M \setminus \{v\}$. Then either $v \in M_1$ or $w \in M_1$.

Lastly, we define a notion for merging two partial solutions.

▶ **Definition 11** (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 $\mathcal{C}(R) \cup \mathcal{C}(R_1) \cup \mathcal{C}(R_2)$ such that

- for H_1 and H_2 contained in distinct sets of $\mathcal{C}(R), \mathcal{C}(R_1), \mathcal{C}(R_2)$, H_1 is adjacent to H_2 in Q iff $V(H_1) \cap V(H_2) \neq \emptyset$,
- for $H_1, H_2 \in \mathcal{C}(R_i)$, H_1 is adjacent to H_2 iff they are contained in the same part 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 iff they are contained in the same component of Q.

▶ **Proposition 12.** *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). 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*

- *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),*
- *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. 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 $F_i := F_i^* \cap G_{A_i,A_{3-i}\cup B}$, and R_i be a reduced forest of F_i such that (Single-edge Rule) 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.

We check that every vertex in $S := (V(R) \setminus (V(R_1) \cup V(R_2))) \cap B$ has at least two neighbors in $(V(R_1) \cap A_1) \cup (V(R_2) \cap A_2)$. Suppose there exists a vertex v in S violating the condition. As $v \in V(R)$, 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 $w \in A_1$. If v has a neighbor other than w in $V(H) \cap A_1$, then v is contained in R_1. So, in H, 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 v in F_1. Then vw is a single-edge component of F_1, and by Single-edge Rule, we selected v as a vertex of R_1. This contradicts $v \notin V(R_1)$. We conclude that every vertex in S has at least two neighbors in $(V(R_1) \cap A_1) \cup (V(R_2) \cap A_2)$.

Conditions 1, 2 and 3 of being a restriction follows from the definition of a restriction and the Single-edge Rule. The details are given in the full version [23].

We now construct a minimal vertex cover M_1 of $G_{A_1, A_2 \cup B} - V(R_1)$, and verify the fourth and fifth conditions of being a restriction. Let M' be the set of all vertices v in M incident with an edge vw in $G_{A_1,A_2} - V(R)$ where vw is not covered by $M \setminus \{v\}$ and $w \notin V(R_1)$.

▶ **Claim 13.** *There is a minimal vertex cover M_1 of $G_{A_1,A_2 \cup B} - V(R_1)$ satisfying the following.*

- *$V(R) \cap M_1 = \emptyset$ and $M \cap A_1 \subseteq M_1$.*
- *Let v be a vertex in $M \cap B$ incident with an edge vw in $G_{A_1,B} - V(R)$ for some $w \notin V(R_1)$ that is not covered by any vertices in $M \setminus \{v\}$. Then either $v \in M_1$ or $w \in M_1$.*

Proof. Let Y be the set of all vertices in $\mathrm{bd}(A_2) \setminus V(H)$ having a neighbor in $\mathrm{bd}(A_1) \setminus V(R_1)$. Let Z be the set of all vertices in $\mathrm{bd}(A_1) \setminus V(R_1) \setminus (M \cap A_1)$ having a neighbor in $(V(R) \setminus V(R_1)) \cap B$. Let M'' be the set obtained from $M' \cup Y \cup Z$ by removing all vertices $v \in M' \cap B$ such that all the neighbors of v in $\mathrm{bd}(A_1) \setminus V(R_1) \setminus (M \cap A_1)$ are contained in Z.

By construction we can show that M'' is a vertex cover of $G_{A_1,A_2 \cup B} - V(R_1)$. We take a minimal vertex cover M_1 of $G_{A_1,A_2 \cup B} - V(R_1)$ contained in M''. We have $V(R) \cap M_1 = \emptyset$. Since each vertex of $M' \cap A$ covers some edge that is not covered by any other vertex in M'', we have $M \cap A_1 = M' \cap A_1 \subseteq M_1$. Since every vertex in Z meets some edge incident with $V(R) \setminus V(R_1)$, Z is contained in M_1. If v is a vertex in $M \cap B$ incident with an edge vw in $G_{A_1,B} - V(R)$ for some $w \notin V(R_1)$ that is not covered by any vertices in $M \setminus \{v\}$, then $v \in M' \cap B$. By construction of M'', either $v \in M'' \cap B$ or $w \in Z$. In particular if $w \notin Z$, then v is the vertex covering the edge vw, and it also remains in M_1. Thus, the fifth condition for being a restriction also holds, as required. ⌟

By Claim 13 we know that Conditions 4 and 5 of being a restriction hold, so we conclude that there is a restriction (R_1, M_1) of (R, M) where F_1^* respects (R_1, M_1). ◀

▶ **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) and for each $i \in \{1, 2\}$,*

- *let (R_i, M_i) be a restriction of (R, M) that $H_i := H \cap G[A_i \cup \mathrm{bd}(A_{3-i} \cup B)] - E(G[\mathrm{bd}(A_{3-i} \cup B)])$ respects (guaranteed by Proposition 12), 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 iff they are contained in the same connected component of H_i.*

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

Now, we prove a proposition regarding the merging operation of two partial solutions. Unfortunately, when we have partial solutions H_1 and H_2 for A_1 and A_2, respectively, $G[V(H_1) \cup V(H_2) \cup V(R)]$ may not be a partial solution. One reason is that since $M_1 \cap B$

may differ from $M \cap B$, H_1 might contain some vertex in $(M \setminus M_1) \cap B$. To avoid a situation where such a vertex is in R_1, we require that $V(R_1) \cap M = \emptyset$ (this is already included in the condition of being a restriction). Thus, such a vertex will be a potential leaf of some vertex in R_1, and we could simply remove it to find a forest avoiding M. The second reason is that for some vertex of $V(R) \cap V(A_1)$, it might have a potential leaf in A_2, but not in B, and thus in $G[V(H_1) \cup V(H_2) \cup V(R)]$ this vertex may not have a potential leaf as a neighbor even if it has degree at most 1 in R. In this case, we can simply add one of the potential leaves.

▶ **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 $G_{A_1 \cup A_2, B} - V(R)$ 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 and M_i are restrictions of R and M,*
- *every vertex in $(V(R)\ V(R_1) \cup V(R_2)) \cap B$ 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.*

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. As (R, R_1, R_2, P_1, P_2) is compatible, we can verify that $H^* := G[V(H_1) \cup V(H_2) \cup V(R)]$ is a forest. Let H be the graph obtained from $H^* - (B \setminus V(R))$ by adding a potential leaf of each vertex in $V(R) \cap (A_1 \cup A_2)$ of degree at most 1 in R and removing all edges between vertices in B. We observe that H is a forest. Since H^* is a forest, $H^* - (B \setminus V(R))$ is 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 edges in $G[B]$. In the remainder, we prove that H respects (R, M); that is, (i) R is a reduced forest of $G_{A_1 \cup A_2, B} \cap H$, and (ii) $V(H) \cap M = \emptyset$.

Condition (ii) is easy to verify: since we remove all vertices in M when we construct H from H^*, we have $V(H) \cap M = \emptyset$. We now verify condition (i). Let $H_{new} := H \cap G_{A_1 \cup A_2, B}$. We first verify that every vertex of $V(H_{new}) \setminus V(R)$ has degree at most 1 in H_{new}.

▶ **Claim 16 (★).** *Every vertex of $V(H_{new}) \setminus V(R)$ has degree at most 1 in H_{new}.*

We argue that we can take R as a reduced forest of H_{new}. Let $v \in V(R)$. If v has degree at least 2 in H_{new}, then v is contained in any reduced forest of H_{new}. Suppose v has degree at most 1 in H_{new}. Suppose $v \in A_1 \cup A_2$. In this case, by the construction, v is incident with its potential leaf in H_{new}, say w. It means that vw is a single-edge component in H_{new}, 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 a neighbor 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_{new}, a contradiction. Thus, v has no neighbor in R_i, and has exactly one potential leaf, say w. By Claim 16, v is the unique neighbor of w in R, and thus vw is a single-edge component of H_{new}. 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_{new}. It shows that R is a reduced forest of H_{new}.

Note that for each $i \in \{1, 2\}$, $V(H_i) \cap A_i$ avoids $M \cap A_i$. Furthermore, when we construct H_{new}, we removed all vertices in $M \cap B$. Therefore, we have $V(H_{new}) \cap M = \emptyset$. ◀

4 Feedback Vertex Set on graphs of bounded mim-width

We give an algorithm that solves the Feedback Vertex Set problem on graphs on n vertices together with a branch decomposition of mim-width w in time $n^{\mathcal{O}(w)}$. We observe that given a graph G, a subset of its vertices $S \subseteq V(G)$ is by definition a feedback vertex set if and only if $G - S$, the induced subgraph of G on vertices $V(G) \setminus S$, is an induced 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/Mim-Width
Input: A graph G on n vertices, a branch decomposition $(T, \mathcal{L})$ of G, an integer k.
Parameter: $w := \mathrm{mimw}(T, \mathcal{L})$.
Question: Does G contain an induced forest of size at least $n - k$?

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 F, interacts with the graph $G_{t+\mathrm{bd}} := G[V_t \cup \mathrm{bd}(\bar{V}_t)] - E(G[\mathrm{bd}(\bar{V}_t)])$. The intersection of F with $G_{t+\mathrm{bd}}$ is an induced forest which throughout the following we denote by $F_{t+\mathrm{bd}} := F[V(G_{t+\mathrm{bd}})]$. 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 F with the crossing graph $G_{t,\bar{t}}$, denoted by $F_{t,\bar{t}} := F[V(G_{t,\bar{t}})]$.

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* forests, introduced in Section 3.

At each node $t \in V(T)$, we only consider reduced forests as possible indices for the table entries. By Lemma 7, the number of reduced forests in each cut of mim-value w is bounded by n^{6w}. We analyze the structure of $F_{t,\bar{t}}$ to motivate the objects that can be used to represent $F_{t,\bar{t}}$ in such a way that the number of all possible table entries is at most $n^{\mathcal{O}(w)}$.

The induced forest $F_{t,\bar{t}}$ has three types of vertices in $G_{t,\bar{t}}$: (1) The vertices of the reduced forest $\mathfrak{R}(F_{t,\bar{t}})$ of $F_{t,\bar{t}}$. (2) The leaves of the induced forest $F_{t,\bar{t}}$, denoted by $L(F_{t,\bar{t}})$. (3) Vertices in $F_{t,\bar{t}}$ that do not have a neighbor in $F_{t,\bar{t}}$ on the opposite side of the boundary, in the following called *non-crossing* vertices and denoted by $\mathrm{NC}(F_{t,\bar{t}})$.

As outlined above, the only type of vertices in $F_{t,\bar{t}}$ that will be used as part of the table indices are the vertices of a reduced forest of $F_{t,\bar{t}}$. Hence, we neither know about the leaves of $F_{t,\bar{t}}$ nor its non-crossing vertices upon inspecting this part of the index. Suppose $v \in (L(F_{t,\bar{t}}) \cup \mathrm{NC}(F_{t,\bar{t}})) \cap V_t$. Then, $F_{t,\bar{t}}$ does not use any vertex in $x \in (N(v) \cap \bar{V}_t) \setminus V(\mathfrak{R}(F_{t,\bar{t}}))$: If v is a leaf in $F_{t,\bar{t}}$, then the presence of the edge vx would make it a non-leaf vertex and if v is a non-crossing vertex, the presence of vx would make v a vertex incident to an edge of the forest crossing the cut. An analogous point can be made for a vertex in $(L(F_{t,\bar{t}}) \cup \mathrm{NC}(F_{t,\bar{t}})) \cap \bar{V}_t$. We capture this property of $F_{t,\bar{t}}$ by considering a minimal vertex cover of $G_{t,\bar{t}} - V(\mathfrak{R}(F_{t,\bar{t}}))$ that avoids all leaves and non-crossing vertices of $F_{t,\bar{t}}$. Such a minimal vertex cover always exists as $L(F_{t,\bar{t}}) \cup \mathrm{NC}(F_{t,\bar{t}})$ is an independent set in $G_{t,\bar{t}}$.

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

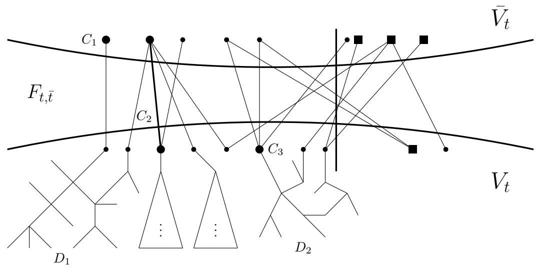

Figure 2 An example of a crossing graph $G_{t,\bar{t}}$ together with an induced forest F and their interaction. The forest $F_{t,\bar{t}} = F[V(G_{t,\bar{t}})]$ is displayed to the left of the dividing line in the drawing and the 4 vertices and 1 edge in bold form a reduced forest R of $F_{t,\bar{t}}$. 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 F.

We are 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 $G_{t+\mathrm{bd}}$. We denote by $\mathcal{R}_t$ the set of all induced forests of $G_{t,\bar{t}}$ on at most $6w$ vertices. For $R \in \mathcal{R}_t$, let $\mathcal{M}_{t,R}$ be the set of all minimal vertex covers of $G_{t,\bar{t}} - V(R)$ and $\mathsf{P}_{t,R}$ the set of all partitions of the components of R. For an illustration of the definition of the table indices, which we start on now, see Figure 2. For $(R, M, P) \in \mathcal{R}_t \times \mathcal{M}_{t,R} \times \mathsf{P}_{t,R}$ and $i \in \{0, \ldots, n\}$, we set $\mathcal{T}[t, (R, M, P), i] \coloneqq 1$ (and to 0 otherwise), iff the following conditions are satisfied.

1. There is an induced forest F in $G[V_t \cup \mathrm{bd}(\bar{V}_t)] - E(G[\mathrm{bd}(\bar{V}_t)])$ with $|V(F) \cap V_t| = i$.
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, $R = \mathfrak{R}(F_{t,\bar{t}})$, meaning that 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 iff they are contained in the same connected component of F.

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 other words, G contains an induced forest of size i if and only if $\mathcal{T}[r, (\emptyset, \emptyset, \emptyset), i] = 1$.

By definition, $|\mathcal{R}_t| = \mathcal{O}(n^{6w})$ and by Minimal Vertex Covers Lemma, $|\mathcal{M}_{t,R}| = n^{\mathcal{O}(w)}$ for each $R \in \mathcal{R}_t$. It is well known that $|\mathsf{P}_{t,R}| \leq (w/\log(w))^{\mathcal{O}(w)}$ by upper bounds on the Bell number (see e.g. [2]). Thus, we have

▶ **Proposition 17.** *There are at most $n^{\mathcal{O}(w)}$ table entries in $\mathcal{T}$.*

We now show how to compute the table entries in $\mathcal{T}$. We can easily fill in the table entries for the leaves of T, for the details see the full version [23]. Here, we focus on how to compute the entries in the internal nodes of T from the entries stored in the tables corresponding to their children. Let $t \in V(T)$ be an internal node with children a and b. Using Propositions 12, 14, 15, we can show the following.

▶ **Proposition 18 (★).** *Let $\mathfrak{I} = [(R, M, P), i] \in (\mathcal{R}_t \times \mathcal{M}_{t,R_t} \times \mathsf{P}_{t,R_t}) \times \{0, \ldots, n\}$ 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 $P = \mathcal{U}(R, R_1, R_2, P_1, P_2)$,*

- *every vertex in* $(V(R) \setminus V(R_1) \cup V(R_2)) \cap B$ *has at least two neighbors in* $(V(R_1) \cap A_1) \cup (V(R_2) \cap A_2)$,
- $i_a + i_b = i$.

Based on Proposition 18, we can proceed with the computation of the table at an internal node t with children a and b. Let $\mathfrak{I} = [(R, M, P), i] \in (\mathcal{R}_t \times \mathcal{M}_{t,R_t} \times \mathsf{P}_{t,R_t}) \times \{0, \ldots, n\}$.

(Step 1) 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) We consider all pairs $\mathfrak{I}_a = [(R_a, M_a, P_a), i_a] \in (\mathcal{R}_a \times \mathcal{M}_{a,R_a} \times \mathsf{P}_{a,R_a}) \times \{0, \ldots, n\}$ and $\mathfrak{I}_b = [(R_b, M_b, P_b), i_b] \in (\mathcal{R}_b \times \mathcal{M}_{b,R_b} \times \mathsf{P}_{b,R_b}) \times \{0, \ldots, n\}$. We check

- (R_a, M_a) and (R_b, M_b) are restrictions of (R, M) to $G_{a,\bar{a}}$ and $G_{b,\bar{b}}$ 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 $P = \mathcal{U}(R, R_1, R_2, P_1, P_2)$,
- every vertex in $(V(R) \setminus V(R_1) \cup V(R_2)) \cap B$ has at least two neighbors in $(V(R_1) \cap A_1) \cup (V(R_2) \cap A_2)$,
- $i_a + i_b = i$.

If there are $\mathfrak{I}_a$ and $\mathfrak{I}_b$ satisfying all of 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 18 and the runtime analysis is deferred to the full version [23].

In Weighted Feedback Vertex Set, we are given a graph and a 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, for a table $[t, (R, M, P)]$ we keep the $\omega(V(F) \cap V_t)$ value for an induced forest F respecting (R, M) and P with maximum $\omega(V(F) \cap V_t)$, as $\mathcal{T}[t, (R, M, P)]$. The procedure for a leaf node 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 in time $n^{\mathcal{O}(w)}$ as well. We have proved Theorem 1.

5 Hamiltonian Cycle for linear mim-width 1

▶ **Theorem 19.** Hamiltonian Cycle *is NP-complete on graphs of linear mim-width* 1*, even if given the mim-width decomposition.*

Proof. Itai et al [20] showed that given a bipartite graph G with maximum degree 3, it is NP-complete to decide if it has a Hamiltonian cycle, while Panda and Pradhan [31] construct, from this graph G, a rooted directed path graph H such that H has a Hamiltonian cycle if and only if G does. The construction of [31] can be used to also output a linear mim-width 1 decomposition of H, in polynomial time. We provide the details in the full version [23]. ◀

6 Powers of graphs

We show that k-powers of graphs of tree-width at most $w - 1$ have mim-width at most w. This is somewhat surprising because this bound does not depend on k. The following lemma captures the property. We denote by $\mathrm{dist}_G(v, w)$ the distance between v and w in G.

▶ **Lemma 20.** *Let* $k, w \in \mathbb{N}$ *and let* (A, B, C) *be a vertex partition of graph* G *such that there are no edges between* A *and* C*, and* B *has size* w*. If* H *is the* k*-power of* G*, then* $\mathrm{mim}_H(A \cup B) \leq w$.

Proof. Let $B := \{b_1, b_2, \dots, b_w\}$. For every vertex v in G, we assign a vector $c_v = (c_1^v, \dots, c_w^v)$ such that $c_i^v = \mathrm{dist}_G(v, b_i)$. Suppose for contradiction that there is an induced matching $\{y_1z_1, y_2z_2, \dots, y_tz_t\}$ of size at least $w+1$ in $H[A \cup B, C]$. Since $t \geq w+1$, there are distinct integers $t_1, t_2 \in \{1, 2, \dots, t\}$ and an integer $j \in \{1, 2, \dots, w\}$ such that

- $\mathrm{dist}_G(y_{t_1}, b_j) + \mathrm{dist}_G(z_{t_1}, b_j) \leq k$ and $\mathrm{dist}_G(y_{t_2}, b_j) + \mathrm{dist}_G(z_{t_2}, b_j) \leq k$.

Then we have either $\mathrm{dist}_G(y_{t_1}, b_j) + \mathrm{dist}_G(z_{t_2}, b_j) \leq k$ or $\mathrm{dist}_G(y_{t_2}, b_j) + \mathrm{dist}_G(z_{t_1}, b_j) \leq k$, which contradicts with the assumption that $y_{t_1}z_{t_2}$ and $y_{t_2}z_{t_1}$ are not edges in H. ◀

▶ **Theorem 21** (★)**.** *Let $k, w \in \mathbb{N}$ and G be a graph that admits a nice tree decomposition of width w all of whose join bags are of size at most w. Then the k-power of G has mim-width at most w. Furthermore, given such a nice tree decomposition, we can output a branch decomposition of mim-width at most w in polynomial time.*

The following notions are of importance in the field of phylogenetic studies, i.e. the reconstruction of ancestral relations in biology, see e.g. [7]. A graph G is a leaf power if there exists a threshold k and a tree T, called a leaf root, whose leaf set is $V(G)$ such that $uv \in E$ if and only if the distance between u and v in T is at most k. Similarly, G is called a min-leaf power if $uv \in E$ if and only if the distance between u and v in T is more than k. Thus, G is a leaf power if an only if its complement is a min-leaf power. It is easy to see that trees admit nice tree decompositions all of whose join bags have size 1 and since every leaf power graph is an induced subgraph of a power of some tree, it has mim-width at most 1 by Theorem 21.

▶ **Corollary 22.** *The leaf powers and min-leaf powers have mim-width at most 1 and given a leaf root, we can compute in polynomial time a branch decomposition witnessing this.*

We further show that powers of graphs of clique-width w have mim-width at most w. We give the details of the proof in the full version [23]; however we remark that the following lemma will imply this result. A graph is *w-labeled* if there is a labeling function $f : V(G) \to \{1, 2, \dots, w\}$.

▶ **Lemma 23** (★)**.** *Let $k, w \in \mathbb{N}$ and let (A, B) be a vertex partition of graph G such that $G[A]$ is w-labeled and two vertices in a label class of $G[A]$ have the same neighborhood in B. If H is the k-power of G, then $\mathrm{mim}_H(A) \leq w$.*

▶ **Theorem 24** (★)**.** *Let $k, w \in \mathbb{N}$ and G be a graph of clique-width w. Then the k-power of G has mim-width at most w. Furthermore, given a clique-width w-expression, we can output a branch decomposition of mim-width at most w in polynomial time.*

7 Conclusion

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

Somewhat surprisingly, we prove that powers of graphs of bounded tree-width or clique-width have bounded mim-width. Heggernes et al. [19] showed that the clique-width of the k-power of a path of length $k(k+1)$ is exactly k. This also shows that the expressive power of mim-width is much stronger than clique-width, since all powers of paths have mim-width just 1. As a special case, we show that Leaf Power graphs have mim-width 1. We believe the notion of mim-width can be of benefit to the study of Leaf Power graphs.

We conclude with repeating an open problem regarding algorithms for computing mim-width. The problem of computing the mim-width of general graphs was shown to be

W[1]-hard, not in APX unless NP = ZPP [34], and no algorithm for computing the mim-width of a graph in XP time is known. As in [34], we therefore ask: Is there an XP algorithm approximating mim-width w by some function $f(w)$ and returning a decomposition? We remark that it is a big open problem whether Leaf Power graphs can be recognized in polynomial time [5, 7, 28, 30]. A positive answer to our question may be used to design such a recognition algorithm using branch decompositions of bounded mim-width.

References

1. Rémy Belmonte and Martin Vatshelle. Graph classes with structured neighborhoods and algorithmic applications. *Theoretical Computer Science*, 511:54–65, 2013.
2. Daniel Berend and Tamir Tassa. Improved bounds on bell numbers and on moments of sums of random variables. *Probability and Mathematical Statistics*, 30(2):185–205, 2010.
3. Benjamin Bergougnoux, Mamadou Moustapha Kanté, and O-Joung Kwon. An optimal XP algorithm for hamiltonian cycle on graphs of bounded clique-width. In *Proceedings 15th International Symposium on Algorithms and Data Structures (WADS)*, pages 121–132, Cham, 2017. Springer.
4. Hans L. Bodlaender. On disjoint cycles. *International Journal of Foundations of Computer Science*, 5(1):59–68, 1994.
5. Andreas Brandstädt. On leaf powers. Technical report, University of Rostock, 2010.
6. Binh-Minh Bui-Xuan, Ondřej Suchỳ, Jan Arne Telle, and Martin Vatshelle. Feedback vertex set on graphs of low clique-width. *European Journal of Combinatorics*, 34(3):666–679, 2013.
7. Tiziana Calamoneri and Blerina Sinaimeri. Pairwise compatibility graphs: A survey. *SIAM Review*, 58(3):445–460, 2016. doi:10.1137/140978053.
8. Jianer Chen, Fedor V. Fomin, Yang Liu, Songjian Lu, and Yngve Villanger. Improved algorithms for feedback vertex set problems. *Journal of Computer and System Sciences*, 74(7):1188–1198, 2008.
9. Marek Cygan, Jesper Nederlof, Marcin Pilipczuk, Michał Pilipczuk, Joham M. M. van Rooij, and Jakub Onufry Wojtaszczyk. Solving connectivity problems parameterized by treewidth in single exponential time. In *Proceedings 52nd Annual Symposium on Foundations of Computer Science (FOCS)*, pages 150–159. IEEE, 2011.
10. Frank K. H. A. Dehne, Michael R. Fellows, Michael A. Langston, Frances A. Rosamond, and Kim Stevens. An $\mathcal{O}(2^{O(k)}n^3)$ FPT algorithm for the undirected feedback vertex set problem. In *Proceedings 11th International Computing and Combinatorics Conference (COCOON)*, pages 859–869. Springer, 2005.
11. Rodney G. Downey and Michael R. Fellows. Fixed-parameter tractability and completeness I: Basic results. *SIAM Journal on Computing*, 24(4):873–921, 1995.
12. Rodney G. Downey and Michael R. Fellows. *Parameterized Complexity*. Springer, 1999.
13. Wolfgang Espelage, Frank Gurski, and Egon Wanke. How to solve NP-hard graph problems on clique-width bounded graphs in polynomial time. In *Proceedings 27th International Workshop on Graph-Theoretic Concepts in Computer Science (WG)*, pages 117–128. Springer, 2001.
14. Paola Festa, Panos M. Pardalos, and Mauricio G. C. Resende. Feedback set problems. In *Handbook of Combinatorial Optimization*, pages 209–258. Springer, 1999.
15. Fedor V. Fomin, Petr A. Golovach, Daniel Lokshtanov, and Saket Saurabh. Intractability of clique-width parameterizations. *SIAM Journal on Computing*, 39(5):1941–1956, 2010.
16. Fedor V. Fomin, Petr A. Golovach, Daniel Lokshtanov, and Saket Saurabh. Almost optimal lower bounds for problems parameterized by clique-width. *SIAM Journal on Computing*, 43(5):1541–1563, 2014.

17 Martin Charles Golumbic and Udi Rotics. On the clique-width of some perfect graph classes. *International Journal of Foundations of Computer Science*, 11(03):423–443, 2000.

18 Jiong Guo, Jens Gramm, Falk Hüffner, Rolf Niedermeier, and Sebastian Wernicke. Compression-based fixed-parameter algorithms for feedback vertex set and edge bipartization. *Journal of Computer and System Sciences*, 72(8):1386–1396, 2006.

19 Pinar Heggernes, Daniel Meister, Charis Papadopoulos, and Udi Rotics. Clique-width of path powers. *Discrete Applied Mathematics*, 205:62–72, 2016.

20 Alon Itai, Christos H. Papadimitriou, and Jayme Luiz Szwarcfiter. Hamilton paths in grid graphs. *SIAM Journal on Computing*, 11(4):676–686, 1982.

21 Lars Jaffke, O-joung Kwon, and Jan Arne Telle. A note on the complexity of feedback vertex set parameterized by mim-width. *arXiv preprints*, 2017. arXiv:1711.05157.

22 Lars Jaffke, O-joung Kwon, and Jan Arne Telle. Polynomial-time algorithms for the longest induced path and induced disjoint paths problems on graphs of bounded mim-width. In *Proceedings 12th International Symposium on Parameterized and Exact Computation (IPEC)*, pages 21:1–21:12, 2017. arXiv:1708.04536.

23 Lars Jaffke, O-joung Kwon, and Jan Arne Telle. A unified polynomial-time algorithm for feedback vertex set on graphs of bounded mim-width. *arXiv preprints*, 2017. arXiv:1710.07148.

24 Bart M. P. Jansen, Venkatesh Raman, and Martin Vatshelle. Parameter ecology for feedback vertex set. *Tsinghua Science and Technology*, 19(4):387–409, 2014.

25 Iyad Kanj, Michael Pelsmajer, and Marcus Schaefer. Parameterized algorithms for feedback vertex set. In *Proceedings 1st International Workshop on Parameterized and Exact Computation (IWPEC)*, pages 235–247. Springer, 2004.

26 Richard M. Karp. Reducibility among combinatorial problems. In *Complexity of Computer Computations*, pages 85–103. Springer, 1972.

27 Dieter Kratsch, Haiko Müller, and Ioan Todinca. Feedback vertex set on at-free graphs. *Discrete Applied Mathematics*, 156(10):1936–1947, 2008.

28 Manuel Lafond. On strongly chordal graph that are not leaf powers. *arXiv preprints*, 2017. arXiv:1703.08018.

29 Stefan Mengel. Lower bounds on the mim-width of some perfect graph classes. *arXiv preprints*, 2016. arXiv:1608.01542, to appear in Discrete Applied Mathematics.

30 Ragnar Nevries and Christian Rosenke. Towards a characterization of leaf powers by clique arrangements. *Graphs and Combinatorics*, 32(5):2053–2077, 2016.

31 B. S. Panda and D. Pradhan. NP-completeness of hamiltonian cycle problem on rooted directed path graphs. *arXiv preprints*, 2008. arXiv:0809.2443.

32 Venkatesh Raman, Saket Saurabh, and C. R. Subramanian. Faster fixed parameter tractable algorithms for undirected feedback vertex set. In *Proceedings 13th International Symposium on Algorithms and Computation (ISAAC)*, pages 241–248. Springer, 2002.

33 Venkatesh Raman, Saket Saurabh, and C. R. Subramanian. Faster fixed parameter tractable algorithms for finding feedback vertex sets. *ACM Transactions on Algorithms*, 2(3):403–415, 2006.

34 Sigve H. Sæther and Martin Vatshelle. Hardness of computing width parameters based on branch decompositions over the vertex set. *Theoretical Computer Science*, 615:120–125, 2016.

35 Lorna Stewart and Richard Valenzano. On polygon numbers of circle graphs and distance hereditary graphs. *Discrete Applied Mathematics*, 2017. in press. `doi:10.1016/j.dam.2017.09.016`.

36 Martin Vatshelle. *New Width Parameters of Graphs*. PhD thesis, University of Bergen, Norway, 2012.