Finding Induced Subgraphs from Graphs with Small Mim-Width

Yota Otachi
Graduate School of Informatics, Nagoya University, Japan

Akira Suzuki
Graduate School of Information Sciences, Tohoku University, Sendai, Japan

Yuma Tamura
Graduate School of Information Sciences, Tohoku University, Sendai, Japan

Abstract

In the last decade, algorithmic frameworks based on a structural graph parameter called mim-width have been developed to solve generally NP-hard problems. However, it is known that the frameworks cannot be applied to the Clique problem, and the complexity status of many problems of finding dense induced subgraphs remains open when parameterized by mim-width. In this paper, we investigate the complexity of the problem of finding a maximum induced subgraph that satisfies prescribed properties from a given graph with small mim-width. We first give a meta-theorem implying that various induced subgraph problems are NP-hard for bounded mim-width graphs. Moreover, we show that some problems, including Clique and Induced Cluster Subgraph, remain NP-hard even for graphs with (linear) mim-width at most 2. In contrast to the intractability, we provide an algorithm that, given a graph and its branch decomposition with mim-width at most 1, solves Induced Cluster Subgraph in polynomial time. We emphasize that our algorithmic technique is applicable to other problems such as Induced Polar Subgraph and Induced Split Subgraph. Since a branch decomposition with mim-width at most 1 can be constructed in polynomial time for block graphs, interval graphs, permutation graphs, cographs, distance-hereditary graphs, convex graphs, and their complement graphs, our positive results reveal the polynomial-time solvability of various problems for these graph classes.

2012 ACM Subject Classification Mathematics of computing → Graph algorithms

Keywords and phrases mim-width, graph algorithm, NP-hardness, induced subgraph problem, cluster vertex deletion

Digital Object Identifier 10.4230/LIPIcs.SWAT.2024.38

Related Version *Full Version*: `https://arxiv.org/abs/2405.15492`

Funding *Yota Otachi*: JSPS KAKENHI Grant Numbers JP18H04091, JP20H05793, JP21K11752, JP22H00513.
Akira Suzuki: JSPS KAKENHI Grant Numbers JP20K11666, JP20H05794.
Yuma Tamura: JSPS KAKENHI Grant Number JP21K21278.

1 Introduction

Efficiently solving intractable graph problems by using structural graph parameters has been extensively studied over the past few decades. Tree-width is arguably one of the most successful parameters in this research direction. Courcelle's celebrated result indicates that every problem expressible in MSO_2 logic is solvable in linear time for bounded tree-width graphs [12]. Various graph problems, including Independent Set, Clique, Dominating Set, Independent Dominating Set, k-Coloring for a fixed k, Feedback Vertex Set, and Hamiltonian Cycle, can be written in MSO_2 logic, and hence Courcelle's theorem covers a wide range of problems. Later, Courcelle et al. also gave an analogous result for a more general parameter than tree-width, namely, clique-width: every problem expressible in MSO_1 logic is solvable in linear time for bounded clique-width graphs (under the assumption that a k-expression for a fixed k of an input graph is given) [13]. However, these results are not applicable directly to problems on interval graphs and permutation graphs, because these graph classes have unbounded clique-width (and thus unbounded tree-width).

In 2012, Vatshelle introduced mim-width [33], and recently, algorithms based on mim-width have been widely developed [1, 2, 3, 4, 5, 7, 8, 9, 15, 16, 21, 23, 24]. Roughly speaking, mim-width is an upper bound on the size of maximum induced matching along a branch decomposition of a graph. (In Section 2, its formal definition will be given.) Mim-width is a more general structural parameter than clique-width in the sense that the class of bounded mim-width graphs properly contains the class of bounded clique-width graphs. Furthermore, many graph classes of unbounded clique-width have bounded mim-width: for example, interval graphs, permutation graphs, convex graphs, k-polygon graphs for a fixed k, circular k-trapezoid graphs for a fixed k, and H-graphs for a fixed graph H. (See [1, 14] for more details.) Bergougnoux et al. gave an algorithmic meta-theorem [2], which states that every problem expressible in A&C DN logic is solvable in polynomial time for bounded mim-width graphs (under the assumption that a suitable branch decomposition of an input graph is given). Independent Set, Dominating Set, Independent Dominating Set, k-Coloring for a fixed k, Feedback Vertex Set etc. can be expressed in A&C DN logic. Thus, Bergougnoux et al. showed that many problems are solvable in polynomial time for a much wider range of graph classes than the class of bounded clique-width graphs.

Unfortunately, A&C DN logic does not cover all problems expressible in MSO_2 logic. Clique and Hamiltonian Cycle cannot be written in A&C DN logic, whereas they can be expressed in MSO_1 logic and MSO_2 logic, respectively. This means that the meta-theorem by Bergougnoux et al. is not applicable to these problems. In fact, it is known that Clique is NP-hard for graphs with linear mim-width[1] at most 6 [33] and Hamiltonian Cycle is NP-hard for graphs with linear mim-width 1 [23]. Note that by combining some known facts, we can show that Clique on graphs with mim-width at most 1 can be solved in polynomial time (see the discussion in the second paragraph of Section 4). These results lead us to ask the following questions:

- What kind of problems expressible in MSO_2 logic are NP-hard for bounded mim-width graphs?
- Is Clique NP-hard for graphs with mim-width less than 6?
- Given a graph with mim-width at most 1, which MSO_2-expressible problems are polynomial-time solvable?

1.1 Our contributions

To answer the questions above, in this paper, we systematically study the complexity of the Induced Π Subgraph problems and their complementary problems, called the Π Vertex Deletion problems, on bounded (linear) mim-width graphs. We first show that for any nontrivial hereditary graph property Π that admits all cliques, there is a constant w such that Induced Π Subgraph and Π Vertex Deletion are NP-hard for graphs with (linear) mim-width at most w. For example, Clique, Induced Cluster Subgraph, Induced Polar Subgraph, and Induced Split Subgraph satisfy the aforementioned conditions, and hence all of them are NP-hard for bounded (linear) mim-width graphs. As a

[1] The linear mim-width of a graph G is the mim-width when a branch decomposition of G is restricted to a caterpillar. The formal definition will be given in Section 2.

byproduct, we also show that connected and dominating variants of them are NP-hard for bounded (linear) mim-width graphs. Moreover, we give sufficient conditions for Induced Π Subgraph and Π Vertex Deletion to be NP-hard for graphs with (linear) mim-width at most 2. Clique, Induced Cluster Subgraph, Induced Polar Subgraph, and Induced Split Subgraph are proven to be in fact NP-hard even for graphs with (linear) mim-width at most 2. We thus reveal that there are various NP-hard problems for bounded mim-width graphs, although they can be expressed in MSO_2 logic. Especially, our result for Clique strengthens the known result that Clique is NP-hard for graphs with mim-width at most 6 [33].

To complement the intractability, we next seek polynomial-time solvable cases for graphs with mim-width at most 1. Here we focus on Induced Cluster Subgraph, also known as Cluster Vertex Deletion. Induced Cluster Subgraph is known to be NP-hard for bipartite graphs [19, 34], while it is solvable in polynomial time for split graphs, block graphs, interval graphs [10], cographs [27], bounded clique-width graphs [13], and convex graphs[2]. Surprisingly, the complexity status of Induced Cluster Subgraph on chordal graphs is still open. We show that, given a graph G with mim-width at most 1 accompanied by its branch decomposition with mim-width at most 1, Induced Cluster Subgraph is solvable in polynomial time. Although the complexity of computing a branch decomposition with mim-width at most 1 of a given graph is still open in general, our result yields a unified polynomial-time algorithm for Induced Cluster Subgraph that works on block graphs, interval graphs, permutation graphs, cographs, distance-hereditary graphs, convex graphs, and their complement graphs because all these graphs have mim-width at most 1 and their branch decompositions of mim-width at most 1 can be obtained in polynomial time [1, 20, 33][3]. Consequently, we give independent proofs for some of the results in [10, 27] via mim-width. Moreover, to the best of our knowledge, this is the first polynomial-time algorithm for Induced Cluster Subgraph on permutation graphs. We also emphasize that our algorithmic technique can be applied to other problems such as Induced Polar Subgraph, Induced Split Subgraph, and so on. Combining our results, we give the complexity dichotomy of the above problems with respect to mim-width.

Due to the space limitation, the proofs of claims marked ♠ are omitted in this paper, which can be found in the full version.

1.2 Previous work on mim-width

Mim-width is a relatively new graph structural parameter introduced by Vatshelle [33] and it has attracted much attention in recent years to design efficient algorithms of problems on graph classes that have unbounded tree-width and clique-width. Combined with the result of Belmonte and Vatshelle [1], Bui-Xuan et al. provided XP algorithms of Locally Checkable Vertex Subset and Vertex Partitioning problems (LC-VSVP for short) parameterized by mim-width w, assuming that a branch decomposition with mim-width w of a given graph can be computed in polynomial time [9]. Many problems, including Independent Set, Dominating Set, Independent Dominating Set, and k-Coloring, are expressible in the form of LC-VSVP. Jaffke et al. later generalized the result to the

[2] If a given graph is convex (more generally K_3-free), Induced Cluster Subgraph is equivalent to Induced Π Subgraph such that Π is the class of graphs with maximum degree at most 1, which is solvable in polynomial time for convex graphs [9].

[3] As far as we know, it was not explicitly stated in any literature that block graphs and distance-hereditary graphs have mim-width at most 1. This follows from the facts that a graph is distance-hereditary if and only if its rank-width is at most 1 [20], and block graphs are distance-hereditary graphs.

distance versions of LC-VSVP [21]. As the name suggests, LC-VSVP can capture problems whose solutions are defined only by local constraints. Longest Induced Path [23] and Feedback Vertex Set [24] are the first problems with global constraints for which it was shown that there exist XP algorithms parameterized by mim-width. Bergougnoux and Kanté designed a framework to deal with problems with global constraints for bounded mim-width graphs [3]. The remarkable meta-theorem given by Bergougnoux et al. is not only a generalization of all the above results in this section, but also a powerful tool for solving more complicated problems on bounded mim-width graphs [2]. Subset Feedback Vertex Set is one of the few examples where there exists an XP algorithm parameterized by mim-width [4] although the meta-theorem does not work for it.

Unfortunately, computing the mim-width of a given graph is W[1]-hard, and there is no polynomial-time approximation algorithm within constant factor unless NP = ZPP [32]. Even the complexity of determining whether a given graph has mim-width at most 1 is a long-standing open problem. Fortunately, it is known that various graph classes have constant mim-width and their branch decompositions with constant mim-width are computable in polynomial time [1, 7, 8, 14, 26, 30]. In particular, some famous graphs, such as block graphs, interval graphs, permutation graphs, cographs, distance-hereditary graphs, and convex graphs, have mim-width at most 1 and their branch decomposition with mim-width at most 1 can be obtained in polynomial time [1, 20]. The class of leaf power graphs, which is the more general class than interval graphs and block graphs, also have mim-width at most 1 [22], although it is not known whether an optimal branch decomposition of a given leaf power graph can be obtained in polynomial time. On the other hand, the following graph classes have unbounded mim-width: strongly chordal split graphs [29], co-comparability graphs [26, 29], circle graphs [29], and chordal bipartite graphs [6].

In contrast to a wealth of research on developing XP algorithms parameterized by mim-width and establishing lower and upper bounds on mim-width for specific graph classes, there has been limited research on the NP-hardness of problems for graph classes with constant mim-width [23, 25, 33].

2 Preliminaries

Let $G = (V, E)$ be a graph. We assume that all the graphs in this paper are simple, undirected, and unweighted. We denote by $V(G)$ and $E(G)$ the vertex set and the edge set of G, respectively. For a vertex v of G, we denote by $N(G; v)$ the *(open) neighborhood* of v in G, that is, $N(G; v) = \{w \in V \mid vw \in E\}$. The *degree* of a vertex v of G is the size of $N(G; v)$. For a vertex subset $V' \subseteq V$, we denote by $G[V']$ the subgraph induced by V'. We use the shorthand $G - V'$ for $G[V \setminus V']$. For positive integers i and j with $i \leq j$, we write $[i, j]$ as the shorthand for the set $\{i, i+1, \ldots, j\}$ of integers. In particular, we write $[1, j] = [j]$.

For two graphs $G_1 = (V_1, E_1)$ and $G_2 = (V_2, E_2)$ with $V_1 \cap V_2 = \emptyset$, the *disjoint union* of G_1 and G_2 is the graph whose vertex set is $V_1 \cup V_2$ and edge set is $E_1 \cup E_2$. For a graph H and a positive integer ℓ, ℓH means the disjoint union of ℓ copies of H. The *complement* of G, denoted by $\overline{G}$, is the graph on the same vertex set $V(G)$ with the edge set $\{uv \mid u, v \in V(G), uv \notin E(G)\}$. An *independent set* I of G is a vertex subset of G such that any two vertices in I are non-adjacent. A *clique* K of G is a vertex subset of G such that any two vertices in K are adjacent. Obviously, an independent set of G forms a clique of $\overline{G}$, and vice versa. A *dominating set* D of G is a vertex subset of G such that $N(G; v) \cap D \neq \emptyset$ for every vertex $v \in V(G) \setminus D$. A graph G is said to be *connected* if there is a path between any two vertices of G. A maximal connected subgraph of G is called a *connected component* of G. A *cut vertex* of G is a vertex whose removal from G increases the number of connected components.

2.1 Graph classes

A graph is *bipartite* if its vertex set can be partitioned into two independent sets. For disjoint vertex sets A and B of a graph G, we denote by $G[A, B]$ the *bipartite subgraph* with the vertex set $A \cup B$ and the edge set $\{ab \in E(G) \mid a \in A, b \in B\}$. A bipartite graph $G = (A \cup B, E)$ consisting of disjoint independent sets A and B is called a *chain graph* if there is an ordering $a_1, a_2, \ldots, a_{|A|}$ of vertices in A such that $N(G; a_1) \subseteq N(G; a_2) \subseteq \cdots \subseteq N(G; a_{|A|})$. Note that, if A has such an ordering, then B also has an ordering $b_1, b_2, \ldots, b_{|B|}$ of vertices in B such that $N(G; b_1) \subseteq N(G; b_2) \subseteq \cdots \subseteq N(G; b_{|B|})$.

A *tree* is a connected acyclic graph. A vertex of a tree is called a *leaf* if it has degree 1; otherwise, it is an *internal vertex*. A *rooted tree* T is a tree with a specific vertex r called the *root* of T. For a rooted tree T and two adjacent vertices x and y of T, we say that x is the *parent* of y, and conversely, y is a *child* of x if x lies on a path from y to r. A *full binary tree* is a rooted tree such that each vertex has zero or exactly two children. A tree T is a *caterpillar* if it contains a path P called a *spine* such that every leaf of T is adjacent to a vertex of P. In this paper, we assume that the spine P is maximum, that is, there is no path longer than P. The vertices of degree at most 1 in P are called the *endpoints* of P. A tree T is called *subcubic* if every internal vertex of T has degree exactly 3.

We denote by K_n and P_n the complete graph and the path graph with n vertices, respectively. We say that a graph G is *H-free* if G does not contain a graph isomorphic to H as an induced subgraph.

2.2 Mim-width

For an edge subset E' of a graph G, we denote $V(E') = \{v, w \in V(G) \mid vw \in E'\}$. An edge subset $M \subseteq E(G)$ is an *induced matching* of G if every vertex of $G[V(M)]$ has degree exactly 1. For a vertex subset $A \subseteq V(G)$, let $\mathsf{mim}(A)$ be the maximum size of an induced matching in the bipartite subgraph $G[A, \overline{A}]$, where $\overline{A} = V(G) \setminus A$.

A *branch decomposition* of a graph G is a pair (T, L), where T is a subcubic tree with $|V(G)|$ leaves and L is a bijection from $V(G)$ to the leaves of T. In particular, a branch decomposition (T, L) is called *linear* if T is a caterpillar. To distinguish vertices of T from those of the original graph G, we call the vertices of T *nodes*. For each edge e of T, as T is acyclic, removing e from T results in two trees T_1^e and T_2^e. Let (A_1^e, A_2^e) be a vertex bipartition of G, where $A_i^e = \{L^{-1}(\ell) \mid \ell \text{ is a leaf of } T_i^e\}$ for each $i \in \{1, 2\}$. The *mim-width* $\mathsf{mimw}(T, L)$ of a branch decomposition (T, L) of G is defined as $\max_{e \in E(T)} \mathsf{mim}(A_1^e)$. The *mim-width* $\mathsf{mimw}(G)$ of G is the minimum mim-width over all branch decompositions of G. Similarly, the *linear mim-width* $\mathsf{lmimw}(G)$ of G is the minimum mim-width over all linear branch decompositions of G. Note that $\mathsf{mimw}(G) \leq \mathsf{lmimw}(G)$ holds for any graph G.

In this paper, to make a branch decomposition easier to handle, we often consider its rooted variant. A *rooted layout* of a graph G is a pair (T', L), where T' is a rooted full binary tree with $|V(G)|$ leaves and L is a bijection from $V(G)$ to the leaves of T'. The mim-width of a rooted layout (T', L) is defined in the same way as a branch decomposition. A rooted layout of G is obtained from a branch decomposition (T, L) of G with the same mim-width by inserting a root r to an arbitrary edge of T. (If $|V(T)| = 1$, we regard the unique node of T as the root r of T'.)

Here we note propositions concerning mim-width. Vatshelle showed that for a graph G and a vertex $v \in V(G)$, it holds that $\mathsf{mimw}(G - v) \leq \mathsf{mimw}(G)$ [33]. One can see that the proof given by Vatshelle suggests the next proposition.

▶ **Proposition 1.** *For a graph G and an induced subgraph G' of G, it holds that* $\mathsf{mimw}(G') \leq \mathsf{mimw}(G)$ *and* $\mathsf{lmimw}(G') \leq \mathsf{lmimw}(G)$.

We here focus on graphs with mim-width at most 1. It is known that a graph G is a chain graph if and only if G is a bipartite graph with a maximum induced matching of size at most 1 [18]. Thus, we obtain the following proposition.

▶ **Proposition 2.** *Let (T, L) be a branch decomposition of a graph G. Then,* $\mathsf{mimw}(T, L) \leq 1$ *if and only if for any edge e of T, the bipartite subgraph $G[A_1^e, A_2^e]$ of G is a chain graph.*

Moreover, for a graph G and a vertex subset $A \subset V(G)$, it is not hard to see that $\mathsf{mim}(A) \leq 1$ on G if and only if $\mathsf{mim}(A) \leq 1$ on $\overline{G}$ from the definition of a chain graph. This implies the following proposition.

▶ **Proposition 3** ([33])**.** *Suppose that a graph G has mim-width at most 1. Then, any branch decomposition (T, L) of G with* $\mathsf{mimw}(T, L) \leq 1$ *is also the branch decomposition of $\overline{G}$ with* $\mathsf{mimw}(T, L) \leq 1$*. Consequently,* $\mathsf{mimw}(G) \leq 1$ *if and only if* $\mathsf{mimw}(\overline{G}) \leq 1$.

Combined with the observation that any cycle of length at least 5 has mim-width 2 and the strong perfect graph theorem [11], Proposition 3 leads to the following proposition.

▶ **Proposition 4** ([33])**.** *All graphs with mim-width at most 1 are perfect graphs.*

2.3 Graph properties and problems

Let Π be a fixed graph property. We often regard Π as a collection of graphs satisfying the graph property. A graph property Π is *nontrivial* if there exist infinitely many graphs satisfying Π and there exist infinitely many graphs that do not satisfy Π. A graph property Π is said to be *hereditary* if for any graph G satisfying Π, every induced subgraph of G also satisfies Π. We denote by $\overline{\Pi}$ the *complementary property* of Π, that is, $\overline{\Pi} = \{\overline{G} : G \in \Pi\}$.

For a graph G, a vertex subset $S \subseteq V(G)$ is called a Π-*set* of G if $G[S]$ satisfies Π. The Induced Π Subgraph problem asks for a Π-set S of maximum size for a given graph G. If $G[S]$ is also required to be connected, then the problem is called the Connected Induced Π Subgraph problem. For example, Independent Set is equivalent to Induced K_2-free Subgraph, and Clique is equivalent to Induced $2K_1$-free Subgraph and Connected Induced P_3-free Subgraph. Note that a vertex set S of G is a Π-set if and only if S is a $\overline{\Pi}$-set of $\overline{G}$. In Induced Π Subgraph, if the Π-set S is also required to be a dominating set of G, then the problem is called the Dominating Induced Π Subgraph problem.

Under the polynomial-time solvability, Induced Π Subgraph is equivalent to the Π Vertex Deletion problem, which asks for a minimum vertex subset S' of G such that $G - S'$ satisfies Π. The vertex subset S' is called a Π-*deletion set* of G. The Vertex Cover problem is equivalent to K_2-free Vertex Deletion. If $G[S']$ is also required to be connected, then the problem is called the Connected Π Vertex Deletion problem. In Π Vertex Deletion, if the Π-deletion set S' is also required to be a dominating set of G, then the problem is called the Dominating Π Vertex Deletion problem.

3 NP-hardness

In this section, we show the NP-hardness of Induced Π Subgraph and Π Vertex Deletion on graphs with linear mim-width at most w, where w is some constant.

▶ **Theorem 5.** *Let Π be a fixed nontrivial hereditary graph property that admits all cliques. Then there is a constant w such that* Induced Π Subgraph *and* Π Vertex Deletion*, as well as their connected variants and their dominating variants, are NP-hard for graphs with linear mim-width at most w, even if a branch decomposition with mim-width at most w of an input graph is given.*

Since Induced Π Subgraph is the complementary problem of Π Vertex Deletion, we only prove the hardness of Π Vertex Deletion.

The *girth* of a graph G is the length of a shortest cycle in G. We reduce Vertex Cover on graphs with girth at least 7, which is known to be NP-complete [31], to Π Vertex Deletion by following the classical reduction technique of Lewis and Yannakakis [28].

First, we define a sequence on a graph. Consider a graph H with p connected components $H_1, H_2, \dots, H_p$. Suppose that H_i for $i \in [p]$ has a cut vertex c and the removal of c from H_i results in q connected components $C_{i,1}, C_{i,2}, \dots, C_{i,q}$ with $|V(C_{i,1})| \geq |V(C_{i,2})| \geq \cdots \geq |V(C_{i,q})|$. For each $j \in [q]$, we denote by $H_{i,j}$ the subgraph induced by $V(C_{i,j}) \cup \{c\}$ and $n_{i,j} = |V(H_{i,j})|$. The cut vertex c gives a non-increasing sequence $\alpha_c = \langle n_{i,1}, n_{i,2}, \dots, n_{i,q}\rangle$. For two sequences α_c and $\alpha_{c'}$ according to cut vertices c and c' of H_i, we write $\alpha_{c'} <_L \alpha_c$ if $\alpha_{c'}$ is smaller than α_c in the sense of lexicographic order. Let α_i be the lexicographically smallest sequence among all sequences according to the cut vertices of H_i. If H_i has no cut vertex, we let $\alpha_i = \langle |V(H_i)| \rangle$. Define $\beta_H = \langle \alpha_1, \alpha_2, \dots, \alpha_p \rangle$, where we assume that $\alpha_1 \geq_L \alpha_2 \geq_L \cdots \geq_L \alpha_p$. For example, for the graph H depicted in Figure 1(a), we have $\beta_H = \langle \langle 4, 2\rangle, \langle 2, 2\rangle \rangle$. For two graphs H with p connected components and H' with q connected components, we write $\beta_{H'} <_R \beta_H$ if $\beta_{H'}$ is smaller than β_H in the sense of lexicographic order: more precisely, assuming that $\beta_H = \langle \alpha_1, \alpha_2, \dots, \alpha_p \rangle$ and $\beta_{H'} = \langle \alpha'_1, \alpha'_2, \dots, \alpha'_q \rangle$, there exists an integer $i \in [\min\{p, q\}]$ such that $\alpha'_j = \alpha_j$ for every $j \in [i-1]$ and $\alpha'_i <_L \alpha_i$; or $q < p$ and $\alpha'_i = \alpha_i$ for every $i \in [q]$.

Consider the complementary property $\overline{\Pi}$ of Π. Note that, since all cliques satisfy Π, all independent sets satisfy $\overline{\Pi}$. Let F be a graph satisfying the following two conditions:

1. there is an integer $\ell \geq 1$ such that ℓF violates $\overline{\Pi}$, whereas $(\ell - 1)F$ satisfies $\overline{\Pi}$; and
2. for any integer $\ell' \geq 1$ and any graph F' with $\beta_{F'} <_R \beta_F$, $\ell' F'$ satisfies $\overline{\Pi}$.

We call F the *base of* $\overline{\Pi}$*-forbidden subgraphs*. Notice that the existence of F is guaranteed because $\overline{\Pi}$ is nontrivial. Moreover, F and ℓ depend on Π solely and are independent of an instance of Vertex Cover, that is, F and ℓ are fixed.

Let $F_1, F_2, \dots, F_p$ be p connected components of F, where $\alpha_1 \geq_L \alpha_2 \geq_L \cdots \geq_L \alpha_p$. We denote by c_1 the cut vertex of F_1 that realizes α_1 (see Figure 1(a)) and by $F_{1,1}$ the induced subgraph of F_1 corresponding to $n_{1,1}$ (see Figure 1(b)). If F_1 has no cut vertex, then c_1 is any vertex of F_1. We then arbitrarily choose a vertex from $N(F_{1,1}; c_1)$ and label it as d. Notice that $N(F_{1,1}; c_1) \neq \emptyset$; otherwise, since $\alpha_1, \alpha_2, \dots, \alpha_p$ are lexicographically sorted, ℓF is an independent set and violates $\overline{\Pi}$, which contradicts that all independent sets satisfy $\overline{\Pi}$. Let F' be the graph obtained by removing $V(F_{1,1}) \setminus \{c_1\}$ from F (see Figure 1(c)).

We now construct an input graph G for Π Vertex Deletion from an input graph H with girth at least 7 for Vertex Cover. Let $n = |V(H)|$ and H^* be the disjoint union of ℓn copies of H. We assume that $n \geq 2$, $k < n - 1$, and H has at least one edge; otherwise, Vertex Cover is trivially solvable. For each vertex u of H^*, make a copy of F' and identify c_1 with u. For each edge uv of H^*, make a copy of $F_{1,1}$ and identify c_1 and d with u and v, respectively. (See Figure 2.) Let H' be the graph resulting from the above transformation. Finally, we let $G = \overline{H'}$. Since ℓ, F', and $F_{1,1}$ are fixed, G can be constructed in polynomial time in the size of H.

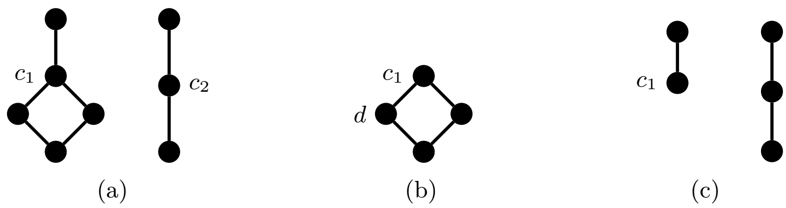

Figure 1 Let F be the graph depicted in (a). The cut vertex c_1 of the left connected component F_1 of F gives $\alpha_1 = \langle 4, 2 \rangle$ and the cut vertex c_2 of the right connected component F_2 of F gives $\alpha_2 = \langle 2, 2 \rangle$, where $\alpha_1 >_L \alpha_2$. Thus, if F is selected as the base of $\overline{\Pi}$-forbidden subgraphs, $F_{1,1}$ and F' are defined as the graphs depicted in (b) and (c), respectively.

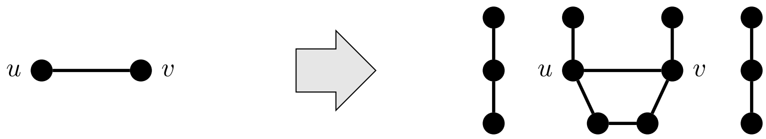

Figure 2 A transformation of an edge uv with $F_{1,1}$ and F', which are the graphs depicted in Figure 1(b) and (c), respectively.

In [28], it is shown that H has a vertex cover of size at most k if and only if H' has a $\overline{\Pi}$-deletion set S of size at most $k\ell n$. Notice that H' has $\ell n \geq 2$ connected components because H' is obtained from H^*, which is the disjoint union of ℓn copies of H. Moreover, we have the following lemma.

▶ **Lemma 6.** *Suppose that H' has a $\overline{\Pi}$-deletion set S of size at most $k\ell n$. Then the following two claims (a) and (b) are true:*

(a) *there are two connected components C_1 and C_2 of H' such that $V(C_1) \setminus S \neq \emptyset$ and $V(C_2) \setminus S \neq \emptyset$; and*

(b) *there are two connected components C'_1 and C'_2 of H' such that $V(C'_1) \cap S \neq \emptyset$ and $V(C'_2) \cap S \neq \emptyset$.*

Proof. In the claim (a), assume for a contradiction that there is at most one connected component C of H' such that $V(C) \setminus S \neq \emptyset$. In other words, $V(H' - C) \subseteq S$ holds. Recall that $k < n - 1$ and H' has $\ell n \geq 2$ connected components. Moreover, each connected component of $H' - C$ has at least n vertices from the construction of H'. Thus, we have

$$|S| \geq |V(H' - C)| \geq n(\ell n - 1) > (k + 1)(\ell n - 1) = k\ell n + \ell n - k - 1 > k\ell n,$$

a contradiction.

To prove the claim (b), assume for a contradiction that there is at most one connected component C' of H' such that $V(C') \cap S \neq \emptyset$. In other words, there are at least $\ell n - 1$ ($\geq \ell$ because $n \geq 2$) connected components of $H' - C'$ that contain no vertex in S. Consider ℓ connected components of $H' - C'$. Since each of them contains F as an induced subgraph, $H' - C'$ contains ℓF as an induced subgraph. However, ℓF violates $\overline{\Pi}$ because F is the base of $\overline{\Pi}$-forbidden subgraphs. This contradicts that S is a $\overline{\Pi}$-deletion set of H'. ◀

Observe that S is a $\overline{\Pi}$-deletion set of H' of size at most $k\ell n$ if and only if S is a Π-deletion set of $G = \overline{H'}$ of size at most $k\ell n$. Combined with Lemma 6, this implies that H has a vertex cover of size at most k if and only if G has a Π-deletion set S of size at most $k\ell n$ such that the induced subgraphs $G[S]$ and $G - S$ are both connected, and S and $V(G) \setminus S$ are dominating sets of G.

Our remaining task is to show that G has linear mim-width at most w for some constant w. To this end, we consider a sequence of subgraphs of H'. Let $V(H^*) = \{v_1, v_2, \dots, v_n\}$ and $E(H^*) = \{e_1, e_2, \dots, e_m\}$, where n and m are the numbers of vertices and edges of H^*, respectively. We make a sequence $\mathcal{H} = \langle H^* = H_0, H_1, \dots, H_{n+m} = H' \rangle$ such that H_i for $i \in [n]$ is obtained from H_{i-1} by attaching a copy of F' to v_i, and H_i for $i \in [n+1, n+m]$ is obtained from H_{i-1} by attaching a copy of $F_{1,1}$ to e_{i-n}. Since $G = \overline{H'}$, the following lemma completes the proof of Theorem 5. (Recall that F is fixed and hence $\mathsf{lmimw}(\overline{F})$ is a constant.)

▶ **Lemma 7.** *For any graph H_i in the sequence $\mathcal{H} = \langle H^* = H_0, H_1, \dots, H_{n+m} = H' \rangle$, a linear branch decomposition of H_i with mim-width at most $\mathsf{lmimw}(\overline{F}) + 2$ can be obtained in polynomial time in the size of H^*.*

Proof. We prove the lemma by induction, where the base case is $H_0 = H^*$. Note that H^* has girth at least 7 because H^* consists of copies of H whose girth is at least 7. Consider a linear branch decomposition (T_0, L_0) of $\overline{H_0}$, where L_0 is an arbitrary bijection from $V(H_0)$ to the leaves of T_0. To show that $\mathsf{mimw}(T_0, L_0) \le 2 \le \mathsf{lmimw}(\overline{F}) + 2$, assume for a contradiction that there is an edge e of T_0 such that $\mathsf{mim}(A_1^e) \ge 3$ for the bipartition (A_1^e, A_2^e). Let x_1x_2, y_1y_2, z_1z_2 be edges that form an induced matching in $G[A_1^e, A_2^e]$, where $x_1, y_1, z_1 \in A_1^e$ and $x_2, y_2, z_2 \in A_2^e$. Then, $x_1y_2, y_2z_1, z_1x_2, x_2y_1, y_1z_2, z_2x_1 \notin E(\overline{H})$ and hence they form a cycle of length 6 in H_0. This contradicts that the girth of H_0 is at least 7.

Consider the case of $i > 0$. We here define a *concatenation* of two linear branch decompositions. Let G_1 and G_2 be vertex-disjoint induced subgraphs of a graph G such that $V(G_1) \cup V(G_2) = V(G)$, and let (T_1, L_1) and (T_2, L_2) be linear branch decompositions of G_1 and G_2, respectively. A concatenation of (T_1, L_1) and (T_2, L_2) is to construct a new linear branch decomposition (T, L) of G as follows. For each $i \in \{1, 2\}$, let e_i be an edge incident to an endpoint of the spine of T_i. Insert nodes t_1 and t_2 into e_1 and e_2, respectively, and then connect t_1 and t_2 by an edge. (If $|V(T_i)| = 1$ for $i \in \{1, 2\}$, we define t_i as the unique node of T_i.) Observe that T is a subcubic caterpillar. Finally, set a bijection L from $V(G)$ to the leaves of T such that $L(v) = L_1(v)$ if $v \in V(G_1)$ and $L(v) = L_2(v)$ if $v \in V(G_2)$.

By the induction hypothesis, there exists a linear branch decomposition (T_{i-1}, L_{i-1}) of $\overline{H_{i-1}}$ such that $\mathsf{mimw}(T_{i-1}, L_{i-1}) \le \mathsf{lmimw}(\overline{F}) + 2$. Recall that H_i is constructed by attaching a copy of F' to v_i or a copy of $F_{1,1}$ to e_{i-n}. We denote by F_i the subgraph of H_i obtained by removing all vertices in $V(H_{i-1})$. We may assume that $|V(F_i)| \ge 1$; otherwise, $H_i = H_{i-1}$ and thus we immediately conclude that $\mathsf{lmimw}(\overline{H_i}) \le \mathsf{lmimw}(\overline{F}) + 2$. Let (T_i', L_i') be a linear branch decomposition of $\overline{F_i}$ such that $\mathsf{mimw}(T_i', L_i') \le \mathsf{lmimw}(\overline{F})$. Notice that, since $\overline{F_i}$ is an induced subgraph of $\overline{F}$, such a linear branch decomposition exists by Proposition 1. Moreover, it can be constructed in constant time because $\overline{F}$ is fixed. We define (T_i, L_i) as a linear branch decomposition obtained by a concatenation of (T_{i-1}, L_{i-1}) and (T_i', L_i'). Clearly, the construction of (T_i, L_i) can be done in polynomial time in the size of H^*.

To show that $\mathsf{mimw}(T_i, L_i) \le \mathsf{lmimw}(\overline{F}) + 2$, assume for a contradiction that there is an edge e of T_i such that the bipartite subgraph $G[A_1^e, A_2^e]$ of $\overline{H_i}$ has an induced matching M of size $\mathsf{lmimw}(\overline{F}) + 3$, where (A_1^e, A_2^e) is the bipartition of $V(H_i)$ given by e. From the construction of (T_i, L_i), the following two cases are considered: (I) $A_1^e \subseteq V(H_{i-1})$ and $V(F_i) \subseteq A_2^e$; and (II) $A_1^e \subseteq V(F_i)$ and $V(H_{i-1}) \subseteq A_2^e$.

Case (I). Let e' be an edge of T_{i-1} such that $A_1^{e'} = A_1^e$ and $A_2^{e'} = A_2^e \setminus V(F_i)$. If $V(M) \subseteq V(H_{i-1})$, then M is also an induced matching of the bipartite subgraph $G[A_1^{e'}, A_2^{e'}]$ defined by the linear branch decomposition (T_{i-1}, L_{i-1}). This implies that $\mathsf{mimw}(T_{i-1}, L_{i-1}) \ge |M| = \mathsf{lmimw}(\overline{F}) + 3$, which contradicts that $\mathsf{mimw}(T_{i-1}, L_{i-1}) \le \mathsf{lmimw}(\overline{F}) + 2$.

Without loss of generality, we assume that M has three distinct edges x_1x_2, y_1y_2, z_1z_2 such that $x_1, y_1, z_1 \in A_1^e \subseteq V(H_{i-1})$, $x_2 \in A_2^e \cap V(F_i)$, and $y_2, z_2 \in A_2^e$. Then, the sequence $\langle x_2, y_1, z_2, x_1, y_2, z_1, x_2 \rangle$ of vertices forms a cycle C of length 6 of H_i. If $i \in [n]$, as $x_2 \in V(F_i)$ is adjacent to at most one vertex in $V(H_{i-1})$ from the construction of H_i, then we have $y_1 = z_1$, a contradiction. Suppose that $i \in [n+1, n+m]$. Recall that, from the construction of H_i, each vertex of F_i is not adjacent to vertices in $V(H_{i-1})$ except for the endpoints of e_{i-n}. Since $x_2 \in V(F_i)$ is adjacent to the distinct vertices $y_1, z_1 \in V(H_{i-1})$, we have $e_{i-n} = y_1z_1$. Furthermore, $x_1 \in V(H_{i-1})$ is not adjacent to any vertex in $V(F_i)$ and hence we have $y_2, z_2 \in V(H_{i-1})$. Therefore, we obtain the cycle $C_1 = \langle y_1, z_2, x_1, y_2, z_1, y_1 \rangle$ with smaller length than that of C, where the vertices of C_1 are in $V(H_{i-1})$. Similarly, if C_1 contains vertices of F_j for $j \in [n+1, i]$, there exists a smaller cycle of H_{i-1} that contains no vertices of F_j. We eventually obtain a cycle C' of H of length less than 6, which contradicts that H has girth at least 7.

Case (II). Recall that at most two vertices in $V(H_{i-1})$, say u and w, are adjacent to some vertex in $V(F_i)$ on H_i and thus no vertex in $V(H_{i-1}) \setminus \{u, w\}$ is adjacent to any vertex in $V(F_i)$ on H_i. If some vertex in $V(M)$ is in $V(H_{i-1}) \setminus \{u, w\}$, then we can take three distinct edges $x_1x_2, y_1y_2, z_1z_2 \in M$ such that $x_1, y_1, z_1 \in A_1^e \subseteq V(F_i)$, $x_2 \in V(H_{i-1}) \setminus \{u, w\} \subseteq A_2^e$, and $y_2, z_2 \in A_2^e$. However, this implies that x_2 is adjacent to y_1 and z_1 on H_i, which contradicts that no vertex in $V(H_{i-1}) \setminus \{u, w\}$ is adjacent to any vertex in $V(F_i)$ on H_i.

If there is no vertex in $V(M)$ is in $V(H_{i-1}) \setminus \{u, w\}$, then there is an induced matching $M' \subseteq M$ of $G[A_1^e, A_2^e]$ such that $V(M') \subseteq V(F_i)$ and $|M'| \geq |M| - |V(M) \cap \{u, w\}| \geq \mathsf{lmimw}(\overline{F}) + 1$. For an edge e' of T_i' such that $A_1^{e'} = A_1^e$ and $A_2^{e'} = A_2^e \setminus V(H_{i-1})$, M' is also an induced matching of $G[A_1^{e'} A_2^{e'}]$ defined by the linear branch decomposition (T_i', L_i'). This implies that $\mathsf{mimw}(T_i', L_i') \geq \mathsf{lmimw}(\overline{F}) + 1$, which contradicts that $\mathsf{mimw}(T_i', L_i') \leq \mathsf{lmimw}(\overline{F})$. ◀

Refining the proof of Lemma 7 yields stronger claims for some problems. (See the full version of this paper.)

▶ **Theorem 8** (♠). *All the following problems, as well as their connected variants and their dominating variants, are NP-hard for graphs with linear mim-width 2: (i) Clique; (ii) Induced Cluster Subgraph; (iii) Induced Polar Subgraph (iv) Induced $\overline{P_3}$-free Subgraph; (v) Induced $\overline{K_3}$-free Subgraph; and (vi) Induced Split Subgraph. The NP-hardness for these problems holds even if a linear branch decomposition with mim-width at most 2 of an input graph is given.*

Theorem 8 strongly suggests that the complements of graphs with linear mim-width 2 have unbounded mim-width, because Independent Set, the complementary problem of Clique, is solvable in polynomial time for bounded mim-width graphs.

4 Polynomial-time algorithms for graphs with mim-width at most 1

A graph G is called a *cluster* if every connected component of G is a complete graph. Induced Cluster Subgraph is equivalent to Induced P_3-free Subgraph and Cluster Vertex Deletion (in terms of polynomial-time solvability). From Theorem 8, Induced Cluster Subgraph is NP-hard for graphs with linear mim-width at most 2.

Recall that all graphs with mim-width at most 1 are perfect graphs by Proposition 4. It is known that Clique is solvable in polynomial time for perfect graphs [17] and hence also for graphs with mim-width at most 1. In contrast, Induced Cluster Subgraph remains

NP-hard for bipartite graphs [19, 34], which are perfect graphs. Thus, the same argument as Clique is not applicable to Induced Cluster Subgraph. Nevertheless, assuming that a rooted layout (T, L) of an input graph with $\mathsf{mimw}(T, L) = 1$ is given, we design a polynomial-time algorithm for Induced Cluster Subgraph.

▶ **Theorem 9.** *Given a graph and its rooted layout of mim-width at most* 1*,* Induced Cluster Subgraph *is solvable in polynomial time.*

It is known that all interval graphs, permutation graphs, distance-hereditary graphs, and convex graphs have mim-width at most 1 and their rooted layout of mim-width at most 1 can be obtained in polynomial time [1, 20]. Moreover, by Proposition 3, rooted layouts with mim-width at most 1 for the complement of these graphs can also be obtained in polynomial time. Thus, our algorithm directly indicates the following corollary.

▶ **Corollary 10.** *There is an algorithm that solves* Induced Cluster Subgraph *in polynomial time for interval graphs, permutation graphs, distance-hereditary graphs, convex graphs, and their complements.*

Here we give an idea of our algorithm. For a rooted layout (T, L) of a given graph with mim-width at most 1, we compute an optimal solution by means of dynamic programming from the leaves to the root of T. To complete the computation in polynomial time, for each node t of T, we discard redundant partial solutions and store essential ones of polynomial size. This approach was also employed in the previous algorithmic work of mim-width [2, 3, 4, 9, 21, 23, 24]. Especially, an equivalence relation called the *d-neighbor equivalence* plays a key role in compressing partial solutions and designing XP algorithms parameterized by mim-width [2, 3, 4, 9, 21]. However, Theorem 8 suggests that the d-neighbor equivalence does not work for designing an algorithm for Induced Cluster Subgraph; otherwise, we would obtain an XP algorithm parameterized by mim-width, which is quite unlikely by Theorem 8. A rooted layout with mim-width at most 1 resolves the difficulty. Recall that $\mathsf{mimw}(T, L) \leq 1$ if and only if $G[A^e_1, A^e_2]$ for any edge e of T is a chain graph as in Proposition 2. This property allows us to give strict total orderings of vertices in A^e_1 and A^e_2 with respect to neighbors of vertices. We define new equivalence relations over the strict total orderings, which enables the dynamic programming to run in polynomial time.

Let G be a graph and $<_A$ be a strict total order on $A \subseteq V(G)$. For a vertex subset $C \subseteq A$, we denote by $\mathsf{head}(C, <_A)$ and $\mathsf{tail}(C, <_A)$ the largest and smallest vertices in C with respect to $<_A$, respectively. More precisely, for a vertex $u \in C$, $u = \mathsf{head}(C, <_A)$ if and only if $v <_A u$ for any vertex $v \in C \setminus \{u\}$, and $u = \mathsf{tail}(C, <_A)$ if and only if $u <_A w$ for any vertex $w \in C \setminus \{u\}$, respectively. (For the sake of convenience, we allow $C = \emptyset$ and in this case we let $\mathsf{head}(C, <_A) = \emptyset$ and $\mathsf{tail}(C, <_A) = \emptyset$.) For a subset $S \subseteq A$, a partition $(C_1, C_2, \dots, C_p)$ of S into p disjoint subsets $C_1, C_2, \dots, C_p$ is called a *component partition of S over G* if for every $i \in [p]$, the subgraph of G induced by C_i is a connected component of $G[S]$. We say that a partition $(C_1, C_2, \dots, C_p)$ of $S \subseteq A$ is *indexed by* $<_A$ if it holds that $\mathsf{head}(C_j, <_A) <_A \mathsf{head}(C_i, <_A)$ for any pair of integers i, j with $1 \leq i < j \leq p$.

For a non-empty vertex subset A of G such that $\mathsf{mim}(A) \leq 1$, a strict total order $<_A$ of A is called a *chain order* if for any two distinct vertices v, w in A, $v <_A w$ means $N(G; v) \setminus A \subseteq N(G; w) \setminus A$. (If $|A| = 1$, we define that the trivial strict total order of A is also a chain order.) By Proposition 2 and the definition of chain graphs, there is a chain order of A if and only if $\mathsf{mim}(A) \leq 1$. Note that $\mathsf{mim}(\overline{A}) \leq 1$ also holds and thus there is a chain order $<_{\overline{A}}$ of $\overline{A}$.

For subsets S_A and S'_A of A, let $(C_1, C_2, \dots, C_p)$ denote the component partition of S_A over G and let $(C'_1, C'_2, \dots, C'_q)$ denote the component partition of S'_A over G, where p and q are positive integers and both the component partitions are indexed by a chain order

$<_A$. We write $S_A \equiv^{\mathsf{cl}}_{G,<_A} S'_A$ if $\mathsf{head}(C_1, <_A) = \mathsf{head}(C'_1, <_A)$, $\mathsf{tail}(C_1, <_A) = \mathsf{tail}(C'_1, <_A)$, and $\mathsf{head}(C_2, <_A) = \mathsf{head}(C'_2, <_A)$. If the graph G and the chain order $<_A$ involved in the component partitions of S_A and S'_A are clear from the context, then we use the shorthand $\equiv^{\mathsf{cl}}_A$. It is not hard to see that $\equiv^{\mathsf{cl}}_A$ is an equivalence relation over subsets of A. A *representative* of S_A, denoted by $\mathsf{rep}_A(S_A)$, is the set $R = \{\mathsf{head}(C_1, <_A), \mathsf{tail}(C_1, <_A), \mathsf{head}(C_2, <_A)\}$. In the same way, we define an equivalence relation $\equiv^{\mathsf{cl}}_{<_{\overline{A}}}$ over subsets of $\overline{A}$ according to a chain order $<_{\overline{A}}$ and a representative $\mathsf{rep}_{\overline{A}}(S_{\overline{A}})$ of $S_{\overline{A}} \subseteq \overline{A}$.

Consider two subsets $S_A, S'_A \subseteq A$ with $|S_A| \geq |S'_A|$. Assume that for any subset $S_{\overline{A}} \subseteq \overline{A}$, $S_A \cup S_{\overline{A}}$ is a cluster set of G if and only if $S'_A \cup S_{\overline{A}}$ is a cluster set of G. This suggests that there is no need to store S'_A during dynamic programming over T. Formally, we give the following lemma.

▶ **Lemma 11** (♠). *For a vertex subset A of a graph G such that $\mathsf{mim}(A) \leq 1$, let $S_A, S'_A \subseteq A$ be cluster sets of G with $S_A \equiv^{\mathsf{cl}}_A S'_A$ and let $S_{\overline{A}}$ be any subset of $\overline{A}$. Then, $S_A \cup S_{\overline{A}}$ is a cluster set of G if and only if $S'_A \cup S_{\overline{A}}$ is a cluster set of G.*

Lemma 11 asserts that the equivalence relation $\equiv^{\mathsf{cl}}_A$ allows us to determine vertex sets to be stored. However, Lemma 11 is not enough to construct a dynamic programming algorithm. If a chain order is arbitrarily given for each node t of T, then the ordering of the stored sets may change, which causes the algorithm to output an incorrect solution. To avoid the inconsistency, we need to define chain orders with additional constraints.

Let (T, L) be a rooted layout of a graph $G = (V, E)$. For a node t of T, we denote by T_t the subtree of T rooted at t. We define $V_t = \{L^{-1}(\ell) \mid \ell \text{ is a leaf of } T_t\}$, $\overline{V_t} = V \setminus V_t$, $G_t = G[V_t]$, and $G_{\overline{t}} = G[\overline{V_t}]$. We use the shorthand notations $G_{t,\overline{t}}$ for the bipartite subgraph $G[V_t, \overline{V_t}]$ and rep_t for the representative rep_{V_t}. We define a strict total order $<_t$ on vertices in V_t, called a *lower chain order*, that satisfies the two conditions below:

(ℓ-1) $<_t$ is a chain order of V_t; and

(ℓ-2) if t has a child c, then for any pair of distinct vertices v, w in V_c, it holds that $v <_c w$ if and only if $v <_t w$.

We also define an *upper chain order* $<_{\overline{t}}$ as a strict total order on vertices in $\overline{V_t}$ that holds the following three conditions:

(u-1) $<_{\overline{t}}$ is a chain order of $\overline{V_t}$;

(u-2) if t has a child c, then for any pair of distinct vertices v, w in $\overline{V_t}$, it holds that $v <_{\overline{t}} w$ if and only if $v <_{\overline{c}} w$; and

(u-3) if t has the parent p, then for any pair of distinct vertices v, w in $\overline{V_t} \cap V_p$, it holds that $v <_{\overline{t}} w$ if and only if $v <_p w$, where $<_p$ is a lower chain order on V_p.

Lemma 12 asserts that the above strict total orders can be found in polynomial time.

▶ **Lemma 12** (♠). *Let (T, L) be a rooted layout of a graph G with $\mathsf{mimw}(T, L) \leq 1$. For every node t of T, a lower chain order $<_t$ and an upper chain order $<_{\overline{t}}$ exist and can be obtained in polynomial time.*

We here give the following two lemmas, which are keys to show the correctness of our algorithm given later.

▶ **Lemma 13** (♠). *Let (T, L) be a rooted layout of a graph G with $\mathsf{mimw}(T, L) \leq 1$ and t be an internal node of T with a child c. For any subset $S \subseteq \overline{V_c} \cap V_t$ of G, it holds that $\mathsf{rep}_{\overline{c}}(S) = \mathsf{rep}_t(S)$.*

▶ **Lemma 14** (♠). *Let (T, L) be a rooted layout of a graph G with $\mathsf{mimw}(T, L) \leq 1$ and let t be an internal node of T with children a and b. For disjoint cluster sets $X \subseteq V_a$ and $Y \subseteq V_b$, if $X \cup Y$ is a cluster set of G, then $\mathsf{rep}_t(X \cup Y) = \mathsf{rep}_t(\mathsf{rep}_t(X) \cup \mathsf{rep}_t(Y))$ holds. Moreover, for a cluster set $Z \subseteq \overline{V_t}$ of G, if $X \cup Z$ (resp. $Y \cup Z$) is a cluster set of G, then $\mathsf{rep}_{\overline{b}}(X \cup Z) = \mathsf{rep}_{\overline{b}}(\mathsf{rep}_{\overline{b}}(X) \cup \mathsf{rep}_{\overline{b}}(Z))$ (resp. $\mathsf{rep}_{\overline{a}}(Y \cup Z) = \mathsf{rep}_{\overline{a}}(\mathsf{rep}_{\overline{a}}(Y) \cup \mathsf{rep}_{\overline{a}}(Z))$) holds.*

We now provide a polynomial-time algorithm for Induced Cluster Subgraph. Suppose that (T, L) is a rooted layout of a graph G with $\mathsf{mimw}(T, L) \leq 1$ and t is a node of T. We let $\mathscr{R}_t = \{\mathsf{rep}_t(S_t) : S_t \subseteq V_t\}$ and $\mathscr{R}_{\overline{t}} = \{\mathsf{rep}_{\overline{t}}(S_{\overline{t}}) : S_{\overline{t}} \subseteq \overline{V_t}\}$. For two sets $R_t \in \mathscr{R}_t$ and $R_{\overline{t}} \in \mathscr{R}_{\overline{t}}$, we define $f_t(R_t, R_{\overline{t}})$ as the function that returns the largest size of a subset $S_t \subseteq V_t$ such that

1. $\mathsf{rep}_t(S_t) = R_t$; and
2. $S_t \cup R_{\overline{t}}$ is a cluster set of G.

We let $f_t(R_t, R_{\overline{t}}) = -\infty$ if there is no subset satisfying the above conditions. For each triple of $t \in V(T)$, $R_t \in \mathscr{R}_t$, and $R_{\overline{t}} \in \mathscr{R}_{\overline{t}}$, we compute $f_t(R_t, R_{\overline{t}})$ by means of dynamic programming from the leaves to the root r of T. As $G = G_r$, we obtain the maximum size of cluster sets of G by computing $\min\{f_r(R_r, \emptyset) : R_r \in \mathscr{R}_r\}$. Notice that, for simplicity, our algorithm computes the size of an optimal solution. One can easily modify our algorithm so that it finds the largest cluster set in the same time complexity.

The case where t is a leaf of T. Denote by v the unique vertex in V_t. Then, $\mathscr{R}_t = \{\emptyset, \{v\}\}$. If $R_t = \emptyset$, only $S_t = \emptyset$ satisfies the prescribed conditions for any $R_{\overline{t}} \in \mathscr{R}_{\overline{t}}$. If $R_t = \{v\}$, then $S_t = \{v\}$ and we have to check that $\{v\} \cup R_{\overline{t}}$ is a cluster set of G. In summary, we have

$$f_t(R_t, R_{\overline{t}}) = \begin{cases} 0 & \text{if } R_t = \emptyset \text{ and } R_{\overline{t}} \text{ is a cluster set of } G, \\ 1 & \text{if } R_t = \{v\} \text{ and } \{v\} \cup R_{\overline{t}} \text{ is a cluster set of } G, \\ -\infty & \text{otherwise.} \end{cases}$$

The case where t is an internal node of T. Suppose that t has children a and b, and $f_a(R_a, R_{\overline{a}})$ and $f_b(R_b, R_{\overline{b}})$ have already been computed for any $R_a \in \mathscr{R}_a$, $R_{\overline{a}} \in \mathscr{R}_{\overline{a}}$, $R_b \in \mathscr{R}_b$, and $R_{\overline{b}} \in \mathscr{R}_{\overline{b}}$. For the largest subset $S_t \subseteq V_t$ that satisfies the prescribed conditions, S_t can be partitioned into two cluster sets $S_t \cap V_a$ and $S_t \cap V_b$. In addition, $(S_t \cap V_b) \cup R_{\overline{t}}$ and $(S_t \cap V_a) \cup R_{\overline{t}}$ form cluster sets of $G[V_{\overline{a}}]$ and $G[V_{\overline{b}}]$, respectively. We guess that $\mathsf{rep}_t(S_t \cap V_a) = \mathsf{rep}_a(S_t \cap V_a) = R_a \in \mathscr{R}_a$ and $\mathsf{rep}_t(S_t \cap V_b) = \mathsf{rep}_b(S_t \cap V_b) = R_b \in \mathscr{R}_b$. By Lemma 14, R_t can be represented as follows:

$$\begin{aligned} R_t &= \mathsf{rep}_t(S_t) \\ &= \mathsf{rep}_t((S_t \cap V_a) \cup (S_t \cap V_b)) \\ &= \mathsf{rep}_t(\mathsf{rep}_t(S_t \cap V_a) \cup \mathsf{rep}_t(S_t \cap V_b)) \\ &= \mathsf{rep}_t(R_a \cup R_b). \end{aligned}$$

To obtain the value $f_t(R_t, R_{\overline{t}})$, we calculate the sum of $f_a(R_a, \mathsf{rep}_{\overline{a}}((S_t \cap V_b) \cup R_{\overline{t}}))$ and $f_b(R_b, \mathsf{rep}_{\overline{b}}((S_t \cap V_a) \cup R_{\overline{t}}))$ for each pair (R_a, R_b) such that $R_a \in \mathscr{R}_a$, $R_b \in \mathscr{R}_b$, and $R_t = \mathsf{rep}_t(R_a \cup R_b)$. Combining Lemmas 13 and 14 with $\mathsf{rep}_{\overline{a}}(R_{\overline{t}}) = \mathsf{rep}_{\overline{t}}(R_{\overline{t}})$, which is observed from the condition (u-2) for an upper chain order, it holds that

$$\begin{aligned} \mathsf{rep}_{\overline{a}}((S_t \cap V_b) \cup R_{\overline{t}}) &= \mathsf{rep}_{\overline{a}}(\mathsf{rep}_{\overline{a}}(S_t \cap V_b) \cup \mathsf{rep}_{\overline{a}}(R_{\overline{t}})) \\ &= \mathsf{rep}_{\overline{a}}(\mathsf{rep}_t(S_t \cap V_b) \cup \mathsf{rep}_{\overline{t}}(R_{\overline{t}})) \\ &= \mathsf{rep}_{\overline{a}}(R_b \cup R_{\overline{t}}). \end{aligned}$$

Similarly, we have $\mathsf{rep}_{\overline{b}}((S_t \cap V_a) \cup R_{\overline{t}}) = \mathsf{rep}_{\overline{b}}(R_a \cup R_{\overline{t}})$. We conclude that

$$f_t(R_t, R_{\overline{t}}) = \max_{R_a \in \mathscr{R}_a \wedge R_b \in \mathscr{R}_b} \{f_a(R_a, \mathsf{rep}_{\overline{a}}(R_b \cup R_{\overline{t}})) + f_b(R_b, \mathsf{rep}_{\overline{b}}(R_a \cup R_{\overline{t}})) : R_t = \mathsf{rep}_t(R_a \cup R_b)\}.$$

Since $\mathscr{R}_t$ and $\mathscr{R}_{\overline{t}}$ are of polynomial size for every t of T, our algorithm runs in polynomial time. This completes the proof of Theorem 9.

We can extend the above algorithm to other several problems. (For more details, see the full version of this paper.) Combined with Theorem 8, we obtain the following dichotomy theorem.

▶ **Theorem 15** (♠). *All the following problems, as well as their connected variants and their dominating variants, are NP-hard for graphs with mim-width at most* 2*: (i)* Clique*; (ii)* Induced Cluster Subgraph*; (iii)* Induced Polar Subgraph*; (iv)* Induced $\overline{P_3}$-free Subgraph*; (v)* Induced Split Subgraph*; and (vi)* Induced $\overline{K_3}$-free Subgraph*. On the other hand, given a graph and its branch decomposition of mim-width at most* 1*, all the above problems, as well as their connected variants and their dominating variants, are solvable in polynomial time.*

5 Concluding remarks

We discuss future work here. Our proof of Theorem 5 relies on the assumption that all cliques satisfy a fixed property Π, and hence Theorem 5 is not applicable to Induced Π Subgraph such that Π excludes some clique. Such problems include Independent Set, Induced Matching, Longest Induced Path, and Feedback Vertex Set. In fact, there exist XP algorithms of the problems listed above when parameterized by mim-width [1, 9, 23, 24]. This motivates us to seek Π such that Induced Π Subgraph is NP-hard for bounded mim-width graphs although Π excludes some clique. As the first step, it would be interesting to consider Induced K_3-free Subgraph.

In [2], Bergougnoux et al. showed that Clique is expressible in A&C DN $+\,\forall$, which is A&C DN logic that allows to use a single universal quantifier $\forall$, and hence their meta-theorem cannot be extended to A&C DN $+\,\forall$. Our results in this paper suggest that the barrier could be broken down for graphs with mim-width at most 1. The next goal is to obtain a more general logic than A&C DN such that all problems expressible in the logic are solvable in polynomial time for graphs with mim-width at most 1.

Finally, we end this paper by leaving the biggest open problem concerning mim-width: Given a graph G, is there a polynomial-time algorithm that computes a branch decomposition with mim-width 1, or concludes that G has mim-width more than 1?

References

1 Rémy Belmonte and Martin Vatshelle. Graph classes with structured neighborhoods and algorithmic applications. *Theoretical Computer Science*, 511:54–65, 2013. doi:10.1016/j.tcs.2013.01.011.

2 Benjamin Bergougnoux, Jan Dreier, and Lars Jaffke. A logic-based algorithmic meta-theorem for mim-width. In Nikhil Bansal and Viswanath Nagarajan, editors, *Proceedings of the 2023 ACM-SIAM Symposium on Discrete Algorithms (SODA 2023)*, pages 3282–3304, 2023. doi:10.1137/1.9781611977554.ch125.

3 Benjamin Bergougnoux and M. Moustapha Kanté. More applications of the d-neighbor equivalence: Acyclicity and connectivity constraints. *SIAM Journal on Discrete Mathematics*, 35(3):1881–1926, 2021. doi:10.1137/20m1350571.

4 Benjamin Bergougnoux, Charis Papadopoulos, and Jan Arne Telle. Node multiway cut and subset feedback vertex set on graphs of bounded mim-width. *Algorithmica*, 84(5):1385–1417, 2022. doi:10.1007/s00453-022-00936-w.

5 Flavia Bonomo-Braberman, Nick Brettell, Andrea Munaro, and Daniël Paulusma. Solving problems on generalized convex graphs via mim-width. *J. Comput. Syst. Sci.*, 140:103493, 2024. doi:10.1016/J.JCSS.2023.103493.

6 Johann Brault-Baron, Florent Capelli, and Stefan Mengel. Understanding model counting for beta-acyclic CNF-formulas. In Ernst W. Mayr and Nicolas Ollinger, editors, *32nd International Symposium on Theoretical Aspects of Computer Science, STACS 2015*, volume 30 of *LIPIcs*, pages 143–156. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2015. doi:10.4230/LIPIcs.STACS.2015.143.

7 Nick Brettell, Jake Horsfield, Andrea Munaro, Giacomo Paesani, and Daniël Paulusma. Bounding the mim-width of hereditary graph classes. *Journal of Graph Theory*, 99(1):117–151, 2022. doi:10.1002/jgt.22730.

8 Nick Brettell, Jake Horsfield, Andrea Munaro, and Daniël Paulusma. List k-colouring P_t-free graphs: A mim-width perspective. *Information Processing Letters*, 173:106168, 2022. doi:10.1016/j.ipl.2021.106168.

9 Binh-Minh Bui-Xuan, Jan Arne Telle, and Martin Vatshelle. Fast dynamic programming for locally checkable vertex subset and vertex partitioning problems. *Theoretical Computer Science*, 511:66–76, 2013. doi:10.1016/j.tcs.2013.01.009.

10 Yixin Cao, Yuping Ke, Yota Otachi, and Jie You. Vertex deletion problems on chordal graphs. *Theoretical Computer Science*, 745:75–86, 2018. doi:10.1016/j.tcs.2018.05.039.

11 Maria Chudnovsky, Neil Robertson, Paul Seymour, and Robin Thomas. The strong perfect graph theorem. *Annals of Mathematics*, 164(1):51–229, 2006. doi:10.4007/annals.2006.164.51.

12 Bruno Courcelle. The monadic second-order logic of graphs. I. Recognizable sets of finite graphs. *Information and Computation*, 85(1):12–75, 1990. doi:10.1016/0890-5401(90)90043-H.

13 Bruno Courcelle, Johann A. Makowsky, and Udi Rotics. Linear time solvable optimization problems on graphs of bounded clique-width. *Theory of Computing Systems*, 33(2):125–150, 2000. doi:10.1007/s002249910009.

14 Fedor V. Fomin, Petr A. Golovach, and Jean-Florent Raymond. On the tractability of optimization problems on H-graphs. *Algorithmica*, 82(9):2432–2473, 2020. doi:10.1007/s00453-020-00692-9.

15 Esther Galby, Paloma T. Lima, and Bernard Ries. Reducing the domination number of graphs via edge contractions and vertex deletions. *Discrete Mathematics*, 344(1):112169, 2021. doi:10.1016/j.disc.2020.112169.

16 Esther Galby, Andrea Munaro, and Bernard Ries. Semitotal domination: New hardness results and a polynomial-time algorithm for graphs of bounded mim-width. *Theoretical Computer Science*, 814:28–48, 2020. doi:10.1016/j.tcs.2020.01.007.

17 Martin Grötschel, László Lovász, and Alexander Schrijver. *Geometric Algorithms and Combinatorial Optimization*, volume 2 of *Algorithms and Combinatorics*. Springer, 1988. doi:10.1007/978-3-642-97881-4.

18 Peter L. Hammer, Uri N. Peled, and Xiaorong Sun. Difference graphs. *Discrete Applied Mathematics*, 28(1):35–44, 1990. doi:10.1016/0166-218X(90)90092-Q.

19 Sun-Yuan Hsieh, Hoàng-Oanh Le, Van Bang Le, and Sheng-Lung Peng. On the d-claw vertex deletion problem. *Algorithmica*, 86(2):505–525, 2024. doi:10.1007/S00453-023-01144-W.

20 Sang il Oum. Rank-width and vertex-minors. *Journal of Combinatorial Theory, Series B*, 95(1):79–100, 2005. doi:10.1016/j.jctb.2005.03.003.

21 Lars Jaffke, O-joung Kwon, Torstein J. F. Strømme, and Jan Arne Telle. Mim-width III. Graph powers and generalized distance domination problems. *Theoretical Computer Science*, 796:216–236, 2019. doi:10.1016/j.tcs.2019.09.012.

22 Lars Jaffke, O-joung Kwon, and Jan Arne Telle. A unified polynomial-time algorithm for feedback vertex set on graphs of bounded mim-width. In Rolf Niedermeier and Brigitte Vallée, editors, *35th Symposium on Theoretical Aspects of Computer Science (STACS 2018)*, volume 96 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 42:1–42:14, Dagstuhl, Germany, 2018. Schloss Dagstuhl – Leibniz-Zentrum für Informatik. doi:10.4230/LIPIcs.STACS.2018.42.

23 Lars Jaffke, O-joung Kwon, and Jan Arne Telle. Mim-width I. Induced path problems. *Discrete Applied Mathematics*, 278:153–168, 2020. doi:10.1016/j.dam.2019.06.026.

24 Lars Jaffke, O-joung Kwon, and Jan Arne Telle. Mim-width II. The feedback vertex set problem. *Algorithmica*, 82(1):118–145, 2020. doi:10.1007/s00453-019-00607-3.

25 Lars Jaffke, Paloma T. Lima, and Roohani Sharma. Structural parameterizations of b-coloring. In Satoru Iwata and Naonori Kakimura, editors, *34th International Symposium on Algorithms and Computation (ISAAC 2023)*, volume 283 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 40:1–40:14, Dagstuhl, Germany, 2023. Schloss Dagstuhl – Leibniz-Zentrum für Informatik. doi:10.4230/LIPIcs.ISAAC.2023.40.

26 Dong Yeap Kang, O-joung Kwon, Torstein J.F. Strømme, and Jan Arne Telle. A width parameter useful for chordal and co-comparability graphs. *Theoretical Computer Science*, 704:1–17, 2017. doi:10.1016/j.tcs.2017.09.006.

27 Hoang-Oanh Le and Van Bang Le. Complexity of the cluster vertex deletion problem on H-free graphs. In Stefan Szeider, Robert Ganian, and Alexandra Silva, editors, *47th International Symposium on Mathematical Foundations of Computer Science (MFCS 2022)*, volume 241 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 68:1–68:10. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2022. doi:10.4230/LIPIcs.MFCS.2022.68.

28 John M. Lewis and Mihalis Yannakakis. The node-deletion problem for hereditary properties is NP-complete. *Journal of Computer and System Sciences*, 20(2):219–230, 1980. doi:10.1016/0022-0000(80)90060-4.

29 Stefan Mengel. Lower bounds on the mim-width of some graph classes. *Discrete Applied Mathematics*, 248:28–32, 2018. doi:10.1016/j.dam.2017.04.043.

30 Andrea Munaro and Shizhou Yang. On algorithmic applications of sim-width and mim-width of (H_1, H_2)-free graphs. *Theoretical Computer Science*, 955:113825, 2023. doi:10.1016/j.tcs.2023.113825.

31 Svatopluk Poljak. A note on stable sets and colorings of graphs. *Commentationes Mathematicae Universitatis Carolinae*, 15(2):307–309, 1974.

32 Sigve Hortemo Sæther and Martin Vatshelle. Hardness of computing width parameters based on branch decompositions over the vertex set. *Theoretical Computer Science*, 615:120–125, 2016. doi:10.1016/j.tcs.2015.11.039.

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

34 Mihalis Yannakakis. Node-deletion problems on bipartite graphs. *SIAM Journal on Computing*, 10(2):310–327, 1981. doi:10.1137/0210022.