Theoretical Computer Science 1032 (2025) 115088

Contents lists available at ScienceDirect

Theoretical Computer Science

journal homepage: www.elsevier.com/locate/tcs

Computing subset vertex covers in H-free graphs ☆

Nick Brettell [a,1], Jelle J. Oostveen [b,2], Sukanya Pandey [c], Daniël Paulusma [d], Johannes Rauch [e,3], Erik Jan van Leeuwen [b,*]

[a] Victoria University of Wellington, Wellington, New Zealand
[b] Utrecht University, Utrecht, the Netherlands
[c] RWTH Aachen University, Aachen, Germany
[d] Durham University, Durham, United Kingdom
[e] Ulm University, Ulm, Germany

A B S T R A C T

We consider a natural generalization of Vertex Cover: the Subset Vertex Cover problem, which is to decide for a graph $G=(V,E)$, a subset $T \subseteq V$ and integer k, if V has a subset S of size at most k, such that S contains at least one end-vertex of every edge incident to a vertex of T. A graph is H-free if it does not contain H as an induced subgraph. We solve two open problems from the literature by proving that Subset Vertex Cover is NP-complete on subcubic (claw, diamond)-free planar graphs and on 2-unipolar graphs, a subclass of $2P_3$-free weakly chordal graphs. Our results show for the first time that Subset Vertex Cover is computationally harder than Vertex Cover (under $\mathsf{P} \neq \mathsf{NP}$). We also prove new polynomial time results, some of which follow from a reduction to Vertex Cover restricted to classes of probe graphs. We first give a dichotomy on graphs where $G[T]$ is H-free. Namely, we show that Subset Vertex Cover is polynomial-time solvable on graphs G, for which $G[T]$ is H-free, if $H = sP_1 + tP_2$ and NP-complete otherwise. Moreover, we prove that Subset Vertex Cover is polynomial-time solvable for $(sP_1 + P_2 + P_3)$-free graphs and bounded mim-width graphs. By combining our new results with known results we obtain a partial complexity classification for Subset Vertex Cover on H-free graphs.

1. Introduction

We consider a natural generalization of the classical Vertex Cover problem: the Subset Vertex Cover problem, introduced in [8]. Let $G=(V,E)$ be a graph and T be a subset of V. A set $S \subseteq V$ is a *T-vertex cover* of G if S contains at least one end-vertex of every edge incident to a vertex of T. We note that T itself is a T-vertex cover. However, a graph may have much smaller T-vertex covers. For example, if G is a star whose leaves form T, then the centre of G forms a T-vertex cover. We can now define the problem; see also Fig. 1.

Subset Vertex Cover	
Instance:	A graph $G=(V,E)$, a subset $T \subseteq V$, and a positive integer k.
Question:	Does G have a T-vertex cover S_T with $\lvert S_T \rvert \leq k$?

☆ This article belongs to Section A: Algorithms, automata, complexity and games, Edited by Paul Spirakis.

* Corresponding author.
E-mail addresses: nick.brettell@vuw.ac.nz (N. Brettell), j.j.oostveen@uu.nl (J.J. Oostveen), pandey@algo.rwth-aachen.de (S. Pandey), daniel.paulusma@durham.ac.uk (D. Paulusma), johannes.rauch@uni-ulm.de (J. Rauch), e.j.vanleeuwen@uu.nl (E.J. van Leeuwen).

1 Nick Brettell was supported by the New Zealand Marsden Fund.
2 Jelle Oostveen was supported by the NWO grant OCENW.KLEIN.114 (PACAN).
3 Johannes Rauch was supported by the German Academic Scholarship Foundation (Studienstiftung des Deutschen Volkes).

https://doi.org/10.1016/j.tcs.2025.115088
Received 3 November 2023; Received in revised form 16 December 2024; Accepted 25 January 2025

Available online 28 January 2025

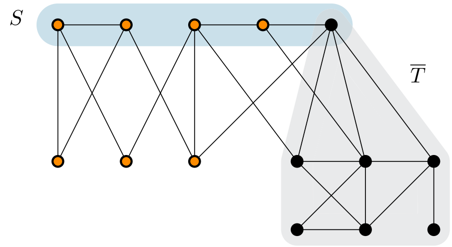

Fig. 1. An instance (G,T,k) of Subset Vertex Cover, where T consists of the orange vertices, together with a solution S (a T-vertex cover of size 5). Note that S consists of four vertices of T and one vertex of $\overline{T} = V \setminus T$. (For interpretation of the colours in the figure(s), the reader is referred to the web version of this article.)

If we set $T = V$, then we obtain the Vertex Cover problem. Hence, as Vertex Cover is NP-complete, so is Subset Vertex Cover.

To obtain a better understanding of the complexity of an NP-complete graph problem, we may restrict the input to some special graph class. In particular, *hereditary* graph classes, which are the classes closed under vertex deletion, have been studied intensively for this purpose. It is readily seen that a graph class $\mathcal{G}$ is hereditary if and only if $\mathcal{G}$ is characterized by a unique minimal set of forbidden induced subgraphs $\mathcal{F}_G$. Hence, for a systematic study, it is common to first consider the case where $\mathcal{F}_{\mathcal{G}}$ has size 1. This is also the approach we follow in this paper. So, for a graph H, we set $\mathcal{F}_{\mathcal{G}} = \{H\}$ for some graph H and consider the class of H-free graphs (graphs that do not contain H as an induced subgraph). We now consider the following research question:

For which graphs H is Subset Vertex Cover, restricted to H-free graphs, still NP-complete and for which graphs H does it become polynomial-time solvable?

We will also address two open problems posed in [8] (see Section 2 for any undefined terminology):

Q1. What is the complexity of Subset Vertex Cover for claw-free graphs?
Q2. Is Subset Vertex Cover NP-complete for P_t-free graphs for some t?

The first question is of interest, as Vertex Cover is polynomial-time solvable even on $rK_{1,3}$-free graphs for every $r \geq 1$ [6], where $rK_{1,3}$ is the disjoint union of r claws (previously this was known for rP_3-free graphs [24] and $2P_3$-free graphs [25]). The second question is of interest due to some recent quasi-polynomial-time results. Namely, Gartland and Lokshtanov [18] proved that for every integer t, Vertex Cover can be solved in $n^{O(\log^3 n)}$-time for P_t-free graphs. Afterwards, Pilipczuk, Pilipczuk and Rzążewski [31] improved the running time to $n^{O(\log^2 n)}$ time. Even more recently, Gartland et al. [19] extended the results of [18,31] from P_t-free graphs to H-free graphs where every connected component of H is a path or a subdivided claw.

Grötschel, Lovász, and Schrijver [21] proved that Vertex Cover can be solved in polynomial time for the class of perfect graphs. The class of perfect graphs is a rich graph class, which includes well-known graph classes, such as bipartite graphs and (weakly) chordal graphs.

Before we present our results, we first briefly discuss the relevant literature.

1.1. Existing results

Whenever Vertex Cover is NP-complete for some graph class $\mathcal{G}$, then so is the more general problem Subset Vertex Cover. Moreover, Subset Vertex Cover can be polynomially reduced to Vertex Cover: given an instance (G,T,k) of the former problem, remove all edges not incident to a vertex of T to obtain an instance (G',k) of the latter problem. Hence, we obtain:

Proposition 1. *The problems Vertex Cover and Subset Vertex Cover are polynomially equivalent for every graph class closed under edge deletion.*

For example, the class of bipartite graphs is closed under edge deletion and Vertex Cover is polynomial-time solvable on bipartite graphs. Hence, by Proposition 1, Subset Vertex Cover is polynomial-time solvable on bipartite graphs. However, a class of H-free graphs is only closed under edge deletion if H is a complete graph, and Vertex Cover is NP-complete even for triangle-free graphs [32]. This means that there could still exist graphs H such that Vertex Cover and Subset Vertex Cover behave differently if the former problem is (quasi)polynomial-time solvable on H-free graphs. The following well-known result of Alekseev [1] restricts the structure of such graphs H.

Theorem 2 *([1]). For every graph H that contains a cycle or a connected component with two vertices of degree at least 3, Vertex Cover, and thus Subset Vertex Cover, is NP-complete for H-free graphs.*

Due to Theorem 2 and the result of Gartland et al. [19], every graph H is now either classified as a quasi-polynomial case or NP-hard case for Vertex Cover. For Subset Vertex Cover the situation is much less clear. So far, only one positive result for H-free graphs is known, due to Brettell et al. [8].

Theorem 3 *([8]). For every $s \geq 0$, Subset Vertex Cover is polynomial-time solvable on $(sP_1 + P_4)$-free graphs.*

We notice one more result for Subset Vertex Cover, which can be obtained by making a connection to the concept of *probe graphs*, introduced by Zhang et al. [35] in the context of genome research. Suppose $\mathcal{G}$ is a class of graphs. The class of probe graphs $\mathcal{G}_p$ of $\mathcal{G}$ contains all graphs G such that there exist an independent set N in G and a set of edges $F \subseteq \binom{N}{2}$ such that $G + F \in \mathcal{G}$. The intuition is that the vertices of $P = V(G) \setminus N$ are the probes, for which there is structural information, and the vertices of N are the non-probes, for which there is no concrete information except that the "actual graph" $G + F$ belongs to some graph class, achieved by adding some edges between vertices of N.

We recall that an instance (G, T, k) of Subset Vertex Cover is reducible to an instance (G', k) of Vertex Cover, where G' is the graph obtained from G by deleting all edges between vertices not in T. In other words, we can solve Subset Vertex Cover in polynomial time on a graph class $\mathcal{G}$ if we can solve Vertex Cover in polynomial time on $\mathcal{G}_p$. Chang et al. [12] observed that probe split graphs are perfect, and Golumbic and Lipshteyn [20] showed that probe interval graphs are probe chordal graphs, and that probe chordal graphs are perfect. As Vertex Cover is polynomial-time solvable on perfect graphs [21], it follows that Subset Vertex Cover can be solved in polynomial time on chordal graphs.

Theorem 4 *([20,21]). Subset Vertex Cover can be solved in polynomial time for chordal graphs.*

1.2. Related work

Subset variants of classic graph transversal problems are widely studied, also in the context of H-free graphs. Indeed, Brettell et al. [8] needed Theorem 3 as an auxiliary result in complexity studies for Subset Feedback Vertex Set and Subset Odd Cycle Transversal restricted to H-free graphs. The first problem is to decide for a graph $G = (V, E)$, subset $T \subseteq V$ and integer k, if G has a set S of size at most k such that S contains a vertex of every cycle that intersects T. The second problem is similar but replaces "cycle" by "cycle of odd length". Brettell et al. [8] proved that both these subset transversal problems are polynomial-time solvable on $(sP_1 + P_3)$-free graphs for every $s \geq 0$. They also showed that Odd Cycle Transversal is polynomial-time solvable for P_4-free graphs and NP-complete for split graphs, which form a subclass of $2P_2$-free graphs, whereas NP-completeness for Subset Feedback Vertex Set on split graphs was shown by Fomin et al. [17]. Recently, Paesani et al. [29] extended the result of [8] for Subset Feedback Vertex Set from $(sP_1 + P_3)$-free graphs to $(sP_1 + P_4)$-free graphs for every integer $s \geq 0$. If H contains a cycle or claw, NP-completeness for both subset transversal problems follows from corresponding results for Feedback Vertex Set [28,32] and Odd Cycle Transversal [13].

Combining all the above results leads to a dichotomy for Subset Feedback Vertex Set and a partial classification for Subset Odd Cycle Transversal (see also [8,29]). Here, we write $F \subseteq_i G$ if F is an induced subgraph of G.

Theorem 5. *For a graph H, Subset Feedback Vertex Set on H-free graphs is polynomial-time solvable if $H \subseteq_i sP_1 + P_4$ for some $s \geq 0$, and NP-complete otherwise.*

Theorem 6. *For a graph $H \neq sP_1 + P_4$ for some $s \geq 1$, Subset Odd Cycle Transversal on H-free graphs is polynomial-time solvable if $H = P_4$ or $H \subseteq_i sP_1 + P_3$ for some $s \geq 0$, and NP-complete otherwise.*

We note that neither Subset Feedback Vertex Set nor Subset Odd Cycle Transversal restricted to some graph class $\mathcal{G}$ can be reduced to their classical counterparts on the probe graph class $\mathcal{G}_p$ by removing edges between any pair of vertices that both do not belong to T. So, in this sense these two subset transversal problems are different in nature than Subset Vertex Cover.

1.3. Our results

In Section 3 we prove two new hardness results, using the same basis reduction, which may have a wider applicability. We first answer Q1 by proving that Subset Vertex Cover is NP-complete even for subcubic planar line graphs of triangle-free graphs, or equivalently, subcubic planar (claw, diamond)-free graphs [26]. We then answer Q2 by proving that Subset Vertex Cover is NP-complete even for 2-unipolar graphs, which are $2P_3$-free (and thus P_7-free).

Our hardness results show a sharp contrast with Vertex Cover, which can be solved in polynomial time for both weakly chordal graphs [21] and $rK_{1,3}$-free graphs for every $r \geq 1$ [6]. Hence, Subset Vertex Cover may be harder than Vertex Cover for a graph class closed under vertex deletion (if P $\neq$ NP). This is in contrast to graph classes closed under edge deletion (see Proposition 1).

In Section 3 we also prove that Subset Vertex Cover is NP-complete for inputs (G, T, k) if the subgraph $G[T]$ of G induced by T is P_3-free. On the other hand, our first positive result, shown in Section 4, shows that the problem is polynomial-time solvable if $G[T]$ is sP_2-free for any $s \geq 2$. In Section 4 we also prove that Subset Vertex Cover can be solved in polynomial time for $(sP_1 + P_2 + P_3)$-free graphs for every $s \geq 1$. Our positive results generalize known results for Vertex Cover. Recall that Subset Vertex Cover is polynomial-time solvable for split graphs. Note that this also follows from our first result and contrasts our NP-completeness result for 2-unipolar graphs, which are generalized split, $2P_3$-free, and weakly chordal.

Combining our new results with Theorem 3 gives us a partial classification and a dichotomy, both of which are proven in Section 5.

Theorem 7. *For a graph $H \neq rP_1 + sP_2 + P_3$ for any $r \geq 0$, $s \geq 2$; $rP_1 + sP_2 + P_4$ for any $r \geq 0$, $s \geq 1$; or $rP_1 + sP_2 + P_t$ for any $r \geq 0$, $s \geq 0$, $t \in \{5,6\}$, Subset Vertex Cover on H-free graphs is polynomial-time solvable if $H \subseteq_i sP_1 + P_2 + P_3$, sP_2, or $sP_1 + P_4$ for some $s \geq 1$, and NP-complete otherwise.*

Theorem 8. *For a graph H, Subset Vertex Cover on instances (G,T,k), where $G[T]$ is H-free, is polynomial-time solvable if $H \subseteq_i sP_2$ for some $s \geq 1$, and NP-complete otherwise.*

Theorems 5–8 show that Subset Vertex Cover on H-free graphs can be solved in polynomial time for infinitely more graphs H than Subset Feedback Vertex Set and Subset Odd Cycle Transversal. This is in line with the behaviour of the corresponding original (non-subset) problems.

In Section 6 we discuss our final new result, which states that Subset Vertex Cover is polynomial-time solvable on every graph class of bounded mim-width, such as the class of circular-arc graphs. In Section 7 we discuss some directions for future work, which naturally originate from the above results.

2. Preliminaries

Let $G = (V,E)$ be a graph. The *degree* of a vertex $u \in V$ is the size of its *neighbourhood* $N(u) = \{v \mid uv \in E\}$. We say that G is *subcubic* if every vertex of G has degree at most 3. An independent set I in G is *maximal* if there exists no independent set I' in G with $I \subsetneq I'$. Similarly, a vertex cover S of G is *minimal* if there exists no vertex cover S' in G with $S' \subsetneq S$. For a graph H we write $H \subseteq_i G$ if H is an *induced* subgraph of G, that is, G can be modified into H by a sequence of vertex deletions. If G does not contain H as an induced subgraph, G is *H-free*. For a set of graphs $\mathcal{H}$, G is $\mathcal{H}$-free if G is H-free for every $H \in \mathcal{H}$. If $\mathcal{H} = \{H_1, \ldots, H_p\}$ for some $p \geq 1$, we also write that G is *$(H_1, \ldots, H_p)$-free*.

The *line graph* of a graph $G = (V,E)$ is the graph $L(G)$ that has vertex set E and an edge between two vertices e and f if and only if e and f share a common end-vertex in G. The *complement* $\overline{G}$ of a graph $G = (V,E)$ has vertex set V and an edge between two vertices u and v if and only if $uv \notin E$.

For two vertex-disjoint graphs F and G, the *disjoint union* $F + G$ is the graph $(V(F) \cup V(G), E(F) \cup E(G))$. We denote the disjoint union of s copies of the same graph G by sG. A *linear forest* is a disjoint union of one or more paths.

Let C_s be the cycle on s vertices; P_t the path on t vertices; K_r the complete graph on r vertices; and $K_{1,r}$ the star on $(r+1)$ vertices. The graph $C_3 = K_3$ is the *triangle*; the graph $K_{1,3}$ the *claw*, and the graph $\overline{2P_1 + P_2}$ is the *diamond* (so the diamond is obtained from the K_4 after deleting one edge). The *subdivision* of an edge uv replaces uv with a new vertex w and edges uw, wv. A *subdivided claw* is obtained from the claw by subdividing each of its edges zero or more times.

A graph is *chordal* if it has no induced C_s for any $s \geq 4$. A graph is *weakly chordal* if it has no induced C_s and no induced $\overline{C_s}$ for any $s \geq 5$. A cycle C_s or an anti-cycle $\overline{C_s}$ is *odd* if it has an odd number of vertices. By the Strong Perfect Graph Theorem [16], a graph is *perfect* if it has no odd induced C_s and no odd induced $\overline{C_s}$ for any $s \geq 5$. Every chordal graph is weakly chordal, and every weakly chordal graph is perfect. A graph $G = (V,E)$ is *unipolar* if V can be partitioned into two sets V_1 and V_2, where $G[V_1]$ is a complete graph and $G[V_2]$ is a disjoint union of complete graphs. If every connected component of $G[V_2]$ has size at most 2, then G is 2-*unipolar*. Unipolar graphs form a subclass of *generalized split graphs*, which are the graphs that are unipolar or their complement is unipolar. It can also readily be checked that every 2-unipolar graph is weakly chordal (but not necessarily chordal, as evidenced by $G = C_4$).

For an integer r, a graph G' is an *r-subdivision* of a graph G if G' can be obtained from G by subdividing every edge of G r times, that is, by replacing each edge $uv \in E(G)$ with a path from u to v of length $r+1$.

3. NP-hardness results

In this section we prove our hardness results for Subset Vertex Cover, using the following notation. Let G be a graph with an independent set I. We say that we *augment* G by adding a (possibly empty) set F of edges between some pairs of vertices of I. We call the resulting graph an *I-augmentation* of G.

The following lemma forms the basis for our hardness gadgets.

Lemma 9. *Every vertex cover of a graph $G = (V,E)$ with an independent set I is a $(V \setminus I)$-vertex cover of every I-augmentation of G, and vice versa.*

Proof. Let G' be an I-augmentation of G. Consider a vertex cover S of G. For a contradiction, assume that S is not a $(V \setminus I)$-vertex cover of G'. Then $G' - S$ must contain an edge uv with at least one of u,v belonging to $V \setminus I$. As $G - S$ is an independent set, uv belongs to $E(G') \setminus E(G)$ implying that both u and v belong to I, a contradiction.

Now consider a $(V \setminus I)$-vertex cover S' of G'. For a contradiction, assume that S' is not a vertex cover of G. Then $G - S'$ must contain an edge uv (so $uv \in E$). As G' is a supergraph of G, we find that $G' - S'$ also contains the edge uv. As S' is a $(V \setminus I)$-vertex cover of G', both u and v must belong to I. As $uv \in E$, this contradicts the fact that I is an independent set. □

To use Lemma 9 we need one other lemma, which follows directly from an observation due to Poljak [32].

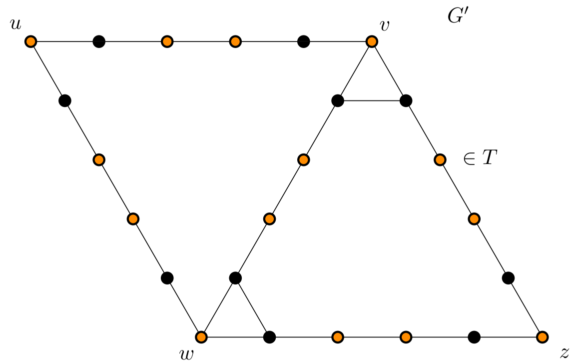

Fig. 2. The graph G' from Theorem 11, where $T = V \setminus W$ consists of the orange vertices.

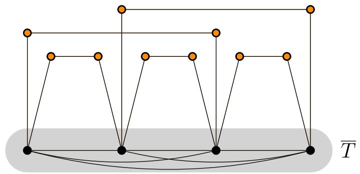

Fig. 3. The graph G' from Theorem 12, where the orange vertices form $T = V \setminus V(G^*)$.

Lemma 10 *([32]). For an integer r, a graph G with m edges has an independent set of size k if and only if the $2r$-subdivision of G has an independent set of size $k + rm$.*

We are now ready to prove our first two hardness results. Recall that a graph is (claw, diamond)-free if and only if it is a line graph of a triangle-free graph. Hence, the result in particular implies NP-hardness of Subset Vertex Cover for line graphs. Recall also that we denote the claw and diamond by $K_{1,3}$ and $\overline{2P_1 + P_2}$, respectively.

Theorem 11. *Subset Vertex Cover is NP-complete for $(K_{1,3}, \overline{2P_1 + P_2})$-free subcubic planar graphs.*

Proof. We reduce from Vertex Cover, which is NP-complete even for cubic planar graphs [27]. As an n-vertex graph has a vertex cover of size at most k if and only if it has an independent set of size at least $n - k$, we find that Vertex Cover is NP-complete even for subcubic planar graphs that are 4-subdivisions due to an application of Lemma 10 with $r = 2$ (note that subdividing an edge preserves both maximum degree and planarity). So, let (G, k) be an instance of Vertex Cover, where $G = (V, E)$ is a subcubic planar graph that is a 4-subdivision of some cubic planar graph G^*, and k is an integer.

In G, we let $U = V(G^*)$ and W be the subset of $V(G) \setminus U$ that consists of all neighbours of vertices of U. Note that W is an independent set in G. We construct a W-augmentation G' as follows; see also Fig. 2. For every vertex $u \in U$ of degree 3 in G, we pick two arbitrary neighbours of u (which both belong to W) and add an edge between them. It is readily seen that G' is $(K_{1,3}, \overline{2P_1 + P_2})$-free, planar and subcubic. By Lemma 9, it holds that G has a vertex cover of size at most k if and only if G' has a $(V \setminus W)$-vertex cover of size at most k. □

Theorem 12. *Subset Vertex Cover is NP-complete for instances (G, T, k), for which G is 2-unipolar and $G[T]$ is a disjoint union of edges.*

Proof. We reduce from Vertex Cover. So, let (G, k) be an instance of Vertex Cover, where $G = (V, E)$ is a graph and k is an integer. By Lemma 10 (take $r = 1$) we may assume that G is a 2-subdivision of a graph G^*. Note that $V(G^*)$ is an independent set in G. We construct a $V(G^*)$-augmentation G' of G by changing $V(G^*)$ into a clique; see also Fig. 3. It is readily seen that G' is 2-unipolar. We set $T := V \setminus V(G^*)$, so $G[T]$ is a disjoint union of edges. By Lemma 9, it holds that G has a vertex cover of size at most k if and only if G' has a T-vertex cover of size at most k. □

Remark 13. It can be readily checked that 2-unipolar graphs are $(2C_3, C_5, C_6, C_3 + P_3, 2P_3, \overline{P_6}, \overline{C_6})$-free graphs, and thus are $2P_3$-free and weakly chordal.

4. Polynomial-time results

In this section, we prove our polynomial-time results for instances (G, T, k) where either G is H-free or only $G[T]$ is H-free. The latter type of results are stronger, but only hold for graphs H with smaller connected components.

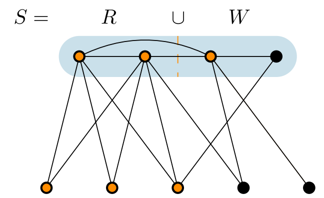

Fig. 4. An example of the $2P_2$-free graph G' of the proof of Theorem 17. Here, T consists of the orange vertices. A solution S can be split up into a minimal vertex cover R of $G'[T]$ and a vertex cover W of $G[V \setminus R]$.

For completeness, we state Kőnig's theorem here, which has as an immediate consequence that Vertex Cover can be solved in polynomial time on bipartite graphs, which is constructively shown by Bondy and Murty [5].

Theorem 14 *(Kőnig's Theorem [23]). In a bipartite graph, the number of edges in a maximum matching is equal to the minimum number of vertices in a vertex cover.*

We start with the case where $H = sP_2$ for some $s \geq 1$. For this case we need the following two well-known results. The *delay* of an enumeration algorithm is the maximum of the time taken before the first output and that between any pair of consecutive outputs.

Theorem 15 *([2]). For every constant $s \geq 1$, the number of maximal independent sets of an sP_2-free graph on n vertices is at most $n^{2s} + 1$.*

Theorem 16 *([33]). For every constant $s \geq 1$, it is possible to enumerate all maximal independent sets of an sP_2-free graph G on n vertices and m edges with a delay of $O(nm)$.*

We now prove that Subset Vertex Cover is polynomial-time solvable for instances (G, T, k), where $G[T]$ is sP_2-free. The idea behind the algorithm is to remove any edges between vertices in $V \setminus T$, as these edges are irrelevant. As a consequence, we may leave the graph class, but this is not necessarily an obstacle. For example, if $G[T]$ is a complete graph, or T is an independent set, we can easily solve the problem. Both cases are generalized by the result below.

Theorem 17. *For every $s \geq 1$, Subset Vertex Cover can be solved in polynomial time for instances (G, T, k) for which $G[T]$ is sP_2-free.*

Proof. Let $s \geq 1$, and let (G, T, k) be an instance of Subset Vertex Cover where $G = (V, E)$ is a graph such that $G[T]$ is sP_2-free. Let $G' = (V, E')$ be the graph obtained from G after removing every edge between two vertices of $V \setminus T$, so $G'[V \setminus T]$ is edgeless. We observe that G has a T-vertex cover of size at most k if and only if G' has a T-vertex cover of size at most k. Moreover, $G'[T]$ is sP_2-free, and we can obtain G' in $O(|E(G)|)$ time. Hence, from now on, we consider the instance (G', T, k).

We first prove the following two claims, see Fig. 4 for an illustration.

Claim 17.1. *A subset $S \subseteq V(G')$ is a T-vertex cover of G' if and only if $S = R \cup W$ for a minimal vertex cover R of $G'[T]$ and a vertex cover W of $G'[V \setminus R]$.*

Proof. We prove Claim 17.1 as follows. Let $S \subseteq V(G')$. First assume that S is a T-vertex cover of G'. Let $I = V \setminus S$. As S is a T-vertex cover, $T \cap I$ is an independent set. Hence, S contains a minimal vertex cover R of $G'[T]$. As $G'[V \setminus T]$ is edgeless, S is a vertex cover of G, or in other words, I is an independent set. In particular, this means that $S \setminus R$ is a vertex cover of $G'[V \setminus R]$.

Now assume that $S = R \cup W$ for a minimal vertex cover R of $G'[T]$ and a vertex cover W of $G'[V \setminus R]$. For a contradiction, suppose that S is not a T-vertex cover of G'. Then $G' - S$ contains an edge $uv \in E'$, where at least one of u, v belongs to T. First suppose that both u and v belong to T. As R is a vertex cover of $G'[T]$, at least one of u, v belongs to $R \subseteq S$, a contradiction. Hence, exactly one of u, v belongs to T, say $u \in T$ and $v \in V \setminus T$, so in particular, $v \notin R$. As $R \subseteq S$, we find that $u \notin R$. Hence, both u and v belong to $V \setminus R$. As W is a vertex cover of $G'[V \setminus R]$, this means that at least one of u, v belongs to $W \subseteq S$, a contradiction. This proves the claim. ⋄

Claim 17.2. *For every minimal vertex cover R of $G'[T]$, the graph $G'[V \setminus R]$ is bipartite.*

Proof. We prove Claim 17.2 as follows. As R is a vertex cover of $G'[T]$, we find that $T \setminus R$ is an independent set. As $G'[V \setminus T]$ is edgeless by construction of G', this means that $G'[V \setminus R]$ is bipartite with partition classes $T \setminus R$ and $V \setminus T$. ⋄

We are now ready to give our algorithm. We enumerate the minimal vertex covers of $G'[T]$. For every minimal vertex cover R, we compute a minimum vertex cover W of $G'[V \setminus R]$. In the end, we return the smallest $S = R \cup W$ that we found.

The correctness of our algorithm follows from Claim 17.1. It remains to analyze the running time. As $G'[T]$ is sP_2-free, we can enumerate all maximal independent sets I of $G'[T]$ and thus all minimal vertex covers $R = T \setminus I$ of $G'[T]$ in $(n^{2s} + 1) \cdot O(nm)$ time due to Theorems 15 and 16. For a minimal vertex cover R, the graph $G'[V \setminus R]$ is bipartite by Claim 17.2. Hence, we can compute a minimum vertex cover W of $G'[V \setminus R]$ in polynomial time by applying Kőnig's Theorem (Theorem 14). We conclude that the total running time is polynomial. □

For our next result (Theorem 20) we need two known results as lemmas.

Lemma 18 *([8]). If* Subset Vertex Cover *is polynomial-time solvable on H-free graphs for some H, then it is so on $(H + P_1)$-free graphs.*

Lemma 19 *([6]). For every $r \geq 1$,* Vertex Cover *is polynomial-time solvable on $rK_{1,3}$-free graphs.*

We are now ready to prove our second polynomial-time result.

Theorem 20. *For every integer s,* Subset Vertex Cover *is polynomial-time solvable on $(sP_1 + P_2 + P_3)$-free graphs.*

Proof. Due to Lemma 18, we can take $s = 0$, so we only need to give a polynomial-time algorithm for $(P_2 + P_3)$-free graphs. Hence, let (G, T, k) be an instance of Subset Vertex Cover, where $G = (V, E)$ is a $(P_2 + P_3)$-free graph.

First, we compute a minimum vertex cover S_{vc} of G. As G is $(P_2 + P_3)$-free, and thus $2K_{1,3}$-free, this takes polynomial time by Lemma 19.

We now set out to compute a minimum T-vertex cover S of G that is not a vertex cover of G. Then $G - S$ must contain an edge between two vertices in $V \setminus T$. We branch by considering all $O(n^2)$ options of choosing this edge. For each chosen edge uv we set out to construct a smallest T-vertex cover S_{uv} of G that does not contain u nor v. Observe that we thus must add every neighbour of u or v that belongs to T to S_{uv}, as otherwise $G - S_{uv}$ would contain an edge ut or vt for some vertex $t \in T$. Hence, we have $N(\{u, v\}) \cap T \subseteq S_{uv}$.

Let $T' = T \setminus N(\{u, v\})$ consist of all vertices of T that are neither adjacent to u nor to v. As G is $(P_2 + P_3)$-free and $uv \in E$, we find that $G[T']$ is P_3-free, and thus $G[T']$ is a disjoint union of complete graphs. We call a connected component of $G[T']$ *large* if it has at least two vertices; else we call it *small* (so every large connected component of $G[T']$ is a complete graph on at least two vertices and every small connected component of $G[T']$ is an isolated vertex).

Case 1. The graph $G[T']$ has at most two large connected components.
Let D_1 and D_2 be the large connected components of $G[T']$ (if they exist). As $V(D_1)$ and $V(D_2)$ are cliques in $G[T']$, at most one vertex of D_1 and at most one vertex of D_2 can belong to $G - S_{uv}$ if S_{uv} is to become a T-vertex cover. We branch by considering all $O(n^2)$ options of choosing at most one vertex of D_1 and at most one vertex of D_2 to be these vertices. For each choice of vertices we do as follows. We add all other vertices of D_1 and D_2 to a set S^*_{uv}. Let T^* be the set of vertices of T' that we have not added to S^*_{uv}. Then T^* consist of all the vertices of the small connected components of $G[T']$ and at most one vertex of each of the at most two large connected components of $G[T']$. Hence, T^* is an independent set.

We delete every edge between any two vertices in $V \setminus T$. Now the graph G^* induced by $T^* \cup (V \setminus T)$ is bipartite, namely with partition classes T^* and $V \setminus T$. It remains to compute a minimum vertex cover S^* of G^*. This can be done in polynomial time by applying Kőnig's Theorem. We let S^*_{uv} consist of S^* together with all vertices of T that we had added to S^*_{uv} already.

Finally, let S_{uv} be a smallest such set S^*_{uv} (found over all $O(n^2)$ branches) together with $N(\{u, v\}) \cap T$, so S_{uv} is a smallest T-vertex cover of G that does not contain u nor v. This completes Case 1.

Case 2. The graph $G[T']$ has at least three large connected components.
Let $D_1, \ldots, D_p$, for some $p \geq 3$, be the large connected components of $G[T']$. Let A consist of all the vertices of the small connected components of $G[T']$.

We start by considering each vertex $w \in V \setminus T$ with one of the following properties:

1. for some i, w has a neighbour and a non-neighbour in D_i; or
2. for some i, j with $i \neq j$, w has a neighbour in D_i and a neighbour in D_j; or
3. for some i, w has a neighbour in D_i and a neighbour in A.

Let W be the set of all vertices of $V \setminus T$ that satisfy at least one of the Properties 1–3 (note that W does not contain u nor v). We say that a vertex $w \in V \setminus T$ is *semi-complete* to some D_i if w is adjacent to all vertices of D_i except at most one. We show the following claim. See Fig. 5 for an illustration.

Claim 20.1. *Every vertex $w \in W$ is semi-complete to D_i for every $i \in \{1, \ldots, p\}$.*

Proof. Let $w \in W$. First, assume w satisfies Property 1. Let x and y be vertices of some D_i, say D_1, such that $wx \in E$ and $wy \notin E$. For a contradiction, assume w is not semi-complete to some D_j. Hence, D_j contains vertices y' and y'', such that $wy' \notin E$ and $wy'' \notin E$. If $j \geq 2$, then $\{y', y'', w, x, y\}$ induces a $P_2 + P_3$ (as D_1 and D_j are complete graphs). This contradicts that G is $(P_2 + P_3)$-free. Hence, w is semi-complete to every $V(D_j)$ with $j \geq 2$. Now suppose $j = 1$. As $p \geq 3$, the graphs D_2 and D_3 exist. As w is semi-complete to

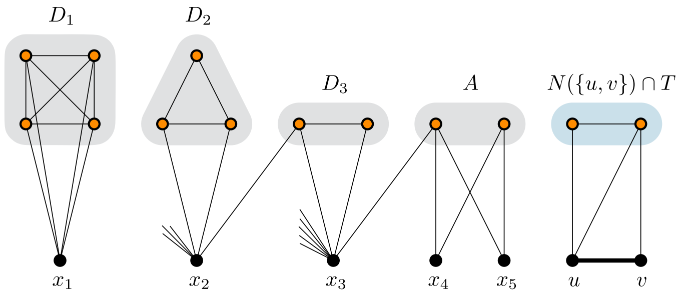

Fig. 5. An illustration of the graph G in the proof of Case 2 of Theorem 20, where T consists of the orange vertices, and $p = 3$. Edges in $G[V \setminus T]$ are not drawn, and for x_2 and x_3 some edges are partially drawn. Vertices x_1, x_4, x_5, u, v do not belong to W, as they do not satisfy one of the Properties 1–3, whereas x_2 belongs to W, as x_2 satisfies Property 1 for D_2 (and also Property 2 for D_2 and D_3), and x_3 belongs to W, as x_3 satisfies Property 3 for D_3.

every D_j for $j \geq 2$ and every D_j is large, there exist vertices $x' \in V(D_2)$ and $x'' \in V(D_3)$ such that $wx' \in E$ and $wx'' \in E$. However, now $\{y', y'', x', w, x''\}$ induces a $P_2 + P_3$, a contradiction.

Now assume w satisfies Property 2, say w is adjacent to $x_1 \in V(D_1)$ and to $x_2 \in V(D_2)$. Suppose w is not semi-complete to some D_j. If $j \geq 3$, then the two non-neighbours of w in D_j, together with x_1, w, x_2, form an induced $P_2 + P_3$, a contradiction. Hence, w is semi-complete to every D_j for $j \geq 3$. If $j \in \{1, 2\}$, say $j = 1$, then let y, y' be two non-neighbours of w in D_1 and let x_3 be a neighbour of w in D_3. Now, $\{y, y', x_2, w, x_3\}$ induces a $P_2 + P_3$, a contradiction. Hence, w is semi-complete to D_1 and D_2 as well.

Finally, assume w satisfies Property 3, say w is adjacent to $z \in A$ and $x_1 \in V(D_1)$. If w is not semi-complete to D_j for some $j \geq 2$, then two non-neighbours of w in D_j, with z, w, x_1, form an induced $P_2 + P_3$, a contradiction. Hence, w is semi-complete to every D_j with $j \geq 2$. As before, by using a neighbour of w in D_2 and one in D_3, we find that w is also semi-complete to D_1. This completes the proof of Claim 20.1. $\diamond$

We now first consider the possible situation where S_{uv} will not contain some vertex $w \in W$. We branch by considering all $O(n)$ options for choosing such a vertex w. For each chosen vertex w, we do as follows. Let $T^w = N(w) \cap T'$ be the set of neighbours of w in T' (recall that $T' = T \setminus N(\{u, v\})$). So, all of the vertices of T^w must belong to S_{uv}. By Claim 20.1, we find that $T' \setminus T^w$ does not contain two vertices from the same large connected component of $G[T']$. Hence, $T' \setminus T^w$ is an independent set. We delete any edge between two vertices from $V \setminus T$, so $V \setminus T$ becomes an independent set as well. We now compute, in polynomial time by Kőnig's Theorem, a minimum vertex cover in the resulting bipartite graph with partition classes $T' \setminus T^w$ and $V \setminus T$. We let S^w_{uv} be the set that is the union of this vertex cover, T^w, and $(N\{u, v\} \cap T)$. By construction, S^w_{uv} is a smallest vertex cover of G that does not contain any vertex of $\{u, v, w\}$. After processing all of the $O(n)$ branches, we keep a smallest set S^w_{uv}, which we denote by S^*_{uv}.

We are left to examine the possible situation where S_{uv} contains every vertex of W. Let G' be the subgraph obtained from G by removing every vertex of $W \cup (N(\{u, v\}) \cap T) \cup \{u, v\}$. We will now describe how to compute in polynomial time a smallest T'-vertex cover S'_{uv} of G' (recall that $T' = T \setminus N(\{u, v\})$).

We start with considering the connected component D'_1 of G' that contains (all) the vertices from D_1. As no vertex from W belongs to G' by definition, D'_1 contains no vertices from $V \setminus T$ satisfying Property 2 or 3. Hence, D'_1 contains no vertices from A or from any D_j with $j \geq 2$ either. So it holds that

$$V(D'_1) \cap T = V(D_1).$$

First suppose that $D'_1 = D_1$. As $D'_1 = D_1$ is a complete graph, we add all vertices of D_1 except for one arbitrary vertex of D_1 to S'_{uv}. Now suppose that there exists a vertex x in $V(D'_1) \setminus V(D_1)$. As D'_1 is connected, we may assume without loss of generality that x has a neighbour in D_1. Consequently, x is complete to D_1, as x does not belong to W and thus does not satisfy Property 1. This implies that we must put at least $|V(D_1)|$ vertices from D'_1 in S'_{uv}, so we might just as well put every vertex of D_1 in S'_{uv}. As $V(D'_1) \cap T = V(D_1)$, we do not need to add any more vertices from D'_1 to S'_{uv}.

We do the same as we did for D_1 for the connected components $D'_2, \ldots, D'_p$ of G' that contain the sets $V(D_2), \ldots, V(D_p)$, respectively.

Now, it remains to consider the induced subgraph F of G' that consists of the connected components of G' containing the vertices of A. Recall that A is an independent set. We delete every edge between two vertices in $V \setminus T$, resulting in another independent set. This changes F into a bipartite graph, and we can compute a minimum vertex cover S^F_{uv} of F in polynomial time due to Kőnig's Theorem. We add S^F_{uv} to S'_{uv}. In this way we computed the set S'_{uv}. We let $S''_{uv} = S'_{uv} \cup W \cup (N(\{u, v\}) \cap T)$. By construction, S''_{uv} is a smallest vertex cover S''_{uv} of G that contains W but does not contain u and v.

Now compare the size of S''_{uv} with the size of S^*_{uv}, and pick one of smallest size as S_{uv}. This completes Case 2.

Finally, for each choice of the edge $uv \in E(G[V \setminus T])$, we consider the output S_{uv}, and take a smallest set found. We compare its size with the size of S_{vc}, again taking a smallest set as the final solution.

The correctness of our algorithm follows from the above description. The number of branches is $O(n^2)$ in Case 1 and $O(n)$ in Case 2. Hence, as there $O(n^2)$ vertex pairs u, v to consider, the total number of branches is $O(n^4)$. As each branch takes polynomial time to process, this means that the total running time of our algorithm is polynomial. This completes our proof. $\square$

5. The proof of Theorems 7 and 8

We first prove Theorem 7, which we restate below.

Theorem 7 *(restated). For a graph $H \neq rP_1 + sP_2 + P_3$ for any $r \geq 0$, $s \geq 2$; $rP_1 + sP_2 + P_4$ for any $r \geq 0$, $s \geq 1$; or $rP_1 + sP_2 + P_t$ for any $r \geq 0$, $s \geq 0$, $t \in \{5,6\}$, Subset Vertex Cover on H-free graphs is polynomial-time solvable if $H \subseteq_i sP_1 + P_2 + P_3$, sP_2, or $sP_1 + P_4$ for some $s \geq 1$, and NP-complete otherwise.*

Proof. Let H be a graph not equal to $rP_1 + sP_2 + P_3$ for any $r \geq 0$, $s \geq 2$; $rP_1 + sP_2 + P_4$ for any $r \geq 0$, $s \geq 1$; or $rP_1 + sP_2 + P_t$ for any $r \geq 0$, $s \geq 0$, $t \in \{5,6\}$. If H has a cycle, then we apply Theorem 2. Else, H is a forest. If H has a vertex of degree at least 3, then the class of H-free graphs contains all $K_{1,3}$-free graphs, and we apply Theorem 11. Else, H is a linear forest. If H contains an induced $2P_3$, then we apply Theorem 12. If not, then $H \subseteq_i sP_1 + P_2 + P_3$, sP_2, or $sP_1 + P_4$ for some $s \geq 1$. In the first case, apply Theorem 20; in the second case Theorem 17; and in the third case Theorem 3. □

We now prove Theorem 8, which we restate below.

Theorem 8 *(restated). For a graph H, Subset Vertex Cover on instances (G, T, k), where $G[T]$ is H-free, is polynomial-time solvable if $H \subseteq_i sP_2$ for some $s \geq 1$, and NP-complete otherwise.*

Proof. First note that any NP-completeness result of Theorem 7 also holds here, as $G[T]$ is H-free if the entire graph is H-free. We immediately get that Subset Vertex Cover is NP-complete if H is not a linear forest. Suppose $P_3 \subseteq_i H$. Observe that in the instance of Theorem 12, $G[T]$ is a disjoint union of edges, so $G[T]$ is P_3-free, and hence, H-free. We see that Subset Vertex Cover is NP-complete when $P_3 \subseteq_i H$. In the remaining case, $H \subseteq_i sP_2$ for some $s \geq 1$, and we apply Theorem 17. □

6. Graphs of bounded clique-width and bounded mim-width

In this section, we give a polynomial algorithm for Subset Vertex Cover on graphs of bounded clique-width and of bounded mim-width.

We begin with a result regarding the clique-width of probe graphs. Recall that $\mathcal{G}_p$ denotes the class of probe graphs of some graph class $\mathcal{G}$ and that a graph G belongs to $\mathcal{G}_p$ if there is some independent set N of G and a set of edges F with end-vertices in N such that $G + F \in \mathcal{G}$. The clique-width $\mathsf{cw}(G)$ of a graph G, introduced by Courcelle, Engelfriet and Rozenberg [14], is the minimum number of labels needed to construct G by means of the following four operations:

1. creating a new labelled vertex;
2. taking the disjoint union of two labelled graphs;
3. adding an edge between every vertex of label i and every vertex of label j, where $i \neq j$; and
4. changing the label of all vertices with label i to label j for some $j \geq i$.

Cographs have clique-width at most 2 [15]. The authors of [11,12] observed that probe cographs have clique-width at most 4. Their argument holds more generally, as we prove below for the sake of completeness. We note that Proposition 21 directly implies Theorem 3 for $s = 0$.

Proposition 21. *Let $G = (V, E)$ be a graph, let N be an independent set of G, and let F be a set of edges with both end-vertices in N. Then we have $\mathsf{cw}(G) \leq 2\mathsf{cw}(G + F)$.*

Proof. We adapt a sequence of operations that constructs $G + F$ using only $\mathsf{cw}(G + F)$ labels to a sequence of operations that constructs G using $2\mathsf{cw}(G + F)$ labels. If an operation is to create a new vertex v with label i, then create v and label it with i_N if $v \in N$ and with i_P otherwise. If an operation is the disjoint union of two labelled graphs, then we keep this operation. If an operation is adding an edge between every vertex of label i and every vertex of label j, where $i \neq j$, then we add an edge between every vertex of label i_P and every vertex of label j_P, every vertex of label i_N and every vertex of label j_P, and every vertex of label i_P and every vertex of label j_N. If an operation is renaming a label i to j, then we rename i_N to j_N and i_P to j_P. Note that these operations construct G because they include all edges from $G + F$ except the edges with both end-vertices in N. Clearly, we use $2\mathsf{cw}(G + F)$ labels, which proves the statement. □

We now introduce some new terminology to explain the notion of mim-width, which was first defined by Vatshelle [34]. Let $G = (V, E)$ be a graph. For $X \subseteq V$, we use 2^X to denote its power set and $\overline{X}$ to denote $V \setminus X$. A set $M \subseteq E$ is a *matching* in G if no two edges in M share an end-vertex. A matching M is an *induced matching* if no end-vertex of any edge $e \in M$ is adjacent to any end-vertex of an edge in $M \setminus \{e\}$. A *rooted binary tree* is a rooted tree of which each node has degree 1 or 3, except for a distinguished node that has degree 2 and is the root of the tree. A *rooted layout* $\mathcal{L} = (L, \delta)$ of G consists of a rooted binary tree L and a bijection δ between V and the leaves of L. For each node $x \in V(L)$, we let L_x be the set of leaves that are a descendant of x (including x if x is

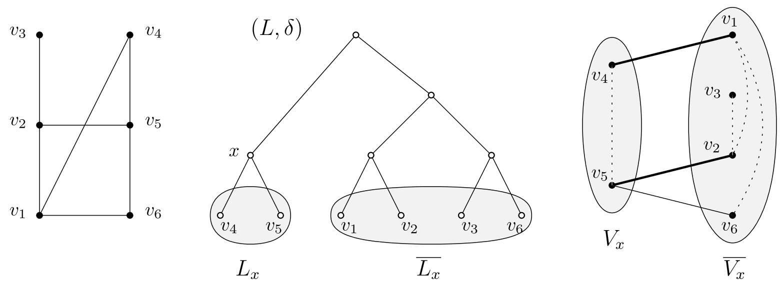

Fig. 6. An example of a graph G (left) with a rooted layout $\mathcal{L} = (L, \delta)$ (middle), taken from [7]. Note that $\mathsf{mim}(V_x) = 2$ (right), and an easy check shows that in fact $\mathsf{mim}(\mathcal{L}) = 2$. The rooted layout $\mathcal{L}'$ obtained from $\mathcal{L}$ by swapping v_2 and v_5 and swapping v_3 and v_4 yields $\mathsf{mim}(G) = 1$.

a leaf). We define V_x as the corresponding set of vertices of G, that is, $V_x = \{\delta(y) \mid y \in L_x\}$. For a set $A \subseteq V$, let $\mathsf{mim}_G(A)$ be the size of a maximum induced matching in the bipartite graph obtained from G by removing all edges between vertices of A and all edges between vertices of $\overline{A} = V \setminus A$. In other words, this is the bipartite graph $(A, \overline{A}, E \cap (A \times \overline{A}))$. The *mim-width* $\mathsf{mim}_G(\mathcal{L})$ of a rooted layout $\mathcal{L} = (L, \delta)$ of graph G is the maximum over all $x \in V(L)$ of $\mathsf{mim}_G(V_x)$. If the graph in question is clear from the context, then we omit the subscript. The *mim-width* $\mathsf{mim}(G)$ of G is the minimum mim-width over all rooted layouts of G. See Fig. 6 for an example.

In general, it is not known if there exists a polynomial-time algorithm for computing a rooted layout $\mathcal{L}$ of a graph G, such that $\mathsf{mim}(\mathcal{L})$ is bounded by a function in the mim-width of G. However, Belmonte and Vatshelle [3] showed that for several graph classes $\mathcal{G}$ of bounded mim-width, including interval graphs and permutation graphs, it is possible to find in polynomial time a rooted layout of a graph $G \in \mathcal{G}$ with mim-width equal to the mim-width of G.

We now prove the following result on the mim-width of probe graphs that is analogous to Proposition 21 for clique-width.

Proposition 22. *Let $G = (V, E)$ be a graph, let N be an independent set of G, let F be a set of edges with both ends in N, and let $\mathcal{L} = (L, \delta)$ be a rooted layout of $G + F$. Then we have $\mathsf{mim}_G(\mathcal{L}) \le 2\mathsf{mim}_{G+F}(\mathcal{L})$ and, in particular, $\mathsf{mim}(G) \le 2\mathsf{mim}(G + F)$.*

Proof. As G and $G + F$ have identical vertex sets, $\mathcal{L}$ is a rooted layout of G too. Let $x \in V(L)$ be such that $\mathsf{mim}_G(V_x) = \mathsf{mim}_G(\mathcal{L})$, and let H denote the bipartite graph $(V_x, \overline{V_x}, E \cap (V_x \times \overline{V_x}))$. In H, let E_1 denote the edges with one end-vertex in $V_x \cap N$, let E_2 denote the edges with one end-vertex in $\overline{V_x} \cap N$, and let E_3 denote the edges with no end-vertex in N. Note that the sets E_1, E_2 and E_3 are a partition of $E(H)$ since N is independent in G. Lastly, let M be a maximum induced matching of H. In particular, $|M| = \mathsf{mim}_G(V_x)$.

By symmetry, we may assume that $|E_1 \cap M| \le |E_2 \cap M|$. The set $M' = M \cap (E_2 \cup E_3)$ is an induced matching of $H + F$ too. On the one hand, we have $|M'| \ge |M|/2$ by the assumption that $|E_2 \cap M| \ge |E_1 \cap M|$. On the other hand, we have $|M'| \le \mathsf{mim}_{G+F}(V_x)$ by definition. Together, this implies

$$\mathsf{mim}_G(\mathcal{L}) = \mathsf{mim}_G(V_x) = |M| \le 2|M'| \le 2\mathsf{mim}_{G+F}(V_x) \le 2\mathsf{mim}_{G+F}(\mathcal{L}).$$

This argument holds for any rooted layout $\mathcal{L}$, so in particular, it holds for the layout $\mathcal{L}$ such that $\mathsf{mim}_{G+F}(\mathcal{L}) = \mathsf{mim}(G + F)$. For any layout $\mathcal{L}$, it holds that $\mathsf{mim}(G) \le \mathsf{mim}_G(\mathcal{L})$, and so we may conclude that $\mathsf{mim}(G) \le 2\mathsf{mim}(G + F)$ holds. This completes the proof. $\square$

Let $\mathcal{G}$ be a class of graphs of bounded mim-width. Then, Proposition 22 implies that $\mathcal{G}_p$ also has bounded mim-width. Therefore, an algorithm for Subset Vertex Cover on classes of bounded mim-width graphs is implied by results in the literature. Suppose that (G, T, k) is an instance of Subset Vertex Cover is given together with a rooted layout $\mathcal{L} = (L, \delta)$ of G. Let G' be the graph obtained from G by deleting all edges not incident to a vertex of T. Then $\mathcal{L}$ is also a rooted layout of G' and $\mathsf{mim}_{G'}(\mathcal{L}) \le 2\mathsf{mim}_G(\mathcal{L})$ by Proposition 22. Now, solving the Subset Vertex Cover instance (G, T, k) amounts to solving the Vertex Cover instance (G', k). The dynamic programming algorithm for Vertex Cover of Bui-Xuan et al. [10] has running time $O(n^4 \cdot \mathsf{nec}_1(T, \delta)^3)$ ($\mathsf{nec}_d(.)$ is defined below). Belmonte and Vatshelle [3] showed that $\mathsf{nec}_d(A) \le |A|^{d \cdot \mathsf{mim}_G(A)}$ for any $A \subseteq V(G)$. Together, this implies an $O(n^{6\mathsf{mim}_G(\mathcal{L})+4})$-time algorithm for Subset Vertex Cover with a given rooted layout $\mathcal{L}$ of G. However, this is a weaker result than that of Theorem 29, which is attained through a direct algorithm that we give next.

We now introduce the notion of neighbour equivalence, which was first defined by Bui-Xuan et al. [10]. Let $G = (V, E)$ be a graph on n vertices. Let $A \subseteq V$ and $d \in \mathbb{N}^+$. We say that $X, W \subseteq A$ are *d-neighbour equivalent* with respect to A, denoted $X \equiv_d^A W$, if $\min\{d, |X \cap N(v)|\} = \min\{d, |W \cap N(v)|\}$ for all $v \in \overline{A}$. Clearly, this is an equivalence relation. We let $\mathsf{nec}_d(A)$ denote the number of equivalence classes of $\equiv_d^A$. For a rooted layout (L, δ) of G, we let $\mathsf{nec}_d(T, \delta)$ denote the maximum of $\mathsf{nec}_d(V_x)$ over all $x \in V(L)$. For each $X \subseteq A$, let $\mathsf{rep}_d^A(X)$ denote the lexicographically smallest set $R \subseteq A$ such that $R \equiv_d^A X$ and $|R|$ is minimum. This is called the *representative* of X. We use $\mathcal{R}_d^A = \{\mathsf{rep}_d^A(X) \mid X \subseteq A\}$. Note that $|\mathcal{R}_d^A| \ge 1$, as the empty set is always a representative. The following lemma allows us to work efficiently with representatives.

Lemma 23 *(Bui-Xuan et al. [10]). It is possible to compute in time $O(\mathsf{nec}_d(A) \cdot n^2 \log(\mathsf{nec}_d(A)))$, the set $\mathcal{R}_d^A$ and a data structure that given a set $X \subseteq A$, computes $\mathsf{rep}_d^A(X)$ in $O(|A| \cdot n \log(\mathsf{nec}_d(A)))$ time.*

We are now ready to give an explicit polynomial-time algorithm for SUBSET VERTEX COVER on graphs of bounded mim-width. Our algorithm is inspired by the algorithm of Bui-Xuan et al. [10] for INDEPENDENT SET and of Bergougnoux et al. [4] for SUBSET FEEDBACK VERTEX SET. Our presentation of the algorithm follows the presentation form in Bergougnoux et al. [4]. In fact, we solve the complementary problem. Given a graph $G=(V,E)$ with a rooted layout $\mathcal{L}=(L,\delta)$, a set $T\subseteq V$, and a weight function ω on its vertices, we find a maximum-weight T-independent set on G. Our goal is to use a standard dynamic programming algorithm. However, the size of the table that we would need to maintain by a naive approach is too large. Instead, we work with representatives of the sets in our table. We show that we can reduce the table size so that it is bounded by the square of the number of 1-neighbour equivalence classes.

First, we define a notion of equivalence between elements of our dynamic programming table. Given a set $T\subseteq V$, a set $X\subseteq V$ is a *T-independent set* if in $G[X]$ there is no edge incident on any vertex of $T\cap X$. Note that X is a T-independent set if and only if $\overline{X}$ is a T-vertex cover.

Definition 24. Let $X,W\subseteq V_x$ be T-independent sets. We say that X and W are *equivalent*, denoted by $X\sim_T W$, if $X\cap T\equiv_1^{V_x} W\cap T$ and $X\setminus T\equiv_1^{V_x} W\setminus T$.

We now prove the following lemma.

Lemma 25. *For every $Y\subseteq\overline{V_x}$ and every T-independent sets $X,W\subseteq V_x$ such that $X\sim_T W$, it holds that $X\cup Y$ is a T-independent set if and only if $W\cup Y$ is a T-independent set.*

Proof. By symmetry, it suffices to prove one direction. Suppose that $X\cup Y$ is a T-independent set, but $W\cup Y$ is not. Note that X and W are T-independent sets by definition and that Y must be a T-independent set as well, because $X\cup Y$ is. Hence, the fact that $W\cup Y$ is not a T-independent set implies there is an edge $uv\in E(G)$ for which:

1. $u\in W\cap T, v\in Y\cap T$,
2. $u\in W\cap T, v\in Y\setminus T$, or
3. $u\in W\setminus T, v\in Y\cap T$.

In the first case, since $v\in Y\cap T$ has a neighbour in $W\cap T$, note that $\min\{1,|(W\cap T)\cap N(v)|\}=1$. Since $X\cap T\equiv_1^{V_x} W\cap T$ by the assumption that $X\sim_T W$, it follows that $\min\{1,|(X\cap T)\cap N(v)|\}=1$. Hence, there is an edge from $v\in Y\cap T$ to $X\cap T$, contradicting that $X\cup Y$ is a T-independent set.

The second case is analogous to the first case. The third case is also analogous, but uses that $X\setminus T\equiv_1^{V_x} W\setminus T$. □

We now introduce a final definition.

Definition 26. For every $\mathcal{A}\subseteq 2^{V_x}$ and $Y\subseteq\overline{V_x}$, let

$$\mathsf{best}(\mathcal{A},Y)=\max\{\omega(X)\mid X\in\mathcal{A}\text{ and }X\cup Y\text{ is a }T\text{-independent set}\}.$$

Given $\mathcal{A},\mathcal{B}\subseteq 2^{V_x}$, we say that $\mathcal{B}$ *represents* $\mathcal{A}$ if $\mathsf{best}(\mathcal{A},Y)=\mathsf{best}(\mathcal{B},Y)$ for every $Y\subseteq\overline{V_x}$.

We use the above definition in our next lemma.

Lemma 27. *Given a set $\mathcal{A}\subseteq 2^{V_x}$, we can compute $\mathcal{B}\subseteq\mathcal{A}$ that represents $\mathcal{A}$ and has size at most $\mathsf{nec}_1(V_x)^2$ in $O(|\mathcal{A}|\cdot n^2\log(\mathsf{nec}_1(V_x))+\mathsf{nec}_1(V_x)\cdot n^2\log(\mathsf{nec}_1(V_x))+\mathsf{nec}_1(V_x)^2)$ time.*

Proof. We obtain $\mathcal{B}$ from $\mathcal{A}$ as follows: for all sets in $\mathcal{A}$ that are equivalent under $\sim_T$, maintain only a set X that is a T-independent set for which $\omega(X)$ is maximum. Note that if among a collection of equivalent sets, there is no T-independent set, then no set is maintained. By construction, $\mathcal{B}$ has size at most $\mathsf{nec}_1(V_x)^2$, because there are at most $\mathsf{nec}_1(V_x)^2$ equivalence classes of $\sim_T$ by Definition 24.

We now prove that $\mathcal{B}$ represents $\mathcal{A}$. Let $Y\subseteq\overline{V_x}$. Note that $\mathsf{best}(\mathcal{B},Y)\le\mathsf{best}(\mathcal{A},Y)$, because $\mathcal{B}\subseteq\mathcal{A}$. Hence, if there is no $X\in\mathcal{A}$ such that $X\cup Y$ is a T-independent set, then $\mathsf{best}(\mathcal{B},Y)=\mathsf{best}(\mathcal{A},Y)=-\infty$. So assume otherwise, and let $W\in\mathcal{A}$ satisfy $\omega(W)=\mathsf{best}(\mathcal{A},Y)$. This means that $W\cup Y$ is a T-independent set and in particular, W is a T-independent set. By the construction of $\mathcal{B}$, there is a $X\in\mathcal{B}$ that is a T-independent set with $X\sim_T W$ and $\omega(X)\ge\omega(W)$. By Lemma 25, $X\cup Y$ is a T-independent set. Hence, $\mathsf{best}(\mathcal{B},Y)\ge\omega(X)\ge\omega(W)=\mathsf{best}(\mathcal{A},Y)$. It follows that $\mathsf{best}(\mathcal{B},Y)=\mathsf{best}(\mathcal{A},Y)$ and thus $\mathcal{B}$ represents $\mathcal{A}$.

For the running time, note that we can implement the algorithm by maintaining a table indexed by pairs of representatives of the 1-neighbour equivalence classes. Note that a pair of representatives uniquely identifies an equivalence class of $\sim_T$. By Lemma 23, we can compute the indices in $O(\mathsf{nec}_1(V_x)\cdot n^2\log(\mathsf{nec}_1(V_x)))$ time, which gives us both the complete set of representatives and the data structure to compute representatives. Then for each $X\in\mathcal{A}$, we can compute its representatives in $O(|V_x|\cdot n\log(\mathsf{nec}_1(V_x)))$ time and check whether it is a T-independent set in $O(n^2)$ time. Creation of the table and returning $\mathcal{B}$ takes $O(\mathsf{nec}_1(V_x)^2)$ time. Hence, the total running time is $O(|\mathcal{A}| \cdot n^2 \log(\mathsf{nec}_1(V_x)) + \mathsf{nec}_1(V_x) \cdot n^2 \log(\mathsf{nec}_1(V_x)) + \mathsf{nec}_1(V_x)^2)$. □

We are now ready to prove the following result.

Theorem 28. *Let G be a graph on n vertices with a rooted layout (L, δ). We can solve* Subset Vertex Cover *in $O(\sum_{x\in V(L)}(\mathsf{nec}_1(V_x))^4 \cdot n^2 \log(\mathsf{nec}_1(V_x)))$ time.*

Proof. It suffices to find a maximum-weight T-independent set of G. For every node $x \in V(L)$, we aim to compute a set $\mathcal{A}_x \subseteq 2^{V_x}$ of T-independent sets such that $\mathcal{A}_x$ represents 2^{V_x} and has size at most $p(x) := \mathsf{nec}_1(V_x)^2 + 1$. Letting r denote the root of L, we then return the set in $\mathcal{A}_r$ of maximum weight. Since $\mathcal{A}_r$ represents 2^{V_r}, this is indeed a maximum-weight T-independent set of G.

We employ a bottom-up dynamic programming algorithm to compute $\mathcal{A}_x$. If x is a leaf with $V_x = \{v\}$, then set $\mathcal{A}_x = \{\emptyset, \{v\}\}$. Clearly, $\mathcal{A}_x$ represents 2^{V_x} and has size at most $p(x)$. So now suppose x is an internal node with children a, b. For any $\mathcal{A}, \mathcal{B} \subseteq 2^{V(G)}$, let $\mathcal{A} \otimes \mathcal{B} = \{X \cup Y \mid X \in \mathcal{A}, Y \in \mathcal{B}\}$. Now let $\mathcal{A}_x$ be equal to the result of the algorithm of Lemma 27 applied to $\mathcal{A}_a \otimes \mathcal{A}_b$. Then, indeed, $|\mathcal{A}_x| \le p(x)$. Using induction, it remains to show the following for the correctness proof:

Claim 28.1. *If $\mathcal{A}_a$ and $\mathcal{A}_b$ represent 2^{V_a} and 2^{V_b} respectively, then the computed set $\mathcal{A}_x$ represents 2^{V_x}.*

Proof. We prove Claim 28.1 as follows. If $\mathcal{A}_a \otimes \mathcal{A}_b$ represents 2^{V_x}, then by Lemma 27 and the transitivity of the 'represents' relation, it follows that $\mathcal{A}_x$ represents 2^{V_x}. So it suffices to prove that $\mathcal{A}_a \otimes \mathcal{A}_b$ represents 2^{V_x}. Let $Y \subseteq \overline{V_x}$. Note that

$$\begin{aligned}\mathsf{best}(\mathcal{A}_a \otimes \mathcal{A}_b, Y) &= \max\{\omega(X) + \omega(W) \mid X \in \mathcal{A}_a, W \in \mathcal{A}_b,\\ &\qquad X \cup W \cup Y \text{ is a } T\text{-independent set}\}\\ &= \max\{\mathsf{best}(\mathcal{A}_a, W \cup Y) + \omega(W) \mid W \in \mathcal{A}_b\}.\end{aligned}$$

Note that $\mathsf{best}(\mathcal{A}_a, W \cup Y) = \mathsf{best}(2^{V_a}, W \cup Y)$, as $\mathcal{A}_a$ represents 2^{V_a}, and thus $\mathsf{best}(\mathcal{A}_a \otimes \mathcal{A}_b, Y) = \mathsf{best}(2^{V_a} \otimes \mathcal{A}_b, Y)$. Using a similar argument, we can then show that $\mathsf{best}(2^{V_a} \otimes \mathcal{A}_b, Y) = \mathsf{best}(2^{V_a} \otimes 2^{V_b}, Y)$. Since $2^{V_x} = 2^{V_a} \otimes 2^{V_b}$, it follows that $\mathsf{best}(\mathcal{A}_a \otimes \mathcal{A}_b, Y) = \mathsf{best}(2^{V_x}, Y)$ and thus $\mathcal{A}_a \otimes \mathcal{A}_b$ represents 2^{V_x}. This completes the proof of Claim 28.1. ⋄

Finally, we prove the running time bound. Using induction, it follows that $|\mathcal{A}_a \otimes \mathcal{A}_b| \le p(x)^2$ for any internal node x with children a, b. Hence, $\mathcal{A}_a \otimes \mathcal{A}_b$ can be computed in $O(p(x)^2 \cdot n)$ time. Then, $\mathcal{A}_x$ can be computed in $O(p(x)^2 \cdot n^2 \log(\mathsf{nec}_1(V_x)) + \mathsf{nec}_1(V_x) \cdot n^2 \log(\mathsf{nec}_1(V_x)) + \mathsf{nec}_1(V_x)^2) = O((\mathsf{nec}_1(V_x))^4 \cdot n^2 \log(\mathsf{nec}_1(V_x)))$ time by Lemma 27. □

It was shown by Belmonte and Vatshelle [3] that $\mathsf{nec}_d(A) \le |A|^{d \cdot \mathsf{mim}(A)}$. Combining their result with Theorem 28 immediately yields the following.

Theorem 29. *Let G be a graph on n vertices with a rooted layout $\mathcal{L} = (L, \delta)$. Then* Subset Vertex Cover *can be solved in $O(\mathsf{mim}_G(\mathcal{L}) \cdot n^{4\mathsf{mim}_G(\mathcal{L})+3} \cdot \log(n))$ time.*

The following corollary is now immediate from the fact that circular-arc graphs have constant mim-width and a rooted layout of constant mim-width can be computed in polynomial time [3].

Corollary 30. *Subset Vertex Cover can be solved in polynomial time on circular-arc graphs.*

As interval graphs are circular-arc, Corollary 30 also holds for interval graphs, but we also note that the result for interval graphs already follows from Theorem 4.

7. Conclusions

Apart from giving a dichotomy for Subset Vertex Cover restricted to instances (G, T, k) where $G[T]$ is H-free (Theorem 8), we gave a partial classification of Subset Vertex Cover for H-free graphs (Theorem 7). Our partial classification resolved two open problems from the literature and showed that for some hereditary graph classes, Subset Vertex Cover is computationally harder than Vertex Cover (if P ≠ NP). This is in contrast to the situation for graph classes closed under edge deletion. Hence, Subset Vertex Cover is worth studying on its own, instead of only as an auxiliary problem (as in [8]). In order to complete the classification of Subset Vertex Cover for H-free graphs it remains to solve precisely the following open cases.

Open problem 1. Determine the computational complexity of Subset Vertex Cover for H-free graphs, where

- $H = sP_2 + P_3$ for $s \ge 2$; or
- $H = sP_2 + P_4$ for $s \ge 1$; or

- $H = sP_2 + P_t$ for $s \geq 0$ and $t \in \{5, 6\}$.

Brettell et al. [8] asked what the complexity of Subset Vertex Cover is for P_5-free graphs. In contrast, Vertex Cover is polynomial-time solvable even for P_6-free graphs [22]. However, the open cases where $H = sP_2 + P_t$ ($s \geq 1$ and $t \in \{4, 5, 6\}$) are even open for Vertex Cover on H-free graphs (though a quasipolynomial-time algorithm is known [18,31]). So for those cases we may want to first restrict ourselves to Vertex Cover instead of Subset Vertex Cover, or aim for a quasipolynomial-time algorithm first.

We also note that our polynomial-time algorithms for Subset Vertex Cover for sP_2-free graphs and $(P_2 + P_3)$-free graphs can easily be adapted to work for Weighted Subset Vertex Cover for sP_2-free graphs and $(P_2 + P_3)$-free graphs. In this more general problem variant, each vertex u is given some positive weight $w(u)$, and the question is whether there exists a T-vertex cover S with weights $w(S) = \sum_{u \in S} w(u) \leq k$. In contrast, Papadopoulos and Tzimas [30] proved that Weighted Subset Feedback Vertex Set is NP-complete for $5P_1$-free graphs, whereas Subset Feedback Vertex Set is polynomial-time solvable even for $(sP_1 + P_4)$-free graphs for every $s \geq 1$ [29] (see also Theorem 5). The hardness construction of Papadopoulos and Tzimas [30] can also be used to prove that Weighted Odd Cycle Transversal is NP-complete for $5P_1$-free graphs [9], while Subset Odd Cycle Transversal is polynomial-time solvable even for $(sP_1 + P_3)$-free graphs for every $s \geq 1$ [8] (see also Theorem 6).

Finally, we recall from Theorem 4 that Subset Vertex Cover can be solved in polynomial time on chordal graphs by using a reduction to Vertex Cover on perfect graphs and applying the linear programming method of [21]. However, it is not known if this algorithm is strongly-polynomial. In contrast, we gave a purely combinatorial algorithm for probe split graphs in Theorem 17 and for probe interval graphs in Corollary 30. This makes the following question interesting.

Open problem 2. Give a combinatorial algorithm for Subset Vertex Cover om the class of chordal graphs.

A standard approach for Vertex Cover on chordal graphs is dynamic programming over the clique tree of a chordal graph. However, a naive dynamic programming algorithm over the clique tree does not work for Subset Vertex Cover. This is because we may need to remember an exponential number of subsets of a bag (clique) and the bags can have arbitrarily large size. Hence, to solve Open Problem 2 new ideas are needed.

CRediT authorship contribution statement

Nick Brettell: Conceptualization, Formal analysis, Investigation, Methodology, Project administration, Validation, Visualization, Writing – original draft, Writing – review & editing. **Jelle J. Oostveen:** Conceptualization, Formal analysis, Investigation, Methodology, Project administration, Validation, Visualization, Writing – original draft, Writing – review & editing. **Sukanya Pandey:** Conceptualization, Formal analysis, Investigation, Methodology, Project administration, Validation, Visualization, Writing – original draft, Writing – review & editing. **Daniël Paulusma:** Conceptualization, Formal analysis, Investigation, Methodology, Project administration, Validation, Visualization, Writing – original draft, Writing – review & editing. **Johannes Rauch:** Conceptualization, Formal analysis, Investigation, Methodology, Project administration, Validation, Visualization, Writing – original draft, Writing – review & editing. **Erik Jan van Leeuwen:** Conceptualization, Formal analysis, Investigation, Methodology, Project administration, Validation, Visualization, Writing – original draft, Writing – review & editing.

Declaration of competing interest

The authors declare the following financial interests/personal relationships which may be considered as potential competing interests: Jelle Oostveen reports financial support was provided by Dutch Research Council. Nick Brettell reports financial support was provided by Royal Society of New Zealand Marsden Fund. Johannes Rauch reports financial support was provided by German Academic Scholarship Foundation (Studienstiftung des Deutschen Volkes). If there are other authors, they declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

No data was used for the research described in the article.

References

[1] Vladimir E. Alekseev, The effect of local constraints on the complexity of determination of the graph independence number, in: Combinatorial-Algebraic Methods in Applied Mathematics, 1982, pp. 3–13 (in Russian).
[2] Egon Balas, Chang Sung Yu, On graphs with polynomially solvable maximum-weight clique problem, Networks 19 (2) (1989) 247–253.
[3] Rémy Belmonte, Martin Vatshelle, Graph classes with structured neighborhoods and algorithmic applications, Theor. Comput. Sci. 511 (2013) 54–65.
[4] Benjamin Bergougnoux, Charis Papadopoulos, Jan Arne Telle, Node multiway cut and subset feedback vertex set on graphs of bounded mim-width, Algorithmica 84 (5) (2022) 1385–1417.
[5] J. Adrian Bondy, Uppaluri S.R. Murty, Graph Theory with Applications, Macmillan Education UK, 1976.
[6] Andreas Brandstädt, Raffaele Mosca, Maximum weight independent set for ℓclaw-free graphs in polynomial time, Discrete Appl. Math. 237 (2018) 57–64.
[7] Nick Brettell, Jake Horsfield, Andrea Munaro, Giacomo Paesani, Daniël Paulusma, Bounding the mim-width of hereditary graph classes, J. Graph Theory 99 (2022) 117–151.

[8] Nick Brettell, Matthew Johnson, Giacomo Paesani, Daniël Paulusma, Computing subset transversals in H-free graphs, Theor. Comput. Sci. 902 (2022) 76–92.
[9] Nick Brettell, Matthew Johnson, Daniël Paulusma, Computing weighted subset odd cycle transversals in H-free graphs, J. Comput. Syst. Sci. 128 (2022) 71–85.
[10] Binh-Minh Bui-Xuan, Jan Arne Telle, Martin Vatshelle, Fast dynamic programming for locally checkable vertex subset and vertex partitioning problems, Theor. Comput. Sci. 511 (2013) 66–76.
[11] David B. Chandler, Maw-Shang Chang, Ton Kloks, Jiping Liu, Sheng-Lung Peng, Probe graph classes, manuscript, 2012.
[12] Maw-Shang Chang, Ton Kloks, Dieter Kratsch, Jiping Liu, Sheng-Lung Peng, On the recognition of probe graphs of some self-complementary classes of perfect graphs, in: Proc. COCOON 2005, vol. 3595, 2005, pp. 808–817.
[13] Nina Chiarelli, Tatiana R. Hartinger, Matthew Johnson, Martin Milanič, Daniël Paulusma, Minimum connected transversals in graphs: new hardness results and tractable cases using the price of connectivity, Theor. Comput. Sci. 705 (2018) 75–83.
[14] Bruno Courcelle, Joost Engelfriet, Grzegorz Rozenberg, Handle-rewriting hypergraph grammars, J. Comput. Syst. Sci. 46 (1993) 218–270.
[15] Bruno Courcelle, Stephan Olariu, Upper bounds to the clique width of graphs, Discrete Appl. Math. 101 (2000) 77–114.
[16] Maria Chudnovsky, Neil Robertson, Paul Seymour, Robin Thomas, The strong perfect graph theorem, Ann. Math. 164 (2006) 51–229.
[17] Fedor V. Fomin, Pinar Heggernes, Dieter Kratsch, Charis Papadopoulos, Yngve Villanger, Enumerating minimal subset feedback vertex sets, Algorithmica 69 (2014) 216–231.
[18] Peter Gartland, Daniel Lokshtanov, Independent set on P_k-free graphs in quasi-polynomial time, in: Proc. FOCS 2020, 2020, pp. 613–624.
[19] Peter Gartland, Daniel Lokshtanov, Tomáš Masařík, Marcin Pilipczuk, Michal Pilipczuk, Paweł Rzążewski, Maximum weight independent set in graphs with no long claws in quasi-polynomial time, CoRR, arXiv:2305.15738, 2023.
[20] Martin C. Golumbic, Marina Lipshteyn, Chordal probe graphs, Discrete Appl. Math. 143 (2004) 221–237.
[21] Martin Grötschel, László Lovász, Alexander Schrijver, Polynomial algorithms for perfect graphs, Ann. Discrete Math. 21 (1984) 325–356.
[22] Andrzej Grzesik, Tereza Klimošová, Marcin Pilipczuk, Michal Pilipczuk, Polynomial-time algorithm for maximum weight independent set on P_6-free graphs, ACM Trans. Algorithms 18 (2022) 4.
[23] Dénes Kőnig, Gráfok és mátrixok, Mat. Fiz. Lapok 38 (1931) 116–119.
[24] Vadim V. Lozin, From matchings to independent sets, Discrete Appl. Math. 231 (2017) 4–14.
[25] Vadim V. Lozin, Raffaele Mosca, Maximum regular induced subgraphs in $2P_3$-free graphs, Theor. Comput. Sci. 460 (2012) 26–33.
[26] Yury Metelsky, Regina Tyshkevich, Line graphs of Helly hypergraphs, SIAM J. Discrete Math. 16 (2003) 438–448.
[27] Bojan Mohar, Face covers and the genus problem for apex graphs, J. Comb. Theory, Ser. B 82 (2001) 102–117.
[28] Andrea Munaro, On line graphs of subcubic triangle-free graphs, Discrete Math. 340 (2017) 1210–1226.
[29] Giacomo Paesani, Daniël Paulusma, Pawel Rzazewski, Classifying subset feedback vertex set for H-free graphs, in: Proc. WG 2022, in: LNCS, vol. 13453, 2022, pp. 412–424.
[30] Charis Papadopoulos, Spyridon Tzimas, Subset feedback vertex set on graphs of bounded independent set size, Theor. Comput. Sci. 814 (2020) 177–188.
[31] Marcin Pilipczuk, Michal Pilipczuk, Paweł Rzążewski, Quasi-polynomial-time algorithm for independent set in P_t-free graphs via shrinking the space of induced paths, in: Proc. SOSA 2021, 2021, pp. 204–209.
[32] Svatopluk Poljak, A note on stable sets and colorings of graphs, Comment. Math. Univ. Carol. 15 (1974) 307–309.
[33] Shuji Tsukiyama, Mikio Ide, Hiromu Ariyoshi, Isao Shirakawa, A new algorithm for generating all the maximal independent sets, SIAM J. Comput. 6 (1977) 505–517.
[34] Martin Vatshelle, New Width Parameters of Graphs, PhD thesis, University of Bergen, Norway, 2012.
[35] Peisen Zhang, Eric A. Schon, Stuart G. Fisher, Eftihia Cayanis, Janie Weiss, Susan Kistler, Philip E. Bourne, An algorithm based on graph theory for the assembly of contigs in physical mapping of DNA, CABIOS 10 (1994) 309–317.