Complexity of the Cluster Vertex Deletion Problem on H-Free Graphs

Hoang-Oanh Le ✉
Independent Researcher, Berlin, Germany

Van Bang Le ✉
Institut für Informatik, Universität Rostock, Germany

Abstract

The well-known Cluster Vertex Deletion problem (CLUSTER-VD) asks for a given graph G and an integer k whether it is possible to delete at most k vertices of G such that the resulting graph is a cluster graph (a disjoint union of cliques). We give a complete characterization of graphs H for which CLUSTER-VD on H-free graphs is polynomially solvable and for which it is NP-complete.

2012 ACM Subject Classification Theory of computation → Graph algorithms analysis; Mathematics of computing → Graph theory; Mathematics of computing → Graph algorithms

Keywords and phrases Cluster vertex deletion, Vertex cover, Computational complexity, Complexity dichotomy

Digital Object Identifier 10.4230/LIPIcs.MFCS.2022.68

1 Introduction and result

A very extensively studied version of graph modification problems asks to modify a given graph to a graph that satisfies a certain property $\mathcal{G}$ by deleting a minimum number of vertices. The case $\mathcal{G}$ being 'edgeless' is the well-known VERTEX COVER problem, one of the classical NP-hard problems. If $\mathcal{G}$ is a 'cluster graph', a graph in which every connected component is a clique, the corresponding problem is another well-known NP-hard problem, the CLUSTER VERTEX DELETION problem (CLUSTER-VD for short). In this paper, we revisit the computational complexity of CLUSTER-VD, formally given below.

CLUSTER-VD	
Instance:	A graph $G = (V, E)$ and an integer k.
Question:	Does there exist a vertex set $S \subseteq V$ of size at most k such that $G - S$ is a cluster graph?

Being an hereditary property on induced subgraphs, CLUSTER-VD is NP-complete [16], even when restricted to planar graphs [20] and to bipartite graphs [21], and to bipartite graphs of maximum degree 3 [13]. Most recent works on CLUSTER-VD deal with exact, fpt and approximation algorithms [1, 2, 14, 19].

It is noticeable that there are only a few known cases where the problem can be solved efficiently: CLUSTER-VD is polynomially solvable on block graphs and split graphs [3], and on graphs of bounded treewidth [18]. On the other hand, the complexity status of CLUSTER-VD on many well-studied graph classes is still open, e.g., chordal graphs discussed in [3] and planar bipartite graphs mentioned in [4].

In this paper we initiate studying the computational complexity of CLUSTER-VD on graphs defined by forbidding certain induced subgraphs. We remark that related approaches for other problems are quite common in the literature, e.g., for VERTEX COVER (aka INDEPENDENT SET) [9, 12] and COLORING [10, 15], and that many popular graph classes are defined or characterized by forbidding induced subgraphs, e.g., chordal and bipartite graphs (by infinitely many forbidden subgraphs), and cographs and line graphs (by finitely many forbidden subgraphs).

47th International Symposium on Mathematical Foundations of Computer Science (MFCS 2022).
Editors: Stefan Szeider, Robert Ganian, and Alexandra Silva; Article No. 68; pp. 68:1–68:10
Leibniz International Proceedings in Informatics
LIPICS Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

All graphs considered are undirected and finite and have no multiple edges or self-loops. Let H be a given graph. A graph G is *H-free* if no induced subgraph in G is isomorphic to H. A path with n vertices and $n-1$ edges is denoted by P_n. The main result of the present paper is the following complexity dichotomy:

▶ **Theorem 1.** *Let H be a fixed graph.* CLUSTER-VD *is polynomially solvable on H-free graphs if H is an induced subgraph of the 4-vertex path P_4, and* NP*-complete otherwise.*

Theorem 1 is remindful of the main result in [15] which characterizes all graphs H for which coloring H-free graphs is easy and for which it is hard. In fact, the present paper was motivated by this H-free coloring theorem.

For a set $\mathcal{H}$ of graphs, $\mathcal{H}$-free graphs are those in which no induced subgraph is isomorphic to a graph in $\mathcal{H}$. We denote by $K_{1,n}$ the tree with $n \geq 3$ vertices and n leaves, by C_n the n-vertex cycle, and as usual, by $\overline{G}$ the complement of a graph G. The union $G+H$ of two vertex-disjoint graphs G and H is the graph with vertex set $V(G) \cup V(H)$ and edge set $E(G) \cup E(H)$; we write pG for the union of p copies of G. For a subset $S \subseteq V(G)$, we let $G[S]$ denote the subgraph of G induced by S; $G - S$ stands for $G[V(G) \setminus S]$. By 'G contains an H' we mean G contains H as an induced subgraph.

A graph G is a *cluster graph* if each of its connected components is a clique. Observe that G is a cluster graph if and only if G is P_3-free. If $S \subseteq V(G)$ is a subset of vertices of G such that $G - S$ is P_3-free, then S is called a *cluster vertex deletion set* of G. An *optimal* cluster vertex deletion set is one of minimum size.

We first address the polynomial part of Theorem 1 in the next section. Then we present two new NP-completeness results for CLUSTER-VD in Sections 3 and 4. These hardness results allow us to clear the NP-completeness part of Theorem 1 in Section 5. The last section concludes the paper.

2 Polynomial cases

The polynomial part in Theorem 1 consists of six cases; see also Fig. 1 for all graphs H for which CLUSTER-VD is polynomially solvable on H-free graphs.

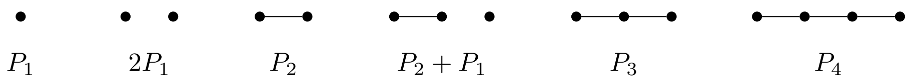

Figure 1 The graphs H for which CLUSTER-VD is polynomially solvable on H-free graphs.

Observe that H-freeness is hereditary, meaning if H' is an induced subgraph of H then H'-free graphs are H-free graphs. Thus, it suffices to prove the polynomial part only for the case where H is the 4-vertex path P_4.

We now are going to describe how to solve CLUSTER-VD in polynomial time when restricted to P_4-free graphs. P_4-free graphs are also called *cographs* [6]. More precisely, for vertex-disjoint graphs $G_i = (V_i, E_i)$, $i = 1, 2$, let $G_1 \textcircled{0} G_2$ be the union (or *co-join*) of G_1 and G_2,

$$G_1 \textcircled{0} G_2 = (V_1 \cup V_2, E_1 \cup E_2),$$

and let $G_1 \textcircled{1} G_2$ be the *join* of G_1 and G_2,

$$G_1 \textcircled{1} G_2 = (V_1 \cup V_2, E_1 \cup E_2 \cup \{uv \mid u \in V_1, v \in V_2\}).$$

With these notations, cographs are exactly those graphs that can be constructed from the one-vertex graph by applying the join and co-join operations. Thus, a cograph is the one-vertex graph or is the join of two smaller cographs or is the co-join of two smaller cographs.

Recall that $S \subseteq V(G)$ is a vertex cover if $G - S$ is edgeless and is a cluster vertex deletion set if $G - S$ is a cluster graph. Let $\tau(G)$ and $\varsigma(G)$ denote the vertex cover number and the cluster vertex deletion number of G, respectively,

$$\tau(G) = \min\{|S| : S \text{ is a vertex cover of } G\},$$
$$\varsigma(G) = \min\{|S| : S \text{ is a cluster vertex deletion set of } G\}.$$

We will see that $\tau(G)$ and $\varsigma(G)$ can be computed efficiently when restricted to cographs. The calculation is based on the following fact:

▶ **Lemma 2.** *For any (not necessarily P_4-free) graphs G_1 and G_2, the following relations hold:*

$$\tau(G_1 \circledcirc G_2) = \tau(G_1) + \tau(G_2); \tag{1}$$
$$\tau(G_1 ① G_2) = \min\{\tau(G_1) + |V(G_2)|, \tau(G_2) + |V(G_1)|\}; \tag{2}$$
$$\varsigma(G_1 \circledcirc G_2) = \varsigma(G_1) + \varsigma(G_2); \tag{3}$$
$$\varsigma(G_1 ① G_2) = \min\{\varsigma(G_1) + |V(G_2)|, \varsigma(G_2) + |V(G_1)|, \tau(\overline{G_1}) + \tau(\overline{G_2})\}. \tag{4}$$

Proof. (1) and (3) are trivial.

(2): Let S_i be a vertex cover of G_i of optimal size $\tau(G_i)$, $i = 1, 2$. Then $S_1 \cup V(G_2)$ and $S_2 \cup V(G_1)$ are vertex covers of $G_1 ① G_2$. Hence $\tau(G_1 ① G_2) \le \min\{|S_1| + |V(G_2)|, |S_2| + |V(G_1)|\} = \min\{\tau(G_1) + |V(G_2)|, \tau(G_2) + |V(G_1)|\}$.

For the other direction, let S be a vertex cover of $G_1 ① G_2$ of optimal size, and write $S_i = S \cap V(G_i)$. Then S_i is a vertex cover of G_i, and moreover, $S_1 = V(G_1)$ or else $S_2 = V(G_2)$. Hence $\tau(G_1 ① G_2) \ge \min\{|S_1| + |V(G_2)|, |S_2| + |V(G_1)|\} \ge \min\{\tau(G_1) + |V(G_2)|, \tau(G_2) + |V(G_1)|\}$.

(4): Let S_i be a cluster vertex deletion set of G_i of optimal size $\varsigma(G_i)$, $i = 1, 2$. Then $S_1 \cup V(G_2)$ and $S_2 \cup V(G_1)$ are cluster vertex deletion sets of $G_1 ① G_2$. Hence $\varsigma(G_1 ① G_2) \le \min\{|S_1| + |V(G_2)|, |S_2| + |V(G_1)|\} = \min\{\varsigma(G_1) + |V(G_2)|, \varsigma(G_2) + |V(G_1)|\}$. Let S_i be a vertex cover of $\overline{G_i}$ of optimal size $\tau(\overline{G_i})$, $i = 1, 2$. Then $S_1 \cup S_2$ is a cluster vertex deletion set of $G_1 ① G_2$, hence $\varsigma(G_1 ① G_2) \le |S_1| + |S_2| = \tau(\overline{G_1}) + \tau(\overline{G_2})$.

For the other direction, let S be a cluster vertex deletion set of $G_1 ① G_2$ of optimal size, and write $S_i = S \cap V(G_i)$. Then S_i is a cluster vertex deletion set of G_i, and moreover,

- if $G_1 - S_1$ is not a clique then $S_2 = V(G_2)$, likewise
- if $G_2 - S_2$ is not a clique then $S_1 = V(G_1)$.

In these two cases, $|S| = \varsigma(G_1 ① G_2) \ge \min\{|S_1| + |V(G_2)|, |S_2| + |V(G_1)|\} \ge \min\{\varsigma(G_1) + |V(G_2)|, \varsigma(G_2) + |V(G_1)|\}$. In the third case where each of $G_1 - S_1$ and $G_2 - S_2$ is a clique, S_1 and S_2 are vertex covers of $\overline{G_1}$ and $\overline{G_2}$, respectively. Hence in this case, $|S| = \varsigma(G_1 ① G_2) = |S_1| + |S_2| \ge \tau(\overline{G_1}) + \tau(\overline{G_2})$. ◀

▶ Remark 3. For any integer $r \ge 2$, Lemma 2 holds accordingly for $G_1 \circledcirc G_2 \circledcirc \cdots \circledcirc G_r = G_1 \circledcirc (G_2 \circledcirc \cdots \circledcirc G_r)$ and $G_1 ① G_2 ① \cdots ① G_r = G_1 ① (G_2 ① \cdots ① G_r)$. We also note that Lemma 2 holds for the weighted version, too.

With each cograph $G = (V, E)$, one can associate a so-called *cotree* T of G as follows.

- The leaves of T are the vertices of G;
- Every internal node of T has a label ⓪ or ①, and has at least two children;
- No two internal nodes of T with the same label are adjacent;
- Two vertices u and v of G are (non-)adjacent if and only if the least common ancestor of u and v in T has label ① (respectively, ⓪).

In particular, the cotree of an n-vertex cograph has at most $2n - 1$ nodes.

Note that, for any internal node v of T, the subtree T_v of T rooted at v is the cotree of the subgraph of G induced by the leaves of T_v. The cograph corresponding to T_v where v has label ⓪ is the disjoint union of the cographs corresponding to the children of v. The cograph corresponding to T_v where v has label ① is the join of the cographs corresponding to the children of v.

In particular, the cotree of $\overline{G}$ can be obtained from the cotree of G by changing the label ⓪ to ① and ① to ⓪.

In [7], a linear time algorithm is given for recognizing if a given graph is a cograph, and if so, constructing its cotree. Note that the cotree can immediately be transformed to an equivalent binary tree; see Fig. 2 for an example of a cograph G and the cotree of G and its binary version. For simplification, we will use the binary cotree in our algorithm below.

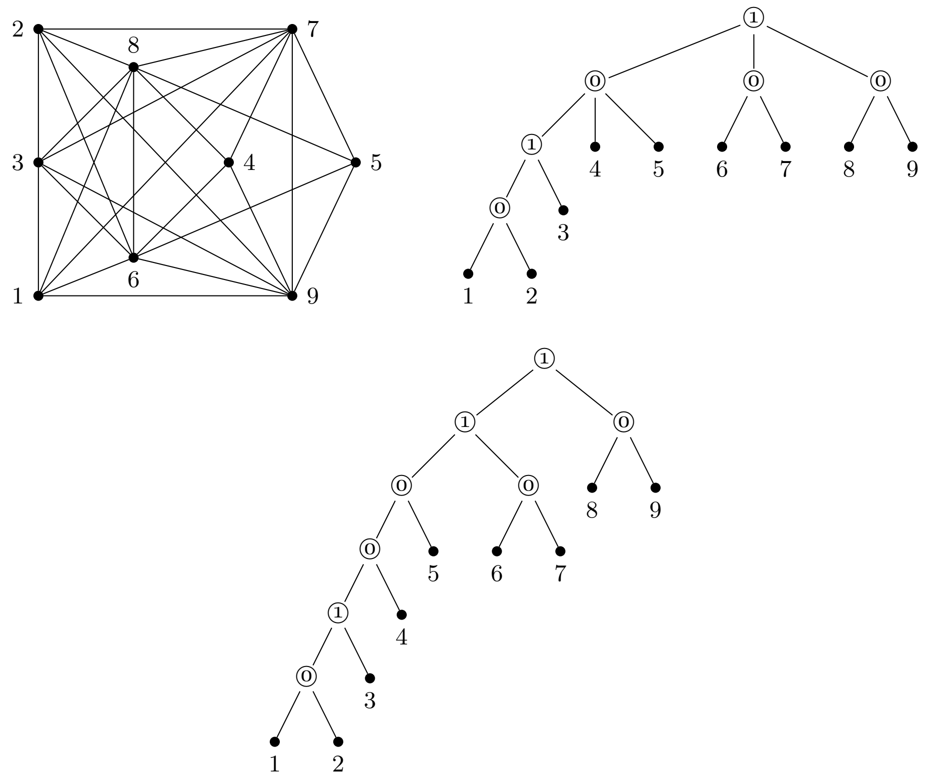

Figure 2 A cograph G, the cotree of G and its binary version.

Now, given a cograph G together with its binary cotree T, the bottom-up Algorithm 1 below computes the cluster vertex deletion number $\varsigma(G)$ of G, as suggested by Lemma 2. The algorithm uses the following notations. For a node v of T,

- if v is an internal node then $\ell(v)$ and $r(v)$ stands for the left child and the right child of v, respectively;

- $n(v)$ denotes the size of the subgraph of G induced by the leaves of T_v. Thus, if v is a leaf then $n(v) = 1$ and if v is the root of T then $n(v) = |V(G)|$;
- $\varsigma(v)$ denotes the cluster vertex deletion number of the subgraph of G induced by the leaves of T_v. Thus, if v is a leaf then $\varsigma(v) = 0$ and if v is the root of T then $\varsigma(v) = \varsigma(G)$;
- $\overline{\tau}(v)$ denotes the vertex cover number of the *complement* of the subgraph of G induced by the leaves of T_v. Thus, if v is a leaf then $\overline{\tau}(v) = 0$ and if v is the root of T then $\overline{\tau}(v) = \tau(\overline{G})$.

Algorithm 1 `computing cluster vertex deletion number`.

Input: A cograph $G = (V, E)$ together with its (binary) cotree T.
Output: $\varsigma(G)$, the cluster vertex deletion number of G

Traverse T by post-order and let v be the current node
if *v is a leaf* **then**
  $n(v) \leftarrow 1$; $\overline{\tau}(v) \leftarrow 0$; $\varsigma(v) \leftarrow 0$
end
else
  $n(v) \leftarrow n(\ell(v)) + n(r(v))$
  if *v has label* ⓪ **then**
    $\overline{\tau}(v) \leftarrow \min\{\overline{\tau}(\ell(v)) + n(r(v)), \overline{\tau}(r(v)) + n(\ell(v))\}$
    $\varsigma(v) \leftarrow \varsigma(\ell(v)) + \varsigma(r(v))$
  end
  if *v has label* ① **then**
    $\overline{\tau}(v) \leftarrow \overline{\tau}(\ell(v)) + \overline{\tau}(r(v))$
    $\varsigma(v) \leftarrow \min\{\varsigma(\ell(v)) + n(r(v)), \varsigma(r(v)) + n(\ell(v)), \overline{\tau}(v)\}$
  end
end

▶ **Proposition 4.** *Given a P_4-free n-vertex graph G together with its cotree, Algorithm 1 correctly computes the cluster deletion number $\varsigma(G)$ of G in $O(n)$ time.*

Proof. The correctness of Algorithm 1 directly follows from Lemma 2. Since per node in the cotree a constant number of operations are performed, the algorithm runs in $O(n)$ time. ◀

We remark that Algorithm 1 can be slightly modified for computing a minimum cluster vertex deletion set. Also, since Lemma 2 holds accordingly for the weighted version, the minimum weight cluster vertex deletion number of cographs can be computed in linear time, too.

Another approach is based on the concept of clique-width of graphs in connection with the so-called $LinEMSOL_{\tau_{1,p}}$ problems (problems definable in monadic second-order logic and allowed in searching optimal sets of vertices with respect to some linear objective function). We refer to [8] for details. It is well known that cographs have clique-width at most 2 and a corresponding 2-expression can be constructed in linear time. Hence, every $LinEMSOL_{\tau_{1,p}}$ problem on cographs can be solved in linear time [8, Theorem 4].

Observe that the optimization version of CLUSTER-VD, MINIMUM CLUSTER-VD, is a $LinEMSOL(\tau_1, p)$ problem, since it can be expressed as follows:

minimize $|S|$ with respect to
$\forall u, v, w\big(\neg S(u) \wedge \neg S(v) \wedge \neg S(w) \wedge E(u,v) \wedge E(v,w) \wedge (u \neq w) \rightarrow E(u,w)\big)$,

where $S(x)$ means $x \in S$ and $E(x,y)$ means $xy \in E(G)$.

We remark that MINIMUM WEIGHT CLUSTER-VD is also a $LinEMSOL_{\tau_{1,p}}$ problem, hence it can be solved in linear time on cographs, too.

3 Cluster-VD on sparse graphs

Recall that CLUSTER-VD is NP-complete on bipartite graphs [21], even when restricted to bipartite graphs of maximum degree 3 [13]. In this section, we show that, for any given tree T containing two vertices of degree 3, CLUSTER-VD remains NP-complete when restricted to T-free bipartite graphs of maximum degree 3 and with arbitrarily large girth. The girth $g(G)$ of a graph G is the smallest length of a cycle in G; we set $g(G) = \infty$ if G is a forest. Thus, for any fixed $g \geq 3$, $g(G) > g$ if and only if G is $\{C_3, C_4, \ldots, C_g\}$-free.

▶ **Theorem 5.** *For any given integer $g \geq 3$ and any given tree T containing two degree-3 vertices,* CLUSTER-VD *is* NP*-complete on T-free bipartite graphs of maximum degree 3 and with girth $> g$.*

In particular, CLUSTER-VD *is* NP*-complete on bipartite graphs of maximum degree 3 and with girth $> g$.*

Proof. We give a polynomial reduction from CLUSTER-VD on bipartite graphs of maximum degree 3 to CLUSTER-VD on T-free bipartite graphs of maximum degree 3 and of girth $> g$. (Note that NP-membership of CLUSTER-VD restricted to these graphs is clear.)

First, given a bipartite graph G of maximum degree 3 with m edges, let G' be obtained from G by subdividing each edge $e = xy$ in G with three new vertices e_x, e_{xy} and e_y, thus obtaining the 5-vertex path $xe_xe_{xy}e_yy$ in G' in which all new vertices are of degree 2. Note that like G, G' is bipartite. We claim that G has a cluster vertex deletion set of size at most k if and only if G' has a cluster vertex deletion set of size at most $k + m$. For one direction, extend a cluster vertex deletion set $S \subseteq V(G)$ to a cluster vertex deletion set $S' \subseteq V(G')$ of size $|S| + m$ as follows: initially, set $S' = S$. Then, for each edge $e = xy$ in G,

- if both x and y are in S or outside S, put e_{xy} into S';
- if $x \in S$ and $y \notin S$, put e_y into S';
- if $x \notin S$ and $y \in S$, put e_x into S'.

To see that $G' - S'$ is P_3-free, notice that by construction, for each edge $e = xy$ in G, exactly one of e_x, e_{xy} and e_y is in S', and if $e_x, e_{xy} \notin S'$ then $x \in S$, and if $e_x, x \notin S'$ then $y \in S$. Since each P_3 in G' has the form xe_xe_{xy}, $e_xe_{xy}e_y$ or $e_xxe'_x$ for some edge $e = xy$ and $e' = xz$, it follows from these facts that $G' - S'$ is P_3-free.

For the other direction, suppose that G' has a cluster deletion set of size at most $k + m$, and consider such a set S' of minimum size. Then, we may assume that, for each edge $e = xy$ in G, S' contains exactly one of e_x, e_{xy} and e_y: note that $e_xe_{xy}e_y$ is a P_3, hence $|S' \cap \{e_x, e_{xy}, e_y\}| \geq 1$, and by minimality, $|S' \cap \{e_x, e_{xy}, e_y\}| \leq 2$. Now, if $|S' \cap \{e_x, e_{xy}, e_y\}| = 2$ for some edge $e = xy$ in G, then S' can be modified to a minimum cluster vertex deletion set containing exactly one of e_x, e_{xy} and e_y as follows:

- suppose that $e_x, e_{xy} \in S'$. Then $x, y \notin S'$ (if $x \in S'$ then $S' - e_x$ would be a cluster vertex deletion set of G', and if $y \in S'$ then $S' - e_{xy}$ would be a cluster vertex deletion set of G', contradicting the minimality of S'), and $S'' = S' - e_{xy} + y$ is a desired cluster vertex deletion set of minimum size;
- suppose that $e_y, e_{xy} \in S'$. Then similar to the above case, $x, y \notin S'$, and $S'' = S' - e_{xy} + x$ is a desired cluster vertex deletion set of minimum size;
- suppose that $e_x, e_y \in S'$. Then $x, y \notin S'$ (if $x \in S'$ or $y \in S'$ then $S'' = S' - e_x$, respectively $S' - e_y$, would be a cluster vertex deletion set of G', contradicting the minimality of S'), and $S'' = S' - e_x + x$ is a desired cluster vertex deletion set of minimum size.

Hence, $S = S' \cap V(G)$ has at most k vertices, and $G - S$ is P_3-free: if there would be an induced P_3 xyz in G with edges $e = xy$ and $e' = yz$, then, as $|S' \cap \{e_x, e_{xy}, e_y\}| = 1$ $= |S' \cap \{e'_y, e'_{yz}, e'_z\}|$, one of the 3-paths xe_xe_{xy}, $e_yye'_y$ and $e'_{yz}e'_zz$ would be outside S'.

Thus, G has a cluster vertex deletion set of size at most k if and only if G' has a cluster vertex deletion set of size at most $k + m$, as claimed.

Now, given $g > 0$ and a tree T with two degree-3 vertices, repeating the construction $t = \max\{\log_4(g/g(G)) + 1, |V(T)|\}$ times, the final bipartite graph has girth $4^t g(G) > g$ and maximum degree 3 and contains no induced subgraph isomorphic to T. ◀

4 Cluster-VD on dense graphs

In this section, we give a polynomial reduction from VERTEX COVER to CLUSTER-VD, showing that CLUSTER-VD remains NP-complete when restricted to $\{3P_1, 2P_2\}$-free graphs.

Recall that the VERTEX COVER problem asks, for a given graph G and an integer k, if one can delete a vertex set S of size at most k such that $G - S$ is edgeless. Let (G, k) be an instance for VERTEX COVER. We may assume that

- G is not perfect.[1] This is because VERTEX COVER is polynomially solvable on perfect graphs (see [11]); notice that G is perfect if and only if $\overline{G}$ is perfect and perfect graphs can be recognized in polynomial time [5]),
- G has girth $> g$ for any given integer $g \geq 3$ (see, e.g., [17]), and
- $k \leq |V(G)|/2$. This fact is probably known and can be easily seen as follows: given G with n vertices and an integer k, let G' be obtained from G by adding $p = \max\{0, 2k - n\}$ isolated vertices. Then $k = |V(G')|/2$ and $(G, k) \in$ VERTEX COVER if and only if $(G', k) \in$ VERTEX COVER. Notice that like G, G' satisfies the first two conditions, too.

From (G, k) we construct an equivalent instance (G', k') for CLUSTER-VD as follows: G' is obtained from two disjoint copies of $\overline{G}$, G_1 and G_2, by adding all possible edges between $V(G_1)$ and $V(G_2)$. Set $k' = 2k$.

We argue that $(G, k) \in$ VERTEX COVER if and only $(G', k') \in$ CLUSTER-VD. First, let $S \subset V(G)$ with $|S| \leq k$ be such that $G - S$ is edgeless. Let S_1 and S_2 be the copy of S in G_1 and G_2, respectively. Then $G_i - S_i$ is a clique in $G_i = \overline{G}$, hence $G' - S'$ is a clique in G' where $S' = S_1 \cup S_2$ with $|S'| = 2|S| \leq 2k = k'$. Conversely, let $S' \subseteq V(G')$ be a cluster vertex deletion set of G' with $|S'| \leq k' = 2k$. Observe that $S' \cap V(G_i)$ is a proper nonempty subset of both $V(G_1)$ and $V(G_2)$: if for some i, $S' \cap V(G_i) = \emptyset$ then G_i (hence G) would be perfect, and if $V(G_i) \subset S'$ then $2k \geq |S'| > |V(G_i)| = |V(G)|$, contradicting $k \leq |V(G)|/2$. It follows from the above that $G' - S'$ is a single clique, implying for each i, $G_i - S_i$ is a

[1] Actually, we will only use the fact that $\overline{G}$ contains at least one induced P_3.

clique in G_i where $S_i = S' \cap V(G_i)$. Since $|S'| \leq 2k$, $|S_1| \leq k$ or $|S_2| \leq k$. Let $|S_1| \leq k$, say, and let $S \subseteq V(G)$ be the set of the corresponding vertices in G. Then $G - S$ is edgeless with $|S| \leq k$.

We have seen that G has a vertex cover of size at most k if and only if G' has a cluster vertex deletion set of size at most k', as claimed.

Now, observe that, for any *connected* graph X, if G is X-free then G' is $\overline{X}$-free. Since G has girth $> g$, we obtain:

▶ **Theorem 6.** *For any fixed $g \geq 3$, CLUSTER-VD is NP-complete on $\{\overline{C_3}, \overline{C_4}, \ldots, \overline{C_g}\}$-free graphs.*

In particular, CLUSTER-VD is NP-complete on $\{3P_1, 2P_2\}$-free graphs.

5 NP-completeness cases

In this section we give the proof of the NP-completeness part of Theorem 1.

Let H be a fixed graph. By Proposition 4, CLUSTER-VD is polynomially solvable on H-free graphs whenever H is an induced subgraph of the 4-vertex path P_4. The following fact is easy to see:

▶ **Observation 7.** *A graph is an induced subgraph of the 4-path P_4 if and only if it is a $\{3P_1, 2P_2\}$-free forest.*

Thus, it remains to consider the cases where H contains a cycle or a $3P_1$ or a $2P_2$ as an induced subgraph.

Now, if H contains a cycle then graphs of girth $> g = |V(H)|$ are H-free, hence Theorem 5 implies that CLUSTER-VD is NP-complete on H-free graphs. If H contains a $3P_1$ or a $2P_2$ then $\{3P_1, 2P_2\}$-free graphs are H-free graphs, hence Theorem 6 implies that CLUSTER-VD is NP-complete on H-free graphs.

The proof of Theorem 1 is complete.

6 Conclusion

We have found a complete characterization of graphs H for which CLUSTER-VD on H-free graphs is polynomially solvable and for which it is NP-complete (Theorem 1).

We remark that a complexity dichotomy for VERTEX COVER on H-free graphs, like Theorem 1 for CLUSTER-VD, seems very hard to achieve. Indeed, it is a long-standing open problem whether there exists a constant t for which VERTEX COVER is NP-complete on P_t-free graphs. So far it is known that such a constant t, if any, must be at least 7 [12].

Let $\mathcal{H}$ be a set of (possibly infinitely many) graphs. A natural question generalizing the case of one forbidden induced subgraph is: what is the complexity of CLUSTER-VD on $\mathcal{H}$-free graphs? The case $\mathcal{H} = \{H\}$ is completely solved by Theorem 1. The case $\mathcal{H} = \{C_\ell \mid \ell \geq 4\}$ addressed in [3] is still open. The next step may be the case of two-element sets $\mathcal{H} = \{H_1, H_2\}$. This case is more complex and currently we are investigating the case $\mathcal{H} = \{H, \overline{H}\}$. Another interesting problem is to clear the complexity of CLUSTER-VD on line graphs, a well-studied graph class defined by excluding nine small induced subgraphs.

References

1 Manuel Aprile, Matthew Drescher, Samuel Fiorini, and Tony Huynh. A tight approximation algorithm for the cluster vertex deletion problem. *Math. Program.*, 2022. doi:10.1007/s10107-021-01744-w.

2 Anudhyan Boral, Marek Cygan, Tomasz Kociumaka, and Marcin Pilipczuk. A fast branching algorithm for cluster vertex deletion. *Theory Comput. Syst.*, 58(2):357–376, 2016. doi:10.1007/s00224-015-9631-7.

3 Yixin Cao, Yuping Ke, Yota Otachi, and Jie You. Vertex deletion problems on chordal graphs. *Theor. Comput. Sci.*, 745:75–86, 2018. doi:10.1016/j.tcs.2018.05.039.

4 Dibyayan Chakraborty, L. Sunil Chandran, Sajith Padinhatteeri, and Raji R. Pillai. Algorithms and complexity of s-club cluster vertex deletion. In Paola Flocchini and Lucia Moura, editors, *Combinatorial Algorithms - 32nd International Workshop, IWOCA 2021, Ottawa, ON, Canada, July 5-7, 2021, Proceedings*, volume 12757 of *Lecture Notes in Computer Science*, pages 152–164. Springer, 2021. doi:10.1007/978-3-030-79987-8_11.

5 Maria Chudnovsky, Gérard Cornuéjols, Xinming Liu, Paul D. Seymour, and Kristina Vuskovic. Recognizing berge graphs. *Combinatorica*, 25(2):143–186, 2005. doi:10.1007/s00493-005-0012-8.

6 Derek G. Corneil, H. Lerchs, and L. Stewart Burlingham. Complement reducible graphs. *Discret. Appl. Math.*, 3(3):163–174, 1981. doi:10.1016/0166-218X(81)90013-5.

7 Derek G. Corneil, Yehoshua Perl, and Lorna K. Stewart. A linear recognition algorithm for cographs. *SIAM J. Comput.*, 14(4):926–934, 1985. doi:10.1137/0214065.

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

9 Peter Gartland and Daniel Lokshtanov. Independent set on P_k-free graphs in quasi-polynomial time. In Sandy Irani, editor, *61st IEEE Annual Symposium on Foundations of Computer Science, FOCS 2020, Durham, NC, USA, November 16-19, 2020*, pages 613–624. IEEE, 2020. doi:10.1109/FOCS46700.2020.00063.

10 Petr A. Golovach, Matthew Johnson, Daniël Paulusma, and Jian Song. A survey on the computational complexity of coloring graphs with forbidden subgraphs. *J. Graph Theory*, 84(4):331–363, 2017. doi:10.1002/jgt.22028.

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

12 Andrzej Grzesik, Tereza Klimosová, Marcin Pilipczuk, and Michal Pilipczuk. Polynomial-time algorithm for maximum weight independent set on P_6-free graphs. *ACM Trans. Algorithms*, 18(1):4:1–4:57, 2022. doi:10.1145/3414473.

13 Sun-Yuan Hsieh, Hoàng-Oanh Le, Van Bang Le, and Sheng-Lung Peng. On the d-claw vertex deletion problem. *CoRR*, abs/2203.06766, 2022. doi:10.48550/arXiv.2203.06766.

14 Falk Hüffner, Christian Komusiewicz, Hannes Moser, and Rolf Niedermeier. Fixed-parameter algorithms for cluster vertex deletion. *Theory Comput. Syst.*, 47(1):196–217, 2010. doi:10.1007/s00224-008-9150-x.

15 Daniel Král, Jan Kratochvíl, Zsolt Tuza, and Gerhard J. Woeginger. Complexity of coloring graphs without forbidden induced subgraphs. In Andreas Brandstädt and Van Bang Le, editors, *Graph-Theoretic Concepts in Computer Science, 27th International Workshop, WG 2001, Boltenhagen, Germany, June 14-16, 2001, Proceedings*, volume 2204 of *Lecture Notes in Computer Science*, pages 254–262. Springer, 2001. doi:10.1007/3-540-45477-2_23.

16 John M. Lewis and Mihalis Yannakakis. The node-deletion problem for hereditary properties is NP-complete. *J. Comput. Syst. Sci.*, 20(2):219–230, 1980. doi:10.1016/0022-0000(80)90060-4.

17 Owen J. Murphy. Computing independent sets in graphs with large girth. *Discret. Appl. Math.*, 35(2):167–170, 1992. doi:10.1016/0166-218X(92)90041-8.

18 Ignasi Sau and Uéverton dos Santos Souza. Hitting forbidden induced subgraphs on bounded treewidth graphs. *Inf. Comput.*, 281:104812, 2021. doi:10.1016/j.ic.2021.104812.

19 Dekel Tsur. Faster parameterized algorithm for cluster vertex deletion. *Theory Comput. Syst.*, 65(2):323–343, 2021. doi:10.1007/s00224-020-10005-w.

20 Mihalis Yannakakis. Node- and edge-deletion np-complete problems. In Richard J. Lipton, Walter A. Burkhard, Walter J. Savitch, Emily P. Friedman, and Alfred V. Aho, editors, *Proceedings of the 10th Annual ACM Symposium on Theory of Computing, May 1-3, 1978, San Diego, California, USA*, pages 253–264. ACM, 1978. doi:10.1145/800133.804355.

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