Journal of the Brazilian Computer Society, 2025, 31:1, doi: 10.5753/jbcs.2025.5191

Characterizations and recognition algorithms for distance-hereditary graphs partitionable into a stable set and a forest

Maria Luíza López da Cruz [**Fluminense Federal University, Niterói, Brazil** | ***marialopez@id.uff.br***]
Victor Rangel Ramos [**Fluminense Federal University, Niterói, Brazil** | ***vrangelramos@gmail.com***]
Rodolfo A. Oliveira [**Fluminense Federal University, S.A. de Pádua, Brazil** | ***rodolfooliveira@id.uff.br***]
Raquel S. F. Bravo [**Fluminense Federal University, Niterói, Brazil** | ***raquel@ic.uff.br***]
Uéverton S. Souza [**IMPA Tech, Instituto de Matemática Pura e Aplicada (IMPA), Rio de Janeiro, Brazil, Fluminense Federal University, Niterói, Brazil** | ***ueverton@ic.uff.br***]
Received: 18 November 2024 • **Accepted:** 23 December 2024 • **Published:** 05 June 2025

Abstract Given a simple graph $G = (V, E)$, the Near-Bipartiteness problem asks whether $V(G)$ can be partitioned into two sets $\mathcal{S}$ and $\mathcal{F}$ such that $\mathcal{S}$ is a stable set and $\mathcal{F}$ induces a forest. Alternatively, the Near-Bipartiteness problem can be seen as the problem of determining whether G admits an independent feedback vertex set $\mathcal{S}$. Since such a problem is NP-complete even for perfect graphs, in this paper, our goal is to study the property of being near-bipartite on distance-hereditary graphs, a well-known subclass of perfect graphs. We show that there is an infinite set of minimal forbidden subgraphs for a distance-hereditary graph to be near-bipartite. In addition, we present a finite set of forbidden subgraphs that give us a sufficient condition for the existence of a near-bipartition in such a graph class. Finally, by using one-vertex-extension trees, we present a linear-time algorithm for the Near-Bipartiteness problem on distance-hereditary graphs.

Keywords: Near-Bipartiteness, Feedback Vertex Set, Stable Set, Independent Feedback Vertex, Distance-Hereditary Graphs

1 Introduction

In 1972, Richard Karp presented the NP-completeness proof of 21 fundamental problems for Computer Science [Karp, 1972]. Feedback Vertex Set and Independent Set are two of these classical problems. Feedback Vertex Set consists of finding a minimum set of vertices, such that its removal eliminates all cycles of the input graph, and Independent Set consists of determining a maximum set of pairwise nonadjacent vertices (also known as stable set). An independent feedback vertex set (IFVS) of a graph is a set of vertices that is mutually feedback vertex set and independent/stable. Defined by Yang A. and Yuan J. in [Smith and Walford, 1975], a graph $G = (V, E)$ has a *near-bipartition* $(\mathcal{S}, \mathcal{F})$ if there exist $\mathcal{S} \subseteq V$ and $\mathcal{F} = V \setminus \mathcal{S}$ such that $\mathcal{S}$ is a stable set, and $\mathcal{F}$ induces a forest ($\mathcal{S}$ and $\mathcal{F}$ may be empty sets). A graph that admits a near-bipartition is a *near-bipartite graph*. Note that the class of near-bipartite graphs is exactly the class of graphs having independent feedback vertex sets.

The problem of recognizing near-bipartite graphs, so-called Near-Bipartiteness, is NP-complete even restrict to graphs with maximum degree four [Smith and Walford, 1975], graphs with diameter three [Bonamy *et al.*, 2018], line graphs [Bonamy *et al.*, 2019], and planar graphs [Bonamy *et al.*, 2017; Dross *et al.*, 2017]. On the other hand, Brandstädt et al. [Brandstädt *et al.*, 2013] showed that Near-Bipartiteness is polynomial-time solvable on cographs. Yang and Yuan [Smith and Walford, 1975] showed that Near-Bipartiteness is polynomial-time solvable for graphs of diameter at most two and that every connected graph of maximum degree at most three is near-bipartite except the complete graph on four vertices, K_4. Bonamy et al. [Bonamy *et al.*, 2019] proved that a minimum independent feedback vertex set of a P_5-free graph can be found in $O(n^{16})$ time. In 2023, Cruz, Bravo, Oliveira, and Souza da Cruz *et al.* [2023] presented a $O(n^2 \cdot m)$-time algorithm to solved Near-Bipartiteness on P_5-free graphs. Besides, Bravo, Oliveira, Silva Junior, and Souza Bravo *et al.* [2023] presented a finite forbidden induced subgraph characterization for P_4-tidy near-bipartite graphs.

FPT algorithms parameterized by k for finding an k-independent feedback vertex can be found in [Agrawal *et al.*, 2017; Misra *et al.*, 2012].

A *coloring* for a graph G is an assignment of colors (labels) to all vertices of G. A *proper coloring* for G is an assignment of color $c(u)$, for each vertex $u \in V$, such that $c(u) \neq c(v)$ if $uv \in E(G)$. A graph G is *k-colorable* if there exists an proper coloring for G with at most k colors. The *chromatic number* of G, $\chi(G)$, is smaller number k for G being k-colorable. A clear necessary condition for a graph to be near-bipartite is the following.

Proposition 1. *If a graph G is near-bipartite then G is 3-colorable.*

By Proposition 1, it holds that K_4 is a natural forbidden subgraph for near-bipartite graphs. A graph G is called *perfect* if for every induced subgraph H of G holds that its chromatic number equals the size of its largest clique, $\chi(H) =$

$\omega(H)$. In particular, $\omega(G) = \chi(G)$. Given the relationship between Near-Bipartiteness and 3-Coloring it becomes interesting to question the behavior of Near-Bipartiteness on subclasses of perfect graphs. Note that a perfect graph is 3-colorable if and only if it is K_4-free. However, it is important to note that the complexity of 3-Coloring and Near-Bipartiteness are not necessarily the same, depending on the graph class being explored. Grötschel, Lovász e Schrijver [Grötschel *et al.*, 1984] proved that Coloring is solved in polynomial time for perfect graphs, while Brandstädt et al. [Brandstädt *et al.*, 2013] proved that Near-Bipartiteness is NP-complete in the same graph class.

In [Brandstädt *et al.*, 2013], it is showed that Near-Bipartiteness is polynomial-time solvable for cographs, but NP-complete on perfect graphs, in this paper we analyse the Near-Bipartiteness problem on *distance-hereditary graphs*, a natural superclass of cographs that are also perfect.

1.1 Auxiliary definitions and notations

Let $G = (V, E)$ be a simple graph. A subgraph of G *induced* by S is denoted by $G[S]$. For a vertex $v \in V$, denote the *open neighborhood of* v as $N(v) = \{u \in V : vu \in E\}$, and the *closed neighborhood of* v as $N[v] = N(v) \cup \{v\}$. For the case of $V' \subseteq V$, denote the *closed neighborhood of* V' as $N[V'] = \{v \in N[v'] : \text{ for all } v' \in V'\}$, and *open neighborhood of* V' as $N(V') = N[V'] \setminus V'$. Consider K_n a complete graph with n vertices, and $\omega(G)$ the size of the maximum clique of G. If every vertex of $S \subseteq V$ is an isolated vertex in $G[S]$, then S is called a *stable set* or an *independent set*. Given two graphs G_1 and G_2, the *union* between G_1 and G_2, $G_1 \cup G_2$, is defined as graph G resulting from $V = V(G_1) \cup V(G_2)$ and $E = E(G_1) \cup E(G_2)$, and the *join* between G_1 and G_2, $G_1 + G_2$, is defined as graph G resulting from $V = V(G_1) \cup V(G_2)$ and $E = E(G_1) \cup E(G_2) \cup \{uv : u \in V(G_1) \text{ and } v \in V(G_2)\}$. For $u, v \in V$, consider $d_G(u, v)$ as the distance between u and v in G. A *chord* in a path is an edge between non-subsequent vertices of such a path. An *induced path* is a path with no chord. A *triangle* is a clique of size three, K_3. A *diagonal* in cycles is an edge between non-subsequent vertices of such a cycle. A *diamond* is an induced cycle C_4 with one diagonal. An *induced cycle* is a cycle with no diagonals. A *hole* is any induced cycle C_k, for $k \geq 5$. A *house* is the cycle C_5 with one diagonal, and a *gem* is the C_5 with two no crossing diagonals. A *domino* is the cycle C_6 with one diagonal yielding two distinct C_4 as subgraphs. The graph G is said to be *acyclic* if it does not contain any cycle as subgraph. All connected and acyclic graph is called a *tree*. If the graph is only acyclic, then it is called a *forest*.

A graph class $\mathcal{C}$ is *hereditary* if for any $G \in \mathcal{C}$ it holds that any induced subgraph of G also belongs to $\mathcal{C}$. Note that the class of near-bipartite graphs is hereditary. Given a graph property Π, a graph G is (vertex) inclusion-wise minimal with respect to Π if G satisfies Π and no induced subgraph of G satisfy Π. For a hereditary graph class $\mathcal{C}$, a graph that is vertex-inclusion-wise minimal with respect to the property of not belonging to class $\mathcal{C}$ is called a minimal forbidden induced subgraph of $\mathcal{C}$, or a minimal obstruction for $\mathcal{C}$.

The set S of all minimal forbidden induced subgraphs of a hereditary graph class $\mathcal{C}$ provides a structural characterization for the class $\mathcal{C}$. For instance, the class of cographs is a hereditary graph class characterized by a finite set of minimal forbidden induced subgraphs: a graph is a cograph if and only if it is P_4-free (does not contain P_4 as induced subgraph). On the other hand, chordal graphs is characterized by an infinite family of minimal forbidden induced subgraphs: a graph is chordal if and only if it does not contains induced cycles of size at least four.

We use standard graph-theoretic concepts, and any undefined notation can be found in [Bondy *et al.*, 1976].

2 Preliminaries

In this section, we present some sufficient conditions for some graphs to admit a near-bipartition.

Theorem 1. *Every $\{K_4, C_4$,even-hole$\}$-free perfect graph is near-bipartite.*

Proof. Let G be a $\{K_4, C_4$,even-hole$\}$-free perfect graph. Since G is perfect and K_4-free then G is 3-colorable. Let V_1, V_2, V_3 be a partition of $V(G)$ into three stable sets. Since G is $\{C_4$,even-hole$\}$-free then $G[V_2 \cup V_3]$ is a $\{C_4$,even-hole$\}$-free bipartite graph, which implies that $G[V_2 \cup V_3]$ is a forest. Therefore, $(\mathcal{S} = V_1, \mathcal{F} = V_2 \cup V_3)$ is a near-bipartition of G. □

Corollary 1. *A chordal graph G is near-bipartite if and only if G is K_4-free.*

A k-tree is an undirected graph formed by starting with a $(k + 1)$-vertex complete graph and then repeatedly adding vertices in such a way that each added vertex v has exactly k neighbors U such that, together, the $k + 1$ vertices formed by v and U form a clique. A graph is a partial k-tree if it is a subgraph of a k-tree.

Corollary 2. *Every partial 2-tree is near-bipartite.*

Proof. By Corollary 1, every 2-tree is near-bipartite, and as the property of being near-bipartite is closed under edge removal, every partial 2-tree is also near-bipartite. □

From Corollary 2, it follows that every series-parallel graph (which includes the outerplanar graphs) is near-bipartite.

Defined in [Howorka, 1977], *distance-hereditary graphs* are connected graphs G in which all induced paths of G are isometric, i.e., if the distance function in every connected induced subgraph of G is the same as in G itself. It is well-known that distance-hereditary graphs is a hereditary graph class and that every distance-hereditary graph is perfect. Let $\mathcal{DH}$ be the class of distance-hereditary graphs.

Proposition 2. *[Bandelt and Mulder, 1986] $G \in \mathcal{DH}$ if and only if G contains neither hole, house, gem nor domino as induced subgraph.* (see Figure 1)

From Theorem 1 the following holds.

Corollary 3. *Every $\{K_4, C_4\}$-free distance-hereditary graph is near-bipartite.*

Corollary 3 motivates the study of near-bipartition aspects of distance-hereditary graphs presented in Section 3.

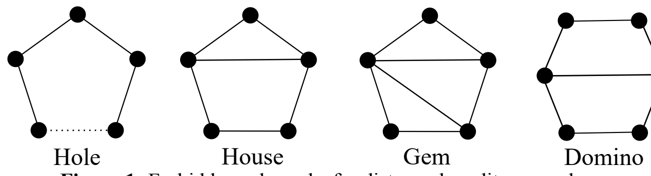


Figure 1. Forbidden subgraphs for distance-hereditary graphs.

3 On distance-hereditary graphs

Consider the following distance-hereditary graphs presented in Figure 2. The graph H_1 has a near-bipartition $(\mathcal{S},\mathcal{F})$ where $\mathcal{S}=\{v_1,v_4\}$ and $\mathcal{F}=\{v_2,v_3,v_5\}$, but there is no bipartition in which v_5 belongs to the stable set. Besides, the graph H_2 has a near-bipartition $(\mathcal{S},\mathcal{F})$ with $\mathcal{S}=\{v_2,v_5\}$ and $\mathcal{F}=\{v_1,v_3,v_4,v_6\}$, and, meanwhile, v_2 and v_5 must belong to $\mathcal{S}$ since another way yields a cycle.

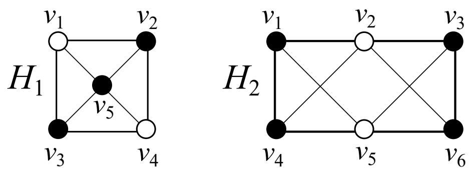


Figure 2. $H_1, H_2 \in \mathcal{DH}$ and its respective near-bipartitions $(\mathcal{S},\mathcal{F})$, where the white vertices are in $\mathcal{S}$ and the black ones are in $\mathcal{F}$.

Using H_1 or H_2, and chains of diamonds or C_4's, in Figure 3 we show the construction of three infinite families of forbidden induced subgraphs for near-bipartite graphs in $\mathcal{DH}$. Thus, Theorem 2 holds.

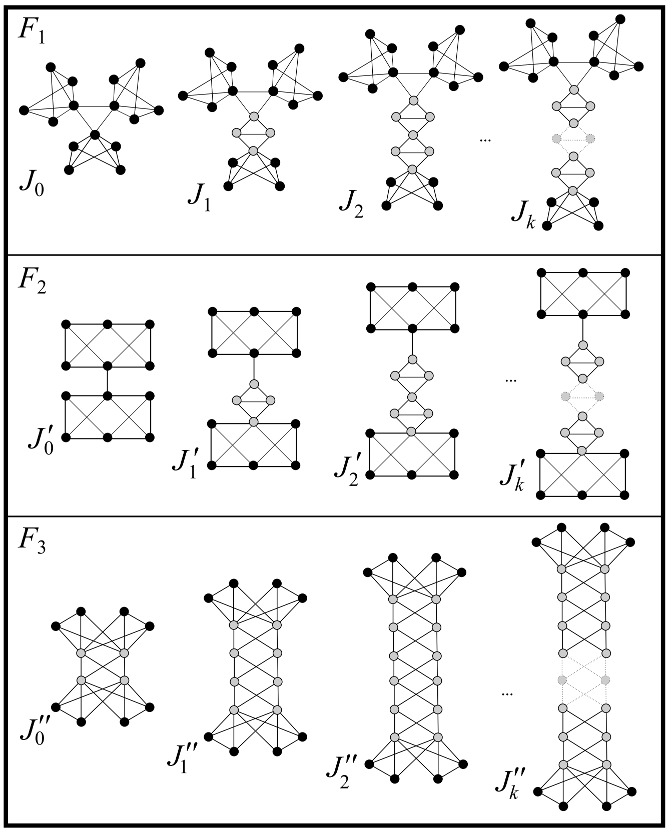


Figure 3. Infinite families of minimal graphs in $\mathcal{DH}$ with no near-bipartition.

Theorem 2. *There is an infinite set of minimal graphs in $\mathcal{DH}$ that have no near-bipartition.*

Notice that from family F_3 in Figure 3, it follows that Theorem 2 holds even restricting to biconnected graphs in $\mathcal{DH}$. Also, observe that there are H_2-free distance-hereditary biconnected graphs that are not near-bipartite, such as for example $I_2+I_2+I_2$ and $K_2+I_2+I_2$ (see Figure 4).

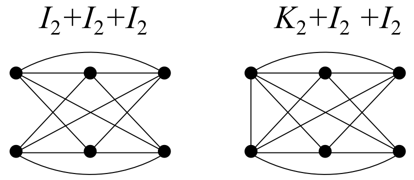


Figure 4. The graphs $I_2+I_2+I_2$, and $K_2+I_2+I_2$.

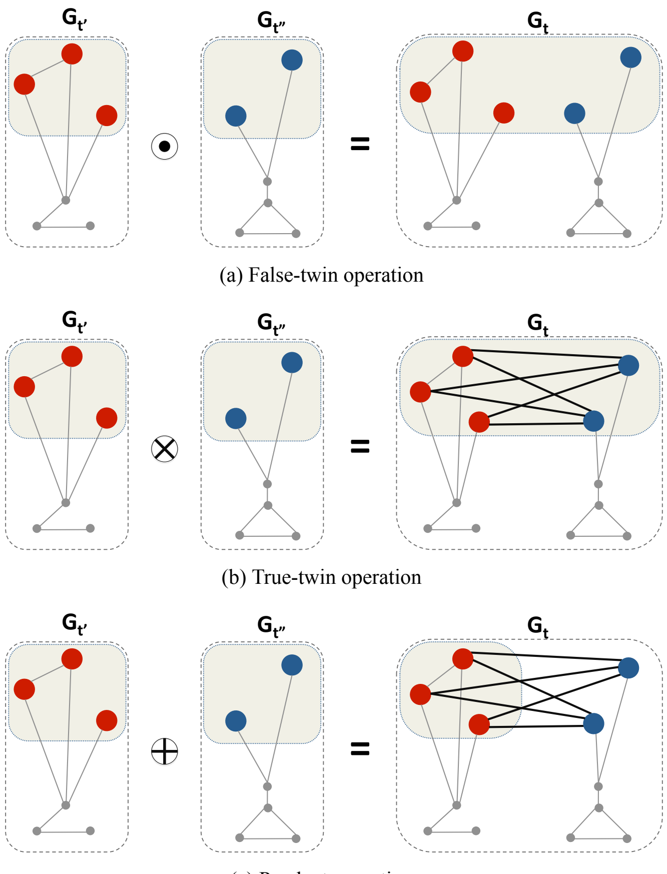
As shown in [Chang *et al.*, 1997], another interesting property of a distance-hereditary graph consist of the following assumptions:

1. a graph with only one vertex v is in $\mathcal{DH}$, and its *twin set* is $\{v\}$;
2. let G_1 and G_2 be in $\mathcal{DH}$ with *twin sets* W_1 and W_2, respectively. Then:
   (a) *true-twin operation* – $\otimes$: the graph obtained from $G_1 \cup G_2$ and with the additional edges from $G_1[W_1]+G_2[W_2]$ is in $\mathcal{DH}$ with *twin set* $W=W_1\cup W_2$;
   (b) *pendant operation* – $\oplus$: the graph obtained from $G_1 \cup G_2$ and with the additional edges from $G_1[W_1]+G_2[W_2]$ is in $\mathcal{DH}$ with *twin set* $W=W_1$;
   (c) *false-twin operation* – $\odot$*:* the graph obtained from $G_1\cup G_2$ in $\mathcal{DH}$ with *twin set* $W=W_1\cup W_2$;
3. Every non-trivial graph $G\in\mathcal{DH}$ can be constructed through successive applications of the operations described above.

In Figure 5, we can see examples of operations for construction/decomposition of distance-hereditary graphs.

For a rooted tree $\mathcal{T}$, let $root(\mathcal{T})$ be the root of $\mathcal{T}$. Every graph $G\in\mathcal{DH}$ admits a decomposition, represented by a binary tree, called an *one-vertex-extension tree* $\mathcal{T}_G$, defined as follows (cf. [Chang *et al.*, 1997]):

- a tree consisting of only one vertex v is a one-vertex-extension tree $\mathcal{T}_G$ of a graph $G=(\{v\},\emptyset)$ with twin set $\{v\}$.
- let $\mathcal{T}_{G_1}$ and $\mathcal{T}_{G_2}$ be one-vertex-extension trees of graphs G_1 and G_2, G_1 with *twin set* W_1 and G_2 with *twin set* W_2.
  1. if G is composed by G_1 and G_2 from the true-twin operation then the tree $\mathcal{T}_G$, in which $root(\mathcal{T}_G)$ is represented by $\otimes$, and $root(\mathcal{T}_{G_1})$ and $root(\mathcal{T}_{G_2})$ are the children of $root(\mathcal{T}_G)$, is a one-vertex-extension tree of G with twin set equal to $W_1\cup W_2$;
  2. if G is composed by G_1 and G_2 from the pendant operation then the tree $\mathcal{T}_G$, in which $root(\mathcal{T}_G)$ is represented by $\oplus$, and $root(\mathcal{T}_{G_1})$ and $root(\mathcal{T}_{G_2})$ are the children of $root(\mathcal{T}_G)$, is a one-vertex-extension tree of G with twin set equal to W_1;



(a) False-twin operation

(b) True-twin operation

(c) Pendant operation

Figure 5. Operations for construction/decomposition of distance-hereditary graphs.

3. if G is composed by G_1 and G_2 from the false-twin operation then the tree $\mathcal{T}_G$, in which $root(\mathcal{T}_G)$ is represented by $\odot$, and $root(\mathcal{T}_{G_1})$ and $root(\mathcal{T}_{G_2})$ are the children of $root(\mathcal{T}_G)$, is a one-vertex-extension tree of G with twin set equal to $W_1 \cup W_2$.

In Figure 6, we can see an example of a one-vertex-extension tree $\mathcal{T}_G$ using those operations, as mentioned earlier, for constructing G. Remark that the one-vertex-extension trees are not always unique for graphs in $\mathcal{DH}$, and the number of operations is exactly the vertex number of G minus one.

Now, we are ready to present a sufficient condition for a distance-hereditary graph to be near-bipartite, which is stronger than that presented in Corollary 3.

Lemma 1. *Let $G \in \mathcal{DH}$ be a (K_4,H_1,H_2)-free graph with one-vertex-extension tree $\mathcal{T}_G$ and twin set W. It holds that if $G[W]$ is acyclic then G has near-bipartitions $(\mathcal{S},\mathcal{F})$ satisfying the following:*

1. *$W \subseteq \mathcal{F}$, if no diamond of G contains two non-adjacent vertices of a same component of $G[W]$. In addition, each vertex of W is in a distinct tree of $\mathcal{F}$, if W is an independent set and no diamond contains two vertices of W.*
2. *$W \subseteq \mathcal{S}$, if W is an idependent set.*
3. *$X \subseteq \mathcal{S}$ and $W \setminus X \subseteq \mathcal{F}$, for any maximal independent set X of $G[W]$.*

Proof. First, we remark that if $G[W]$ is acyclic then for any node of $\mathcal{T}_G$ its twin set is also acyclic. Note that a cycle in a children's twin set is "forgotten" in the parent's twin set only using pendant operation, which implies K_4, H_1, or hole in G.

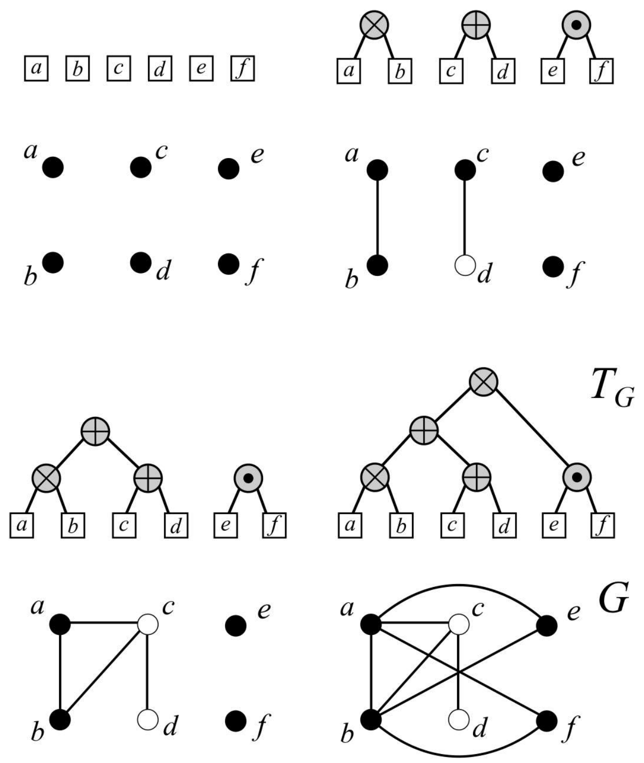


Figure 6. A one-vertex-extension tree T_G for a graph G in $\mathcal{DH}$. First, all vertices are trivial in graphs $(\{a\},\emptyset)$, $(\{b\},\emptyset)$, $(\{c\},\emptyset)$, $(\{d\},\emptyset)$, $(\{e\},\emptyset)$ e $(\{f\},\emptyset)$. In each iteration for obtaining T_G, the twin set is represented by black vertices.

Thus, we are able to prove this lemma by induction on n, the number of vertices of G.

If $V(G) = \{v\}$ $(n = 1)$ then $W = \{v\}$, and $(\emptyset,\{v\})$, $(\{v\},\emptyset)$ are near-bipartitions satisfying the desired conditions.

We assume that the claim holds for any (K_4,H_1,H_2)-free graph $G' \in \mathcal{DH}$ with at most k vertices having one-vertex-extension tree $\mathcal{T}_{G'}$ and twin set W where $G'[W]$ is acyclic.

Let G be a graph with $k+1$ vertices.

If $root(\mathcal{T}_G)$ is a false-twin node with children G' and G'' then, by hypothesis, the claim is valid for G' and G''. Since $(\mathcal{S}' \cup \mathcal{S}'', \mathcal{F}' \cup \mathcal{F}'')$ is a near-bipartition of G whenever $(\mathcal{S}',\mathcal{F}')$ is a near-bipartiton of G' and $(\mathcal{S}'',\mathcal{F}'')$ is a near-bipartition of G'', the claim is also valid for G.

If $root(\mathcal{T}_G)$ is a true-twin node with twin set $W = W' \cup W''$, children G' with twin set W' and G'' with twin set W'', and $G[W]$ is acyclic, then W' or W'' has size one. Let $W' = \{v\}$. By hypothesis, G' has near-bipartitions with v in the stable part and in the forest. Since $G[W]$ is acyclic then W'' is an independent set, and, by hypothesis, there are near-bipartitions of G'' with W'' in the stable part and in the forest as well. From this near-bipartitions of G' and G'' one can form the dsedired near-bipartitions of G, in particular, the near-bipartition containing all W in the forest, case no diamond of G contains two vertices of W'', follows from a near-bipartition of G'' where each vertex of W'' is in a distinct tree, such a near-bipartition exists by hypothesis.

If $root(\mathcal{T}_G)$ is a pendant node with twin set $W = W'$, children G' with twin set W' and G'' with twin set W'', and $G[W]$ is acyclic, then W' or W'' are independent sets (G is K_4-free).

1. If both W' and W'' are independent sets, then from near-bipartitions of G' and G'' where W' is in the forest and W'' is not in the forest (and vice versa), one can obtain the desired partitions for G.

2. If $G[W']$ has more than one edge, then W'' has size one, otherwise G has a H_2 or a H_1 as induced subgraphs. Let $W'' = \{u\}$. In this case, a partition of G' with W' in the forest and a partition of G'' with u in the stable part form a partition of G with $W = W'$ in the forest. We remark that $G[W']$ is P_4-free, otherwise G has an induced gem. Thus, any tree of $G[W']$ is a star (K_2 and K_1 are stars). If there is a diamond of G' containing two leaves of a star of $G[W']$ then this diamond together with the root of this star and the vertex u form an induced H_2 in G. Therefore, such a diamond does not exist and, by hypothesis, there are near-bipartitions of G' with W' in the forest.
On the other hand, to obtain a near-bipartition of G where $X \subseteq \mathcal{S}$ and $W \setminus X \subseteq \mathcal{F}$, for a maximal independent set X of $G[W]$, we need to have a near-bipartition of G'' having $W'' = \{u\}$ in the forest (by hypothesis, such a near-bipartition exists) and a near-bipartition of G' having each vertex of $W' \setminus X$ in a distinct tree of the forest part. The desired property holds if $G[W']$ has only one component, because there is no diamond with two leaves of a star. Now, assume that $G[W']$ has more than one component. Let $\{a, b\} \in W' \setminus X$ be vertices of distinct components of $G[W']$ and a', b' be neighbors of a and b in $G[W']$, respectively. Note that there is no vertex z of G' adjacent to both a and b, otherwise $G[\{z, a, b, a', b', u\}]$ contains a house or a H_2 as an induced subgraph. Finally, there is no path P between a and b in G', otherwise $G[P \cup \{a, a', b', b', u\}]$ contains a hole, a house, or an H_2 as an induced subgraph. Therefore, a near-bipartition of G' having $W' \setminus X$ in the forest part has each vertex of $W' \setminus X$ in a distinct tree. By hypothesis, we have such a near-bipartition for any maximal independent set X of $G'[W']$, so the claim holds.
3. If $G[W']$ has exactly one edge, then from a near-bipartition of G' having W' in the forest and a near-bipartition of G'' with W'' in the stable part one can form a near-bipartition of G with $W = W'$ in the forest. Besides, any maximal independent set of $G[W']$ contains all but one vertex of W'. So, from a near-bipartition with X in the stable part one can form a near-bipartition with $W \setminus X$ in the forest if G'' has a partition with W'' in the forest part where each vertex of W'' is in a distinct tree, by hypothesis such a near-bipartition exists, otherwise there is a diamond with two vertices of W'' and such a diamond with a K_2 of W' form an induced H_2 of G.
4. The case where $G[W'']$ has edges, and W' is an independent set is similar to the two previous cases. By hypothesis, we have near-bipartitions of G' having W' in the stable part and near-bipartitions of G'' having W'' in the forest. In addition, to have a near-bipartition of G with W' in the forest, we need a near-bipartition of G'' having each vertex of $W'' \setminus X$ in a distinct tree of the forest part for some X in the stable part. As discussed previously, it follows that such a near-bipartition exists. Therefore, the claim holds.

This completes the proof. □

Theorem 3. *Let $G \in \mathcal{DH}$ be a (K_4,H_1,H_2)-free graph. It holds that G is a near-bipartite graph.*

Proof. Let $G \in \mathcal{DH}$ be a (K_4,H_1,H_2)-free graph with one-vertex-extension tree $\mathcal{T}_G$, and twin set W. If $G[W]$ is acyclic, by Lemma 1, we are done. Assume that $G[W]$ has cycles.

We denote as a primordial node a node of $\mathcal{T}_G$ with twin set inducing cycles, where none of its descendant nodes in $\mathcal{T}_G$ have a twin set inducing a cycle.

Note that primordial nodes are true-twin nodes, because false-twin nodes do not add edges, and pendant nodes have twin-set equal to the twin set of one of its children.

Note that a primordial node where its twin set induce a cycle does not have ancestor nodes that are pendant or true-twin nodes (otherwise G has induced K_4, H_1, or hole).

A graph in $\mathcal{DH}$ obtained by a false-twin operation between two graphs G' and G'' has a near-bipartition if and only if G' and G'' have near-bipartitions. Therefore, it is enough to analyze the primordial nodes of $\mathcal{T}_G$.

We consider a primordial node as the root of a subtree $\mathcal{T}_{G^*}$, and let G^* be the subgraph associated with $\mathcal{T}_{G^*}$.

Consider $root(\mathcal{T}_{G^*})$ a primordial node (true-twin node) with twin set $W = W' \cup W''$, having children G' with twin set W' and G'' with twin set W''. Then, both W' and W'' are independent sets, or one is independent and the other has only one edge, or one of them has size one and the other induces a forest; otherwise, G contains a K_4, an H_1, or an H_2 as an induced subgraph.

If W' and W'' are independent sets then, by Lemma 1, G' admits a near-bipartition $(\mathcal{S}', \mathcal{F}')$ where $W' \subseteq \mathcal{S}'$, and G'' admits a near-bipartition $(\mathcal{S}'', \mathcal{F}'')$ where $W'' \subseteq \mathcal{F}''$. Therefore, G has a near-bipartition $(\mathcal{S}' \cup \mathcal{S}'', \mathcal{F}' \cup \mathcal{F}'')$.

Now, assume that W' is an independent set, and $G[W'']$ has edges.By Lemma 1, G' admits a near-bipartition $(\mathcal{S}', \mathcal{F}')$ where $W' \subseteq \mathcal{S}'$. To claim that G'' admits a near-bipartition $(\mathcal{S}'', \mathcal{F}'')$ where $W'' \subseteq \mathcal{F}''$ we remark that in G'' there is a no diamond containing two leaves of a star of $G[W'']$, because it forms an induced H_2 with the root of the star and some vertex of W'. Therefore, G has a near-bipartition $(\mathcal{S}' \cup \mathcal{S}'', \mathcal{F}' \cup \mathcal{F}'')$. □

3.1 Dynamic programming using one-vertex-extension trees

Regarding width parameterizations, it is well-known that every problem expressible in MSOL$_1$ can be solved in linear time on graphs with bounded clique-width [Courcelle *et al.*, 2000]. Also, it is easy to see that Near-Bipartitiness, the problem of determining whether the vertex set of a graph can be partitioned into two sets $\mathcal{S}$ and $\mathcal{F}$ such that $\mathcal{S}$ is a stable set and $\mathcal{F}$ induces a forest, can be expressed in such a monadic second-order logic. Therefore, Near-Bipartitiness can be solved in linear time on graphs with bounded clique-width, which includes distance-hereditary graphs [Golumbic and Rotics, 2000]. However, the linear-time algorithms based on the MSOL model-checking framework [Courcelle, 1990] typically do not provide useful algorithms in practice since the dependence on the clique-width involves huge multiplicative constants, even when the clique-width is bounded by three (see [Flum and Grohe, 2006]). As said in [Flum and Grohe, 2006], the size of the automaton grows as fast as a tower of 2's of height linear in the sentence length. To see this, note that for every negation in the sentence, the automaton has to be determinized, which causes an exponential blowup in size [Flum and Grohe, 2006]. Therefore, for Near-Bipartitiness, the constant provided by the model-checking framework would be large even if the clique-width were equal to three. Therefore, the main objective of the following discrete algorithm is not to classify the problem as "easy" (linear-time solvable), but rather to present an explicit and efficient way to implement in practice a resolution to the problem on distance-hereditary graphs. Which obviously must involve low multiplicative constants.

Recall that a one-vertex-extension tree $\mathcal{T}_{G_t}$ of a graph $G_t \in \mathcal{DH}$ has 4 types of nodes (leaf, false-twin, true-twin, pendant). Next, we present dynamic programming based on traversing $\mathcal{T}_{G_t}$ in a bottom-up manner, computing the following *Boolean functions*:

(i) IFVS(G_t): *true* if and only if G_t has an independent feedback vertex set;

(ii) DESC-IFVS(G_t): *true* if and only if G_t has an independent feedback vertex set such that its removal disconnects the remaining vertices of W_t, that is, each vertex of W_t either is removed or it will be in a distinct tree of the resulting forest. When $|W_t| = 1$, DESC-IFVS(G_t) evaluates as IFVS(G_t), i.e., the property of being in a distinct tree is guaranteed;

(iii) FORB-IFVS(G_t): *true* if and only if G_t has an independent feedback vertex set containing no vertex of W_t;

(iv) DESC-FORB-IFVS(G_t): *true* if and only if G_t has an independent feedback vertex set containing no vertex of W_t, such that its removal disconnects all vertices of W_t;

(v) TWINS-DELETED-IFVS(G_t): *true* if and only if G_t has an independent feedback vertex set containing all vertices of W_t;

(vi) TWINS-DELETED-IFVS$^{-1}(G_t)$: *true* if and only if G_t has an independent feedback vertex set containing all vertices of W_t except one vertex $v \in W_t$.

Next, according to each type of node, the computation of each Boolean function will be presented. We will show that for each node of $\mathcal{T}_{G_t}$ to compute each of the six functions, it is sufficient to perform a constant number of queries to previously calculated functions in its children. Thus, given a one-vertex-extension, the problem can be solved in linear time. Clearly all six functions return *true* when G_t is a leaf node. Therefore, it remains to analyse three types of nodes.

⋆ False-twin nodes - $G_t = G_{t'} \odot G_{t''}$

In a false-twin node, there is no formation of cycles, so, the recurrences only care with subproblems in each subgraph. Therefore:

1) IFVS(G_t) = IFVS($G_{t'}$) $\wedge$ IFVS($G_{t''}$)

2) DESC-IFVS(G_t) = DESC-IFVS($G_{t'}$) $\wedge$ DESC-IFVS($G_{t''}$)

3) FORB-IFVS(G_t) =

FORB-IFVS($G_{t'}$) $\wedge$ FORB-IFVS($G_{t''}$)

4) DESC-FORB-IFVS(G_t) =

DESC-FORB-IFVS($G_{t'}$) $\wedge$ DESC-FORB-IFVS($G_{t''}$)

5) TWINS-DELETED-IFVS(G_t) =

TWINS-DELETED-IFVS($G_{t'}$) $\wedge$ TWINS-DELETED-IFVS($G_{t''}$);

6) TWINS-DELETED-IFVS$^{-1}(G_t)$ =

[TWINS-DELETED-IFVS$^{-1}(G_{t'})$ $\wedge$ TWINS-DELETED-IFVS($G_{t''}$)]

$\vee$

[TWINS-DELETED-IFVS($G_{t'}$) $\wedge$ TWINS-DELETED-IFVS$^{-1}(G_{t''})$]

⋆ True-twin nodes - $G_t = G_{t'} \otimes G_{t''}$

In a true-twin node, due to the existence of a join between the vertices of $W_{t'}$ and $W_{t''}$, it is necessary to adapt the recurrences so that the cycles formed by the join are also broken. We will define the recurrences according to subcases because the *join* interferes differently according to the characteristics below:

(a) Trivial: $|W_{t'}| = |W_{t''}| = 1$;

(b) Star: $|W_{t'}| = 1$ and $|W_{t''}| \geq 2$ or $|W_{t'}| \geq 2$ and $|W_{t''}| = 1$;

(c) Bipartite: $|W_{t'}| \geq 2$, $|W_{t''}| \geq 2$ and $W_{t'}$, $W_{t''}$ are both stable sets;

(d) General: $|W_{t'}| \geq 2$, $|W_{t''}| \geq 2$ and $W_{t'}$ or $W_{t''}$ is not a stable set.

Figure 7 illustrates the possible cases to be considered.

The trivial case

In the trivial case, there is no formation of cycles between the vertices of the twin sets, so it is only necessary to prevent both vertices of W_t from being selected for a independent feedback vertex set (solution), given the property of being independent/stable.

(1) IFVS(G_t) =

[IFVS($G_{t'}$) $\wedge$ FORB-IFVS($G_{t''}$)]

$\vee$

[FORB-IFVS($G_{t'}$) $\wedge$ IFVS($G_{t''}$)]

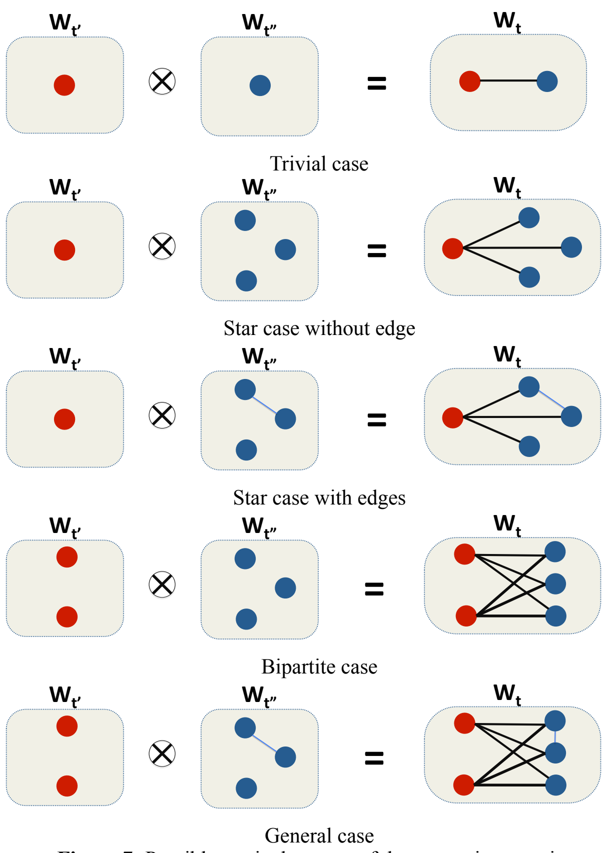


Figure 7. Possible particular cases of the true-twin operation.

(2) DESC-IFVS(G_t) =

[FORB-IFVS($G_{t'}$) $\wedge$ TWINS-DELETED-IFVS($G_{t''}$)]

$\vee$

[TWINS-DELETED-IFVS($G_{t'}$) $\wedge$ FORB-IFVS($G_{t''}$)]

(3) FORB-IFVS(G_t) = FORB-IFVS($G_{t'}$) $\wedge$ FORB-IFVS($G_{t''}$)

(4) DESC-FORB-IFVS(G_t) = *false.*

(5) TWINS-DELETED-IFVS(G_t) = *false.*

(6) TWINS-DELETED-IFVS^{-1}(G_t) =

[FORB-IFVS($G_{t'}$) $\wedge$ TWINS-DELETED-IFVS($G_{t''}$)]

$\vee$

[TWINS-DELETED-IFVS($G_{t'}$) $\wedge$ FORB-IFVS($G_{t''}$)]

The star case

Here, the join between the twin sets creates cycles when there is an edge in the set of twins with more than one vertex, or when there is a path between a pair of twins of the same subgraph. Without loss of generality, we assume $|W_{t'}| = 1$. Thus:

(1) IFVS(G_t) =
[TWINS-DELETED-IFVS($G_{t'}$) $\wedge$ FORB-IFVS($G_{t''}$)]
$\vee$
[FORB-IFVS($G_{t'}$) $\wedge$ DESC-IFVS($G_{t''}$)]

(2) DESC-IFVS(G_t) = *false* if $W_{t''}$ induces at least one edge; otherwise:
[TWINS-DELETED-IFVS($G_{t'}$) $\wedge$ DESC-FORB-IFVS($G_{t''}$)]
$\vee$
[FORB-IFVS($G_{t'}$) $\wedge$ TWINS-DELETED-IFVS($G_{t''}$)]

(3) FORB-IFVS(G_t) =
FORB-IFVS($G_{t'}$) $\wedge$ DESC-FORB-IFVS($G_{t''}$)

(4) DESC-FORB-IFVS(G_t) = *false.*

(5) TWINS-DELETED-IFVS(G_t) = *false.*

(6) TWINS-DELETED-IFVS^{-1}(G_t) =
FORB-IFVS($G_{t'}$) $\wedge$ TWINS-DELETED-IFVS($G_{t''}$)

Note that the vertex that must stay in the forest (if any) must be in $W_{t'}$.

The bipartite case

To eliminate the cycles generated by the join between the pivots, we can only remove vertices from one of the sets of pivots to compose the IFVS. And in order not to leave at least two vertices on each side, the solution (if any) uses one of the pivot sets integrally or uses all of them except for only one vertex of the chosen set. In this second case, we have one more concern; we need to handle the cycles formed as in the previous section.

(1) IFVS(G_t) =
[TWINS-DELETED-IFVS($G_{t'}$) $\wedge$ FORB-IFVS($G_{t''}$)]
$\vee$
[TWINS-DELETED-IFVS^{-1}(G_t) $\wedge$ DESC-FORB-IFVS($G_{t''}$)]
$\vee$
[FORB-IFVS($G_{t'}$) $\wedge$ TWINS-DELETED-IFVS($G_{t''}$)]
$\vee$
[DESC-FORB-IFVS($G_{t'}$) $\wedge$ TWINS-DELETED-IFVS^{-1}(G_t)]

(2) DESC-IFVS(G_t) =
[TWINS-DELETED-IFVS($G_{t'}$) $\wedge$ DESC-FORB-IFVS($G_{t''}$)]
$\vee$
[DESC-FORB-IFVS($G_{t'}$) $\wedge$ TWINS-DELETED-IFVS($G_{t''}$)]

Note that in this case, all other functions must return *false*.

The general case

Note that only one side induces an edge; otherwise, G_t has a K_4. Without loss of generality, we will assume that $W_{t'}$ is a stable set.

(1) IFVS(G_t) =
[TWINS-DELETED-IFVS($G_{t'}$) $\wedge$ FORB-IFVS($G_{t''}$)]
$\vee$
[DESC-FORB-IFVS($G_{t'}$) $\wedge$
TWINS-DELETED-IFVS^{-1}(G_t)]

The possibility of using the entire $W_{t''}$, in this case, is impracticable due existence of an edge in $G[W_{t''}]$.

Finally, all other functions must return *false*, which completes our case analysis for the true-twin nodes.

$\star$ Pendant nodes - $G_t = G_{t'} \oplus G_{t''}$

This case is quite similar to the previous one; however, the fact that the resulting set of twins is only $W_{t'}$, implies in handling some particularities.

Figure 8 illustrates all the possible particular cases of the pendant operation.

The trivial case

This computation must differ from the true-twin case because now $W_t = W_{t'}$.

(1) IFVS(G_t) =
[IFVS($G_{t'}$) $\wedge$ FORB-IFVS($G_{t''}$)]
$\vee$
[FORB-IFVS($G_{t'}$) $\wedge$ IFVS($G_{t''}$)]

(2) DESC-IFVS(G_t) = IFVS(G_t)

This case is equal to IFVS(G_t) because the resulting twin set $W_t = W_{t'}$ has size one and satisfies the property of being in a distinct component, by definition.

(3) FORB-IFVS(G_t) = [FORB-IFVS($G_{t'}$) $\wedge$ IFVS($G_{t''}$)]

(4) DESC-FORB-IFVS(G_t) = FORB-IFVS(G_t)

This case equals FORB-IFVS(G_t) because $|W_t| = 1$ and it is disconnected by definition.

(5) TWINS-DELETED-IFVS(G_t) =
[TWINS-DELETED-IFVS($G_{t'}$) $\wedge$ FORB-IFVS($G_{t''}$)]

(6) TWINS-DELETED-IFVS^{-1}(G_t) = FORB-IFVS(G_t)
Since W_t contains only one vertex, it should not be part of the solution, so it equals PROIB-IFVS(G_t).

The star case

Here, the recurrence formulas change according to which twin set contains only one vertex.

Case A: $|W_{t'}| = 1$ **e** $|W_{t''}| > 1$

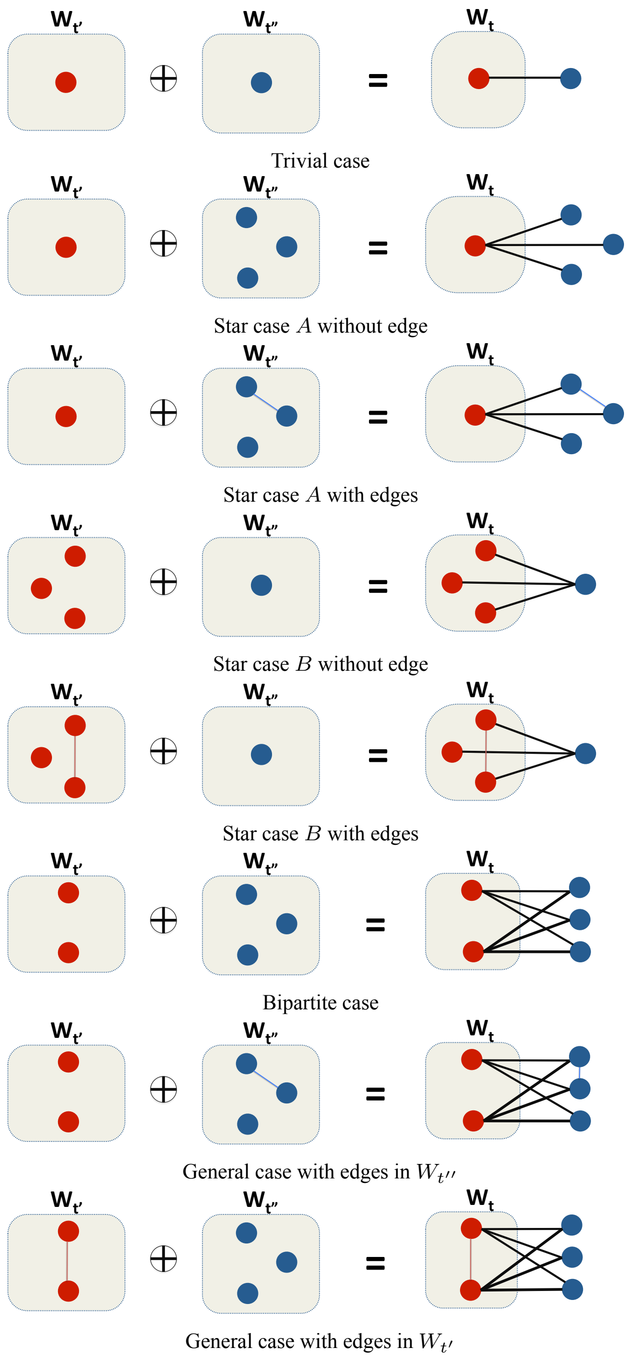


Figure 8. Possible particular cases of the pendant operation.

(1) IFVS(G_t) =

[TWINS-DELETED-IFVS($G_{t'}$) $\wedge$ FORB-IFVS($G_{t''}$)]
$\vee$
[FORB-IFVS($G_{t'}$) $\wedge$ DESC-IFVS($G_{t''}$)]

This recurrence is the same as that defined in true-twin case.

(2) DESC-IFVS(G_t) = IFVS(G_t)

For the same reason as the trivial case.

(3) FORB-IFVS(G_t) = [FORB-IFVS($G_{t'}$) $\wedge$ DESC-IFVS($G_{t''}$)]

(4) DESC-FORB-IFVS(G_t) = FORB-IFVS(G_t)

(5) TWINS-DELETED-IFVS(G_t) =

[TWINS-DELETED-IFVS($G_{t'}$) $\wedge$ FORB-IFVS($G_{t''}$)]

(6) TWINS-DELETED-IFVS$^{-1}(G_t)$ =

[FORB-IFVS($G_{t'}$) $\wedge$ DESC-IFVS($G_{t''}$)]

Caso B: $|W_{t'}| > 1$ **e** $|W_{t''}| = 1$

(1) IFVS(G_t) =

[DESC-IFVS($G_{t'}$) $\wedge$ FORB-IFVS($G_{t''}$)]
$\vee$
[FORB-IFVS($G_{t'}$) $\wedge$ TWINS-DELETED-IFVS($G_{t''}$)]

(2) DESC-IFVS(G_t) =

[PROIB-DESC-IFVS($G_{t'}$) $\wedge$ TWINS-DELETED-IFVS($G_{t''}$)]
$\vee$
[TWINS-DELETED-IFVS($G_{t'}$) $\wedge$ FORB-IFVS($G_{t''}$)]
$\vee$
[TWINS-DELETED-IFVS$^{-1}(G_{t'})$ $\wedge$ FORB-IFVS($G_{t''}$)]

The first component considers the use of $W_{t''}$ in the solution. If the vertex in $W_{t''}$ is not used, then at most, one vertex in $W_{t'}$ should stay in the forest (if any); otherwise the property of being disconnected will be violated.

(3) FORB-IFVS(G_t) =

[FORB-IFVS($G_{t'}$) $\wedge$ TWINS-DELETED-IFVS($G_{t''}$)]
$\vee$
[FORB-DESC-IFVS($G_{t'}$) $\wedge$ FORB-IFVS($G_{t''}$)]

Note that if $W_{t'}$ induces edges then FORB-DESC-IFVS($G_{t'}$) will be false and the answer will depend on the first queries.

(4) FORB-DESC-IFVS(G_t) is *false* if $W_{t'}$ induces edges, otherwise

[FORB-DESC-IFVS($G_{t'}$) $\wedge$ TWINS-DELETED-IFVS($G_{t''}$)]

(5) TWINS-DELETED-IFVS(G_t) is *false* if $W_{t'}$ induces edges, otherwise

[TWINS-DELETED-IFVS($G_{t'}$) $\wedge$ FORB-IFVS($G_{t''}$)]

(6) TWINS-DELETED-IFVS$^{-1}(G_t)$ =

[TWINS-DELETED-IFVS$^{-1}(G_{t'})$ $\wedge$ FORB-IFVS($G_{t''}$)]

Note that a necessary condition for TWINS-DELETED-IFVS$^{-1}(G_{t'})$ be equal to *true* is $G[W_{t'}]$ to have a vertex cover of size one.

The bipartite case

(1) IFVS(G_t) is computed using exactly the same queries as the analogous case described for true-twins.

(2) DESC-IFVS(G_t) =

[TWINS-DELETED-IFVS($G_{t'}$) $\wedge$ FORB-IFVS($G_{t''}$)]
$\vee$
[TWINS-DELETED-IFVS$^{-1}(G_{t'})$ $\wedge$ DESC-FORB-IFVS($G_{t''}$)]
$\vee$
[FORB-DESC-IFVS($G_{t'}$) $\wedge$ TWINS-DELETED-IFVS($G_{t''}$)]

Since we want a disconnected solution, query TWINS-DELETED-IFVS$^{-1}(G_{t''})$ does not apply.

(3) FORB-IFVS(G_t) =

[FORB-IFVS($G_{t'}$) $\wedge$ TWINS-DELETED($G_{t''}$)]
$\vee$
[FORB-DESC-IFVS($G_{t'}$) $\wedge$ TWINS-DELETED$^{-1}(G_{t'})$]

(4) FORB-DESC-IFVS(G_t) =

[FORB-DESC-IFVS($G_{t'}$) $\wedge$ TWINS-DELETED-IFVS($G_{t''}$)]

(5) TWINS-DELETED-IFVS(G_t) =

[TWINS-DELETED-IFVS($G_{t'}$) $\wedge$ FORB-IFVS($G_{t''}$)]

(6) TWINS-DELETED-IFVS$^{-1}(G_t)$ =

[TWINS-DELETED-IFVS$^{-1}(G_t)$ $\wedge$ DESC-FORB-IFVS($G_{t''}$)]

The general case

(1) IFVS(G_t) =

If $W_{t'}$ is stable and $W_{t''}$ induces edges,

[TWINS-DELETED-IFVS($G_{t'}$) $\wedge$ FORB-IFVS($G_{t''}$)]
$\vee$
[DESC-FORB-IFVS($G_{t'}$) $\wedge$ TWINS-DELETED-IFVS$^{-1}(G_t)$]

Note that the fact that there is an edge in $W_{t''}$ makes unfeasible the case in which one vertex of $W_{t'}$ is left in the forest.

If $W_{t'}$ induces edges and $W_{t''}$ is stable,

[FORB-IFVS($G_{t'}$) $\wedge$ TWINS-DELETED-IFVS($G_{t''}$)]
$\vee$
[TWINS-DELETED-IFVS$^{-1}(G_t)$ $\wedge$
DESC-FORB-IFVS($G_{t''}$)]

This case is similar to the previous one, inverting the recurrences used in each node $G_{t'}$ and $G_{t''}$, due to the presence of edges in $W_{t'}$.

(2) DESC-IFVS(G_t) =

If $W_{t'}$ is stable and $W_{t''}$ induces edges,

[TWINS-DELETED-IFVS($G_{t'}$) $\wedge$ FORB-IFVS($G_{t''}$)]

If $W_{t'}$ induces edges and $W_{t''}$ is stable,

[TWINS-DELETED-IFVS$^{-1}(G_t)$ $\wedge$
DESC-FORB-IFVS($G_{t''}$)]

(3) FORB-IFVS(G_t) =

If $W_{t'}$ is stable and $W_{t''}$ induces edges,

[DESC-FORB-IFVS($G_{t'}$) $\wedge$
TWINS-DELETED-IFVS$^{-1}(G_t)$]

If $W_{t'}$ induces edges and $W_{t''}$ is stable,

[FORB-IFVS($G_{t'}$) $\wedge$ TWINS-DELETED-IFVS($G_{t''}$)]

(4) FORB-DESC-IFVS(G_t) = *false*

If $W_{t''}$ induces edges, at least one vertex of $W_{t''}$ must remain in the forest. Thus, the property of being disconnected will be violated. If $W_{t''}$ induces edges, we will have a forbidden edge, which would again make the property of being disconnected be violated.

(5) TWINS-DELETED-IFVS(G_t) is *false* if $W_{t'}$ induces edges, otherwise

[TWINS-DELETED-IFVS($G_{t'}$) $\wedge$ FORB-IFVS($G_{t''}$)]

(6) TWINS-DELETED-IFVS$^{-1}(G_t)$ is *false* if $W_{t'}$ is stable and $W_{t''}$ induces edges (the remaining vertices would induce a triangle). Otherwise,

[TWINS-DELETED-IFVS$^{-1}(G_t)$ $\wedge$
FORB-DESC-IFVS($G_{t''}$)]

By separating the computation of each function in cases follows that the correctness of each recurrence is straightforward. Since a one-vertex-extension tree of a $G \in \mathcal{DH}$ can be computed in $O(n+m)$ Chang *et al.* [1997], the following theorem holds.

Theorem 4. *There is a linear-time dynamic programming to solve* Near-Bipartiteness *on* $\mathcal{DH}$.

4 Concluding remarks

Although there are infinite families of minimal H_1-free graphs as well as H_2-free graphs in $\mathcal{DH}$ having no near-bipartition (see Figure 3), we show that a (H_1, H_2)-free distance-hereditary graph is near-bipartite if and only if it does not contain a K_4 as subgraph.

Also, we present a simple linear-time dynamic programming to solve Near-Bipartiteness on $\mathcal{DH}$. Recall that simpler algorithms manifest a better understanding of the problem and are more likely to be implemented and trusted by practitioners; they are more easily taught and are more likely to be included in algorithms textbooks; and they attract a broader set of researchers to difficult algorithmic problems. Therefore, our simple algorithm is pertinent even though Near-Bipartitiness is MSOL$_1$-expressible and distance-hereditary graphs have clique-width at most three.

Regarding distance-hereditary graphs, we leave open a full characterization of the infinite families of minimal obstructions for a graph in $\mathcal{DH}$ to be near-bipartite.

Acknowledgements

We thank all colleagues and collaborators who, through discussions, insights, and suggestions, contributed to the improvement of this work. We also thank our home institutions for the technical and administrative support provided during the development of this study.

Funding

This work was financially supported by the Conselho Nacional de Desenvolvimento Científico e Tecnológico (CNPq) and the Coordenação de Aperfeiçoamento de Pessoal de Nível Superior (CAPES). We extend our gratitude to these funding agencies for enabling the execution and completion of this research, contributing directly to the advancement of knowledge in Graph Theory and Computational Complexity.

Competing interests

The authors declare that they have no competing interests regarding the publication of this article.

Authors' Contributions

This work makes an important contribution to Graph Theory by addressing the Near-Bipartiteness problem in distance-hereditary graphs, a significant subclass of perfect graphs and a subject of interest in several research studies. Given that the problem is NP-complete even for perfect graphs, our objective was to explore the conditions under which a distance-hereditary graph can be a near-bipartite graph, meaning it admits a near-bipartition.
Our results include identifying an infinite set of minimal forbidden subgraphs that prevent a distance-hereditary graph from being near-bipartite, as well as a finite set of forbidden subgraphs that provides a sufficient condition for the existence of a near-bipartition in this class. Additionally, using one-vertex-extension trees, we developed a linear-time algorithm that solves the Near-Bipartiteness problem in distance-hereditary graphs. This approach improves theoretical knowledge about near-bipartite graphs and provides a linear-time algorithm, establishing a tool with both theoretical and practical impact.

References

Agrawal, A., Gupta, S., Saurabh, S., and Sharma, R. (2017). Improved algorithms and combinatorial bounds for independent feedback vertex set. In *11th International Symposium on Parameterized and Exact Computation (IPEC 2016)*. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik. DOI: 10.4230/LIPIcs.IPEC.2016.2.

Bandelt, H.-J. and Mulder, H. M. (1986). Distance-hereditary graphs. *Journal of Combinatorial Theory, Series B*, 41(2):182–208. DOI: 10.1016/0095-8956(86)90043-2.

Bonamy, M., Dabrowski, K. K., Feghali, C., Johnson, M., and Paulusma, D. (2017). Recognizing graphs close to bipartite graphs. In *42nd International Symposium on Mathematical Foundations of Computer Science (MFCS 2017)*. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik. DOI: 10.4230/LIPIcs.MFCS.2017.70.

Bonamy, M., Dabrowski, K. K., Feghali, C., Johnson, M., and Paulusma, D. (2018). Independent feedback vertex sets for graphs of bounded diameter. *Information Processing Letters*, 131:26–32. DOI: 10.1016/j.ipl.2017.11.004.

Bonamy, M., Dabrowski, K. K., Feghali, C., Johnson, M., and Paulusma, D. (2019). Independent feedback vertex set for p_5-free graphs. *Algorithmica*, 81(4):1342–1369. DOI: 10.1007/s00453-018-0474-x.

Bondy, J. A., Murty, U. S. R., *et al.* (1976). *Graph theory with applications*, volume 290. Macmillan London. Book.

Brandstädt, A., Brito, S., Klein, S., Nogueira, L. T., and Protti, F. (2013). Cycle transversals in perfect graphs and cographs. *Theoretical Computer Science*, 469:15–23. DOI: 10.1016/j.tcs.2012.10.030.

Bravo, R., Oliveira, R., da Silva, F., and Souza, U. S. (2023). Partitioning p4-tidy graphs into a stable set and a forest. *Discrete Applied Mathematics*, 338:22–29. DOI: 10.1016/j.dam.2023.05.016.

Chang, M.-S., Hsieh, S.-Y., and Chen, G.-H. (1997). Dynamic programming on distance-hereditary graphs. In *International Symposium on Algorithms and Computation*, pages 344–353. Springer. DOI: 10.1007/3-540-63890-3_37.

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

Courcelle, B., Makowsky, J., and Rotics, U. (2000). Linear time solvable optimization problems on graphs of bounded clique-width. *Theory of Computing Systems*, 33(2):125–150. DOI: 10.1007/s002249910009.

da Cruz, M. L. L., Bravo, R. S. F., Oliveira, R., and Souza, U. S. (2023). Near-bipartiteness, connected near-bipartiteness, independent feedback vertex set and acyclic vertex cover on graphs having small dominating sets. In Wu, W. and Guo, J., editors, *Combinatorial Optimization and Applications - 17th International Conference, COCOA 2023, Hawaii, HI, USA, December 15-17, 2023, Proceedings, Part I*, volume 14461 of *Lecture Notes in Computer Science*, pages 82–93. Springer. DOI: 10.1007/978-3-031-49611-0_6.

Dross, F., Montassier, M., and Pinlou, A. (2017). Partitioning a triangle-free planar graph into a forest and a forest of bounded degree. *European Journal of Combinatorics*, 66:81–94. DOI: 10.1016/j.ejc.2017.06.014.

Flum, J. and Grohe, M. (2006). Parameterized complexity theory. *Texts Theoret. Comput. Sci. EATCS Ser.* DOI: 10.1007/3-540-29953-X.

Golumbic, M. and Rotics, U. (2000). On the clique-width of some perfect graph classes. *International Journal of Foundations of Computer Science*, 11(3):423–443. DOI: 10.1142/S0129054100000260.

Grötschel, M., Lovász, L., and Schrijver, A. (1984). Polynomial algorithms for perfect graphs. *Ann. Discrete Math*, 21:325–356. DOI: 10.1016/S0304-0208(08)72943-8.

Howorka, E. (1977). A characterization of distance-hereditary graphs. *The quarterly journal of mathematics*, 28(4):417–420. DOI: 10.1093/qmath/28.4.417.

Karp, R. M. (1972). Reducibility among combinatorial problems. In *Complexity of computer computations*, pages 85–103. Springer. DOI: $10.1007/978-1-4684-2001-2_9$.

Misra, N., Philip, G., Raman, V., and Saurabh, S. (2012). On parameterized independent feedback vertex set. *Theoretical Computer Science*, 461:65–75. DOI: 10.1016/j.tcs.2012.02.012.

Smith, G. and Walford, R. (1975). The identification of a minimal feedback vertex set of a directed graph. *IEEE Transactions on Circuits and Systems*, 22(1):9–15. DOI: 10.1109/TCS.1975.1083961.