To appear in the *International Journal of Computer Mathematics*
Vol. 00, No. 00, February 2013, 1–24

RESEARCH ARTICLE

Clique-perfectness and balancedness of some graph classes

Flavia Bonomo[a], Guillermo Durán[b], Martín D. Safe[c∗] and Annegret K. Wagler[d]

[a]*CONICET and Departamento de Computación, FCEN, Universidad de Buenos Aires, Argentina*; [b]*CONICET, Instituto de Cálculo, and Departamento de Matemática, FCEN, Universidad de Buenos Aires, Argentina and Departamento de Ingeniería Industrial, FCFM, Universidad de Chile, Chile*; [c]*Instituto de Ciencias, Universidad Nacional de General Sarmiento, Argentina*; [d]*CNRS and LIMOS, UFR Sciences et Techniques, Université Blaise Pascal, France*

(*Received 00 Month 201X; final version received 00 Month 201X*)

A graph is clique-perfect if the maximum size of a clique-independent set (a set of pairwise disjoint maximal cliques) and the minimum size of a clique-transversal set (a set of vertices meeting every maximal clique) coincide for each induced subgraph. A graph is balanced if its clique-matrix contains no square submatrix of odd size with exactly two ones per row and column. In this work, we give linear-time recognition algorithms and minimal forbidden induced subgraph characterizations of clique-perfectness and balancedness of P_4-tidy graphs and a linear-time algorithm for computing a maximum clique-independent set and a minimum clique-transversal set for any P_4-tidy graph. We also give a minimal forbidden induced subgraph characterization and a linear-time recognition algorithm for balancedness of paw-free graphs. Finally, we show that clique-perfectness of diamond-free graphs can be decided in polynomial time by showing that a diamond-free graph is clique-perfect if and only if it is balanced.

Keywords: balanced graphs; clique-perfect graphs; diamond-free graphs; P_4-tidy graphs; paw-free graphs; recognition algorithms

AMS Subject Classification: 05C17; 05C75; 05C85

1. Introduction

Perfect graphs were defined by Berge by requiring equality in the min-max type inequality between two graph parameters: the *clique number* and the *chromatic number*. The *clique number* $\omega(G)$ of a graph G is the largest size of a clique in G and the *chromatic number* $\chi(G)$ is the minimum number of colors needed to color the vertices of G in such a way that any two adjacent vertices receive different colors. Clearly, the min-max type inequality $\omega(G) \leq \chi(G)$ holds for every graph G. A graph is *perfect* [3] if the clique number and the chromatic number coincide for each induced subgraph. It is not difficult to see that a perfect graph contains neither induced cycles C_{2k+1} for $k \geq 2$, termed *odd holes*, nor their complements, called *odd antiholes*. In 1961, Berge [3] formulated the Strong Perfect Graph Conjecture (SPGC) asserting that a graph is perfect if and only if it has neither odd holes nor odd antiholes. The proof of the validity of the SPGC took around 40 years along which the SPGC was proved to be true for graphs belonging to many

∗Corresponding author. Email addresses: fbonomo@dc.uba.ar (F. Bonomo), gduran@dm.uba.ar (G. Durán), msafe@ungs.edu.ar (M.D. Safe), and wagler@isima.fr (A.K. Wagler) different graph classes. Early examples include [56], [59], and [62], where the SPGC was verified for claw-free graphs, cographs, and planar graphs. The SPGC was proved to be true for P_4-tidy graphs, paw-free graphs, and diamond-free graphs in [44], [54], and [63], respectively. The SPGC was shown to hold in its full generality in [22] and is now known as the Strong Perfect Graph Theorem. The complexity of the perfect graph recognition problem remained open until almost the same time as the SPGC. It was finally in [21] that a polynomial-time recognition algorithm for perfect graphs was given.

Perfect graphs were also characterized in polyhedral terms. A *clique-matrix* of a graph is a maximal clique vs. vertex incidence matrix; i.e., a clique matrix $A = (a_{ij})$ of a graph G is a $0, 1$-matrix having one row for each maximal clique of G and one column for each vertex of G and such that $a_{ij} = 1$ if and only if the vertex corresponding to the jth column belongs to the maximal clique corresponding to the ith row. The clique-matrix of a graph is unique up to permutation of rows and columns. It was shown in [23] that a graph is perfect precisely when its clique-matrix is perfect, where a $0, 1$-matrix A is *perfect* if the fractional set packing polytope $P(A) = \{x \in \mathbb{R}^n_+ \colon Ax \leq \mathbf{1}\}$ has only integral extreme points.

Theorem 1.1 ([23])**.** *A graph is perfect if and only if it has a perfect clique-matrix.*

The class of clique-perfect graphs was also defined by requiring equality in a min-max type inequality between two graph parameters: the *clique-independence number* and the *clique-transversal number*. The *clique-independence number* $\alpha_{\mathrm{c}}(G)$ of a graph G is the maximum size of a *clique-independent set* (a set of pairwise disjoint maximal cliques) and the *clique-transversal number* $\tau_{\mathrm{c}}(G)$ is the minimum size of a *clique-transversal set* (a set of vertices meeting every maximal clique). Clearly, these two graph parameters satisfy the min-max type inequality $\alpha_{\mathrm{c}}(G) \leq \tau_{\mathrm{c}}(G)$ for every graph G. A graph G is *clique-perfect* if $\alpha_{\mathrm{c}}(H) = \tau_{\mathrm{c}}(H)$ for each induced subgraph H of G. The name 'clique-perfect' was coined by Guruswami and Pandu Rangan [46] but the equality between these two parameters was implicitly studied since long before; e.g., in [6] (see Theorem 1.3 ahead). The class of clique-perfect graphs is neither a subclass nor a superclass of the class of perfect graphs [11]. Nevertheless, some graph classes are known to be clique-perfect. For instance, bipartite graphs are clique-perfect by virtue of the celebrated König–Egerváry Theorem given below. Recall that a graph is *bipartite* if its vertex set can be partitioned into two (possibly empty) stable sets, a *matching* is a set of pairwise vertex-disjoint edges, and a *vertex cover* of a graph is a set of vertices meeting every edge of the graph.

Theorem 1.2 (König–Egerváry Theorem [36, 50])**.** *For each bipartite graph, the maximum size of a matching equals the minimum size of a vertex cover.*

Notice that for each bipartite graph G, $\alpha_{\mathrm{c}}(G)$ coincides with the maximum size of a matching plus the number of isolated vertices, while $\tau_{\mathrm{c}}(G)$ coincides with the minimum size of a vertex cover plus the number of isolated vertices. (A vertex is *isolated* if there is no edge incident to it.) Hence, since the class of bipartite graphs is *hereditary* (i.e., closed under taking induced subgraphs), the König–Egerváry Theorem is equivalent to the fact that bipartite graphs are clique-perfect. Further examples of subclasses of the class of clique-perfect graphs are the classes of comparability graphs [1], complements of forests [7], and distance-hereditary graphs [51]. Unlike perfect graphs, the problems of characterizing clique-perfect graphs by forbidden induced subgraphs and determining the computational complexity of recognizing clique-perfect graphs are still unsolved. Some partial results in this direction were obtained in [8, 9, 13, 15, 52], where necessary and sufficient conditions for a graph G to be clique-perfect in terms of forbidden induced subgraphs as well as polynomial-time algorithms for deciding whether a given graph G is clique-perfect were found when restricting G to belong to one of several different graph classes. Interestingly, the problems of determining $\alpha_{\mathrm{c}}(G)$ and $\tau_{\mathrm{c}}(G)$ are both NP-hard even if G is a split graph [19] and determining $\tau_c(G)$ is NP-hard even if G is a triangle-free graph [37]. More NP-hardness results of this type for α_c and τ_c were proved in [46]. Some polynomial-time algorithms for determining $\alpha_c(G)$ and $\tau_c(G)$ when G belongs to one of several different graph classes were devised in [1, 17, 19, 31–34, 46, 51].

As noticed in the preceding paragraph, the König–Egerváry Theorem can be restated by saying that bipartite graphs are clique-perfect. Berge and Las Vergnas [6] generalized this result by defining a superclass of bipartite graphs, known as the class of *balanced graphs*, and proving that it consists of clique-perfect graphs only. In order to define balanced graphs, we need to define the notion of balanced matrices. A $0,1$-matrix is *balanced* [4] if it contains no square submatrix of odd size with exactly two ones per row and per column or, equivalently, if it contains no edge vs. vertex incidence matrix of a chordless odd cycle as a submatrix. Balanced matrices are well known in polyhedral combinatorics because they lead to nice formulations for both set packing and set covering problems [4, 41]. Precisely, a $0,1$-matrix is balanced if and only if each of its submatrices is perfect or, equivalently, if each of its submatrices is ideal, where a $0,1$-matrix is *ideal* if the fractional set covering polyhedron $Q(A) = \{x \in \mathbb{R}^n_+ \colon Ax \geq \mathbf{1}\}$ has only integral extreme points.

A graph is *balanced* if it has a balanced clique-matrix. Since balanced matrices are perfect, it follows from Theorem 1.1 that the class of balanced graphs is a subclass of the class of perfect graphs. Moreover, balanced graphs are also clique-perfect, as follows from the aforesaid generalization of König–Egerváry Theorem due to Berge and Las Vergnas, which we now state explicitly.

Theorem 1.3 ([6]). *For each balanced graph G, $\alpha_c(G) = \tau_c(G)$.*

Since the class of balanced graphs is hereditary, the above result implies that balanced graphs are clique-perfect. Moreover, a maximum clique-independent set and a minimum clique-transversal set of any given balanced graph can be found in polynomial time by linear programming [31, 32]. See Figure 1 for a diagram of the inclusions among balanced and clique-perfect graphs with some other graph classes, including perfect and bipartite graphs. (Cographs and trivially perfect graphs will be defined later in Section 2.)

The name 'balanced graphs' appeared first in [5], but these graphs were already considered by Berge in [4]; for instance, he stated the following.

Theorem 1.4 ([4]). *A graph G is balanced if and only if every odd cycle in G contains at least one edge with the property that every maximal clique containing this edge contains a third vertex of the cycle.*

Moreover, it follows from Proposition 7 in [4] that balanced graphs belong to the class of *hereditary clique-Helly graphs*; i.e., the class of graphs whose induced subgraphs satisfy that the intersection of any nonempty family of pairwise intersecting maximal cliques is nonempty. Hereditary clique-Helly graphs were characterized by Prisner [57] as those graphs containing none of the graphs in Figure 2, which we call the *pyramids*, as induced subgraphs. In particular, since balanced graphs are hereditary clique-Helly, no pyramid can be an induced subgraph of a balanced graph.

A characterization of balanced graphs in terms of forbidden induced subgraphs was given in [10]. For each graph G and each subset W of its vertex set, let $N_G(W) = \bigcap_{w \in W} N_G(w)$ if W is nonempty; whereas, $N_G(\emptyset) = V(G)$. Similarly, if e is an edge with endpoints v and w, $N_G(e) = N_G(u) \cap N_G(w)$. An *extended odd sun* [10] is a graph G having an odd cycle C such that for each edge $e \in E(C)$ (i.e., joining two consecutive vertices of C) there is a clique W_e such that $W_e \subseteq N_G(e) - V(C)$, $N_G(W_e) \cap N_G(e) \cap V(C) = \emptyset$, $|W_e| \leq |N_G(e) \cap V(C)|$, and $V(G) = V(C) \cup \bigcup_{e \in E(C)} W_e$. The extended odd suns with the smallest number of vertices are C_5 and the pyramids. Moreover, every odd hole is an extended odd sun (by letting $W_e = \emptyset$ for each e). Notice that C_3 is not an extended

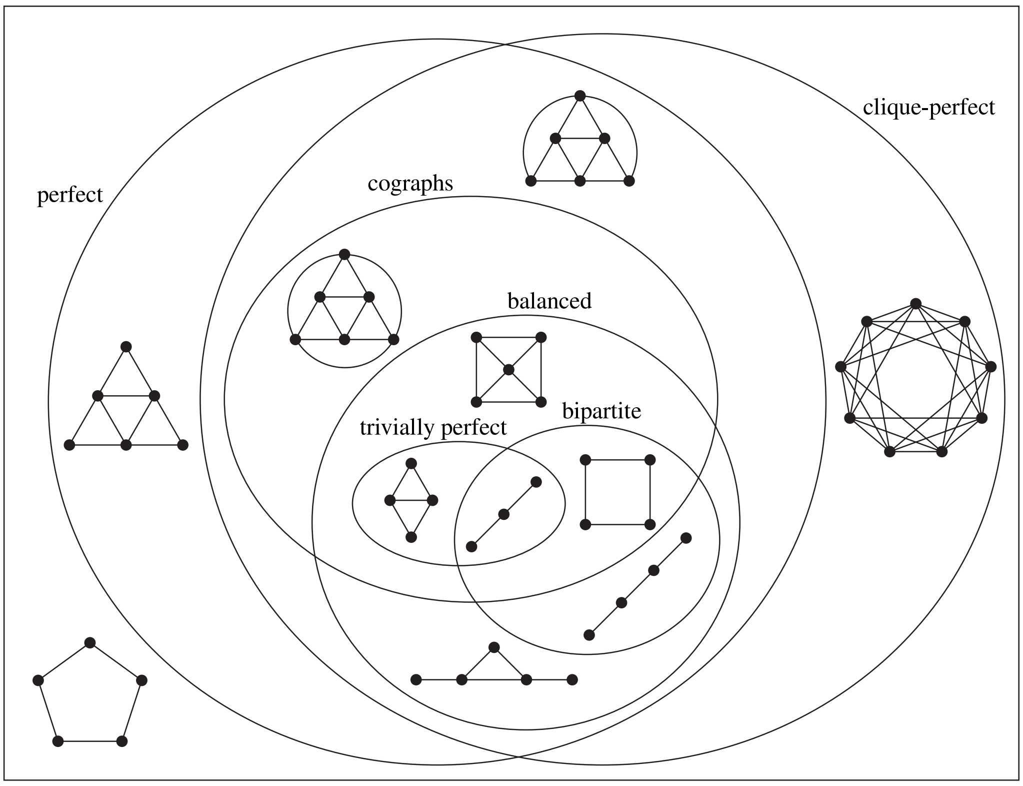

Figure 1. Examples of graphs in the intersections among several graph classes discussed in this work

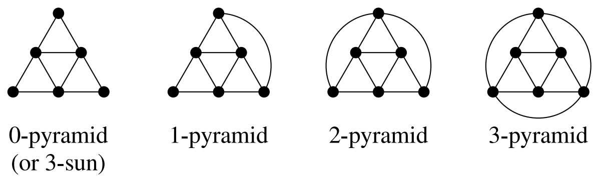

Figure 2. The pyramids

odd sun, since otherwise we would be forced to choose $W_e = \emptyset$ for each edge e, but then $N(W_e) \cap N(e) \cap V(C) = \{v\}$ where v is the only vertex non-incident to e (because $N(W_e) = N(\emptyset) = V(C)$ and $N(e) = \{v\}$). The characterization of balanced graphs by forbidden induced subgraphs is the following.

Theorem 1.5 ([10])**.** *A graph is balanced if and only if it contains no induced extended odd sun.*

Nevertheless, as already noticed in [10], extended odd suns do not constitute a *minimal* set of forbidden induced subgraphs because some extended odd suns contain some other extended odd suns as induced subgraphs (see Figure 3). In fact, it remains unsolved the problem of characterizing balanced graphs by *minimal forbidden induced subgraphs*; i.e., graphs that are not balanced but whose proper induced subgraphs are balanced. (An induced subgraph H of a graph G is *proper* if H is different from G.) Some partial results in this direction were given in [14] where balancedness of some classes of circular-arc graphs was characterized in terms of minimal forbidden induced subgraphs. Moreover, in [16], minimal forbidden induced subgraph characterizations and linear-time recognition algorithms for balancedness of graphs belonging to one of three different graph classes were presented. The currently best time complexity of a recognition algorithm for the whole class of balanced graphs is $O(m^9 + n)$ (where n and m denote the number of

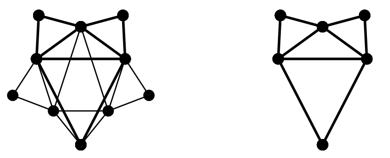
Figure 3. On the left, an extended odd sun that is not minimal. Bold lines correspond to the edges of a proper induced extended odd sun, depicted on the right.

vertices and edges of the input graph), which is achieved by computing a clique-matrix using the algorithm in [61] and then relying on the algorithm given in [65] to decide whether or not such clique-matrix is balanced; see [16] for the details of the derivation.

In this work, we study clique-perfectness and balancedness of P_4-tidy graphs, balancedness of paw-free graphs, and the connection between clique-perfectness and balancedness of diamond-free graphs. (The definition of P_4-tidy graphs, paw-free graphs, and diamond-free graphs are deferred to Sections 2, 3, and 4, respectively.)

The structure of the paper is as follows. In the next subsection, we introduce basic definitions and notations and give a short introduction to modular decomposition of graphs. In Section 2, we give forbidden induced subgraph characterizations and linear-time recognition algorithms for clique-perfectness and balancedness of P_4-tidy graphs and a linear-time algorithm for computing a maximum clique-independent set and a minimum clique-transversal set for any given P_4-tidy graph. All the algorithms in Section 2 work by performing a simple $O(n)$ time traversal of the modular decomposition tree. In Section 3, we give a minimal forbidden induced subgraph characterization of those paw-free graphs that are balanced, from which we derive that balancedness of paw-free graphs can be decided in linear time. In Section 4, we study clique-perfectness of diamond-free graphs. In fact, although there is a forbidden induced subgraph characterization for clique-perfectness of diamond-free graphs [9], no polynomial-time recognition algorithm is known. We prove the existence of a polynomial-time recognition algorithm for clique-perfectness of diamond-free graphs by proving that a diamond-free graph is clique-perfect if and only if it is balanced. The main results of Subsection 2.2 and Section 3 appeared (without proof) in the extended abstract [12].

1.1 *Definitions, notations, and modular decomposition*

All graphs in this work are finite, undirected, and have no loops or multiple edges. For all graph-theoretic notions and notations not defined here, we refer to West [64].

Let G be a graph. We denote by $V(G)$ its vertex set, by $E(G)$ its edge set, by $\overline{G}$ its complement, by $N_G(v)$ the neighborhood of a vertex v in G, and by $N_G[v]$ the closed neighborhood $N_G(v) \cup \{v\}$. A vertex of G is *pendant* if it is adjacent to exactly one other vertex and *universal* if it is adjacent to all the other vertices. We denote by $G[W]$ the subgraph of G induced by $W \subseteq V(G)$. If H is a graph, then G is *H-free* if G contains no induced subgraph isomorphic to H. If $\mathcal{H}$ is a collection of graphs, then G is *$\mathcal{H}$-free* if G is H-free for each $H \in \mathcal{H}$. A cycle is *odd* if it has an odd number of vertices. The chordless path (resp. cycle) on k vertices is denoted by P_k (resp. C_k). The graph K_2 (resp. $2K_1$) is the graph with two vertices and one (resp. no) edge. The disjoint union and the join of two graphs G_1 and G_2 are denoted by $G_1 + G_2$ and $G_1 \vee G_2$, respectively. We denote the size of a set S by $|S|$. A *clique* (resp. *stable set*) is a set of pairwise adjacent (resp. nonadjacent) vertices. We use *maximum* to mean maximum-sized, whereas *maximal* means inclusion-wise maximal. The use of *minimum* and *minimal* is analogous.

By *substituting* a vertex v of a graph G by another graph H, we mean substituting the vertex v in G by the graph H and adding all possible edges between the vertices of H and the neighbors of v in G.

We say that a graph is *co-connected* if its complement is connected. The *anticomponents* of a graph are its maximal co-connected induced subgraphs; equivalently, the anticomponents of a graph G are the complements of the components of $\overline{G}$. In particular, if $G_1, \dots, G_k$ are the anticomponents of a graph G, then $G = G_1 \vee \cdots \vee G_k$.

When discussing about algorithms and data structures, we denote by n the number of vertices of the input graph G.

Modular decomposition of a graph

Let G be a graph. A set M of vertices of G is a *module* if every vertex outside M is either adjacent to all vertices of M or to none of them. The empty set, the singletons $\{v\}$ for each $v \in V(G)$, and $V(G)$ are the *trivial modules* of G. A graph is *prime* if it has more than two vertices and it has only trivial modules. A nonempty module M of G is *strong* if, for every other module M' of G, either $M \cap M' = \emptyset$, M, or M'. The *modular decomposition tree* $T(G)$ of a graph G is a rooted tree having one node for each strong module of G and such that a node h representing a strong module M has as its children the nodes representing the maximal strong modules of G properly contained in M. Therefore, the root of $T(G)$ represents $V(G)$ and the leaves of $T(G)$ the singletons $\{v\}$ for each $v \in V(G)$. We will identify the module $\{v\}$ with the vertex v and say that the leaves of $T(G)$ are the vertices of G. For each node h of $T(G)$, we denote by $M(h)$ the strong module of G represented by h. By construction, $M(h)$ is the set of vertices of G having h as their ancestor in $T(G)$.

For each node h of $T(G)$, we denote the induced subgraph $G[M(h)]$ by $G[h]$ and call it the *graph represented by h*. Each internal node h of $T(G)$ is a *P-node*, an *S-node*, or an *N-node*, according to whether $G[h]$ is disconnected, $\overline{G[h]}$ is disconnected, or both $G[h]$ and $\overline{G[h]}$ are connected, respectively. (P-nodes, S-nodes, and N-nodes are also sometimes called *parallel*, *series*, and *neighborhood* nodes, respectively.) Hence, if h is an internal node of $T(G)$ and $h_1, \dots, h_k$ are the children of h in $T(G)$, then the following conditions hold:

(1) If $G[h]$ is disconnected, then h is a P-node and $G[h_1], \dots, G[h_k]$ are the components of $G[h]$.
(2) If $\overline{G[h]}$ is disconnected, then h is an S-node and $G[h_1], \dots, G[h_k]$ are the anticomponents of $G[h]$.
(3) If $G[h]$ and $\overline{G[h]}$ are both connected, then h is an N-node and $M(h_1), \dots, M(h_k)$ is the set of maximal strong modules of $G[h]$ properly contained in $M(h)$.

In the above three cases, it holds that $\{M(h_1), \dots, M(h_k)\}$ is a partition of $M(h)$ [18, 43].

Let h be an N-node of $T(G)$ and let $h_1, \dots, h_k$ be its children. Let $\pi(h)$ denote the graph having vertex set $\{h_1, \dots, h_k\}$ and such that h_i is adjacent to h_j if and only if there is some edge in G joining a vertex of $M(h_i)$ to a vertex of $M(h_j)$. By construction, $\pi(h)$ is a prime graph. Notice that since $M(h_i)$ and $M(h_j)$ are disjoint modules of G, h_i is adjacent to h_j in $\pi(h)$ if and only if every vertex of $M(h_i)$ is adjacent to every vertex of $M(h_j)$ in G. Hence, $G[h]$ coincides with the graph that arises from $\pi(h)$ by successively substituting h_i by $G[h_i]$ for each $i \in \{1, \dots, k\}$. We denote by $\pi(G)$ the set $\{\pi(h)\colon\ h \text{ is an N-node of } T(G)\}$. The following result shows that each induced prime subgraph of a graph G is also an induced subgraph of some graph in $\pi(G)$.

Theorem 1.6 ([39]). *Let Z be a prime graph. A graph G is Z-free if and only if each graph of $\pi(G)$ is Z-free.*

In this work, we assume, without loss of generality, that at each N-node h of the modular decomposition tree $T(G)$ there is attached a description of the prime graph $\pi(h)$ by means of adjacency lists. The interested reader is referred to [29, 30, 53, 60], where linear-time algorithms for computing the rooted tree $T(G)$ are given and to [2], where it is shown how the adjacency list of the prime graphs $\pi(h)$ for all the N-nodes h can be attached also in linear time.

For the sake of simplicity, we will denote $|V(G[h])|$ by $n(h)$ for every node h of $T(G)$. Moreover, if h is an N-node, we will denote $|V(\pi(h))|$ by $n_\pi(h)$. Notice that since $T(G)$ has n leaves and each internal node has at least two children, $T(G)$ has less than $2n$ nodes. An important property that we will use frequently is that the sum of $n_\pi(h)$ as h ranges over all the N-nodes of $T(G)$ is at most $2n$ (see, e.g. [2]).

2. Clique-perfectness and balancedness of P_4-tidy graphs

A *cograph* is a P_4-free graph. In [59], Seinsche proved that cographs are perfect by relying on the following structure theorem.

Theorem 2.1 ([59]). *If G is a cograph having at least two vertices, then either G or $\overline{G}$ is disconnected.*

Clearly, the above theorem implies that the modular decomposition trees of cographs, known as *cotrees*, have no N-nodes. By exploiting the absence of N-nodes in cotrees, a linear-time recognition algorithm and easy polynomial or even linear-time algorithms for solving classical graph problems (widely regarded as hard for general graphs) were proposed for cographs [24–26]. In [46], linear-time algorithms for computing a maximum clique-independent set and a minimum clique-transversal set for any given cograph were devised by also relying on the simplicity of the cotree structure. The following property of α_{c} and τ_{c} was also useful.

Theorem 2.2 ([46]). *If G_1 and G_2 are two graphs, then*

$$\alpha_{\mathrm{c}}(G_1 \vee G_2) = \min\{\alpha_{\mathrm{c}}(G_1), \alpha_{\mathrm{c}}(G_2)\} \qquad \textit{and} \qquad \tau_{\mathrm{c}}(G_1 \vee G_2) = \min\{\tau_{\mathrm{c}}(G_1), \tau_{\mathrm{c}}(G_2)\}.$$

Distance hereditary graphs form a superclass of the class of cographs. In [51], distance-hereditary graphs were shown to be clique-perfect and linear-time algorithms for computing a maximum clique-independent set and a minimum clique-transversal set for any given distance-hereditary graph were proposed, by relying on a decomposition tree of distance-hereditary graphs introduced in [20].

The class of P_4-tidy graphs is a different superclass of the class of cographs that extends many other generalizations of cographs that arise by bounding the occurrence of induced P_4's according to different criteria; e.g., P_4-sparse graphs [47], P_4-lite graphs [48], and P_4-extendible graphs [49]. A graph is P_4-*tidy* [44] if for each 4-vertex set A inducing a P_4 in G there is at most one vertex v such that $A \cup \{v\}$ induces a graph with two induced P_4's. There is a structure theorem for P_4-tidy graphs which is an extension of Theorem 2.1 and that can be stated in terms of starfishes and urchins. A *starfish* is a graph whose vertex set can be partitioned into three sets S, C and R, where each of the following conditions holds:

(1) $S = \{s_1, \dots, s_t\}$ is a stable set and $C = \{c_1, \dots, c_t\}$ is a clique, for some $t \geq 2$.
(2) s_i is adjacent to c_j if and only if $i = j$.
(3) R is allowed to be empty and if it is not, then all the vertices in R are adjacent to all the vertices in C and nonadjacent to all the vertices in S.

An *urchin* is a graph whose vertex set can be partitioned into three sets S, C, and R satisfying conditions (1) and (3) above but that instead of condition (2) satisfies:

(2') s_i is adjacent to c_j if and only if $i \neq j$.

Clearly, urchins are the complements of starfishes and vice versa. If G is a starfish or an urchin, then the triple (S, C, R) is called the *partition* of G, the vertices of S are the *ends* of G, the vertex set C is the *body* of G, and the subgraph induced by the vertices of R is the *head* of G. Notice that a starfish or urchin is prime (in the sense of Subsection 1.1) if and only if its head R has at most one vertex. A *fat starfish* (resp. *fat urchin*) arises from a starfish (resp. urchin) with partition (S, C, R) by substituting exactly one vertex of $S \cup C$ by K_2 or $2K_1$. In [44], the following structure theorem for P_4-tidy graphs was proved.

Theorem 2.3 ([44])**.** *If G is a P_4-tidy graph, then exactly one of the following statements holds:*

(1) G or $\overline{G}$ is disconnected.
(2) G is isomorphic to C_5, P_5, $\overline{P_5}$, a starfish, a fat starfish, an urchin, or a fat urchin.

Let G be a P_4-tidy graph and let h be an N-node of $T(G)$. The above theorem implies that $\pi(h)$ is isomorphic to C_5, P_5, $\overline{P_5}$, a prime starfish, or a prime urchin. Moreover, if $\pi(h)$ is isomorphic to C_5, P_5, or $\overline{P_5}$, then h has five children, each of which is a leaf. If, on the contrary, $\pi(h)$ is a prime starfish or a prime urchin, each of its children in $T(G)$ is a leaf except maybe for one child h_R representing the head and/or one other child representing $2K_1$ or K_2. It was shown in [44] that in $O(n_\pi(h))$ time it can be decided whether or not $\pi(h)$ is a starfish (resp. an urchin) and, if affirmative, find its partition.

Similarly to the case of cographs, Theorem 2.3 leads to a linear-time recognition algorithm for P_4-tidy graphs as well as to simple and efficient algorithms on P_4-tidy graphs for different otherwise hard problems [44]. In this section, we will exploit the structure of P_4-tidy graphs to prove minimal forbidden induced subgraph characterizations and devise simple linear-time recognition algorithms for both clique-perfectness and balancedness of P_4-tidy graphs. In addition, we will give a linear-time algorithm for computing a maximum clique-independent set and a minimum clique-transversal set for any given P_4-tidy graph. All the algorithms in this section work by performing a simple $O(n)$ time traversal of the modular decomposition tree.

2.1 *Clique-perfectness of P_4-tidy graphs*

The first goal of this subsection is to find a minimal forbidden induced subgraph characterization and a linear-time recognition algorithm for clique-perfectness of P_4-tidy graphs. We begin by determining the values of α_c and τ_c for those P_4-tidy graphs that are connected and co-connected.

Lemma 2.4. *If G is a connected and co-connected P_4-tidy graph, then one of the following statements holds:*

(1) G is isomorphic to C_5, $\alpha_c(G) = 2$, and $\tau_c(G) = 3$.
(2) G is isomorphic to P_5 or $\overline{P_5}$, and $\alpha_c(G) = \tau_c(G) = 2$.
(3) G is a starfish with t ends or a fat starfish arising from it, and $\alpha_c(G) = \tau_c(G) = t$.
(4) G arises from an urchin with 3 ends by substituting a vertex of the body by $2K_1$, and $\alpha_c(G) = \tau_c(G) = 2$.
(5) G is an urchin with at least 3 ends or a fat urchin arising from it, G is not a fat urchin arising form an urchin with 3 ends by substituting a vertex of the body by $2K_1$, $\alpha_c(G) = 1$, and $\tau_c(G) = 2$.

Proof. Since G is a connected and co-connected P_4-tidy graph, it follows from Theorem 2.3 that G is isomorphic to C_5, P_5, or $\overline{P_5}$, a starfish, a fat starfish, an urchin, or a fat urchin. The values of α_c and τ_c for C_5, P_5, and $\overline{P_5}$ are clear by simple inspection.

Consider first the case where G is a starfish with partition (S, C, R) or a fat starfish arising from it and let $t = |S|$. If G is a fat starfish arising by substituting a vertex c of the body by $2K_1$, then $\{N_G[s']\colon\ s' \in S - \{s\}\} \cup \{\{s, c'\}\}$, where s was the only neighbor of c in S and c' is one of the vertices by which c was substituted, is a clique-independent set of size t; otherwise, $\{N_G[s]\colon\ s \in S\}$ is a clique-independent set of size t. Thus, $\alpha_c(G) \geq t$. If G is a fat starfish that arises by substituting a vertex c of the body by K_2 or $2K_1$, then $(C - \{c\}) \cup \{s\}$, where s was the only neighbor of c in S, is a clique-transversal set of size t; otherwise, C is a clique-transversal set of size t. In either case, $\tau_c(G) \leq t$. Since $\alpha_c(G) \leq \tau_c(G)$, we conclude that $\alpha_c(G) = \tau_c(G) = t$. Without loss of generality, we assume from now on that G is neither a starfish nor a fat starfish.

Consider now the case where G is an urchin with partition (S, C, R) or a fat urchin arising from it and let $t = |S|$. Since we are assuming that G is neither a starfish nor a fat starfish, $t \geq 3$. Therefore, there are at least two vertices c_1 and c_2 of C that remain in G (i.e., that were not substituted in case G is a fat urchin) and $\{c_1, c_2\}$ is a clique-transversal set of G because any maximal clique of G containing neither c_1 nor c_2 must contain a non-neighbor s_1 of c_1 and a non-neighbor s_2 of c_2 in G, but s_1 and s_2 are necessarily nonadjacent by construction, a contradiction. Moreover, since G has no universal vertex, $\{c_1, c_2\}$ is a minimum clique-transversal set and $\tau_c(G) = 2$. If G is an urchin or G arises by substituting a vertex of S by K_2 or $2K_1$, then each maximal clique of G contains at least $t - 1$ vertices of C and, since $|C| = t \geq 3$, all the maximal cliques of G are pairwise intersecting; i.e., $\alpha_c(G) = 1$ and $\{N_G[s]\}$ where s is any vertex of S that remains in G (i.e., that was not substituted in case G is a fat urchin) is a maximum clique-independent set. It only remains to consider the case where G is a fat urchin arising from an urchin with partition (S, C, R) by substituting a vertex c of C by two new vertices inducing K_2 or $2K_1$. In this case, each maximal clique of G contains at least $t - 2$ vertices of the $t - 1$ vertices of $C - \{c\}$. Therefore, if either (i) $t = 3$ and G arises by substituting c by K_2 or (ii) $t > 3$, then the maximal cliques of G are pairwise intersecting and $\{N_G[s]\}$, where s is the vertex of S which was nonadjacent to c, is a maximum clique-independent set because $\alpha_c(G) = 1$. It only remains to consider the case where $t = 3$ and G arises by substituting a vertex of the body by $2K_1$; i.e., $S = \{s_1, s_2, s_3\}$ and $C = \{c_1, c_2, c_3\}$ where s_i is adjacent to c_j if and only if $i \neq j$, and G arises by substituting c_1 by $2K_1$ whose vertices are c_1' and c_1''. In this case, $\{\{c_1', c_2, s_3\}, \{c_1'', s_2, c_3\}\}$ is a clique-independent set, which is maximum because $\tau_c(G) = 2$ and thus $\alpha_c(G) = 2$. □

Our result below gives a characterization in terms of minimal forbidden induced subgraphs of those P_4-tidy graphs that are clique-perfect.

Theorem 2.5. *If G is a P_4-tidy graph, then the following assertions are equivalent:*

(1) G is clique-perfect.
(2) G contains no induced C_5 and no induced 3-sun.
(3) Each graph of $\pi(G)$ is isomorphic to P_5, $\overline{P_5}$, or a starfish.

Proof. We first prove that (1) ⇔ (2). The implication (1) ⇒ (2) is clear because C_5 and the 3-sun are not clique-perfect and the class of clique-perfect graphs is hereditary. We now prove that (2) ⇒ (1). Suppose that G is not clique-perfect. Hence, G must contain an induced subgraph H which is *minimally not clique-perfect*; i.e., which is not clique-perfect but each of its proper induced subgraphs is clique-perfect. The minimality of H implies that H is connected since otherwise H would be the disjoint union of clique-perfect graphs and thus H would be clique-perfect. Analogously, $\overline{H}$ is also connected since otherwise H would be the join of two clique-perfect graphs and, by virtue of Theorem 2.2, H would be clique-perfect. Hence, H is a connected and co-connected P_4-tidy graph. Moreover, the fact that each proper induced subgraph of H is clique-perfect but H is not clique-perfect forces $\alpha_c(H) \neq \tau_c(H)$. Hence, Lemma 2.4 implies that H is isomorphic to C_5, an urchin or a fat urchin. Notice that if H is an urchin or a fat urchin, then H contains an induced 3-sun and, because of the minimality, H is isomorphic to the 3-sun. We conclude that H is isomorphic to C_5 or the 3-sun. Since H is, by construction, an induced subgraph of G, this proves (2) $\Rightarrow$ (1).

Let us now prove that (1) $\Leftrightarrow$ (3). Since C_5 and the 3-sun are prime, if follows from Theorem 1.6 that G is $\{C_5$,3-sun$\}$-free if and only if each graph of $\pi(G)$ is $\{C_5$,3-sun$\}$-free. If h is an N-node of $T(G)$, then Theorem 2.3 implies that $\pi(h)$ is isomorphic to C_5, P_5, $\overline{P_5}$, a prime starfish, or a prime urchin. Hence, $\pi(h)$ is $\{C_5$,3-sun$\}$-free if and only if $\pi(h)$ is isomorphic to P_5, $\overline{P_5}$, or a starfish. Since h is an arbitrary N-node of $T(G)$, the equivalence between (1) and (3) follows. □

From the equivalence between assertions (1) and (3) of the above theorem, a linear-time recognition algorithm for clique-perfectness of P_4-tidy graphs follows.

Corollary 2.6. *It can be decided whether or not any given P_4-tidy graph G is clique-perfect in linear time; in fact, in $O(n)$ time once the modular decomposition tree of G is built.*

Proof. Recall from Subsection 1.1 that the modular decomposition tree of G can be built in linear time. Since $T(G)$ has less than $2n$ nodes, a traversal of all of its nodes can be arranged in $O(n)$ time. For each of the N-nodes h of $T(G)$ it takes constant time to determine whether $\pi(h)$ is isomorphic to P_5 or $\overline{P_5}$ and it takes $O(n_\pi(h))$ time to determine whether or not $\pi(h)$ is a starfish. Hence, since the sum of $n_\pi(h)$ as h ranges over all the N-nodes of $T(G)$ is at most $2n$, it can be decided whether or not assertion (3) of Theorem 2.5 holds in $O(n)$ time once the modular decomposition tree of G is built. □

We will now show that both α_c and τ_c can be computed in linear time for any given P_4-tidy graph. Below we propose an algorithm which, given any P_4-tidy graph, traverses the nodes of its modular decomposition tree in *post-order* (where each node is visited after all of its children), computing for each node h the values of $\alpha_c(G[h])$ and $\tau_c(G[h])$ which we will denote simply by $\alpha_c(h)$ and $\tau_c(h)$.

Algorithm 1. Computes α_c and τ_c for any given P_4-tidy graph

```
Input: A $P_4$-tidy graph $G$
Initialization: Build the modular decomposition tree of $G$
Step 1: Visit the nodes of $T(G)$ in post-order, doing the following for each node $h$:
    if $h$ is a leaf then
        $\alpha_c(h) = \tau_c(h) = 1$
    else if $h$ is a P-node with children $h_1, \dots, h_k$ then
        $\alpha_c(h) = \alpha_c(h_1) + \cdots + \alpha_c(h_k)$ and $\tau_c(h) = \tau_c(h_1) + \cdots + \tau_c(h_k)$
    else if $h$ is an S-node with children $h_1, \dots, h_k$ then
        $\alpha_c(h) = \min\{\alpha_c(h_1), \dots, \alpha_c(h_k)\}$ and $\tau_c(h) = \min\{\tau_c(h_1), \dots, \tau_c(h_k)\}$
    else if $\pi(h)$ is isomorphic to $C_5$ then
        $\alpha_c(h) = 2$ and $\tau_c(h) = 3$
    else if $\pi(h)$ is isomorphic to $P_5$ or $\overline{P_5}$ then
        $\alpha_c(h) = \tau_c(h) = 2$
    else if $\pi(h)$ is a starfish with $t$ ends then
        $\alpha_c(h) = \tau_c(h) = t$
    else if $\pi(h)$ is an urchin with 3 ends and a vertex of its body represents $2K_1$ then
        $\alpha_c(h) = \tau_c(h) = 2$
    else
```

$\alpha_c(h) = 1$ and $\tau_c(h) = 2$.

Step 2: Output $\alpha_c(G) = \alpha_c(r)$ and $\tau_c(G) = \tau_c(r)$, where r is the root of $T(G)$.

Theorem 2.7. *Algorithm 1 correctly computes $\alpha_c(G)$ and $\tau_c(G)$ for any given P_4-tidy graph in linear time; in fact, in $O(n)$ time once the modular decomposition tree of G is built.*

Proof. The correctness of the values of $\alpha_c(h)$ and $\tau_c(h)$ as computed by Algorithm 1 follows from Theorem 2.2 and Lemma 2.4. Recall from Subsection 1.1 that the modular decomposition tree of G can be built in linear time. Since $T(G)$ has fewer than $2n$ nodes, arranging the post-order traversal takes $O(n)$ time. Moreover, since for each N-node h of $T(G)$ it takes $O(n_\pi(h))$ time to decide whether or not $\pi(h)$ is a starfish (resp. an urchin) and, if affirmative, find its partition, Algorithm 1 takes only $O(n)$ time once the modular decomposition tree of G is built (recall that the sum of $n_\pi(h)$ as h ranges over all the N-nodes of $T(G)$ is at most $2n$). □

We will argue now that Algorithm 1 can be modified so as to output a maximum clique-independent set and a minimum clique-transversal set also in linear time. For that purpose, instead of computing $\alpha_c(h)$ and $\tau_c(h)$ for each node h, we will compute two lists: $L_A(h)$ and $L_T(h)$, consisting of a maximum clique-independent set and a minimum clique-transversal set, respectively. The elements of the list $L_A(h)$ will, in its turn, be lists, each of which representing one maximal clique of the clique-independent set and having precisely the vertices of such maximal clique as its elements. The elements of the list $L_T(h)$ will be the vertices of the clique-transversal set. We denote a list L with elements $e_1, \ldots, e_\ell$ by $L = \langle e_1, \ldots, e_\ell \rangle$ and we write $|L|$ to denote the number of elements of L.

For instance, if h is a leaf of $T(G)$ representing a vertex v, we let $L_A(h) = \langle\langle v \rangle\rangle$ (a list whose only element is a list whose only element is v) to represent the maximum clique-independent set $\{\{v\}\}$ and $L_T(h) = \langle v \rangle$ to represent the minimum clique-transversal set $\{v\}$. If h is a P-node, we let $L_A(h)$ be the concatenation of the lists $L_A(h_1), \ldots, L_A(h_k)$ and $L_T(h)$ be the concatenation of $L_T(h_1), \ldots, L_T(h_k)$. If h is an S-node, we let $L_A(h)$ be the list of $\min\{|L_A(h_1)|, \ldots, |L_A(h_k)|\}$ elements such that for each $i \in \{1, \ldots, k\}$ the ith element of $L_A(h)$ is the concatenation of all the ith elements of $L_A(h_1), \ldots, L_A(h_k)$, and we let $L_T(h)$ be a list of minimum length among $L_T(h_1), \ldots, L_T(h_k)$. The proof of Lemma 2.4 contains instructions on how to build $L_A(h)$ and $L_T(h)$ in case h is an N-node; notice that in this case, the lists L_A and L_T of the children of h play no role when constructing $L_A(h)$ and $L_T(h)$.

Theorem 2.8. *Algorithm 1 modified as above computes a maximum clique-independent set and a minimum clique-transversal set of any given P_4-tidy graph G in linear time; in fact, in $O(n)$ time once the modular decomposition tree of G is built.*

Proof. The correctness of the modified algorithm follows from the correctness of Algorithm 1 because the lists built as described above clearly represent valid clique-independent sets and clique-transversal sets of the correct sizes. Analogously, regarding the time complexity, we only have to prove that all the lists $L_A(h)$ and $L_T(h)$ for the nodes h of $T(G)$ can be build in $O(n)$ time. It is clear that the time involved in building the lists $L_A(h)$ and $L_T(h)$ is proportional to the number of children if h is a P-node or S-node and proportional to $n_\pi(h)$ if h is an N-node, except for the many concatenations needed to build all the lists $L_A(h)$ for the S-nodes h; nevertheless, we will show that in fact the total number of concatenations performed in all the S-nodes is at most n. Altogether, since the total number of nodes of $T(G)$ and the sum of $n_\pi(h)$ as h ranges over the N-nodes of $T(G)$ are both at most $2n$, the statement of the theorem follows. Notice that we are assuming that when building $L_A(h)$ for an S-node h each concatenation between two lists can be performed in $O(1)$ time. This is possible as long as we do not mind about modifying the original lists to be concatenated, which is the case here, because the lists associated to a node are only needed when building the lists corresponding to its parent.

For each node h of $T(G)$, let $c(h)$ be the total number of concatenations needed to compute all the lists $L_A(h')$ for the S-nodes h' in the subtree of $T(G)$ rooted at h. We will prove that $c(r) \leq n(r)$ where r is the root of $T(G)$. We will reach this conclusion by proving the following stronger claim by structural induction: $c(h)+\alpha_c(h) \leq n(h)$ for each node h of $T(G)$. If h is a leaf, $c(h)+\alpha_c(h) = 1 = n(h)$. Assume that h is an internal node and, by induction hypothesis, that all its children $h_1, \ldots, h_k$ satisfy $c(h_i)+\alpha_c(h_i) \leq n(h_i)$. If h is a P-node, then $c(h)+\alpha_c(h) = \sum_{i=1}^k c(h_i) + \sum_{i=1}^k \alpha_c(h_i) = \sum_{i=1}^k (c(h_i)+\alpha_c(h_i)) \leq \sum_{i=1}^k n(h_i) = n(h)$. If h is an S-node, then $c(h)+\alpha_c(h) = \big((k-1)\min_{i\in\{1,\ldots,k\}} \alpha_c(h_i) + \sum_{i=1}^k c(h_i)\big) + \min_{i\in\{1,\ldots,k\}} \alpha_c(h_i) \leq \sum_{i=1}^k (c(h_i)+\alpha_c(h_i)) \leq \sum_{i=1}^k n(h_i) = n(h)$. It only remains to consider the case where h is an N-node. On the one hand, if $\pi(h)$ is isomorphic to C_5, P_5, $\overline{C_5}$, then $c(h)+\alpha_c(h) = 2 < 5 = n(h)$. On the other hand, if $\pi(h)$ is a starfish or an urchin, then $c(h)+\alpha_c(h) \leq \sum_{i=1}^k c(h_i) + k/2 < \sum_{i=1}^k (c(h_i)+1) \leq \sum_{i=1}^k n(h_i) = n(h)$ because $\alpha_c(h_i) \geq 1$ for each $i \in \{1, \ldots, k\}$. We conclude by induction that indeed $c(h)+\alpha_c(h) \leq n(h)$ for each node h of $T(G)$. In particular, if r is the root of $T(G)$, then $c(r) \leq n(r) = n$; i.e., the number of concatenations needed to construct all the lists $L_A(h)$ for the S-nodes h of $T(G)$ is less than n, as claimed. □

2.2 *Balancedness of P_4-tidy graphs*

In this subsection, we will give a forbidden induced subgraph characterization and a linear-time recognition algorithm for balancedness of P_4-tidy graphs. A key role in this section is played by the results below which give necessary and sufficient conditions for the join of two or more graphs (either P_4-tidy or not) to be balanced. In order to state the results, we rely on the notion of trivially perfect graphs. Namely, a graph G is *trivially perfect* if and only if for every induced subgraph H of G, the maximum size of a stable set of H equals the number of maximal cliques of H [45]. Trivially perfect graphs coincide with $\{P_4, C_4\}$-free graphs [45] and are known to be balanced [38].

The lemma below gives two sufficient conditions and one necessary condition for the join of two graphs to be balanced.

Lemma 2.9. *If a graph G is the join of two graphs G_1 and G_2, then each of the following statements holds:*

(a) If G_1 is balanced and G_2 is a complete graph, then G is balanced.
(b) If G_1 and G_2 are trivially perfect, then G is balanced.
(c) If G is balanced and G_2 is not a complete graph, then G_1 is trivially perfect.

Proof. (a) Suppose that G_1 is balanced and G_2 is a complete graph. If M_G is a clique-matrix of G, then a clique-matrix of G_1 arises from M_G by removing each of the columns corresponding to the vertices of G_2, which are columns completely filled with ones. Hence, G is balanced because G_1 is balanced.

(b) Suppose that G_1 and G_2 are trivially perfect and, by the way of contradiction, that G is not balanced. By definition, there are some maximal cliques $Q_1, \ldots, Q_{2t+1}$ of G and some pairwise different vertices $v_1, \ldots, v_{2t+1}$ of G for some $t \geq 1$ such that $Q_i \cap \{v_1, v_2, \ldots, v_{2t+1}\} = \{v_i, v_{i+1}\}$ for each $i \in \{1, \ldots, 2t+1\}$ (where v_{2t+2} stands for v_1). In particular, $C = v_1v_2 \ldots v_{2t+1}v_1$ is an odd cycle of G. Since C is odd, there are two consecutive vertices of C that are either both vertices of G_1 or both vertices of G_2. Without loss of generality, suppose that v_1 and v_2 belong to G_1. Let

$Q'_i = Q_i \cap V(G_1)$ and $Q''_i = Q_i \cap V(G_2)$, for each $i \in \{1, 2, \dots, 2t+1\}$. Hence, $Q_i = Q'_i \cup Q''_i$ where Q'_i is a maximal clique of G_1 and Q''_i is a maximal clique of G_2, for each $i \in \{1, 2, \dots, 2t+1\}$. Since v_1 does not belong to Q'_2, there is some vertex $w_2 \in Q'_2$ such that w_2 is not adjacent to v_1. Symmetrically, there is a vertex $w_1 \in Q'_{2t+1}$ such that w_1 is not adjacent to v_2. Thus, $\{w_1, v_1, v_2, w_2\}$ induces a C_4 or a P_4 in G_1 and G_1 is not trivially perfect, a contradiction. This contradiction arose from supposing that G was not balanced.

(c) Suppose that G is balanced and G_2 is not a complete graph. Then, G_1 is trivially perfect, since otherwise G_1 would contain an induced P_4 or C_4 and, since G_2 is not a complete graph, G would contain an induced $P_4 \vee 2K_1 \cong$ 2-pyramid or $C_4 \vee 2K_1 \cong$ 3-pyramid, respectively, a contradiction. □

We are now ready to prove the theorem below which gives necessary and sufficient conditions for the join of two or more graphs to be balanced.

Theorem 2.10. *If a graph G is the join of the graphs $G_1, \dots, G_k$, then G is balanced if and only if at least one of the following statements holds:*

(1) One of $G_1, \dots, G_k$ is balanced and the remaining ones are complete graphs.
(2) Two of $G_1, \dots, G_k$ are trivially perfect and the remaining ones are complete graphs.

Proof. Let G be the join of the graphs $G_1, \dots, G_k$. If (1) or (2) holds, then G is balanced by repeated application of statements (a) and (b) of Lemma 2.9. Conversely, suppose that G is balanced. Hence, G contains no induced 3-pyramid and, consequently, we assume, without loss of generality, that $G_3, \dots, G_k$ are complete graphs. Since G_1 and G_2 are induced subgraphs of G, G_1 and G_2 are also balanced. Therefore, if at least one of G_1 and G_2 is a complete graph, then (1) holds. If, on the contrary, none of G_1 and G_2 is a complete graph, then Lemma 2.9(c) implies that G_1 and G_2 are trivially perfect and (2) holds. □

Corollary 2.11. *Minimally not balanced graphs different from the 2-pyramid and the 3-pyramid are co-connected.*

Proof. Let H be a minimally not balanced graph whose complement $\overline{H}$ is disconnected. Since $\overline{H}$ is disconnected, H is the join of two graphs H_1 and H_2 with at least one vertex each. Since H is minimally not balanced, H_1 and H_2 are balanced. Since H is not balanced, Lemma 2.9(b) implies that H_1 or H_2 is not trivially perfect. Without loss of generality, assume that H_1 is not trivially perfect; i.e., H_1 contains an induced P_4 or an induced C_4. Since $H = H_1 \vee H_2$ is not balanced and H_1 is balanced, Lemma 2.9(a) guarantees that H_2 is not a complete graph. Thus, H_2 contains an induced $2K_1$. Finally, $H = H_1 \vee H_2$ contains an induced $P_4 \vee 2K_1 \cong$ 2-pyramid or an induced $C_4 \vee 2K_1 \cong$ 3-pyramid. By minimality, H is isomorphic to either the 2-pyramid or the 3-pyramid. □

In the proof of Lemma 2.12 and Theorem 2.13 below, we will make repeated use of the following trivial facts about balanced graphs.

Fact 1. If a graph G is balanced, then each graph that arises from G by substituting a vertex by K_2 is also balanced.

Fact 2. If a graph G is balanced, then each graph that arises from G by adding a pendant vertex is also balanced.

Our next result characterizes those connected and co-connected P_4-tidy graphs that are balanced.

Lemma 2.12. *If G is a connected and co-connected P_4-tidy graph, then G is balanced if and only if one of the following statements hold:*

(1) G is isomorphic to P_5 or $\overline{P_5}$.
(2) G is isomorphic to a starfish with a balanced (or empty) head.
(3) G is isomorphic to a fat starfish arising from a starfish with balanced (or empty) head by substituting a vertex of the body by K_2 or an end by K_2 or $2K_1$.
(4) G is isomorphic to a fat starfish arising from a starfish with trivially perfect (or empty) head by substituting a vertex of the body by $2K_1$.

Proof. Let G be a connected and co-connected P_4-tidy graph.

We will first prove that if G satisfies one of the conditions (1), (2), (3), or (4) then G is balanced. If statement (1) holds, then G is balanced by definition. If statement (2) holds, then G is balanced because G arises from the join of the head of the starfish and a complete graph (which is balanced by Lemma 2.9(a)) by repeatedly adding pendant vertices (which preserves balancedness by Fact 2). If statement (3) holds, then G arises from the join of the head of the starfish and a complete graph (which is balanced by Lemma 2.9(a)) by repeatedly adding pendant vertices (which preserves balancedness by Fact 2) and finally either adding an additional pendant vertex or substituting a vertex by K_2 (in either case balancedness is preserved by Facts 1 and 2). If statement (4) holds, then G is balanced because G arises from the join of the head of the starfish and the result of substituting $2K_1$ into a complete graph (which is balanced by Lemma 2.9(b) because it is the join of two trivially perfect graphs) by repeatedly adding pendant vertices (which preserves balancedness by Fact 2). We conclude that if G satisfies (1), (2), (3), or (4), then G is balanced.

For the converse, we suppose that G satisfies none of the conditions (1), (2), (3), or (4) and we will see that G is not balanced. In fact, since G is a connected and co-connected P_4-tidy graph, Theorem 2.3 implies that one of the following holds:

- *G is isomorphic to C_5.* In this case G is not balanced because it is not even perfect.
- *G is a starfish whose head is not balanced or is a fat starfish arising from it.* In this case G is not balanced because the head of the starfish is a non-balanced induced subgraph of G.
- *G is a fat starfish arising from a starfish whose head is not trivially perfect by substituting a vertex of the body by $2K_1$.* In this case G is not balanced because G contains as an induced subgraph the join of the head of the starfish and $2K_1$, which is not balanced by Lemma 2.9(c).
- *G is an urchin with at least 6 vertices or a fat urchin arising from it.* In this case G is not balanced because it contains an induced 3-sun. □

The characterization of those P_4-tidy graphs that are balanced in terms of minimal forbidden induced subgraphs is as follows.

Theorem 2.13. *If G is a P_4-tidy graph, then the following assertions are equivalent:*

(1) G is balanced.
(2) G is perfect and hereditary clique-Helly.
(3) G contains no induced C_5, 3-sun, 2-pyramid, or 3-pyramid.

Proof. The implications (1) $\Rightarrow$ (2) and (2) $\Rightarrow$ (3) follow from the discussion in the Introduction. We only need to prove that (3) $\Rightarrow$ (1) holds. For that, we show that the only P_4-tidy graphs that are minimally not balanced are C_5, the 3-sun, the 2-pyramid, and the 3-pyramid.

Let H be a P_4-tidy graph that is minimally not balanced. By minimality, H is connected. If $\overline{H}$ were disconnected, then, by Corollary 2.11, H would be isomorphic to a 2-pyramid or a 3-pyramid. Suppose, on the contrary, that $\overline{H}$ is connected. Hence, H is a connected and co-connected P_4-tidy graph which is not balanced and, as in the proof of Lemma 2.12, one of the following conditions holds: (i) H is isomorphic to C_5, (ii) H is a starfish whose head is not balanced or a fat starfish arising from it, (iii) H is a fat starfish arising from a starfish whose head is not trivially perfect by substituting a vertex of the body by $2K_1$, or (iv) H is an urchin or a fat urchin. If condition (i) holds, there is nothing else to prove. The minimality of H implies that H has no pendant vertices (because of Fact 2) and, consequently, conditions (ii) and (iii) cannot hold. Finally, if conditions (iv) holds, then H contains an induced 3-sun and, because of the minimality, H is isomorphic to the 3-sun. We conclude that indeed the only P_4-tidy graphs that are minimally not balanced are C_5, the 3-sun, the 2-pyramid, and the 3-pyramid. □

Our next goal in this section is to give a simple linear-time algorithm for deciding whether or not any given P_4-tidy graph is balanced. For that purpose, we will establish a connection between the balancedness of a graph and a property of its modular decomposition tree. The function f defined as follows will play an important role in that connection. Let f be the function assigning a nonnegative number to each graph G (with at least one vertex) defined recursively as follows:

- If G is disconnected and $G_1, \ldots, G_k$ are the components of G, then
$$f(G) = \max\{1, f(G_1), \ldots, f(G_k)\}.$$
- If $\overline{G}$ is disconnected and $G_1, \ldots, G_k$ are the anticomponents of G, then
$$f(G) = f(G_1) + \cdots + f(G_k).$$
- If G has at least two vertices and both G and $\overline{G}$ are connected, then $f(G) = 2$.
- If G has exactly one vertex, then $f(G) = 0$.

The theorem below contains as statement (3) the connection between the balancedness of a graph and a property of its modular decomposition tree that we need. Later in this subsection we will exploit this connection in order to give a simple linear-time algorithm for deciding balancedness of P_4-tidy graphs.

Theorem 2.14. *If G is a graph, then the following statements hold:*

(1) G is a complete graph if and only if $f(G) = 0$.
(2) G is trivially perfect if and only if $f(G) \leq 1$.
(3) G is balanced if and only if $f(G) \leq 2$ and each graph represented by an N-node of $T(G)$ is balanced.

Proof. We give a proof for each of the statements.

(1) *G is a complete graph if and only if $f(G) = 0$.* If G is a complete graph, then either G has just one vertex or $\overline{G}$ is disconnected and each anticomponent of G has exactly one vertex. In either case, the definition of f implies that $f(G) = 0$. Conversely, we now prove that if $f(G) = 0$ then G is a complete graph. If G has only one vertex, then G is a complete graph. Suppose now that G has at least two vertices. If $f(G) = 0$ then, by definition of f, $\overline{G}$ is disconnected and the anticomponents $G_1, \ldots, G_k$ of G satisfy $f(G_1) = \cdots = f(G_k) = 0$. The fact that the anticomponents of a graph are co-connected implies that $G_1, \ldots, G_k$ have exactly one vertex each and, consequently, G is a complete graph.
(2) *G is trivially perfect if and only if $f(G) \leq 1$.* We will prove that if G is trivially perfect then $f(G) \leq 1$, by induction on the number of vertices of G. Let G be a trivially perfect graph. If G has only one vertex, then $f(G) \leq 1$. Suppose that G has at least two vertices and that for each trivially perfect graph G' having fewer

vertices than G it holds that $f(G') \leq 1$. Since G is trivially perfect, in particular G is a cograph. Hence, Theorem 2.1 implies that either G or $\overline{G}$ is disconnected. If G is disconnected and $G_1, \ldots, G_k$ are its components, then $G_1, \ldots, G_k$ are all trivially perfect and, by induction hypothesis, $f(G_1) \leq 1, \ldots, f(G_k) \leq 1$ and thus $f(G) = \max\{1, f(G_1), \ldots, f(G_k)\} \leq 1$. Suppose that, on the contrary, $\overline{G}$ is disconnected and let $G_1, \ldots, G_k$ be the anticomponents of G. Notice that at most one of the anticomponents of G is not a complete graph, since otherwise G would contain an induced $2K_1 \vee 2K_1 \cong C_4$, a contradiction with the fact that G is trivially perfect. Therefore, without loss of generality, we assume that $G_2, \ldots, G_k$ are complete graphs and, by virtue of (1), $f(G_2) = \cdots = f(G_k) = 0$. Moreover, since G_1 is an induced subgraph of G, G_1 is also trivially perfect and the induction hypothesis implies that $f(G_1) \leq 1$. Therefore, $f(G) = f(G_1) + \cdots + f(G_k) \leq 1$.
Conversely, we will prove that if $f(G) \leq 1$ then G is trivially perfect, by induction on the number of vertices of G. Let G be a graph such that $f(G) \leq 1$. If G has exactly one vertex, then G is trivially perfect. Suppose now that G has at least two vertices and that for each graph G' having fewer vertices than G the fact that $f(G') \leq 1$ implies that G' is trivially perfect. Since G has at least two vertices and $f(G) \neq 2$, G is either disconnected or has disconnected complement. Suppose first that G is disconnected and let $G_1, \ldots, G_k$ be its components. By definition of f, the fact that $f(G) \leq 1$ implies that $f(G_1) \leq 1, \ldots, f(G_k) \leq 1$. Thus, the induction hypothesis ensures that each of $G_1, \ldots, G_k$ is trivially perfect and consequently $G = G_1 + \cdots + G_k$ is also trivially perfect. Suppose now that $\overline{G}$ is disconnected and let $G_1, \ldots, G_k$ be its anticomponents. The fact that $f(G) \leq 1$ together with the definition of f implies, without loss of generality, that $f(G_1) \leq 1$ and $f(G_2) = \cdots = f(G_k) = 0$. Since $f(G_1) \leq 1$ and G_1 has fewer vertices than G, the induction hypothesis ensures that G_1 is trivially perfect; i.e., G_1 is $\{P_4, C_4\}$-free. Moreover, by virtue of (1), each of $G_2, \ldots, G_k$ is a complete graph and $G = G_1 \vee G_2 \vee \cdots \vee G_k$ arises from G_1 by adding universal vertices. Since G_1 is $\{P_4, C_4\}$-free and the addition of universal vertices cannot create new induced P_4's or C_4's, G is also $\{P_4, C_4\}$-free; i.e., G is trivially perfect.

(3) *G is balanced if and only if $f(G) \leq 2$ and each graph represented by an N-node of $T(G)$ is balanced.* Suppose first that $f(G) \leq 2$ and that each graph represented by an N-node of $T(G)$ is balanced. We will prove that G is balanced by induction on the number of vertices of G. If G has only one vertex, then G is balanced. Suppose that G has at least two vertices and that, for each graph G' having fewer vertices than G, the facts that $f(G') \leq 2$ and that each graph represented by an N-node of $T(G')$ is balanced imply that G' itself is balanced. Consider first the case where G is disconnected and let $G_1, \ldots, G_k$ be its components. By the definition of f, the fact that $f(G) \leq 2$ implies that $f(G_1) \leq 2, \ldots, f(G_k) \leq 2$. Moreover, since $T(G_1), \ldots, T(G_k)$ are the subtrees of $T(G)$ rooted at the children of the root of $T(G)$, the graphs represented by the N-nodes of $T(G_i)$ are balanced, for each $i \in \{1, \ldots, k\}$. By induction hypothesis, $G_1, \ldots, G_k$ are balanced and, consequently, $G = G_1 + \cdots + G_k$ is also balanced. Consider now the case where $\overline{G}$ is disconnected and let $G_1, \ldots, G_k$ be its anticomponents. The fact that $f(G) \leq 2$ together with the definition of f imply, without loss of generality, that either $f(G_1) \leq 2$ and $f(G_2) = \cdots = f(G_k) = 0$, or $f(G_1) \leq 1$, $f(G_2) \leq 1$, and $f(G_3) = \cdots = f(G_k) = 0$. And again, since $T(G_1)$ is a subtree of $T(G)$ rooted at some child of the root of $T(G)$, the graphs represented by the N-nodes of $T(G_1)$ are balanced. By induction hypothesis and the statements (1) and (2), either G_1 is balanced and $G_2, \ldots, G_k$ are complete graphs, or G_1 and G_2 are trivially perfect and $G_3, \ldots, G_k$ are complete graphs. In either case, Theorem 2.10 implies that G is balanced. Finally, consider the

case where G and $\overline{G}$ are disconnected. In this case, G itself is the graph represented by the root r of $T(G)$ where r is an N-node and, by hypothesis, G is balanced.

For the converse, suppose now that G is balanced. Since the graphs represented by the nodes of $T(G)$ are induced subgraphs of G, all these graphs are balanced. We will prove that also $f(G) \leq 2$, by induction on the number of vertices of G. If G has only one vertex, then $f(G) = 0 \leq 2$. Assume that G has at least two vertices and that, for each graph G' having fewer vertices than G, the fact that G' is balanced implies that $f(G') \leq 2$. Consider first the case where G is disconnected and let $G_1, \ldots, G_k$ be the components of G. Since $G_1, \ldots, G_k$ are balanced (because they are induced subgraphs of G), the induction hypothesis implies that $f(G_i) \leq 2$ for each $i \in \{1, \ldots, k\}$ and $f(G) = \max\{1, f(G_1), \ldots, f(G_k)\} \leq 2$. Consider now the case where $\overline{G}$ is disconnected and let $G_1, \ldots, G_k$ be its anticomponents. By Theorem 2.10, we assume, without loss of generality, that either G_1 is balanced and $G_2, \ldots, G_k$ are complete graphs, or G_1 and G_2 are trivially perfect and $G_3, \ldots, G_k$ are complete graphs. Therefore, by inductive hypothesis and the statements (1) and (2), either $f(G_1) \leq 2$ and $f(G_2) = \cdots = f(G_k) = 0$, or $f(G_1) \leq 1$, $f(G_2) \leq 1$, and $f(G_3) = \cdots = f(G_k) = 0$. In either case, $f(G) = f(G_1) + \cdots + f(G_k) \leq 2$. Finally, if G and $\overline{G}$ are connected, then $f(G) = 2$ by definition. □

Notice that if for each node h of a modular decomposition tree $T(G)$ we denote by $f(h)$ the value $f(G[h])$, then the following holds:

$$f(h) = \begin{cases} \max\{1, f(h_1), \ldots, f(h_k)\} & \text{if } h \text{ is a P-node with children } h_1, \ldots, h_k \\ f(h_1) + \cdots + f(h_k) & \text{if } h \text{ is an S-node with children } h_1, \ldots, h_k \\ 2 & \text{if } h \text{ is an N-node} \\ 0 & \text{if } h \text{ is a leaf.} \end{cases}$$

Therefore, given the modular decomposition tree of a graph G, the values $f(h)$ for all the nodes h of $T(G)$ can be computed in $O(n)$ time by visiting the nodes of $T(G)$ in post-order (recall that $T(G)$ has less than $2n$ nodes.)

By combining all the above results, we will next show that the algorithm below recognizes balancedness of any given P_4-tidy graphs in linear time.

Algorithm 2. Decides balancedness of any given P_4-tidy graph

```
Input: A $P_4$-tidy graph $G$
Initialization: Build the modular decomposition tree of $G$
Step 1: Compute $f(h)$ for every node $h$ of $T(G)$
Step 2: if $f(h) > 2$ for some node $h$ of $T(G)$ then output '$G$ is not balanced' and stop
Step 3: Visit once each of the N-nodes $h$ of $T(G)$ (in any order), doing the following:
        if ($\pi(h)$ is isomorphic to $C_5$) or
           ($\pi(h)$ is a starfish whose vertex $h_R$ representing the head satisfies $f(h_R) > 1$
            and $\pi(h)$ has a vertex in the body representing $2K_1$) or
           ($\pi(h)$ is an urchin with at least 6 vertices)
               then output '$G$ is not balanced' and stop
Step 4: Output '$G$ is balanced'
```

Theorem 2.15. *Algorithm 2 correctly decides whether or not any given P_4-tidy graph is balanced in linear time; in fact, in $O(n)$ time once the modular decomposition tree of G is built.*

Proof. Let G be any P_4-tidy graph. Recall from Subsection 1.1 that the modular decomposition tree of G can be computed in linear time. In addition, from the discussion preceding Algorithm 2, Steps 1 and 2 take only $O(n)$ additional time. Since $T(G)$ has less than $2n$ nodes and since deciding whether or not the condition of the conditional clause of Step 3 holds for an N-node h of $T(G)$ takes $O(n_\pi(h))$ time and the sum of $n_\pi(h)$ as h ranges over the N-nodes of $T(G)$ is at most $2n$, Step 3 takes $O(n)$ time altogether. Therefore, Algorithm 2 has linear time complexity and takes only $O(n)$ time once the modular decomposition tree of G is built.

We now prove the correctness of the output. By Lemma 2.12 and Theorem 2.14, if the algorithm outputs 'G is not balanced', then the output is correct. Suppose now that the algorithms answers 'G is balanced'. This means that $f(h) \leq 2$ for each node h of $T(G)$ and that the condition in the conditional clause of Step 3 holds for no N-node h of $T(G)$. Suppose, by the way of contradiction that G is not balanced. Since $f(G) \leq 2$, Theorem 2.14 implies that there is at least one N-node of $T(G)$ representing a graph which is not balanced. Among the N-nodes of $T(G)$ representing graphs which are not balanced, let h_0 be one that minimizes the number of vertices of $G[h_0]$. Notice that if h' is any child of h_0, then Theorem 2.14 implies that h' represents a balanced graph: in fact, $f(h') \leq 2$ and, since the modular decomposition tree of $G[h']$ is the subtree of $T(G)$ rooted at h', the N-nodes h'' of $T(G[h'])$ represent balanced graphs because $G[h'']$ has fewer vertices than $G[h_0]$. Since h_0 is an N-node which is not balanced, reasoning as in the proof of Lemma 2.12, one of the following cases holds:

- $G[h_0]$ *is isomorphic to* C_5. In this case $\pi(h_0)$ is isomorphic to C_5 and the algorithm should have answered 'G is not balanced', a contradiction.
- $G[h_0]$ *is a starfish whose head is not balanced or is a fat starfish arising from it.* In this case, if h' is the child of h_0 representing the head of the starfish, then h' is a child of h_0 but represents a graph that is not balanced, a contradiction.
- $G[h_0]$ *is a fat starfish arising from a starfish whose head is not trivially perfect by substituting a vertex of the body by* $2K_1$. In this case, $\pi(h_0)$ is a starfish such that the only vertex h_R of the head satisfies $f(h_R) > 1$ (because of Theorem 2.14) and $\pi(h_0)$ has one vertex of the body representing $2K_1$. Consequently, the algorithm should have answered 'G is not balanced', a contradiction.
- $G[h_0]$ *is an urchin with at least* 6 *vertices or a fat urchin arising from it.* In this case, $\pi(h_0)$ would be an urchin with at least 6 vertices and the algorithm should have answered 'G is not balanced', a contradiction.

In all cases we reach a contradiction. These contradiction arose from assuming that there is some graph G which is not balanced but for which the algorithm answers 'G is balanced'. This completes the proof of the correctness of the algorithm. □

3. Balancedness of paw-free graphs

The *paw* is the graph that arises from a triangle by adding a pendant vertex (see Figure 4). Paw-free graphs were considered in connection with perfectness in [54], [55] and [58]. In this section, we consider balancedness of the class of paw-free graphs. We prove a minimal forbidden induced subgraph characterization of balancedness of paw-free graphs, leading to a linear-time algorithm for deciding whether or not any given paw-free graph is balanced.

Clique-perfectness of paw-free graphs was already considered in [13] and the following results were proved.

Theorem 3.1 ([13]). *For each paw-free graph G, the following assertions are equivalent:*

(1) G is clique-perfect.
(2) G is perfect.

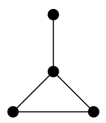

Figure 4. The paw

(3) G has no odd holes.

Moreover, it can be decided in linear time whether or not any given paw-free graph is clique-perfect.

The proof of the above theorem relies on the characterization of paw-free graphs due to Olariu [55] given below. A graph is *complete multipartite* if its vertex set can be partitioned into (possibly empty) stable sets $A_1, \ldots, A_k$ such that every vertex of A_i is adjacent to every vertex of A_j for every $i \neq j$. If, in addition, $k = 2$, then the graph is called *complete bipartite.*

Theorem 3.2 ([55]). *A graph is paw-free if and only if each of its components is either triangle-free or a complete multipartite graph.*

It is easy to derive from the above theorem that $\alpha_c(G)$ can be computed in polynomial time for any given paw-free graph G. In fact, if G is a paw-free graph and H is a component of G, then either (i) H is a triangle-free graph and $\alpha_c(H)$ coincides with the maximum size of a matching, which can be found in polynomial time [35, 42], or (ii) H is a complete multipartite graph and thus a cograph and $\alpha_c(H)$ can be computed in linear time [46]. Nevertheless, recall that $\tau_c(G)$ is NP-hard even if G is triangle-free [37] and, consequently, also even if G is a paw-free graph.

The following result gives structural characterizations of those paw-free graphs that are balanced including one by minimal forbidden induced subgraphs.

Theorem 3.3. *For each paw-free graph G, the following assertions are equivalent:*

(1) G is balanced.
(2) G is perfect and hereditary clique-Helly.
(3) G has no odd holes and contains no induced 3-pyramid.
(4) Each component of G is either bipartite or is the join of a complete bipartite graph and a complete graph.

Proof. The implications (1) $\Rightarrow$ (2) and (2) $\Rightarrow$ (3) follow from the discussion in the Introduction.

We will now prove that (3) $\Rightarrow$ (4). Let G be a paw-free graph with no odd holes and no induced 3-pyramid. Let H be any component of G. Since H has no odd holes, if H were triangle-free, then H would be bipartite, as desired. So, without loss of generality, we will assume that H is not triangle-free. By Theorem 3.2, H is a complete multipartite graph. Let $\{A_1, \ldots, A_k\}$ be a partition of the vertex set of H into nonempty stable sets such that every vertex of A_i is adjacent to every vertex of A_j for every $i \neq j$, and such that $|A_1| \geq |A_2| \geq \cdots \geq |A_k|$. Since H is not triangle-free, $k \geq 3$. Moreover, $|A_3| = |A_4| = \cdots = |A_k| = 1$ because G is 3-pyramid-free. Let $A = A_1 \cup A_2$ and $B = A_3 \cup \cdots \cup A_k$. Hence, H is the join of the complete bipartite graph $G[A]$ and the complete graph $G[B]$. This completes the proof of (3) $\Rightarrow$ (4).

Finally, we prove that (4) $\Rightarrow$ (1). Let G be a graph such that each of its components is either a bipartite graph or the join of a complete graph and a complete bipartite graph. Since bipartite graphs are balanced and the join of a complete graph and a balanced graph is balanced (by virtue of Lemma 2.9(a)), it follows that every component of G is balanced. Therefore, G is balanced. □

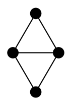

Figure 5. The diamond

The above characterization also implies a linear-time algorithm for deciding whether any given paw-free graph G is balanced because it can be decided whether or not assertion (4) of Theorem 3.3 holds in linear time: in fact, for each component H of G, we can decide in linear time whether or not either H is bipartite or the non-universal vertices of H induce a complete bipartite graph.

Corollary 3.4. *It can be decided in linear time whether or not any given paw-free graph is balanced.*

4. Clique-perfectness and balancedness of diamond-free graphs

Recall from the Introduction that forbidden induced subgraph characterizations of clique-perfectness are known for graphs belonging to one of several different graph classes. It turns out that for each of these graph classes, also a polynomial-time or even a linear-time algorithm for the recognition of clique-perfectness of graphs belonging to that class is known, with the only exception being the class of diamond-free graphs, where the *diamond* is the graph that arises from K_4 by removing one edge (see Figure 5).

Clique-perfectness of diamond-free graphs was characterized by means of forbidden induced subgraphs in [9]. The corresponding forbidden induced subgraphs are the proper odd cycles, where a cycle C is *proper* if no two consecutive vertices of C form a triangle with a third vertex of C.

Theorem 4.1 ([9]). *For each diamond-free graph G, G is clique-perfect if and only if G contains no induced proper odd cycle.*

In the same work, it is left unresolved whether there is a polynomial-time algorithm for deciding clique-perfectness of diamond-free graphs. We deduce the existence of such a polynomial-time algorithm by reducing the problem to that of deciding balancedness.

Theorem 4.2. *For each diamond-free graph G, G is clique-perfect if and only if G is balanced.*

Proof. Since balanced graphs are clique-perfect, we only need to prove that diamond-free clique-perfect graphs are balanced, or equivalently, that a diamond-free graph that is not balanced is not clique-perfect. Let G be a diamond-free graph that is not balanced. By Theorem 1.4, there is some odd cycle C of G such that for each $e \in E(C)$ there exists a maximal clique Q_e containing e but no other vertex of C. We claim that C is proper. Suppose, by the way of contradiction, that C is not proper; i.e, that there are two consecutive vertices u and v of C forming a triangle with a third vertex w of $V(C)$. By construction, Q_{uv} contains u and v but not w. Hence, since Q_{uv} is a maximal clique, there is some vertex $x \in Q_{uv}$ such that x is nonadjacent to w. In particular, $\{u, v, w, x\}$ induces a diamond in G, a contradiction. This contradiction arose from assuming that C was not proper. Thus, C is a proper odd cycle and, by virtue of Theorem 4.1, G is not clique-perfect, as desired. □

Since balanced graphs can be recognized in polynomial time, it follows that clique-perfectness of diamond-free graphs can be decided in polynomial time.

Corollary 4.3. *It can be decided in polynomial time whether or not any given diamond-free graph is clique-perfect.*

5. Further remarks

In Section 2, we studied clique-perfectness and balancedness of P_4-tidy graphs. Namely, we gave linear-time recognition algorithms and minimal forbidden induced subgraph characterizations of clique-perfectness and balancedness of P_4-tidy graphs, as well as a linear-time algorithm for computing a maximum clique-independent set and a minimum clique-transversal set of any given P_4-tidy graph. Subsection 2.1 extend results given in [46] for cographs, which where already extended for a different superclass of cographs in [51]. A key element of Subsection 2.2 is Theorem 2.10 which characterizes when the join of any two graphs is balanced and that may prove useful in the study of balancedness beyond the graph classes considered in this work. A different approach for obtaining linear-time algorithms for P_4-tidy graphs was introduced in [28] based on the fact that the class of P_4-tidy graphs has bounded *clique-width*. This approach allows for linear-time solutions for recognition and optimization problems that are expressible in terms of certain monadic second-order logic. For instance, combining the equivalence between assertions (1) and (2) of our Theorem 2.5 and the expressibility of the latter assertion in that logic, their approach gives an alternative linear-time algorithm for deciding clique-perfectness of P_4-tidy graphs. An entirely analogous situation holds for the recognition of balancedness of P_4-tidy graphs and the equivalence between assertions (1) and (3) of our Theorem 2.13 because (3) is also expressible in that logic. Moreover, the minimum clique-transversal set problem lies within the scope of their approach, which implies an alternative linear-time algorithm for finding a minimum clique-transversal set of any given P_4-tidy graph. Nevertheless, while an algorithm obtained with this approach also depends on building a modular decomposition tree of the input graph, the rest of the algorithm may account for huge constant hidden factors in the linear-time complexity (even for small clique-width bounds) [27], which is unavoidable if one wishes to get results for general monadic second-order formulas [40]. Therefore, when dealing with a specific problem, it is of interest to find algorithms that work by performing a simple post-processing of the modular decomposition tree. Following this direction, the algorithms presented in Section 2 consist in simple $O(n)$ time traversals of the modular decomposition tree of the input graph. Moreover, the fact that a maximum clique-independent set can be found in linear time for any given P_4-tidy graph (as we proved in Theorem 2.8), does not seem to follow from the approach of [28] as the maximum clique-independent set problem seems not to be directly expressible in the corresponding logic (where quantification over sets of vertex sets is not allowed).

In Section 3, we characterized those paw-free graphs that are balanced by minimal forbidden induced subgraphs, from which a linear-time recognition algorithm for balancedness of paw-free graphs follows. Analogous results on clique-perfectness of paw-free graphs have been obtained in [13].

In Section 4, we settled the computational complexity of deciding clique-perfectness of diamond-free graphs (which was left unresolved in [9]) and proved that clique-perfectness coincides with balancedness for diamond-free graphs. The class of diamond-free graphs was the only one for which a forbidden induced subgraph characterization of clique-perfectness of graphs in the class was known but the existence of a polynomial-time algorithm for deciding clique-perfectness of graphs in the class was unknown. Notice that it remains unsolved the problem of characterizing clique-perfectness (or, equivalently, balancedness) of diamond-free graphs by minimal forbidden induced subgraphs; i.e, making explicit the structure of those proper odd cycles that are minimal under taking induced subgraphs.

Acknowledgments

The authors are indebted to the referees for insightful comments, corrections, and suggestions that helped to improve this paper. The first three named authors were partially supported by UBACyT Grant 20020100100980, CONICET PIP 112-200901-00178, and ANPCyT PICT-2012-1324 (Argentina). G. Durán was also partially supported by FONDECyT Grant 1110797 and Millennium Science Institute 'Complex Engineering Systems' (Chile).

References

[1] V. Balachandran, P. Nagavamsi, and C. Pandu Rangan, *Clique transversal and clique independence on comparability graphs*, Inform. Process. Lett. 58 (1996), pp. 181–184.

[2] S. Baumann, *A linear algorithm for the homogeneous decomposition of graphs*, Report TUM M9615, Fakultät für Mathematik, Technische Universität München, Munich, Germany, 1996.

[3] C. Berge, *Färbung von Graphen, deren sämtliche beziehungsweise, deren ungerade Kreise starr sind (Zusammenfassung)*, Wiss. Z. Martin-Luther-Univ. Halle-Wittenberg, Math.-Naturwiss. Reihe 10 (1961), pp. 114–115.

[4] C. Berge, *Balanced matrices*, Math. Program. 2 (1972), pp. 19–31.

[5] C. Berge and V. Chvátal, *Introduction*, in *Topics on Perfect Graphs*, *North-Holland Mathematics Studies*, vol. 88, Noth-Holland, 1984, pp. vii–xiv.

[6] C. Berge and M. Las Vergnas, *Sur un théorème du type König pour hypergraphes*, Ann. N.Y. Acad. Sci. 175 (1970), pp. 32–40.

[7] F. Bonomo, *On subclasses and variations of perfect graphs*, Doctoral dissertation, Departamento de Computación, FCEyN, Universidad de Buenos Aires, Buenos Aires, Argentina (2005).

[8] F. Bonomo, M. Chudnovsky, and G. Durán, *Partial characterizations of clique-perfect graphs I: subclasses of claw-free graphs*, Discrete Appl. Math. 156 (2008), pp. 1058–1082.

[9] F. Bonomo, M. Chudnovsky, and G. Durán, *Partial characterizations of clique-perfect graphs II: Diamond-free and Helly circular-arc graphs*, Discrete Math. 309 (2009), pp. 3485–3499.

[10] F. Bonomo, G. Durán, M.C. Lin, and J.L. Szwarcfiter, *On balanced graphs*, Math. Program. 105 (2006), pp. 233–250.

[11] F. Bonomo, G. Durán, M. Groshaus, and J.L. Szwarcfiter, *On clique-perfect and K-perfect graphs*, Ars Combin. 80 (2006), pp. 97–112.

[12] F. Bonomo, G. Durán, M.D. Safe, and A.K. Wagler, *On minimal forbidden subgraph characterizations of balanced graphs*, Electron. Notes Discrete Math. 35 (2009), pp. 41–46.

[13] F. Bonomo, G. Durán, F. Soulignac, and G. Sueiro, *Partial characterizations of clique-perfect and coordinated graphs: Superclasses of triangle-free graphs*, Discrete Appl. Math. 157 (2009), pp. 3511–3518.

[14] F. Bonomo, G. Durán, M.D. Safe, and A.K. Wagler, *Balancedness of some subclasses of circular-arc graphs*, Electron. Notes Discrete Math. 36 (2010), pp. 1121–1128.

[15] F. Bonomo, G. Durán, M.D. Safe, and A.K. Wagler, *Clique-perfectness of complements of line graphs*, Electron. Notes Discrete Math. 37 (2011), pp. 327–332.

[16] F. Bonomo, G. Durán, M.D. Safe, and A.K. Wagler, *On minimal forbidden subgraph characterizations of balanced graphs*, Discrete Appl. Math. 161 (2013), pp. 1925–1942.

[17] A. Brandstädt, V.D. Chepoi, and F.F. Dragan, *Clique r-domination and clique r-packing problems on dually chordal graphs*, SIAM J. Discrete Math. 10 (1997), pp. 109–127.

[18] B. Buer and R.H. Möhring, *A fast algorithm for the decomposition of graphs and posets*, Math. Oper. Res. 8 (1983), pp. 170–184.

[19] M. Chang, M. Farber, and Z. Tuza, *Algorithmic aspects of neighbourhood numbers*, SIAM J. Discrete Math. 6 (1993), pp. 24–29.

[20] M.S. Chang, S.Y. Hsieh, and G.H. Chen, *Dynamic programming on distance-hereditary graphs*, in *Algorithms and Computation, 8th International Symposium, ISAAC '97, Singapore, December 17-19, 1997, Proceedings*, *Lect. Notes Comput. Sci.*, vol. 1350, Springer, 1997, pp. 344–353.

[21] M. Chudnovsky, G.P. Cornuéjols, X. Liu, P.D. Seymour, and K. Vušković, *Recognizing Berge graphs*, Combinatorica 25 (2005), pp. 143–186.

[22] M. Chudnovsky, N. Robertson, P.D. Seymour, and R. Thomas, *The strong perfect graph theorem*, Ann. Math. 164 (2006), pp. 51–229.

[23] V. Chvátal, *On certain polytopes associated with graphs*, J. Combin. Theory Ser. B 18 (1975), pp. 138–154.

[24] D. Corneil, Y. Perl, and L. Stewart, *Cographs: recognition, applications and algorithms*, Congr. Numer. 43 (1984), pp. 249–258.

[25] D.G. Corneil, H. Lerchs, and L. Stewart Burlingham, *Complement reducible graphs*, Discrete Appl. Math. 3 (1981), pp. 163–174.

[26] D.G. Corneil, Y. Perl, and L.K. Stewart, *A linear recognition algorithm for cographs*, SIAM J. Comput. 14 (1985), pp. 926–934.

[27] B. Courcelle, *A multivariate interlace polynomial and its computation for graphs of bounded clique-width*, Electron. J. Comb. 15 (2008), # R69.

[28] B. Courcelle, J.A. Makowsky, and U. Rotics, *Linear time solvable optimization problems on graphs of bounded clique-width*, Theory Comput. Syst. 33 (2000), pp. 125–150.

[29] A. Cournier and M. Habib, *A new linear algorithm for modular decomposition*, in *Trees in Algebra and Programming - CAAP'94, 19th International Colloquium, Edinburgh, U.K., April 11-13, 1994, Proceedings*, *Lect. Notes Comput. Sci.*, vol. 787, Springer, 1994, pp. 68–84.

[30] E. Dahlhaus, J. Gustedt, and R.M. McConnell, *Efficient and practical algorithms for sequential modular decomposition*, J. Algorithms 41 (2001), pp. 360–387.

[31] E. Dahlhaus, P.D. Manuel, and M. Miller, *Maximum h-colourable subgraph problem in balanced graphs*, Inform. Process. Lett. 65 (1998), pp. 301–303.

[32] G. Durán, M.C. Lin, and J.L. Szwarcfiter, *On clique-transversal and clique-independent sets*, Ann. Oper. Res. 116 (2002), pp. 71–77.

[33] G. Durán, M.C. Lin, S. Mera, and J.L. Szwarcfiter, *Algorithms for clique-independent sets on subclasses of circular-arc graphs*, Discrete Appl. Math. 154 (2006), pp. 1783–1790.

[34] G. Durán, M.C. Lin, S. Mera, and J.L. Szwarcfiter, *Algorithms for finding clique-transversals of graphs*, Ann. Oper. Res. 157 (2008), pp. 37–45.

[35] J. Edmonds, *Paths, trees, and flowers*, Can. J. Math. 17 (1965), pp. 449–467.

[36] J. Egerváry, *Matrixok kombinatorikus tulajdonságairól*, Mat. Fiz. Lapok 38 (1931), pp. 16–28.

[37] P. Erdős, T. Gallai, and Z. Tuza, *Covering the cliques of a graph with vertices*, Discrete Math. 108 (1992), pp. 279–289.

[38] M. Farber, *Characterizations of strongly chordal graphs*, Discrete Math. 43 (1983), pp. 173–189.

[39] J.L. Fouquet and V. Giakoumakis, *On semi-P_4-sparse graphs*, Discrete Math. 165–166 (1997), pp. 277–300.

[40] M. Frick and M. Grohe, *The complexity of first-order and monadic second-order logic revisited*, Ann. Pure Appl. Logic 130 (2004), pp. 3–31.

[41] D.R. Fulkerson, A.J. Hoffman, and R. Oppenheim, *On balanced matrices*, in *Pivoting and Extensions: In honor of A.W. Tucker*, M. Balinski, ed., *Mathematical Programming Study*, vol. 1, North-Holland, Amsterdam, 1974, pp. 120–133.

[42] H. Gabow and R. Tarjan, *Faster scaling algorithms for general graph-matching problems*, J. ACM 38 (1991), pp. 815–853.

[43] T. Gallai, *Transitiv orientierbare Graphen*, Acta Math. Acad. Sci. Hung. 18 (1967), pp. 25–66.

[44] V. Giakoumakis, F. Roussel, and H. Thuillier, *On P_4-tidy graphs*, Discrete Math. Theor. Comput. Sci. 1 (1997), pp. 17–41.

[45] M.C. Golumbic, *Trivially perfect graphs*, Discrete Math. 24 (1978), pp. 105–107.

[46] V. Guruswami and C. Pandu Rangan, *Algorithmic aspects of clique-transversal and clique-independent sets*, Discrete Appl. Math. 100 (2000), pp. 183–202.

[47] C.T. Hoàng, *Perfect graphs*, Ph.D. thesis, School of Computer Science, McGill University, Montreal, Canada (1985).

[48] B. Jamison and S. Olariu, *A new class of brittle graphs*, Stud. Appl. Math. 81 (1989), pp. 89–92.

[49] B. Jamison and S. Olariu, *On a unique tree representation for P_4-extendible graphs*, Discrete Appl. Math. 34 (1991), pp. 151–164.

[50] D. Kőnig, *Graphok és Matrixok*, Mat. Fiz. Lapok 38 (1931), pp. 116–119.

[51] C.M. Lee and M.S. Chang, *Distance-hereditary graphs are clique-perfect*, Discrete Appl. Math. 154 (2006), pp. 525–536.

[52] J. Lehel and Z. Tuza, *Neighborhood perfect graphs*, Discrete Math. 61 (1986), pp. 93–101.

[53] R.M. McConnell and J.P. Spinrad, *Modular decomposition and transitive orientation*, Discrete Math. 201 (1999), pp. 189–241.

[54] H. Meyniel, *The graphs whose odd cycles have at least two chords*, in *Topics on Perfect Graphs*, C.

Berge and V. Chvátal, eds., *North-Holland Mathematics Studies*, vol. 88, Noth-Holland, 1984, pp. 115–119.

[55] S. Olariu, *Paw-free graphs*, Inform. Process. Lett. 28 (1988), pp. 53–54.

[56] K. Parthasarathy and G. Ravindra, *The strong perfect-graph conjecture is true for $K_{1,3}$-free graphs*, J. Combin. Theory Ser. B 21 (1976), pp. 212–223.

[57] E. Prisner, *Hereditary clique-Helly graphs*, J. Combin. Math. Combin. Comput. 14 (1993), pp. 216–220.

[58] R.B. Sandeep, *Perfectly colorable graphs*, Inform. Process. Lett. 111 (2011), pp. 960–961.

[59] D. Seinsche, *On a property of the class of n-colorable graphs*, J. Combin. Theory Ser. B 16 (1974), pp. 191–193.

[60] M. Tedder, D. Corneil1, M. Habib, and C. Paul, *Simpler linear-time modular decomposition via recursive factorizing permutations*, in *Automata, Languages and Programming, 35th International Colloquium, ICALP 2008, Reykjavik, Iceland, July 7-11, 2008, Proceedings, Part I*, *Lect. Notes Comput. Sci.*, vol. 5125, Springer, 2008.

[61] S. Tsukiyama, M. Idle, H. Ariyoshi, and Y. Shirakawa, *A new algorithm for generating all the maximal independent sets*, SIAM J. Comput. 6 (1977), pp. 505–517.

[62] A. Tucker, *The strong perfect graph conjecture for planar graphs*, Can. J. Math. 25 (1973), pp. 103–114.

[63] A. Tucker, *Coloring perfect $(K_4 - e)$-free graphs*, J. Combin. Theory Ser. B 42 (1987), pp. 313–318.

[64] D.B. West, *Introduction to Graph Theory*, 2nd ed., Prentice-Hall (2001).

[65] G. Zambelli, *A polynomial recognition algorithm for balanced matrices*, J. Combin. Theory Ser. B 95 (2005), pp. 49–67.