Hierarchical and modularly-minimal vertex colorings*

Dulce I. Valdivia , Maribel Hernández Rosales

Departamento de Ingeniería Genética, Centro de Investigación y de Estudios Avanzados del IPN (CINVESTAV), Km. 9.6 Libramiento Norte Carretera Irapuato-León, MX-36821, Irapuato, GTO, México

Manuela Geiß

Software Competence Center Hagenberg GmbH, Softwarepark 32a, A-4232 Hagenberg, Austria

Peter F. Stadler[†]

Bioinformatics Group, Department of Computer Science & Interdisciplinary Center for Bioinformatics, Universität Leipzig, Härtelstraße 16-18, D-04107 Leipzig, Germany and Max Planck Institute for Mathematics in the Sciences, Inselstraße 22, D-04103 Leipzig, Germany and

Institute for Theoretical Chemistry, University of Vienna, Währingerstrasse 17, A-1090 Wien, Austria and

Facultad de Ciencias, Universidad Nacional de Colombia, Sede Bogotá, Colombia Santa Fe Insitute, 1399 Hyde Park Rd., Santa Fe NM 87501, USA

Marc Hellmuth

Department of Mathematics, Faculty of Science, Stockholm University, SE 106 91 Stockholm, Sweden

Received 21 June 2021, accepted 9 February 2022, published online 27 October 2022

Abstract

*Dedicated to Wilfried Imrich on occasion of his 80th birthday. In a roundabout way, we have to blame Wilfried. If he had not infected a few theoretical chemists with his enthusiasm about graphs and their products during a small seminar on Turracher Höhe in March 1988, this paper would never have been written.

This work was supported in part by the German Research Foundation (DFG, STA 850/49-1), the German Federal Ministry of Education and Research (BMBF, project no. 031A538A, de.NBI-RBC), the Mexican Consejo Nacional de Ciencia y Tecnología (CONACyT, 278966 FONCICYT2; scholarship CVU 901154), the Austrian Federal Ministries BMK and BMDW, and the Province of Upper Austria in the frame of the COMET Programme managed by FFG.

[†]Corresponding author.

Cographs are exactly the hereditarily well-colored graphs, i.e., the graphs for which a greedy vertex coloring of every induced subgraph uses only the minimally necessary number of colors $\chi(G)$. Greedy colorings are shown to be a subclass of hierarchical vertex colorings that naturally appear in phylogenetic combinatorics. The hierarchical colorings of cographs form a special class of minimal colorings. The fact that cotrees are the modular decomposition trees of cographs suggests a natural generalization: A coloring σ is modularly-minimal if $|\sigma(M)| = \chi(M)$ for every strong module M of G . We show that every graph admits a modularly minimal coloring. On cographs, furthermore, the modularly minimal colorings coincide with the hierarchical colorings. For certain hereditary graph classes efficient algorithms can be designed to compute modularly-minimal colorings.

Keywords: Proper vertex coloring, Grundy number, cographs, modular decomposition, chromatic number, P_4 -sparse.

Math. Subj. Class.: 05C15, 05C05, 90C27, 92B10

1 Introduction

Graph coloring problems appear as a natural formalization of diverse real-life applications, describing in essence a partitioning of objects into classes under a given set of constraints [17, 19]. In this contribution, we investigate a specific type of vertex coloring that naturally appears in computational biology. Biological species evolve in a tree-like fashion, resulting in a species tree whose leaves are the currently living species, while inner nodes represent the speciation events. The genome of each species comprises a set of genes that upon speciation are transmitted identically to all offsprings. In between speciations, however, the genomes may change by duplication and loss of genes. This again defines a tree, the gene tree T , whose leaves are the surviving genes and whose inner nodes represent speciation and duplication events. The gene tree can be thought of as embedded in the species tree in an order-preserving manner, see e.g. [29] for a review. In evolutionary biology it is important to distinguish whether the last common ancestor of two genes in T is a speciation or a duplication event. In the former case these two genes are called orthologs, in the latter they are paralogs [6, 7, 30]. The orthology graph G of a set of genes represents the genes as vertices and connects orthologous gene pairs by an edge, see Figure 1 for an illustrative example. It can be shown that the orthology graph is always a cograph. The gene tree, furthermore, is a refinement of the corresponding cotree [13]. The species in which a gene resides is naturally represented by a vertex coloring σ . It turns out that the gene tree and the events that it represents, impose stringent restrictions on the colorings of the cograph G that may arise in this manner [9].

These “hierarchical colorings”, which will be defined in detail in the following section, form the starting point for this contribution. As we shall see, they depend explicitly on refinements of the cotree. We therefore need to consider the set of binary cotrees instead of the unique discriminating cotree of a given cograph [5]. Binary cotrees also play a key role in characterizing other types of special colorings of cographs, see e.g. [21]. Due to the special properties of greedy coloring of cographs [4], we first explore the relationship

E-mail addresses: dulce.valdivia@cinvestav.mx (Dulce I. Valdivia), maribel.hr@cinvestav.mx (Maribel Hernández Rosales), Manuela.Geiss@scch.at (Manuela Geiß), studla@bioinf.uni-leipzig.de (Peter F. Stadler), marc.hellmuth@math.su.se (Marc Hellmuth)

Figure 1: Sketch of the evolution of “insulin genes”; cf. [14] for more details about the biological background. Extant genes form the leaves of the gene tree. Two leaves are orthologous whenever their most recent common ancestor is a speciation event (S). The graph representation of the resulting orthology relation is shown below. It is a cograph in which genes are colored based on the species in which they reside (pentagon, square and triangle). The coloring is a strictly hierarchical and modularly-minimal coloring, see text for further details.

of hierarchical colorings and greedy colorings of cographs, and show that a coloring of a cograph is a greedy coloring if and only if it is hierarchical w.r.t. all of its binary cotrees. On the other hand, a coloring is minimal on each intermediate step along a binary cotree if and only if it is hierarchical w.r.t. that binary cotree.

In Section 3 we use the fact that cotrees are the modular decomposition trees of cographs to generalize the idea of hierarchical colorings to undirected graphs in general. A coloring σ of G is *modularly-minimal* if every subgraph $G[X]$ induced by a strong module X of G is colored with the minimal possible number of colors $\chi(G[X])$. The main result of this contribution is that every graph G has a modularly-minimal coloring. For some classes of graphs, including P_4 -sparse graphs, modularly-minimal colorings can be constructed in polynomial time.

2 Cographs and their hierarchical colorings

We consider simple, finite undirected graphs $G = (V, E)$, which we call *graphs* for short. A *tree* T is an acyclic connected graph and it is *rooted* if there is a uniquely distinguished vertex called the *root* of T . A *caterpillar* is a rooted tree T such that each non-leaf vertex has exactly two children and the subgraph induced by the non-leaf vertices is a path with the root of T at one end of this path.

Let $G = (V, E)$ be a graph. A (proper vertex) coloring of G is a surjective map $\sigma: V \rightarrow S$ from the vertex set V to some non-empty set S such that $xy \in E$ implies $\sigma(x) \neq \sigma(y)$. We will often refer to such coloring as an $|S|$ -coloring and the elements in S are called *colors*. The minimum number $|S|$ of colors such that there is an $|S|$ -coloring

of G is known as the *chromatic number* $\chi(G)$. For a subset $W \subseteq V$ (resp., a subgraph H of G) we denote with $\sigma(W)$ (resp., $\sigma(H)$) the set of colors assigned to the vertices in W (resp. $V(H)$) using σ .

A *greedy coloring* of G is a proper coloring obtained by ordering the set of colors and coloring the vertices of G in a random order with the first available color. The *Grundy number* $\gamma(G)$ is the maximum number of colors required in a greedy coloring of G [4]. Obviously $\gamma(G) \geq \chi(G)$. Determining $\chi(G)$ and $\gamma(G)$ for arbitrary graphs are NP-complete problems (cf. [18] and [31], respectively). A graph G is called *well-colorable* if $\chi(G) = \gamma(G)$ [31]. It is *hereditarily well-colorable* if every induced subgraph is well-colorable. For later reference, we provide the following useful lemma:

Lemma 2.1. *Let σ be a greedy coloring of a disconnected graph G with connected components $G_1, \dots, G_k$, $k \geq 2$ and let $H = \bigcup_{i \in I} G_i$ be the disjoint union of the graphs G_i with $i \in I$ for some nonempty subset $I \subseteq \{1, \dots, k\}$. Then the restriction of σ to H is a greedy coloring of H .*

Proof. W.l.o.g. let the color set $\{1, \dots, |\sigma(G)|\}$ be naturally ordered from small to large integers. Since σ is a greedy coloring it necessarily colors every connected component G_i with colors $\{1, \dots, |\sigma(G_i)|\}$. Moreover, let us preserve the ordering $\prec_i$ on the vertices in each G_i according to the order they are visited during the greedy coloring in G . It is easy to verify that the first available color in G to color a vertex x in G_i is precisely the first available to color x when using the greedy coloring w.r.t. $\prec_i$ in G_i only. In other words, the greedy coloring can be applied independently on the connected components, which completes the proof. $\square$

Definition 2.2 ([5]). A graph G is a *cograph* if $G = K_1$, G is the disjoint union $G = \bigcup_i G_i$ of cographs G_i , or G is a join $G = \nabla_i G_i$ of cographs G_i .

Since both operations, $\bigcup_i$ and ∇_i , are commutative and associative, each cograph except the K_1 can be written as the join or disjoint union of two cographs. This recursive construction induces a rooted binary tree T , whose leaves are individual vertices corresponding to a K_1 and whose interior vertices correspond to the union and join operations. We write $L(T)$ for the leaf set and $V^0(T) := V(T) \setminus L(T)$ for the set of inner vertices of T . The set of children of a vertex u is denoted by $\text{child}(u)$. For edges $e = uv$ in T we adopt the convention that v is a child of u . We define a labeling function $t : V^0(T) \rightarrow \{0, 1\}$, where an interior vertex u of T is labeled $t(u) = 0$ if it is associated with a disjoint union, and $t(u) = 1$ for joins. We will refer to (T, t) as a *cotree*. The tree $T(u)$ denotes the subtree of T that is rooted at u . To simplify the notation we will write $G(u) := G[L(T(u))]$ for the subgraph of G induced by the vertices in $L(T(u))$. Note that $G(u)$ is the graph consisting of the single vertex u if u is a leaf of T . Furthermore, $G(u)$ is a cograph by definition.

Given a cograph G , there is a unique *discriminating cotree*¹ (T^*, t^*) in which adjacent operations are distinct, i.e., $t^*(u) \neq t^*(v)$ for all interior edges $uv \in E(T)$. The discriminating cotree (T^*, t^*) is obtained from every arbitrary binary cotree (T, t) by contracting all edges uv with $t(u) = t(v)$ into a single vertex $\widehat{uv}$ with label $t^*(\widehat{uv}) = t(u) = t(v)$. Conversely, every binary cotree of G can be obtained by replacing an inner vertex of u and its children $u_1, \dots, u_k$ by an arbitrary binary tree with root u , leaves $u_1, \dots, u_k$ and all its inner vertices w labeled by $t(w) = t^*(u)$, see e.g. [1, 5] for details.

¹In [5] the discriminating cotree is defined as the cotree associated with G . Here we call every tree (T, t) a cotree as it is always a “refinement” of some discriminating cotree that explain the same cograph [1].

It is possible to refine the discriminating cotree by subdividing a disjoint union or join into multiple disjoint unions or joins, respectively [1, 5]. It is well known that every induced subgraph of a cograph is again a cograph. A graph is a cograph if and only if it does not contain a path P_4 on four vertices as an induced subgraph [5]. The cographs are also exactly the hereditarily well-colored graphs [4]. The chromatic number of a cograph G can be computed recursively, as observed in [5, Tab.1]. Starting from $\chi(K_1) = 1$ as base case we have

$$\begin{aligned}\chi(G) &= \chi\left(\bigcup_i G_i\right) = \max_i \chi(G_i) \text{ or} \\ \chi(G) &= \chi\left(\bigtriangledown_i G_i\right) = \sum_i \chi(G_i)\end{aligned}\tag{2.1}$$

Definition 2.3. Let (H_1, σ_{H_1}) and (H_2, σ_{H_2}) be two vertex-disjoint colored graphs. Then $(H_1, \sigma_{H_1}) \nabla (H_2, \sigma_{H_2}) := (H_1 \nabla H_2, \sigma)$ and $(H_1, \sigma_{H_1}) \cup (H_2, \sigma_{H_2}) := (H_1 \cup H_2, \sigma)$ denotes their join and union, respectively, where $\sigma(x) = \sigma_{H_i}(x)$ for every $x \in V(H_i)$, $i \in \{1, 2\}$.

Hierarchically colored cographs (hc-cographs) were introduced in [9] as the colored graphs recursively defined by

(K1) $(G, \sigma) = (K_1, \sigma)$, i.e., a colored vertex, or

(K2) $(G, \sigma) = (H_1, \sigma_{H_1}) \nabla (H_2, \sigma_{H_2})$ and $\sigma(V(H_1)) \cap \sigma(V(H_2)) = \emptyset$, or

(K3) $(G, \sigma) = (H_1, \sigma_{H_1}) \cup (H_2, \sigma_{H_2})$ and $\sigma(V(H_1)) \cap \sigma(V(H_2)) \in \{\sigma(V(H_1)), \sigma(V(H_2))\}$,

where $\sigma(x) = \sigma_{H_i}(x)$ for every $x \in V(H_i)$, $i \in \{1, 2\}$ and (H_1, σ_{H_1}) and (H_2, σ_{H_2}) are hc-cographs.

Obviously, the graph G underlying an hc-cograph is a cograph. Thus, the recursive construction of an hc-cograph G implies a binary cotree (T, t) that can be constructed with a top down approach as follows: Denote the root of (T, t) by r . It is associated with the graph $G(r) = G$. In the general step we consider an induced subgraph $G(u)$ of G associated with a vertex u of T . If $G(u)$ is connected, then $t(u) = 1$ and $G(u)$ is the joint of pair of induced subgraphs $G(v_1)$ and $G(v_2)$. To identify these graphs, consider the connected components $\overline{G_1}, \dots, \overline{G_k}$ of the complement $\overline{G(u)}$ of $G(u)$. We have

$$G(u) = \overline{\bigcup_{i=1}^k \overline{G_i}} = \bigtriangledown_{i=1}^k \overline{\overline{G_i}} = \bigtriangledown_{i=1}^k G_i.\tag{2.2}$$

We therefore set $G(v_1) = G_1$ and $G(v_2) = \overline{\bigcup_{i=2}^k \overline{G_i}} = \bigtriangledown_{i=2}^k G_i$. By construction, we therefore have $G(u) = G(v_1) \nabla G(v_2)$ with disjoint color sets $\sigma(G(v_1))$ and $\sigma(G(v_2))$. If $G(u)$ is disconnected, define $t(u) = 0$, identify one of the components, say G_1 , with the smallest numbers of colors $|\sigma(G_1)|$ and set $G(v_1) = G_1$ and $G(v_2) = G(u) \setminus G(v_1)$. The fact that $G(u)$ is an hc-cograph ensures that $\sigma(G(v_1)) \subseteq \sigma(G(v_2))$. In both the connected and the disconnected case we attach v_1 and v_2 as the children of u in T . The reconstruction of (T, t) can be performed in linear time.

Definition 2.4. A coloring σ of a cograph G is a *hierarchical coloring w.r.t. the binary cotree* (T, t) if (T, t) is a cotree of G and (G, σ) is a hc-cograph recursively constructed according to (T, t) .

As noticed in [9], a coloring σ of a cograph G may be hierarchical w.r.t. a binary cotree (T, t) but not hierarchical w.r.t. another binary cotree (T', t') that yields the same cograph. An example is shown in Figure 2.

Figure 2: The induced cotree of a cograph G affects the hierarchical coloring property of σ , where $\sigma(a) = \sigma(d) \neq \sigma(b) = \sigma(c)$ and different colors are represented by different symbols ($\square$, $\circ$). In (T, t) , the first tree from left to right, (K1) – (K3) are satisfied making σ hierarchical. However, the second tree (T', t') does not satisfy (K3) since the parent node of $c \simeq K_1$ and $d \simeq K_1$ corresponds to a disjoint union operation and $\sigma(c) \cap \sigma(d) = \emptyset$. Thus, σ is not hierarchical w.r.t. (T', t') .

Every hierarchical coloring of a cograph is also a proper coloring (cf. [9, Lemma 43]). The property of being a cograph is hereditary (i.e., every induced subgraph of a cograph is a cograph). However, this is not necessarily true for hc-cographs. As an example consider the induced disconnected subgraph with vertices c and d of the hc-cograph in Figure 2 that violates (K3). Nevertheless, if G is an hc-cograph, then each of its connected components must be an hc-cograph (cf. [9, Lemma 44]). We show now that hc-cographs are always optimally colored.

Theorem 2.5. Every hierarchical coloring of a cograph G w.r.t. some binary cotree is a $\chi(G)$ -coloring.

Proof. Let $G = (V, E)$ be a cograph with binary cotree (T, t) and σ be a hierarchical coloring of G w.r.t. (T, t) . We proceed by induction w.r.t. $|V|$. The statement is trivially true for $|V| = 1$, i.e., $G = K_1$, since $\chi(K_1) = 1$. Now suppose $|V| > 1$. Thus, $G = G_1 \nabla G_2$ or $G = G_1 \cup G_2$ for some cographs $G_1 = (V_1, E_1)$ and $G_2 = (V_2, E_2)$ with $1 \leq |V_1|, |V_2| < |V|$. Since G_1 and G_2 are “explained” by their respective binary cotrees rooted at the two children of the root of (T, t) and since G is a hc-cograph, we can conclude that G_1 and G_2 satisfy (K1), (K2) and (K3); therefore, they are hc-cographs. By induction hypothesis we have $|\sigma(V_1)| = \chi(G_1)$ and $|\sigma(V_2)| = \chi(G_2)$.

First consider $G = G_1 \nabla G_2$. Since $xy \in E(G)$ for all $x \in V_1$ and $y \in V_2$, we have $\sigma(x) \neq \sigma(y)$, and hence $\sigma(V_1) \cap \sigma(V_2) = \emptyset$. Thus, $\sigma(V) = \sigma(V_1) \cup \sigma(V_2)$ and therefore,

$$|\sigma(V)| = |\sigma(V_1)| + |\sigma(V_2)| \stackrel{\text{IH}}{=} \chi(G_1) + \chi(G_2) \stackrel{\text{Equation (2.1)}}{=} \chi(G).$$

We note that the coloring condition in (K2) therefore only enforces that σ is a proper vertex coloring.

Now suppose $G = G_1 \cup G_2$. Axiom (K3) implies $|\sigma(V)| = \max(|\sigma(V_1)|, |\sigma(V_2)|)$. Hence,

$$|\sigma(V)| = \max(|\sigma(V_1)|, |\sigma(V_2)|) \stackrel{\text{IH}}{=} \max(\chi(G_1), \chi(G_2)) \stackrel{\text{Equation (2.1)}}{=} \chi(G). \quad \square$$

Since the connected components of hc-cographs are again hc-cographs, the following statement is an immediate consequence of the recursive construction of hc-cographs:

Corollary 2.6. *If σ is a hierarchical coloring of G w.r.t. the binary cotree (T, t) , then $|\sigma(G(u))| = \chi(G(u))$ for all nodes u of (T, t) and $|\sigma(G')| = \chi(G')$ for all connected components G' of G .*

As detailed in [4], we have $\chi(G) = \gamma(G)$ for cographs. Thus, it seems natural to ask whether every greedy coloring is hierarchical w.r.t. some binary cotree (T, t) . Making use of the fact that $\chi(G) = \gamma(G)$, we assume w.l.o.g. that the color set is $S = \{1, 2, \dots, \chi(G)\}$ whenever we consider greedy colorings of a cograph.

We shall say that a cograph G is a *minimal counterexample for some property $\mathcal{P}$* if (1) G does not satisfy $\mathcal{P}$ and (2) every induced subgraph of G (i.e., every “smaller” cograph) satisfies $\mathcal{P}$.

Lemma 2.7. *Let G be a cograph, (T, t) an arbitrary binary cotree for G and σ a greedy coloring of G . Then σ is a hierarchical coloring w.r.t. (T, t) .*

Proof. Assume G is a minimal counterexample, i.e., G is a minimal cograph for which a coloring σ exists that is a greedy coloring but not a hierarchical coloring. If G is connected, then either $G \simeq K_1$ or $G = \nabla_{i=1}^n G_i$ and $\sigma(V) = \bigcup_{i=1}^n \sigma(V_i)$ for some $n > 1$, i.e., (K1) or (K2) is satisfied. By assumption, σ is not a hierarchical coloring, hence σ must fail to be a hierarchical coloring on at least one of the subgraphs G_i , contradicting the assumption that G is a minimal counterexample. Thus, G cannot be connected.

Therefore, assume that $G = \bigcup_{i=1}^n G_i$ for some $n > 1$. Since G is represented by a binary cotree (T, t) , the root of T must have exactly two children u and v . Hence, we can write $G = G(u) \cup G(v)$. Since G is a minimal counterexample and since, by Lemma 2.1, σ restricted to $G(u)$, resp., $G(v)$ is a greedy coloring of $G(u)$, resp., $G(v)$, we can conclude that σ induces a hierarchical coloring on $G(u)$ and $G(v)$. However, since σ is, in particular, a greedy coloring of $G(u)$ and $G(v)$, $\sigma(V(G(u))) \subseteq \sigma(V(G(v)))$ or $\sigma(V(G(v))) \subseteq \sigma(V(G(u)))$ must hold. But this immediately implies that (G, σ) satisfies (K3) and thus σ is a hierarchical coloring of G . Therefore, G is not a minimal counterexample, which completes the proof. $\square$

Since every cograph is well-colorable we obtain as an immediate consequence

Corollary 2.8. *Every cograph has a hierarchical coloring w.r.t. each of its binary cotrees (T, t) .*

The converse of Lemma 2.7 is not true. Figure 2 shows an example of a hierarchical coloring that is not a greedy coloring.

Theorem 2.9. *A coloring σ of a cograph G is a greedy coloring if and only if it is a hierarchical coloring w.r.t. every binary cotree (T, t) of G .*

Figure 3: Both colorings **A** and **B** of $K_2 \cup K_1 \cup K_1$ are hierarchical colorings w.r.t. some cotrees. To see this, note that the coloring **A** is produced by $(K_1 \nabla K_1) \cup (K_1 \cup K_1)$, while coloring **B** is the result of $((K_1 \nabla K_1) \cup K_1) \cup K_1$. Only **A** is a greedy coloring. The coloring **B** uses different colors for the two single-vertex components and thus is not greedy. Different colors are represented by different symbols ($\square, \circ$).

Proof. By Lemma 2.7, every greedy coloring is a hierarchical coloring for every binary cotree (T, t) . Now suppose σ is a hierarchical coloring for every binary cotree (T, t) and let G be a minimal cograph for which σ is not a greedy coloring. As in the proof of Lemma 2.7, we can argue that G cannot be a minimal counterexample if G is connected: in this case, $G = \nabla_{i=1}^n G_i$ and $\sigma(V) = \bigcup_{i=1}^n \sigma(V_i)$ for all colorings, and thus σ is a greedy coloring if and only if it is a greedy coloring with disjoint color sets for each G_i . Hence, a minimal counterexample must have at least two connected components.

Let $G = \bigcup_{i=1}^n G_i$ for some $n > 1$ and define a partition of $\{1, \dots, n\}$ into sets $I_1, \dots, I_\ell$, $\ell \geq 1$, such that for every $r \in \{1, \dots, \ell\}$ we have $i, j \in I_r$ if and only if $\chi(G_i) = \chi(G_j)$. Since every $G^r := \bigcup_{i \in I_r} G_i$, $1 \leq r \leq \ell$, is a cograph, each G^r can be represented by a (not necessarily unique) binary cotree (T^r, t^r) . Note, we have $\chi(G^r) = \chi(G_i)$ for all $i \in I_r$. Now, we can construct a binary cotree (T, t) for G as follows: let T^* be a caterpillar with leaf set $L(T^*) = \{l_1, \dots, l_\ell\}$. We choose $T^* = (\dots((l_1, l_2), l_3), \dots, l_\ell)$ (in Newick format). Note, if $\ell = 1$, then $T^* \simeq K_1$. Now, the root of every tree $T^1, \dots, T^r$ is identified with a unique leaf in $L(T^*)$ such that the root of T_i is identified with $l_i \in L(T^*)$ and the root of T_j is identified with $l_j \in L(T^*)$, where $i < j$ if and only if $\chi(G^i) < \chi(G^j)$. This yields the tree T . The labeling t for (T, t) is provided by keeping the labels of each (T^r, t^r) and by labeling all other inner vertices of T by 0. It is easy to see that (T, t) is a binary cotree for G . By assumption, σ is hierarchical w.r.t. (T, t) . We denote by $C(T^*) \subseteq V(T)$ the set of inner vertices of T^* . Since σ is hierarchical w.r.t. (T, t) and thus in particular w.r.t. any subtree (T^r, t^r) , we have $\sigma(V(G_i)) \cap \sigma(V(G_j)) \in \{\sigma(V(G_i)), \sigma(V(G_j))\}$ for any $i, j \in I_r$, $1 \leq r \leq \ell$, by (K3). Hence, as $\chi(G^r) = \chi(G_i) = \chi(G_j)$, it must necessarily hold $\sigma(V(G_i)) = \sigma(V(G_j))$ for all $i, j \in I_r$, i.e., all connected components G_i with the same chromatic number are colored by the same color set. By construction, at every node $v \in C(T^*)$ of the caterpillar structure, with children v' and v'' , the components $G' := G(v')$ and $G'' := G(v'')$ satisfy $\chi(G') < \chi(G'')$. Invoking (K3) we therefore have $\sigma(G') \subset \sigma(G'')$ and $G^* := G(v)$ is colored by the color set $\sigma(G^*) = \sigma(G'')$. These set inclusions therefore imply a linear ordering of the colors such that colors in $\sigma(G')$ come before those in $\sigma(G'') \setminus \sigma(G')$. Thus σ is a greedy coloring provided that the restriction of σ to each of the connected components G_i of G is a greedy coloring, which is true due to the assumption that G is a minimal counterexample. Thus no minimal counterexample exists, and the coloring σ is indeed a greedy coloring of G . $\square$

Not every minimal coloring of a cograph is hierarchical w.r.t. some binary cotree. For instance, if $G = G_1 \cup G_2$ is the disjoint union of two connected graphs G_1 and G_2 , then

it suffices that $\chi(G_1) < |\sigma(V_1)| = \chi(G)$. That is, we may use more colors than necessary on G_1 . Note that for every cotree (T, t) of G there is a vertex u in T such that $G_1 = G(u)$. Hence, for every cotree we have $\chi(G(u)) < |\sigma(G(u))|$ with $G(u) = G_1$. Contraposition of Corollary 2.6 implies that σ cannot be a hierarchical coloring of G w.r.t. any cotree (T, t) of G . In the following we restrict our attention to those coloring that satisfy the necessary conditions of Corollary 2.6, which is specified in the next

Definition 2.10. Let G be a cograph with cotree (T, t) . A coloring σ is (T, t) -minimal if $|\sigma(G(u))| = \chi(G(u))$ for every vertex $u \in V(T)$.

Note, every (T, t) -minimal coloring of a cograph G is always a $\chi(G)$ -coloring, since $|\sigma(G(r))| = \chi(G(r)) = \chi(G)$ with r being the root of T .

Theorem 2.11. A coloring σ of cograph G is (T, t) -minimal for a binary cotree (T, t) if and only if it is hierarchical w.r.t. (T, t) .

Proof. By Corollary 2.6, σ is (T, t) -minimal if it is hierarchical w.r.t. (T, t) .

Now suppose there is a minimal cograph G with a coloring σ that is (T, t) -minimal but not hierarchical w.r.t. (T, t) . If G is connected, then $G = \nabla_{i=1}^n G_i$ for some $n \geq 2$ and the restrictions of σ to the connected components G_i use disjoint color sets. Hence, σ is a hierarchical coloring whenever the restriction to each G_i is a hierarchical coloring. Thus a minimal counterexample cannot be connected. Now suppose $G = \cup_{i=1}^n G_i$ for some $n \geq 2$. Since (G, σ) is by assumption a minimal counterexample, each connected component (G_i, σ_i) is a hierarchical cograph (w.r.t. subtrees of (T, t)). Since (T, t) is binary, its root u has two children u_1 and u_2 that correspond to $G_1 = G(u_1)$ and $G_2 = G(u_2)$, resp., such that $G = G_1 \cup G_2$. By Equation (2.1), we can choose the notation such that $\chi(G) = \chi(G_1)$. By definition, σ induces a coloring σ_1 on G_1 and σ_2 on G_2 . Clearly, each coloring σ_i is $(T(u_i), t|_{T(u_i)})$ -minimal. Since G is a minimal counterexample, σ_1 and σ_2 are both hierarchical colorings of G_1 and G_2 , respectively, in other words (G_1, σ_1) and (G_2, σ_2) are hc-cographs w.r.t. subtrees of (T, t) that are rooted at u_1 and u_2 , respectively. Moreover, $\chi(G) = \chi(G_1)$ implies $\sigma_2(V(G_2)) \subseteq \sigma_1(V(G_1))$. In summary, therefore, $(G, \sigma) = (G_1, \sigma_1) \cup (G_2, \sigma_2)$ satisfies (K3), thus it is a cograph with a hierarchical coloring σ . Hence, there cannot exist a minimal cograph with a coloring σ that is (T, t) -minimal but not a hierarchical coloring w.r.t. (T, t) . $\square$

From Theorem 2.11 and Corollary 2.8 we obtain

Corollary 2.12. For every cograph G and every binary cotree (T, t) of G there is a (T, t) -minimal coloring.

A $\chi(G)$ -coloring of a cograph can be obtained in linear time [16]. Given a cograph G and its corresponding binary cotree (T, t) it is also not difficult to construct a (T, t) -minimal coloring. In the following we assume that T is stored with arbitrary but fixed sibling order to ensure that postorder traversal [3] of T is well-defined.

Theorem 2.13. Given a binary cotree (T, t) , Algorithm 1 returns a (T, t) -minimal coloring σ of the corresponding cograph G in $O(|V(T)|) = O(|V(G)|)$ time.

Proof. We need to show that the Algorithm constructs a $(T(u), t)$ -minimal coloring for every vertex u of T . For the leaves this is trivial. We claim that Algorithm 1 correctly generates a $(T(u), t)$ -minimal coloring at each inner vertex u of (T, t) provided the colorings

Algorithm 1 (T, t) -minimal coloring of $L(T)$ for a binary cotree (T, t) .

Require: Binary cotree (T, t)

```

1 initialize coloring  $\sigma$  s.t. all  $v \in V(G)$  have different colors
2 initialize list of colors  $S(v) = [\sigma(v)]$  for all  $v \in V(G)$ 
3 initialize color forest  $F$  with isolated vertices  $\sigma(v)$ ,  $v \in V$ 
4 for all  $u \in V^0(T)$  in postorder do
5   Let  $v, w$  be the children of  $u$  in  $T$ 
6   if  $t(u) = 0$  then
7      $x \leftarrow \arg \max\{\chi(G(v)), \chi(G(w))\}$ ;  $y \leftarrow \{v, w\} \setminus \{x\}$ 
8      $S(u) \leftarrow S(x)$ ;  $\chi(G(u)) \leftarrow \chi(G(x))$ 
9     randomly choose an injective map  $\phi : S(y) \rightarrow S(u)$ 
10    for all  $s \in S(y)$  do
11      insert edge  $(\phi(s), s)$  into  $F$ 
12    else
13       $S(u) = S(v) \cup S(w)$ ;  $\chi(G(u)) = \chi(G(v)) + \chi(G(w))$ 
14    for all subtrees  $F'$  in  $F$  do
15       $r \leftarrow$  unique source of  $F'$ 
16      for all  $s \in V(F')$  do
17         $c(s) = r$   $\triangleright c$  identifies  $s$  with “final color”  $r$ 
18    for all  $v \in L(T)$  do
19       $\sigma(v) \leftarrow c(\sigma(v))$ 

```

at the two children v and w of u are $(T(v), t)$ -minimal and $(T(w), t)$ -minimal, respectively. We first note that postorder traversal of T ensures that a vertex u processed only after all its descendants have been processed. It is clear that the color sets $G(v)$ and $G(w)$ are disjoint while u is processed. It follows immediately, therefore, that the coloring of $G(u)$ is $(T(u), t)$ -minimal if $t(u) = 1$, and it suffices to keep track of the list of colors $S(u)$ used to color $L(T(u))$ and thus $G(u)$.

For $t(u) = 0$, Algorithm 1 determines the child $x \in \{v, w\}$ with the larger chromatic number $\chi(G(x)) = \max\{\chi(G(v)), \chi(G(w))\}$ among the two children of u . Since $t(u) = 0$, $G(u)$ is the disjoint union of $G(v)$ and $G(w)$. Hence, $\chi(G(u)) = \chi(G(x))$ and we can color $G(u)$ with the color set $S(x) = \sigma(G(x))$ of $G(x)$. The other child, y , must use a subset of the colors $S(x)$. Thus, we re-color $S(y) = \sigma(G(y))$ using an injective function $\phi : S(y) \rightarrow S(x)$ and keep track of the assignment $s \mapsto \phi(s)$ in the forest F . After recoloring, we have $\sigma(G(y)) \subseteq \sigma(G(x))$, and thus the resulting coloring of $G(u)$ is again $(T(u), t)$ -minimal. Since each leaf v is trivially $(T(v), t)$ -minimally colored, we conclude that Algorithm 1 is correct. The coloring so far is only represented indirectly by that sequence of color re-assignments, i.e., the edges $(\phi(s), s)$ pointing from the “new color” $\phi(s)$ to the previous one. Since each color s is mapped to another at most once and not used again later in the postorder traversal, these edges define a forest F . A color s has an in-edge $(\phi(s), s)$ in F if and only if it was reassigned at some point. This is the case if and only if $s \notin S(\text{root}(T))$, i.e., the sources (roots of subtrees F' of F are exactly the colors that a used to color the leaves $v \in L(T)$, i.e., the vertices of the cograph G . More precisely, the color eventually assigned to $v \in L(T)$ is the color at the root of the subtree of F in which the initial color $\sigma(v)$ resides. The last part of the algorithm makes this assignment explicit.

In order to estimate the running time for an input graph G of with $n := |V(G)|$ vertices, we first note that the initialization steps in Line 1 – 3 can be performed in $O(n)$ total time. To evaluate the effort required for the iteration over the inner vertices of T , we note that the color lists $S(u)$ are only needed while a vertex is processed. Thus it suffices to point to them. Similarly, the concatenation of $S(v) \cup S(w)$ can be achieved with a single pointer operation. Hence, the latter two tasks can be done in constant time. A random injective function $\phi: S(y) \rightarrow S(u)$ can be constructed using $O(|S(y)|)$ operations. The same is true for the insertion of the $|S(y)|$ edges into the forest F . Since initially all vertex colors are distinct and the colors in $S(y)$ are no longer used in the ancestors of y , the total effort for re-assigning colors is in $O(n)$. Once we reach the root of T , the forest F consists of $\chi(G)$ trees and a total of n vertices. The assignment of the root-color to each vertex s below the root of a subtree of F requires $O(n)$ operations for the entire forest. Finally, the initial colors $\sigma(v)$ are replaced by the root-color $c(\sigma(v))$ in $O(n)$ total time. Hence, the total effort for Algorithm 1 is bounded by $O(n)$. $\square$

Algorithm 1 is not primarily intended as a replacement for greedy coloring. Since the binary cotree (T, t) serves as model of the evolution of a gene family, hc-colorings for a given cotree are of use as random models of evolutionary scenarios e.g. for benchmarking purposes. It has the advantage that the cograph (orthology relation) does not need to be constructed explicitly. Nevertheless, it may be useful also if larger samples of colorings for the same given cograph are of interest. Each greedy coloring requires $O(|V| + |E|)$ operations since it operates on the cograph G . Expend the same effort, one can construct (T, t) , and Algorithm 1 now produces a random coloring in $O(|V|)$ time. If we insist that the injective color assignments ϕ are chosen uniformly, a reservoir sampling algorithm can be used that introduces an additional factor $O(\log |V|)$ [20].

The recursive structure of hc-cographs can also be used to count the number of distinct hierarchical colorings of G w.r.t. a given binary cotree (T, t) . To specify the term “distinct coloring”, let us define for a graph G with coloring σ the equivalence relation Σ_σ by putting $(u, v) \in \Sigma_\sigma$ precisely if $\sigma(u) = \sigma(v)$ and let $\Sigma_\sigma[v]$ denote the equivalence class of Σ_σ that contains v . Two colorings σ and σ' of G are *distinct* if, for some vertex $v \in V(G)$ it holds that $\Sigma_\sigma[v] \neq \Sigma_{\sigma'}[v]$. For example, in complete graphs K_n we have $|\Sigma_\sigma[v]| = 1$ for all vertices $v \in V(K_n)$ and thus no two colorings of a K_n are distinct. For a graph G , we denote with $Z(G)$ the number of hierarchical colorings of G . If u is a leaf, then $Z(G(u)) = 1$, otherwise, u has exactly two children, $\text{child}(u) = \{v_1, v_2\}$. For $t(u) = 1$, we have $Z(G(u)) = Z(G(v_1)) \cdot Z(G(v_2))$ since the color sets are disjoint. If $t(u) = 0$, assume, w.l.o.g. $s_1 := |\sigma(G(v_1))| \leq |\sigma(G(v_2))| =: s_2$, $Z(G(u)) = Z(G(v_1)) \cdot Z(G(v_2)) \cdot g(s_1, s_2)$, where $g(s_1, s_2)$ is the number of injections between a set of size s_1 into a set of size s_2 , i.e., $g(s_1, s_2) = \binom{s_2}{s_1} s_1!$. The number of colorings can now be computed by bottom-up traversal on T .

The total number of hierarchical colorings can be obtained by considering a caterpillar tree for the step-wise union of connected components. For each connected component G_i with $\chi(G_i) = s_i$, and $s = \max_i s_i$ there are $\binom{s}{s_i}$ choices of the colors, i.e., $g(s, s_i)$ injections and thus $Z(G) = \prod_i (g(s, s_i) \cdot Z(G_i))$ colorings. We note in passing that the chromatic polynomial of a cograph, and thus the number of colorings using the minimal number of colors, can also be computed in polynomial time [22]. We leave it as an open problem to establish the connection between the hierarchical colorings and the chromatic polynomial of a cograph.

3 Modularly-minimal colorings

The definition of hierarchical colorings in the previous section crucially depends on the structure of cographs and their associated cotrees. In order to extend the concept to arbitrary graphs, we first need some additional notation. We denote the neighborhood of a vertex $v \in V$ by $N(v)$ and recall:

Definition 3.1 ([8]). Let $G = (V, E)$ be an arbitrary graph. A non-empty vertex set $X \subseteq V$ is a *module* of G if, for every $y \in V \setminus X$, either $N(y) \cap X = \emptyset$ or $X \subseteq N(y)$ is true. A module M is *strong* if it does not overlap with any other module M' , i.e., if $M \cap M' \in \{M, M', \emptyset\}$.

In particular V and the singletons $\{v\}$, $v \in V$ are strong modules. The *maximal modular partition* of a graph $G = (V, E)$ with $|V| > 1$, denoted by $\text{Pmax}(G) = \{M_1, \dots, M_k\}$, is a partition of the vertex set V into inclusion-maximal strong modules distinct from V . In particular, if G or $\overline{G}$ are disconnected, then the respective connected components are the elements of $\text{Pmax}(G)$. By slight abuse of notation, we say that an induced subgraph H of G is a module of G , if $V(H)$ is a module of G . Similarly, if M is a module of G we also refer to the induced subgraph $G[M]$ as a module of G .

The *modular decomposition* [8] of a graph G is based on $\text{Pmax}(G)$ and recursively decomposes G into strong modules in $\text{Pmax}(G)$. This recursive decomposition of G corresponds to the modular decomposition (MD) tree of G , that is, a vertex-labeled tree $(\tilde{T}, \tilde{t})$ where each of its vertices is associated with a strong module X of G and a label t that distinguishes the three cases:

- (i) *parallel*: the induced subgraph of G by X , $G[X]$, is disconnected,
- (ii) *series*: $\overline{G[X]}$ is disconnected, and
- (iii) *prime*: both $G[X]$ and $\overline{G[X]}$ are connected.

We write $\mathcal{M}(G)$ for the set of strong modules of G . The maximal modular partition $\text{Pmax}(G)$, the MD tree, and the set of strong modules $\mathcal{M}(G)$ of G are unique [12]. The modular decomposition of G , and thus its set of strong modules, can be obtained in linear time [24].

We first note for later reference that all proper colorings necessarily satisfy a generalization of (K2) for series nodes of the MD tree. By abuse of notation, we will call a node u of the MD tree $(\tilde{T}, \tilde{t})$ *parallel*, *series*, or *prime* if the corresponding vertex set $L(\tilde{T}(u))$ is a parallel, series or prime module of G .

Lemma 3.2. *Let σ be a proper coloring of a graph G and let X be a strong series module of G with $\text{Pmax}(G[X]) = \{M_1, \dots, M_k\}$. Then $\sigma(M_i) \cap \sigma(M_j) = \emptyset$ whenever $i \neq j$.*

Proof. Since the M_i are strong modules of X , each $x \in M_i$ is adjacent to every $y \in X \setminus M_i$. Since σ is a proper coloring, we have $\sigma(x) \neq \sigma(y)$, i.e., no color appearing in M_i can appear elsewhere in X . $\square$

A graph is a cograph if and only if all nodes in its MD tree are series or parallel [5]. Since a cograph is either a K_1 or it can be written as $G = G_1 \cup G_2$ or $G = G_1 \nabla G_2$, both G_1 and G_2 are modules of G . It immediately follows that for every binary cotree (T, t) , the vertex sets $L(T(u)) = V(G(u))$ are modules of G for all vertices u in T . In general, however, these modules are not strong.

Lemma 3.3. *Let (T, t) be a binary cotree of a cograph G . Then, every strong module of G is an induced subgraph $G(u)$ for some vertex u in T .*

Proof. Let M be a strong module of G and assume, for contradiction, that $M \neq V(G(u)) = L(T(u))$ for all u in T . Then there is a vertex v in T such that $M \subsetneq V(G(v))$ and $V(G(v))$ is an inclusion-minimal set belonging to a node v of T containing M . Let v_1 and v_2 be the two children of v in T . By construction, M intersects both $V(G(v_1))$ and $V(G(v_2))$, and is properly contained in their union $V(G(v))$. However, since M is strong, it can neither overlap $V(G(v_1))$ nor $V(G(v_2))$ and thus, $V(G(v_1)), V(G(v_2)) \subseteq M$. Therefore, $M = V(G(v_1)) \cup V(G(v_2)) = V(G(v))$; a contradiction. Thus, $M = V(G(u))$ for some vertex u in T . $\square$

The discriminating cotree (T^*, t^*) of a cograph G coincides with its modular decomposition tree $(\tilde{T}, \tilde{t})$ [5]. Together with Lemma 3.2, this suggests to generalize the concept of hierarchical colorings to arbitrary graphs.

Definition 3.4. A coloring σ of G is *hierarchical* if, for every disconnected strong module $G[X]$ of G there is a strong module $M_j \in \text{Pmax}(G[X])$ such that $\sigma(M_i) \subseteq \sigma(M_j)$ for all $M_i \in \text{Pmax}(G[X])$.

Property (K3), furthermore suggests a stronger variant:

Definition 3.5. A coloring σ of G is *strictly hierarchical* if for every disconnected strong module $G[X]$ of G we have $\sigma(M_i) \cap \sigma(M_j) \in \{\sigma(M_i), \sigma(M_j)\}$ for all $M_i, M_j \in \text{Pmax}(G[X])$.

If σ is a strictly hierarchical coloring, then a module M_j whose color set has maximum size, satisfies $\sigma(M_i) \subseteq \sigma(M_j)$ for all $i \in \{1, \dots, k\}$. Thus strictly hierarchical implies hierarchical. The converse, however, is not always true, since $\sigma(M_i) \subseteq \sigma(M_j)$ for all $M_i \in \text{Pmax}(G[X])$ does not prevent the modules distinct from M_j from having overlapping color sets.

Theorem 3.6. *Every graph G has a hierarchical and a strictly hierarchical coloring.*

Proof. Since strictly hierarchical implies hierarchical, it suffices to show that $G = (V, E)$ has a strictly hierarchical coloring. We proceed with induction on $|V|$. Clearly, the single vertex graph K_1 has a strictly hierarchical coloring. Now suppose that every graph with less than $|V|$ vertices has a strictly hierarchical coloring. $\text{Pmax}(G) = \{M_1, \dots, M_k\}$. By induction hypothesis, each $G[M_i]$ has a strictly hierarchical coloring σ_i .

If V is a series or prime module, then the colorings $\sigma_1, \dots, \sigma_k$ can be chosen w.l.o.g. to use pairwise disjoint color sets and thus, easily extend to a proper coloring σ of G . Due to the hierarchical structure of strong modules, every strong module of G must be contained in one of the M_i , $1 \leq i \leq k$. Since, for every $G[M_i]$, the coloring σ restricted to $G[M_i]$ is a strictly hierarchical coloring and since Definition 3.5 imposes no further conditions on prime and series modules, the graph G has a strictly hierarchical coloring.

Otherwise, if V is a parallel module, then every σ_i can be chosen, w.l.o.g., to use only color sets $\{1, \dots, s_i\}$ with $s_i = |\sigma_i(M_i)|$. Since V is a parallel module, there are no edges between distinct M_i and M_j . Hence, the colorings $\sigma_1, \dots, \sigma_k$ easily extend to a proper coloring σ of G . Furthermore, it obviously satisfies $\sigma(M_i) \subseteq \sigma(M_j)$ if and only if $s_i \leq s_j$, and thus $\sigma(M_i) \cap \sigma(M_j) \in \{\sigma(M_i), \sigma(M_j)\}$. This and the fact that σ restricted to each $G[M_i]$ is a strictly hierarchical coloring, implies that σ is a strictly hierarchical coloring of G . $\square$

We next show that the (strictly) hierarchical colorings are a direct generalization of the hierarchical colorings of cographs. To this end, we need the following notation. Let G be a cograph with modular decomposition tree $(\tilde{T}, \tilde{t})$. We will consider inner nodes u in $\tilde{T}$ whose children u_i are given by $G(u_i) = G[M_i]$ for $\{M_1, \dots, M_k\} = \text{Pmax}(G(u))$. We then replace every node u of $(\tilde{T}, \tilde{t})$ and its children $u_1, \dots, u_k$ by (specified) binary trees (T_u, t_u) with root u and leaves $u_1, \dots, u_k$. If u is series (resp. parallel) we put $t_u(v) = 1$ (resp. $t_u(v) = 0$) for all inner vertices v in T_u . It is an easy exercise to verify that the resulting tree (T, t) is indeed a cotree of G .

Lemma 3.7. *Let G be a cograph and σ a coloring of G . Then σ is hierarchical if and only if it is hierarchical w.r.t. some binary cotree (T, t) of G . Moreover, σ is strictly hierarchical if and only if it is hierarchical w.r.t. all cotrees of G .*

Proof. Suppose that σ is hierarchical. If u is a series node, the color sets are disjoint and we have $G(u) = \nabla_{i=1}^k G(u_i)$. Hence, for any binary tree (T_u, t_u) its inner nodes represent joins and clearly satisfy (K2). Next, consider a parallel node u in $\tilde{T}$. Hence, there is at least one child of u , say u_1 , such that $\sigma(M_i) \subseteq \sigma(M_1)$ for all $1 \leq i \leq k$. Thus we can use a caterpillar corresponding to the “pairwise” cograph structure $((\dots((G[M_1] \cup G[M_{i_2}]) \cup G[M_{i_3}]) \dots) \cup G[M_{i_k}])$ with $\{i_2, \dots, i_k\} = \{2, \dots, k\}$ arbitrarily chosen. In the resulting tree, the root u and all its newly constructed inner nodes are labeled 0. Clearly this satisfies (K3) in every step. Taken together, these constructions turn $(\tilde{T}, \tilde{t})$ into a binary cotree (T, t) that such G becomes an hc-cograph w.r.t. (T, t) , and thus σ is hierarchical w.r.t. (T, t) .

Now, suppose that σ is strictly hierarchical. If u is a series node, then we can use exactly the same arguments as above to conclude that, after replacing all series nodes u by arbitrary binary trees (T_u, t_u) , the resulting tree satisfies (K2).

Next, suppose that u is a parallel node and let (T_u, t_u) be an arbitrary binary tree. We first show that for every inner vertex w of T_u we have $\sigma(V(G(w))) = \sigma(M_j)$ for some $j \in \{1, \dots, k\}$. So let w be an arbitrary vertex. Let $I \subseteq \{1, \dots, k\}$ be a maximal subset such that w is an ancestor of u_i for all $i \in I$. Since σ is strictly hierarchical and thus hierarchical, there is a $j \in I$ such that $\sigma(M_i) \subseteq \sigma(M_j)$ for all $i \in I$. Note, $G(w) = G[M]$ with $M = \cup_{i \in I} M_i$. Taken the latter two arguments together, we can conclude that $\sigma(G(w)) = \sigma(M) = \sigma(M_j)$. The latter is, in particular, also true for the children w_1 and w_2 of w , i.e., $\sigma(G(w_1)) = \sigma(M_{i_1})$ and $\sigma(G(w_2)) = \sigma(M_{i_2})$ for some $i_1, i_2 \in I$. This and the fact that σ is strictly hierarchical implies that $\sigma(G(w_1)) \cap \sigma(G(w_2)) \in \{\sigma(G(w_1)), \sigma(G(w_2))\}$. Note, w corresponds to $G(w) = G(w_1) \cup G(w_2)$. Taken the latter two arguments together, implies that (K3) holds for every inner vertex of (T_u, t_u) .

In summary, we can therefore replace $(\tilde{T}, \tilde{t})$ by an arbitrary binary tree (T, t) that refines $(\tilde{T}, \tilde{t})$, i.e., (T, t) can be obtained from $(\tilde{T}, \tilde{t})$ by a sequence of edge-contractions. Thus σ is a hierarchical coloring of G w.r.t. every cotree (T, t) of G .

Now consider the reverse implications. The modular decomposition tree $(\tilde{T}, \tilde{t})$ is obtained from any cotree (T, t) of G by stepwise contraction of all non-discriminating edges uv i.e., those with $t(u) = t(v)$, into a new vertex w_{uv} with label $\tilde{t}(w_{uv}) = t(u)$ (cf. [1]). The vertices of $\tilde{T}$ are exactly the strong modules of G . Again, we consider inner nodes u in $\tilde{T}$ whose children u_i are given by $G(u_i) = G[M_i]$ for $\{M_1, \dots, M_k\} = \text{Pmax}(G(u))$. By construction, every inner vertex v on the path between u_i and u in T is labeled $t(v) = t(u)$. Since the definition of (strictly) hierarchical coloring does not impose constraints on series nodes u (in which case $G(u)$ is connected), there is nothing to show for the case $t(u) = 1$. Hence, suppose that $t(u) = 0$.

Suppose now σ is a hierarchical coloring w.r.t. some binary cotree (T, t) . Let v be such an inner vertex on the path between u_i and u . Clearly, $\sigma(G(v))$ is the union of the color set of some of the modules $M_1, \dots, M_k$. This and the fact that (K3) is satisfied for every vertex in T implies $\sigma(G(v)) = \sigma(G(u_j))$ for some successor u_j of v . Since the latter is, in particular, true for u , we have $\sigma(G(u)) = \sigma(G(u_j)) = \sigma(M_j)$ for some $j \in \{1, \dots, k\}$. This and $\sigma(M_i) \subseteq \sigma(G(u))$ implies that σ is hierarchical.

Finally, suppose σ is a hierarchical coloring w.r.t. every binary cotree (T, t) , that is the disjoint union of the $G(u_i)$ to obtain $G(u)$ is constructed in an arbitrary order. Therefore, for every pair $u_i \neq u_j$ of children of u in $\tilde{T}$ there is a binary cotree in which u_i and u_j have a common parent. Therefore, by (K3), we have $\sigma(M_i) \subseteq \sigma(M_j)$ or $\sigma(M_j) \subseteq \sigma(M_i)$ for all $1 \leq i < j \leq k$, and thus σ is strictly hierarchical. $\square$

As an immediate consequence, we can restate Theorem 2.9 in the following form:

Corollary 3.8. *Let G be a cograph. Then, σ is a greedy coloring if and only if it is strictly hierarchical.*

Definitions 3.4 and 3.5 impose no conditions on the prime nodes in the MD tree. Motivated by the equivalence of (T, t) -minimal and hierarchical colorings of cographs, it seems natural to consider colorings in which all strong modules are minimally colored.

Definition 3.9. A coloring σ of G is *modularly-minimal* if it satisfies $|\sigma(G[X])| = \chi(G[X])$ for all $X \in \mathcal{M}(G)$.

Since $V(G) \in \mathcal{M}(G)$, every modularly-minimal coloring is a $\chi(G)$ -coloring. The converse is not always true. As an example, consider the disconnected graph $G = K_3 \cup P_3$ whose components (and thus strong modules) are isomorphic to a K_3 and an induced path P_3 on three vertices. Clearly, $\chi(G) = 3$. A 3-coloring σ of G that uses all three colors for the P_3 is still minimal, but not modularly-minimal since $\chi(P_3) = 2$.

Lemma 3.10. *Every modularly-minimal coloring of a graph is hierarchical.*

Proof. For every graph G with proper coloring σ and connected components G_i it holds that $\chi(G) = \max_i \chi(G_i)$. Hence, there is a connected component G_j with $\chi(G) = \chi(G_j)$ and thus $\sigma(G) = \sigma(G_j)$. Now assume that σ is a modularly-minimal coloring of G . Then for every parallel module X with $\text{Pmax}(G[X]) = \{M_1, \dots, M_k\}$ it holds that $\chi(G[X]) = \chi(G[M_j]) = |\sigma(M_j)|$ and thus $\sigma(X) = \sigma(M_j)$ for some $j \in \{1, \dots, k\}$. Therefore, $\sigma(M_i) \subseteq \sigma(M_j)$ for all $i \in \{1, \dots, k\}$, and thus, σ is hierarchical. $\square$

However, not every modularly-minimal coloring is strictly hierarchical. As an example, consider the cograph $G = K_4 \cup K_2 \cup K_2$. Its strong modules are the singletons $\{v\}$, $v \in V$, the set V and the vertex sets of its connected components. We have $\chi(G) = \chi(K_4) = 4$ and $\chi(K_2) = 2$. Consider a coloring σ in which the two copies of K_2 are colored $\{1, 2\}$ and $\{3, 4\}$ respectively. Clearly σ is modularly-minimal and hierarchical w.r.t. the binary cotree $(K_4 \cup K_2) \cup K_2$ but not w.r.t. the alternative binary cotree $(K_2 \cup K_2) \cup K_4$. In the latter case, (K3) is violated.

Theorem 3.11. *Let G be a cograph with binary cotree (T, t) and σ be a coloring of G . The following statements are equivalent:*

- (1) σ is a hierarchical coloring w.r.t. (T, t)

- (2) σ is (T, t) -minimal
- (3) σ is modularly-minimal.
- (4) σ is hierarchical.

Proof. The equivalence (1) and (2) is provided by Theorem 2.11, the equivalence (1) and (4) is provided by Lemma 3.7. We show first that (1) implies (3). Suppose that σ is a hierarchical coloring w.r.t. the binary cotree (T, t) of G . By Lemma 3.3, all strong modules of G belong to some vertex in T . By Theorem 2.11, σ is (T, t) -minimal. The latter two arguments imply that $|\sigma(G[M])| = \chi(G[M])$ for all strong modules of G . Hence, σ is modularly-minimal. Finally, (3) implies (4) by Lemma 3.10, which completes the proof. $\square$

The tight connection between hierarchical and modularly-minimal colorings for graphs prompts the question whether minimal colorings like (strictly) hierarchical colorings can be constructed for all graphs.

Theorem 3.12. *Every graph admits a modularly-minimal coloring.*

Proof. We proceed by induction on the number of vertices. Obviously, K_1 has a modularly-minimal coloring. Let $G = (V, E)$ be a graph and suppose that every graph with less than $|V|$ vertices has a modularly-minimal coloring. Let $\text{Pmax}(G) = \{M_1, \dots, M_k\}$ be the (unique) maximal modular partition of G . Let σ be a $\chi(G)$ -coloring of G .

By induction hypothesis, every graph $G[M_i]$ has a modularly-minimal coloring. Since $\chi(G[M_i]) \leq |\sigma(G[M_i])|$, we can reuse the colors in every $\sigma(G[M_i])$ to obtain a $\chi(G[M_i])$ -coloring of $G[M_i]$ that is modularly-minimal for each $G[M_i]$. This results in a new coloring σ^* of G .

We show that σ^* is a proper coloring of G . Let uv be an edge of G . If u, v reside in the same module $M_i \in \text{Pmax}(G)$, then we have $\sigma^*(u) \neq \sigma^*(v)$ as every module of $\text{Pmax}(G)$ is modularly-minimal colored. If u, v reside in the distinct modules $M_i, M_j \in \text{Pmax}(G)$ then all vertices in M_i are adjacent to all vertices in M_j , which implies $\sigma(G[M_i]) \cap \sigma(G[M_j]) = \emptyset$. Since the recoloring σ^* uses only those colors in each module that already appeared, we have $\sigma^*(G[M_i]) \cap \sigma^*(G[M_j]) = \emptyset$. Hence, $\sigma^*(u) \neq \sigma^*(v)$, i.e., σ^* is proper. In particular, we have not introduced new colors and thus, σ^* remains a $\chi(G)$ -coloring.

Because of the hierarchical structure of strong modules, every strong module X of G distinct from V is contained in M_i for some $M_i \in \text{Pmax}(G)$ and thus satisfies, by construction, $|\sigma^*(G[X])| = \chi(G[X])$. For the strong module V we have $|\sigma^*(G[V])| = \chi(G)$. Thus σ^* is a modularly-minimal coloring of G . $\square$

Corollary 3.13. *Every graph admits a modularly-minimal strictly hierarchical coloring.*

Proof. By Theorem 3.12, every graph G admits a modularly-minimal coloring σ , that is, by Lemma 3.10, a hierarchical coloring of G . For every parallel module X we define an arbitrary order on the color set $\sigma(X)$. As outlined in the proof of Lemma 3.10 every module $M_i \in \text{Pmax}(G[X])$ is colored with $\chi(G[M_i])$ colors of $\sigma(M_i) \subseteq \sigma(X) = \sigma(M_j)$ for some module $M_j \in \text{Pmax}(G[X])$. Similar to the proof of Theorem 3.6 we recolor each module M_i with colors $1, \dots, \chi(M_i)$. The resulting coloring σ' is still modularly-minimal and satisfies $\sigma(M_i) \subseteq \sigma(M_j)$ whenever $\chi(G[M_i]) \leq \chi(G[M_j])$. It follows that σ' is strictly hierarchical. $\square$

Algorithm 2 Modularly-minimal coloring a graph G with MD tree $(\tilde{T}, \tilde{t})$.

Require: Graph G and MD tree $(\tilde{T}, \tilde{t})$

```

1 Initialize a coloring  $\sigma$  s.t. all  $v \in V(G)$  have different colors
2 for all  $u \in V^0(T)$  in postorder do
3   if  $u$  is parallel then
4      $\mathcal{G} \leftarrow \{G(w) : w \in \text{child}(u)\}$ 
5      $G^* \leftarrow \arg \max_{w \in \text{child}(u)} |\chi(G(w))|$ 
6      $S \leftarrow \sigma(V(G^*))$ 
7     for  $H \in \mathcal{G} \setminus \{G^*\}$  do
8       randomly choose an injective map  $\phi : \sigma(H) \rightarrow S$ 
9       for all  $x \in H$  do
10         $\sigma(x) \leftarrow \phi(\sigma(x))$ 
11   else if  $u$  is prime then
12     Construct a modularly-minimal coloring of  $G(u)$  with colors contained in
         $\sigma(G(u))$  and adjust  $\sigma$  accordingly

```

Modularly-minimal colorings provide a useful device to design efficient coloring algorithms for certain hereditary graph classes. More precisely, a polynomial-time coloring algorithm can be devised for every hereditary graph class for which a minimal coloring can be constructed efficiently given minimal colorings of its strong modules. Such an algorithm is outlined in Algorithm 2 and used to show that so-called P_4 -sparse graphs can be modularly-minimal colored in polynomial time.

Lemma 3.14. *Given a graph G and corresponding MD tree $(\tilde{T}, \tilde{t})$, Algorithm 2 returns a modularly-minimal and thus a $\chi(G)$ -coloring σ .*

Proof. We need to show that the Algorithm constructs a modularly-minimal coloring for every vertex u of T . For the leaves this is trivial. Postorder traversal of T furthermore ensures that upon processing an inner node u , colorings for $G(v)$ and $G(w)$ for both children v and w of u have already been computed. By analogous arguments as in the proof of Theorem 2.13, we can conclude that the coloring of $G(u)$ is modularly-minimal provided that u is a series or parallel node. Moreover, if $G(u)$ is prime, then the entire subgraph $G(u)$ is modularly-minimally colored with colors used for the subgraphs $G(u_i)$, where $u_i \in \text{child}(u)$. A modularly-minimal coloring exists by Theorem 3.12. In particular, since the color sets of $\sigma(G(u_i))$ and $\sigma(G(u_j))$ of any two distinct children $u_i, u_j \in \text{child}(u)$ are disjoint while u is processed, $\sigma(G(u))$ contains enough colors to obtain a $\chi(G(u))$ coloring, which completes the proof. $\square$

Note that constructing a modularly-minimal coloring for prime nodes u is a hard problem in general. However, if it can be solved efficiently for each prime node, then Algorithm 2 provides an efficient algorithm to construct a modularly-minimal coloring of G . Of course, this is trivially true for cographs since these lack prime nodes in the MD tree. The following example of P_4 -sparse graphs shows that there are also interesting graph classes for which this is possible in a nontrivial manner.

Modularly-minimal colorings of P_4 -sparse graphs

A graph G is P_4 -sparse if each of its five-vertex induced subgraphs contains at most one induced path P_4 [15]. They form a class of frequently studied generalization of cographs. As shown in [15], G is P_4 -sparse if and only if every induced subgraph H of G with at least two vertices satisfies exactly one of the following conditions:

- (i) H is disconnected,
- (ii) $\overline{H}$ is disconnected, or
- (iii) H is a spider (defined below).

In particular, therefore, every prime node in the modular decomposition tree of G is a spider, and its children except $G[R]$ (unless $R = \emptyset$) are leaves corresponding to the vertices of the body and the legs. In order to construct a modularly-minimal coloring of a P_4 -sparse graph, it therefore suffices to find a suitable way of coloring spiders.

Definition 3.15 ([15, 25]). A graph G is a *thin spider* if its vertex set can be partitioned into three sets V_K , V_S , and V_R such that

- (i) $K := G[V_K]$ is a clique;
- (ii) $S := G[V_S]$ is edge-less;
- (iii) $|V_K| = |V_S| \geq 2$;
- (iv) every vertex in V_R is adjacent to all vertices of V_K and none of the vertices of V_S ; and
- (v) each vertex in V_K has a unique neighbor in V_S and *vice versa*.

A graph G is a *thick spider* if its complement $\overline{G}$ is a thin spider.

The subgraphs K , S , and R are usually referred to as the *body*, the set of *legs*, and *head*, resp., of a thin spider. By definition the head R of a thin spider is a strong module in G , while all other strong modules of a thin spider are trivial, that is, the vertices in $K \cup S$ are all trivial modules (due to the 1-1 correspondence between the vertices in K and S which precludes that any larger subset of K or S could be a module), see also [11]. The same is true for thick spiders since (strong) modules are preserved under complementation.

It is not difficult to determine the chromatic number of a spider and to construct a corresponding coloring:

Lemma 3.16. *If G is a thin spider then $\chi(G) = \chi(R) + |K|$, and if G is a thick spider then $\chi(G) = \chi(\overline{R}) + |S|$, where $\chi(R) = 0$ if the head R is empty.*

Proof. First, let G be a thin spider with non-empty head R . Then $\chi(G) \geq |K| + \chi(R)$ since by definition every vertex of K is connected with every vertex of R , i.e., the color sets of R and K must be disjoint. Since the body K is a clique, it requires $\chi(K) = |K|$ colors. Each leg $x \in S$ is connected to a unique vertex $x' \in K$ of the body. Since $|S| = |K| \geq 2$, one can always choose a color in $\sigma(K)$ different from $\sigma(x')$ to color x , hence $|K| + \chi(R)$ are sufficient to color G . If the head R is empty, the $|K|$ colors of the body are sufficient by the same argument.

Now suppose G is a thick spider, i.e., $\overline{G}$ is a thin spider. Thinking of K , S , and R as induced subgraphs of G , we note that $\overline{S}$ is a clique in $\overline{G}$, $\overline{K}$ is an independent set in $\overline{G}$, and every vertex of $\overline{R}$ is connected to every vertex of $\overline{S}$ and none of the vertices in $\overline{K}$. Thus $\chi(\overline{G}) \geq |S| + \chi(\overline{R})$. Each vertex of $x \in \overline{K}$ is adjacent to all but one vertex in $\overline{S}$, which we call x' . Since there is a 1-1 correspondence between x and x' , we can give them same color, i.e., $|S| + \chi(\overline{R})$ are sufficient. Again, if the head R is empty, the $|S|$ colors of the legs are sufficient. $\square$

Corollary 3.17. *A coloring of a P_4 -sparse graph is modularly-minimal if each spider (K, S, R) in its MD tree is colored such that $\sigma(S) = \sigma(K)$ and σ is a modularly-minimal coloring of its head.*

Proof. By definition, every prime vertex u in the MD tree of a P_4 -sparse graphs must be a spider $G(u) = (K, S, R)$. Therefore, by Lemma 3.14 and the construction in Algorithm 2, it suffices to show that a P_4 -sparse graph is modularly-minimal colored, if each spider $G(u) = (K, S, R)$ in its MD tree is modularly-minimal colored. Note, $\text{Pmax}(G(u)) = \cup_{v \in K \cup S} \{\{v\}\} \cup \{R\}$, that is, each child of u corresponds to a vertex $v \in K \cup S$ and R . By construction all color sets of the children of u are pairwise disjoint. Using that $|K| = |S|$, we can obtain $\sigma(S) = \sigma(K)$ by reusing $\sigma(K)$ to color S as described in the proof of Lemma 3.16. Furthermore, σ is a modularly-minimal coloring of the head R . As outlined in the proof of Lemma 3.16, this yields a minimal coloring of the spider (K, S, R) . Since $\text{Pmax}(G(u)) = \cup_{v \in K \cup S} \{\{v\}\} \cup \{R\}$, every strong module in the spider $G(u)$ is modularly-minimally colored, which completes the proof. $\square$

It is well known that a $\chi(G)$ -coloring of a P_4 -sparse graph can be constructed in linear time [16]. As a consequence of Corollary 3.17, this is also true for modularly-minimal colorings.

Corollary 3.18. *A modularly-minimal coloring of P_4 -sparse graph $G = (V, E)$ can be computed in $O(|V| + |E|)$ time.*

Proof. The modular decomposition of G can be obtained in $O(|V| + |E|)$ time [24]. Replace Line 12 in Algorithm 2 by the construction as in Corollary 3.17 and consider a vertex u in the MD tree. If u is series, there is nothing to do. If u is parallel, the recoloring of the connected components can be performed in $O(|V(G(u))|)$ time. If u is prime, i.e., $G(u)$ is spider, we only need to recolor its legs with the colors of the body as in the proof of Lemma 3.16, which can also be done in $O(|V(G(u))|)$ time. Each node u of the MD tree corresponds to strong module $M = V(G(u)) \in \mathcal{M}(G)$ and thus can be handled in $O(|M|)$ time. By [23, Theorem 22] we have $O(\sum_{M \in \mathcal{M}(G)} |M|) \leq 2|E| + 3|V|$, and thus the total effort is $O(|V| + |E|)$, that is, linear in the size of G . $\square$

4 Concluding remarks

The existence of modularly-minimal colorings can serve as guiding principle for recursive algorithms to compute the chromatic number. In particular, whenever it is possible to efficiently compute the chromatic number of a prime module X from the quotient graph $G[X]/\text{Pmax}(G[X])$ and chromatic number of the modules $M \in \text{Pmax}(G[X])$, one obtains an efficient algorithm for this purpose. The virtue of Theorem 3.12 in this context is to relieve the need for controlling the colorings of the modules. Examples of such a construction are the recursive computation of the chromatic number for (P_5, gem) -free graphs

[2] and P_4 -tidy graphs [10]. Here, we provided an algorithm to compute a minimal coloring of P_4 -sparse graphs in polynomial time. The approach may be useful more generally for vertex colorings of graphs with forbidden subgraphs surveyed in [28]. It is tempting to consider modular-minimality as a guiding principle for other optimization problems.

Moreover, the general idea of (strictly) hierarchical or modularly-minimal coloring is not restricted to the modular decomposition. Many interesting classes of graphs admit recursive constructions [26, 27]. For every graph class of this type, one can ask whether minimal colorings can be constructed from minimal colorings of the constituents, i.e., whether recursively minimal colorings exist.

ORCID iDs

Dulce I. Valdivia <https://orcid.org/0000-0002-7702-652X>

Maribel Hernández Rosales <https://orcid.org/0000-0003-1878-1223>

Manuela Geiß <https://orcid.org/0000-0002-4134-6820>

Peter F. Stadler <https://orcid.org/0000-0002-5016-5191>

Marc Hellmuth <https://orcid.org/0000-0002-1620-5508>

References

- [1] S. Böcker and A. W. M. Dress, Recovering symbolically dated, rooted trees from symbolic ultrametrics, *Adv. Math.* **138** (1998), 105–125, doi:10.1006/aima.1998.1743, <https://doi.org/10.1006/aima.1998.1743>.
- [2] H. L. Bodlaender, A. Brandstädt, D. Kratsch, M. Rao and J. Spinrad, On algorithms for (P_5, gem) -free graphs, *Theor. Comp. Sci.* **349** (2005), 2–21, doi:10.1016/j.tcs.2005.09.026, <https://doi.org/10.1016/j.tcs.2005.09.026>.
- [3] W. A. Burkhard, Nonrecursive traversals of trees, *Computer J.* **18** (1975), 227–230, doi:10.1093/comjnl/18.3.227, <https://doi.org/10.1093/comjnl/18.3.227>.
- [4] C. A. Christen and S. M. Selkow, Some perfect coloring properties of graphs, *J. Comb. Theory, Ser. B* **27** (1979), 49–59, doi:10.1016/0095-8956(79)90067-4, [https://doi.org/10.1016/0095-8956\(79\)90067-4](https://doi.org/10.1016/0095-8956(79)90067-4).
- [5] D. G. Corneil, H. Lerchs and L. Steward Burlingham, Complement reducible graphs, *Discr. Appl. Math.* **3** (1981), 163–174, doi:10.1016/0166-218X(81)90013-5, [https://doi.org/10.1016/0166-218X\(81\)90013-5](https://doi.org/10.1016/0166-218X(81)90013-5).
- [6] W. M. Fitch, Distinguishing homologous from analogous proteins, *Syst Zool* **19** (1970), 99–113, doi:10.2307/2412448, <https://doi.org/10.2307/2412448>.
- [7] T. Gabaldón and E. Koonin, Functional and evolutionary implications of gene orthology, *Nat. Rev. Genet.* **14** (2013), 360–366, doi:10.1038/nrg3456, <https://doi.org/10.1038/nrg3456>.
- [8] T. Gallai, Transitiv orientierbare graphen, *Acta Math Acad. Sci. Hungaricae* **18** (1967), 25–66, doi:10.1007/bf02020961, <https://doi.org/10.1007/bf02020961>.
- [9] M. Geiß, P. F. Stadler and M. Hellmuth, Reciprocal best match graphs, *J. Math. Biol.* **80** (2020), 865–953, doi:10.1007/s00285-019-01444-2, <https://doi.org/10.1007/s00285-019-01444-2>.
- [10] V. Giakoumakis, F. Roussel and H. Thuillier, On P_4 -tidy graphs, *Discr. Math. Theor. Comp. Sci.* **1** (1997), 17–41, doi:10.46298/dmtcs.232, <https://doi.org/10.46298/dmtcs.232>.

- [11] V. Giakoumakis and J.-M. Vanherpe, On extended P_4 -reducible and extended P_4 -sparse graphs, *Theor. Comp. Sci.* **180** (1997), 269–286, doi:10.1016/S0304-3975(96)00220-4, [https://doi.org/10.1016/S0304-3975\(96\)00220-4](https://doi.org/10.1016/S0304-3975(96)00220-4).
- [12] M. Habib and C. Paul, A survey of the algorithmic aspects of modular decomposition, *Comp. Sci. Review* **4** (2010), 41–59, doi:10.1016/j.cosrev.2010.01.001, <https://doi.org/10.1016/j.cosrev.2010.01.001>.
- [13] M. Hellmuth, M. Hernandez-Rosales, K. T. Huber, V. Moulton, P. F. Stadler and N. Wieseke, Orthology relations, symbolic ultrametrics, and cographs, *J. Math. Biol.* **66** (2013), 399–420, doi:10.1007/s00285-012-0525-x, <https://doi.org/10.1007/s00285-012-0525-x>.
- [14] D. M. Irwin, Evolution of the insulin gene: Changes in gene number, sequence, and processing, *Front. Endocrinol.* **12** (2021), 649255, doi:10.3389/fendo.2021.649255, <https://doi.org/10.3389/fendo.2021.649255>.
- [15] B. Jamison and S. Olariu, A tree representation for P_4 -sparse graphs, *Discrete Appl. Math.* **35** (1992), 115–129, doi:10.1016/0166-218X(92)90036-A, [https://doi.org/10.1016/0166-218X\(92\)90036-A](https://doi.org/10.1016/0166-218X(92)90036-A).
- [16] B. Jamison and S. Olariu, Linear time optimization algorithms for P_4 -sparse graphs, *Discr. Appl. Math.* **61** (1995), 155–175, doi:10.1016/0166-218X(94)00012-3, [https://doi.org/10.1016/0166-218X\(94\)00012-3](https://doi.org/10.1016/0166-218X(94)00012-3).
- [17] T. R. Jensen and B. Toft, *Graph Coloring Problems*, John Wiley & Sons, New York, 1995, doi:10.1002/9781118032497, <https://doi.org/10.1002/9781118032497>.
- [18] R. M. Karp, Reducibility Among Combinatorial Problems, in: R. E. Miller, J. W. Thatcher and J. D. Bohlinger (eds.), *Complexity of Computer Computations*, Plenum, New York, pp. 85–103, 1972, doi:10.1007/978-1-4684-2001-2_9, https://doi.org/10.1007/978-1-4684-2001-2_9.
- [19] R. M. R. Lewis, *A Guide to Graph Colouring: Algorithms and Applications*, Springer, Heidelberg, 2016, doi:10.1007/978-3-319-25730-3, <https://doi.org/10.1007/978-3-319-25730-3>.
- [20] K.-H. Li, Reservoir-sampling algorithms of time complexity $O(n(1+\log(N/n)))$, *ACM Trans. Math. Software* **20** (1994), 481–493, doi:10.1145/198429.198435, <https://doi.org/10.1145/198429.198435>.
- [21] A. Lyons, Acyclic and star colorings of cographs, *Discr. Appl. Math.* **159** (2011), 1842–1850, doi:10.1016/j.dam.2011.04.011, <https://doi.org/10.1016/j.dam.2011.04.011>.
- [22] J. A. Makowsky, U. Rotics, I. Averbouch and B. Godlin, Computing graph polynomials on graphs of bounded clique-width, in: F. V. Fomin (ed.), *Graph-Theoretic Concepts in Computer Science*, Springer, Berlin, Heidelberg, volume 4271 of *Lecture Notes Comp. Sci.*, 2006 pp. 191–204, doi:10.1007/11917496_18, https://doi.org/10.1007/11917496_18.
- [23] R. M. McConnell and F. de Montgolfier, Linear-time modular decomposition of directed graphs, *Discr. Appl. Math.* **145** (2005), 198–209, doi:10.1016/j.dam.2004.02.017, <https://doi.org/10.1016/j.dam.2004.02.017>.
- [24] R. M. McConnell and J. P. Spinrad, Modular decomposition and transitive orientation, *Discrete Math.* **201** (1999), 189–241, doi:10.1016/S0012-365X(98)00319-7, [https://doi.org/10.1016/S0012-365X\(98\)00319-7](https://doi.org/10.1016/S0012-365X(98)00319-7).
- [25] J. Nastos and Y. Gao, Bounded search tree algorithms for parameterized cograph deletion: Efficient branching rules by exploiting structures of special graph classes, *Discr. Math. Algor. Appl.* **4** (2012), 1250008, doi:10.1142/S1793830912500085, <https://doi.org/10.1142/S1793830912500085>.

- [26] M. Noy and A. Ribó, Recursively constructible families of graphs, *Adv. Appl. Math.* **32** (2004), 350–363, doi:10.1016/S0196-8858(03)00088-5, [https://doi.org/10.1016/S0196-8858\(03\)00088-5](https://doi.org/10.1016/S0196-8858(03)00088-5).
- [27] A. Proskurowski, Recursive graphs, recursive labelings and shortest paths, *SIAM J. Comput.* **10** (1981), 391–397, doi:10.1137/0210028, <https://doi.org/10.1137/0210028>.
- [28] B. Randerath and I. Schiermeyer, Vertex colouring and forbidden subgraphs – A survey, *Graphs Comb.* **20** (2004), 1–40, doi:10.1007/s00373-003-0540-1, <https://doi.org/10.1007/s00373-003-0540-1>.
- [29] L. Y. Rusin, E. Lyubetskaya, K. Y. Gorbunov and V. Lyubetsky, Reconciliation of gene and species trees, *BioMed Res Int.* **2014** (2014), 642089, doi:10.1155/2014/642089, <https://doi.org/10.1155/2014/642089>.
- [30] R. L. Tatusov, E. V. Koonin and D. J. Lipman, A genomic perspective on protein families, *Science* **278** (1997), 631–637, doi:10.1126/science.278.5338.631, <https://doi.org/10.1126/science.278.5338.631>.
- [31] M. Zaker, Results on the Grundy chromatic number of graphs, *Discr. Math.* **306** (2006), 3166–3173, doi:10.1016/j.disc.2005.06.044, <https://doi.org/10.1016/j.disc.2005.06.044>.