(Nearly-)tight bounds on the contiguity and linearity of cographs

Christophe Crespelle, Philippe Gambette

(Nearly-)Tight Bounds on the Contiguity and Linearity of Cographs

Christophe Crespelle[a], Philippe Gambette[b]

[a] *Université Claude Bernard Lyon 1, DANTE/INRIA, LIP UMR CNRS 5668, ENS de Lyon, Université de Lyon.*
[b] *Université Paris-Est, LIGM UMR CNRS 8049, Université Paris-Est Marne-la-Vallée, 5 boulevard Descartes, 77420 Champs-sur-Marne, France.*

Abstract

In this paper we show that the contiguity and linearity of cographs on n vertices are both $O(\log n)$. Moreover, we show that this bound is tight for contiguity as there exists a family of cographs on n vertices whose contiguity is $\Omega(\log n)$. We also provide an $\Omega(\log n / \log\log n)$ lower bound on the maximum linearity of cographs on n vertices. As a by-product of our proofs, we obtain a min-max theorem, which is worth of interest in itself, stating equality between the rank of a tree and the minimum height of one of its path partitions.

Keywords: Contiguity, Linearity, Cographs, Graph encoding

Introduction

One of the most widely used operation in graph algorithms is the *neighborhood query*: given a vertex x of a graph G, one wants to obtain the list of neighbors of x in G. The classical data structure that allows to do so is the adjacency lists. It stores a graph G in $O(n+m)$ space, where n is the number of vertices of G and m its number of edges, and answers an adjacency query on any vertex x in $O(d)$ time, where d is the degree of vertex x.

This time complexity is optimal, as soon as one wants to produce the list of neighbors of x. On the other hand, in the last decades, huge amounts of data organized in the form of graphs or networks have appeared in many contexts such as genomics, biology, physics, linguistics, computer science, transportation and industry. In the same time, the need, for industrials and academics, to algorithmically treat this data in order to extract relevant information has grown in the same proportions. For these applications dealing with very large graphs, a space complexity of $O(n+m)$ is often very limiting. Therefore, as pointed out by [1], finding compact representations of a graph providing optimal time

Email addresses: christophe.crespelle@inria.fr (Christophe Crespelle), philippe.gambette@univ-mlv.fr (Philippe Gambette)

neighborhood queries is a crucial issue in practice. Such representations allow to store the graph entirely in memory while preserving the complexity of algorithms using neighborhood queries. The conjunction of these two advantages has great impact on the running time of algorithms managing large amount of data.

One possible way to store a graph G in a very compact way and preserve the complexity of neighborhood queries is to find an order σ on the vertices of G such that the neighborhood of each vertex x of G is an interval in σ. In this way, one can store the list of vertices of the graph in the order defined by σ and assign two pointers to each vertex: one toward its first neighbor in σ and one toward its last neighbor in σ. Therefore, one can answer adjacency queries on vertex x simply by listing the vertices appearing in σ between its first and last pointer. It must be clear that such an order on the vertices of G does not exist for all graphs G. Nevertheless, this idea turns out to be quite efficient in practice and some compression techniques are precisely based on it [2, 3]: they try to find orders of the vertices that group the neighborhoods together, as much as possible.

Then, a natural way to relax the constraints of the problem so that it admits a solution for a larger class of graphs is to allow the neighborhood of each vertex to be split in at most k intervals in order σ. The minimum value of k which makes possible to encode the graph in this way is a parameter called *contiguity* [4].

Another possible way of generalization is to use at most k orders $\sigma_1, \ldots, \sigma_k$ on the vertices of G such that the neighborhood of each vertex is the union of exactly one interval taken in each of the k orders. This defines a parameter called the *linearity* of G [5] which is always less than or equal to the contiguity of G.

Only little is known about these two graph parameters. For example, the classes of graphs having contiguity (resp. linearity) at most k, where k is an integer greater than 1, have not been characterized, even for $k = 2$. Actually, the nature of these parameters seems to be quite different from those of the other classical graph parameters (e.g. based on decompositions), and it turns out that the classes of graphs known to have a nice behavior with regard to classical parameters, such as e.g. cographs (see Section 1), interval graphs and permutation graphs (see [6] for definitions of these classes), do not necessarily have a nice behavior with regard to contiguity and linearity. Thus, studying these two parameters is of key interest both for their practical implications and for their theoretical properties. In this paper, we aim at determining what is, in the worst case, the contiguity and linearity of cographs, that is, in other words, the maximum contiguity (resp. linearity) of cographs on n vertices.

Let us mention that in the following, all graphs are undirected, simple and loopless. For each of the two parameters we consider here, namely contiguity and linearity, there are actually two slightly different notions depending on whether one considers open neighborhoods (i.e. the set of neighbors of the vertex x, excluding the vertex x itself) or closed neighborhoods (i.e. the set of neighbors of the vertex x plus the vertex x). The corresponding notions are called *open contiguity* (resp. *open linearity*) and *closed contiguity* (resp.

closed linearity). For contiguity, it does not make a big difference, as the open and closed parameters differ by at most one. For linearity, the situation is slightly different as it is not known whether the open linearity may exceed the closed linearity by more than one. But anyway, these two parameters are still equivalent in the sense that they differ at most by a multiplicative constant (at most two in this case). This is enough for us, as we consider the asymptotic behavior and we do not take into account multiplicative constants. Regarding the comparison between contiguity and linearity, it is straightforward to see that linearity is always less than contiguity (since one may duplicate the same order k times), but it is an open question to determine whether it can be significantly less for some graphs or not.

Related works

As we mentioned earlier, only little is known about contiguity and linearity of graphs. In the context of $0-1$ matrices, [4, 7] studied closed contiguity and showed that deciding whether an arbitrary graph has closed contiguity at most k is NP-complete for any fixed $k \geq 2$. For arbitrary graphs, [8] (Corollary 3.4) gave an upper bound on the value of closed contiguity which is $n/4 + O(\sqrt{n \log n})$.

Regarding graphs with bounded contiguity or linearity, only the class of graphs having closed contiguity 1 (or equivalently closed linearity 1) and the class of graphs having open contiguity 1 (or equivalently open linearity 1) have been characterized. The former is the class of proper (or unit) interval graphs [9], which is the subclass of interval graphs that admit a model whose intervals all have the same length. The latter class, i.e. graphs having *open contiguity* 1, is referred to as the class of biconvex graphs [6]. Biconvex graphs are a subclass of bipartite graphs properly containing bipartite permutation graphs, and which have, in terms of dimension theory for posets, dimension at most 3.

Finally, let us mention that [5] showed that both contiguity and linearity (closed and open) are unbounded for interval graphs as well as for permutation graphs, and that the four parameters can be up to $\Omega(\log n / \log \log n)$, where n is the number of vertices of the graph.

Our results

In this paper we show that, even for the class of cographs, the contiguity and the linearity are unbounded. Nevertheless, we show that they are both dominated by $O(\log n)$ for a cograph on n vertices. To this purpose, we show that the contiguity and linearity of a cograph G do not exceed the maximum height of a complete binary tree included (as a minor) in the cotree of G. As a by-product of our proof, we also establish a min-max theorem which is worth of interest in itself: the maximum height of a complete binary tree included (as a minor) in a tree T (known as the *rank* of tree T [10, 11]) is equal to the mimimun height of a *path partition* of T (see Definition 7).

Moreover, we exhibit a family of cographs $(G_n)_{n \in \mathbb{N}}$ on n vertices whose asymptotic contiguity is $\Omega(\log n)$, which implies that our $O(\log n)$ bound is tight. For the case of linearity, we exhibit a family of cographs whose asymptotic

linearity is $\Omega(\log n / \log\log n)$. This leaves open the question of determining whether the linearity of cographs may be up to $\Omega(\log n)$ or not. In addition, it should be noted that we show that these lower bounds on the contiguity and linearity of cographs in the worst case are both reached on the families of cographs whose cotree is a complete binary tree. This emphasizes the question of knowing whether these two parameters are equivalent, for a cograph G, to some function of the maximum height of a complete binary tree included as a minor in the cotree of G.

Outline of the paper.

Section 1 gives the definitions and notations we use in the following. Section 2 gives a min-max theorem on the rank of the tree, which contains the main idea of our approach for proving the upper bounds. In Section 3, we show that the closed contiguity and closed linearity of a cograph are both $O(\log n)$. Finally, in Section 4, we show that complete binary cotrees have contiguity $\Omega(\log n)$ and linearity $\Omega(\log n / \log\log n)$.

1. Preliminaries.

All graphs considered here are finite, undirected, simple and loopless. In the following, G denotes a graph, V (or $V(G)$ to avoid ambiguity) denotes its vertex set and E (or $E(G)$) its edge set. And we use the notation $G = (V, E)$. The set of subsets of V is denoted by 2^V. Throughout the paper, n stands for the cardinality $|V|$ of the vertex set of G. An edge between vertices x and y will be arbitrarily denoted by xy or yx. The (open) neighborhood of x is denoted by $N(x)$ (or $N_G(x)$ to avoid ambiguity) and its closed neighborhood by $N[x] = N(x) \cup \{x\}$. The subgraph of G induced by the set of vertices $X \subseteq V$ is denoted by $G[X] = (X, \{xy \in E \mid x, y \in X\})$.

There are several characterizations of the class of *cographs*. They are often defined as the graphs that do not admit the P_4 (path on 4 vertices) as induced subgraph. Equivalently, they are the graphs obtained from a single vertex under the closure of the parallel composition and the series composition. The parallel composition of two graphs $G_1 = (V_1, E_1)$ and $G_2 = (V_2, E_2)$ is the disjoint union of G_1 and G_2, i.e. the graph $G_{par} = \big(V_1 \cup V_2, E_1 \cup E_2\big)$. The series composition of two graphs G_1 and G_2 is the disjoint union of G_1 and G_2 plus all possible edges from a vertex of G_1 to one of G_2, i.e. the graph $G_{ser}\big(V_1 \cup V_2, E_1 \cup E_2 \cup \{xy, x \in V_1, y \in V_2\}\big)$.

This gives a very nice representation of the graphs in the class. One can represent a cograph G by a tree whose leaves are the vertices of the graph and whose internal nodes (non-leaf nodes) are labeled P, for parallel, or S, for series, corresponding to the operations used in the construction of G. It is straightforward to see that it is always possible to find such a labeled tree T representing G such that every internal node has at least two children, no two parallel nodes are adjacent in T and no two series nodes are adjacent. This tree T is unique and is called the *cotree* of G. It is well-known that the cotree of G defined in

this way is nothing else but the modular decomposition tree of G (see [12] for a survey on modular decomposition). And this is another characterization of the class of cographs: they are the graphs entirely decomposable by modular decomposition, i.e. the graphs having no prime node in their modular decomposition tree.

Note that the adjacencies between vertices of a cograph can easily be read on its cotree, in the following way.

Remark 1. *Two vertices x and y of a cograph G having cotree T are adjacent iff the least common ancestor u of leaves x and y in T is a series node. Otherwise, if u is a parallel node, x and y are not adjacent.*

Modular decomposition theory is based on an operation called *substitution composition* (see [13, 14]). It turns out that contiguity and linearity have a nice behavior with regard to substitution composition, and our proof of the $O(\log n)$ upper bound for the contiguity and linearity of cographs is based on this notion, for which a formal definition is given below. Intuitively, the substitution composition consists in replacing a vertex x of a graph G by another graph H, keeping the adjacency relationships between vertices of H unchanged and making all the vertices of H adjacent to the neighbors of x in G.

Definition 1. *The result of the* substitution composition *of a vertex x of a graph G by a graph H is the graph denoted $G_{x\leftarrow H}$ and defined by $V(G_{x\leftarrow H}) = (V(G) \setminus \{x\}) \cup V(H)$ and $E(G_{x\leftarrow H}) = (E(G) \setminus \{xz \mid z \in N_G(x)\}) \cup E(H) \cup \{yz \mid y \in V(H)$ and $z \in N_G(x)\}$.*

The result of the substitution composition of a vertex of a cograph by another cograph is also a cograph and its cotree is obtained as explained below.

Remark 2. *For any cographs G and H, having cotrees T_G and T_H, and any vertex x of G, $G_{x\leftarrow H}$ is a cograph and its cotree T_{GxH} is obtained as follows:*

1. *Build T_{GxH} by replacing the leaf x of T_G by the root r_H of T_H.*
2. *If the parent u of leaf x in G has the same label as r_H, then remove r_H from T_{GxH} and assign u as parent to the children of r_H.*

For a rooted tree T and a node $u \in T$, the depth of u in T is the number of edges in the path from the root to u (the root has depth 0). The *height* of T, denoted by $height(T)$ or simply $h(T)$, is the greatest depth of its leaves. We denote by $\mathcal{C}(v)$ the children of a node v in T. For a rooted tree T, the subtree of T rooted at u, denoted by T_u, is the tree induced by node u and all its descendants in T.

Definition 2. *A* monotonic path *P of a rooted tree T is a path such that there exists some node $u \in T$ such that all nodes of P except u are ancestors of u. The unique node of P which has no ancestor in P is called the* root of the monotonic path *P.*

Definition 3. *A* rooted caterpillar *is a rooted tree whose internal nodes form a monotonic path.*

It is worth to note that in the rest of the article we consider only rooted trees and monotonic paths.

In the following, the notion of *minors* of rooted trees is central. This is a special case of minors of graphs (see e.g. [15]), for which we give a simplified definition in the context of rooted trees.

Definition 4. *The* contraction of edge uv *in a rooted tree T, where u is the parent of v consists in removing v from T and assigning its children (if any) to node u.*
A rooted tree T' is a minor *of a rooted tree T if it can be obtained from T by a sequence of edge contractions.*

Let us now formally define the contiguity and linearity of a graph.

Definition 5. *A* closed p-interval-model *(resp. open p-interval-model) of a graph $G = (V, E)$ is a linear order σ on V such that $\forall v \in V, \exists (I_1, \ldots, I_p) \in (2^V)^p$ such that $\forall i \in [\![1, p]\!]$, I_i is an interval of σ and $N[x] = \bigcup_{1 \leq i \leq p} I_i$ (resp. $N(x) = \bigcup_{1 \leq i \leq p} I_i$).*
The closed contiguity *(resp.* open contiguity*) of G, denoted by $cont(G)$ (resp. $cont^o(G)$), is the minimum integer p such that there exists a closed p-interval-model (resp. open p-interval-model) of G.*

Definition 6. *A* closed p-line-model *(resp. open p-line-model) of a graph $G = (V, E)$ is a tuple $(\sigma_1, \ldots, \sigma_p)$ of linear orders on V such that $\forall v \in V, \exists (I_1, \ldots, I_p) \in (2^V)^p$ such that $\forall i \in [\![1, p]\!]$, I_i is an interval of σ_i and $N[x] = \bigcup_{1 \leq i \leq p} I_i$ (resp. $N(x) = \bigcup_{1 \leq i \leq p} I_i$).*
The closed linearity *(resp.* open linearity*) of G, denoted by $lin(G)$ (resp. $lin^o(G)$), is the minimum integer p such that there exists a closed p-line-model (resp. open p-line-model) of G.*

Remark 3. *Note that in a p-interval-model, the intervals I_i of Definition 5 necessarily form a partition of $N[x]$ (or $N(x)$), while the intervals I_i assigned to a vertex x are not necessarily disjoint in the definition of a p-line-model (Definition 6).*

In all the paper, we abusively extend the notion of linearity and contiguity to cotrees referring to the linearity and contiguity of their associated cographs.

We have the following inequalities on open and closed linearity and contiguity.

Lemma 1. *For an arbitrary graph, we have the following inequalities:*

$$lin(G) \leq lin^o(G) + 1 \leq cont^o(G) + 1 \leq cont(G) + 2$$

Proof. The first inequality comes from the fact that if we have an open k-line-model for G, we can add an arbitrary order σ to it and take as the interval of each vertex x in σ a trivial interval reduced to x. In this way, we obtain a closed $(k+1)$-line-model, showing that $lin(G) \leq lin^o(G) + 1$.
For the second inequality, if we have an open k-interval-model for G based on an order σ, we can make an open k-line-model consisting of k copies of order σ: for each vertex x and for each i between 1 and k, the interval of x in copy number i is the interval number i of x in the k-interval model. This shows that $lin^o(G) \leq cont^o(G)$, and the second inequality follows from it.
Finally, for the third and last equality, notice that from a closed k-interval-model σ, we can easily obtain an open $(k+1)$-interval-model by simply removing each vertex x from $N[x]$: this may break at most one interval I_i of x in σ into two pieces, while leaving the other intervals unchanged. Thus, $cont^o(G) \leq cont(G) + 1$ and the last inequality of Lemma 1 follows. □

In the rest of the article, we consider only closed notions but, from the above inequalities, the bounds we obtain also hold for the open notions.

2. A min-max theorem on the rank of a tree

In this section we prove a min-max theorem (Theorem 1) linking the maximum height of a complete binary tree contained as a minor in a given tree T (see Definition 4 above) and the minimum height of partition of T into monotonic paths (see Definition 7 below). This theorem is the key of our approach, we use it in Section 3 to show that the contiguity of a cograph G is dominated by the maximum height of a complete binary tree T' contained in the cotree T of G.

We now give the definitions needed to state the min-max theorem we aim at, starting with the definition of a path partition and related notions.

Definition 7. *A* path partition $\mathcal{P}$ *of a tree T is a partition $\{P_1, \ldots, P_k\}$ of $V(T)$ such that for any $i \in [\![1, k]\!]$, the subgraph $T[P_i]$ of T induced by P_i is a monotonic path (see Definition 2).*
The partition tree *of a path partition $\mathcal{P}$, denoted by $T_p(\mathcal{P})$, is the tree whose nodes are P_i's and where the node of $T_p(\mathcal{P})$ corresponding to P_i is the parent of the node corresponding to P_j iff some node of P_i is the parent in T of the root of P_j.*
The height of a path partition $\mathcal{P}$ of a tree T, denoted by $h(\mathcal{P})$, is the height $h(T_p(\mathcal{P}))$ of its partition tree.

Note that in the preceding definition, it is anyway impossible that some node of P_i is the parent in T of some node u of another P_j that is not the root of P_j, since in this case u would have at least two parents in T.

Remark 4. *Note that the partition tree $T_p(\mathcal{P})$ is obtained from T by contracting all edges of the monotonic paths in $\mathcal{P}$.*

(a) (b)

Figure 1: A tree T and a path partition $\mathcal{P} = \{P_1, P_2, P_3, P_4, P_5, P_6\}$ of T (a), as well as the partition tree of $\mathcal{P}$ (b).

Figure 1 gives an example of a path partition of some tree and the corresponding partition tree. Let us now formally define the rank and the path-height of a tree, which are the two notions involved in our min-max theorem.

Definition 8. *As [10, 11], we define the* rank *of a tree T as the maximum height of a complete binary tree being a minor of T, that is:*

$$rank(T) = \max\{h(T') \mid T' \text{ is a complete binary tree and a minor of } T\}.$$

And we define the path-height of T as the minimum height of a path partition of T, that is:

$$ph(T) = \min\{h(\mathcal{P}) \mid \mathcal{P} \text{ is a path partition of } T\}.$$

For the rest of the paper, we extend the definition of the rank to any vertex v of a tree T by defining the rank of v as the rank of the subtree of T rooted at v.

The rest of the section is devoted to prove Theorem 1 which states that for any tree, its rank and its path-height are equal. Lemma 2 first proves it for the particular case of complete binary trees as this case will be a key step in the proof of the general result on arbitrary trees.

Lemma 2. *For a rooted complete binary tree T, $rank(T) = ph(T) = h(T)$.*

Proof. The fact that $h(T) = rank(T)$ is obvious from the definition of $rank(T)$. On the other hand, one can build a path partition $\mathcal{P}$ of height $h(T)$ by taking $\mathcal{P} = \{\{x\}\}_{x \in V(T)}$. In other words, each path in $\mathcal{P}$ is reduced to one single vertex of T. In this way, $T_p(\mathcal{P}) = T$ which gives $ph(T) \leq h(T)$.

Conversely, we prove that $ph(T) \geq h(T)$ by induction on $h(T)$. This property is true for the trivial tree T with only one vertex, for which we have $ph(T) = 0 \geq 0 = h(T)$. Now, assume that it is true for any rooted complete binary tree of height $k \geq 0$, and consider a rooted complete binary tree T of height $k+1$. Let $\mathcal{P}$ be a path partition of T such that $h(\mathcal{P}) = ph(T)$, and let P be the monotonic path of $\mathcal{P}$ containing the root r of T, as shown in Figure 2.

Since path P is monotonic, it does not contain at least one child u of r. Then, the subset $\mathcal{P}'$ of $\mathcal{P}$ made of the paths containing some node of

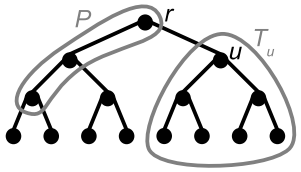

Figure 2: The induction step of the proof that the path-height of a complete binary tree is at least as big as its height, in the proof of Lemma 2.

T_u is a path partition of T_u. And from the induction hypothesis, we get $h(\mathcal{P}') \geq ph(T_u) \geq h(T_u) \geq k$. Since $h(\mathcal{P}) \geq h(\mathcal{P}') + 1$, we obtain $h(\mathcal{P}) \geq k + 1$, and so $ph(T) \geq k + 1$. This ends the induction and the proof of Lemma 2. □

Actually, in the proof of Theorem 1, we use only the part of Lemma 2 stating that $ph(T) \geq h(T)$ which is precisely the non-trivial part of the lemma. Let us now state Theorem 1, which is the min-max theorem we aim at.

Theorem 1. *For any rooted tree T, we have $rank(T) = ph(T)$.*

Proof. $\leq$: Let T' be a complete binary tree which is a minor of T of maximum height among such complete binary trees. By definition, we have $rank(T) = h(T')$ and from Lemma 2 we know that $h(T') = ph(T')$. It follows that $rank(T) = ph(T')$. Then, in order to prove $rank(T) \leq ph(T)$, we show that $ph(T') \leq ph(T)$.

Indeed, consider an arbitrary path partition $\mathcal{P}$ of T. We will build a path partition $\mathcal{P}'$ of T' whose height is at most the height of $\mathcal{P}$. By definition, tree T' can be obtained from T by a series of edge contractions. The partition $\mathcal{P}'$ we build is induced from $\mathcal{P}$ in the sense that we obtain it by updating partition $\mathcal{P}$ step by step along the series of edge contractions of T resulting in T'. Along this process, when an edge uv of the current tree T_{cur}, where u is the parent of v, is contracted in order to obtain the next tree T_{next}, we delete node v and assign its children to u, as in Definition 4. An example of edge contraction in T_{cur} and its effect on the current path partition is shown in Figure 3. Note that there are cases where the modification of the path partition is different from the case depicted on Figure 3, in particular when v is the only node in some path P of the partition.

This ensures that any set P of nodes inducing a monotonic path in T_{cur} also induces a monotonic path in T_{next} (except in the case where $P = \{v\}$: P becomes the empty set after the contraction and disappears from the path partition we maintain). Consequently, a path partition $\mathcal{P}_{next}$ of T_{next} can be obtained from the partition $\mathcal{P}_{cur}$ of T_{cur} in the following way: for all the nodes remaining in T_{next} (i.e. all the nodes of T_{cur} except node v), we keep them in the same path of the partition as they were previously belonging to in $\mathcal{P}_{cur}$.

It is not difficult to see that the partition tree T_p^{next} of $\mathcal{P}_{next}$ defined this way has height at most that of T_p^{cur}. Indeed, from the definition of the partition tree (see Definition 7), during the contraction of edge uv, the only paths of

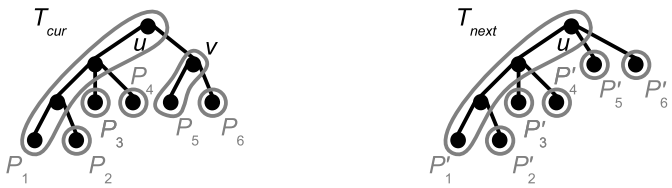

Figure 3: Left: a tree T_{cur} with its path partition $\mathcal{P}_{cur} = \{P_1, P_2, P_3, P_4, P_5, P_6\}$ whose partition tree T_p^{cur} has height 2. Right: the tree T_{next} resulting from the contraction of edge uv in T_{cur}, together with its path partition $\mathcal{P}_{next} = \{P'_1, P'_2, P'_3, P'_4, P'_5, P'_6\}$. All the paths in $\mathcal{P}_{next}$ are the same as those in $\mathcal{P}_{cur}$, except P'_5 which has lost node v. The height of T_p^{next} decreased, it is now 1.

$\mathcal{P}_{cur}$ that are likely to change their parent in T_p^{next} are those paths whose root is in $\mathcal{C}(v)$, as the other nodes of T_{cur} do not change their parent. For such a path $P_{v'}$ rooted at a child v' of v, the parent of $P_{v'}$ in T_p^{next} will be the path P_u containing u, which may be either its parent or its grandparent in T_p^{cur}, depending on whether nodes u and v are in the same path of $\mathcal{P}_{cur}$ or not. Since the only nodes of T_p^{cur} that change their parents in T_p^{next} change it for their grandparent in T_p^{cur}, it follows that $h(\mathcal{P}_{next}) \leq h(\mathcal{P}_{cur})$.

Consequently, at the end of the series of edge contractions that results in T', we obtain a path partition $\mathcal{P}'$ of T' such that $h(\mathcal{P}') \leq h(\mathcal{P})$. As this holds for any path partition $\mathcal{P}$ of T, we conclude that $ph(T') \leq ph(T)$.

$\geq$: We show that $rank(T) \geq ph(T)$ by induction on $rank(T)$. We use the following induction hypothesis $H(k)$: "for any tree T such that $rank(T) = k$, we have $ph(T) \leq rank(T)$".

Clearly, if $rank(T) = 0$, then T is a monotonic path. Thus, there is a trivial partition of T into a single path and the path-height of this partition is 0 : $ph(T) \leq rank(T)$.

Now, let $k \geq 0$ such that $\forall i \in [\![0, k]\!], H(k)$ holds. We show that $H(k+1)$ is satisfied. Let T be a tree such that $rank(T) = k+1$, and let S_{k+1} be the subset of nodes u of T such that $rank(T_u) = k+1$. We first show that S_{k+1} is a monotonic path of T containing the root.

Clearly, by definition of $rank(T)$, S_{k+1} contains the root of T, so let us show that it induces a monotonic path of T. First, note that if u is such that $rank(T_u) = k+1$ then for all ancestors v of u, $rank(T_v) = k+1$. Indeed, it is clear that $rank(T_v)$ is not less than $rank(T_u)$, and since $rank(T) = k+1$, we also have $rank(T_v) \leq k+1$. Now, assume for contradiction that there exist two nodes $u, v \in S_{k+1}$ such that none of them is the ancestor of the other. Let w be the least common ancestor of u and v in T. Let u' and v' be the two distinct children of w that are the ancestors of respectively u and v. From above, u' and v' are such that $rank(T_{u'}) = rank(T_{v'}) = k+1$. It follows that $rank(T_w) \geq k+2$: contradiction. Thus, all the nodes of S_{k+1} are comparable for the ancestor relationship, and since we already showed that all the ancestors of a node in S_{k+1} are in S_{k+1}, it follows that S_{k+1} induces a monotonic path of T containing the root.

Now, let $S' = \{u \in V(T) \setminus S_{k+1} \mid parent(u) \in S_{k+1}\}$, and let $u \in S'$. Clearly, since $u \notin S_{k+1}$, $rank(T_u) \leq k$ and the induction hypothesis implies that $ph(T_u) \leq k$. For any $u \in S'$, we denote by $\mathcal{P}_u$ a path partition of T_u of height at most k. Then, $\{S_{k+1}\} \cup \bigcup_{u \in S'} \mathcal{P}_u$ is a path partition of T and has height at most $k+1$. Thus, $ph(T) \leq k+1 = rank(T)$ and $H(k+1)$ holds, which ends the induction, showing that for any tree T, $rank(T) \geq ph(T)$. □

Note that Theorem 1 implies that the path-height of any tree T is at most logarithmic in its number of vertices. Indeed, $ph(T) = h(T')$ where T' is a complete binary tree being a minor of T. Then, $h(T') = \log_2(|V(T')|) \leq \log_2(|V(T)|)$, and thus $ph(T) \leq \log_2(|V(T)|)$. Combining this with Lemma 5 of Section 3 will provide the upper bound on the contiguity of cographs stated by Theorem 2, which is optimal from Theorem 3.

3. Upper bounds for contiguity and linearity of cographs

The aim of this section is to prove that the closed contiguity of any cograph is dominated by $\log n$ (Theorem 2). From Lemma 1, this implies that the same holds for open contiguity, open linearity and closed linearity. As we mentioned earlier, our proof relies on a decomposition approach, whose basic step is the root-path decomposition defined below. This basic step of decomposition decreases the contiguity of G by at most some constant. And we will show that using it only a logarithmic number of times (once per level of a minimum-height path partition of the cotree of G) is enough to entirely decompose the graph into single vertices. This will provide us with the logarithmic bound we aim at for the closed contiguity of cographs.

Definition 9. *A* root-path decomposition *(see Fig. 4) of a rooted tree T is a set $\{T_1, \ldots, T_p\}$ of disjoint subtrees of T, with $p \geq 2$, such that every leaf of T belongs to some T_i, with $i \in [\![1, p]\!]$, and the sets of parents in T of the roots of T_i's is a monotonic path containing the root of T.*

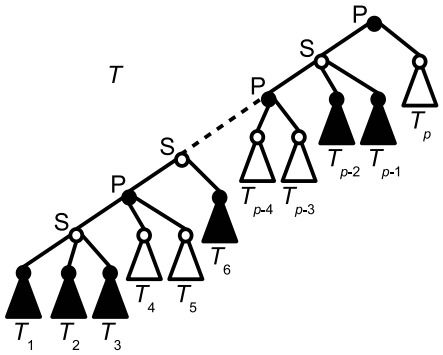

Figure 4: The root-path decomposition $\{T_1, \ldots, T_p\}$ of a rooted tree T.

In Definition 9 above, the roots of T_i's and their parents induce a caterpillar in T. It turns out that the contiguity of a caterpillar cotree is bounded by a constant. This is what is used in Lemma 3 below to state that the contiguity of a cotree admitting a root-path decomposition $\{T_1, \ldots, T_p\}$ is not more than the greatest contiguity of the T_i's plus some constant.

Lemma 3 (Caterpillar Composition Lemma). *Given a cograph $G = (V, E)$ and a root-path decomposition $\{T_i\}_{1 \leq i \leq p}$ of its cotree,*

$$cont(G) \leq 2 + \max_{i \in [\![1,p]\!]} cont(G[X_i]),$$

where X_i is the set of leaves of T_i.

Proof. For convenience of description of the closed contiguity model of G, we assume without loss of generality that the T_i's are numbered in such a way that for any $i, j \in [\![1, p]\!]$, if the parent of the root of T_i is an ancestor of the parent of the root of T_j then $i > j$. Moreover, we color the trees T_i of the root-path decomposition in the following way: if the root of T_i is a series node, T_i is colored white, otherwise T_i is colored black.

Let σ_i be a closed contiguity model realizing the closed contiguity of $G[X_i]$, where X_i is the set of leaves of T_i. We now build an order σ on the vertices of G such that for each $x \in V(G)$, $N[x]$ is the union of at most $l + 2$ intervals of σ, where $l = \max_{i \in [\![1,p]\!]} cont(G[X_i])$, which proves the lemma.

For clarity in the description of σ, we denote by $\sigma_i + \sigma_j$ the concatenation of the orders σ_i and σ_j, which allows us to use the sum notation $\sum_{i=1 \text{ to } p} \sigma_i = \sigma_1 + \sigma_2 + \ldots + \sigma_p$. Beware that in the case of the concatenation operation, the sum is not commutative: it must be done in the specified order, from 1 to p. Then the order σ we build is simply defined as (see Figure 5):

$$\sigma = \sum_{1 \leq i \leq p,\ T_i \text{ is black}} \sigma_i + \sum_{1 \leq j \leq p,\ T_j \text{ is white}} \sigma_j.$$

We now prove that, in σ, the closed neighborhood $N[x]$ of any vertex $x \in V(G)$ is split in at most $l + 2$ intervals. We separate the case where x is a leaf of some white T_i from the case where x is a leaf of some black T_i.

In the former case, the neighbors of x that are not in T_i are exactly those vertices belonging to some black X_j such that $j > i$ (see Remark 1). Then, we have $N[x] = (N[x] \cap X_i) \cup \bigcup_{j>i \text{ and } X_j \text{ is black}} X_j$. As σ_i realizes the closed contiguity of $G[X_i]$, which is by definition at most l, $N[x] \cap X_i$ is split in at most l intervals in σ_i. And since $\bigcup_{j>i \text{ and } X_j \text{ is black}} X_j$ is an interval of σ, it follows that $N[x]$ is split in at most $l + 1$ intervals in σ.

Now consider the case where x is a leaf of some black T_i. The neighbors of x that are not in T_i are exactly those vertices belonging to some black X_j, $j \neq i$, or belonging to some white $X_{j'}$ such that $j' < i$. Then, we have $N[x] = (N[x] \cap X_i) \cup (\bigcup_{j<i,\ X_j \text{ black}} X_j) \cup (\bigcup_{j>i,\ X_j \text{ black}} X_j \cup \bigcup_{j'<i,\ X_{j'} \text{ white}} X_{j'})$. Again, from the definition of σ_i, $N[x] \cap X_i$ is split in at most l intervals in σ_i. Moreover, $\bigcup_{j<i,\ X_j \text{ black}} X_j$ and $\bigcup_{j>i,\ X_j \text{ black}} X_j \cup \bigcup_{j'<i,\ X_{j'} \text{ white}} X_{j'}$ are two intervals of σ. Thus, $N[x]$ is split in at most $l + 2$ intervals in σ. As this holds

for any vertex x in G, it follows that $cont(G) \leq 2 + l$, which ends the proof of the lemma. □

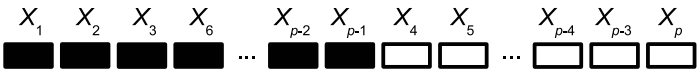

Figure 5: The general structure of the order σ used in Lemma 3 for the caterpillar partition of Fig 4.

Lemma 4 below is at the core of the recursive decomposition scheme we use to exhibit an encoding of an arbitrary cograph G realizing an $O(\log n)$ closed contiguity. It states that it is always possible to find a root-path decomposition of a cotree T such that the ranks of the T_i's are strictly less than the rank of T. Its proof relies on the min-max theorem of Section 2 (Theorem 1) which links the rank and the path-height of a tree.

Lemma 4. *Given a rooted tree T such that $rank(T) = k \geq 1$, there exists a root-path decomposition $\{T_1, \ldots, T_p\}$ of T such that for each $i \in [\![1, p]\!]$, $rank(T_i) \leq k - 1$.*

Proof. Since T has rank k, from Theorem 1, T has path-height k. Consider a path partition $\mathcal{P}$ of T of height k where the monotonic path containing the root r of T is denoted by P_r. If the lowest node of P_r is a leaf, removing it from P_r and letting it form its own path still gives a path partition of height k, since $k \geq 1$. We thereby assume that P_r contains only internal nodes of T. We denote by $u_1, \ldots, u_p$ the nodes of T whose parent is in P_r. Clearly, for each $i \in [\![1, p]\!]$, the paths of $\mathcal{P}$ that contain some node of T_{u_i} form a path partition $\mathcal{P}_i$ of T_{u_i}. And since $\mathcal{P}$ has height k, then for all $i \in [\![1, p]\!]$, $\mathcal{P}_i$ has height at most $k - 1$. It follows from Theorem 1 that $rank(T_{u_i}) \leq k - 1$. Thus, $\{T_{u_1}, \ldots, T_{u_p}\}$ is a suitable root-path decomposition of T, which achieves the proof of the lemma. □

Now, putting everything together, we are ready to state Lemma 5, showing that the contiguity of a cograph is bounded by a constant times the rank of its cotree.

Lemma 5. *Let G be a cograph and T its cotree. We have $cont(G) \leq 2\, rank(T)+ 1$.*

Proof. We prove this property by induction on the rank k of T.

For $k = 1$, we have to show that $cont(G) \leq 3$. Since T is a cotree, all its internal nodes have at least two children. It follows, since T has rank 1, that all its internal nodes are ancestors of each other, otherwise T would have rank at least 2. Thus, T is a rooted caterpillar (see Definition 3). Let us denote $V(G) = \{x_i\}_{1 \leq i \leq n}$. Since T is a caterpillar, the family $\{T_i\}_{1 \leq i \leq n}$, where T_i is the trivial tree formed with the single leaf x_i, is a root-path decomposition of T. With the notations of Lemma 3, the set of leaves of T_i is $X_i = \{x_i\}$. And since for any $i \in [\![1, n]\!]$, $cont(G[X_i]) \leq 1$, Lemma 3 concludes that $cont(G) \leq 2+1 = 3$.

Now, let $k \geq 1$ such that the property holds for all $k' \in [\![1, k]\!]$, that is any cograph G whose cotree T has rank k' satisfies $cont(G) \leq 2\,rank(T) + 1$. We prove this also holds for $k+1$. Let T be the cotree of a cograph G such that $rank(T) = k+1$. From Lemma 4, there exists a root-path decomposition $\{T_1, \ldots, T_p\}$ of T such that for all $i \in [\![1, p]\!]$, $rank(T_i) \leq k$. Thus, by applying the induction hypothesis on each T_i, we know that $\forall i \in [\![1, p]\!], cont(G[X_i]) \leq 2\,rank(T_i)+1 \leq 2k+1$, where X_i is the set of leaves of T_i. By applying Lemma 3, we deduce that $cont(G) \leq 2 + \max_{i \in [\![1,p]\!]} cont(G[X_i]) \leq 2k + 3 = 2(k+1) + 1$. This ends the induction and the proof of the Lemma. □

As noted previously, the rank of a cotree T is at most logarithmic in the number of vertices of G, which gives the upper bound we aim at.

Theorem 2. *The closed contiguity of a cograph is at most logarithmic in its number of vertices, or more formally, if $G = (V, E)$ is a cograph, then $cont(G) \leq 2\log_2 |V| + 1$.*

Proof. Let us denote by k the rank of the cotree T of G. From the definition of the rank, we deduce that T has at least 2^k leaves, which gives $|V| \geq 2^k$. It follows that $k \leq \log_2 |V|$. And since from Lemma 5, $cont(G) \leq 2k + 1$, we conclude that $cont(G) \leq 2\log_2 |V| + 1$. □

From the inequalities of Lemma 1, we immediately get the following corollary.

Corollary 1. *For any cograph G on n vertices, the open contiguity, the open linearity and the closed linearity of G are all $O(\log n)$.*

4. Lower bounds for contiguity and linearity of cographs

In this section, we prove that the $O(\log n)$ bound we obtained for the contiguity of cographs is tight. In other words, there exist families of cographs such that the contiguity of a cograph of the family on n vertices is $\Omega(\log n)$. Here we show that the cographs whose cotree is a complete binary tree, which already played a key role in the proof of the upper bound, are such a family (Theorem 3).

For linearity, we could not find a family of cographs for which we can prove that the linearity is $\Omega(\log n)$. But again, we prove that the cographs whose cotree is a complete binary tree are close to the upper bound. More precisely, their linearity is $\Omega(\log n / \log\log n)$ (Theorem 4).

Finally, note that even though our theorems are stated for closed contiguity and closed linearity, we derive the same bounds for the open notions, from the inequalities linking them to the closed notions.

Theorem 3. *Let G be a cograph whose cotree is a complete binary tree. Then, $cont(G) = \Omega(\log n)$.*

Proof. In order to prove this theorem, we first show that a cograph whose cotree T is a complete binary tree of height $4k$ with a series root has closed contiguity at least k. The proof is by induction on k.

For $k = 1$, the statement is self-evident as any graph has closed contiguity at least 1. Consider a cotree T which is a complete binary tree of height $4(k+1)$ with a series root denoted by r, as shown in Figure 6. Root r has two parallel children denoted by u and v. Consider a grandchild u_1 of u and its two series children v_1 and v_2. Consider now a great-grandchild v_3 of u such that the least common ancestor of nodes v_3 and v_1 in T is node u. Let us denote by X_1, X_2, X_3 the set of leaves of respectively $T_{v_1}, T_{v_2}, T_{v_3}$. Let x be a leaf of T_v. Vertex x is adjacent in G to all the vertices of $X_1 \cup X_2 \cup X_3$, and for any distinct i, j in $[\![1, 3]\!]$, all the vertices of X_i are non-adjacent to all the vertices of X_j. Moreover, $T_{v_1}, T_{v_2}, T_{v_3}$ are complete binary cotrees of height $4k$ whose root is a series node. Then, by the induction hypothesis, $cont(G[X_i]) \geq k$ for any $i \in [\![1, 3]\!]$.

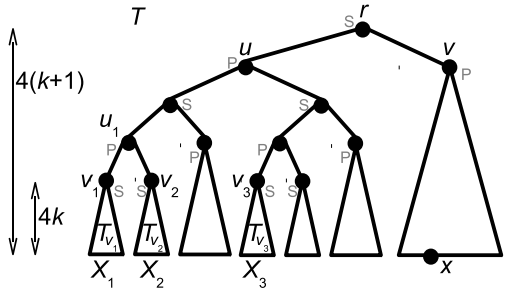

Figure 6: The complete binary cotree T of height $4(k+1)$ used in the proof of Theorem 3.

Consider a linear order σ on the vertices of G. We denote by a (resp. b) the vertex of $X_1 \cup X_2 \cup X_3$ which is closer to x on the left (resp. on the right) in σ, if it exists (i.e. if there is some node of $X_1 \cup X_2 \cup X_3$ on the left (resp. right) of x in σ). Clearly, there exists $i \in [\![1, 3]\!]$ such that none of a, b is in X_i. Then, for all the vertices $y \in X_i$, there exists some vertex $z \in \{a, b\}$ such that z is between x and y in σ, and z is not adjacent to y. Since X_i has contiguity at least k, there exists some vertex $y_1 \in X_i$ such that its neighborhood in $G[X_i]$ is split in at least k intervals in σ. And it turns out that x, which is adjacent to y_1, necessarily belongs to a new interval of neighbors of y_1 in σ, which is different from these k intervals, since all the vertices of X_i are separated from x in σ by some $z \in \{a, b\}$, which is not adjacent to y_1. It follows that the neighborhood of y_1 is split in at least $k+1$ intervals in σ. Thus, $cont(G) \geq k+1$, which ends the induction showing that a cograph whose cotree is a complete binary tree of height $4k$ with a series root has contiguity at least k.

And consequently, for any cograph G on n vertices whose cotree T is a complete binary tree, we have $cont(G) = \Omega(\log n)$.

□

As mentioned in the introduction, the open and closed contiguity differ by

at most one. Then from the lower bound on the closed contiguity, we deduce the same for open contiguity.

Corollary 2. *The open contiguity of cographs whose cotree is a complete binary tree is* $\Omega(\log n)$.

Proof. To prove the corollary we need to state the inequality between open and closed contiguity in the opposite direction than the one stated in Lemma 1. Namely, we show that for any graph G, we have $cont(G) \leq cont^o(G) + 1$. This is straightforward, as in an open model, one can always add one interval for each vertex x made of the vertex x itself and no other vertices. In this way, one obtains a closed contiguity model with at most one additional interval per vertex, which shows that $cont(G) \leq cont^o(G)+1$. Corollary 2 directly follows. □

For linearity, we could not prove an $\Omega(\log n)$ bound by the same arguments as for contiguity. Roughly speaking, the reason why the proof for contiguity works is that in order σ not all the three sets X_1, X_2 and X_3 can be next to x. But in the case of linearity there is not only one order σ but k orders $\sigma_1, \ldots, \sigma_k$. Then, up to $2k$ sets of vertices X_i can be next to x in some order. This is the reason why, in [5], a proof is made by using $2k + 1$ sets instead of 3 like in the proof above. But as this number $2k + 1$ is not constant, it does not give an $\Omega(\log n)$ bound but only an $\Omega(\log n/ \log\log n)$ bound instead. Actually, the proof of [5] is written for permutation graphs and interval graphs, but it turns out that the graphs used to do so are also cographs.

Consequently, we do not need to prove that the $\Omega(\log n/ \log\log n)$ bound holds for cographs. What we show here, using the result of [5], is that this bound is also reached on the cographs whose cotree is a complete binary tree. As these particular cographs play a key role in the upper bound proof, this emphasizes on the question whether the linearity of a cograph G is equal, in order of magnitude, to some function of the rank of its cotree, i.e. the maximum height of a complete binary tree included in it as a minor. This is one of the main questions raised by our work.

Theorem 4. *Let G be a cograph whose cotree is a complete binary tree. Then,* $lin(G) = \Omega(\log n/ \log\log n)$.

Proof. The result of [5] we use to prove Theorem 4 is as follows: consider the transitive closure[1] of the rooted and directed $(2k + 1)$-ary tree[2] T_k of height k, for $k \geq 1$, and let G_k be its underlying undirected graph. It is shown in [5] that $lin(G_k) \geq k$. Here, we show that a cograph whose cotree is a complete binary tree of height $2k\lceil\log_2(2k + 1)\rceil + 1$ contains G_k as an induced

[1]The transitive closure of a directed graph G is the graph G' formed by putting an arc from x to y in G' iff there is a directed path from x to y in G (see [16] for a more formal definition).

[2]That is the tree T_k of height k where every internal node of T_k has exactly $2k+1$ children and the arcs of the tree are directed from children to their parent node.

subgraph, and therefore has linearity at least k. From this, we obtain the property stated by the theorem: for G a cograph whose cotree is a complete binary tree, $lin(G) = \Omega(\log n / \log\log n)$.

In the first part of the proof, we show that a cograph G whose cotree T is a complete binary tree of height $2k\lceil\log_2(2k+1)\rceil + 1$ contains G_k as an induced subgraph. First, note that G contains as an induced subgraph the cograph whose cotree T' is a complete binary tree of height $2k\lceil\log_2(2k+1)\rceil$ having a series node as the root. Indeed, if the root r of T is a parallel node, just take the tree T_u rooted at an arbitrary child of r. Otherwise, if r is already a series node, simply replace the lower level of internal nodes of T (which are series nodes) by leaves: the cotree T' obtained satisfies the required property.

Now, we prove that the cograph whose cotree is T' contains G_k as an induced subgraph. We make it by induction on d for a complete binary cotree of height $2d\lceil\log_2(2k+1)\rceil$. Formally, we denote by G_k^d the underlying undirected graph of the transitive closure of the rooted directed $(2k+1)$-ary tree of height d, as shown in Figure 7(a) for $k = 1$ and $d = 2$. We also denote by T_k^d the complete binary cotree of height $2d\lceil\log_2(2k+1)\rceil$ whose root is a series node, and we denote by F_k^d its corresponding cograph.

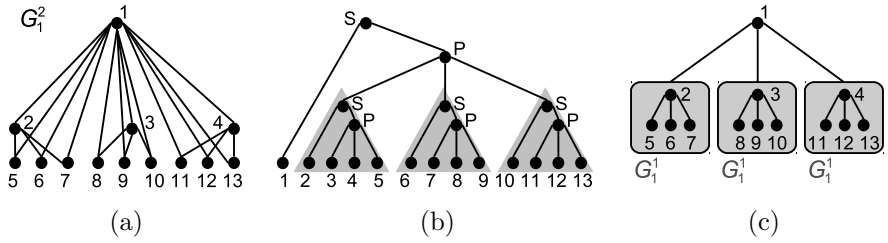

Figure 7: The graph G_1^2 (a), its cotree (b) and its representation as the result of a substitution composition of 3 copies of G_1^1 in the leaves of the star $K_{1,3}$ (c). Be aware that, for sake of clarity, we give the example of the construction of graph G_1^2 though in the proof of Theorem 4 we need only to consider graphs G_k^d such that $d \leq k$.

We use the following induction hypothesis $H(d)$: F_k^d contains G_k^d as an induced subgraph. Assume that $H(d)$ holds for some $d \geq 1$, we show that $H(d+1)$ is true. To that purpose, we first note that (*) G_k^{d+1} is obtained as the substitution composition (see Definition 1) of $2k+1$ copies of G_k^d into the leaves of $K_{1,2k+1}$ (the star with $2k+1$ leaves), as shown in Figure 7(c). In other words, take the star $K_{1,2k+1}$, replace each of the $2k+1$ leaves by a copy of G_k^d and link all the vertices of each copy to the center of the star $K_{1,2k+1}$: you obtain G_k^{d+1}. Now consider any node u at depth $2\lceil\log_2(2k+1)\rceil$ in T_k^{d+1}, the tree T_u is exactly T_k^d. Now replace in T_k^{d+1} each node u at depth $2\lceil\log_2(2k+1)\rceil$ by a leaf: you obtain T_k^1. This shows, from Remark 2, that F_k^{d+1} is obtained by substitution composition of one copy of F_k^d into each vertex of F_k^1. Moreover, by the induction hypothesis, F_k^d contains G_k^d as an induced subgraph. Then, from property (*) above, in order to show that F_k^{d+1} contains G_k^{d+1} as an induced subgraph, it suffices to show that F_k^1 contains $K_{1,2k+1}$ as an induced subgraph

(which also stands for the initialization step of our induction, with $d = 1$).

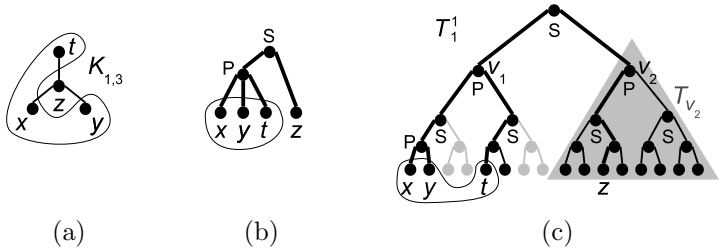

Figure 8: The star $K_{1,3}$ (a), its cotree (b) and the complete binary cotree T_1^1 of height $2\lceil \log_2(3) \rceil$ (c), whose corresponding cograph contains the star $K_{1,3}$ as an induced subgraph, as highlighted by the bold edges.

The root of T_k^1 is a series node with two children v_1 and v_2, as illustrated in Figure 8(c) for the case $k = 1$. Choose an arbitrary leaf of T_{v_2} as being the center z of $K_{1,2k+1}$. We show that the cograph whose cotree is T_{v_1} contains a stable (a subset of vertices having no edges between them) of size $2k + 1$. T_{v_1} is of height $2\lceil \log_2(2k + 1) \rceil - 1$ and its root is a parallel node. Then, T_{v_1} contains $\lceil \log_2(2k + 1) \rceil$ levels of parallel nodes. For each of the series nodes in T_{v_1} arbitrarily choose one of its two children and remove the subtree below (the removed subtrees appear in light gray in T_1^1 in the example in Figure 8): one obtains a tree having $2^{\lceil \log_2(2k+1) \rceil} \geq 2k + 1$ leaves, and for any two of these leaves x, y, their least common ancestor in T_{v_1} is a parallel node. This implies, from Remark 1, that x and y are not adjacent. And it follows that T_{v_1} contains a stable of size $2k + 1$. Thus, F_k^1 contains $K_{1,2k+1}$ as an induced subgraph. Moreover, by induction hypothesis, F_k^d contains G_k^d as an induced subgraph. It follows that the graph resulting from the substitution composition of vertices of F_k^1 by copies of F_k^d, which is precisely F_k^{d+1}, contains as an induced subgraph the graph resulting from the substitution composition of the leaves of $K_{1,2k+1}$ by copies of G_k^d, which is precisely G_k^{d+1}. As a conclusion, F_k^{d+1} contains G_k^{d+1} as an induced subgraph, which ends the induction and the proof of the fact that G contains G_k as an induced subgraph. It follows, as $lin(G_k) \geq k$ from the result of [5], that G has closed linearity at least k.

The rest of the proof of Theorem 4 is simply analytic considerations. Let now G be a cograph whose cotree is a complete binary tree of arbitrary height and denote by n the number of vertices of G. Consider the greatest integer k such that $2k\lceil \log_2(2k + 1) \rceil + 1 \leq height(G)$. Then we have $2k\lceil \log_2(2k + 1) \rceil + 1 \leq \log_2 n \leq 2(k + 1)\lceil \log_2(2k + 3) \rceil + 1$. It follows that $\log(n) = \Theta(k \log k)$. Applying the log to this relation we obtain $\log \log n = \log k + o(\log k)$. This immediately gives $\log \log n = \Theta(\log k)$ and so $\log k = \Theta(\log \log n)$. Since $\log(n) = \Theta(k \log k)$, we also have $k \log k = \Theta(\log(n))$. Then, $k = \Theta(\log(n))/\log k = \Theta(\log(n))/\Theta(\log \log n) = \Theta(\log(n)/\log \log n)$. Finally, since from the definition of k we have $2k\lceil \log_2(2k + 1) \rceil + 1 \leq height(G)$, then G contains as induced subgraph a cograph whose cotree is a complete binary tree of height $2k\lceil \log_2(2k + 1) \rceil + 1$, for which we proved above that the linearity is at least k.

Consequently, $lin(G) \geq k = \Theta(\log(n)/\log\log n)$, which completes the proof of Theorem 4.

□

From Theorem 4 and from the first inequality of Lemma 1, we directly deduce the following corollary.

Corollary 3. *For any cograph G whose cotree is a complete binary tree, we have $lin^o(G) = \Omega(\log n/\log\log n)$.*

5. Conclusion

We showed that the contiguity and linearity of cographs are both bounded by $O(\log n)$. Moreover, we showed that, in the worst case, the contiguity of cographs on n vertices may be up to $\Omega(\log n)$ and that their linearity may be up to $\Omega(\log n/\log\log n)$. This means that our bounds for contiguity are tight, while the asymptotic behavior of the maximum linearity of cographs on n vertices is still to be determined. Then, the first open problem suggested by our work is to fill the gap between $O(\log n)$ and $\Omega(\log n/\log\log n)$ for the linearity of cographs.

In this perspective, we note that our proofs for both the upper and lower bound rely on the cographs whose cotree is a complete binary tree. Therefore, it brings the question to know whether the linearity of a cograph is equal, in order of magnitude, to some function $f(h)$ of the height h of the maximum complete binary tree being a minor of its cotree, the first two possibilities being $f(h) = h$ and $f(h) = h/\log h$. Finding such an f would give the asymptotic behavior of the linearity of cographs.

The same question holds for the contiguity, the only possible function in this case being $f(h) = h$. If the answer to this question turned out to be positive, then it would guarantee that the contiguity model of graph G proposed by our constructive proof of Lemma 3 realizes a value of the contiguity that is in a constant approximation ratio of the actual contiguity of G. This would thereby provide an approximation algorithm for the contiguity of cographs.

Another key perspective of our work is to extend the $O(\log n)$ bound on these two parameters to larger classes of graphs. For example, do these results hold for permutation graphs (which are a proper generalization of cographs) and for interval graphs?

Acknowledgments. The authors thank Pierre Charbit and Stéphan Thomassé for useful discussions on the subject, as well as George Oreste Manoussakis for proofreading a draft of this article.

[1] G. Turan, On the succinct representation of graphs, Discr. Appl. Math. 8 (1984) 289–294.

[2] P. Boldi, S. Vigna, The webgraph framework I: compression techniques, in: WWW'04, ACM, 2004, pp. 595–602.

[3] P. Boldi, S. Vigna, Codes for the world wide web, Internet Mathematics 2 (4) (2005) 407–429.

[4] P. Goldberg, M. Golumbic, H. Kaplan, R. Shamir, Four strikes against physical mapping of DNA, Journal of Computational Biology 2 (1) (1995) 139–152.

[5] C. Crespelle, P. Gambette, Efficient neighbourhood encoding for interval graphs and permutation graphs and $O(n)$ breadth-first search, in: 20^{th} International Workshop on Combinatorial Algorithms (IWOCA'09), no. 5874 in LNCS, 2009, pp. 146–157.

[6] A. Brandstädt, V. Le, J. Spinrad, Graph Classes: a Survey, SIAM Monographs on Discrete Mathematics and Applications, 1999.

[7] R. Wang, F. Lau, Y. Zhao, Hamiltonicity of regular graphs and blocks of consecutive ones in symmetric matrices, Discr. Appl. Math. 155 (17) (2007) 2312–2320.

[8] C. Gavoille, D. Peleg, The compactness of interval routing, SIAM Journal on Discrete Mathematics 12 (4) (1999) 459–473.

[9] F. Roberts, Representations of indifference relations, Ph.D. thesis, Stanford University (1968).

[10] A. Ehrenfeucht, D. Haussler, Learning decision trees from random examples, Information and Computation 82 (3) (1989) 231–246.

[11] R. Gavaldà, D. Thérien, Algebraic characterizations of small classes of boolean functions, in: Proceedings of the 20th Annual Symposium on Theoretical Aspects of Computer Science (STACS'03), Vol. 2607 of LNCS, 2003, pp. 331–342.

[12] M. Habib, C. Paul, A survey of the algorithmic aspects of modular decomposition, Computer Science Review 4 (1) (2010) 41–59.

[13] R. Möhring, Algorithmic aspect of the substitution decomposition in optimization over relations, set systems and boolean functions, Annals of Operations Research 4 (1985) 195–225.

[14] R. Möhring, F. Radermacher, Substitution decomposition for discrete structures and connections with combinatorial optimization, Annals of Discrete Mathematics 19 (1984) 257–356.

[15] L. Lovász, Graph minor theory, Bulletin of the American Mathematical Society 43 (1) (2006) 75–86.

[16] D. B. West, Introduction to Graph Theory (2nd edition), Prentice Hall, 2000.