Linear recognition of almost interval graphs

Vincenzo Politelli
Supervisor: Prof. Chien-Chung Huang

1 Introduction and first definitions

In this report, we are going to provide a pedagogical presentation of the vertex deletion algorithm for interval graphs presented in [Cao14].

First of all, we introduce the notation we will adopt in this text. Given a graph G we are going to denote its set of vertices (resp. edges) by $V(G)$ (resp. $E(G)$). We also denote $|V(G)|$ by $|G|$ and $|E(G)|$ by $\|G\|$. Given a vertex v of G we will denote its closed (resp. open) neighborhood in G by $N[v]$ (resp. $N(v)$). We say that a vertex v of G is simplicial when $N[v]$ induces a clique in G. We will denote the set of simplicial vertices of G by $S(G)$.

We now introduce the notion of interval graph.

Definition 1.1. We say that a graph G is interval if there exists a family of intervals $\mathcal{I} = (I_v)_{v \in V(G)}$ such that for all $u, v \in I_v$, $u \sim v$ if and only if $I_u \cap I_v$. The family I is called an *interval model* of G.

Throughout this text, we will mostly assume without loss of generality that, when providing an interval model of a graph, all intervals are closed and bounded and all their non-empty intersections are not singletons.

As proved in [LB62], interval graphs are exactly the hole-free graphs (also known as chordal graphs) which do not have asteroidal triples. We remind that a hole is a chordless cycle and an asteroidal triple (from now on "at") is a triple of vertices $\{x, y, z\}$ such that for all couple of vertices in the said triple, there exists a path connecting them that avoids the neighbours of the third one. The authors of [LB62], went further in identifying the minimal chordal graphs (under the operation of taking iduced subgraphs) containing an at. Such graphs are shown in Figure 1 and are called minimal chordal asteroidal witnesses, caws for short. All in all, holes and caws are the minimal forbidden induced subgraphs for interval graphs. We will denote the set of holes and caws by $\mathcal{F}_I$.

We now provide some terminology for caws. The vertices of caws which are colored in Figure 1 are called *terminals*, those which are colored in green are called *shallow terminals*. When dealing with $\dagger_d$ or $\ddagger_d$, the set of vertices $\{l, b_1, c_1, c_2, s, b_d, r\}$ (where c_1 and c_2 are identified in the case of $\dagger_d$) is called *frame* of the caw, whereas the $b_1 - b_d$ path is called the *base* of the caw.

Associated to the family of interval graphs is the family of *locally* interval graphs. That is, the family characterized by the set of forbidden induces subgraphs composed of holes of six or more vertices and caws which are not $\dagger_d$ or $\ddagger_d$ with $d \geq 3$. We will denote this new family of forbidden induced subgraphs by $\mathcal{F}_{LI}$. The holes which are contained in $\mathcal{F}_{LI}$ are called *short* holes, those which are not are called *long* holes; the caws which are contained in $\mathcal{F}_{LI}$ are called *small* caws and those which are not are called *large* caws.

One last family of graphs that we will need is that of normal and Helly circular-arc graphs. Similarly to interval graphs, circular-arcs graphs are those graphs which are represented by a *circular arc model* $\mathcal{A} = (A_v)_{v \in V(G)}$ where $A_v = [l_v, r_v]$ represents a closed arc with l_v and r_v respectively the counterclockwise and clockwise ends of the arc. We also assume $l_v, r_v > 0$ and so, $l_v > r_v$ in case the arc passes through the point 0. Circular-arc graphs, however, are far less understood than interval graphs because of some pathological cases. This motivates the definition of normal and Helly circular-arc graphs.

Definition 1.2. A graph G is a normal and Helly circular-arc graph if for all circular-arc models of G, no two arcs intersect at both ends and no three arcs pairwise intersect without sharing a common point.

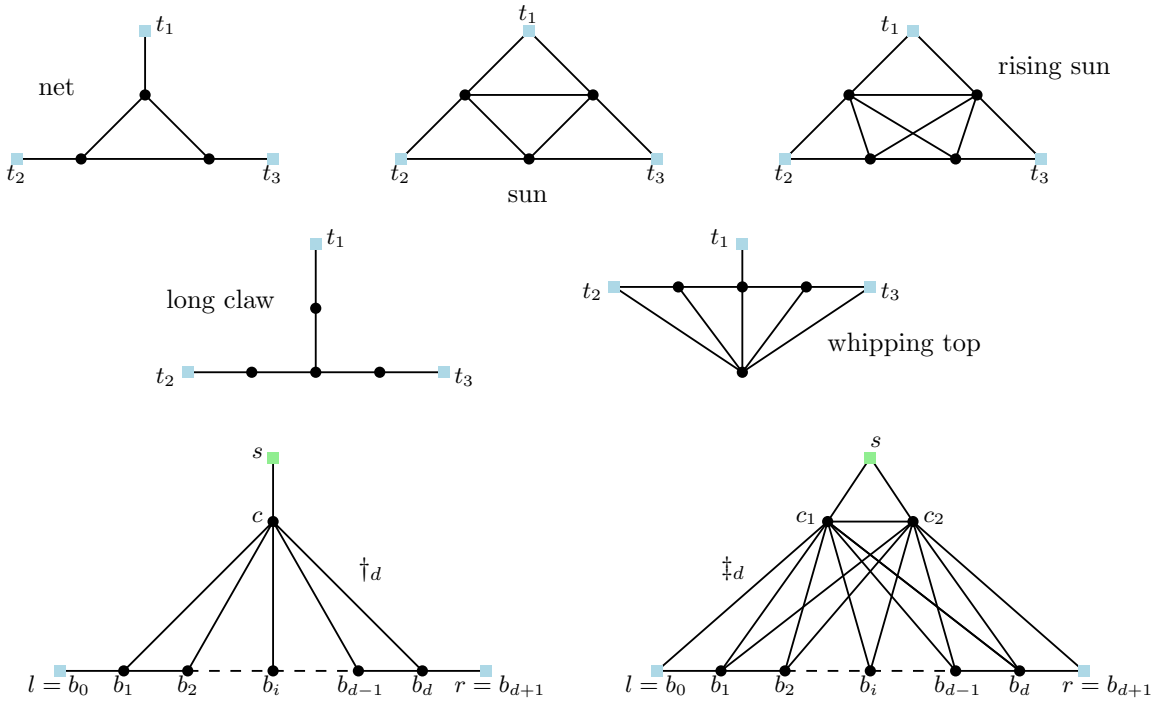

Figure 1: Minmimal chordal asteroidal witnesses (caws)

An important characterisation of normal and Helly circular-arc graphs is provided by [LSS13]. We have that a graph G is a normal and Helly circular-arc graph if and only if for all circular-arc models of G no three or less arcs cover the whole circle.

2 Vertex deletion problems that are FPT

The question we would like to answer in this text is that of finding an optimal way of modifying a given graph into an interval graph. This problem is a special case of a large and important family of problems that we are going to describe.

Definition 2.1. Let $\mathcal{C}$ be a class of graphs.

- If $\mathcal{C}$ has the property that, for any $G \in \mathcal{C}$, any induced subgraph of G is also in $\mathcal{C}$, then we say that $\mathcal{C}$ is a hereditary property.
- If there exists a family $\mathcal{F}$ such that a graph $G \in \mathcal{C}$ if and only if $G' \notin \mathcal{F}$ for any induced subgraph G' of G, then $\mathcal{C}$ is said to have a forbidden set characterization. If, moreover, $\mathcal{F}$ is finite, $\mathcal{C}$ is said to have a finite forbidden set characterization.

Notice that that of being an interval graph is a hereditary property and the class of interval graphs has a (non-finite) forbidden set characterization.

Given a hereditary property $\mathcal{C}$, a natural problem that arises is the *$\mathcal{C}$-graph modification problem*, that is, the problem of computing the minimum number of vertices additions, edge additions or edge deletions needed to transform a given graph G into a graph in $\mathcal{C}$. Notice that the fact that $\mathcal{C}$ is a hereditary property makes the edge addition problem trivial.

From now on, we are only going to consider the vertex deletion problem. This is because, if a graph G can be made interval with k_+ edge additions and k_- edge deletions, then it can be made so with $k_+ + k_-$ vertex deletions. Thus, knowing how to solve the vertex deletion problem with k deletions, we know how to solve the edge addition (resp. deletion) with k additions (resp. deletions).

This kind of problems are hard in general, indeed, as proved in [KD79], the vertex deletion problem is NP-hard for any non-trivial hereditary property. This motivates the research for FPT or approximation schemes for the vertex-deletion problem. As proved in [Cai96], the $\mathcal{C}$-vertex deletion problem for a class $\mathcal{C}$ which has a finite forbidden set characterization is indeed FPT in the maximum number of vertex deletions. In what follows, we are presenting a proof of this fact.

If $\mathcal{C}$ has a finite forbidden set characterization, let $\mathcal{F}$ be the set of forbidden induced subgraphs and let ν the maximum number of vertices of a graph in $\mathcal{F}$. Given a graph G on n vertices and m edges, we can check in $\mathcal{O}(n^\nu)$ whether G belongs to the class $\mathcal{C}$. Indeed, for each graph F in $\mathcal{F}$ we can enumerate the functions from $V(F)$ to $V(G)$ which are at most n^ν and check in time $\mathcal{O}(\nu^2) = \mathcal{O}(1)$ whether any of them is an isomorphism.

We get the following proposition.

Proposition 2.2. ([Cai96]) For any hereditary property $\mathcal{C}$, if $\mathcal{C}$ is recognizable in time $T(n, m)$, with n, m respectively the number of nodes and edges of the graph; then, for any $G \notin \mathcal{C}$, we can find a minimal forbidded induced subgraph of $\mathcal{C}$ in G in time $\mathcal{O}(n \cdot T(n, m))$.

Proof. Suppose that there exists an algorithm M that tests the membership of its argument G in $\mathcal{C}$ in time $T(n, m)$. We first provide an algorithm and then prove its correcteness and time complexity.

Algorithm 1: Minimal forbidden induced subgraph

Input: An graph G and a hereditary property $\mathcal{C}$.
Output: An (inclusion-wise) minimal forbidden induced subgraph of G, otherwise the empty graph.

```
1 F ← ∅;
2 V ← V(G);
3 while M(G) = "False" do
4 |   choose any vertex v ∈ V;
5 |   if M(G − v) then
6 |   |   F ← F ∪ {v};
7 |   else
8 |   |   G ← G − v;
9 return G[F].
```

The proof of the time complexity is clear. The correctness follows from the fact that if $M(G) =$ "False" and $M(G - v) =$ "True", then v is a node of an inclusion-wise minimal forbidden induced subgraph. □

From this we get that Algorithm 2 is FPT and solves the vertex deletion problem.

We analise the algorithm. Given a graph G and a hereditary property $\mathcal{C}$ with a finite forbidden set characterization, Proposition 2.2 and the previous observations tell us that the total complexity of the algorithm, whose correctness is clear, is $\mathcal{O}(\nu^k \cdot n \cdot n^\nu) = \mathcal{O}(\nu^k \cdot n^{\nu+1})$. Which gives us the desired FPT algorithm.

The class of interval graphs and many other important graph classes such as trees and chordal graphs have the hereditary property but do not have a finite forbidden set characterization since they have arbitrarily big minimal obstructions such as cycles. Interval graphs also have very far less understood arbitrarily big minimal obstructions such as caws. Therefore the approach of Algorithm 2 does not serve our purpose. Moreover, such algorithm is also very inefficient on special cases such as interval graph.

One possible stategy to tackle this problem is the so-called *two-phase approach*. That is, one first divides the class of forbidden induced subgraphs into two disjoint classes $\mathcal{F} = \mathcal{F}_B \cup \mathcal{F}_U$ such that $\mathcal{F}_B$ contains some selected minimal obstructions and $\mathcal{F}_U$ contains the rest of the graphs in $\mathcal{F}$. We are going to call the graphs in $\mathcal{F}_B$ the *small* obstructions and the graphs in $\mathcal{F}_U$ the *large* obstructions. Phase 1 consists of disposing of small obstructions while, in phase 2, we get rid of

Algorithm 2: Vertex deletion problem

Input: A graph G and a hereditary property $\mathcal{C}$ with a finite forbidden set characterization; an integer k representing the maximum number of vertex deletions.

Output: "True" if G can be made into a graph in $\mathcal{C}$ in at most k vertex deletions, "False" otherwise.

```
count ← 0;
while count < k do
    find a minimal induced forbidden subgraph H in G with Algorithm 1;
    if H is found then
        branch on deleting a vertex from H;
    else
        return "True";
    count++;
return "False".
```

large ones, exploiting combinatorial properties of the graphs which lack small forbidden induced subgraphs.

One example of this two-phase approach is that of the algorithm provided in [Kra+06]. It is a linear time algorithm for detecting a hole or an at form a non-interval graph. It first calls the linear time hole detection algorithm described in [TY84] (phase 1). This either returns a hole, or reduces the problem to finding an at in a chordal graph (phase 2). As noted in [Cao14], we can turn the algorithm from [Kra+06] into a linear time algorithm which returns either a hole or a caw. We will refer to this algorithm as `find_forbidden`$(\cdot)$. We will also be calling the hole-detecting algorithm in [TY84] `find_hole`$(\cdot)$.

The problem of the two-phase approach is the fact that, in general, small obstructions are not likely to be found in linear time. Thus, the natural choice of setting $\mathcal{F}_B := \mathcal{F}_{LI}$ and carry out the two-phase approach described above does not lead to optimal results. In order to achieve linear time in the interval vertex deletion problem we relax phase 1 to either disposing of an induced subgraph of G in $\mathcal{F}_{LI}$ or finding a certain clique decomposition of G which allows us to remove the remaining obstructions. As an important note on what follows, we remark that whenever we find an induced subgraph of G in $\mathcal{F}_{LI}$ during the execution of the vertex deletion algorithm, we immediately branch on deleting one vertex of it.

We finally note that both Algorithm 2 and the vertex deletion algorithm in [Cao14] pose themselves the constraint of modifying the input graph with at most k modifications. What they do in practice, however, is to find the minimal number of modifications needed so that the input graph falls into the desired class if such number does not exceed k, or otherwise return "False".

Thus, we assume without loss of generality that the input graph G does not contain any universal vertices. Indeed, if it does, we can strip them off, because any minimal modification of G into an interval graph does not include universal vertices. To see why this is true, suppose that there is a universal vertex u into a minimal set S of vertices to be deleted to make G into an interval graph G_I. G_I has an interval model $\mathcal{I} = (I_v)_{v \in V(G_I)}$. Adding the interval $\bigcup_{v \in V(G_I)} I_v$ to $\mathcal{I}$ gives an interval model of $G \setminus (S \setminus \{u\})$ which contradicts the minimality of S.

3 Modules

One important notion behind the vertex deletion algorithm in [Cao14] is that of the module of a graph.

Definition 3.1. Given a graph G, a set $M \subseteq V(G)$ is a module if all $v \in M$ have the same neighborhood outside M. We call all singletons and $V(G)$ *trivial* modules.

G is said to be *prime* if it only has trivial modules.

A module M is said to be *strong* if for all modules M' that intersect M, either $M \subseteq M'$ or $M' \subseteq M$.

Also, a module M is *maximal* if the only module of G properly containing it is $V(G)$.

We provide some observations. One can check by hand that the 4-hole is the only non-prime graph in $\mathcal{F}_I$.

Notice that prime graphs with at least one edge are always connected. Otherwise, any of it's connected components with at least two vertices would form a non-trivial module of such graph.

Notice that two modules of a graph G are either non-adjacent or completely adjacent.

We also immediately see that two modules which are both strong and maximal are disjoint. Moreover, since singletons are strong, every vertex v appears in one maximal strong module. These observations give the following propositions.

Proposition 3.2. The maximal strong modules of a graph G define a partition of its vertices.

Thus, maximal strong modules of G define an equivalence relation on its vertices. Morever, since two modules are either non-adjacent or completely adjacent, the maximal strong modules of a graph G define a quotient graph $Q(G)$ whose nodes are the maximal strong modules of G which are connected in $Q(G)$ if and only if they are completely adjacent in G.

For our purposes, an important property of maximal strong modules, is that we can find them in linear time (see [HP10]). Thus, they can be used for algorithmic purposes. We denote the linear-time subroutine computing Q by `compute_quotient`$(\cdot)$.

We immediately see that if G is not connected, its maximal strong modules are its connected components. Thus, $Q(G)$ is an edgeless graph. Likewise, if the complement of G, which we call $\overline{G}$, is not connected, we get that $Q(G)$ is a complete graph. In general, it holds that G and $\overline{G}$ have the same set of maximal strong modules and $Q\left(\overline{G}\right) = \overline{Q(G)}$.

If neither G nor $\overline{G}$ are connected, we have the following proposition.

Proposition 3.3. If G and its complement are connected, then the quotient graph Q whose vertex set is a set of maximum strong modules that partition $V(G)$ is prime.

Proof. Given a module M, let $\Gamma(M)$ denote the set of neighbors of the vertices of M which do not lie in M. Given a graph G, let $Q(G)$ denote its quotient graph where the chosen partition of $V(G)$ is made of maximum strong modules. Notice that G is connected if and only if $Q(G)$ is. Finally, let $\overline{G}$ be the complement of G.

Now, suppose that G and $\overline{G}$ are connected. Let $M = \{M_1, M_2, \ldots, M_p\}$ be the maximum strong modules that form a partition of $V(G)$ and let $Q(G)$ be the associated quotient graph. Suppose that $Q(G)$ contains a non-trivial module $\mu_1 \subseteq M$. Since the set $L_1 = \bigcup_{M_i \in \mu_i}$ is a module in G and $M_i \subseteq L_1$ for some i, we have that L_1 is not a strong module of G by the maximality of the modules in M.

Therefore, there exists a module $L_1' \subseteq V(G)$ such that $L_1' \cap L_1 \neq \emptyset$, but none of the two sets is included into the other. Since the modules in M are all strong, we immediately have that L_1' is the disjoint union of some modules in M. Call the set of such modules μ_1'.

We can see that μ_1' is a module in $Q(G)$ and $\mu_1 \cap \mu_1' \neq \emptyset$. Moreover, μ_1 and μ_1' are not comparable with inclusion.

Notice that $\mu_1 \cup \mu_1'$ must be a proper subset of M, because, otherwise, any vertex in the intersection of $\mu_1 \cap \mu_1'$ would be a disconnected component of $Q\left(\overline{G}\right)$, which cannot be since $\overline{G}$ is connected.

If $\Gamma(\mu_1) \subseteq \mu_1'$, then $\mu_1 \cup \mu_1'$ would be a component of $Q\left(\overline{G}\right)$, but this would mean that $Q\left(\overline{G}\right)$ has more than one connected component, since $\mu_1 \cup \mu_1'$ is a proper subset of M, as proved above. Therefore $\Gamma(\mu_1) \not\subseteq \mu_1'$.

All in all, we proved that $\mu_1 \cup \mu_1' := \mu_2$ is a non-trivial module of $Q(G)$. We can do the same on μ_2 recursively until we obtain that $\mu_l \cup \Gamma(\mu_l)$ forms a partition of M with μ_l a non-trivial module of $Q(G)$, and so $\Gamma(\mu_l) \neq \emptyset$. But this means that $\Gamma(\mu_l)$ forms a disconnected component of $Q\left(\overline{G}\right)$, which is a contradiction. Thus, $Q(G)$ does not have any non-trivial modules. □

We automatically have the following corollary.

Corollary 3.4. Given a graph G, $Q(G)$ is either edgeless, complete or prime.

One reason why quotient graphs $Q(G)$ decided by maximal strong modules of a graph G are so useful, is that, while $Q(G)$ is smaller that G, it still contains a some useful information about G itself. We express this fact through the following lemma.

Lemma 3.5. Let G and $Q(G)$ be defined as above. Let X be an induced subgraph of G. Then, if X is prime, either it is contained into a maximal strong module of G or X is an induced subgraph of $Q(G)$.

Conversely, if X is any induced subraph of $Q(G)$, then X is also a induced subgraph of G.

Proof. For any maximal strong module M of G, the set $M' := V(X) \cap M$ is a module of X. Therefore it has to be either a singleton or the entire $V(X)$. In the latter case, X is a subgraph of $G[M]$. In the former, by the definition of modularity and by the fact that maximal strong modules partition $V(X)$, the maximal strong modules intersecting X induce a subgraph in $Q(G)$ which is isomorphic to X.

Conversely, if X is an induced subgraph in $Q(G)$, we define S to be the set containing exactly one representative of each module of G which is also a vertex of X. Then, by the definition of $Q(G)$, the induced subgraph $G[S]$ is isomorphic to X. □

This lead us directly to one of the main propositions of [Cao14].

Proposition 3.6. Let $* \in \{\mathrm{I}, \mathrm{LI}\}$. We have that $G \in \mathcal{F}_*$ and does not have a universal vertex if and only if all of the following holds.

1. The quotient graph Q decided by maximal strong modules is not a clique;
2. Q is in $\mathcal{F}_*$;
3. $G[M] \in \mathcal{F}_*$ for all modules M which are simplicial in Q;
4. $G[M]$ is a clique for all modules M which are non-simplicial in Q.

Proof. We prove the forward direction by contraposition.

Suppose that Q is a clique. If there is a module M deciding Q which forms a clique in G, we have a universal vertex. Thus, take two modules M and M' deciding Q and consider vertices $v_1, v_2 \in M$ and $u_1, u_2 \in M'$ with $v_1 \not\sim v_2$ and $u_1 \not\sim u_2$. This forms an induced 4-hole.

Suppose that a quotient graph Q decided my maximal strong modules of G is not in $\mathcal{F}_*$. Then, there is a graph $H \in \mathcal{F}_*$ induced in Q. Thus H is an induced subgraph in G.

Suppose that $G[M] \notin \mathcal{F}_*$ for some module M which is simplicial in Q. Then $G[M]$ contains an induced graph $H \in \mathcal{F}_*$. Thus, so does G.

Suppose that $G[M]$ is not a clique for M a non-simplicial vertex of Q. Then, we get that $|M| \geq 2$ and that there are M_1 and M_2 adjacent to M in Q such that M_1 and M_2 are not adjacent. Take two non-adjacent $u_1, u_2 \in M$ and any $v_1 \in M_1$ and $v_2 \in M_2$. We have that u_1, v_1, u_2, v_2 form an induced 4-hole since v_1 and v_2 are not adjacent in G.

We prove the backward direction by contraposition.

Suppose that $G \notin \mathcal{F}_*$ with G with no universal vertices. Thus, Q is not a clique. This settles point 1.

Since $G \notin \mathcal{F}_*$, G has an induced subgraph $H \in \mathcal{F}_*$. If H is prime, i.e., H is not a 4-hole, then either H is an induced subgraph in Q or in one of the modules deciding Q (otherwise it would contradict the primality of H). If H is not prime, i.e. a 4-hole, it either is completely contained into one module or each vertex is contained into a different module (thus settling point 2 and 3), or it is contained in two or three modules.

In this last case notice that at least one module contains exactly two non-adjacent vertices. So it is not a clique module. We also claim that at least two non-adjacent vertices of the 4-hole lie in a non-simplicial vertex. Indeed, suppose that the vertices of the 4-hole lie in exactly three modules,

then two of these modules must contain exactly one vertex of the 4-hole and must be non-adjacent while being both adjacent to the third one, which is thus non-simplicial. Otherwise, suppose that all the vertices of the 4-hole, lie in exactly two modules M_1 and M_2. If M_1 is simplicial, then there must be a module M_3 not adjacent to M_1, because otherwise, either Q is a complete graph or $\{M_1, M_2\}$ forms a module of Q, which is prime (since it is not a clique nor it is edgless, since M_1 and M_2 are adjacent), a contradiction. Therefore M_1 and M_3 are not adjacent and M_2 is not simplicial.

We just found a non-simplicial vertex in Q which contains two non-adjacent vertices, and thus it is not a clique module. This settles point 4. □

Proposition 3.6 tells us that, if we ensure that G does not have universal vertices, and once $Q(G)$ is compted by `compute_quotient` (G), in order to ensure that G is interval, we only need to check that $Q(G)$ is interval itself, non-simplicial modules are cliques and that $G[M]$ is an interval graph for all simplicial modules M.

In order to achive the minimal number of modifications while enforcing condition 3 of Proposition 3.6, we would like that, if $G[M]$ is not and interval graph, the graph obtained by substituting $G[M]$ by its largest induced interval subgraph is part of some minimum of modifications required to transform G into an interval graph. This is indeed the case, provided that $Q(G)$ is 4-hole free. Before proving so, we fist make an observation.

Maximal strong modules enjoy a remarkable "modularity" property. Let G be a graph such that $Q(G)$ is a prime graph. Let M be a module of G and let H be a graph of vertex set M'. Consider replacing $G[M]$ by H in such a way that $u \in V(G) \setminus M$ contained in the maximal strong module $M'' \in V(Q(G))$ and $v \in M'$ are adjacent if and only if M and M'' are adjacent in $V(Q)$. Then we have the following.

Lemma 3.7. Consider the graph G' obtained after replacing $G[M]$ by H. Then, M' is a maximal strong module of G' and $Q(G)$ is isomorphic to $Q(G')$.

Proof. First, we notice that, by construction, M' and all the modules of G except for M are still modules of G'. We are now going to prove that such modules are stong and maximal in G'. By abuse of notation, we denote $V(Q(G))$ by $V(Q)$ and $V(Q(G'))$ by $V(Q')$.

We first prove that they are strong by contradiction. Let X be a module of G' such that $M' \cap X \neq \emptyset$ and $M' \not\subseteq X$ and $X \not\subseteq M$. Let $\{M_1, M_2, \dots, M_p\}$ be the set of maximal strong modules in $V(Q)$ such that $X \cap M_i \neq \emptyset$. If X is a module of G', then, since modules are either completely adjacent or completely non adjacent, we deduce that $\{M_1, M_2, \dots, M_p\} \setminus \{M'\}$ forms a module of Q. Since such module is non-trivial (M is not in the module) and Q is prime, we get a contradiction. Therefore X is not a module of G' and M' is strong in G'.

Let $N \in V(Q) \setminus \{M\}$. Using the strength of M' in G' and that of N in G, we deduce that N is strong in G'.

Now we prove maximality. Suppose that for some $N \in (V(Q) \setminus \{M\}) \cup \{M'\}$ such module is not maximal in G'. By the strength of the modules in $V(Q)$, we deduce that there exists $\Gamma' \subsetneq (V(Q) \setminus \{M\}) \cup \{M'\}$ such that $N \in \Gamma'$ and $\bigcup_{L \in \Gamma'} L \subsetneq V(G')$ is a module of G'. Let $\Gamma = (\Gamma' \setminus \{M'\}) \cup \{M\}$ if $M' \in \Gamma'$ or $\Gamma = \Gamma'$ otherwise. The above observation imply that Γ is a non-trivial module of Q, which is a contradiction. Therefore all the modules in $(V(Q) \setminus \{M\}) \cup \{M'\}$ are strong and maximal in G' and cover $V(G')$.

It's clear by construction that Q' is isomorphic to Q. □

This property is useful because it allows to consider maximal strong modules as "black boxes" which can contain any subgraph.

We are now ready to prove an important result.

Theorem 3.8. ([Cao14]) Let G be a graph of which every 4-hole is contained is some maximal strong module, and let $G[U]$ be a maximum induced interval subgraph of G. For any maximal strong module M of G intersecting U, the set $M \cap U$ is a module of $G[U]$, and replacing $G[M \cap U]$ by any maximum induced interval subgraph of $G[M]$ in $G[U]$ gives a maximum induced interval subgraph of G.

Proof. The first assertion is proved by Lemma 3.7. We focus on the second one.

Let U' be the set of vertices that induces the new graph. Since $G[M \cap U]$ is interval, by the maximality of $M \cap U'$, we deuce that $|U'| \geq |U|$.

We are left to prove that $G[U']$ is interval. By way of contradiction, suppose that $G[U']$ is not interval. Thus, there exists a subgraph $X \in \mathcal{F}_I$ in $G[U']$. Since G is 4-hole free, by Proposition 3.6, at least one of $M \cap U'$ and $N(M) \cap U'$ induces a clique in G. Since G is 4-hole free, X must be prime and, by Lemma 3.5, X contains exactly one vertex of M. Call such vertex x. By assumption, there exists $x' \in M \cap U$. Let $X' := X \setminus \{x\} \cup \{x'\}$. We thus have $X' \subseteq U$, but $G[X']$ is isomorphic to $G[X]$. Thus contradicting the fact that $G[U]$ is an interval graph. □

This discussions suggests to strip G of its universal vertices and compute $Q(G)$. If $Q(G)$ is edgeless, then we solve each module individually. Or, if $Q(G)$ is a clique, we have two cases. Either find two non-clique modules of $Q(G)$, and thus we can easily find a 4-hole in G (just like in the proof of Proposition 3.6) and branch on deleting one vertex of it. We can also find only one non-clique module, in which case we solve it individually. We cannot have only clique modules in this case, otherwise we would have a universal vertex. Finally, if $Q(G)$ is prime, Theorem 3.8 and 3.6 tell us that if we transform $Q(G)$ into an interval graph (in particular, a 4-hole free graph) a solution is given by solving individually the simplicial modules in $Q(G)$.

Therefore, from now on, we will be looking only at the quotient graph $Q(G)$, which we will denote by Q for the sake of brevity. Moreover, we will be referring to the vertices of Q as the *modules* of Q, not to be confused with the *vertices* of the original graph G. We will also keep looking for minimal modifications turning Q in an interval graph (if it is not one already). Only, in this case, the minimality of modifications will not consist in deleting the minimal number of modules of Q but in deleting the modules of Q so to minimize the total number of vertices contained in such modules. For any set S of modules of Q we denote by $\#S$ the cardinality of their union (which is the sum of their cardinality, since they are disjoint).

4 Clique decompositions

We now introduce the notion of a clique decomposition of a graph Q.

Definition 4.1. Given a graph Q, we call a clique decomposition of Q a connected graph $\mathcal{K}$ whose vertex set is the set of (inclusion-wise) maximal cliques of Q (called bags) and edge set such that for all $v \in Q$, the subgraph induced by the set of bags containing v is connected.

It turns out that interval graphs are characterized by their clique decomposition.

Proposition 4.2. A graph Q is an interval graph if and only if it has a clique-path decompostion.

Proof. Suppose that a graph Q is interval. Let $\mathcal{I} = (I_v)_{v \in V(Q)}$ be an interval model of it. We assume without loss of generality that all I_v are closed intervals, all intersections $I_v \cap I_{v'}$ are not singletons and that such interval model is normalized, that is, $\inf\{a \mid [a,b] \in \mathcal{I}\} = 0$ and $\sup\{b \mid [a,b] \in \mathcal{I}\} = 1$.

For any $a \in [0,1]$, we define $V(a) = \{v \mid a \in I_v\}$. Now, we define inductively the family $(a_0, \ldots, a_l)$ such that $a_i \in [0,1]$ for all $0 \leq i \leq l$.

We define $a_0 := 0$ and $a_{i+1} := \sup\{a \in [a_i, 1] \mid V(a_i) \subseteq V(a)\}$.

Let $(a_0, a_1, \ldots, a_l)$ be the family obtained at the end of this construction. It is clear that $V(a_1), \ldots, V(a_l)$ are the maximal cliques of G. Moreover, if $v \in V(a_i) \cap V(a_k)$, then $v \in V(a_j)$ for all $i < j < k$. Thus yielding a clique-path decomposition of Q.

Conversely, let $M_0, \ldots, M_{l-1}$ be a clique-path decompostion of a graph Q. Let $M := \bigcup_{i=0}^{l-1} M_i$. For all $v \in M$, let $\texttt{left}(v)$ (resp. $\texttt{right}(v)$) be the smallest (resp. largest) index i such that $v \in M_i$. For $v \in V(Q)$, let $I_v := \left[\texttt{left}(v) - \frac{1}{3}, \texttt{right}(v) + \frac{1}{3}\right]$. The interval model $\mathcal{I} = (I_v)_{v \in V(Q)}$ represents a graph with the given clique-path decomposition. □

Given Proposition 4.2, in order to transform a graph Q into an interval graph, it suffices to to modify (always with the minimal number of vertex deletions) a clique decomposition of Q into a clique-path decomposition.

Moreover, as proved in [FG65], one can check in linear time whether a graph is interval and, in case of a prositive answer, output a clique-path decomposition of it. We will denote such function by `compute_clique_path` $(\cdot)$.

5 Olive-ring decomposition

In this section we will give an outline of the results and the ideas behind the notion of the olive-ring decomposition used in the vertex deletion algorithm in [Cao14].

An olive-ring is a graph made of a main cycle with all other vertices attached to it having degree one.

At this point of the algorithm, the author of [Cao14] focuses on finding an olive-ring decomposition of Q by first computing a hole decomposition of $Q - S(Q)$. The result is very technical but we would still like to give a glimpse into the technique used. The proofs of the statements in this section require some pretty involved constructions but they are not very hard to check by hand. For the detailed explanations we refer the reader to sections 4 and 5 of [Cao14].

We start by using the algorithm `find_hole` (Q) to find a hole in Q if there is one. If none is found, then we can apply the following lemma.

Lemma 5.1. ([Cao14]) Let W be a large caw of a prime graph Q, then we can in find a subgraph of Q in $\mathcal{F}_{LI}$ if the shallow terminal of W is non-simplicial in Q.

This lemma tells us that, if the graph Q is chordal, then the graph $Q - S(Q)$ is either interval, or we can return a subgraph of Q in $\mathcal{F}_{LI}$. We can subsequently compute a clique-path decomposition of $Q - S(Q)$ with the linear-time subroutine `compute_clique_path` $(Q - S(Q))$. and then join the two ends of the path without breaking any of the conditions of the definition of a clique decomposition, thus getting a clique hole decomposition of $Q - S(Q)$.

On the other end, if Q is not chordal, we must use a more sophisticated tool. We first suppose that the hole returned by the function `find_hole` (Q) is long, otherwise, we just return the short hole. We have the following technical lemma which allows us to give a more precise description of the structure of the graphs we will be considering from now on.

Lemma 5.2. ([Cao14]) Let H be a hole of a prime graph Q. We can, in linear time, find a subgraph of Q in $\mathcal{F}_{LI}$ if there exists a vertex v satisfying one of the following three conditions:

1. the neighbours of v in H are not consectuive;
2. v is adjacent to $|H| - 2$ or more vertices in H;
3. v is non-simplicial and nonadjacent to H.

Moreover, if there exists a subset $U \subseteq Q$, such that $Q[U]$ is connected and the neighbors of U in H do not induce a subpath, we can return in linear time a subgraph of Q in $\mathcal{F}_{LI}$.

Therefore, if Q is a locally interval graph, for any vertex h of H, the subgraph $Q - N[h]$ must be chordal, otherwise h and any hole of $Q - N[h]$ will satisfy condition 3 of Lemma 5.2. Moreover, combining lemmas 5.1 and 5.2, we deduce that $Q - N[h] - S(Q)$ is an interval graph. In order to exploit this observation, we require the more sophisticated notion of an auxiliary graph.

From now on, when we consider a hole H, we will label its consecutive vertices as $h_0, h_1, \ldots, h_{|H|-1}$, where indices are understood modulo $|H|$. Note that to univocally assign these labels, it suffices to specify two consecutive vertices h_0 and h_1 of H.

By lemma 5.2, we can assume that the vertices in $N[v] \cap H$ induce a subpath for any vertex v and any fixed hole H. Thus, once we are given a hole H, we can define for all vertices v adjacent to it, the indices `first` (v) and `last` (v) such that

- $-|H| < \texttt{first}(v) \leq 0 \leq \texttt{last}(v) < |H|$, if $v \sim h_0$; or
- $0 < \texttt{first}(v) \leq \texttt{last}(v) < |H|$, otherwise.

Now, starting from the hole returned by $\texttt{find\_hole}(Q)$, we want to find a potentially new hole on which we enforce some local properties on one of its vertices (in fact, on h_0). We have the following.

Lemma 5.3. ([Cao14]) We can, in linear time, find either a subgraph of Q in $\mathcal{F}_{LI}$, or a hole H with a vertex h_0 such that for every vertex $v \in N(h_{-1}) \cap N(h_1)$ we have $N[v] \subseteq N[h_0]$.

From now on, we assume that the hole H we are working with has the properties stated in Lemma 5.3. We observe that by Lemma 5.2, we can assume that for all vertices v adjacent to H, $N[v] \cap V(H)$ induces a subpath. Also, by Lemma 5.3, we can assume that every common neighbor v of h_1 and h_{-1} is adjacent to h_0. Lemma 5.3 also tells us that for any v adjacent to h_0, $v \not\sim h_2$ and $v \not\sim h_{-2}$, since $N[v] \subseteq N[h_0]$. We can also assume that such v is adjacent to at least one of h_1 or h_{-1}. Otherwise, since $v \sim h_0$, $v \not\sim h_2$ and $v \not\sim h_{-2}$, if u is adjacent to H, we call the second part of Lemma 5.2 with $U = \{v, u\}$; otherwise, we just return *long claw* $\{u, v, h_0, h_1, h_2, h_{-2}\}$. Finally, for any $v \sim h_0$, such that $v \sim u$ with $u \not\sim h_0$, we can assume that v is not adjacent to all of h_0, h_1 and h_{-1}. Otherwise, we would have $N[v] \subseteq N[h_0]$ and thus $u \in N[h_0]$, which is a contradiction. All in all, we have that for all $v \sim h_0$, such that $v \sim u$, with $u \not\sim h_0$, we can assume that v is adjacent to exactly one of h_1 and h_{-1} and so, that either $\texttt{first}(v) < \texttt{last}(v) = 0$, or $0 = \texttt{first}(v) < \texttt{last}(v)$. We are now ready to give the definition of the auxiliary graph $\mho(Q)$.

Definition 5.4. Given a graph Q, and a hole H of Q with the properties required by Lemma 5.3. Let $T := N[h_0]$, let L and R be two distinct copies of T and let $\overline{T} := V(Q) \setminus T$. In order to distinguish two copies of the same vertex $v \in T$, we denote by v^l its copy in L and by v^r its copy in R. The vertex set $\mho(Q)$ is the union $\overline{T} \cup L \cup R \cup \{w\}$, where w is an additional fresh vertex. For every edge $uv \in E(Q)$, we add to the edge set of $\mho(Q)$

- an edge uv if neither u not v is in T;
- two edges $u^l v^l$ and $u^r v^r$ if both u and v are in T ;
- an edge uv^l if $u \in \overline{T}$, $v \in T$ and $0 = \texttt{first}(v) < \texttt{last}(v)$.
- an edge uv^r and an edge wv^l if $u \in \overline{T}$, $v \in T$ and $\texttt{first}(v) < \texttt{last}(v) = 0$.

Notice that, by this definition, we have $h_1^l \sim h_2$ and $h_{-2} \sim h_{-1}^r$. Also, the addition of the vertex w is done so that we can "remember" which vertices in R are connected to $\overline{T}$ by looking at vertices in L. We add that the auxiliary graph of Q is also cheap to build.

Proposition 5.5. ([Cao14]) The number of vertices (resp. edges) of the graph $\mho(Q)$ are upper bounded by $2|G|$ (resp. $2\|G\|$). Moreover, an adjacency matrix of $\mho(Q)$ and $\mho(Q - S(Q))$, can be built in linear time.

The idea behind Definition 5.4 is that the auxilary graph either "opens up" all holes of Q, or we can find subgraphs of Q in $\mathcal{F}_{LI}$ in linear time. To make this more precise, we give the following lemma.

Lemma 5.6. ([Cao14]) If $\mho(Q)$ is not chordal, then we can find a subgraph of Q in $\mathcal{F}_{LI}$.

We also have a version of Lemma 5.1 for $\mho(Q)$.

Lemma 5.7. If $\mho(Q - S(Q))$ is not an interval graph, then we can find a subgraph of Q in $\mathcal{F}_{LI}$.

Thus, we can assume that $\mho(Q - S(Q))$ is an interval graph which we can use to construct a hole decomposition of $Q - S(Q)$. Notice that, given a vertex v and a set of vertices S, one can check in $\mathcal{O}(d(v))$ time whether v is adjacent to S or not. Thus, we can check in linear time whether any of the modules of $Q - S(Q)$ is adjacent to the set $L \cup R \cup \big(\overline{T} \cap V(H)\big)$. If any such module is found, then, it is surely not simplicial in Q, we can thus call Lemma 5.2 on H and v as input and return a subgraph of Q in $\mathcal{F}_{LI}$. We can thus assume that all vertices of $Q - S(Q)$ are adjacent to H and so, all modules in $\overline{T}$ are adjacent to one among $h_1^l, h_2, \ldots h_{-1}^r$. Let $\mathcal{C}$ be the clique-path decomposition of $\mho(Q - S(Q))$ with consecutive bags numbered $K_0, K_1, \ldots, K_{|\mathcal{C}|-1}$. By the definition of the auxiliary graph, one

can show that either the vertex w is simplicial in $\mho(Q - S(Q))$ or we can return a subgraph of Q in $\mathcal{F}_{LI}$ in linear time [Cao14]. So, we can assume that w is simplicial in $\mho(Q - S(Q))$ and thus its closed neighborhood $N[w]$, is a maximal clique of $\mho(Q - S(Q))$. In this case we have that $N[w]$ is an end bag of $\mathcal{C}$. We can argue this by showing that every neighbor of $N[w]$ is connected to h_0^l in $\mho(Q - S(Q)) - N[w]$. In fact, we have that every vertex in $L \setminus N[w]$, is adjacent to h_0^l, by definition. Also, every vertex of $\overline{T}$ is conected to one of $h_1^l, h_2, \ldots, h_{-1}^r$. Since neither h_0^l nor h_0^r are simplicial in $\mho(Q)$, we get that $h_0^l, h_1^l, h_2, \ldots, h_{-1}^r, h_0^r$ remains a path in $\mho(Q - S(Q)) - N[w]$. Thus $\mho(Q - S(Q)) - N[w]$ is connected and $N[w]$ is an end bag of $\mathcal{C}$. Without loss of generality, assume that $N[w] = K_0$. Notice that since w is simplicial, the set $S = \{v^r \mid \text{ there exists } u \text{ with } v^r \sim u\}$ is a clique. Moreover, it is the minimal clique separator of $\{w\} \cup L \cup \overline{T}$ and $R \setminus S$. Therefore h_0^r only appears in bags whose index is larger than the largest index in of the bag containing S. Let l be the largest index such that $h_0^r \notin K_l$, One can show that the graph $\mathcal{K}$ constructed by taking the subpath $K_1, \ldots, K_l$ and joining its ends is a clique hole decomposition of $Q - S(Q)$.

All in all, we have the following theorem.

Theorem 5.8. ([Cao14]) Given a graph G, we can in linear time, either build its quotient graph Q and a hole decomposition of $Q - S(Q)$, or output a subgraph in $\mathcal{F}_{LI}$.

Moreover, if such hole decomposition is found, one can retrieve in linear time a normal Helly circular-arc interval model of $Q - S(Q)$ as well as the shortest hole of Q (if there is one).

Finally, if such a hole decomposition is found, its consecutive bags are labeled as $K_0, K_1, \ldots, K_{|\mathcal{K}|-1}$, with $|\mathcal{K}| \leq |Q|$.

We remark right away that, when we compute the shortest hole of Q, if such hole is short, that is, it is either a 4 or a 5-hole, we branch on deleting one vertex of it. We can thus assume that the shortest hole of Q has six or more vertices.

The advantage of having a hole decomposition $\mathcal{K}$ of Q is that every hole of Q spans all the bags of $\mathcal{K}$.

Proposition 5.9. Given a hole decomposition $\mathcal{K}$ of a graph Q, for all bags K of $\mathcal{K}$, and every hole H of Q, there exists a vertex v of H contained in K.

Proof. By way of contradiction, suppose that there exists a bag K which does not contain any vertex of a hole H.

For all vertces v of H, let K_v denote the set of bags containing v. Let K' be the closest bag to K (say counterclockwise in $\mathcal{K}$) containing a vertex v of H. Let u, w be the two neighbors of v in H. Since $u \not\sim w$, we have $K_u \cap K_w = \emptyset$. Thus, at least one of K_u and K_w is contained in K_v. Suppose without loss of generality that $K_u \subseteq K_v$. Let $u' \neq v$ be the other neighbor of u (so $u' \neq w$ since H has at least four vertices). Since $K_u \cap K_{u'} \neq \emptyset$ and $K_u \subseteq K_v$, we have $K_{u'} \cap K_v \neq \emptyset$ and so $v \sim u'$, which gives us a contradiction. □

Remark 5.10. Thus, it suffices to make the hole decomposition into a path to break all the holes in the original graph. However, in our case, we want to achieve the minimal number of deletions needed to break such holes.

We now introduce some notation. For any vertex v, let $\texttt{left}(v)$ and $\texttt{right}(v)$ be defined as the indices such that $v \notin K_{\texttt{left}(v)-1}$, $v \notin K_{\texttt{right}(v)+1}$ and $v \in K_i$ for all $i \in [\texttt{left}(v), \texttt{right}(v)]$, where indices here are understood modulo $|\mathcal{K}|$. The existence of the above indices is assured by the fact that the bags containing any vertex v induce a proper subpath in $\mathcal{K}$.

With this notation, we are ready to present another important result.

Lemma 5.11. ([Cao14]) Given a hole decomposition of $Q - S(Q)$, we can, in linear time, construct a clique decomposition of Q which is an olive-ring. Moreover, a module of Q does not appear in any bag of the main cycle if and only if it is a terminal of a caw of which all other modules are in the main cycle.

Proof. Before presenting the proof, we remark that for any $s \in S(Q)$ the set $N[s]$ is an inclusion-wise maximal clique. It is thus a bag of any clique decomposition of Q.

Let $\mathcal{K}$ be the hole decomposition of $Q - S(Q)$. Traversing all bags in $\mathcal{K}$, we record $\texttt{left}(v)$ and $\texttt{right}(v)$ for all v modules of Q. We also record $|K_i \cap K_{i+1}|$ for all $i = 0, \ldots, |\mathcal{K}| - 1$. We finally store, for every module $s \in S(Q)$, the values $p := \max_{v \in N(s)} \texttt{left}(v)$ and $q := \min_{v \in N(s)} \texttt{right}(v)$, so that $N(s) \subseteq K_l \cap K_{l+1}$ for all $l \in [p, q-1]$.

We do the following.

- If $p = q$ and $K_p = N(s)$, then we add s to K_p;
- if there exists l such that $l \in [p, q-1]$ and $|K_l \cap K_{l+1}| = |N(s)|$, then we insert $N[s]$ as a new bag between K_l and K_{l+1};
- otherwise, add $N[s]$ as a pendant bag to K_p.

It is easy to check that this pocedure yileds an olive-ring decomposition of Q. Now, let $P(Q)$ be the set of modules that appear only in pendant bags (so, clearly, $P(Q) \subseteq S(Q)$). We notice in passing, that the addition of the simplicial modules in the main cycle keeps true the property of the existence of a normal Helly circular-arc model of subgraph induced by the modules appearing in the main cycle of the decomposition, which in this case is $Q - P(Q)$.

We now take care of the second assertion of the lemma. Suppose that a module of Q is a terminal of a caw of which all other modules are in the main cycle. If such module appeared itself of the main cycle then, since no caw has a hole, the fact of having a normal Helly circular-arc model of $Q - P(Q)$, would imply the existence of an interval model of a caw, which is impossible.

On the other hand, let v be a module in $P(Q)$. So, there are vertices $x \in K_{p-1} \setminus K_p$ and $y \in K_{q+1} \setminus K_q$. Since $(K_l \cap K_{l+1}) \setminus N(s)$ is non-empty, there exists an $x - y$ path entirely contained in the main cycle and avoiding neighbors of s. We can also find in linear time an $s - x$ path with only one intermediate vertex $x' \in N(s)$ such that $\texttt{right}(x') = p$ and an $s - y$ path with only one intermediate vertex $y' \in N(s)$ such that $\texttt{left}(y') = q$. Such path witness the at $\{s, x, y\}$.

The union of these paths contains a caw. Since all other vertices are from the main cycle, this caw necessarily contains s (since otherwise, as previously remarked, we would get an interval model of a caw, which is impossible). Since s is simplicial, it is a terminal of this caw. Which is the desired result. □

We will call the algorithm that either returns an olive ring decomposition or a subgraph in $\mathcal{F}_I$, $\texttt{decompose}(\cdot)$.

Corollary 5.12. Since caws do not have circular-arc models, the subgraph $Q - P(Q)$ is a normal Helly circular-arc graph which is also maximal in Q.

As a last remark, it is easy to see that, given a hole decomposition $\mathcal{K}$ for $Q - P(Q)$, a circular-arc model with circle length $|\mathcal{K}|$ for $Q - P(Q)$ is given by

$$A_v = \begin{cases} \left[\texttt{left}(v) - \frac{1}{3}, \texttt{right}(v) + \frac{1}{3}\right], & \text{if } \texttt{left}(v) > 0, \\ \left[|\mathcal{K}| - \frac{1}{3}, \texttt{right}(v) + \frac{1}{3}\right], & \text{if } \texttt{left}(v) = 0. \end{cases}$$

Moreover, since holes (of $Q - P(Q)$) are of length at least 4, such circular arc model is also normal and Helly.

6 From olive-ring to path decomposition

Now that we have an olive-ring decomposition of Q, we only need to strip off its pendant bags and break its main cycle in order to get an interval graph. However, we need to do so using the minimum possible number of vertex deletions.

In order to strip off the pendant bags, we need to introduce the notion of a minimal frame of a caw.

Definition 6.1. Given a frame F of a large caw, we denote by $[F]$ the set of bags $\texttt{right}(l), \ldots, \texttt{left}(r)$, (the inclusion-wise minimal set of consecutive bags whose union contains all non-terminal modules of F). We denote by $I(F) = \bigcup_{l \in [\texttt{right}(l)+1, \texttt{left}(r)-1]} K_l \setminus N(Q)(s)$, the set of internal modules of F. $I(F)$ is the set of modules that can be used to construct the base of a caw with frame F.

F is said to be minimal if there is no other frame F' such that $[F'] \subseteq [F]$.

Now consider, if there is one, a minimal frame F in Q. We have the following technical lemma whose proof can be found in [Cao14].

Lemma 6.2. ([Cao14]) We can, in linear time, check whether or not Q has a caw. If it is large, we can also return a minimal frame F of it such that the following holds.

Let $G[U^*]$ be a maximum induced interval subgraph of G such that U^* intersects every module of F. Let $S_i := K_i \cap K_{i+1} \setminus N_Q(s)$ where $\texttt{right}(v) \leq i < \texttt{left}(v)$ and let l be such that $\#S_l$ is minimum. Then, there exists a maximum induced interval subgraph $G[U]$ disjoint from all modules in S_l. Moreover, S_l can be found in linear time.

We will denote the function of Lemma 6.2 returning a minimal frame F by $\texttt{find\_minimal\_frame}(\cdot)$.

With this lemma, we ensure that there always exists a minimum solution that deletes either one module from F or all the modules in S_l.

Even though the proof of Lemma 6.2 is very technical, the idea behind it is simple. When we want to dispose of a caw, we can either branch on deleting one vertex from its frame or delete the smallest $l - r$ clique separator in the subgraph (of Q) induced by $\{l, r\} \cup I(F)$.

Suppose that S_l is deleted. Then, we also dispose of all the caws with frames F' such that $[F] \subseteq [F']$. Moreover, let S_l be defined as above. It is easy to see that we can insert the pendant bag v belonging to F in between K_l and K_{l+1} without breaking any condition of the definition of a clique decomposition. This also preserves the fact that the graph $Q - (P(Q) \setminus v)$ is a normal Helly circular-arc graph. This procedure allows us to strip all the pendant bags of the olive-ring decomposition possibly reinserting some of them in the main cycle obtaining a clique-hole decomposition. For the sake of notation, we renumber the bags of the clique-hole decomposition so that consecutive bags have consecutive indices.

The last thing to understand is how to break all holes of $Q - P(Q)$ (if any are left) given a hole decomposition $\mathcal{K}$. We introduce the notion of a hole cover of a graph G.

Definition 6.3. Given a graph G, a *hole cover* of G is a set of vertices V_- such that $G - V_-$ is a chordal graph.

We have the following lemma.

Lemma 6.4. Given any clique-hole decomposition $\mathcal{K}$ of a normal Helly circular-arc graph Q, the hole covers of Q are given by the the sets that contain the intersection of consecutive bags.

Proof. On the one hand, take $M := K_l \cap K_{l+1}$ for K_l and K_{l+1} two consecutive bags of $\mathcal{K}$. By the chordlessness property of graphs, for any hole H and any vertex v of H, we have that v is at the intersection of two consecutive bags. Since the the set of bags containing v induces a connected path in $\mathcal{K}$, so a subpath of $\mathcal{K}$, v is also at the intersection of two bags. Moreover, by Proposition 5.9, all intersections of consecutive cliques contain at least one vertex (actually, exactly two) of any hole. Thus M is a hole cover of Q.

On the other hand, take a hole separator M of Q. Then it is a clique and it must contain the intersection of two consecutive bags. If not, we can find the minimum non-trivial cycle C from any vertex in $K_0 \cap K_1$ to itself that avoids M. Such non-trivial cycle exists because for any two consecutive bags of $\mathcal{K}$ there is a vertex v in their intersection which is not contained in M.

Since Q is a normal Helly circular-arc graph, C must have four or more vertices. Moreover, by the minimality of C, it must be chordless. Thus, C is a hole. Which concludes the proof. □

From this lemma the following proposition is deduced immediately.

Proposition 6.5. ([Cao14]) Let $\mathcal{K}$ be the hole decomposition of Q. For any inclusion-wise minimal hole cover V_- of Q there exists an index l such that $V_- = K_l \cap K_{l+1}$.

It follows that, the minimal modification that breaks all holes of Q is the deletion of the vertices in $K_l \cap K_{l+1}$ so that $\#(K_l \cap K_{l+1})$ is minimal.

7 The algorithm

We are now ready to present the pseudocode of the discussed procedure in Algorithm 3. We recall that, whenever we find a small caw in any of our subroutines, we implicitly branch on deleting one vertex of it.

We note in passing that the vailidity of line 32 in Algorithm 3 is given by Theorem 3.8.

The correctness of Algorithm 3 was shown throughout this text. As for its complexity, each branching step has at most 8 sub-instances (this maximum is attained only at line 42) each of which can be computed in linear time. Therefore, there are at most 8^k total sub-instances and the total complexity of the agorithm is $\mathcal{O}\left(8^k \cdot (|G| + \|G\|)\right)$. In the case in which G is a connected graph, this reduces to $\mathcal{O}\left(8^k \cdot \|G\|\right)$.

We also can turn this algorithm into an approximation scheme in the following way. Whenever we return a small caw or a short hole, we delete the whole subgraph. When we branch on deleting some entire modules at line 42 of Algorithm 3, we only delete exactly one vertex from each module of the frame F and exactly one vertex from the union of the modules in S_l. At least one of these deletions is in some minimum modification and so the number of minimum modifications of the remaining graph decreases by at least one. Since we also have that small caws and short holes have at most 6 (≤ 8) vertices, we have a performance ratio of 8. Moreover, the algorithm clearly stops after at most $\mathcal{O}(|G|)$ deletions and thus such algorithm can beg implemented in $\mathcal{O}(|G| \cdot (|G| + \|G\|))$ time.

We have the following theorem

Theorem 7.1. Algorithm 3 solves the minimum vertex deletion problem for interval graphs in $\mathcal{O}\left(8^k \cdot (|G| + \|G\|)\right)$ time. Moreover there exists an $8 - \mathcal{O}(|G| \cdot (|G| + \|G\|))$ approximation scheme of the same problem.

8 Concluding remarks

The strategy used by the algorithm presented in this text that of first computing an olive-ring decomposition or returning a small caw or a short hole; and then stripping the leaves of the olive-ring decomposition and breaking its main cycle.

When first approaching such algorithm, our goal was to achieve an FPT algorithm and/or an appoximation scheme similar to the one presented in [Cao14], but relative to the class of chordal graphs (also known as triangulated graphs), which are more general than interval graphs as they can be characterized as intersection graphs of a family of subtrees of a given tree, which is a more complex topological object than an interval (see [Gol04]). The idea of computing a clique decomposition of which reflects a normal and Helly circular-arc property of the underlying graph might be useful in the solution of this problem. However, the clique decomposition might be much more involved than an olive-ring. Moreover, the ways to compute such decomposition remains unclear. Indeed, the algorithm presented in this text, relies heavily on the fact that, whenever we find a small caw, we branch on disposing of it and get a graph with more precise combinatorial properties. The fact that holes are the only minimal obstructions of chordal graphs make the main technical lemmas of [Cao14] useless. The details that need to be filled in order to get efficiently to a useful clique decomposition thus need some more pondering.

References

[LB62] C. Lekkeikerker and J. Boland. "Representation of a finite graph by a set of intervals on the real line". eng. In: *Fundamenta Mathematicae* 51.1 (1962), pp. 45–64. URL: `http://eudml.org/doc/213681`.

Algorithm 3: Interval vertex deletion

Input: A graph G and an integer k representing the maximum number deletions.
Output: A minimal set $V_{\min}$ of vertices of G such that $G - V_{\min}$ is interval, "No" otherwise.

```
interval_vertex_deletion(G, k)
    V_min ← ∅;
    k_min ← k;
    if k_min < 0 then
        return "No";
    if find_forbidden(G) = "No" then
        return V_min ;
    strip G of its universal vertices;
    Q ← compute_quotient(G);
    if Q is edgeless then
        for each module M of Q do
            Ṽ_min ← interval_vertex_deletion(G[M], k);
            if Ṽ_min = "No" then
                return "No";
            V_min ← V_min ∪ Ṽ_min;
            k_min ← k_min − |Ṽ_min|;
        return V_min ;
    if Q is a complete graph then
        while there exist two non-clique modules of Q do
            find a 4-hole and branch on deleting one vertex form it;
            k_min ← k_min − 1;
        find the only non-clique module of Q, call it M;
        return interval_vertex_deletion(G[M], k_min);
    K ← decompose(Q);
    if a short hole or a small caw is found then
        branch on deleting one vertex from it;
        k_min ← k_min − 1;
    while Q contains a non-simplicial module do
        find a 4-hole and branch on deleting one vertex from it;
        k_min ← k_min − 1;
    for each simplicial module M of Q do
        Ṽ_min ← interval_vertex_deletion(G[M], k_min);
        if Ṽ_min = "No" then
            return "No";
        V_min ← V_min ∪ Ṽ_min;
        k_min ← k_min − |Ṽ_min|;
    while K is not a hole do
        F ← find_minimal_frame(Q, K) ;
        if a small caw or a short hole is found then
            branch on deleting one vertex from it;
            k_min ← k_min − 1 ;
        branch on deleting either a module from F or S_l (as defined in Lemma 6.2)
    find the index l such that #(K_l ∩ K_{l+1}) is minimal (as defined in Proposition 6.5);
    delete all vertices in K_l ∩ K_{l+1} ;
    return V_min.
```

[FG65] D. R. Fulkerson and O. A. Gross. “Incidence matrices and interval graphs.” In: *Pacific Journal of Mathematics* 15.3 (1965), pp. 835–855.

[KD79] M. S. Krishnamoorthy and Narsingh Deo. “Node-Deletion NP-Complete Problems”. In: *SIAM Journal on Computing* 8.4 (1979), pp. 619–625. DOI: 10.1137/0208049. eprint: https://doi.org/10.1137/0208049. URL: https://doi.org/10.1137/0208049.

[TY84] Robert E. Tarjan and Mihalis Yannakakis. “Simple Linear-Time Algorithms to Test Chordality of Graphs, Test Acyclicity of Hypergraphs, and Selectively Reduce Acyclic Hypergraphs”. In: *SIAM Journal on Computing* 13.3 (1984), pp. 566–579. DOI: 10.1137/0213035. eprint: https://doi.org/10.1137/0213035. URL: https://doi.org/10.1137/0213035.

[Cai96] Leizhen Cai. “Fixed-parameter tractability of graph modification problems for hereditary properties”. In: *Information Processing Letters* 58.4 (1996), pp. 171–176. ISSN: 0020-0190. DOI: https://doi.org/10.1016/0020-0190(96)00050-6. URL: https://www.sciencedirect.com/science/article/pii/0020019096000506.

[Gol04] Martin Charles Golumbic. *Algorithmic Graph Theory and Perfect Graphs (Annals of Discrete Mathematics, Vol 57)*. NLD: North-Holland Publishing Co., 2004. ISBN: 0444515305.

[Kra+06] Dieter Kratsch et al. “Certifying Algorithms for Recognizing Interval Graphs and Permutation Graphs”. In: *SIAM Journal on Computing* 36.2 (2006), pp. 326–353. DOI: 10.1137/S0097539703437855. eprint: https://doi.org/10.1137/S0097539703437855. URL: https://doi.org/10.1137/S0097539703437855.

[HP10] Michel Habib and Christophe Paul. “A survey of the algorithmic aspects of modular decomposition”. In: *Computer Science Review* 4.1 (2010), pp. 41–59. ISSN: 1574-0137. DOI: https://doi.org/10.1016/j.cosrev.2010.01.001. URL: https://www.sciencedirect.com/science/article/pii/S157401371000002X.

[LSS13] Min Chih Lin, Francisco J. Soulignac, and Jayme L. Szwarcfiter. “Normal Helly circular-arc graphs and its subclasses”. In: *Discrete Applied Mathematics* 161.7 (2013), pp. 1037–1059. ISSN: 0166-218X. DOI: https://doi.org/10.1016/j.dam.2012.11.005. URL: https://www.sciencedirect.com/science/article/pii/S0166218X12004295.

[Cao14] Yixin Cao. *Linear Recognition of Almost Interval Graphs*. 2014. arXiv: 1403.1515 [cs.DM]. URL: https://arxiv.org/abs/1403.1515.