Revisiting the Recognition of Proper Interval Graphs

Lilian Markenzon *
NCE - Universidade Federal do Rio de Janeiro
markenzon@nce.ufrj.br

Christina F. E. M. Waga
IME - Universidade do Estado do Rio de Janeiro
waga@ime.uerj.br

Abstract

A well-known subclass of chordal graphs is formed by proper interval graphs. Due to their very special structural properties, several problems proved hard to solve for interval graphs can have better solutions for this subclass. In this paper, we address the recognition problem, proposing an update of one of the first existing linear algorithms. The outcome is a simple and efficient algorithm. In addition, we present a certifying algorithm for the recognition of proper interval graphs.

1 Introduction

A well-known subclass of chordal graphs is formed by proper interval graphs. Due to their very special structural properties, several problems proved hard to solve for interval graphs can have better solutions for this subclass. In this paper, we address the recognition problem, for which there are already some algorithms in the literature. We propose an update of one of the first linear algorithms presented. The resulting algorithm has time complexity of $O(n+m)$ and its implementation is very simple. In addition, we extend the proposal, showing a certifying recognition algorithm for the class.

*Partially supported by grants 304676/2013-6 and 473798/2012-3, CNPq, Brazil.

Let $G = (V, E)$ be a graph, with $|E| = m$, $|V| = n > 0$. The *set of neighbors* of a vertex $v \in V$ is denoted by $N(v) = \{w \in V \mid \{v, w\} \in E\}$ and its *closed neighborhood* is denoted by $N[v] = N(v) \cup \{v\}$. A vertex v is said to be *simplicial* in G if $N(v)$ is a clique in G. An *asteroidal triple* (AT) in a graph $G = (V, E)$ is a triple of distinct vertices such that each pair is connected by some path avoiding the neighborhood of the third vertex.

A graph G is said to be *chordal* when every cycle of length 4 or more has a *chord* (i.e. an edge joining two non-consecutive vertices of the cycle). Basic concepts about chordal graphs can be found in Blair and Peyton [2] and Golumbic [6]. All graphs in this text are assumed to be chordal.

A subset $S \subset V$ is a *vertex separator* for non-adjacent vertices u and v (a uv-separator) if the removal of S from the graph separates u and v into distinct connected components. If no proper subset of S is a uv-separator then S is a *minimal uv-separator*. When the pair of vertices remains unspecified, we refer to S as a *minimal vertex separator*.

A *clique-tree* of G is defined as a tree T whose vertices are the maximal cliques of G such that for every two maximal cliques Q and Q' each clique in the path from Q to Q' in T contains $Q \cap Q'$. The set of maximal cliques of G is denoted by $\mathbb{Q}$. Blair and Peyton [2] proved that, for a clique-tree $T = (V_T, E_T)$, a set $S \subset V$ is a minimal vertex separator of G if and only if $S = Q' \cap Q''$, for some edge $\{Q', Q''\} \in E_T$. Moreover, the multiset $\mathbb{M}$ of the minimal vertex separators of G is the same for every clique-tree of G. The *multiplicity* of the minimal vertex separator S, denoted by $\mu(S)$, is the number of times that S appears in $\mathbb{M}$. The algorithm presented in Markenzon and Pereira [10] computes the set $\mathbb{S}$ of minimal vertex separators of a chordal graph G and their multiplicities in linear time.

2 Interval graphs

In this section we review some concepts of interval graphs.

An *interval graph* is the intersection graph of a family I of intervals on the real line. There are several characterizations of this class.

Theorem 1 [8] *Let $G = (V, E)$ be a non-complete chordal graph. Then, G is an interval graph if and only if G is AT-free graph, i.e., G does not contain any asteroidal triple.*

Theorem 2 [5] *Let $G = (V, E)$ be a non-complete chordal graph. Then, G is an interval graph if and only if G has a clique-tree that is a path.*

The clique-tree which is a path will be called a *clique-path*, denoted by $\langle Q_1, \ldots, Q_q \rangle$.

Olariu [12] has defined an ordering of vertices which characterizes interval graphs.

Theorem 3 *A graph $G = (V, E)$ is an interval graph if and only if there exists a linear order $<*$ on V such that for every choice of vertices u, v, w*

$$u <* v <* w, \{u, w\} \in E \textit{ implies } \{u, v\} \in E. \tag{1}$$

The ordering $v_1 <* v_2 <* \ldots <* v_n$ is called a *greedy ordering.*

The following algorithm determines the greedy ordering of an interval graph. It walks through a clique-path, placing the vertices of the cliques in buckets, which are ordered by the reappearance of the vertices in subsequent cliques. A label $H(v)$, for $v \in V$, shows if v must be placed in the bucket or not, and it also points to the maximal clique in which the vertex appears for the last time; the label $B(v)$ points to the maximal clique in which it appears for the first time. Notice that the algorithm below is slightly improved, since in the original version proposed by Looges and Olariu [9], the buckets were ordered separately in a second step.

```
Algorithm Greedy-ordering;
Input: interval graph G and a clique-path ⟨Q_1,...,Q_q⟩;
Output: greedy ordering δ;
begin
   for v ∈ V do H(v) ← 0;          % initialization
   for i = 1,...,q do
      for v ∈ Q_i do              % vertices of the clique Q_i
         if H(v) = 0 then
            add v to bucket[i];   % bucket is a double linked list
            B(v) ← i;
         H(v) ← i;
         v is moved to the end of the list bucket to which it belongs;
   δ ← ⟨ ⟩;
   for i = 1,...,q do δ ← δ || bucket[i];
end.
```

The symbol $||$ represents the concatenation of sequences. At the end of the process, each list $bucket[i], 1 \leq i \leq q$, has at least one element.

Algorithm *Greedy-ordering* has time complexity of $O(n + m)$. Firstly, the label $H(v)$ is initialized for $v \in V$. Then the algorithm analyses the maximal cliques, which is known to be $O(n+m)$. Each vertex belongs only to one bucket. The implementation of *bucket* by double linked lists allows to move the vertices to the end of the list in constant time. Hence, the algorithm is linear.

3 Proper interval graphs

A *proper interval graph* is an interval graph that has an intersection model in which no interval properly contains another. If each interval in the model has unit length, G is called a *unit interval graph.* Roberts [14] has proved that the classes of proper interval graphs and unit interval graphs are the same. Corneil [3] summarized in the next theorem the results from Roberts [14, 15].

Theorem 4 *The following are equivalent:*

1. $G = (V, E)$ *is a unit interval graph.*
2. $G = (V, E)$ *is a proper interval graph.*
3. $G = (V, E)$ *is an interval graph with no induced claw* ($K_{1,3}$).
4. *There is an ordering of* V *such that for all* $v \in V$, $N[v]$ *is consecutive ("the neighborhood condition").*
5. *There is an ordering of* V *such that vertices contained in the same maximal clique are consecutive ("the clique condition").*

Recognition algorithms for proper interval graphs [9, 4, 3] are based on the following two approaches. In the first approach, we must previously recognize if the graph is an interval graph; in the second one, the recognition is performed directly, that is, without determining if the graph is an interval graph. The algorithm by Looges and Olariu [9] follows the first approach and it is criticized for this, since the recognition of an interval graph, although of linear complexity, has not a simple implementation. The determination of the greedy ordering supports their characterization of proper interval graphs, as we can see in the next theorem.

Theorem 5 [9] *A graph* $G = (V, E)$ *is a proper interval graph if and only if there exists a linear order* $<\cdot$ *on* V *such that for every choice of vertices* u, v, w

$$u <\cdot v <\cdot w \text{ and } \{u, w\} \in E \text{ implies } \{u, v\}, \{v, w\} \in E. \quad (2)$$

Corollary 5.1 *An interval graph is a proper interval graph if and only if a greedy ordering of the vertices of* G *satisfies condition (2).*

The ordering that obeys condition (2) is called a *canonical ordering.*

The canonical ordering is equivalent to the orderings presented in Theorem 4, as it is shown next.

Theorem 6 *Let $G = (V, E)$ be a proper interval graph. Then a canonical ordering $\sigma = \langle v_1, \ldots, v_n \rangle$ of V obeys the clique condition.*

Proof: Consider a maximal clique Q and a canonical ordering σ of G. Let $i = \min\{\ell : v_\ell \in Q\}$ and let $k = \max\{\ell : v_\ell \in Q\}$.

Consider vertices $v_{j_1}, \ldots, v_{j_s}$ such that $i < j_1 < \ldots < j_s < k$.

If $s = 1$ then, as σ is a canonical ordering, $\{v_i, v_{j_1}\}, \{v_{j_1}, v_k\} \in E$ and $v_{j_1} \in Q$. Suppose that $v_{j_1}, \ldots, v_{j_{s-1}} \in Q$. As σ is a canonical ordering and $\{v_{j_{s-1}}, v_k\} \in E$, then $\{v_{j_{s-1}}, v_{j_s}\}, \{v_{j_s}, v_k\} \in E$ and $v_{j_s} \in Q$. ∎

4 The recognition algorithm

The previous section has presented the theoretical framework for a recognition algorithm:

Step 1: recognize if G is an interval graph;

if so, build a clique-path $\langle Q_1, \ldots, Q_q \rangle$ and go to Step 2;

Step 2: build the greedy ordering δ;

Step 3: verify if δ obeys the neighborhood condition or the clique condition (Theorem 4).

The algorithm is simple but, as we already mentioned, its main drawback is Step 1, the need to recognize if the graph is an interval graph. Thus, we will modify this step, proving other properties of proper interval graphs. In order to develop our results, we recall another class of graphs, defined by Kumar and Madhavan [7].

A chordal graph is called a *uniquely representable chordal graph* (briefly *ur-chordal graph*) if it has exactly one clique-tree. An interval graph that is uniquely representable is called a *uniquely representable interval graph* (briefly *ur-interval graph*).

Theorem 7 [7] *Let $G = (V, E)$ be a chordal graph. Then, G is uniquely representable if and only if there is no proper containment between any minimal vertex separators and all minimal vertex separators are of multiplicity one.*

In the next theorem, we show that the ur-interval graphs are directly related to proper interval graphs. It was previously proved by Panda and Das [13]. Their proof can be significantly improved as shown below.

Theorem 8 *If G is a proper interval graph then G is an ur-interval graph.*

Proof: Let $\mathbb{S}$ be the set of minimal vertex separators of G. If G is an *ur*-interval graph then $\mu(S) = 1$ and $S \not\subset S'$, $S' \not\subset S$, for all $S, S' \in \mathbb{S}$. We will prove that if $\mu(S) \neq 1$ or $S \subset S'$, for some $S, S' \in \mathbb{S}$, then G is not a proper interval graph.

If $\mu(S) \neq 1$ then there are at least three maximal cliques Q, Q' and Q'' such that $Q \cap Q' = Q \cap Q'' = Q' \cap Q'' = S$. For $x \in Q \setminus (Q' \cup Q'')$, $y \in Q' \setminus (Q \cup Q'')$ and $z \in Q'' \setminus (Q \cup Q')$ there is a path from x to y, x to z and y to z that goes through at least a vertex of S, say w. The subgraph induced by vertices x, y, w and z is a claw and G is not a proper interval graph.

If $S \subset S'$ there are at least three cliques such that $S \subset Q$, $S' \not\subset Q$, $S' \subset Q'$ and $S' \subset Q''$. Obviously S is a subset of Q' and Q''. Hence, as in the previous case a claw can be identified and G cannot be a proper interval graph. ■

As a result of Theorem 8, the recognition algorithm can be simplified:

Step 1: build a clique-tree T of G;

if T is a path $\langle Q_1, \ldots, Q_q \rangle$ then go to Step 2, otherwise G is not a proper interval graph;

Step 2: build the greedy ordering δ;

Step 3: verify if δ obeys the neighborhood condition or the clique condition (Theorem 4).

Notice that the construction of the clique-tree [16] is much more simple than the interval graph recognition and directly provides the sequence of cliques.

In order to develop Step 3, we use the clique condition. For each maximal clique we determine the least and the greatest indices of its vertices in the greedy ordering. It is immediate that the difference of these indices must be the size of the clique. Algorithm *Test-clique-condition* implements this step.

The recognition of a proper interval graph has time complexity of $O(n+m)$. Step 1 can be performed by the linear algorithm presented in [16]. We already know (Section 2) that Step 2 is linear. Algorithm *Test-clique-condition* is very simple; the test performed on the maximal cliques has time complexity of $O(n + m)$.

Algorithm *Test-clique-condition*;
Input: G, the clique-path $\langle Q_1, \ldots, Q_q \rangle$,
a greedy ordering $\delta = \langle v_1, \ldots, v_n \rangle$;
Output: recognition of G as a proper interval graph;
begin
$proper \leftarrow true$; $j \leftarrow 1$;
while $j \leq q$ and $proper$ **do**
$first\text{-}index[j] \leftarrow \min\{i \mid v_i \in Q_j\}$;
$last\text{-}index[j] \leftarrow \max\{i \mid v_i \in Q_j\}$;
if $last\text{-}index[j] - first\text{-}index[j] \neq |Q_j| - 1$ **then**
$proper \leftarrow false$;
$p \leftarrow q$; $j \leftarrow q$;
$j \leftarrow j + 1$;
if $proper$ **then** YES (G is a proper interval graph) (*)
then NO (G is not a proper interval graph) (**)
end.

5 The certifying algorithm

A *certifying algorithm* is an algorithm that provides a certificate for each answer that it produces. A *certificate* is an evidence that the answer has not been compromised by a bug in the implementation [1].

In this section, we present a linear-time certifying algorithm for the recognition of proper interval graphs. We show that the certificate of non-membership presented here can be authenticated in $O(n + m)$ time.

The existing certifying algorithm for the recognition of proper interval graphs [11] is based on a multi-sweep min-LexBFS algorithm. Our approach takes advantage of the computation of Step 3, in the previous section. The certificate for the answer YES is immediate: the ordering δ. For the answer NO, we need to build a claw.

Let $G = (V, E)$ be an interval graph, $P = \langle Q_1, \ldots, Q_q \rangle$ a clique-path of G and $\delta = \langle v_1, \ldots, v_n \rangle$ a greedy ordering of G performed on P. Algorithm *Test-clique-condition* verifies if δ obeys the clique condition. If G is not a proper interval graph, there is a maximal clique such that its vertices are not consecutive in δ. Let Q_p be the first maximal clique in the clique-path in which this happens. Let v_i, v_k, $i < k$, be vertices of Q_p, with i the least and k the greatest of the indices in δ of vertices belonging to the clique. Let $v_j \notin Q_p$, $i < j < k$, be a vertex of δ.

Since G is connected, at least the first vertex v_i of Q_p belongs to Q_{p-1}. Vertex v_j belongs to some maximal clique before Q_p. Hence, it also belongs to Q_{p-1} because it appears in δ after v_i and the greedy ordering obeys the clique condition until Q_{p-1}.

Moreover, v_i also belongs to Q_{p-2}, because if not it would belong to the same bucket as v_j and, when considered in Q_p by the Algorithm *Greedy-ordering*, it would be moved to the end, appearing after v_j in δ. The following cases must be considered:

- v_j belongs only to Q_{p-1} (v_j is simplicial). In this case,

  $v_i \in (Q_p \cap Q_{p-1} \cap Q_{p-2})$;

  let $a \in Q_{p-2} \setminus Q_{p-1}$ and let $b \in Q_p \setminus Q_{p-1}$;

  claw: $V = \{v_i, v_j, a, b\}$; $E = \{\{v_i, a\}, \{v_i, b\}, \{v_i, v_j\}\}$;

- v_j is not simplicial. We know that the first clique to which v_j belongs is $Q_r, r = B(v_j)$. Vertex v_i belongs to Q_{r-1}.

  $v_i \in (Q_p \cap \ldots \cap Q_{r-1})$;

  $v_j \in (Q_{p-1} \cap \ldots \cap Q_r)$;

  let $a \in Q_{r-1} \setminus Q_r$ and let $b \in Q_p \setminus Q_{p-1}$;

  claw: $V = \{v_i, v_j, a, b\}$; $E = \{\{v_i, a\}, \{v_i, b\}, \{v_i, v_j\}\}$.

Hence, in order to provide a certificate for the new algorithm, the lines marked with (*) and (**) become:

> **if** *proper* **then** YES; certificate: δ
> **then** NO; certificate: build the claw with clique Q_p

The authentication is immediate. The ordering δ can be tested by any of the conditions presented. An alternative authentication could be to build the unit interval model, using the linear algorithm presented by Spinrad [16]. As to the claw, the analysis presented above can be performed with time complexity of $O(n + m)$.

References

[1] Alkassar, E., Böhme, S., Mehlhorn, K., Rizkallah, C., Schweitzer, P., *An introduction to certifying algorithms*, Information Technology **53**(6) (2011) 287-293.

[2] Blair, J.R.S., Peyton, B., An introduction to chordal graphs and clique trees. In: George, J.A., Gilbert, J. R., Liu, J. W. H. (Eds.), *Graph*

theory and sparse matrix computation, Springer Verlag, IMA 56, 1-30, 1993.

[3] Corneil, D.G., *A simple 3-sweep LBFS algorithm for the recognition of unit interval graphs*, Discrete Appl. Math. **138** (2004) 371-379.

[4] Deng, X., Hell, P., Huang, J., *Linear-time representation algorithms for proper circular-arc graphs and proper interval graphs*, SIAM J. Comput. **25** (1996) 390-403.

[5] Gilmore, P.C., Hoffman, A.J., *A characterization of comparability graphs and of interval graphs*, Canad. J. Math. **16** (1964) 539-548.

[6] Golumbic, M.C., Algorithmic graph theory and perfect graphs (2^{nd} edition), Academic Press, New York, 2004.

[7] Kumar, P.S., Madhavan, C.E.V., *Clique tree generalization and new subclasses of chordal graphs*, Discrete Appl. Math. **117** (2002) 109-131.

[8] Lekkerkerker, C.G., Boland, J., *Representation of a finite graph by a set of intervals on the real line*, Fund. Math. **51** (1962) 45-64.

[9] Looges, P.J., Olariu, S., *Optimal greedy algorithms for indifference graphs*, Comput. Math. Appl. **25** (1993) 15-25.

[10] Markenzon, L., Pereira, P.R.C., *One-phase algorithm for the determination of minimal vertex separators of chordal graphs*, Int. Trans. Oper. Res. **17** (2010) 683-690.

[11] Meister, D., *Recognition and computation of minimal triangulations for AT-free claw-free and co-comparability graphs*, Discrete Appl. Math. **146** (2005) 193-218.

[12] Olariu, S., An optimal greedy heuristic to color interval graphs, Inform. Process. Lett. **37** (1991) 21-25.

[13] Panda, B.S., Das, S.K., *A parallel algorithm for generating bicompatible elimination orderings of proper interval graphs*, Inform. Process. Lett. **109** (2009) 1041-1046.

[14] Roberts, F.S., Indifference graphs, F. Harary (Ed.), Proof Techniques in Graph Theory, Academic Press, New York, NY, 139-146, 1969.

[15] Roberts, F.S., *On the compatibility between a graph and a simple order*, J. Comb. Theory, **11** (1971) 28-38.

[16] Spinrad, J.P., Efficient graph representations. Fields Institute Monographs 19, AMS, Providence, 2003.